Grant Agreement nº	732463
Project Acronym:	OpenReq
Project Title:	Intelligent Recommendation Decision Technologies for Community-Driven Requirements Engineering
Call identifier:	H2020-ICT-2016-1
Instrument:	RIA (Research and Innovation Action
Topic	ICT-10-16 Software Technologies
Start date of project	January 1st, 2017
Duration	36 months

D6.2 First OpenReq integrated version and APIs

Lead contractor:	ENG
Author(s):	ENG, HITEC, TUG, UH, UPC, VOGELLA
Submission date:	August 2018
Dissemination level:	PU

Project co-funded by the European Commission under the H2020 Programme.

Abstract: This document describes the infrastructure where the microservices provided by the components developed within WP2 – WP5 are integrated and deployed, and presents the first OpenReq integration scenarios.

This document has been produced in the context of the OpenReq Project. The OpenReq project is part of the European Community's h2020 Programme and is as such funded by the European Commission. All information in this document is provided "as is" and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability. For the avoidance of all doubts, the European Commission has no liability is respect of this document, which is merely representing the authors view.

Table of Contents

List of Figures

List of Tables

1 INTRODUCTION

This section presents where and how to access the resources for the 1st OpenReq integrated version, and describes how this document is organized.

1.1 Resources

The source code of the services described in this document is available on the Tuleap Git repository at the following address (password protected):
https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/?group_id=101

1.2 Document layout

This document describes the infrastructure where the components deployed within WP2 – WP5 are integrated, tested and deployed and presents the first OpenReq integration scenarios.

Section 2 describes the ENG infrastructure and the process followed during deployment, from the code uploading in the repository to the service start on the server.

Section 3 gives an overview of the 1st OpenReq version, listing all the components developed within the various WPs and the services they offer.

Section 4 to 7 describe in detail the microservices currently deployed on the ENG infrastructure, reporting the APIs needed to invoke them. In particular, Section 4 is related to the Requirements Intelligence Engine microservices (WP2), Section 5 covers microservices provided by the Personal Recommendations for Stakeholder's Engine (WP3), Section 6 considers the Group Decision Engine microservices (WP4) and Section 7 describes microservices offered by the Dependency Engine (WP5). Finally, Section 8 presents some integration scenarios of the API.

2 Project Infrastructure Overview

This section describes the FiWare infrastructure used by ENG to host the OpenReq integrated version. Information about the deployment process of the OpenReq components on the infrastructure are also presented.

2.1 Deployment infrastructure

The deployment infrastructure, made available by ENG and hosted by FiWare, includes two virtual machines:

- openreq-testbed, a virtual machine (VM) with a public IP address (217.172.12.199) where OpenReq services are deployed. One secure shell (SSH) access to this VM has been granted to each partner. This VM has 32 GB RAM and 80 GB disk space.
  - Installed software:
    - Java, Python and Maven, i.e. tools required to compile the code uploaded in Git
    - Docker
    - Docker Compose
    - Jenkins, installed as Docker container and configured as slave node
    - Ngix web server
    - Portainer.io, installed as Docker container, an UI allowing to manage Docker hosts
    - Mongo DB, installed as Docker container
    - My SQL
- jenkins-master, a VM serving as Jenkins master, with 4 GB RAM and 20 GB disk space
  - Installed software:
  - Docker
  - Jenkins, installed as Docker container

2.2 Deployment process

This section outlines the deployment process of OpenReq services.
The procedure followed to automate the build and deployment of a new service developed in OpenReq is the following:

1. **Code uploading in the Git repository**. As a first step, the source code of the components has to be pushed to the Tuleap git repository in one of the following branches:

    - `releases` or
    - `master` (in case of a stable component).

    Also details on the endpoints of the microservice, useful to test it once deployed in the server, should be provided, e.g. API documentation in form of a Swagger file.

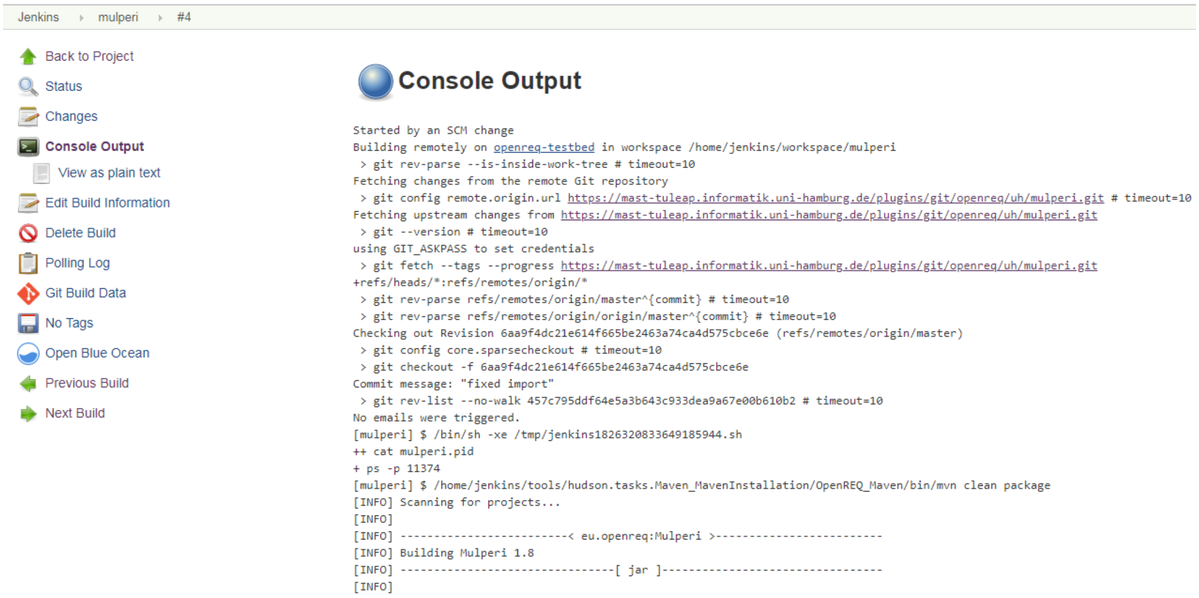
2. **Jenkins job request**. Once the repository contains everything needed to build, deploy and test the component, an email should be sent to ENG adding details on:

    - The port where the MS will be deployed (chosen in the range assigned to the specific partner)
    - The base path
    - The users that will be granted full control of the job
    - The person(s) that will receive notifications on build results by email

3. **Jenkins Job setting and configuration**. After having received the deployment request by the MS developer, ENG setups and configures the Jenkins job to compile the component code, generate the artefacts (e.g. jars or Docker images) and execute them on the server.
   The job is configured to enable automated build. The service will be built and re-deployed automatically every time a change in the aforementioned branches of the git repository takes place. The first build is triggered manually by ENG.

If the build is successful, it generates the jar or Docker images and start the server on the port specified at the beginning of the deployment process.

The main problems encountered during this process were due to the difference between the deployment infrastructure and the one used during development. To overcome issues happening during building, developers were given full control of the Jenkins jobs for their components. In this way, each partner can view building logs (as shown in Figure 1) and fix job configuration errors.

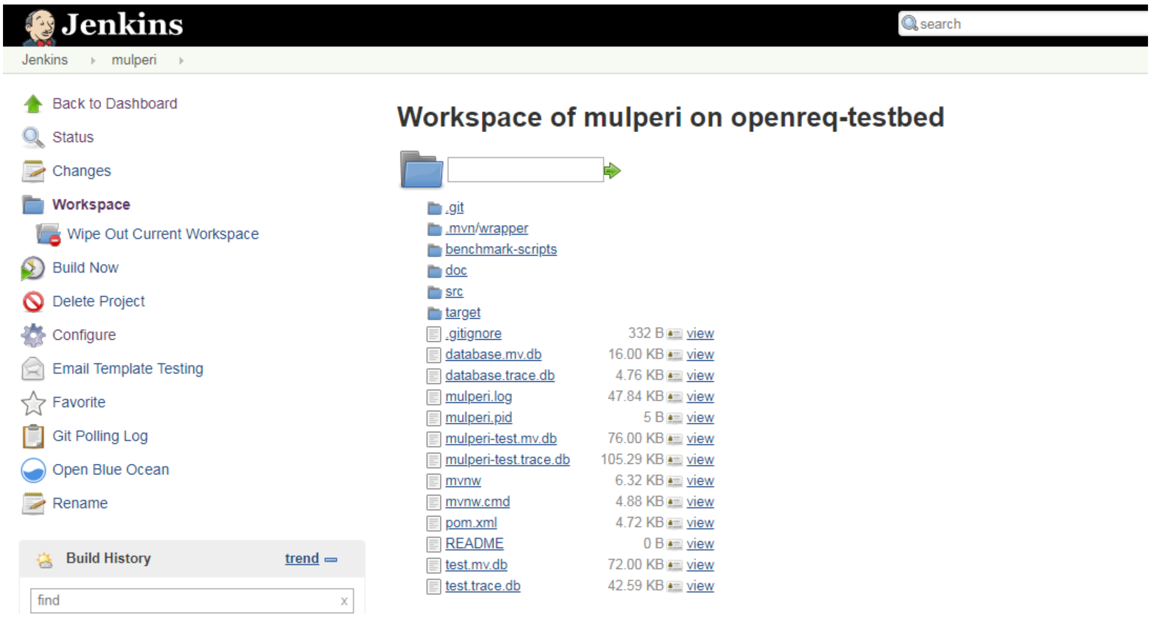


```
Started by an SCM change
Building remotely on openreq-testbed in workspace /home/jenkins/workspace/mulperi
 > git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/uh/mulperi.git # timeout=10
Fetching upstream changes from https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/uh/mulperi.git
 > git --version # timeout=10
using GIT_ASKPASS to set credentials
 > git fetch --tags --progress https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/uh/mulperi.git
+refs/heads/*:refs/remotes/origin/*
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
 > git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision 6aa9f4dc21e614f665be2463a74ca4d575cbce6e (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 6aa9f4dc21e614f665be2463a74ca4d575cbce6e
Commit message: "fixed import"
 > git rev-list --no-walk 457c795ddf64e5a3b643c933dea9a67e00b610b2 # timeout=10
No emails were triggered.
[mulperi] $ /bin/sh -xe /tmp/jenkins1826320833649185944.sh
++ cat mulperi.pid
+ ps -p 11374
[mulperi] $ /home/jenkins/tools/hudson.tasks.Maven_MavenInstallation/OpenREQ_Maven/bin/mvn clean package
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------< eu.openreq:Mulperi >------------------------
[INFO] Building Mulperi 1.8
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
```

Figure 1: Jenkins console output showing the (first part of) building log of component *mulperi*.

Problems occurred at runtime could be investigated (and solved), by examining:

- The deployed components workspace, including logs, through Jenkins, as shown in Figure 2.

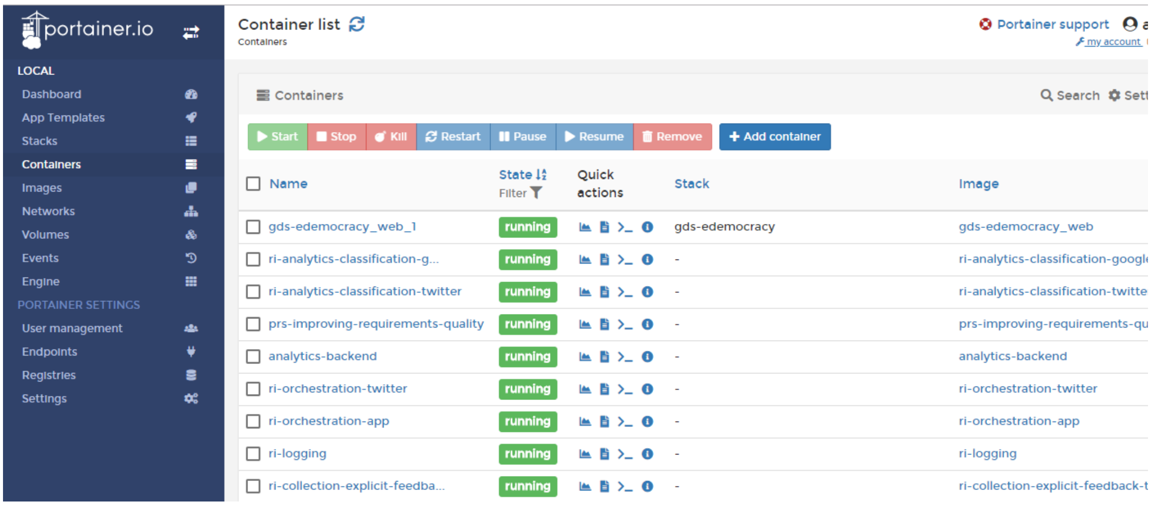


Workspace of mulperi on openreq-testbed

.git		
.mvn/wrapper		
benchmark-scripts		
doc		
src		
target		
.gitignore	332 B	view
database.mv.db	16.00 KB	view
database.trace.db	4.76 KB	view
mulperi.log	47.84 KB	view
mulperi.pid	5 B	view
mulperi-test.mv.db	76.00 KB	view
mulperi-test.trace.db	105.29 KB	view
mvnw	6.32 KB	view
mvnw.cmd	4.88 KB	view
pom.xml	4.72 KB	view
README	0 B	view
test.mv.db	72.00 KB	view
test.trace.db	42.59 KB	view

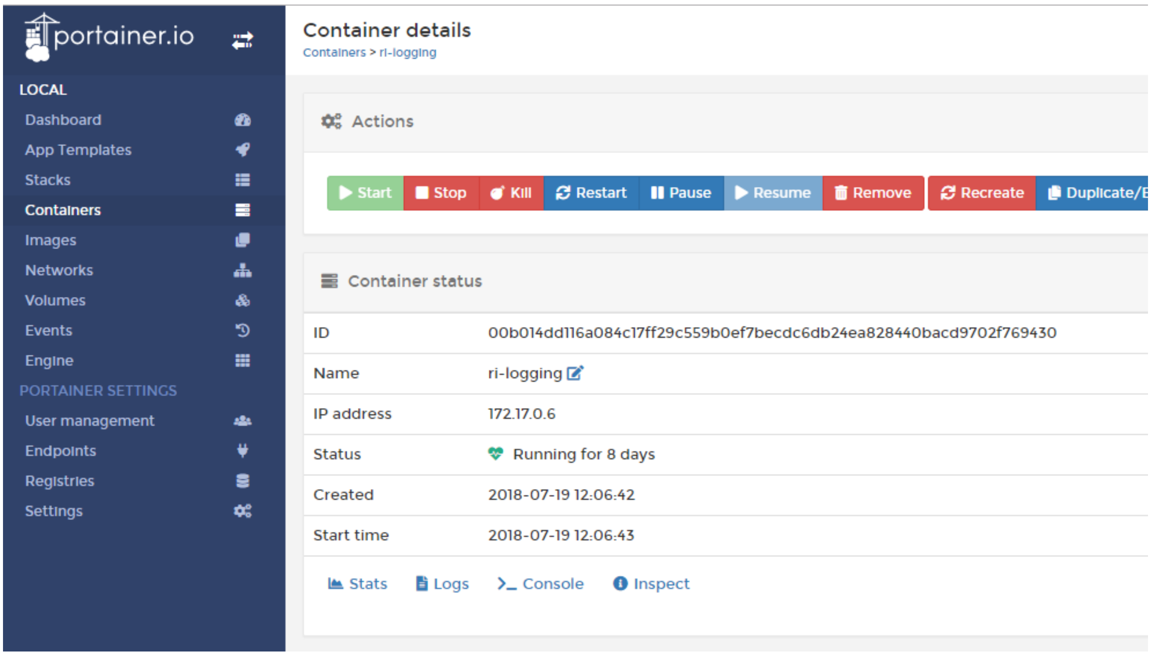
Figure 2: Workspace of component *mulperi*

- Docker images and containers (Figure 3), presented in Portainer.io. A page showing the details of each container, including logs, is also available, as illustrated in Figure 4.



□ Name	State Filter	Quick actions	Stack	Image
□ gds-edemocracy_web_1	running		gds-edemocracy	gds-edemocracy_web
□ ri-analytics-classification-g...	running		-	ri-analytics-classification-googl
□ ri-analytics-classification-twitter	running		-	ri-analytics-classification-twitte
□ prs-improving-requirements-quality	running		-	prs-improving-requirements-qu
□ analytics-backend	running		-	analytics-backend
□ ri-orchestration-twitter	running		-	ri-orchestration-twitter
□ ri-orchestration-app	running		-	ri-orchestration-app
□ ri-logging	running		-	ri-logging
□ ri-collection-explicit-feedba...	running		-	ri-collection-explicit-feedback-t

Figure 3: Docker containers started in the server.



ID	00b014dd116a084c17ff29c559b0ef7becdc6db24ea828440bacd9702f769430
Name	ri-logging
IP address	172.17.0.6
Status	Running for 8 days
Created	2018-07-19 12:06:42
Start time	2018-07-19 12:06:43

Figure 4: Upper part of the Portainer page showing details of container *ri-logging*.

3 First OpenReq Version Overview

In this section we briefly summarize the architecture of the system and present a comprehensive list of the deployed microservices.

3.1 Overall View

The OpenReq architecture, shown in Figure 5, is composed of different parts: the *OpenReq REST Services*, the *trial applications* that use the services [1], and the services *databases*. Data between the services is exchanged using a JSON schema based on the OpenReq Ontology. Besides the trial partner applications, the OpenReq Live (i.e., a web interface) accesses the services to showcase the major OpenReq functionalities

The OpenReq services are offered through a set of microservices. These microservices are developed independently to lower the coupling between them, but the services also exploit/integrate the services of other components. The decision to utilize a service-oriented architecture rests largely in the ease-of-development between partners, but also includes the final integration of components. The integration infrastructure involves automatic builds orchestrated by Jenkins[1], pulling development changes from Git. Once built, the services are deployed and monitored by Portainer[2]. Jenkins makes the builds for Docker[3] and Maven[4] simple and automated, whilst Portainer runs the builds and monitors them so each partner can track and maintain their services.

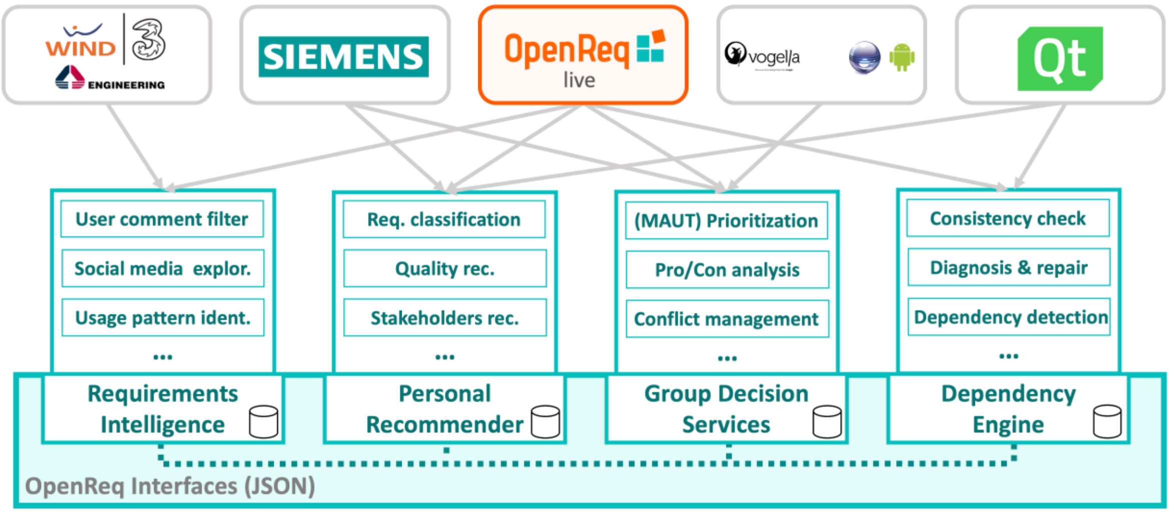


Figure 5: OpenReq overall architecture [1]

There are four engines (components) providing services:

- The **Requirements Intelligence Engine**, developed in WP2;
- The **Personal Recommendations for Stakeholder’s Engine**, investigated within WP3;

[1] https://jenkins.io/
[2] https://portainer.io/
[3] https://www.docker.com/
[4] https://maven.apache.org/

- The **Group Decision Engine**, examined by WP4;
- The **Dependency Engine**, provided by WP5.

The microservices provided by the above components are listed in the following section and described in detail in the rest of the document.

3.2 Deployed microservices

The following list reports all the microservices currently deployed in the OpenReq ENG environment. In parenthesis, we report the main partner responsible for the service.

- **Requirements Intelligence Engine** MSs:
  - analytics-backend (ENG)
  - ri-analytics-classification-google-play-review (HITEC)
  - ri-analytics-classification-twitter (HITEC)
  - ri-analytics-rationale-miner (HITEC)
  - ri-collection-explicit-feedback-google-play-page (HITEC)
  - ri-collection-explicit-feedback-google-play-review (HITEC)
  - ri-collection-explicit-feedback-twitter (HITEC)
  - ri-orchestration-app (HITEC)
  - ri-orchestration-twitter (HITEC)
  - ri-storage-app (HITEC)
  - ri-storage-twitter (HITEC)
  - ri-logging (HITEC)
  - update-manager (ENG)
  - twitter-extraction (TUGRAZ)
- **Personal Recommendations for Stakeholder's Engine** MSs:
  - prs-improving-requirements-quality (HITEC)
  - uploader (ENG)
  - uploader (ENG)
  - entrypoint (ENG)
  - parsing-conversion-doc (ENG)
  - parsing-segmentation (ENG)
  - parsing-to_dict (ENG)
  - parsing-enrich (ENG)
  - parsing-enrich-prettify (ENG)
  - lemmatizer-<string:extr_type> (ENG)
  - keywords-supervised-<string:extr_type>
  - parsing-types (ENG)
  - parsing-features (ENG)
  - parsing-structure (ENG)
  - parsing-metrics (ENG)
  - requirements-classifier (UPC)
  - issue-prioritizer (VOGELLA)
  - issue-prioritizer-api (TUGRAZ)

- **Group Decision Engine** MSs:
    - Openreq-live (TUGraz)
    - gds-edemocracy (HITEC)
- **Dependency Engine** MSs:
    - mulperi (UH)
    - springcaas (UH)
    - cross-reference-detection (UPC)
    - similarity-detection (UPC)
    - dependency-detection (UPC)

There are other services currently deployed in the server that are peculiar to Qt trial and will therefore not further described in this document, but are still reported below for completeness:

- milla (UH)
- mallikas (UH)
- qthulhu (QT)

4 REQUIREMENTS INTELLIGENCE ENGINE

This section presents and overview of the Requirements Intelligence Engine as well as a description of the available APIs.

4.1 Overview

The Requirements Intelligence Engine, developed in WP2, allows the analysis of natural language in text-based documents or user feedback (either explicit or implicit). In addition to that, also interactive visualization is supported. The architecture of this component and the provided services are described in detail in deliverable [2].

The operational details of the microservices currently available in the OpenReq infrastructure are reported in the following section.

4.2 Microservices and APIs

This section reports the specifics of the 14 microservices composing the Requirements Intelligence Engine component.

4.2.1 analytics-backend

4.2.1.1 Overview

Name	analytics-backend
Developer	ENG
Dependencies	None
Base path	/openReq/
Port	10601
Technology	Docker, Python & Flask

Table 1. Requirements Intelligence Engine – MS *analytics backend*

This microservice performs topic extraction of the tweets addressed to a specified Twitter account. The microservice identifies the topics of major interest and what a large amount of tweets talk about. This gives the possibility to pinpoint inconveniences, system failures, dissatisfaction, or customer’s necessities.

4.2.1.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/analytics-backend.git

4.2.1.3 API Documentation

The following tables describe the available HTTP requests. The API documentation, in form of a Swagger file, is available at http://217.172.12.199:10601/openReq/apidocs/

cleanText

URL	http:// 217.172.12.199:10601/openReq/cleanText
METHOD	POST
URL parameters	None
Request example	[{"message":"Posso essere chiamato da un operatore 3"}, {"message":"Ci sono appena andato a un negozio e o faccio una promozione o niente! Voglio pagare quando chiamo e non vo https: / /t.co /lhRqzJPfpT"}]
Response example	["posso chiamato operatore", "appena andato negozio promozione voglio pagare chiamo"]

computeTopics

URL	http:// 217.172.12.199:10601/openReq/computeTopics
METHOD	POST
URL parameters	None

Request example	{
	"w2v_model_id":1,
	"som_model_id":1,
	"codebook_cluster_model_id":1,
	"tweets":[
	{"message":"Posso essere chiamato da un operatore 3"},
	{"message":"Ci sono appena andato a un negozio e o faccio una promozione o niente! Voglio pagare quando chiamo e non vo "
	}
	]
	}

Response example	[
	{
	"directed": false,
	"graph": {},
	"links": [{
	"source": 0,
	"target": 1
	},{
	"source": 0,
	"target": 4
	},{
	"source": 0,
	"target": 7
	}],
	"multigraph": false,
	"nodes": [{
	"id": 0,
	"name": "offerta",
	"pos": [
	0.39845688850044275,
	0.01701041209436488]
	},{
	"id": 1,
	"name": "promozione",
	"pos": [
	0.5467295096202749,
	-0.18447308286329653]
	},{

```
        "id": 4,
        "name": "numero",
        "pos": [
          0.7412007341069454,
          0.06619163623027753]
      },{
        "id": 7,
        "name": "abbonamento",
        "pos": [
          -0.007811836655060454,
          0.12406653407660848]
      }
    ]
  },
  {
    "directed": false,
    "graph": {},
    "links": [{
        "source": 0,
        "target": 1
      },{
        "source": 0,
        "target": 2
      },{
        "source": 0,
        "target": 11
    }],
    "multigraph": false,
    "nodes": [{
        "id": 0,
        "name": "wind",
        "pos": [
          0.1025021960790923,
          -0.005834864831025231
      },{
        "id": 1,
        "name": "operatore",
        "pos": [
          -0.2125952797941764,
          0.19447134425055665]
      },{
        "id": 2,
        "name": "telefono",
        "pos": [
          -0.07288693121191979,
          -0.23931113221735195]
      },
      {
        "id": 11,
        "name": "posso",
        "pos": [
          0.4806237910010984,
          -0.02734068831978047]
      }
    ]
  }
]
```

doSomAndPlot

URL	http:// 217.172.12.199:10601/openReq/doSomAndPlot
METHOD	POST
URL parameters	None
Request example	{

	"type_chart": "json", "w2v_model_id":1, "som_model_id":1, "tweets":[{"message":"Posso essere chiamato da un operatore 3"}, {"message":"Ci sono appena andato a un negozio e o faccio una promozione o niente! Voglio pagare quando chiamo e non vo "}] }

Response example	{
	"directed": false,
	"graph": {},
	"links": [{
	"source": 0,
	"target": 3
	},{
	"source": 0,
	"target": 4
	},{
	"source": 1,
	"target": 2
	},{
	"source": 2,
	"target": 3
	}],
	"multigraph": false,
	"nodes": [{
	"id": 0,
	"name": "chiamo",
	"pos": [
	-0.11059179996954856,
	0.5377064474591033]
	},{
	"id": 1,
	"name": "pagare",
	"pos": [
	0.18664644639572908,
	-1]
	},{
	"id": 2,
	"name": "promozione",
	"pos": [
	0.1805092121315994,
	-0.5184325667170537]
	},{
	"id": 3,
	"name": "posso",
	"pos": [
	0.09470297420983105,
	0.025920574142384148]
	},{
	"id": 4,
	"name": "chiamato",
	"pos": [
	-0.3512668327676108,
	0.9548055451155663]
	}
	]
	}

getCodebookActivation

URL	http:// 217.172.12.199:10601/openReq/getCodebookActivation
METHOD	GET
URL parameters	som_model_id

Returned Data	image/png

getCostOfSom

URL	http:// 217.172.12.199:10601/openReq/getCostOfSom
METHOD	GET
URL parameters	som_model_id

Response example	{
	"cost of model": "38.86 %"
	}

getEmbeddedWords

URL	http:// 217.172.12.199:10601/openReq/getEmbeddedWords
METHOD	POST
URL parameters	None

Request example	{
	"w2v_model_id":1,
	"tweets":[
	{"message":"Posso essere chiamato da un operatore 3"},
	{"message":"Ci sono appena andato a un negozio e o faccio una promozione o niente! Voglio pagare quando chiamo e non vo "}
	]
	}

Response example	[[0.03953748196363449, -0.33750391006469727, ..., 0.17522814869880676,], …, [-1.6901140213012695, -0.5863699913024902, …, 1.2313200235366821]]

getUmatrix

URL	http:// 217.172.12.199:10601/openReq/getUmatrix
METHOD	GET
URL parameters	som_model_id
Returned Data	image/png

textRanking

URL	http:// 217.172.12.199:10601/openReq/textRanking
METHOD	POST
URL parameters	None

Request example	{
	"tweets":[
	{"message":"Posso essere chiamato da un operatore 3"},
	{"message":"Ci sono appena andato a un negozio e o faccio una promozione o niente! Voglio pagare quando chiamo e non vo "}
	],
	"w2v_model_id":1,
	“bigram_model_id”,: 1
	}

Response example	[["appena", "pagare", "voglio", "negozio", "andato"],["posso", "operatore"]]

keywordsExtraction

URL	http:// 217.172.12.199:10601/openReq/keywordsExtraction
METHOD	POST
URL parameters	None
Request example	{ "tweets":[{"message":"Posso essere chiamato da un operatore 3"},\ {"message":"Ci sono appena andato a un negozio e o faccio una promozione o niente! Voglio pagare quando chiamo e non vo "}], "w2v_model_id":1, "bigram_model_id":1}
Response example	[["pagare", "voglio"],["negozio", "andato", "appena"]]

trainSom

URL	http:// 217.172.12.199:10601/openReq/trainSom
METHOD	POST
URL parameters	None

Request example	{
	"w2v_model_id":1
	}

Response example	{
	"som_model_id": 1531217093
	}

trainNgram

URL	http:// 217.172.12.199:10601/openReq/trainNgram
METHOD	POST
URL parameters	None

Request example	{
	"tweets":[
	{"message":"Posso essere chiamato da un operatore 3"},
	{"message":"Ci sono appena andato a un negozio e o faccio una promozione o niente! Voglio pagare quando chiamo e non vo "}
	]
	}

Response example	{
	"bigram_model_id": 1531217258
	}

trainCodebookClustering

URL	http:// 217.172.12.199:10601/openReq/trainCodebookClustering
METHOD	POST
URL parameters	None

Request example	{
	"som_model_id":1
	}

Response example	{
	"codebook_cluster_model_id": 1531218052
	}

trainWord2vec

URL	http:// 217.172.12.199:10601/openReq/trainWord2vec
METHOD	POST

URL parameters	None

Request example	{
	"tweets":[
	{"message":"Posso essere chiamato da un operatore 3"},
	{"message":"Ci sono appena andato a un negozio e o faccio una promozione o niente! Voglio pagare quando chiamo e non vo "}
	]
	}

Response example	{
	"w2v_model_id": 1531218052
	}

4.2.1.4 Internal storage

None

4.2.2 ri-analytics-classification-google-play-review

4.2.2.1 Overview

Name	ri-analytics-classification-google-play-review
Developer	HITeC
Dependencies	None
Base path	/hitec/classify/domain/google-play-reviews/
Port	9651
Technology	Docker

Table 2. Requirements Intelligence Engine – MS *ri-analytics-classification-google-play-review*

The goal of this microservice is to classify a list of app reviews as either a “bug report” or a “feature request.” The source code necessary for these tasks is bundled in a Docker container. Each task is composed of sub-tasks. Such sub-tasks are data cleaning, machine learning feature extraction, and classification on pre-trained models. The response of the microservice is a list of app reviews that now include the class they belong to.

4.2.2.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/ri-analytics-classification-google-play-review

4.2.2.3 API Documentation

In the following table the available HTTP request is described in detail.

URL	http://217.172.12.199:9651/hitec/classify/domain/google-play-reviews/
METHOD	POST
URL parameters	None

Request example	[{ "review_id": "05df34353efd", "package_name": "co.myapp", "author": "mustermann", "date_posted": 20180524, "rating": 5, "title": "I like this app because…", "body": "it has really nice features", "perma_link": "https: //…." }]
Response example	[{ "review_id": "05df34353efd", "package_name": "co.myapp", "author": "mustermann", "date_posted": 20180524, "rating": 5, "title": "I like this app because…", "body": "it has really nice features", "perma_link": "https: //….", "cluster_is_feature_request": false, "cluster_is_problem_report": true }]

4.2.2.4 Internal storage

None

4.2.3 ri-analytics-classification-twitter

4.2.3.1 Overview

Name	ri-analytics-classification-twitter
Developer	HITeC
Dependencies	None
Base path	/hitec/classify/domain/tweets/
Port	9655
Technology	Docker

Table 3. Requirements Intelligence Engine – MS *ri-analytics-classification-twitter*

The goal of this microservice is to take a list of tweets, extract their natural language features and classify them either as a problem report, inquiry, or irrelevant.

4.2.3.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/ri-analytics-classification-twitter

4.2.3.3 API Documentation

In the following table the available HTTP request is described in detail.

URL	http://217.172.12.199:9655/hitec/classify/domain/tweets/lang/it
METHOD	POST
URL parameters	None
Request example	[{ "created_at": 20180501, "favorite_count": 0, "retweet_count": 0, "text" : "The football match was amazing!", "status_id": "154756323465724654", "user_name": "Mustermann", "in_reply_to_screen_name": "Fifa", "lang": "it" }]
Response example	[{ "created_at" : 20180501, "favorite_count" : 0, "retweet_count" : 0, "text" : "The football match was amazing!", "status_id" : "154756323465724654", "user_name" : "Mustermann", "in_reply_to_screen_name" : "Fifa", "lang" : "it", “tweet_class” : “problem_report }]

4.2.3.4 Internal storage

None

4.2.4 ri-analytics-rationale-miner

4.2.4.1 Overview

Name	ri-analytics-rationale-miner
Developer	HITeC
Dependencies	None
Base path	/hitec/urminer-review/
Port	9704
Technology	Docker

Table 4. Requirements Intelligence Engine – MS *ri-analytics-rationale-miner*

When users write reviews they often also mention their rationale of why, e.g., they chose an alternative to the software under review or why a certain decision is taken. This microservice extracts the probability of a user rationale existing in an Amazon review. The microservice supports the user rationale categories decision, alternative, justification, and criteria being part of an Amazon review.

4.2.4.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/ri-analytics-rationale-miner

4.2.4.3 API Documentation

In the following table the available HTTP request is described in detail.

getJustificationProbabilities

URL	http://217.172.12.199:9704/hitec/urminer-review/justification
METHOD	POST

Request example	{
	"Body": "I bought this product because I am a long time committed Quicken user, and I was not willing to switch to the Mac version.",
	"Title": "Mac Version ",
	"Rating": 3.0
	}

Response example	{
	"data": {
	"True": 0.39,
	"False": 0.39,
	}
	}

getDecisionProbabilities

URL	http://217.172.12.199:9704/hitec/urminer-review/decision
METHOD	POST

Request example	{
	"Body": "I bought this product because I am a long time committed Quicken user, and I was not willing to switch to the Mac version.",
	"Title": "Mac Version ",
	"Rating": 3.0
	}

Response example	{
	"data": {
	"True": 0.39,
	"False": 0.39,
	}
	}

getCriteraProbabilities

URL	http://217.172.12.199:9704/hitec/urminer-review/criteria
METHOD	POST

Request example	{
	"Body": "I bought this product because I am a long time committed Quicken user, and I was not willing to switch to the Mac version.",
	"Title": "Mac Version ",
	"Rating": 3.0
	}

Response example	{
	"data": {
	"True": 0.39,
	"False": 0.39,
	}

	}

getAlternativeprobabilities

URL	http://217.172.12.199:9704/hitec/urminer-review/alternative
METHOD	POST

Request example	{
	"Body": "I bought this product because I am a long time committed Quicken user, and I was not willing to switch to the Mac version.",
	"Title": "Mac Version ",
	"Rating": 3.0
	}

Response example	{
	"data": {
	"True": 0.39,
	"False": 0.39,
	}
	}

4.2.4.4 Internal storage

None

4.2.5 ri-collection-explicit-feedback-google-play-page

4.2.5.1 Overview

Name	ri-collection-explicit-feedback-google-play-page
Developer	HITeC
Dependencies	None
Base path	/hitec/crawl/app-page/google-play/
Port	9622
Technology	Docker

Table 5. Requirements Intelligence Engine – MS *ri-collection-explicit-feedback-google-play-page*

The goal of this microservice is to collect data from the Google Play Store—the official store for Android apps. In particular, this service collects the data available on the page representing the app. The response contains information of the app page in JSON format.

4.2.5.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/ri-collection-explicit-feedback-google-play-page

4.2.5.3 API Documentation

In the following table the available HTTP request is comprehensively described.

URL	http://217.172.12.199:9622/hitec/crawl/app-page/google-play/{package_name}
METHOD	GET
URL parameters	package_name: the package name of the app page to crawl

Response example	{
	    "name": "WhatsApp Messenger",
	    "package_name": "com.whatsapp",
	    "date_crawled": "2017-11-22T14:07:29.866407+01:00",
	    "category": "Communication",
	    "usk": "USK: All ages",
	    "price": "free",
	    "price_value": 0,
	    "price_currency": "€",
	    "description": "This is a great app",
	    "whats_new": "fixed several bugs",
	    "rating": 4.4,
	    "star_count": 61050950,
	    "count_per_rating": {
	        "1": 0,
	        "2": 0,
	        "3": 0,
	        "4": 0,
	        "5": 0
	    },
	    "developer": "WhatsApp Inc.",
	    "top_developer": false,
	    "contains_ads": false ,
	    "in_app_purchase": false ,
	    "last_update": 20171027,
	    "os": "ANDROID",
	    "requires_os_version": "Varies+",
	    "current_software_version": "Varieswithdevice",
	    "similar_apps": ["com.xyz", "com.qwerty"]
	}

4.2.5.4 Internal storage

None

4.2.6 ri-collection-explicit-feedback-google-play-review

4.2.6.1 Overview

Name	ri-collection-explicit-feedback-google-play-review
Developer	HITeC
Dependencies	None
Base path	/hitec/crawl/app-reviews/google-play/
Port	9621
Technology	Docker

Table 6. Requirements Intelligence Engine – MS *ri-collection-explicit-feedback-google-play-review*

The goal of this microservice is to collect data from the Google Play Store—the official store for Android apps. In particular, this service collects the user reviews of a given app. The response contains a list of reviews belonging to a certain app in JSON format.

4.2.6.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/ri-collection-explicit-feedback-google-play-review

4.2.6.3 API Documentation

In the following table the available HTTP request is described in detail.

URL	http://217.172.12.199:9621/hitec/crawl/app-reviews/google-play/{package_name}/limit/{limit}
METHOD	GET
URL parameters	package_name: the name of the App of which we crawl the reviews, limit: the maximum number of reviews to retrieve
Response example	[{ "review_id": "05df34353efd", "package_name": "co.myapp", "author": "mustermann", "date_posted": 20180524, "rating": 5, "title": "I like this app because…", "body": "it has really nice features", "perma_link": "https: //…." }]

4.2.6.4 Internal storage

None

4.2.7 ri-collection-explicit-feedback-twitter

4.2.7.1 Overview

Name	ri-collection-explicit-feedback-twitter
Developer	HITeC
Dependencies	None
Base path	/hitec/crawl/tweets/
Port	9624
Technology	Docker

Table 7. Requirements Intelligence Engine – MS *ri-collection-explicit-feedback-twitter*

The goal of this microservice is to collect data from Twitter. In particular, this service collects tweets that mention a given account. The response contains a list of tweets in a JSON format.

4.2.7.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/ri-collection-explicit-feedback-twitter

4.2.7.3 API Documentation

In the following tables the available HTTP requests are described in detail.

Get a list of all recent Tweets[5]

URL	http://217.172.12.199:9624/hitec/crawl/tweets/mention/{account_name}/lang/{lang}
METHOD	GET
URL parameters	account_name: the name of the Twitter profile to crawl, lang: the language the tweets should be written in
Response example	[{ "created_at" : 20180501, "favorite_count" : 0, "retweet_count" : 0, "text" : "The football match was amazing!", "status_id" : "154756323465724654", "user_name" : "Mustermann", "in_reply_to_screen_name" : "Fifa", "lang" : "en" }]

Get a list of all recent Tweets until the API limit is reached[6]

URL	http://217.172.12.199:9624/hitec/crawl/tweets/mention/{account_name}/lang/{lang}/fast
METHOD	GET
URL parameters	account_name: the name of the Twitter profile to crawl, `days`: the days we want to crawl, counting backwards from current date, lang: the language the tweet was officially labelled with
Response example	{["created_at": 20180723, "favorite_count": 0, "retweet_count": 0, "text": "@WindItalia Fatto! Grazie !", "status_id": "1021503454513188866", "user_name": "PrevitaliAlex", "in_reply_to_screen_name": "WindItalia", "lang": "it", "tweet_class": null },{ "created_at": 20180723, "favorite_count": 0, "retweet_count": 0, "text": "@WindItalia ciao entro quanto avviene la portabilità della linea fissa in Wind?", "status_id": "1021501529810001920",

[5] Performs pagination and therefore takes a long time (>30min)

[6] Usually takes a few seconds

	"user_name": "PrevitaliAlex", "in_reply_to_screen_name": "WindItalia", "lang": "it", "tweet_class": null },{ "created_at": 20180723, "favorite_count": 0, "retweet_count": 0, "text": "@WindItalia problemi di linea a morbegno (so)? La mia linea non va da sta mattina. Cosa succede? Grazie", "status_id": "1021494333948813312", "user_name": "ariannadelpizzo", "in_reply_to_screen_name": "WindItalia", "lang": "it", "tweet_class": null }, "in_reply_to_screen_name" : "Fifa", "lang" : "en" }]

Get a list of Tweets in a given language that mentions the specified account

URL	http://217.172.12.199:9624/hitec/crawl/tweets/mention/ {account_name}/from/{date}/lang/{lang}
METHOD	GET
URL parameters	account_name: the name of the Twitter profile to crawl, date: specify the date from which the crawler starts, lang: the language the tweets should be written in
Response example	[{ "created_at" : 20180501, "favorite_count" : 0, "retweet_count" : 0, "text" : "The football match was amazing!", "status_id" : "154756323465724654", "user_name" : "Mustermann", "in_reply_to_screen_name" : "Fifa", "lang" : "en" }]

Get a list of Tweets that mention a specific account for a defined time-frame

URL	http://217.172.12.199:9624/hitec/crawl/tweets/mention/{account_name}/history-in-days/{days}/lang/{lang}
METHOD	GET
URL parameters	account_name: the name of the Twitter profile to crawl, days: the days we want to crawl, counting backwards from current date, lang: the language the tweets should be written in
Response example	[{ "created_at" : 20180501, "favorite_count" : 0, "retweet_count" : 0, "text" : "The football match was amazing!",

	"status_id" : "154756323465724654", "user_name" : "Mustermann", "in_reply_to_screen_name" : "Fifa", "lang" : "en" }]

4.2.7.4 Internal storage

None

4.2.8 ri-logging

4.2.8.1 Overview

Name	ri-logging
Developer	HITeC
Dependencies	None
Base path	/
Port	9798
Technology	Docker

Table 8. Requirements Intelligence Engine – MS *ri-logging*

This microservice collects implicit feedback from user interactions with OpenReq UIs and user/microservice interactions with the backend.
The UI interactions are captured by a Javascript library that only needs to be imported by the UI. No integration inside the UIs code (apart from `<script src=“<url>“></script>`) is needed. The events in the UI are sent to this logger microservices API and saved in a text file or database. The logs are accessible through this microservices API.
The backend interaction is captured by a component (not part of the microservice) of Ngix. Every request and response that reaches the backend (i.e., all OpenReq microservices) is logged in a file. The log file is accessible through an API.
The access to the log files is restricted to owners of a bearer token.

4.2.8.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/ri-logging

4.2.8.3 API Documentation

In the following tables the available HTTP requests are reported.

Operation

URL	http://217.172.12.199:9798/fe/log
METHOD	POST
URL parameters	None

Header	{"sessionId": "<sessionId>"}
Data params	{<log-item>}

Operation

URL	http://217.172.12.199:9798/fe/log
METHOD	GET
URL parameters	None
Header	{"sessionId": "<sessionId>"}

Response example	{
	"_id" : ObjectId("5b17ec2b2fd3113325b5a7d7"),
	"ip" : "127.0.0.1",
	"event_type" : "mouseover",
	"header" : {
	"Host" : "0.0.0.0:9798",
	"Connection" : "keep-alive",
	"Content-Length" : "26329",
	"Pragma" : "no-cache",
	"Cache-Control" : "no-cache",
	"Sessionid" : "hsuPP5K47wmO9QgXNhtN",
	"Origin" : "http://localhost",
	"User-Agent" : "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/66.0.3359.181 Safari/537.36",
	"Content-Type" : "application/json",
	"Accept" : "*/*",
	"Dnt" : "1",
	"Referer" : "http://localhost/rome-demo/index.html",
	"Accept-Encoding" : "gzip, deflate",
	"Accept-Language" : "en-US,en;q=0.9,fr;q=0.8,de;q=0.7,es;q=0.6,it;q=0.5,ru;q=0.4,ro;q=0.3"
	},
	"body" : {
	"type" : "mouseover",
	…<more items>
	}
	}

4.2.8.4 *Internal storage*

The logs are stored in a MongoDB.

4.2.9 ri-orchestration-app

4.2.9.1 *Overview*

Name	ri-orchestration-app
Developer	HITeC
Dependencies	• ri-analytics-classification-google-play-review • ri-collection-explicit-feedback-google-play-review • ri-storage-app
Base path	/hitec/orchestration/app/

Port	9702
Technology	Docker

Table 9. Requirements Intelligence Engine – MS *ri-orchestration-app*

This is the orchestration microservice responsible to coordinate all microservices for app store data analysis. The goal is to simplify the access to the diverse microservices and to coordinate them to achieve a certain goal. The main goal is to define apps that should continuously be observed by OpenReq. In a given interval, the apps and their user reviews are crawled, then classified, and finally stored in the database.

4.2.9.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/ri-orchestration-app

4.2.9.3 API Documentation

The following table describe the available HTTP request in detail.

URL	http://217.172.12.199:9702/hitec/orchestration/app/observe/google-play/package-name/{package_name}/interval/{interval}
METHOD	POST
URL parameters	package_name: the package name of the app we want to crawl interval: specifies how often we want to observe it, e.g., daily

4.2.9.4 Internal storage

None

4.2.10 ri-orchestration-twitter

4.2.10.1 Overview

Name	ri-orchestration-twitter
Developer	HITeC
Dependencies	• ri-storage-twitter • ri-collection-explicit-feedback-twitter • ri-analytics-classification-twitter
Base path	/hitec/orchestration/twitter/
Port	9703
Technology	Docker

Table 10. Requirements Intelligence Engine – MS *ri-orchestration-twitter*

This is the orchestration microservice, which is responsible to coordinate all microservices for Twitter data analysis. The goal is to simplify the access to the diverse microservices and to coordinate them to achieve a certain goal. The main goal is to define Twitter accounts that should continuously be observed by OpenReq. In its current state, the microservice crawls tweets that mention a given account and stores the result in the database.

4.2.10.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/ri-orchestration-twitter

4.2.10.3 API Documentation

In the following table the available HTTP request is described

URL	http://217.172.12.199:9703/hitec/orchestration/twitter/observe/tweet/account/{account_name}/interval/{interval}/lang/{lang}
METHOD	POST
URL parameters	account_name: the name of the Twitter profile to observe, interval: specifies how often we want to observe it, e.g., daily, lang: the language the tweets should be written in

4.2.10.4 Internal storage

None

4.2.11 ri-storage-app

4.2.11.1 Overview

Name	ri-storage-app
Developer	HITeC
Dependencies	None
Base path	/hitec/repository/app/
Port	9681
Technology	Docker

Table 11. Requirements Intelligence Engine – MS *ri-storage-app*

This microservice is the interface to the actual database and persists JSON objects.

4.2.11.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/ri-storage-app

4.2.11.3 API Documentation

The following table describe the available HTTP requests in detail.

Operation

URL	http://217.172.12.199:9681/hitec/repository/app/store/app-page/google-play/
METHOD	POST
URL parameters	None

Request example	{
	    "name": "WhatsApp Messenger",
	    "package_name": "com.whatsapp",
	    "date_crawled": "2017-11-22T14:07:29.866407+01:00",
	    "category": "Communication",
	    "usk": "USK: All ages",
	    "price": "free",
	    "price_value": 0,
	    "price_currency": "€",
	    "description": "This is a great app",
	    "whats_new": "fixed several bugs",
	    "rating": 4.4,
	    "star_count": 61050950,
	    "count_per_rating": {
	        "1": 0,
	        "2": 0,
	        "3": 0,
	        "4": 0,
	        "5": 0
	    },
	    "developer": "WhatsApp Inc.",
	    "top_developer": false,
	    "contains_ads": false ,
	    "in_app_purchase": false ,
	    "last_update": 20171027,
	    "os": "ANDROID",
	    "requires_os_version": "Varies+",
	    "current_software_version": "Varieswithdevice",
	    "similar_apps": ["com.xyz", "com.qwerty"]
	}

Response	200 if ok

Operation

URL	http://217.172.12.199:9681/hitec/repository/app/store/app-review/google-play/
METHOD	POST
URL parameters	None

Request example	[{
	  "review_id": "05df34353efd",
	  "package_name": "co.myapp",
	  "author": "mustermann",
	  "date_posted": 20180524,
	  "rating": 5,
	  "title": "I like this app because…",
	  "body": "it has really nice features",
	  "perma_link": "https: //….",
	  "cluster_is_feature_request": false,
	  "cluster_is_problem_report": true
	}]

Response	200 if ok

Operation

URL	http://217.172.12.199:9681/hitec/repository/app/google-play/package-name/{package_name}/class/{class}
METHOD	GET
URL parameters	package_name: the package name of the app we want to crawl class: problem_report/feature_request
Response example	[{ "review_id": "05df34353efd", "package_name": "co.myapp", "author": "mustermann", "date_posted": 20180524, "rating": 5, "title": "I like this app because…", "body": "it has really nice features", "perma_link": "https: //….", "cluster_is_feature_request": false, "cluster_is_problem_report": true }]

Operation

URL	http://217.172.12.199:9681/hitec/repository/app/observe/app/google-play/package-name/{package_name}/interval/{interval}
METHOD	POST
URL parameters	package_name: the package name of the app we want to crawl Interval: specifies how often we want to observe it, e.g., daily
Response	200 if ok

4.2.11.4 Internal storage

The microservices uses a MongoDB.

4.2.12 ri-storage-twitter

4.2.12.1 Overview

Name	ri-storage-twitter
Developer	HITeC
Dependencies	None
Base path	/hitec/repository/twitter/

Port	9682
Technology	Docker

Table 12. Requirements Intelligence Engine – MS *ri-storage-twitter*

This microservice is the interface to the actual database and persists JSON objects.

4.2.12.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/ri-storage-twitter

4.2.12.3 API Documentation

The following tables describe the available HTTP requests in detail.

Store tweets

URL	http://217.172.12.199:9682/hitec/repository/twitter/store/tweet/
METHOD	POST
URL parameters	None
Request example	[{ "created_at" : 20180501, "favorite_count" : 0, "retweet_count" : 0, "text" : "The football match was amazing!", "status_id" : "154756323465724654", "user_name" : "Mustermann", "in_reply_to_screen_name" : "Fifa", "lang" : "en" }]
Response	200 if ok

Store observable

URL	http://217.172.12.199:9682/hitec/repository/twitter/store/observable/
METHOD	POST
URL parameters	None

Request example	{
	"account_name" : “WindItalia”,
	"interval" : “daily”,
	"lang" : "en"
	}

Response	200 if ok

Operation

URL	http://217.172.12.199:9682/hitec/repository/twitter/account_name/{account_name}/class/{tweet_class}

METHOD	GET
URL parameters	account_name: the twitter account addressed, tweet_class: problem_report/inquiry/irrelevant
Response example	[{ "created_at" : 20180501, "favorite_count" : 0, "retweet_count" : 0, "text" : "The football match was amazing!", "status_id" : "154756323465724654", "user_name" : "Mustermann", "in_reply_to_screen_name" : "Fifa", "lang" : "en", “tweet_class” : “problem_report }]

4.2.12.4 Internal storage

The microservices uses a MongoDB.

4.2.13 update-manager

4.2.13.1 Overview

Name	Update-manager
Developer	ENG
Dependencies	None
Base path	/eng/update-manager/
Port	10709
Technology	SpringBoot

Table 13. Requirements Intelligence Engine – MS *update-manager*

This microservice auto-updates other microservices. It takes as input a policy containing rules. Rules can be time- or time- and volume-dependent (e.g., based on the amount of new training examples available for the underlying machine learning model). When a rule is matched, the microservice requests a training microservice and, if required, updates the model.

4.2.13.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/update-manager.git

4.2.13.3 API Documentation

In the following tables the available HTTP requests are described in detail.

Registry and start

URL	http://217.172.12.199:10709/eng/update-manager/registration
METHOD	POST

URL parameters	None

Request example	{
	"ownerId": "testerOpenreq",
	"policyId": "001",
	"addressNewData": "http://.../openreq/trainable/newDataManager/getNewDataVolume",
	"addressTraining": "http://.../openreq/trainable/trainModel",
	"addressUpdate": "http://.../openreq/trainable/updateModel",
	"rules": [
	{
	"period": 1,
	"timeUnit": "DAYS",
	"volume": 1000,
	"startDateTime": "2018-04-29T18:00:00+02:00[Europe/Berlin]"
	}
	]
	}

Response	200 if ok

Update policy

URL	http://217.172.12.199:10709/eng/update-manager/registration
METHOD	PUT
URL parameters	None

Request example	{
	"ownerId": "testerOpenreq",
	"policyId": "001",
	"addressNewData": "http://.../openreq/trainable/newDataManager/getNewDataVolume",
	"addressTraining": "http://.../openreq/trainable/trainModel",
	"addressUpdate": "http://.../openreq/trainable/updateModel",
	"rules": [
	{
	"period": 1,
	"timeUnit": "DAYS",
	"volume": 1000,
	"startDateTime": "2018-04-29T18:00:00+02:00[Europe/Berlin]"
	}
	]
	}

Response	200 if ok

Delete policy

URL	http://217.172.12.199:10709/eng/update-manager/registration
METHOD	DELETE
URL parameters	None
Request example	{ "ownerId": "testerOpenreq", "policyId": "001", "addressNewData": "http://.../openreq/trainable/newDataManager/getNewDataVolume", "addressTraining": "http://.../openreq/trainable/trainModel",

	"addressUpdate": "http://.../openreq/trainable/updateModel",
	"rules": [{
	"period": 1,
	"timeUnit": "DAYS",
	"volume": 1000,
	"startDateTime": "2018-04-29T18:00:00+02:00[Europe/Berlin]"
	}]
	}

Response	200 if ok

4.2.13.4 Internal storage

Internal file.

4.2.14 twitter-extraction

4.2.14.1 Overview

Name	twitter-extraction
Partner	TU Graz
Dependencies	None
Base path	/
Port	9001
Technology	Docker

Table 14. Requirements Intelligence Engine – MS *twitter-extraction*

4.2.14.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/tug/twitter-extraction

4.2.14.3 API Documentation

In the following tables the main HTTP requests are described in detail.

Retrieve a list with values for given set of requirements indicating their popularity for the crowd on twitter.

URL	http://217.172.12.199:9001/popularity/hashtag
METHOD	POST
URL parameters	None
Body parameters	id: identifier of the requirement title: title/name of the requirement description: textual description of the requirement
Request example	[{ "id": "1", "title": "Distance Measurement",

	"description": "For statistical purposes, a distance measurement is necessary which needs data from a GPS sensor. This data is needed for the evaluation software and therefore stored in memory." }, { "id": "2", "title": "Data-Storage Function", "description": "For evaluation purposes, the data should be stored in the internal memory. The memory is used for saving the measured information such as the distance, the height, the average heart rate, and the calorie consumption. The stored data in the memory will then be used by the evaluation software." }]
Response example	[{"id": 1, "popularity": 0.72}, {"id": 2, "popularity": 0.28}]

4.2.14.4 Internal storage

No internal storage is used. In future versions, cached data will be directly stored in files on the disk.

5 Personal Recommendations for Stakeholder's Engine

This section presents and overview of the Recommendations for Stakeholders Engine as well as a description of the available APIs

5.1 Overview

The Personal Recommendations for Stakeholder's Engine offers recommendations for individual stakeholders. These recommendations are related to the screening and recommendation of relevant requirements, to the improvement of requirements quality, to the prediction of requirements properties, and to the identification of relevant stakeholders. The architecture of this component and the provided services are comprehensively described in [3]. The operational details of the microservices currently available in the OpenReq infrastructure are reported in the following section.

5.2 Microservices and APIs

This section reports the specifics of the 17 microservices composing the Personal Recommendations for Stakeholders Engine.

5.2.1 prs-improving-requirements-quality

5.2.1.1 Overview

Name	prs-improving-requirements-quality
Developer	HITEC
Dependencies	None
Base path	/
Port	9799
Technology	Docker, Python, & Flask

Table 15. Personal Recommendations for Stakeholder's Engine – MS prs-improving-requirements-quality

This microservice gives quick suggestions on how to improve the quality of natural language requirements using different NLP techniques

5.2.1.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/prs-improving-requirements-quality.git

5.2.1.3 API Documentation

In the following tables the available HTTP requests are described in detail.

Operation Check Lexical

URL	http://217.172.12.199:9799/check-lexical
METHOD	POST

URL parameters	None

Request example	{
	"requirements": [{
	"id": 1,
	"elements": [{
	"id": 1,
	"name": "description",
	"text": "This is actually a good requirement.",
	"created_at": 1526385600
	}],
	"status": "new",
	"created_at": 1526385600
	}]
	}

Response example	{
	"1": {
	"1": [{
	"description": "Requirements shall avoid possibilities",
	"index_end": 16,
	"index_start": 8,
	"text": "actually",
	"title": "Actually"
	},{
	"description": "Potentially unclear reference.",
	"index_end": 4,
	"index_start": 0,
	"text": "This",
	"title": "Pronoun"
	},{
	"description": "These vague words and symbols are likely to introduce ambiguity.",
	"index_end": 23,
	"index_start": 19,
	"text": "good",
	"title": "Vague"
	}]
	}
	}

Operation Check Regexps

URL	http://217.172.12.199:9799/check-regexps
METHOD	POST
URL parameters	None

Request example	{
	"requirements": [{
	"id": 1,
	"elements": [{
	"id": 1,
	"name": "description",
	"text": "The system shall read HTML and PDF or DOC files.",
	"created_at": 1526385600
	}],
	"status": "new",
	"created_at": 1526385600
	}]
	}

Response example	{
	"1": {
	"1": [{
	"description": "The combination of “and” and “or” leads to unclear associativity",
	"index_end": 37,

	"index_start": 27, "text": "and PDF or", "title": "Unclear Associativity" }] } }

Operation Check POS Regexps

URL	http://217.172.12.199:9799/check-pos-regexps
METHOD	POST
URL parameters	None

Request example	{
	"requirements": [{
	"id": 1,
	"elements": [{
	"id": 1,
	"name": "description",
	"text": "The system will be tested.",
	"created_at": 1526385600
	}],
	"status": "new",
	"created_at": 1526385600
	}]
	}

Response example	{
	"1": {
	"1": [{
	"description": "Authors should state requirements in active form, as passive conceals who is responsible for the action.",
	"index_end": 25,
	"index_start": 16,
	"text": "be tested",
	"title": "Passive Ambiguity"
	}]
	}
	}

5.2.1.4 *Internal storage*

None

5.2.2 uploader

5.2.2.1 *Overview*

Name	uploader
Developer	ENG
Dependencies	None
Base path	/api_t_33/uploader
Port	10602

Technology	Docker

Table 16. Personal Recommendations for Stakeholder's Engine – MS uploader

This service uploads a .docx file and calls the orchestrator service which orchestrates other microservices (see section 5.2.3). The uploader takes a document name as input and returns a list of enriched dictionaries, one for each document block, as output. The above services (i.e., uploader, entrypoint, and the orchestrated microservices) have been included in the component called *prs-improving-requirements-quality-features*.

5.2.2.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.2.3 API Documentation

The HTTP request is described in detail in the following table. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/uploader
Method	GET or POST
URL parameters	None
Request example	{“file”: “example_tender1.docx”}
Response example	{"content": {"enrichedParagraphs": {"prettifyEnrichedParagraphs": [{"requirement": {"created_at": 1530354687, "id": "L37_P1"}, "requirementParts": [{"content":{"min.": 0, "maximum": 0, "specifically": 0, "can be": 0, "namely": 0, "minimum": 0, "max.": 0}, "created_at": 1530354687, "id": "L37_P1", "name": "d0"}, {"content": {"limitations": 0, "": 0, "tests": 0, "installation": 0, "installations": 0, "equipment": 0, "disposal": 0, "measures": 0, "relocation": 0, "removal": 0, "facilities": 0, "protection": 0, "reconstruction": 0, "organization": 0, "performance": 0, "construction": 0, "new situation": 0, "conditions": 0}, "created_at ": 1530354687, "id": "L37_P1", "name": "d1"}, {"content": {"": 0, "it is necessary to": 0, "may": 0, "shall": 0, "must be": 0, "should be": 0, "is necessary": 0}, "created_at": 1530354687, "id": "L37_P1", "name": "d2"}, {"content": {"will be": 0, "is planned": 0, "is envisaged": 0, "it's planned": 0, "it's foreseen": 0 , "is foreseen": 0, "following": 0}, "created_at": 1530354687, "id": "L37_P1", "name": "d3"}, {"content": ["Gradec,_Zagreb", "Train_station", "Dugo_Selo", "Gradec,_Zagreb"], "created_at": 1530354687, "id": "L37_P1", "name": "dbpediaEntities"}, {"content": {"Gradec,_Zagreb": ["http://dbpedia.org/resource/Category:History_of_Zagreb", "http://dbpedia.org/resource/Category:Gornji_Grad\u2013Medve\u0161\u010dak","http://dbpedia.org/resource/Category:Neighborhoods_of_Zagreb"],"Train_station":["http://dbpedia.org/resource/Category:Railway_stations"], "Dugo_Se lo": ["http://dbpedia.org/resource/Category:Cities_and_towns_in_Croatia", "http://dbpedia.org/resource/Category:Zagreb_County_(former)", "http://dbpedia.org/resource/Category:Populated_places_in_Zagreb_County"]}, "created_at": 1530354687, "id": "L37_P1", "name": "dbpediaEntitiesType"}, {"content": [{"subchild": [], "objects_extended": ["The station", "Ivan \u017dabno railway"],"lemmatized_verb":"connect", "verbs": "connected","dependence": [], "adjectives": [], "objects":["station", "railway"], "subjects_extended": ["The Gradec Railway Station"], "subjects": ["Station"], "hash_number": "12414132254122877891"}, {"subchild": [], "objects_extended": ["the second track"], "lemmatized_verb": "add", "verbs": "added", "dependence": "12414132254122877891", "adjectives": ["such"], "objects": [" track"], "subjects_extended": [], "subjects": [], "hash_number": "9384810846185141164"}, {"subchild": [], "objects_extended": ["the turnout track"], "lemmatized_verb": "build", "verbs": "built", "dependence": "12414132254122877891", "adjectives": [], "objects": ["track"], "subjects_extended": [], "subjects": [], "hash_number": "4356439182563492767"}], "created_at": 1530354687, "id": "L37_P1", "nam e": "structure"}, {"text": "The Gradec Railway Station be a new railway station on the Dugo Selo \u2013 Kri\u017eevci railway consist of two

	throughway track and one truncate track . The station be build such that the second track be add and build on the left side of the exist track and the turnout track be build on the right side of the exist track with which the new Gradec - Sv. Ivan \u017dabno railway be connect .", "created_at": 1530354687, "id": "L37_P1", "name": "lemmatizedContent"}, {"text": {"ease": 72.8, "kincaid": 9.0}, "created_at": 1530354687, "id": "L37_P1", "name": "formalMetrics"}, {"text": "1.3.2.4\tGradec Railway Station\n", "created_at": 1530354687, "id": "L37_P1", "name": "Title"}, {"text": "The Gradec Railway Station is a new railway station on the Dugo Selo \u2013 Kri\u017eevci railway consisting of two throughway tracks and one truncated track. The station is built such that the second track is added and built on the left side of the existing track and the turnout track is built on the right side of the existing track with which the new Gradec- Sv. Ivan \u017dabno railway is connected.\n", "created_at": 1530354687, "id": "L37_P1", "name": "Paragraphs"}]}, "error": {"status": 600, "code": "OK", "description": "Application run normally."}}

5.2.2.4 *Internal storage*

None

5.2.3 entrypoint

5.2.3.1 *Overview*

Name	entrypoint
Developer	ENG
Dependencies	None
Base path	/api_t_33/entrypoint
Port	10602
Technology	Docker

Table 17. Personal Recommendations for Stakeholder’s Engine – MS entrypoint

This service orchestrates several microservices (described in the following sections) responsible of document splitting, preprocessing, features extraction, enrichment and output mapping (for further details see deliverable D3.3). It takes a string with path and filename passed as a JSON object as input, and returns an enriched dictionary of blocks.

5.2.3.2 *Source code*

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.3.3 *API Documentation*

The HTTP request is extensively described in the following table. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/entrypoint
Method	POST
URL parameters	None

Request example	{“documentName”:”‘D:\\shared\openreq\data\example_tender1.docx”}
Response example	{"content": {"enrichedParagraphs": {"prettifyEnrichedParagraphs": [{"requirement": {"created_at": 1530354687, "id": "L37_P1"}, "requirementParts": [{"content":{"min.": 0, "maximum": 0, "specifically": 0, "can be": 0, "namely": 0, "minimum": 0, "max.": 0}, "created_at": 1530354687, "id": "L37_P1", "name": "d0"}, {"content": {"limitations": 0, "": 0, "tests": 0, "installation": 0, "installations": 0, "equipment": 0, "disposal": 0, "measures": 0, "relocation": 0, "removal": 0, "facilities": 0, "protection": 0, "reconstruction": 0, "organization": 0, "performance": 0, "construction": 0, "new situation": 0, "conditions": 0}, "created_at ": 1530354687, "id": "L37_P1", "name": "d1"}, {"content": {"": 0, "it is necessary to": 0, "may": 0, "shall": 0, "must be": 0, "should be": 0, "is necessary": 0}, "created_at": 1530354687, "id": "L37_P1", "name": "d2"}, {"content": {"will be": 0, "is planned": 0, "is envisaged": 0, "it's planned": 0, "it's foreseen": 0 , "is foreseen": 0, "following": 0}, "created_at": 1530354687, "id": "L37_P1", "name": "d3"}, {"content": ["Gradec,_Zagreb", "Train_station", "Dugo_Selo", "Gradec,_Zagreb"], "created_at": 1530354687, "id": "L37_P1", "name": "dbpediaEntities"}, {"content": {"Gradec,_Zagreb": ["http://dbpedia.org/resource/Category:History_of_Zagreb", "http://dbpedia.org/resource/Category:Gornji_Grad\u2013Medve\u0161\u010dak","http://dbpedia.org/resource/Category:Neighborhoods_of_Zagreb"],"Train_station":["http://dbpedia.org/resource/Category:Railway_stations"], "Dugo_Se lo": ["http://dbpedia.org/resource/Category:Cities_and_towns_in_Croatia", "http://dbpedia.org/resource/Category:Zagreb_County_(former)", "http://dbpedia.org/resource/Category:Populated_places_in_Zagreb_County"]}, "created_at": 1530354687, "id": "L37_P1", "name": "dbpediaEntitiesType"}, {"content": [{"subchild": [], "objects_extended": ["The station", "Ivan \u017dabno railway"],"lemmatized_verb":"connect", "verbs": "connected","dependence": [], "adjectives": [], "objects":["station", "railway"], "subjects_extended": ["The Gradec Railway Station"], "subjects": ["Station"], "hash_number": "12414132254122877891"}, {"subchild": [], "objects_extended": ["the second track"], "lemmatized_verb": "add", "verbs": "added", "dependence": "12414132254122877891", "adjectives": ["such"], "objects": [“track"], "subjects_extended": [], "subjects": [], "hash_number": "9384810846185141164"}, {"subchild": [], "objects_extended": ["the turnout track"], "lemmatized_verb": "build", "verbs": "built", "dependence": "12414132254122877891", "adjectives": [], "objects": ["track"], "subjects_extended": [], "subjects": [], "hash_number": "4356439182563492767"}], "created_at": 1530354687, "id": "L37_P1", "name": "structure"}, {"text": "The Gradec Railway Station be a new railway station on the Dugo Selo \u2013 Kri\u017eevci railway consist of two throughway track and one truncate track . The station be build such that the second track be add and build on the left side of the exist track and the turnout track be build on the right side of the exist track with which the new Gradec - Sv. Ivan \u017dabno railway be connect .", "created_at": 1530354687, "id": "L37_P1", "name": "lemmatizedContent"}, {"text": {"ease": 72.8, "kincaid": 9.0}, "created_at": 1530354687, "id": "L37_P1", "name": "formalMetrics"}, {"text": "1.3.2.4\tGradec Railway Station\n", "created_at": 1530354687, "id": "L37_P1", "name": "Title"}, {"text": "The Gradec Railway Station is a new railway station on the Dugo Selo \u2013 Kri\u017eevci railway consisting of two throughway tracks and one truncated track. The station is built such that the second track is added and built on the left side of the existing track and the turnout track is built on the right side of the existing track with which the new Gradec- Sv. Ivan \u017dabno railway is connected.\n", "created_at": 1530354687, "id": "L37_P1", "name": "Paragraphs"}]}, "error": {"status": 600, "code": "OK", "description": "Application run normally."}}'

5.2.3.4 Internal storage

None

5.2.4 parsing-conversion-doc

5.2.4.1 Overview

Name	parsing-conversion-doc
Developer	ENG
Dependencies	None
Base path	/api_t_33/parsing/conversion/doc

Port	10602
Technology	Docker

Table 18. Personal Recommendations for Stakeholder's Engine – MS parsing-conversion-doc

This service parses a document and returns parsed text. The input is a JSON object with path and filename specified, the output is a JSON object with parsed text.

5.2.4.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.4.3 API Documentation

In the following table the available HTTP request is described in detail. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/parsing/conversion/doc
Method	POST
URL parameters	None
Request example	{'documentName': 'D:\\shared\openreq\data\example_tender1.docx'}
Response	200 if OK

5.2.4.4 Internal storage

None

5.2.5 parsing-segmentation

5.2.5.1 Overview

Name	parsing-segmentation
Developer	ENG
Dependencies	None
Base path	/api_t_33/parsing/segmentation
Port	10602
Technology	Docker

Table 19. Personal Recommendations for Stakeholder's Engine – MS parsing-segmentation

The service reads parsed text and segments it into blocks. It takes a JSON object with path and filename specified as input, and returns a JSON object with a list of blocs as output. It optionally saves the blocs in a .txt file

5.2.5.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.5.3 API Documentation

The HTTP request description is comprehensively reported in the table below. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/parsing/segmentation
Method	POST
URL parameters	None
Request example	{“documentName”: “D:\\shared\openreq\data\example_tender1.txt”}
Response example	{“content”: {“listOfBlocks” : [“Railway line Dugo Selo – Križevci is a constituent part of the branch Vb , Paneuropean corridor pass through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo . The exist railway line has one track , large inter-station distance and is already now of limited transportation and maximum railway capacity with no possibility of capacity enhance” ”]}, "error": {"status": 600, "code": "OK", "description": "Application run normally."}}

5.2.5.4 Internal storage

None

5.2.6 parsing-to_dict

5.2.6.1 Overview

Name	parsing-to_dict
Developer	ENG
Dependencies	None
Base path	/api_t_33/parsing/to_dict
Port	10602
Technology	Docker

Table 20. Personal Recommendations for Stakeholder’s Engine – MS parsing-to_dict

The service converts a list of blocks into a list of structured dictionaries, with the following keys:

- “Line”: the line number of each block
- “Title”: the main document section title where the block is contained
- “Dependence”: the bullet title that contains other subsections [optional]
- “Bullet”: the bullet of the block [optional]

- “Paragraph”: a block session defined by a capital letter at the beginning of the string and ‘\n’ at the end. A block can contain one or more paragraphs. [optional]
- “PID”: the unique identifier of a paragraph. [optional]

5.2.6.2 *Source code*

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.6.3 *API Documentation*

The available HTTP request is described in detail below. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/parsing/to_dict
Method	POST
URL parameters	None
Request example	{“listOfBlocks” : ["Railway line Dugo Selo – Križevci is a constituent part of the branch Vb , Paneuropean corridor pass through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo . The exist railway line has one track, large inter-station distance and be already now of limited transportation and maximum railway capacity with no possibility of capacity enhance”]}
Response example	{“content” : [{“Line” : “L71", “Dependence” : “1\tTECHNICAL DESCRIPTION\n” “Title” : “1.1\tINTRODUCTION\n ” , ‘’Bullet” : “ ”, “Paragraphs” : ["Railway line Dugo Selo – Križevci be a constituent part of the branch Vb , Paneuropean corridor pass through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo . The exist railway line have one track , large inter-station distance and is already now of limited transportation and maximum railway capacity with no possibility of capacity enhance\n"] , “PID” : [“P1”] }], "error": {"status": 600, "code": "OK", "description": "Application run normally."}}

5.2.6.4 *Internal storage*

None

5.2.7 *parsing-enrich*

5.2.7.1 *Overview*

Name	parsing-enrich
Developer	ENG
Dependencies	None
Base path	/api_t_33/parsing/enrich
Port	10602

Technology	Docker

Table 21. Personal Recommendations for Stakeholder's Engine – MS parsing-enrich

This service takes a list of structured dictionaries as input and returns a list of enriched dictionaries, one for each paragraph or bullet.

5.2.7.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.7.3 API Documentation

The available HTTP request is described in detail below. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/parsing/enrich
Method	POST
URL parameters	None
Request example	{“structuredDictList” : [{“Line” : “L71", “Dependence” : “1\tTECHNICAL DESCRIPTION\n” “Title” : “1.1\tINTRODUCTION\n ” , ‘’Bullet” : “ ”, “Paragraphs” : ["Railway line Dugo Selo – Križevci be a constituent part of the branch Vb , Paneuropean corridor pass through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo . The exist railway line has one track , large inter-station distance and is already now of limited transportation and maximum railway capacity with no possibility of capacity enhance\n"], “PID” : [“P1”]}
Response example	{“content” : {“EnrichedDictList : [{"lemmatizedContent": "Railway line Dugo Selo – Križevci be a constituent part of the branch Vb , Paneuropean corridor pass through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo . The exist railway line have one track , large inter-station distance and be already now of limited transportation and maximum railway capacity with no possibility of capacity enhance .", "dbpediaEntitiesType": {"Koprivnica": ["http://dbpedia.org/resource/Category:Cities_and_towns_in_Croatia", "http://dbpedia.org/resource/Category:1272_establishments_in_Europe", "http://dbpedia.org/resource/Category:Bjelovar-Križevci_County", "http://dbpedia.org/resource/Category:Populated_places_established_in_the_13th_century", "http://dbpedia.org/resource/Category:Populated_places_in_Koprivnica-Križevci_County", "http://dbpedia.org/resource/Category:Koprivnica", "http://dbpedia.org/resource/Category:13th-century_establishments_in_Croatia"], "Dugo_Selo": ["http://dbpedia.org/resource/Category:Cities_and_towns_in_Croatia", "http://dbpedia.org/resource/Category:Zagreb_County_(former)", "http://dbpedia.org/resource/Category:Populated_places_in_Zagreb_County"], "M201_railway_(Croatia)": ["http://dbpedia.org/resource/Category:International_railway_lines", "http://dbpedia.org/resource/Category:Railway_lines_in_Croatia"], "Croatia": ["http://dbpedia.org/resource/Category:Balkan_countries",

```
            "http://dbpedia.org/resource/Category:Countries_in_Europe",
            "http://dbpedia.org/resource/Category:Croatia",
   "http://dbpedia.org/resource/Category:Croatian-speaking
   _countries_and_territories",
            "http://dbpedia.org/resource/Category:Liberal_democracies",
   "http://dbpedia.org/resource/Category:Member_states_of_NATO",
   "http://dbpedia.org/resource/Category:Member_states_of_
   the_Council_of_Europe"
    "http://dbpedia.org/resource/Category:Member_states_of_
    the_European_Union",
   "http://dbpedia.org/resource/Category:Member_states_
   of_the_Union_for_the_Mediterranean",
   "http://dbpedia.org/resource/Category:Member_states_
   of_the_United_Nations",
   "http://dbpedia.org/resource/Category:Republics",
   "http://dbpedia.org/resource/Category:Slavic_countries_
   and_territories",
   "http://dbpedia.org/resource/Category:Southeastern_
   European_countries",
   "http://dbpedia.org/resource/Category:Southern_European_
   countries",
   "http://dbpedia.org/resource/Category:States_and_
   territories_established_in_1991" ],
         "Pan-European_identity": [
   "http://dbpedia.org/resource/Category:National_identities",
   "http://dbpedia.org/resource/Category:Political_neologisms",
   "http://dbpedia.org/resource/Category:Pan-Europeanism" ]},
      "Title": "1.1\tINTRODUCTION\n",
      "d2": { "": 0, "shall": 0, "may": 0, "must be": 0,  "should be": 0, "it is
   necessary to": 0,  "is necessary": 0 },
      "d3": {"is planned": 0, "will be": 0, "is envisaged": 0, "it's planned": 0,
   it's foreseen": 0,  "is foreseen": 0,  "following": 0},
      "d0": {"min.": 0, "maximum": 1, "specifically": 0, "can be": 0,
   "namely": 0,  "minimum": 0, "max.": 0},
      "d1": {"": 0,"limitations": 0, "tests": 0, "installation": 0,
   "Installations": 0, "facilities": 0,  "protection": 0, "disposal": 0,
   "measures": 0,  "relocation": 0, "removal": 0,  "equipment": 0,
   "construction": 0, "reconstruction": 0, "performance": 0,
   "organization": 0, "new situation": 0, "conditions": 0},
      "formalMetrics": null,
      "PID": "P1",
      "dbpediaEntities": ["Dugo_Selo", "Pan-European_identity", "Croatia",
   "M201_railway_(Croatia)", "Koprivnica",  "Dugo_Selo" ],
      "entities": [{"subchild": [], "adjectives": [],
   "hash_number": "1248239241591158246",
   "objects_extended": [ "one track , large inter - station distances"],
   "lemmatized_verb": "have",  "objects": ["track"], "subjects":
   ["Križevci","line], "subjects_extended": ["Railway line Dugo Selo –
   Križevci","The existing railway line"], "dependence": [], "verbs": "has"},
   {"subchild": [], "adjectives": [],
   "hash_number":"3411606890003347522",  "objects_extended": [],
   "lemmatized_verb": "be", "objects": [], "subjects": [],
   "subjects_extended": [],"dependence": "1248239241591158246",
   "verbs": "is" }],
      "Paragraphs": "Railway line Dugo Selo – Križevci is a constituent
   part of the branch Vb, Paneuropean  corridor passing through the
   territory of the Republic of Croatia and the railway line M201 State
   border – Koprivnica – Dugo Selo. The existing railway line has one
   track, large inter-station distances and is already now of limited
   transportation and maximum railway capacity with no possibilities of
   capacity enhancing.\n",
      "Line": "L71"}]
"error": {"status": 600, "code": "OK", "description": "Application run
   normally."}}’
```

5.2.7.4 Internal storage

None

5.2.8 *parsing-enrich-prettify*

5.2.8.1 Overview

Name	parsing-enrich-prettify
Developer	ENG
Dependencies	None
Base path	/api_t_33/parsing/enrich/prettify
Port	10602
Technology	Docker

Table 22. Personal Recommendations for Stakeholder's Engine – MS parsing-enrich-prettify

The service maps the output from the parsing-enrich service presented in Section 5.2.7 (i.e. a list of dictionaries for each block) onto the following JSON schema

5.2.8.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.8.3 API Documentation

In the following table the available HTTP request is described in detail. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/parsing/enrich/prettify
Method	POST
URL parameters	None
Request example	{"enrichedDictList" : [{"EnrichedDictList : [{"lemmatizedContent": "Railway line Dugo Selo – Križevci be a constituent part of the branch Vb , Paneuropean corridor pass through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo . The exist railway line have one track , large inter-station distance and be already now of limited transportation and maximum railway capacity with no possibility of capacity enhance .", "dbpediaEntitiesType": {"Koprivnica": ["http://dbpedia.org/resource/Category:Cities_and_towns_in_Croatia", "http://dbpedia.org/resource/Category:1272_establishments_in_Europe", "http://dbpedia.org/resource/Category:Bjelovar-Križevci_County", "http://dbpedia.org/resource/Category:Populated_places_established_in_the_13th_century", "http://dbpedia.org/resource/Category:Populated_places_in_Koprivnica-Križevci_County", "http://dbpedia.org/resource/Category:Koprivnica", "http://dbpedia.org/resource/Category:13th-century_establishments_in_Croatia"], "Dugo_Selo": ["http://dbpedia.org/resource/Category:Cities_and_towns_in_Croatia", "http://dbpedia.org/resource/Category:Zagreb_County_(former)", "http://dbpedia.org/resource/Category:Populated_places_in_Zagreb_County"], "M201_railway_(Croatia)": ["http://dbpedia.org/resource/Category:International_railway_lines",

	"http://dbpedia.org/resource/Category:Railway_lines_in_Croatia"], "Croatia": ["http://dbpedia.org/resource/Category:Balkan_countries", "http://dbpedia.org/resource/Category:Countries_in_Europe", "http://dbpedia.org/resource/Category:Croatia", "http://dbpedia.org/resource/Category:Croatian-speaking _countries_and_territories", "http://dbpedia.org/resource/Category:Liberal_democracies", "http://dbpedia.org/resource/Category:Member_states_of_NATO", "http://dbpedia.org/resource/Category:Member_states_of_ the_Council_of_Europe", "http://dbpedia.org/resource/Category:Member_states_of_ the_European_Union", "http://dbpedia.org/resource/Category:Member_states_ of_the_Union_for_the_Mediterranean", "http://dbpedia.org/resource/Category:Member_states_ of_the_United_Nations", "http://dbpedia.org/resource/Category:Republics", "http://dbpedia.org/resource/Category:Slavic_countries_ and_territories", "http://dbpedia.org/resource/Category:Southeastern_ European_countries", "http://dbpedia.org/resource/Category:Southern_European_ countries", "http://dbpedia.org/resource/Category:States_and_ territories_established_in_1991"], "Pan-European_identity": ["http://dbpedia.org/resource/Category:National_identities", "http://dbpedia.org/resource/Category:Political_neologisms", "http://dbpedia.org/resource/Category:Pan-Europeanism"]}, "Title": "1.1\tINTRODUCTION\n", "d2": { "": 0, "shall": 0, "may": 0, "must be": 0, "should be": 0, "it is necessary to": 0, "is necessary": 0 }, "d3": {"is planned": 0, "will be": 0, "is envisaged": 0, "it's planned": 0, it's foreseen": 0, "is foreseen": 0, "following": 0}, "d0": {"min.": 0, "maximum": 1, "specifically": 0, "can be": 0, "namely": 0, "minimum": 0, "max.": 0}, "d1": {"": 0,"limitations": 0, "tests": 0, "installation": 0, "Installations": 0, "facilities": 0, "protection": 0, "disposal": 0, "measures": 0, "relocation": 0, "removal": 0, "equipment": 0, "construction": 0, "reconstruction": 0, "performance": 0, "organization": 0, "new situation": 0, "conditions": 0}, "formalMetrics": null, "PID": "P1", "dbpediaEntities": ["Dugo_Selo", "Pan-European_identity", "Croatia", "M201_railway_(Croatia)", "Koprivnica", "Dugo_Selo"], "entities": [{"subchild": [], "adjectives": [], "hash_number": "1248239241591158246", "objects_extended": ["one track , large inter - station distances"], "lemmatized_verb": "have", "objects": ["track"], "subjects": ["Križevci","line"], "subjects_extended": ["Railway line Dugo Selo – Križevci","The existing railway line"], "dependence": [], "verbs": "has"}, {"subchild": [], "adjectives": [], "hash_number":"3411606890003347522", "objects_extended": [], "lemmatized_verb": "be", "objects": [], "subjects": [], "subjects_extended": [],"dependence": "1248239241591158246", "verbs": "is" }], "Paragraphs": "Railway line Dugo Selo – Križevci is a constituent part of the branch Vb, Paneuropean corridor passing through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo. The existing railway line has one track, large inter-station distances and is already now of limited transportation and maximum railway capacity with no possibilities of capacity enhancing.\n", "Line": "L71"}]]}
Response example	{“enrichedDictList” : [{“EnrichedDictList : [{"lemmatizedContent": "Railway line Dugo Selo – Križevci be a constituent part of the branch Vb , Paneuropean corridor pass through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo . The exist railway line have one track , large inter-station distance

```
and be already now of limited transportation and maximum railway
 capacity with no possibility of capacity enhance .",
    "dbpediaEntitiesType": {"Koprivnica":
 ["http://dbpedia.org/resource/Category:Cities_and_towns_in_Croatia",
 "http://dbpedia.org/resource/Category:1272_establishments
 _in_Europe",
 "http://dbpedia.org/resource/Category:Bjelovar-Križevci_County",
 "http://dbpedia.org/resource/Category:Populated_places_
 established_in_the_13th_century",
 "http://dbpedia.org/resource/Category:Populated_places
 _in_Koprivnica-Križevci_County",
 "http://dbpedia.org/resource/Category:Koprivnica",
 "http://dbpedia.org/resource/Category:13th-century_
 establishments_in_Croatia"],  "Dugo_Selo":
 ["http://dbpedia.org/resource/Category:Cities_and_towns_in_Croatia",
 "http://dbpedia.org/resource/Category:Zagreb_County_(former)",
 "http://dbpedia.org/resource/Category:Populated_places_
 in_Zagreb_County"],
              "M201_railway_(Croatia)": [
 "http://dbpedia.org/resource/Category:International_railway_lines",
 "http://dbpedia.org/resource/Category:Railway_lines_in_Croatia" ],
        "Croatia":
 ["http://dbpedia.org/resource/Category:Balkan_countries",
 "http://dbpedia.org/resource/Category:Countries_in_Europe",
"http://dbpedia.org/resource/Category:Croatia",
 "http://dbpedia.org/resource/Category:Croatian-speaking
 _countries_and_territories",
            "http://dbpedia.org/resource/Category:Liberal_democracies",
 "http://dbpedia.org/resource/Category:Member_states_of_NATO",
 "http://dbpedia.org/resource/Category:Member_states_of_
  the_Council_of_Europe",
  "http://dbpedia.org/resource/Category:Member_states_of_
  the_European_Union",
 "http://dbpedia.org/resource/Category:Member_states_
 of_the_Union_for_the_Mediterranean",
 "http://dbpedia.org/resource/Category:Member_states_
 of_the_United_Nations",
 "http://dbpedia.org/resource/Category:Republics",
 "http://dbpedia.org/resource/Category:Slavic_countries_
 and_territories",
 "http://dbpedia.org/resource/Category:Southeastern_
 European_countries",
 "http://dbpedia.org/resource/Category:Southern_European_
 countries",
 "http://dbpedia.org/resource/Category:States_and_
 territories_established_in_1991" ],
        "Pan-European_identity": [
 "http://dbpedia.org/resource/Category:National_identities",
 "http://dbpedia.org/resource/Category:Political_neologisms",
 "http://dbpedia.org/resource/Category:Pan-Europeanism" ]},
    "Title": "1.1\tINTRODUCTION\n",
    "d2": { "": 0, "shall": 0, "may": 0, "must be": 0,  "should be": 0, "it is
 necessary to": 0,  "is necessary": 0 },
    "d3": {"is planned": 0, "will be": 0, "is envisaged": 0, "it's planned": 0,
 it's foreseen": 0,  "is foreseen": 0,  "following": 0},
    "d0": {"min.": 0, "maximum": 1, "specifically": 0, "can be": 0,
 "namely": 0,  "minimum": 0, "max.": 0},
    "d1": {"": 0,"limitations": 0, "tests": 0, "installation": 0,
 "Installations": 0, "facilities": 0,  "protection": 0, "disposal": 0,
 "measures": 0,  "relocation": 0, "removal": 0,  "equipment": 0,
 "construction": 0, "reconstruction": 0, "performance": 0,
 "organization": 0, "new situation": 0, "conditions": 0},
    "formalMetrics": null,
    "PID": "P1",
    "dbpediaEntities": ["Dugo_Selo", "Pan-European_identity", "Croatia",
 "M201_railway_(Croatia)", "Koprivnica",  "Dugo_Selo" ],
    "entities": [{"subchild": [], "adjectives": [],
 "hash_number": "1248239241591158246",
 "objects_extended": [ "one track , large inter - station distances"],
 "lemmatized_verb": "have",  "objects": ["track"], "subjects":
 ["Križevci","line], "subjects_extended": ["Railway line Dugo Selo –
 Križevci","The existing railway line"], "dependence": [], "verbs": "has"},
```

	{"subchild": [], "adjectives": [], "hash_number":"3411606890003347522", "objects_extended": [], "lemmatized_verb": "be", "objects": [], "subjects": [], "subjects_extended": [],"dependence": "1248239241591158246", "verbs": "is" }], "Paragraphs": "Railway line Dugo Selo – Križevci is a constituent part of the branch Vb, Paneuropean corridor passing through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo. The existing railway line has one track, large inter-station distances and is already now of limited transportation and maximum railway capacity with no possibilities of capacity enhancing.\n", "Line": "L71"}]]}'

5.2.8.4 Internal storage

None

5.2.9 lemmatizer-<string:extr_type>

5.2.9.1 Overview

Name	lemmatizer-<string:extr_type>
Developer	ENG
Dependencies	None
Base path	/api_t_33/lemmatizer/<string:extr_type>
Port	10602
Technology	Docker

Table 23. Personal Recommendations for Stakeholder's Engine – MS lemmatizer-<string:extr_type>

The service applies lemmatization to input text. It takes a block and returns the block lemmatized.

5.2.9.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.9.3 API Documentation

The available HTTP request is described in detail in the following table. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/lemmatizer/<string:extr_type>
Method	POST
URL parameters	Language model alias for the lemmatizer, e.g. 'en' for English
Request example	{"documents" : [u'Railway line Dugo Selo – Križevci is a constituent part of the branch Vb, Paneuropean corridor passing through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo. The

	existing railway line has one track, large inter-station distances and is already now of limited transportation and maximum railway capacity with no possibilities of capacity enhancing.\n']}
Response example	{“content” : {"lemmatizedDocuments": [["Railway", "line", "Dugo", "Selo", "\\u2013", "Kri\\u017eevci", "be", "a", "constituent", "part", "of", "the", "branch", "Vb", ",", "Paneuropean", "corridor", "pass", "through", "the", "territory", "of", "the", "Republic", "of", "Croatia", "and", "the", "railway", "line", "M201", "State", "border", "\\u2013", "Koprivnica", "\\u2013", "Dugo", "Selo", ".", "The", "exist", "railway", "line", "have", "one", "track", ",", "large", "inter-station", "distance", "and", "be", "already", "now", "of", "limited", "transportation", "and", "maximum", "railway", "capacity", "with", "no", "possibility", "of", "capacity", "enhance", "."], ["By", "this", "project", ",", "the", "section", "Dugo", "Selo", "\\u2013", "Kri\\u017eevci", "be", "turn", "into", "two-track", "railway", "line", "with", "much", "more", "acceptable", "usage", "characteristic", "by", "which", "the", "condition", "of", "interoperability", "be", "meet", ",", "the", "capacity", "be", "enhance", "and", "the", "time", "of", "travel", "be", "significantly", "reduce", "."], ["By", "planned", "activity", "for", "the", "project", "of", "reconstruct", "the", "existing", "and", "build", "the", "second", "track", "on", "the", "section", "Dugo", "Selo", "\\u2013", "Kri\\u017eevci", ",", "we", "wish", "to", "achieve", "the", "velocity", "of", "train", "of", "160", "km/h", "with", "additional", "increase", "of", "safety", "in", "traffic", ",", "significant", "enhancement", "of", "capacity", "with", "reduction", "in", "time", "of", "travel", "and", "increased", "quality", "in", "travel", ",", "acceptance", "of", "interoperable", "freight", "train", "750", "m", "in", "length", "and", "of", "allowable", "weight", "25", "t/o", "namely", "of", "train", "for", "transport", "of", "passenger", "which", "be", "400", "m", "in", "length", "and", "to", "additionally", "protect", "the", "environment", "and", "enhance", "the", "level", "of", "quality", "of", "life", "of", "the", "people", "live", "along", "the", "railway", "line", "."], ["For", "that", "purpose", "it", "be", "necessary", "to", "perform", "the", "planned", "activity", "which", "include", "relocation", "of", "all", "exist", "installation", "and", "removal", "of", "exist", "facility", ",", "construct", "of", "the", "second", "track", "and", "the", "reconstruction", "of", "the", "exist", "one", "as", "well", "a", "the", "exist", "railway", "station", "(", "Dugo", "Selo", ",", "Vrbovec", "and", "Kri\\u017eevci", ")", ",", "construction", "of", "the", "new", "railway", "station", "(", "Gradec", ")", "and", "of", "station", "along", "with", "the", "reconstruction", "and", "design", "of", "exist", "railway", "station", "building", ".", "It", "also", "include", "the", "deviation", "of", "the", "railway", "line", "M103", "Dugo", "Selo", "\\u2013", "Novska", ",", "from", "the", "railway", "station", "Dugo", "Selo", "to", "Ostrna", ",", "construction", "of", "new", "and", "the", "reconstruction", "of", "exist", "bridge", ",", "close", "of", "railway-road", "and", "pedestrian", "crossing", "and", "the", "construction", "of", "crossing", "above", "the", "level", "and", "pedestrian", "underpass", ",", "construction", "of", "maintenance", "road", "and", "the", "road", "which", "connect", "the", "crossing", "above", "the", "level", "with", "exist", "traffic", "road", ",", "arrangement", "of", "internal", "and", "external", "drainage", "on", "open", "railway", "section", ",", "railway", "station", ",", "maintenance", "road", "and", "facility", ".", "It", "include", "the", "raise", "of", "stable", "facility", "for", "electric", "traction", ",", "modernization", "of", "safety", "signal", "and", "telecommunication", "device", "and", "installation", "of", "ETCS", "level", "1", "insurance", "and", "construction", "of", "noise", "protection", "wall", "and", "maintenance", "road", "."], ["Starting", "railway", "station", "on", "the", "envisaged", "section", "be", "the", "railway", "station", "in", "Dugo", "Selo", ".", "Within", "the", "main", "project", ",", "facility", "be", "position", "accord", "to", "exist", "chainages", "which", "be", "completely", "level", "with", "the", "chainages", "which", "start", "at", "the", "railway", "station", "Dugo", "Selo", "for", "which", "a", "new", "kilometre", "position", "km", "0+000=445+159", "have", "be", "choose", ".", "Chainages", "increase", "from", "the", "railway", "station", "in", "Dugo", "Selo", "towards", "the", "railway", "station", "in", "Kri\\u017eevci", "and", "look", "in", "that", "direction", ",", "during", "the", "description", "of", "individual", "work", ",", "side", "of", "the", "railway", "line", "namely", "railway", "station", "(", "left-right", ")", "be", "mention", "."]]}, "error": {"status": 600, "code": "OK", "description": "Application run normally."}}

5.2.9.4 *Internal storage*

None

5.2.10 keywords-supervised-<string:extr_type>

5.2.10.1 Overview

Name	keywords/supervised/<string:extr_type>
Developer	ENG
Dependencies	None

Base path	/api_t_33/keywords/supervised/<string:extr_type>
Port	10602
Technology	Docker

Table 24. Personal Recommendations for Stakeholder's Engine – MS keywords-supervised-<string:extr_type>

The service extracts DBpedia entities and a list of custom entities (optional) defined by the user.
It also allows the user to specify the confidence level for entity spotting and classification in the input.

5.2.10.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.10.3 API Documentation

In the following table the available HTTP requests is extensively reported. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/keywords/supervised/<string:extr_type>
Method	POST
URL parameters	Language model alias for entity linking, e.g. 'en' for English
Request example	{"document": "u'Railway line Dugo Selo – Križevci is a constituent part of the branch Vb, Paneuropean corridor passing through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo. The existing railway line has one track, large inter-station distances and is already now of limited transportation and maximum railway capacity with no possibilities of capacity enhancing.\n", "confidence": 0.25}
Response example	{"content" : {"dbpediaEntities" : ["Rail_transport", "Dugo_Selo", "Constituent_country", "Visual_Basic", "Pan-European_identity", "Territory_(animal)", "Republic_of_Ireland", "Croatia", "M201_railway_(Croatia)", "States_and_territories_of_India", "Border", "Koprivnica", "Dugo_Selo", "Joseph_Stalin", "Existence", "Megafauna", "Interdisciplinarity", "Train_station","Term_limit", "Transport", "Maxima_and_minima", "Headway", "Headway", "Enhancer_(genetics)"], "dbpediaUri": ["http://dbpedia.org/resource/Rail_transport", "http://dbpedia.org/resource/Dugo_Selo","http://dbpedia.org/resource/Visual_Basic", "http://dbpedia.org/resource/Territory_(animal)", "http://dbpedia.org/resource/Croatia", "http://dbpedia.org/resource/Border", "http://dbpedia.org/resource/Koprivnica", "http://dbpedia.org/resource/Dugo_Selo", "http://dbpedia.org/resource/Joseph_Stalin","http://dbpedia.org/resource/Existence", "http://dbpedia.org/resource/Megafauna", "http://dbpedia.org/resource/Interdisciplinarity", "http://dbpedia.org/resource/Train_station", "http://dbpedia.org/resource/Term_limit", "http://dbpedia.org/resource/Transport", "http://dbpedia.org/resource/Headway","http://dbpedia.org/resource/Headway", "http://dbpedia.org/resource/Republic_of_Ireland", "http://dbpedia.org/resource/Constituent_country", "http://dbpedia.org/resource/Enhancer_(genetics)", "http://dbpedia.org/resource/Maxima_and_minima", "http://dbpedia.org/resource/Pan-European_identity", "http://dbpedia.org/resource/M201_railway_(Croatia)", "http://dbpedia.org/resource/States_and_territories_of_India"] "normalizedDocument" : "u Rail_transport line Dugo_Selo Kri evci is a Constituent_country part of the branch Visual_Basic Pan-European_identity corridor passing

	through the Territory_(animal) of the Republic_of_Ireland of Croatia and the railway line M201_railway_(Croatia) States_and_territories_of_India Border Koprivnica Dugo_Selo Joseph_Stalin Existence railway line has one track Megafauna Interdisciplinarity Train_station distances and is already now of Term_limit Transport and Maxima_and_minima railway Headway with no possibilities of Headway Enhancer_(genetics)", "patternMatching" : [] }, "error": {"status": 600, "code": "OK", "description": "Application run normally."}}

5.2.10.4 Internal storage

None

5.2.11 parsing-types

5.2.11.1 Overview

Name	parsing-types
Developer	ENG
Dependencies	None
Base path	/api_t_33/parsing/types
Port	10602
Technology	Docker

Table 25. Personal Recommendations for Stakeholder's Engine – MS parsing-types

This microservice takes the list of DBpedia entities, creates a SparQL query, and finds the list of DBpedia types that are related to each entity. It then returns the list of DBPedia types.

5.2.11.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.11.3 API Documentation

The available HTTP request is described in detail in the following table. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/parsing/types
Method	POST
URL parameters	None
Request example	{'listOfEntities':[u'Koprivnica', u'Distances']}
Response example	{u'content': {u'typeDict': {u'Koprivnica': [u'http://dbpedia.org/resource/Category:Cities_and_towns_in_Croatia', u'http://dbpedia.org/resource/Category:1272_establishments_in_Europe', u'http://dbpedia.org/resource/Category:Bjelovar-Kri\u017eevci_County', u'http://dbpedia.org/resource/Category:Populated_places_established_in_the_13th_century', u'http://dbpedia.org/resource/Category:Populated_places_in_Koprivnica-

	Kri\u017eevci_County', u'http://dbpedia.org/resource/Category:Koprivnica', u'http://dbpedia.org/resource/Category:13th-century_establishments_in_Croatia'], u'Distances': []}}, u'error': {u'status': 600, u'code': u'OK', u'description': u'Application run normally.'}}

5.2.11.4 Internal storage

None

5.2.12 parsing-features

5.2.12.1 Overview

Name	parsing-features
Developer	ENG
Dependencies	None
Base path	/api_t_33/parsing/features
Port	10602
Technology	Docker

Table 26. Personal Recommendations for Stakeholder’s Engine – MS parsing-features

The service takes a paragraph as input, searches for words and expressions in text that can help to identify a requirement, then calculates readability scores (Flesch Ease and Kincaid metrics) and returns a structured dictionary as output.

5.2.12.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.12.3 API Documentation

The following table reports in detail the available HTTP request. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/parsing/features
Method	POST
URL parameters	None
Request example	{“document”: ”u'Railway line Dugo Selo – Križevci is a constituent part of the branch Vb, Paneuropean corridor passing through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo. The existing railway line has one track, large inter-station distances and is already now of limited transportation and maximum railway capacity with no possibilities of capacity enhancing.\n”}
Response example	{u'content': {u'featuresDict': {u'will be': 0, u'is planned': 0, u'is envisaged': 0, u"it's planned": 0, u"it's foreseen": 0, u'is foreseen': 0, u'following': 0}}, u'error': {u'status': 600, u'code': u'OK', u'description': u'Application run normally.'}}

5.2.12.4 Internal storage

None

5.2.13 parsing-structure

5.2.13.1 Overview

Name	parsing-structure
Developer	ENG
Dependencies	None
Base path	/api_t_33/parsing/structure
Port	10602
Technology	Docker

Table 27. Personal Recommendations for Stakeholder's Engine – MS parsing-structure

This microservice takes a text as input, analyzes its syntactic structure and returns a text split into sentence, where each sentence is structured as subject - verb - object - subordinates. The service also analyzes subordinate clauses and connects them to their respective main sentence.

5.2.13.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.13.3 API Documentation

In the following table the available HTTP request is comprehensively described. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/parsing/structure
Method	POST
URL parameters	None
Request example	{“document”: ”u'Railway line Dugo Selo – Križevci is a constituent part of the branch Vb, Paneuropean corridor passing through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo. The existing railway line has one track, large inter-station distances and is already now of limited transportation and maximum railway capacity with no possibilities of capacity enhancing.\n”}
Response example	{‘content’ : {"entities": [{"subchild": [], "adjectives": [], "hash_number": "1248239241591158246", "objects_extended": ["one track , large inter - station distances"], "lemmatized_verb": "have", "objects": ["track"], "subjects": ["Križevci","line], "subjects_extended": ["Railway line Dugo Selo – Križevci","The existing railway line"], "dependence": [], "verbs": "has"}, {"subchild": [], "adjectives": [], "hash_number":"3411606890003347522", "objects_extended": [], "lemmatized_verb": "be", "objects": [], "subjects": [], "subjects_extended": [],"dependence": "1248239241591158246", "verbs": "is" }]},

	u'error': {u'status': 600, u'code': u'OK', u'description': u'Application run normally.'}}

5.2.13.4 Internal storage

None

5.2.14 parsing-metrics

5.2.14.1 Overview

Name	parsing-metrics
Developer	ENG
Dependencies	None
Base path	/api_t_33/parsing/metrics
Port	10602
Technology	Docker

Table 28. Personal Recommendations for Stakeholder's Engine – MS parsing-metrics

This microservice calculates readability metrics (Flesch Reading Ease and Kincaid) on the input text and returns a dictionary with text and respective metrics scores.

5.2.14.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/ENG/prs-improving-requirements-quality-features.git

5.2.14.3 API Documentation

The available HTTP request is extensively reported in the following table. The Swagger file is available at http://217.172.12.199:10602/apidocs/.

URL	http://217.172.12.199:10602/api_t_33/parsing/metrics
Method	POST
URL parameters	None
Request example	{"document": "u'Railway line Dugo Selo – Križevci is a constituent part of the branch Vb, Paneuropean corridor passing through the territory of the Republic of Croatia and the railway line M201 State border – Koprivnica – Dugo Selo. The existing railway line has one track, large inter-station distances and is already now of limited transportation and maximum railway capacity with no possibilities of capacity enhancing.\n"}
Response example	{u'content': {u'dictMetrics': {u'ease': 15.14, u'kincaid': 18.7}}, u'error': {u'status': 600, u'code': u'OK', u'description': u'Application run normally.'}}

5.2.14.4 Internal storage

None

5.2.15 requirements-classifier

5.2.15.1 Overview

Name	requirements-classifier
developer	UPC
Dependencies	None
Base path	/upc/classifier-component/
Port	9402, 9403
Technology	SpringBoot, Mahout and Hadoop

Table 29. Personal Recommendations for Stakeholder's Engine – MS requirements-classifier

This microservice recommends similar requirements by analyzing requirements from past projects. Four functionalities are available in this microservice:

- *Train*: this request analyzes the requirements received and creates a model for the classifier of one company.
- *Classify*: this request receives requirements and using the model created in the *Train* step returns a recommendation for every requirement along with its confidence value.
- *Update*: this request updates the model with the requirements it receives as parameter.
- *Train&test*: this request returns the result of k cross-validation using the requirements received in the request. Splits the requirements in k groups, trains a classifier for each group with all of the requirements received except the ones in the test group. It then calculates scores, such as accuracy, using the requirements in the test group. Returns the average scores of the model.

5.2.15.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/upc/requirements-classifier

5.2.15.3 API Documentation

The available HTTP requests are described in detail in the following tables.

Train

URL	http://217.172.12.199:9402/upc/classifier-component/train?company={company]&property={property}
Method	POST
URL parameters	company: Company name to which the requirements belong to. property: Property of the classifier (e.g. isReq).
Request example	{ "requirements" :[{ "id": "4523", "property_value": "Prose", "text" : "Reports and correspondence"

	},
	{
	"id": "1039",
	"property_value": "DEF",
	"text" : "Sufficient number of parking spaces must be provided close to the office."
	}]
	}

Response	None

Classify

URL	http://217.172.12.199:9402/upc/classifier-component/classify?company={company]&property={property}
Method	POST
URL parameters	company: Company name that is doing the classification. property: Property of the classifier (e.g. isReq）.

Request example	{
	"requirements" :[
	{
	"id": "5432",
	"text" : "Construction Site"
	},
	{
	"id": "10030",
	"text" : "Contractor must arrange the Construction Site in accordance with Articles 133 and 134 of the Building Act and Commission Regulation (EC)"
	}]
	}

Response example	{
	"recommendations": [
	{
	"requirement": “5432”,
	"property_value": "Prose",
	"confidence": 93.44250563767345
	},
	{
	"requirement": “10030”,
	"property_value": "DEF",
	"confidence": 93.79632918450244
	}
	]
	}

Update

URL	http://217.172.12.199:9402/upc/classifier-component/update?company={company]&property={property}
Method	POST
URL parameters	company: Company name that is doing the classification. property: Property of the classifier (e.g. isReq).

Request example	{
	"requirements" :[
	{
	"id": "4523",
	"property_value": "Prose",
	"text" : "Reports and correspondence"
	},
	{
	"id": "1039",
	"property_value": "DEF",
	"text" : "Sufficient number of parking spaces must be provided close to the office."
	}]
	}

Response	None

Train & test

URL	http://217.172.12.199:9404/upc/similarity-detection/DB/AddReqs?k={number_of_tests}
Method	POST
URL parameters	number_of_tests: Number of tests that will be used to do the cross-validation.

Request example	{
	“requirements" :[
	{
	"id": "4523",
	"property_value": "Prose",
	"text" : "Reports and correspondence"
	},
	{
	"id": "100147",
	"property_value": "DEF",
	"text" : " Contractor shall submit Progress Reports"
	},
	{
	"id": "1004",
	"property_value": "Prose",
	"text" : "STATIC SWITCHING"
	},
	{
	"id": "1039",
	"property_value": "DEF",
	"text" : "Sufficient number of parking spaces must be provided close to the office."
	}]
	}

Response example	{
	"kappa": 0.3062,
	"accuracy": 80.05743333333332,
	"reliability": 43.948283333333336,
	"reliability_std_deviation": 0.45151666666666673,
	"weighted_precision": 0.80275,
	"weighted_recall": 0.8005666666666666,
	"weighted_f1_score": 0.8007833333333334
	}

5.2.15.4 Internal storage

SQLite database storing the training of the classifier including model, dictionary, label index, and frequencies documents for each company and property. These files are needed to classify the requirements.

5.2.16 issue-prioritizer

5.2.16.1 Overview

Name	most discussed bugs of the month
Partner	vogella
Dependencies	None
Base path	/
Port	9801
Technology	SpringBoot

Table 30. Personal Recommendations for Stakeholder's Engine – issue-prioritizer

The most discussed bugs of the month endpoint reads the issues in bugzilla, which have the most comments on a monthly basis. This is a great indicator concerning the relevance of certain issues. It helps committers of a Eclipse project to realize what are the important issues discussed by the community.
The eclipse plug-ins provided by Vogella make use of this rest endpoint to visualize the most discussed issues.

5.2.16.2 Source Code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/vogella/issue-prioritizer

5.2.16.3 API Documentation

The available HTTP request is outlined below.

URL	http://217.172.12.199:9801/mostDiscussedBugsOfTheMonth
METHOD	GET
URL parameters	product: collection of products in bugzilla, e.g., platform component: collection of components in bugzilla, e.g., ui
Response example	[{ "id": 535807, "resolution": "", "status": "NEW", "creator": "juergen@jwi.de", "summary": "Always open editor in second screen (monitor)", "platform": "PC", "product": "Platform", "component": "UI", "severity": "enhancement", "priority": "P3",

```
    "version": "4.8",
    "creationTime": "2018-06-12T12:05:50.000+0000",
    "cc": [],
    "blocks": [],
    "keywords": [],
    "seeAlso": [],
    "comments": [
      {
        "time": "2018-06-12T12:05:50.000+0000",
        "text": "I have two physical displays...",
        "bugId": 0,
        "count": 0,
        "attachmentId": 0,
        "tags": [],
        "creator": "juergen@jwi.de",
        "creationTime": null,
        "id": 2956071,
        "markdown": false,
        "private": false
      }
    ],
    "attachments": [],
    "userPriority": 0,
    "open": false
  }]
```

5.2.16.4 Internal storage

This Microservice uses MongoDB to store the data.

5.2.17 issue-prioritizer-api

5.2.17.1 Overview

Name	issue-prioritizer-api
Partner	TU Graz
Dependencies	None
Base path	/prioritizer
Port	9002
Technology	Docker

Table 31. Personal Recommendations for Stakeholder's Engine – MS issue-prioritizer-api

5.2.17.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/tug/issue-prioritizer-api

5.2.17.3 API Documentation

In the following tables the main HTTP requests are described in detail. The Swagger file is available at http://217.172.12.199:9002/swagger-ui.html.

Retrieve a list of personalized recommendations of issues/requirements.

URL	http://217.172.12.199:9002/prioritizer/compute
METHOD	POST

URL parameters	None
Body parameters	assignee: email address of the developer/stakeholder products: JSON array containing the products components: JSON array containing the components of products

Request example	{
	"assignee":"register.eclipse@brychcy.de",
	"products": ["Platform", "JDT"],
	"components": ["IDE", "Core"]
	}

Response example	{
	"rankedBugs": [
	{
	"summary": "[projection] Custom regions in code folding",
	"product": "JDT",
	"component": "Core",
	"id": 63808,
	"priority": 12.43
	},
	{
	"summary": "[user library] JavaCore should provide API methods to get/store user libraries information",
	"product": "JDT",
	"component": "Core",
	"id": 99208,
	"priority": 9.6
	},
	// ...
	{
	"summary": "[Markers] Filter on resource name",
	"product": "Platform",
	"component": "IDE",
	"id": 220531,
	"priority": 0
	}
	],
	"errorMessage": null,
	"error": false
	}

Retrieve a link to show a pie/donut chart describing important keywords of a developer/stakeholder.

URL	http://217.172.12.199:9002/prioritizer/chart
METHOD	POST
URL parameters	None

Body parameters	{
	"assignee":"register.eclipse@brychcy.de",
	"products": ["Platform", "JDT"],
	"components": ["IDE", "Core"]
	}

Request example	{
	"assignee":"register.eclipse@brychcy.de",
	"products": ["Platform", "JDT"],
	"components": ["IDE", "Core"]
	}

Response example	{
	"errorMessage": null,
	"error": false,
	"url": "http://217.172.12.199:9002/prioritizer/chart/c/33rNMLt9"
	}

5.2.17.4 Internal storage

Internal storage is used for caching purposes only (MySQL database).

5.2.18 similar-related-requirements-recommender

5.2.18.1 Overview

Name	similar-related-requirements-recommender
Partner	TU Graz
Dependencies	None
Base path	/
Port	9006
Technology	Docker

Table 32. Recommendations about similar / related requirements – MS similar-related-requirements-recommender

5.2.18.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/tug/similar-related-requirements-recommender

5.2.18.3 API Documentation

In the following tables the main HTTP requests are described in detail.

Retrieve a list of requirements that are similar / related to a given requirement.

URL	http://217.172.12.199:9006/v1/compute
METHOD	POST
URL parameters	None
Body parameters	id: identifier of the requirement title: title/name of the requirement description: textual description of the requirement
Request example	[{ "id": 1, "title": "Evaluation Software", "description": "For evaluating the recorded training data, an evaluation software is required. This software requires the connection and the access to the clock's internal memory. The evaluation should contain measured information regarding the distance, the height, the average heart rate, and the calorie consumption."

	},
	 {
	 "id": 2,
	 "title": "Distance Measurement",
	 "description": "For statistical purposes, a distance measurement is necessary which needs data from a GPS sensor. This data is needed for the evaluation software and therefore stored in memory."
	 },
	 {
	 "id": 3,
	 "title": "GPS",
	 "description": "To capture position data, a GPS sensor should be used. Through the measured position and time information, the speed and the distance can be measured."
	 },
	 {
	 "id": 4,
	 "title": "Speed Measurement",
	 "description": "As evaluation after a workout, the average speed must be shown. The following statistics should be displayed: average speed and maximum speed. For measuring the average and maximum speed, time and distance have to be measured, and a storage unit for storing the data is necessary."
	 },
	 {
	 "id": 5,
	 "title": "Ideal BMI",
	 "description": "Based on the data on height, weight, body fat, age and gender, the watch should be able to calculate the ideal BMI for a user."
	 },
	 {
	 "id": 6,
	 "title": "Infrared",
	 "description": "In order to be able to connect the watch with a computer, WLAN, Bluetooth, and infrared modules must be available."
	 },
	 {
	 "id": 7,
	 "title": "Data-Storage Function",
	 "description": "For evaluation purposes, the data should be stored in the internal memory. The memory is used for saving the measured information such as the distance, the height, the average heart rate, and the calorie consumption. The stored data in the memory will then be used by the evaluation software."
	 },
	 {
	 "id": 8,
	 "title": "Time Measurement",
	 "description": "The clock must have an internal timer which is used for saving the current time and the measured time during a workout. This time information is then stored in memory and used for further evaluation of the data."
	 }
	]

Response example	[{ "id": 1, "predictions": [7], "title": "evaluation software" }, { "id": 2, "predictions": [7], "title": "distance measurement" }, { "id": 3, "predictions": [], "title": "gps" }, { "id": 4, "predictions": [],

	"title": "speed measurement"
	},
	{
	"id": 5,
	"predictions": [],
	"title": "ideal bmi"
	},
	{
	"id": 6,
	"predictions": [],
	"title": "infrared"
	},
	{
	"id": 7,
	"predictions": [1, 2],
	"title": "data-storage function"
	},
	{
	"id": 8,
	"predictions": [7],
	"title": "time measurement"
	}
	]

5.2.18.4 Internal storage

This service does not store anything – the calculation will be performed each time the service is called.

6 Group Decision Engine

This section presents and overview of the Group Decision Engine as well as a description of the available APIs

6.1 Overview

The Group Decision Engine offers support for groups of users in group decision processes. The operational details of the microservice currently available in the OpenReq infrastructure are reported in the following section.

6.2 Microservices and APIs

This section reports the specifics of the microservice composing the Group Decision Engine

6.2.1 openreq-live

6.2.1.1 Overview

Name	openreq-live
Partner	TU Graz
Dependencies	None
Base path	/
Port	9001
Technology	Maven

Table 33. Group Decision Engine – MS openreq-live

6.2.1.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/tug/openreq-live

6.2.1.3 API Documentation

In the following tables the main HTTP requests are described in detail.

Create a new project in openreq-live.

URL	http://217.172.12.199:9001/api/v1/project/create
METHOD	POST
URL parameters	None
Body parameters	title: title/name of the project description: textual description of the project

Request example	{
	"title": "My first project",
	"description": "Some description."
	}

Response example	{
	"error": false,
	"uniqueKey": "7wCwcHJh"
	}

Retrieve all unassigned requirements of a project in openreq-live.

URL	http://217.172.12.199:9001/api/v1/project/{uniqueKey}/unassigned
METHOD	GET
URL parameters	uniqueKey: public identifier of the project

Response example	{
	"requirements": [{
	„id“: 101,
	„projectSpecificRequirementId“: 1,
	„title“: „My first requirement“,
	„description”: “Some description”,
	“status”: “NEW”
	}, ...]
	}

Create a new requirement in openreq-live.

URL	http://217.172.12.199:9001/api/v1/requirement/create
METHOD	POST
URL parameters	None
Body parameters	title: title/name of the requirement description: textual description of the requirement projectUniqueKey: public identifier of the project status: initial status of the requirement

Request example	{
	"title": "My first requirement",
	"description": "Some description",
	"projectUniqueKey": "7wCwcHJh",
	"status": "NEW"
	}

Response example	{
	"error": false,
	"errorMessage": null
	}

6.2.1.4 Internal storage

This service uses a MySQL database to store project data.

6.2.2 gds-edemocracy

6.2.2.1 Overview

Name	gds-edemocracy
Developer	HITeC
Dependencies	None
Base path	/
Port	9750
Technology	Docker, Exlir & Phoenix

Table 34. Group Decision Engine – gds-edemocracy

This microservice exposes several metrics for a voting system based on delegative democracy (i.e., liquid democracy). The system allows a group of stakeholders to vote for a set of requirements (e.g., to be included/excluded), or decide to vote for another stakeholders to take up the decision

6.2.2.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/hitec/gds-edemocracy

6.2.2.3 API Documentation

The following tables list the main HTTP requests. The complete list is available at http://217.172.12.199:9750/api/swagger/index.html

Retrieve the requirements to be voted for a project.

URL	http://217.172.12.199:9750/api/projects/{project_id}/tickets
METHOD	GET
URL parameters	`project_id`: The ID of a project
Example	Request URL: http://217.172.12.199:9750/api/projects/1 Response: { "title": "OpenReq", "tickets": [{ "url": null, "title": "Dashboard for requirements analytics", "id": 56, "external_id": 385, "description": "Visualize the result of the requirements analytics on a configurable dashboard" }, { "url": null, "title": "Integrate with marketing tools", "id": 58,

	"external_id": 386,
	"description": "The results of the requirements analytics shall be integrated into upstream tools for marketing"
	},
	{
	"url": null,
	"title": "Analyze customer tickets system",
	"id": 60,
	"external_id": 384,
	"description": "Anoymized custumer tickets shall be analyzed for predicting escalation"
	}
	],
	"phase_end_at": "2018-07-17T08:13:17.076229",
	"phase_candidates_at": "2018-07-10T08:13:17.076215"
	}

Retrieve the participants in a project vote.

URL	http://217.172.12.199:9750/api/projects/{project_id}/participation
METHOD	GET
URL parameters	project_id: The ID of a project
Example	Request URL: http://217.172.12.199:9750/api/projects/1/participation Response: [{ "user_id": 1, "role": "candidate", "project_id": 1, "candidate_summary": "I'm very passionate about Javascript.\n" }, { "user_id": 2, "role": "candidate", "project_id": 1, "candidate_summary": "My main focus in the development of the Javascript language is a sane type system.\nEven though being a dynamic language shouldn't stop Javascript from helping the user prevent common mistakes.\n" }, { "user_id": 3, "role": "candidate", "project_id": 1, "candidate_summary": "As a Javascript poweruser, productivity is all that matters.\nIt was fine for years until the influx of bootcamp graduates decided that JS hat to arrive in the twenty-first century.\nIf your proposal fixes a bug, fine. We don't have to change the very core of the language just because we can.\n" }, { "user_id": 4, "role": "candidate", "project_id": 1, "candidate_summary": "15 Years Google Developer, Mountain View\n" }, { "user_id": 5, "role": "candidate", "project_id": 1, "candidate_summary": "In this age of web apps, javascript has become a de facto language for user interfaces. From this vantage point, it has to compete with the likes of Swift et al.\nOnce you start writing both languages, it becomes apparent that javascript feels inferior. Its main disadvantage being the old and confusing prototypal inheritance system.\n\nI'm in favour

	of abolishing the prototype system and implementing a modern OOP approach from the bottom up.\n"
	},
	{
	"user_id": 7,
	"role": "user",
	"project_id": 1
	},
	{
	"user_id": 8,
	"role": "candidate",
	"project_id": 1,
	"candidate_summary": "As a Javascript poweruser, productivity is all that matters.\nIt was fine for years until the influx of bootcamp graduates decided that JS hat to arrive in the twenty-first century.\nIf your proposal fixes a bug, fine. We don't have to change the very core of the language just because we can.\n"
	}
	]

Add a participant to a voting.

A participant can take three roles: “user” (i.e., can vote for candidates but not for requirements), “candidate” (i.e., voted by users, can vote for requirements), “guest” (i.e., cannot vote but just observe the voting process).

URL	http://217.172.12.199:9750/api/projects/{project_id}/participations/current
METHOD	POST
URL parameters	project_id: The ID of a project
Body parameters	user_id: the participant id joining a project project_id: the project the participant is joining role: the participant role when joining a project

Example	Request URL: http://217.172.12.199:9750/api/projects/1/participations/current
	
	Response:
	{
	"user_id": 8,
	"role": "candidate",
	"project_id": 1
	}

Add a vote for a candidate (if logged in as “user”) or to a requirement (if logged in as a “candidate”) in the current voting phase.

URL	http://217.172.12.199:9750//api/projects/{project_id}/participations/current/votes
METHOD	PUT
URL parameters	project_id: The ID of a project
Body parameters	vote: a reference to candidate in the project by id or to a requirement in the project by id

Example	Request URL: http://217.172.12.199:9750/api/projects/1/participations/current/votes Response: {"votes": [1]}

6.2.2.4 Internal storage

The internal storage is deployed together with the gds-edemocracy microservice in a PostgreSQL database using docker compose.

7 Dependency Engine

This section presents and overview of the Dependency Engine as well as a description of the available APIs.

7.1 Overview

Dependency engine extracts and manages the dependencies of requirements. New dependencies are extracted using natural language processing technologies. A set of requirements are sent to a microservice that processes the requirements and writes the newly found dependencies as a "proposed" status. The existing dependencies are managed by constructing a declarative representation of the requirements and their dependencies that allows inspection of the requirements as a transitive closure of all dependency network. The architecture of this engine and the provided services are extensively described in [4]. The operational details of the microservices currently available in the OpenReq infrastructure are reported in the following section.

7.2 Microservices and APIs

This section reports the specifics of the 5 microservices composing the Dependency Engine.

7.2.1 mulperi

7.2.1.1 Overview

Name	Mulperi
Developer	UH
Dependencies	Requires SpringCaaS
Base path	/
Port	9202
Technology	SpringBoot

Table 35. Dependency Engine – mulperi

Mulperi is a pipe-and-filter style relay service that receives the requirements in certain format and uses the AI engine of SpringCaaS for the dependency inferences..

7.2.1.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/uh/mulperi.git

7.2.1.3 API Documentation

The following tables describe in detail the available HTTP requests. The Swagger file is available at http://217.172.12.199:9202/swagger-ui.html.

Requirements model upload and consistency check

URL	http://217.172.12.199:9202/uploadDataAndCheckForConsistency
METHOD	POST
URL parameters	OpenReq JSON

Request example	{
	"project": {
	"id": "ABC",
	"name": "ABCExample",
	"specificRequirements": ["A", "B", "C"]
	},
	"requirements": [
	{
	"id": "A",
	"name": "a",
	"effort": 2
	},
	{
	"id": "B",
	"name": "b",
	"effort": 1
	},
	{
	"id": "C",
	"name": "c",
	"effort": 3
	}
	],
	"releases": [
	{
	"id": 1,
	"capacity": 10,
	"status": "new",
	"requirements": ["A"]
	},
	{
	"id": 2,
	"capacity": 10,
	"status": "new",
	"requirements": ["B", "C"]
	}
	],
	"dependencies": [
	{
	"dependency_type": "requires",
	"from": "B",
	"to": "C"
	}
	]
	}[4]

Response example	{
	"response": {
	"consistent": true
	}
	}

Requirements model upload, and consistency check and diagnosis

URL	http://217.172.12.199:9202/uploadDataCheckForConsistencyAndDoDiagnosis
METHOD	POST.
URL parameters	OpenReq JSON

Request example	{
	"project": {
	"id": "ABC",
	"name": "ABCExample",
	"specificRequirements": ["A", "B", "C"]
	},
	"requirements": [
	{
	"id": "A",
	"name": "a",
	"effort": 2
	},
	{
	"id": "B",
	"name": "b",
	"effort": 1
	},
	{
	"id": "C",
	"name": "c",
	"effort": 3
	}
	],
	"releases": [
	{
	"id": 1,
	"capacity": 5,
	"status": "new",
	"requirements": ["A", "B", "C"]
	}
	],
	"dependencies": [
	{
	"dependency_type": "requires",
	"from": "B",
	"to": "C"
	}
	]
	}

Response example	{
	"response": {
	"consistent": false,
	"diagnosis": [
	[
	{
	"requirement": "B"
	}
	]
	]
	}
	}

Requirements model upload, and consistency check, diagnosis and repair proposal(s)

URL	http://217.172.12.199:9202/uploadDataCheckForConsistencyDoDiagnosisAndProposeRepair
METHOD	POST
URL parameters	OpenReq JSON

Request example	{
	"project": {
	"id": "ABC",
	"name": "ABCExample",
	"specificRequirements": ["A", "B", "C"]
	},
	"requirements": [
	{
	"id": "A",
	"name": "a",
	"effort": 2
	},
	{
	"id": "B",
	"name": "b",
	"effort": 1
	},
	{
	"id": "C",
	"name": "c",
	"effort": 3
	}
	],
	"releases": [
	{
	"id": 1,
	"capacity": 10,
	"status": "new",
	"requirements": ["A"]
	},
	{
	"id": 2,
	"capacity": 10,
	"status": "new",
	"requirements": ["B", "C"]
	}
	],
	"dependencies": [
	{
	"dependency_type": "requires",
	"from": "B",
	"to": "C"
	}
	]
	}

Response example	{
	"response": {
	"consistent": false,
	"diagnosis": [
	[
	{
	"requirement": "B"
	}
	]
	]
	"repairs": [
	{
	"reassignedRequirements": ["B"]
	}

	] } }

Transitive closure of consequences for a set of requirements

URL	http://217.172.12.199:9202/checkDependencies/<modelName>
METHOD	POST
URL parameters	Model name and requirement id Note that either of the three above methods must have been called beforehand to upload model.

Example	Input: QT (modelName), B (requirement id),
	
	{
	"requirement": {
	"id": "B",
	"status": "NEW",
	"created_at": 0
	},
	"dependent_requirements": [
	{
	"id": "C",
	"status": "NEW",
	"created_at": 0
	}
	],
	"dependencies": [
	{
	"id": "B_C_REQUIRES",
	"dependency_type": "REQUIRES",
	"from": "B",
	"to": "C",
	"created_at": 0
	}
	]
	}

7.2.2 springCaaS

7.2.2.1 Overview

Name	SpringCaaS
Developer	UH
Dependencies	None
Base path	/
Port	9201
Technology	SpringBoot

Table 36. Dependency Engine – springCaaS

The inference service utilizing Choco solver. SpringCaaS is a standalone service that is called from Mulperi but never accessed directly from any other service. Therefore, SpringCaaS is a required but private service in OpenReq infrastructure. For further details, see D5.2.

7.2.2.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/uh/springcaas.git

7.2.2.3 API Documentation

API is not meant for public access but access takes place through Mulperi.

7.2.2.4 Internal storage

SpringCaaS uses the local file system for persistent storage.

7.2.3 cross-reference-detection

7.2.3.1 Overview

Name	cross-reference-detection
Developer	UPC
Dependencies	None
Base path	/upc/cross-reference-detection
Port	9401
Technology	SpringBoot

Table 37. Dependency Engine – cross-reference-detection

This microservice identifies cross-reference expressions among the requirements. A cross-reference expression is a piece of text within a project which refers to related information elsewhere, in the same or different project or document. For example, for the following sentence “This decision [referring to the previous requirement] has been taken due to the requirements stated in section 5.1” two requests are available:

- JSON: in this case, the requirements are given to the microservice in an specific JSON exchange format specified by the consortium.
- file: in this case, the requirements are given to the microservice in an HTML document that contains not only the requirements, but also their organization in sections and subsections.

In both cases, two parameters *n* and *m* can be added, which state the indexes of start and end of the requirements that will be analysed for extracting cross-references.
Apart from SpringBoot, this component uses jsoup library to parse the HTML file and Swagger for its documentation.

7.2.3.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/upc/cross-reference-detection.git

7.2.3.3 API Documentation

In the following tables the available HTTP requests are described in detail. More information can be found in the Swagger file, available at http://217.172.12.199:9401/upc/cross-reference-detection/swagger-ui.html.

JSON

URL	http://217.172.12.199:9401/upc/cross-reference-detection/json/{projectId}
METHOD	POST
URL parameters	projectId: id of the project containing the requirements to analyze

Request example	{
	"project" : [
	{
	"id" : "Pr2",
	"name" : "Cross-reference",
	"specificRequirements" : ["150890", "162454"]
	}
	],
	"requirement" : [
	{
	"id": "150890",
	"issuenum": "500",
	"name": "Dialog not working",
	"text": "The dialog window does not work with more than one user."
	},
	{
	"id": "162454",
	"issuenum": "502",
	"name": "Part II: dialog not working (ref. BUG 500)",
	"text": "Following issue BUG 500, the dialog do not work for multiple users under Windows O.S."
	}
	]
	}

Response example	{
	"project" : [
	{
	"id" : "Pr2",
	"name" : "Cross-reference",
	"specificRequirements" : ["150890", "162454"]
	}
	],
	"requirement" : [
	{
	"id": "150890",
	"issuenum": "500",
	"name": "Dialog not working",
	"text": "The dialog window does not work with more than one user."
	},
	{
	"id": "162454",
	"issuenum": "502",
	"name": "Part II: dialog not working (ref. BUG 500)",
	"text": "Following issue BUG 500, the dialog do not work for multiple users under Windows O.S."
	}
	],
	“dependency”: [
	{
	"dependency_type": "cross_reference",

	"status": proposed", "from": "162454", "to": "154606" }] }

file

URL	http://217.172.12.199:9401/upc/cross-reference-detection/file
METHOD	POST
URL parameters	None
File body	<html> <head></head> <body> <h1><span>1 <span> </span>FIRST SECTION</span></h1> <h2><span>1.1<span> </span>FIRST SUBSECIION</span></h2> <p>The system must be tested [...]</p> <h2><span>1.2<span> </span>SECOND SUBSECTION</span></h2> <p>As it is mentioned in the previous subsection 1.1, the system has to be validated [...]</p> </body> </html>

Response example	{
	"dependency": [
	{
	"dependency_type": "cross_reference",
	"status": "proposed",
	"from": "23",
	"to": "100"
	},
	{
	"dependency_type": "external_cross_reference",
	"status": "proposed",
	"from": "100",
	"to": ""
	}
	],
	"requirement": [
	{
	"id": "23",
	"name": "23",
	"text": "structure is described in detail in point 3.2.1. of this book.",
	"sect": "1",
	"subsect": "1.1",
	"parag": "1",
	"subparag": null
	},
	{
	"id": "100",
	"name": "100",
	"text": "new cables should be placed on the left side of the railway (article 2 of the contract)",
	"sect": "3",
	"subsect": "3.2.1",
	"parag": "1",
	"subparag": a
	}
	
	],

	"project": [
	{
	"id": "file",
	"name": "example_tender1_confidential",
	"specificRequirements": ["23","100"]
	}
	]
	}

7.2.3.4 Internal storage

This microservice does not need any internal storage.

7.2.4 similarity-detection

7.2.4.1 Overview

Name	similarity-detection
Developer	UPC
Dependencies	Not apply
Base path	/upc/similarity-detection
Port	9404, 9405, 9406
Technology	SpringBoot

Table 38. Dependency Engine – similarity-detection

This microservice aims to identify similar requirements, i.e., requirements that specify the same constraint. On top of identifying similar requirements, the services return a similarity score between 0 and 1, where 1 represents the same constraint and 0 a completely different one.

Four requests are available in this microservice:

- ReqReq: this request calculates the similarity score of two texts.
- ReqProject: this request calculates the similarity score among a specific requirement and each one of the requirements of a project .
- Project: this request calculates the similarity score among all the requirements of a project.
- AddReqs: the microservice adds the necessary NLP pre-processing to an internal database (necessary to speed-up the response time of the ReqProject and Project requests)

In the first three requests, a parameter indicates the component that will be used to calculate the similarity scores (DKPro, Semilar, a weight among them, all of them). ReqProject and Project requests, have two additional parameters, the threshold (which establishes the minimum similarity score that the returned results should have) and num_elements (which establishes the maximum number of results that should be returned).

Apart from SpringBoot, this microservice embeds Semilar and DKPro components, which are used as a basis to calculate the similarity scores. SQLite is used for the internal database, and Swagger for its documentation.

7.2.4.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/upc/similarity-detection.git

7.2.4.3 API Documentation

In the following tables the available HTTP requests are described in detail. More information can be found in the Swagger file, available at http://217.172.12.199:9404/swagger-ui.html.

ReqReq

URL	http://217.172.12.199:9404/upc/similarity-detection/ReqReq?req1={id1}&req2={id2}&component={component}
METHOD	POST
URL parameters	req1: id of the requirement to compare. It must be already preprocessed. req2: id of the requirement to compare. It must be already preprocessed. component: enumeration that indicates the component that should be used for calculating the similarity score. • *DKPro* • *Semilar* • *AllWeighted* (returns the mean of the similarity scores of the other components) • *All* (returns all the similarity scores of all the components)
Request example	It follows the OpenReq JSON format

Response example	It follows the OpenReq JSON format
	Adds to “dependency” objects like:
	{
	"component" : "{component}",
	"dependency_type":"similar",
	"dependency_score":0.965,
	"fromid":"{id1}",
	"toid":"{id2}",
	"status":"proposed"
	}

ReqProject

URL	http://217.172.12.199:9404/upc/similarity-detection/ReqProject?req={idReq}&project={idProject}&component={component}&threshold=0.3&num_elements=25
METHOD	POST
URL parameters	req: id of the requirement to compare. It must be already preprocessed.

	project: id of the project to compare. It must be already preprocessed. component: enumeration that indicates the component that should be used for calculating the similarity score. • *DKPro* • *Semilar* • *AllWeighted* (returns the mean of the similarity scores of the other components) • *All* (returns for each pair of requirements the highest similarity scores of all the components). threshold: float between 0 and 1 that establishes the minimum similarity score that the returned results should have num_elements: maximum number of results that should be returned
Request example	It follows the OpenReq JSON format

Response example	It follows the OpenReq JSON format
	Adds to dependency objects like:
	{
	"component" : "{component}",
	"dependency_type":"similar",
	"dependency_score":0.965,
	"fromid":"{id1}",
	"toid":"{id2}",
	"status":"proposed"
	}

Project

URL	http://217.172.12.199:9404/upc/similarity-detection/Project?project={idProject}&component={component}&threshold=0.3&num_elements=25
METHOD	POST
URL parameters	project: id of the project to compare. It must be already preprocessed. component: enumeration that indicates the component that should be used for calculating the similarity score. • *DKPro* • *Semilar* • *AllWeighted* (returns the mean of the similarity scores of the other components) • *All* (returns for each pair of requirements the highest similarity scores of all the components). threshold: float between 0 and 1 that establishes the minimum similarity score that the returned results should have num_elements: maximum number of results that should be returned
Request example	It follows the OpenReq JSON format

Response example	It follows the OpenReq JSON format
	Adds to dependency objects like:
	{
	"component" : "{component}",
	"dependency_type":"similar",
	"dependency_score":0.965,
	"fromid":"{id1}",
	"toid":"{id2}",
	"status":"proposed"
	}

AddReqs

URL	http://217.172.12.199:9404/upc/similarity-detection/DB/AddReqs
METHOD	POST
URL parameters	None

Request example	[
	{
	"id": "1",
	"text": “The button should be blue.”
	},
	{
	"id": "2",
	"text": “The buttons shall be square.”
	},
	{
	"id": "3",
	"text": “The buttons should be of different colours.”
	},
	{
	"id": "4",
	"text": “The buttons shall be red.”
	}
	]

Response	None

7.2.4.4 Internal storage

This microservice stores in an SQLite database the NLP preprocessing needed for the Semilar component, so the service response is faster.

7.2.5 dependency-detection

7.2.5.1 Overview

Name	dependency-detection
Partner	UPC
Dependencies	Not apply
Base path	/upc/dependency-detection

Port	9407
Technology	SpringBoot

Table 39. Dependency Engine – dependency-detection

This microservice aims to identify dependencies among the requirements. Two requirements have a dependency when their concepts are implicitly related. These implicit relations are detailed in an input ontology file, where the connection between classes (concepts) shows their dependency type, i.e., damages, requires, refines, etc. One request is available in this microservice:

- *JSON/ontology*: in this case, the requirements are given to the microservice in a specific JSON exchange format specified by the consortium. The ontology is given to the microservice in an RDF/XML format file as a part of the body along with the JN.

In the request, the parameter *projectId* indicates the project in the input JSON containing the the requirements to analyse.

Apart from SpringBoot, this component uses the OpenNLP library to analyse the requirements, the Jena library to handle the input ontology file, the Jackson library to read and generate the input/output JSON, the Apache commons-io library to assist with developing IO functionality, and Swagger for its documentation

7.2.5.2 Source code

https://mast-tuleap.informatik.uni-hamburg.de/plugins/git/openreq/upc/dependency-detection

7.2.5.3 API Documentation

The available HTTP request is outlined below. More information can be found in the Swagger, available at http://217.172.12.199:9407/upc/dependency-detection/swagger-ui.html.

json/ontology

URL	http://217.172.12.199:9407/upc/dependency-detection/json/ontology/{projectId}
METHOD	POST
URL parameters	projectId: id of the project containing the requirements to analyze

Request example	{
	"project" : [
	{
	"id" : "Pr3",
	"name" : "dependency",
	"specificRequirements" : ["A", "B"]
	}
	],
	"requirement" : [
	{
	"id": "A",
	"name": "a",
	"text": "The system should use Mahout."
	},
	{

	"id": "B", "name": "b", "text": "The server should use Java." }] }
Ontology file	<?xml version="1.0"?> <rdf:RDF xmlns="http://www.semanticweb.org/openreq/demo#" xml:base="http://www.semanticweb.org/openreq/demo" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns:xml="http://www.w3.org/XML/1998/namespace" xmlns:xsd="http://www.w3.org/2001/XMLSchema#" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"> <owl:Ontology rdf:about="http://www.semanticweb.org/openreq/demo"/> <owl:ObjectProperty rdf:about="http://www.semanticweb.org/openreq/demo#requires"/> <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/openreq/demo#id"> </owl:DatatypeProperty> <owl:DatatypeProperty rdf:about="http://www.semanticweb.org/openreq/demo#requirement"> </owl:DatatypeProperty> <owl:Class rdf:about="http://www.semanticweb.org/openreq/demo#mahout"> <rdfs:subClassOf> <owl:Restriction> <owl:onProperty rdf:resource="http://www.semanticweb.org/openreq/demo#requires"/> <owl:someValuesFrom rdf:resource="http://www.semanticweb.org/openreq/demo#java"/> </owl:Restriction> </rdfs:subClassOf> </owl:Class> <owl:Class rdf:about="http://www.semanticweb.org/openreq/demo#java"> </owl:Class> </rdf:RDF>

Response example	{
	"project" : [{
	"id" : "Pr3",
	"name" : "dependency",
	"specificRequirements" : ["A", "B"]
	}],
	"requirement" : [
	{
	"id": "A",
	"name": "a",
	"text": "The system should use Mahout."
	},{
	"id": "B",
	"name": "b",
	"text": "The server should use Java."
	}],
	"dependency":[{
	"dependency_type": "requires",
	"from": "B",
	"to": "A",
	"status": "proposed"
	}]
	}

7.2.5.4 Internal storage

This microservice does not need any internal storage.

8 1ST INTEGRATED VERSION

This section presents the integrations of the component APIs (Section 3-7) in two scenarios, *OpenReq Live*, a web-based application, streamlines several RE tasks showcasing how the components work together. The *OpenReq Eclipse plugin* shows a first integration of the APIs in an established product. To complete the picture, a third scenario, *Jira plugin,* is briefly mentioned.

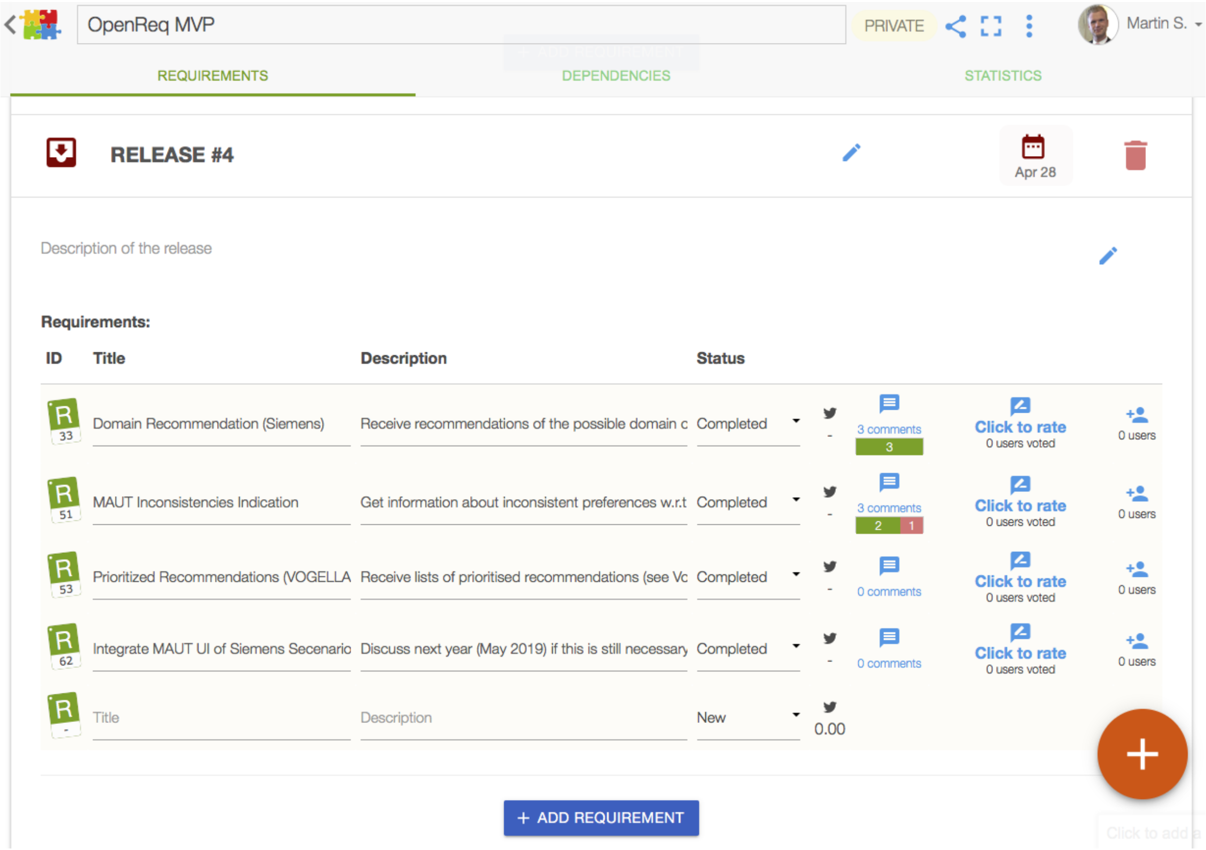
8.1 OpenReq Live

OpenReq Live is developed following a "minimal viable product" approach where the feedback of the OpenReq partners and other user communities accessed during OpenReq studies is directly considered when prioritizing the functionalities of future releases. The idea of following a "minimal viable product" approach was generated out of discussions with the advisory board in a plenary meeting of OpenReq in December 2017 in Vienna. A major focus of the User Interface (UI) design is to follow a responsive design style [7] which allows the application of OpenReq functionalities also in mobile environments.

OpenReq Live is based upon the latest web-technologies and will run on a Spring Boot application including the Thymeleaf as well as the Bootstrap framework. A major goal of "OpenReq Live" is to show OpenReq functionalities in an integrated fashion which goes beyond the application in the individual OpenReq trials. We want to emphasize that the UI elements presented in this deliverable represent a showcase of OpenReq which can serve as a reference design for the trial scenarios that are based on individual UI elements, for example, the Siemens trial UI is based on a Doors integration.

Similar as the "minimal viable products" also the designs included in this deliverable will be improved repeatedly depending on the feedback from trial partners, usability studies, and the feedback of OpenReq communities and users engaged in the integration of OpenReq functionalities in the "Open Call". OpenReq Live will also serve as a basis for different user studies [8] planned in the context of work package 4 (Group Decision Support – see deliverable D4.1 [5]).

In OpenReq Live, requirements can be *assigned (*via *drag & drop)* to specific releases, which are described by a name as well as the release date (the created releases are sorted by their release dates). Requirements not yet assigned to a specific release are available in a dedicated section "unassigned requirements" on top of the page. The following screenshot (Figure 6) illustrates a planned release #4 of the OpenReq Live which is part of the whole OpenReq Live release plan.



RELEASE #4

Apr 28

Description of the release

Requirements:

ID	Title	Description	Status				
33	Domain Recommendation (Siemens)	Receive recommendations of the possible domain c	Completed	-	3 comments 3	Click to rate 0 users voted	0 users
51	MAUT Inconsistencies Indication	Get information about inconsistent preferences w.r.t	Completed	-	3 comments 2 1	Click to rate 0 users voted	0 users
53	Prioritized Recommendations (VOGELLA	Receive lists of prioritised recommendations (see Vc	Completed	-	0 comments	Click to rate 0 users voted	0 users
62	Integrate MAUT UI of Siemens Secenario	Discuss next year (May 2019) if this is still necessary	Completed	-	0 comments	Click to rate 0 users voted	0 users
-	Title	Description	New	0.00			

+ ADD REQUIREMENT

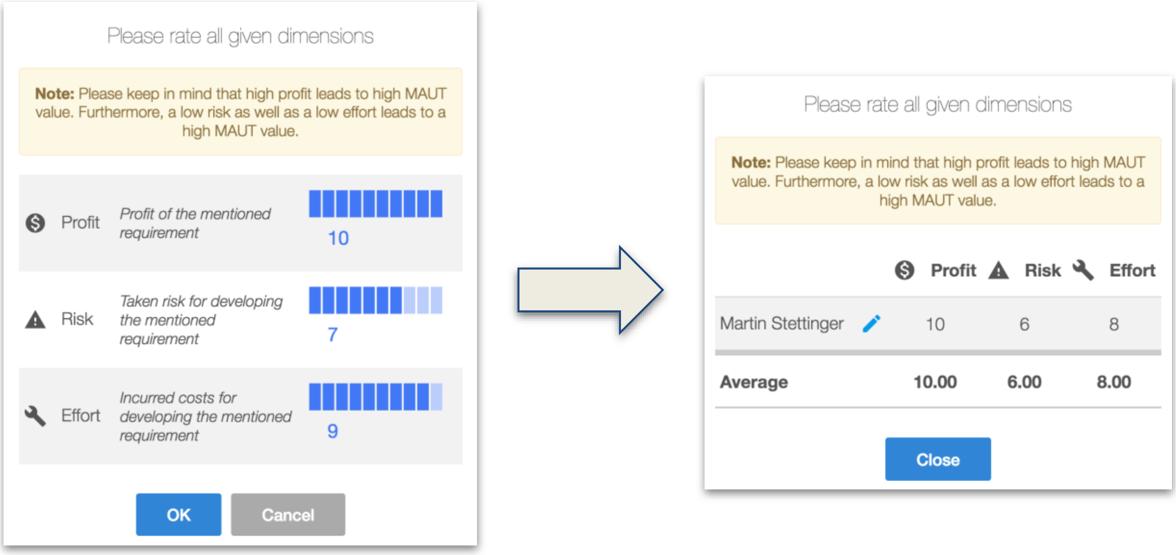
Click to add a

Figure 6: Fragment of the Release Plan for the OpenReq prototype.

Figure 6 shows all requirements assigned to release "Release #4" which is due to April, 28th 2018. Requirements are described by a *name*, a *descriptive text* as well as a *status*. Further information of requirements is a popularity value of Twitter[7] (shown below the Twitter bir d) which indicates how much discussion is going on in Twitter channels about the requirement (optional decision support value). The relevant Twitter channels can be defined in the project settings. Next to the Twitter bird there is room for discussions related to the requirement. Stakeholders can discuss a requirement as well as provide comments connected with sentiment values (comment pro, comment contra, neutral comment). These sentiment values support stakeholders when rating requirements. Multi attribute utility theory (MAUT [6]) is used to represent the preferences of stakeholders -- for example, prioritizations are based on the attributes of the requirements (e.g., profit, effort, and risk). To the right-side stakeholders can be assigned to requirements (recommendations for relevant stakeholders are provided).

Following figure (Figure 7) shows the preference acquisition interface for articulating personal preferences.

[7] www.twitter.com



Please rate all given dimensions

Note: Please keep in mind that high profit leads to high MAUT value. Furthermore, a low risk as well as a low effort leads to a high MAUT value.

Profit	*Profit of the mentioned requirement*	10
Risk	*Taken risk for developing the mentioned requirement*	7
Effort	*Incurred costs for developing the mentioned requirement*	9

Please rate all given dimensions

Note: Please keep in mind that high profit leads to high MAUT value. Furthermore, a low risk as well as a low effort leads to a high MAUT value.

	Profit	Risk	Effort
Martin Stettinger	10	6	8
Average	**10.00**	**6.00**	**8.00**

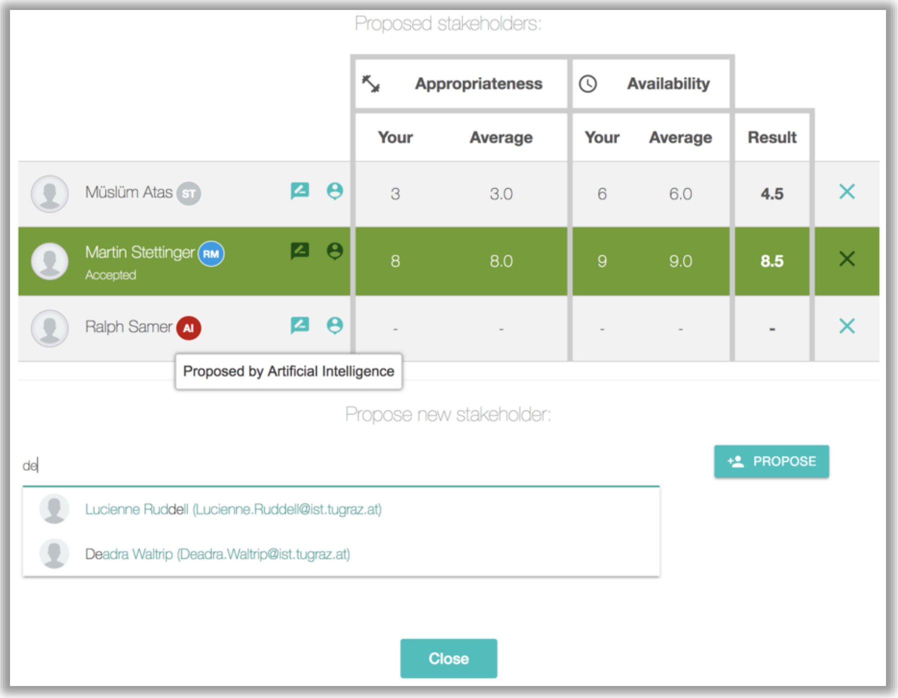
Figure 7: MAUT-based evaluation of the requirements with the dimensions profit, effort, and risk.

(The right-hand side appears after the specification of the individual preferences and depicts the overview of the given ratings of all stakeholders.)

To avoid biases during the preference specification, the individual preferences of the other stakeholders will be presented after the current user articulated his / her preferences (right screen).

The requirements can be assigned to specific stakeholders. Each of the project stakeholders as well as the requirements manager can propose assignments of stakeholders to the specific requirements. Additional stakeholders will be automatically recommended (indicated with “proposed by AI“). A green background of a stakeholder means that this stakeholder has already accepted the proposed assignment.

The next figure (Figure 8) presents the stakeholder assignment for a specific requirement.



Proposed stakeholders:

	Appropriateness		Availability		
	Your	Average	Your	Average	Result
Müslüm Atas ST	3	3.0	6	6.0	4.5
Martin Stettinger RM Accepted	8	8.0	9	9.0	8.5
Ralph Samer AI	-	-	-	-	-

Proposed by Artificial Intelligence

Propose new stakeholder:

de

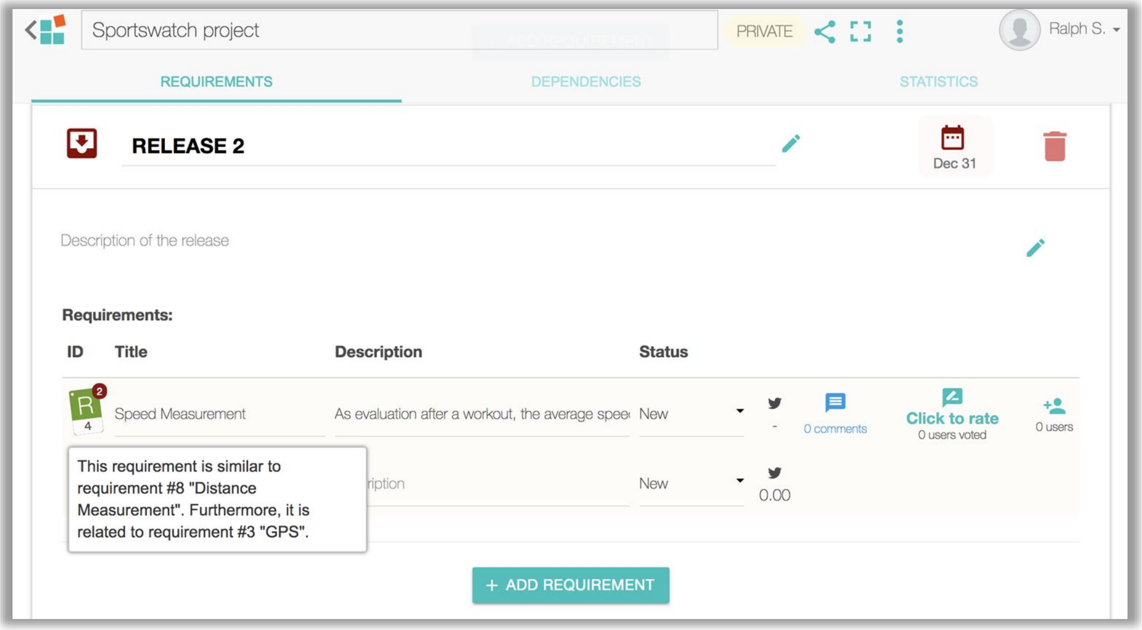
Lucienne Ruddell (Lucienne.Ruddell@ist.tugraz.at)

Deadra Waltrip (Deadra.Waltrip@ist.tugraz.at)

Figure 8: Assigned stakeholders for a specific requirement

(ST = proposed by stakeholder, RM = proposed by requirements manager, AI = proposed by artificial intelligence). A green line indicates that the specific stakeholder already accepted the proposed assignment.

In addition to this OpenReq Live also helps the users in avoiding duplicate / similar requirements as well as showing relations to other already available requirements by presenting an indicator of those requirements. Following screenshot (Figure 9) depicts such an indication.



Sportswatch project

RELEASE 2

Dec 31

Description of the release

Requirements:

ID	Title	Description	Status				
R 4	Speed Measurement	As evaluation after a workout, the average spee	New	-	0 comments	Click to rate 0 users voted	0 users
		ription	New	0.00			

This requirement is similar to requirement #8 "Distance Measurement". Furthermore, it is related to requirement #3 "GPS".

Figure 9: Indication of similar / related requirements.

Based on the developed recommendation algorithms OpenReq Live is able to present precise information about relationships to other requirements as well as shows the ability to avoid duplicate requirements by indicating requirements that talk semantically about the same content.

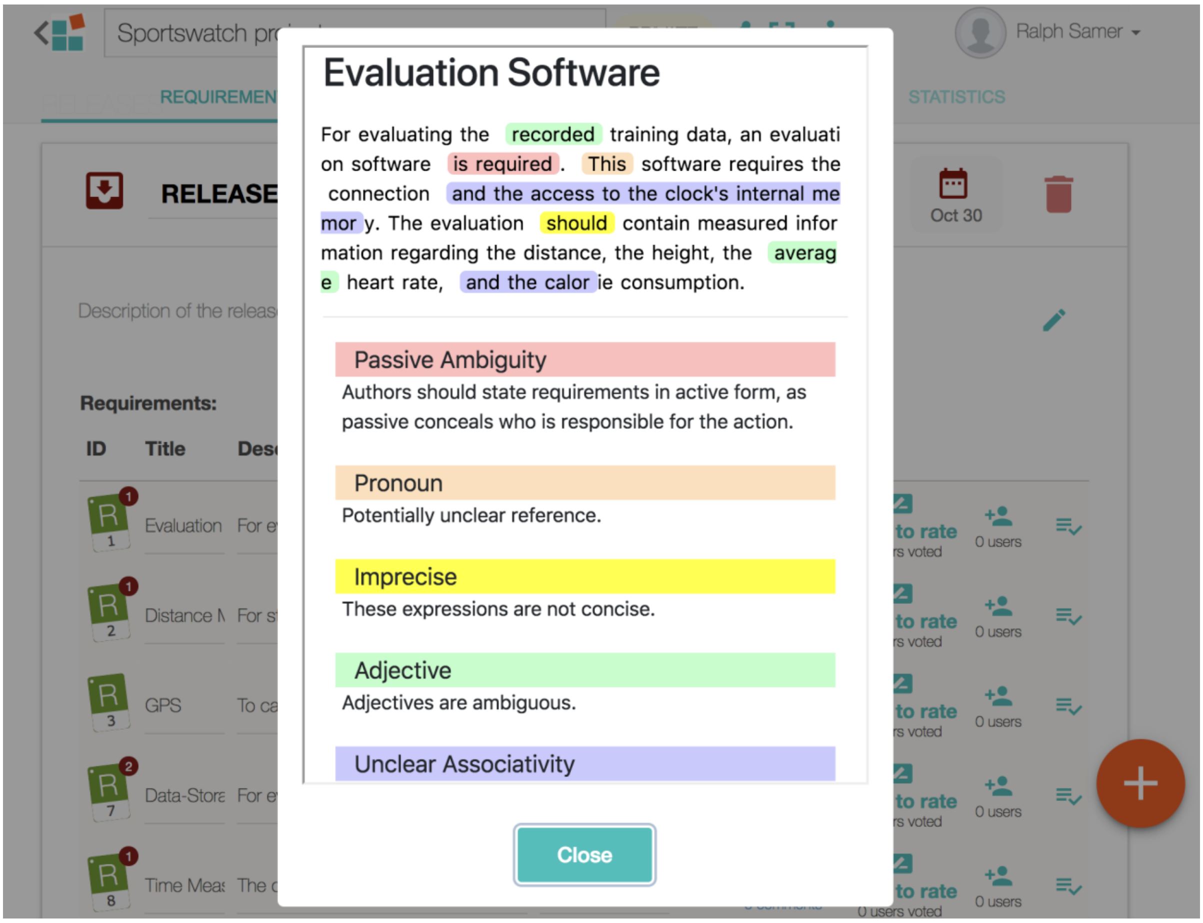
Individual trials in OpenReq will be based on different User Interface (UI) technologies based on the available technical infrastructure. For example, the Siemens trial is based on a IBM Doors integration and therefore restricted by the corresponding UI technology provided by this environment. Comparable UI restrictions exist for the QT and Vogella user interfaces (QT will develop a JIRA and Vogella an Eclipse plugin). In OpenReq Live, we focus on the design of User Interface elements (screens) than can be integrated (in adapted form) within the different OpenReq trial implementations and show the core functionalities of the OpenReq product.

The UI approach of OpenReq Live serves as a collection of reference designs that support OpenReq UI developers (and beyond) in the design of their individual user interfaces. Furthermore, OpenReq Live serves as a component that helps to demonstrate in an integrated fashion the application of the developed OpenReq recommendation and decision technologies for requirements engineering. The basic approach in this context will be a development process based on immediate feedback from trial partners and user communities that will help to efficiently develop highly relevant features immediately applicable in different requirements engineering scenarios. Selected features will be integrated into a sequence of MVPs that will be iteratively extended to a full-fledged OpenReq requirements engineering UI.

The current API functions of OpenReq Live are described in section 6.2.1 – these will be, based on the feedback of the trial partners, further extended in future. Also, the service of calculating similar / related requirements is already available separately (see section 5.2.18). Services, like for instance, MAUT-based recommendation, MAUT-based inconsistency detection, Pro / Con Analysis and stakeholder recommendations are currently part of the OpenReq Live itself and will systematically become dedicated services in future (based on the needs of the trial partners).

8.1.1 Requirements Improvement Recommendation

In addition to the aforementioned features in Section 8.1, functionality from WP3 has been integrated into OpenReq Live. This is a service to give users feedback on the quality of their requirements through recommendations for improving the text. Figure 10 is a screenshot of the integrated functionality into OpenReq Live.



Evaluation Software

For evaluating the recorded training data, an evaluation software is required. This software requires the connection and the access to the clock's internal memory. The evaluation should contain measured information regarding the distance, the height, the average heart rate, and the calorie consumption.

Passive Ambiguity
Authors should state requirements in active form, as passive conceals who is responsible for the action.

Pronoun
Potentially unclear reference.

Imprecise
These expressions are not concise.

Adjective
Adjectives are ambiguous.

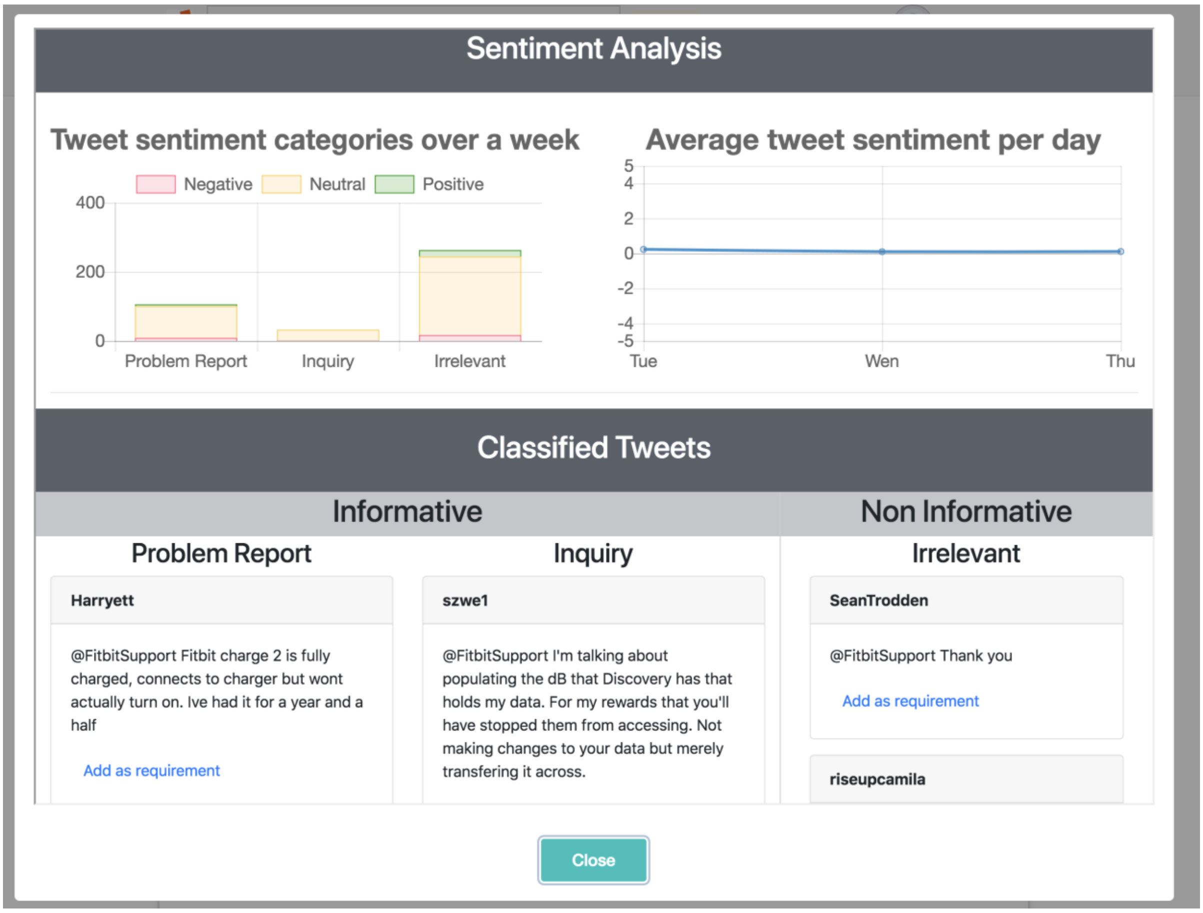
Unclear Associativity

Close

Figure 10: Recommendations for quality improvements of entered requirement texts

8.1.2 Explicit Feedback Analytics

Functionality from WP2 has also been integrated into OpenReq Live. This service allows stakeholders to get insights about what Twitter users think about their product. The view offered by this integration shows analysis of the sentiment for frequent topics, and classifies tweets in informative (problem report and inquiry), as well as uninformative (irrelevant). Using this information and integrated “Add as Requirement” functionality, stakeholders can convert these Twitter opinions and recommendations directly into requirements in OpenReq Live. Figure 11 shows this integrated functionality.



Harryett

@FitbitSupport Fitbit charge 2 is fully charged, connects to charger but wont actually turn on. Ive had it for a year and a half

Add as requirement

szwe1

@FitbitSupport I'm talking about populating the dB that Discovery has that holds my data. For my rewards that you'll have stopped them from accessing. Not making changes to your data but merely transfering it across.

SeanTrodden

@FitbitSupport Thank you

Add as requirement

riseupcamila

Figure 11: Twitter discussions on w.r.t. the given requirement

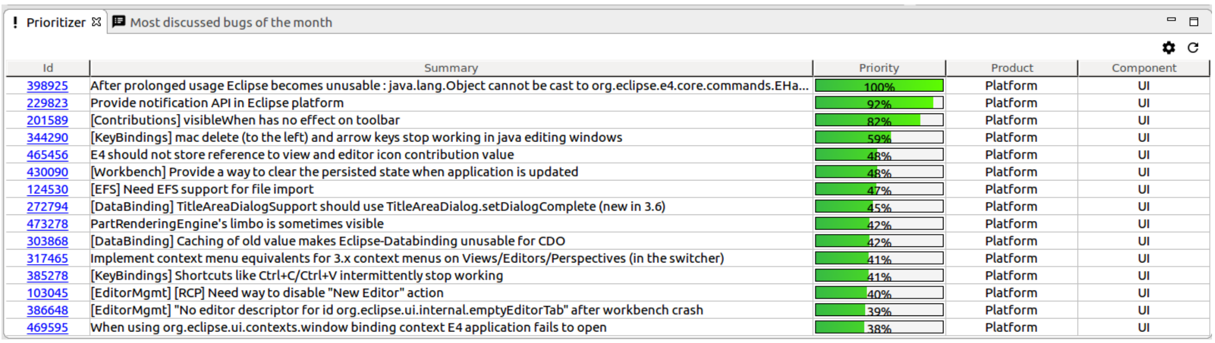
8.2 OpenReq Eclipse plugin

The Eclipse IDE offers a large variety of plug-ins, which can be installed to assist a software developer during his/her work. The OpenReq Eclipse plug-in adds intelligent bug analysis capabilities for the user and also analyses the behaviour of the user during his/her work, which might lead to new requirements for the Eclipse IDE.

The Eclipse Bugzilla issue tracking system (https://bugs.eclipse.org/bugs/) consists of thousands of bugs and feature requests, which have been reported by different individuals. Since the Eclipse projects are open source, it is crucial that individual companies and individual contributors participate fixing these issues.

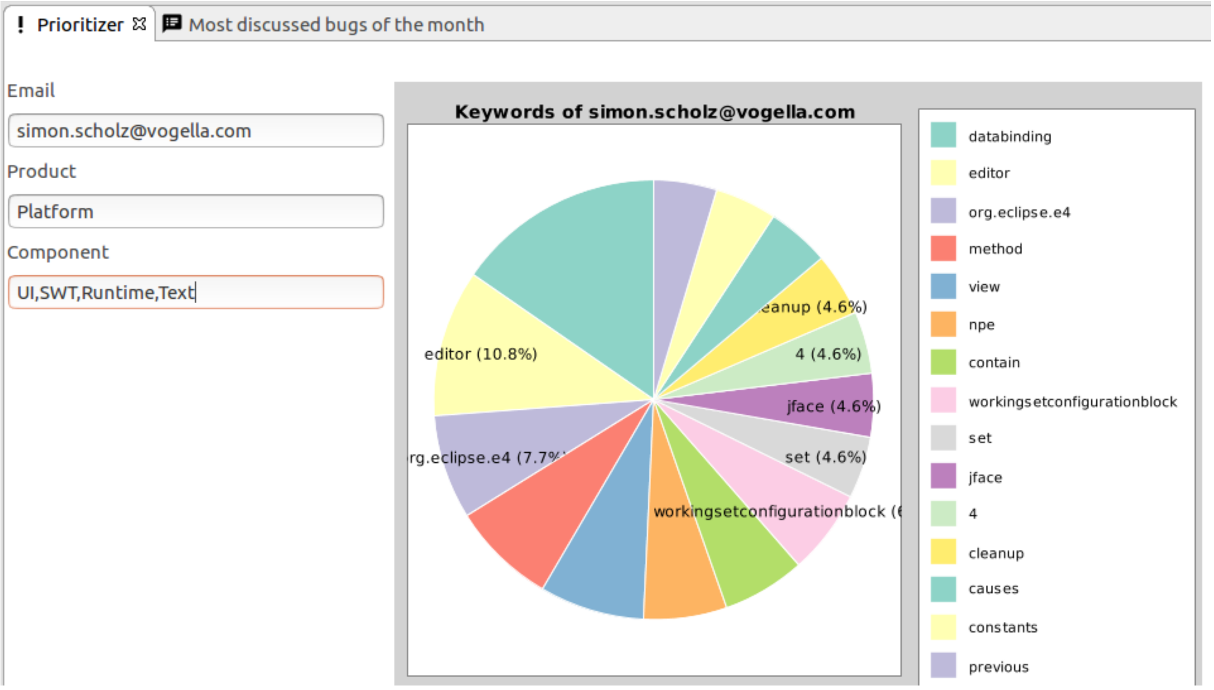
For newcomers and even for experienced contributors it is not that easy to figure out, which issues should be resolved first. Often bugs are fixed depending on personal interest and experience. In order to improve the process of choosing an issue and having a vibrant Open Source project the OpenReq Eclipse plug-in can prioritize issues for an individual.

The Prioritizer view in Eclipse can show issues, which have been prioritized by an algorism, which calculates the best fits for the individual developer.



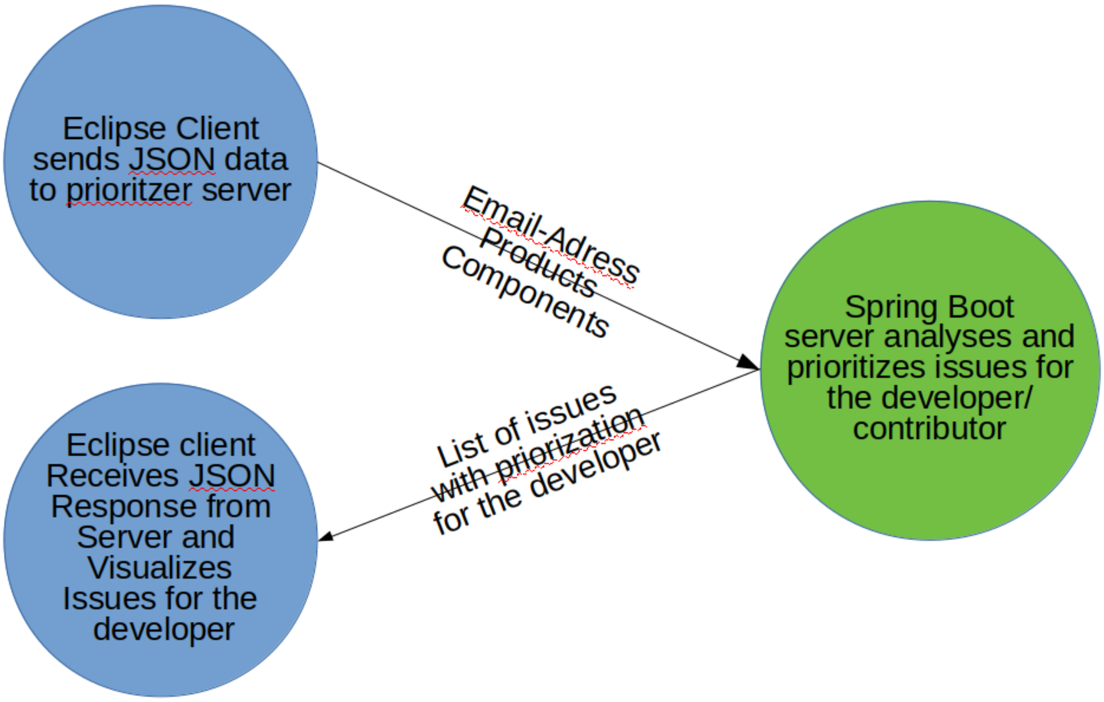
Id	Summary	Priority	Product	Component
398925	After prolonged usage Eclipse becomes unusable : java.lang.Object cannot be cast to org.eclipse.e4.core.commands.EHa...	100%	Platform	UI
229823	Provide notification API in Eclipse platform	92%	Platform	UI
201589	[Contributions] visibleWhen has no effect on toolbar	82%	Platform	UI
344290	[KeyBindings] mac delete (to the left) and arrow keys stop working in java editing windows	59%	Platform	UI
465456	E4 should not store reference to view and editor icon contribution value	48%	Platform	UI
430090	[Workbench] Provide a way to clear the persisted state when application is updated	48%	Platform	UI
124530	[EFS] Need EFS support for file import	47%	Platform	UI
272794	[DataBinding] TitleAreaDialogSupport should use TitleAreaDialog.setDialogComplete (new in 3.6)	45%	Platform	UI
473278	PartRenderingEngine's limbo is sometimes visible	42%	Platform	UI
303868	[DataBinding] Caching of old value makes Eclipse-Databinding unusable for CDO	42%	Platform	UI
317465	Implement context menu equivalents for 3.x context menus on Views/Editors/Perspectives (in the switcher)	41%	Platform	UI
385278	[KeyBindings] Shortcuts like Ctrl+C/Ctrl+V intermittently stop working	41%	Platform	UI
103045	[EditorMgmt] [RCP] Need way to disable "New Editor" action	40%	Platform	UI
386648	[EditorMgmt] "No editor descriptor for id org.eclipse.ui.internal.emptyEditorTab" after workbench crash	39%	Platform	UI
469595	When using org.eclipse.ui.contexts.window binding context E4 application fails to open	38%	Platform	UI

In the Settings ⚙ of the Prioritizer view the user can choose in which product and component he/she is interested and with the email address the algorism determines, which issues are of interest.



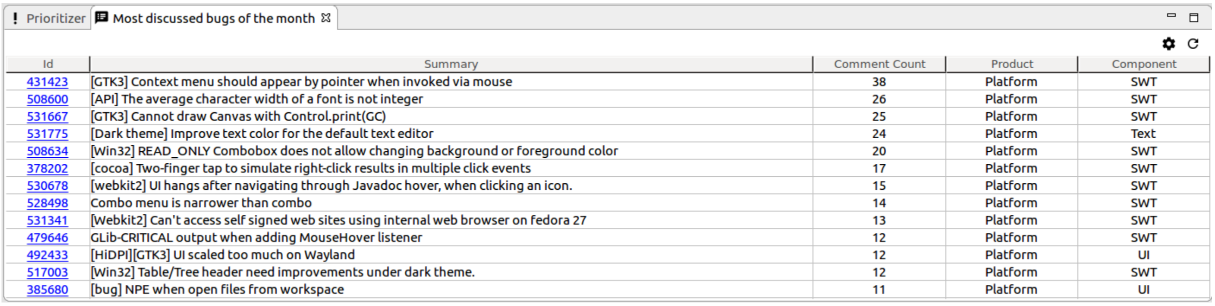
Email
simon.scholz@vogella.com
Product
Platform
Component
UI,SWT,Runtime,Text

The following diagram shows how data flows between a Spring Boot server and the Eclipse client.


Eclipse Client sends JSON data to prioritzer server
Email-Adress
Products
Components
Spring Boot server analyses and prioritizes issues for the developer/ contributor
List of issues with priorization for the developer
Eclipse client Receives JSON Response from Server and Visualizes Issues for the developer

Internally it uses the http://217.172.12.199:9002/prioritizer/chart API and http://217.172.12.199:9002/prioritizer/compute API to receive the chart with keywords for the developer and the prioritized issues.

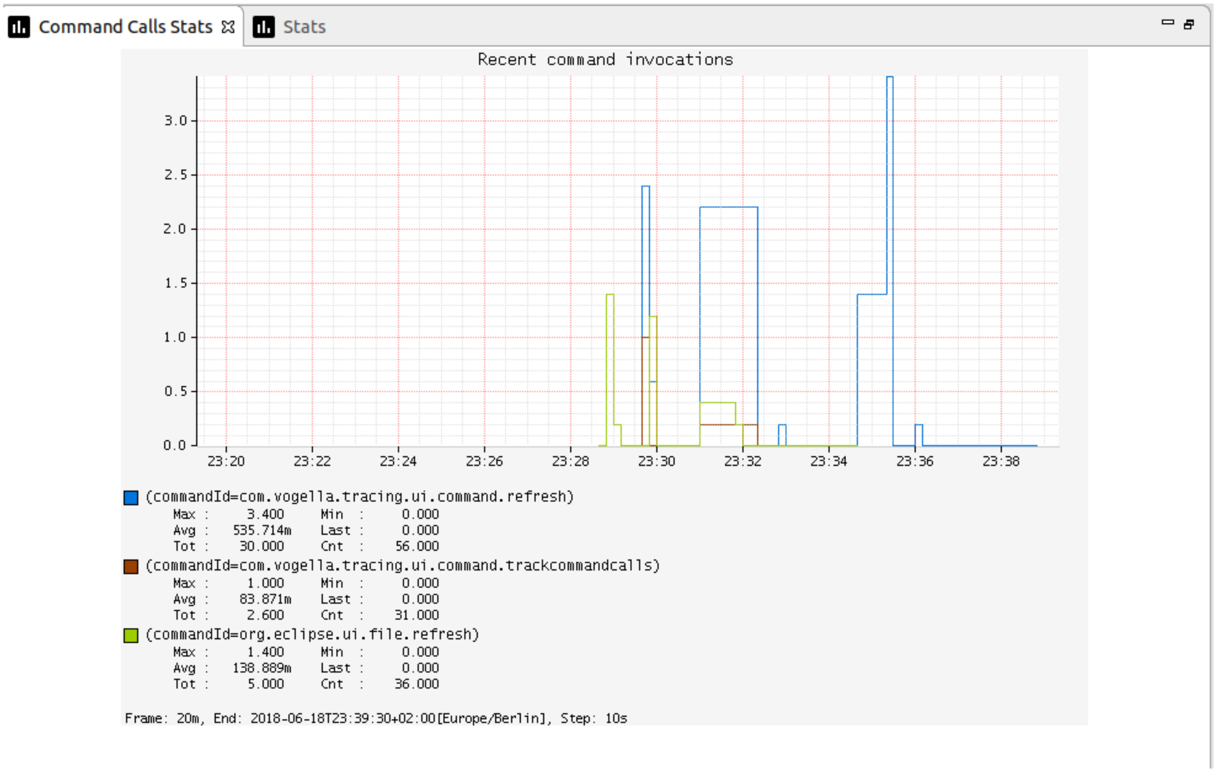
A user can also view the most discussed issues of a month in order to track what's currently going on in the project, get the latest updates and can participate in the discussions.



Id	Summary	Comment Count	Product	Component
431423	[GTK3] Context menu should appear by pointer when invoked via mouse	38	Platform	SWT
508600	[API] The average character width of a font is not integer	26	Platform	SWT
531667	[GTK3] Cannot draw Canvas with Control.print(GC)	25	Platform	SWT
531775	[Dark theme] Improve text color for the default text editor	24	Platform	Text
508634	[Win32] READ_ONLY Combobox does not allow changing background or foreground color	20	Platform	SWT
378202	[cocoa] Two-finger tap to simulate right-click results in multiple click events	17	Platform	SWT
530678	[webkit2] UI hangs after navigating through Javadoc hover, when clicking an icon.	15	Platform	SWT
528498	Combo menu is narrower than combo	14	Platform	SWT
531341	[Webkit2] Can't access self signed web sites using internal web browser on fedora 27	13	Platform	SWT
479646	GLib-CRITICAL output when adding MouseHover listener	12	Platform	SWT
492433	[HiDPI][GTK3] UI scaled too much on Wayland	12	Platform	UI
517003	[Win32] Table/Tree header need improvements under dark theme.	12	Platform	SWT
385680	[bug] NPE when open files from workspace	11	Platform	UI

Besides the prioritization of issues for individual developers the OpenReq Eclipse plug-ins also provide the capabilities to collect metrics about the user behaviour in order to generate new issues to improve the usability of the Eclipse IDE.

These metrics are recorded by using micrometer io (https://micrometer.io/), for example:



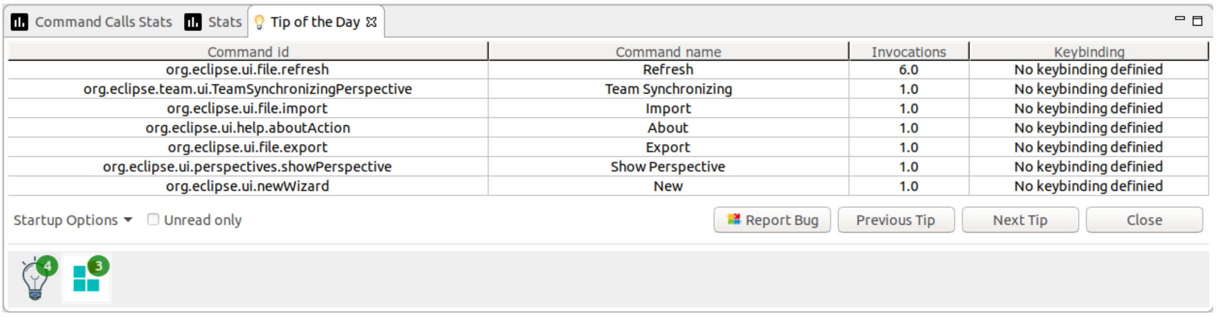
```
🟦 (commandId=com.vogella.tracing.ui.command.refresh)
      Max :     3.400     Min  :     0.000
      Avg :   535.714m    Last :     0.000
      Tot :    30.000     Cnt  :    56.000
🟫 (commandId=com.vogella.tracing.ui.command.trackcommandcalls)
      Max :     1.000     Min  :     0.000
      Avg :    83.871m    Last :     0.000
      Tot :     2.600     Cnt  :    31.000
🟩 (commandId=org.eclipse.ui.file.refresh)
      Max :     1.400     Min  :     0.000
      Avg :   138.889m    Last :     0.000
      Tot :     5.000     Cnt  :    36.000

Frame: 20m, End: 2018-06-18T23:39:30+02:00[Europe/Berlin], Step: 10s
```

These graphs can be visualized by opening the Command Calls Stats view in the Eclipse IDE.

These graphs show how frequently certain commands are invoked when working with the Eclipse IDE.

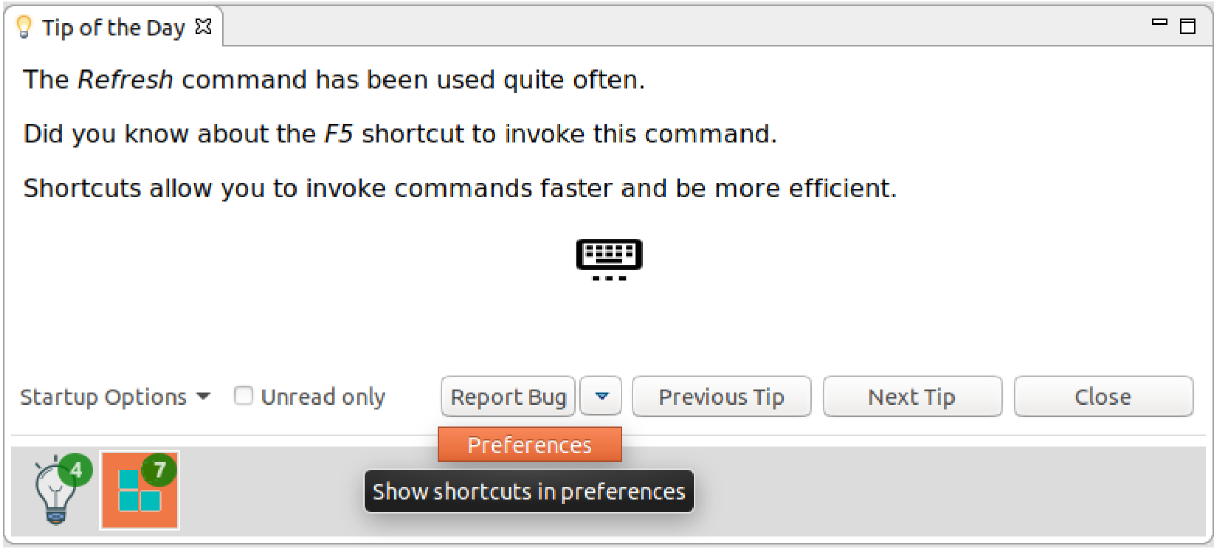
The latest Eclipse IDE release also offers a “Tip of the day” dialog, which shows tips to the user. The OpenReq Eclipse Plug-ins also contributes tips to this dialog, e.g., in case the user invokes commands relatively often by using menu entries or toolbar entries instead of shortcuts on the keyboard.

For all frequently used commands:



Command id	Command name	Invocations	Keybinding
org.eclipse.ui.file.refresh	Refresh	6.0	No keybinding definied
org.eclipse.team.ui.TeamSynchronizingPerspective	Team Synchronizing	1.0	No keybinding definied
org.eclipse.ui.file.import	Import	1.0	No keybinding definied
org.eclipse.ui.help.aboutAction	About	1.0	No keybinding definied
org.eclipse.ui.file.export	Export	1.0	No keybinding definied
org.eclipse.ui.perspectives.showPerspective	Show Perspective	1.0	No keybinding definied
org.eclipse.ui.newWizard	New	1.0	No keybinding definied

More precise for individual frequently used commands:



The *Refresh* command has been used quite often.

Did you know about the *F5* shortcut to invoke this command.

Shortcuts allow you to invoke commands faster and be more efficient.

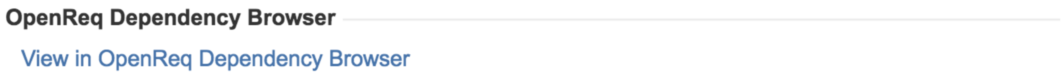
The "Report Bug" button makes use of the innosensr issue tracking system by TU Graz: http://innosensr.com/swagger-ui.html#!/api45controller/createRequirementJsonUsingPOST

Further information, implementation details, sample setup and updates can also found here:

https://github.com/vogellacompany/openreq

8.3 Qt JIRA plugin

The Qt JIRA plugin adds a section to the information page of a JIRA item, this section contains a link to the OpenReq Dependency Browser ("Qthulhu"). Upon clicking the link, a new tab opens which displays the dependency graph of that particular issue.



OpenReq Dependency Browser

View in OpenReq Dependency Browser

To avoid performance issues the graph is displayed in a new tab and not inside the item page of JIRA.

9 REFERENCES

[1] OpenReq Consortium, «D1.4 - Project standards and infrastructure document,» 2017.

[2] OpenReq Consortium, "D2.2 - Requirements Intelligence Engine version 1," 2018.

[3] OpenReq Consortium, "D3.2 - Recommender Engine - Version 1," 2018.

[4] OpenReQ Consortium, "D5.2 - Requirements Dependency Engine Version 1," 2017.

[5]OpenReq Consortium, "D4.1 - OpenReq Approach to Group Decision Support," 2018.

[6] J. Dyer. MAUT - Multi Attribute Utility Theory. In Multiple Criteria Decision Analysis: State of the Art Surveys, volume 78 of Intl. Series in Operations Research & Management Science, pages 265–292. Springer New York, 2005.

[7] J. Bernacki, I. Blazejczyk, A. Indyka-Piasecka, M. Kopel, E. Kukla, and B. Trawinski. Responsive Web Design: Testing Usability of Mobile Web Applications, Asian Conference on Intelligent Information and Database Systems (ACIIDS 2016), pp. 257-269, 2016

[8] L. Chen and P. Pu. Evaluating recommender systems from the user's perspective: survey of the state of the art, User Modeling and User-Adapted Interaction, 22(4–5):317–355, 2012