H2020-SC6-CO-CREATION-2016-2017 / CO-CREATION FOR GROWTH AND INCLUSION
Grant Agreement nº 769975

SoCaTel

A multi-stakeholder co-creation platform for better access to Long-Term Care services

Start date of project: 01/12/2017
Duration: 36 months

Deliverable: D4.5

Data Source Exposure through well-constructed APIs

By CyRIC

Due date of deliverable: 31/08/2019
Actual submission date: 13/09/2019

Responsible WP: WP4
WP responsible partner: CYR
Deliverable responsible partner: CYR
Revision: vs 2.0

Dissemination level		
PU	Public	
CO	Confidential, only for members of the consortium (including the Commission Services)	X
CI	Classified, information as referred to in Commission Decision 2001/844/EC	

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

AUTHORS

Author	Institution	Contact (e-mail, phone)
Rafael Constantinou	**CyRIC**	**r.constantinou@cyric.eu**
Loizos Koukoumas	**CyRIC**	**l.koukoumas@cyric.eu**
Rizk Allah Touma	**Everis**	**rizk.allah.touma@everis.com**
Victor Colome Carcole	**URV**	**victorcolomec@gmail.com**

DOCUMENT CONTROL

Document version	Date	Change
V 0.1	30/07/2019	Table of Contents
V 0.2	15/08/2019	First Draft
V 0.3	06/09/2019	Final Draft
V 1.0	10/09/2019	Completed Version for Review
V 2.0	13/09/2019	Reviewed Version

VALIDATION

Reviewers		Validation date
CyRIC	Pantelis Nicolaou	10/09/2019
Everis	Emmanuel Jamin	11/09/2019

DOCUMENT DATA

Keywords	SoCaTel, Data Source Exposure, APIs
Contact	Name: Pantelis Nicolaou Partner: CyRIC Tel: +357 22 282828 E-mail: p.nicolaou@cyric.eu
Delivery date	15/09/2019

This SoCaTel project has received funding from the European Union's Horizon 2020 Research and Innovation Programme. The opinions expressed in this document reflect only the author's view and reflects in no way the European Commission's opinions. The European Commission is not responsible for any use that may be made of the information it contains.

EXECUTIVE SUMMARY

The main objective of this document is to outline how the collected data sources are being handled and processed by the Data Mining and Analytics Engine, as well as the Recommendation Engine. In addition, the document describes how these two engines expose their functionality and disseminate their results to other system components through well-defined RESTful API endpoints, that are also fully defined.

The RESTful endpoints are then used by the Portal Layer to facilitate and enhance user functionality by providing useful insights and recommendations to different portal sections as defined by the portal design. There insights and recommendations are helping the users to decide potential suitable co-creation groups to join, statistics and insights of co-creation groups that already participate, relevant service recommendation listings, etc.

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

Table of Contents

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement Nº 769975.

Glossary

Abbreviation	Expression
SOCATEL	A multi-stakeholder co-creation platform for better access to Long-Term Care services
KB	Knowledge Base (WP4)
DAI	Data Acquisition Layer Derived Insights
RDI	Raw Data Derived Insights
SRE	SoCaTel Recommendation Engine
SREs	SoCaTel Recommendation Engine's Recommendations
SRE.x	SoCaTel Recommendation Engine – Recommendation x
DTO	Data Transfer Object
REC	Recommendation
ES	Elasticsearch

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

List of Figures

List of Tables

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

1 INTRODUCTION – KNOWLEDGE BASE USAGE

The Data Mining and Analytics algorithms used by the Data Integration Layer offer a wide range of statistics, analytics and information retrieval insights in every aspect of the data collected throughout the platform. As *Chapter 3 - Data Mining and Analytics Engine*, outlines, the Data Integration Layer uses the information that is gathered from all available collected resources that are derived and consumed by the different origins of the data.

The first origin of data, is the portal's related data which are replicated from the portal's operational repository to the KB using a replication procedure which is defined and explained in *Chapter 2 - Portal's Data Replication* . The second origin of data are the data derived from the declared Twitter and Facebook services of the registered organisations. In addition, Open, Research and Statistical Data are also consumed by the Data Acquisition Layer as it is already defined in D4.1.

All data, that belong to either of the two listed origins, are being handled first by the Data Acquisition Layer that practically retrieves them from the various web sources or from the KB2.1 (Chapter 6 ,Figure 4) which is the Raw Data ES Repository (portal's operational data that are periodically replicated to the platform). The Data Acquisition Layer then rediverts these data to the Semantic pre-processing Layer for semantic conversion and annotation.

The Semantic Pre-processing layer is responsible for semantically annotating data into useful RDFs relationships of data that in the long-term is being utilised by the Data Integration Layer which indirectly uses GraphQL to retrieve relevant information. The relevant information is either being used by the Recommendation Engine as input or is directly made available for consumption by the portal layer in the form of mining and analytics insights.

The above defined process, that details how the Knowledgebase is being used and exploited by the recommendation Engine is explained in the next chapters with the accompanied set of RESTful API calls that are being used as a medium of communication between the Portal and the KB. The RESTful API layer that is explained in Chapter 5 - RESTful API Definitions is the upper layer of communication between the portal and the KB.

2 PORTAL'S DATA REPLICATION PROCESS

2.1 INTRODUCTION

The Data Integration Layer's main objective is to provide a rich set of recommendations to the SoCaTel portal as it is already explained in D4.3 Context Aware Recommendation Engine. This rich set of recommendations provide an

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

overall enrichment of the co-creation phase with the multiple data analysis, statistics and recommendations that are displayed throughout the portal (Search Page, Co-Creation Groups, Co-Creation Dashboard, etc.). This provides an additional layer of statistical information that can be used to facilitate the overall user experience and participation to co-creation groups with knowledge that becomes more and more useful; from an analytical perspective; while a discussion around a topic evolves.

As D4.5 explains, the Recommendation Engine uses state-of-the-art open source tools that are used as a black-box deployment to fulfil all the SREs that are listed in the deliverable. These SREs, are explained in *Chapter 4* - SoCaTel Recommendation Engine. SoCaTel Recommendation Engine is highly dependent to portal data in order to achieve its purpose. Therefore, the pseudonymised replication of data is essential because the Recommendation Engine is a memory and performance intensive set of tools that would overwhelm the user experience when running on the portal side.

For the above-mentioned cause, and as the architectural design of SoCaTel suggests in D3.2 SoCaTel Platform Architecture Design, there is a specialised portal's replication process (R2 – Transfer Data API Calls) that replicates the portal's operational repository from the Portal Side deployment to the KB.

The Portal's replication process consists of the process that starts from when the data stored in the portal's database is transferred to the Raw Data Repository within the KB (KB2.1 from page 51, Figure 4). All the data generated in the platform, either by the end-users or the portal itself, is stored in the MySQL Database.

2.2 ANONYMISATION PROCESS

The following process will explain how the anonymisation and replication occurs from the SoCaTel portal to the KB. After every action, that occurs within the portal which is stored on the MYSQL instance of the deployment, in parallel to this the anonymisation process takes place to anonymise any sensitive data. Anonymised data will then be replicated to the KB.

A *Publisher* module has been implemented that is called every time data is created, updated or deleted. This Publisher is responsible for creating the DTO with the specific data that is sent and at the same time encrypting the *user_id* wherever it appears.

Besides the user table, that many fields are not omitted from transmission in order to keep the user's privacy in the portal, there are some other differences in the Locality, Language, Skill and Theme relationships. Those differences are because the values of these tables are predefined in the database, therefore they are not sent to the Elasticsearch as they are not created or updated by the user

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

or the platform. Since the Elasticsearch does not have those indexes, it is necessary to send the entire information of those relationships in each created or updated index. Furthermore, the many-to-many relationships are included in the indexes rather than sending another index for each relationship. See an example in *Table 3 – ES User Table Example* Row.

This *user_id* encryption, as explained in Deliverable 3.4, consists of a mixture of Noise addition and Encryption: when the DTO is being created, each time the *user_id* appears, noise is applied both in front of and behind it and then it is encrypted. A simple example is shown in the following table:

Anonymization process
Having: user_id = 1000, the noise in front = no1se, the noise behind = no2se. ➢ This is the process to send the user_id from the Portal Side to the Elasticsearch: 1. The noise is added in front of and behind the user_id: *1000 ---> no1se1000no2se* 2. The noisy user_id is encrypted with 256-bit AES, using an 8-byte salt and a 16-byte password: *no1se1000no2se ---> X9wPuUuEimYbvNVNmx8O/A==* 3. This encrypted noisy *user_id* is sent to the Elasticsearch. ➢ This is the process to send back the encrypted noisy *user_id* from the Elasticsearch to the Portal side: 1. The encrypted noisy *user_id* is sent through Rest API calls from the Recommendation Engine. 2. The encrypted noisy *user_id* is retrieved by the Portal Side and it is decrypted using the same password and salt: *X9wPuUuEimYbvNVNmx8O/A== ---> no1se1000no2se*

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

3. The 5 first and 5 last characters are removed from the noisy *user_id*, removing the noise and obtaining the *user_id*:

No1se1000no2se ---> 1000

Table 1 – Anonymisation Process

The decision of combining these two techniques can be explained with the following table, which contains all the anonymisation techniques:

	Is Singling out still a risk?	Is Linkability still a risk?	Is Inference still a risk?
Pseudonymisation	Yes	Yes	Yes
Noise addition	Yes	May not	May not
Substitution	Yes	Yes	May not
Aggregation or K-anonymity	No	Yes	Yes
L-diversity	No	Yes	May not
Differential privacy	May not	May not	May not
Hashing/Tokenization	Yes	Yes	May not

Figure 1 – Strengths and Weaknesses of the Pseudonymisation Techniques[1]

Looking at this table, a single or combination of choices was necessary to achieve the desired level of security. First, we discarded the techniques of Aggregation, L-diversity or Differential privacy as those need aggregation to work so it would be necessary for hierarchical data based on some ontologies to work, and this is not this case with SoCaTel. The next discarded option was Hashing/Tokenization as this is not reversible, and it is needed to revert the anonymisation to retrieve the data from the Recommendation Engine. Remaining the three first options, it was decided to do a mixture of pseudonymisation (encryption), as it is reversible, along with one of the others. The noise addition was selected finally as it has a better linkability risk prevention.

2.3 REPLICATION/TRANSFER PROCESS

Once the DTO, to be sent, has been created, the Publisher creates an Event where it stores the data necessary to create, update or delete the Elasticsearch index. This Event is composed of 4 different fields:

- *Source* → is a generic field that corresponds to the information to be stored in the Elasticsearch (the DTO that has been previously created).
- *Type* → is the type of operation to be done (CREATE, UPDATE, DELETE).

[1] Strengths and Weaknesses of the Pseudonymisation Techniques: https://www.dataprotection.ro/servlet/ViewDocument?id=1085

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

- *Index* → is the name of the corresponding index in the Elasticsearch, for example, the user corresponds to the *so_user* index, which intentionally corresponds to the same name that the user table has in the MySQL database for consistency issues and ease of development between the different developers.
- *Id* → is the identifier of the source.

Once the event is created, it is executed asynchronously and in parallel to other possible events through an ApplicationEventMulticaster[2], thus forwarding it to the corresponding listener of the event.

The implemented Listener receives this event and is responsible for creating the corresponding Elasticsearch index by using the Elasticsearch Java library. In the case of the Create and Update indexes, the event's source is added in JSON format, transformed from Java using the Jackson tool. Once the index is created, it is sent to the Elasticsearch instance through the RestHighLevelClient library, which connects to the socatel-es.ozwillo-preprod.eu:9200 endpoint, where the Elastic instance is, using basic authorization credentials (user and password). This RestHighLevelClient[3] library provides a set of Rest API calls to operate on the Elasticsearch from Java, this API calls were named in previous deliverables as Transfer Data API calls.

2.4 DATA REPLICATION EXAMPLES

Once some amount of data is stored in the Elasticsearch, the Recommendation Engine will have been able to be trained and will be able to generate recommendations. These recommendations will be requested by the Portal Side through RESTful APIs.

The data used for the Data Mining and Analytics are the data being replicated and anonymised by the portal which are then being sent to the KB. In detail, the following are considered which is either read from the Semantic Repository (GraphDB) (KB1) or directly read from Elasticsearch (KB2.1).

A JSON example of each table row is being given inline below:

[2] ApplicationEventMulticaster: https://docs.spring.io/spring-framework/docs/current/javadoc-api/org/springframework/context/event/ApplicationEventMulticaster.html

[3] RestHighLevelClient: https://github.com/elastic/elasticsearch

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

1. History

ES History Table Row


```
{
        "history_id" : 194,
        "history_text" : "history.delete_account",
        "history_timestamp" : 1566484729022,
        "history_type" : 9,
        "history_level" : 0,
        "user_id" : "6e0a2a341891c8e1cccdc7580746f0d2",
        "organisation_id" : null,
        "group_id" : null
}
```


Table 2 – ES History Table Example Row

2. User

ES User Table Row


```
{
        "user_id" : "4c1560b764fa63ea2322aae23a775b5c",
        "locality" : {
          "locality_id" : 1004,
          "locality_name" : "Vilanova",
          "locality_parent" : {
            "locality_id" : 1000,
            "locality_name" : "Spain",
            "locality_parent" : null
          }
        },
        "primary_language" : {
          "language_id" : 2,
          "language_code" : "es",
          "language_name" : "Spanish"
        },
        "secondary_language" : {
          "language_code" : "en",
          "language_id" : 1,
          "language_name" : "English"
        },
        "organisation_id" : null,
        "themes" : [
          {
            "theme_id" : 3,
            "theme_name" : "Administration"
          },
          {
            "theme_id" : 5,
            "theme_name" : "ABS"
          }
```


This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

```
        ],
        "groups" : [
          {
            "group_id" : 25
          },
          {
            "group_id" : 27
          }
        ],
        "skills" : [
          {
            "skill_name" : "Computer",
            "skill_id" : 4
          },
          {
            "skill_name" : "Design",
            "skill_id" : 6
          }
        ]
}
```


Table 3 – ES User Table Example Row

3. Organisation

ES Organisation Table Row


```
{
        "organisation_id" : 14,
        "organisation_name" : "Developers",
        "organisation_structure" : 0,
        "organisation_website" : "https://socatel.eu",
        "twitter_screen_name" : null,
        "twitter_account_description" : null,
        "twitter_user_id" : null,
        "twitter_oauth_token" : null,
        "twitter_oauth_secret" : null,
        "facebook_page_id" : null,
        "facebook_oauth_token" : null
}
```


Table 4 – ES Organisation Table Example Row

4. Group

ES Group Table Row


```
{
        "group_id" : 28,
        "group_name" : "Topic Title",
        "group_description" : "Topic Description",
        "group_status" : 1,
        "group_create_time" : 1567169127000,
```


This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

```
        "group_next_step_time" : null,
        "locality" : {
          "locality_id" : 1000,
          "locality_name" : "Spain",
          "locality_parent" : null
        },
        "language" : {
          "language_id" : 2,
          "language_code" : "es",
          "language_name" : "Spanish"
        },
        "user_initiator_id" : "4c1560b764fa63ea2322aae23a775b5c",
        "themes" : [
          {
            "theme_id" : 3,
            "theme_name" : "Administration"
          },
          {
            "theme_id" : 8,
            "theme_name" : "Health"
          }
        ],
        "users" : [
          "4a31e6f3e2671e2dc5cc628d488fec66",
          "4c1560b764fa63ea2322aae23a775b5c"
        ]
}
```


Table 5 – ES Group Table Example Row

5. Post

ES Post Table Row


```
{
        "post_id" : 57,
        "post_text" : "@FrancisK - ADMR tzst 2",
        "post_timestamp" : 1566805028037,
        "post_upvotes" : 0,
        "post_downvotes" : 0,
        "post_type" : 0,
        "post_phase" : 0,
        "post_visible" : 0,
        "post_pin" : 0,
        "post_parent_id" : 34,
        "author_user_id" : "f8742793b10ac5417bfe5fafd8a3e40d",
        "group_id" : 6,
        "organisation_id" : null
}
```


Table 6 – ES Post Table Example Row

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

6. Proposition

ES Proposition Table Row


```
{
        "proposition_id" : 1,
        "proposition_text" : "This is a proposition",
        "proposition_type" : 0,
        "proposition_upvotes" : 1,
        "proposition_downvotes" : 0,
        "proposition_visible" : 0,
        "proposition_pin" : 0,
        "proposition_timestamp" : 1566917707000,
        "post_id" : 95,
        "user_id" : "4a31e6f3e2671e2dc5cc628d488fec66"
}
```


Table 7 – ES Proposition Table Row

7. Service

ES Service Table Row


```
{
        "service_id" : 1,
        "service_name" : "Service Test",
        "service_description" : "Service Description",
        "service_website" : null,
        "locality" : {
          "locality_id" : 1004,
          "locality_name" : "Vilanova",
          "locality_parent" : {
            "locality_id" : 1000,
            "locality_name" : "Spain",
            "locality_parent" : null
          }
        },
        "language" : {
          "language_id" : 2,
          "language_code" : "es",
          "language_name" : "Spanish"
        },
        "service_status" : 0,
        "service_hashtag" : null,
        "themes" : [
          {
            "theme_id" : 2,
            "theme_name" : "Activities"
          }
        ],
        "twitter_screen_name" : null,
        "twitter_account_description" : null,
```


This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.


```
        "twitter_user_id" : null,
        "twitter_oauth_token" : null,
        "twitter_oauth_secret" : null,
        "facebook_page_id" : null,
        "facebook_oauth_token" : null
}
```


Table 8 – ES Service Table Row

8. User Post Vote

ES User Post Vote Table Row


```
{
        "id" : "3338a24e0f1d0c01a4d9ce1e1c9307f2-50",
        "user_id" : "3338a24e0f1d0c01a4d9ce1e1c9307f2",
        "post_id" : 50,
        "user_post_vote_type" : 2
}
```


Table 9 – ES User Post Vote Table Row

9. User Proposition Vote

ES User Proposition Vote Table Row


```
{
        "id" : "4a31e6f3e2671e2dc5cc628d488fec66-1",
        "user_id" : "4a31e6f3e2671e2dc5cc628d488fec66",
        "proposition_id" : 1,
        "proposition_user_vote_type" : 2
}
```


Table 10 – ES User Proposition Vote Table Row

3 DATA MINING AND ANALYTICS ENGINE

The Data Mining and Analytics Engine chapter outlines the basic elements of the Data Mining and Analytics algorithms that are used as part of the Data Integration Layer. In detail, the following subsections outline the I2 – Data Mining and I3 – Data Analytics and the how the flow of information from the Elasticsearch and the GraphDB is being used to perform the Data Mining and Analysis.

The Data Mining and Analytics, in the form of *insights,* is then becoming available for consumption by the recommendation engine and the portal layer. The latter process is explained in the next Chapter 4 - SoCaTel Recommendation Engine.

This section presents and correlates the set of reusable Data Mining and Analytics algorithms and metrics, that are already outlined and explained in D4.3 – Context Aware Recommendation Engine in Chapter 3 – Context Aware Recommendation Engine Definitions. These algorithms and their associated

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

metrics are used in accordance with the Data Acquisition Layer's obtained data as these are defined in the previous submitted deliverables such as:

- D4.1 Data Standardization and Semantic Interoperability
- D4.2 Data Storage Implementation and Analytical Infrastructure
- D4.3 Context Aware Recommendation Engine
- D3.2 SoCaTel Platform Concrete Architecture Design
- D3.4 SoCaTel Core Platform Implementation

The Data Mining and Analytics outcome will thus, hereafter, divided into the following sub-categories based on their origin of consumed data. The two (2) main source of origin of data are the data derived from:

i. Data Acquisition Handlers (D[1-5])
ii. Raw Data (D6) which are pseudonymised and replicated from the portal side to the KB with a process explained in *Section 2* - **Portal's Data Replication**

The generated Data Mining and Analytics insights are then made available to the SoCaTel portal to enrich the overall user experience by providing useful information to all kinds of user interaction levels such as the following:

1. Co-Creation Group Insights and Statistics
2. Search Existing Services Insights
3. Search Existing Co-Creation Services
4. Input to the Recommendation Engine for better ranking the SREs results (SREs)

The insights and analytics provide an overall user insight enrichment to the portal and provide the platform users with an additional layer of information that help them decide which co-creation group is closer to their interests, professional expertise, region, language. Moreover, it also helps users to understand which co-creation groups are more popular than others and which associated services need improvement. Finally, the Administrative Statistics help the portal administrator to decide which declared services are underutilised or need further improvement, etc.

3.1 DATA MINING AND ANALYTICS INSIGHTS

This subsection presents the Data Mining and Analytics derived set of insights according the data origin categories outlined in the previous section. The distinction of the insights is based on the identified categories. More categories and/or insights will be added in the nearest future upon need. The insights presented in this section; are the basic insights a portal of this kind needs to have for providing an enriched functionality to the participating users.

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

The first data origin category is the Data Acquisition Layer extracted data. This category refers to the data that are streamed from the various declared resources such as Twitter/Facebook accounts, Open Data Datasets, Research and Statistical Datasets, Governmental Datasets and Linked Open Data. All sources of this kind are declared in the portal by the different organisations registered, the moderators of co-creation groups and the administrators of the portal. As soon as these resources become known to the KB, an extraction mechanism starts extracting all information necessary using the implementation explanation methods described in D4.1 and D3.4.

The Data Mining and Analytics Insights of the first category is found in the table below.

D[2-5] - Data Acquisition Layer Derived Insights	
DAI	DAI - 1. Natural Language Processing on Twitter/Facebook Services DAI - 2. Open Data → Data Analytics on Datasets based on pre-defined options (standard deviation, min/max, sampling, aggregation, sum, mean, etc.)

Table 11 – DAI. Data Acquisition Layer Derived Insights

The second data origin category is the Raw Data. The Raw Data are the portal data which are being replicated and anonymised by the portal and are then sent to the KB using a replication and anonymisation method explained in the following sections. The table below outlines the basic derived insights that the KB provides to the portal.

D6 - Raw Data Derived Insights	
RDI	RDI - 1. Most Visited Categories and Associated Co-Creation Groups RDI - 2. Most Interacted Co-Creation Group – User Joined/Viewed a Group RDI - 3. Most Interacted Co-Creation Group – Comments/Likes → Traction RDI - 4. Group Member Infographics based on Age Group, User Interests, etc. RDI - 5. Natural Language Processing on User Posts (most popular words in groups/word-cloud, most active users, etc.) RDI - 6. Sentiment and Affective Analysis on User Posts

Table 12 – RDI. Raw Data Derived Insights

A special case of Derived Insights is the Semantic Data Derived Analytics. These are the Data Analytics which are derived from the Semantic Annotation that is being conducted by the Semantic Pre-processing Layer that handles all data that are obtained from the Data Acquisition Layer.

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

D[1-6] – Semantic Data Derived Analytics	
SDI	SDI - 1. Suggest services related to co-creation group topic SDI - 2. Suggest social media accounts related to co-creation group topic SDI - 3. Suggest services related to co-creation group location SDI - 4. Suggest co-creation groups related to user interests SDI - 5. Suggest co-creation groups related to user location

Table 13 – SDI. Semantic Data Derived Analytics

The Data Mining and Analytics insights are often correlated and inter-connected with the SREs – SoCaTel Recommendation Engine Recommendations as these are defined in D4.3 – Context Aware Recommendation Engine Deliverable. The next sections present the derived insights in a tabular form by outlining the following information (to be used as a section explanatory guide for the tables that follow):

1. Presentation Layer View
    - Presentation Layer Views based on the defined Presentation Layer Views, as these are defined in *D3.4 – SoCaTel Platform Implementation deliverable in section 3.8.1 Presentation Layer Views.*
2. Associated SRE
    - The Associated SRE creates a symbolic link to the SREs defined in D4.3 – Context Aware Recommendation Engine defined recommendations. As it is already mentioned in this deliverable there are many recommendations which can be further improved and are satisfied with the use of Data Mining and Analytics. There are other SREs which use the Data Mining and Analytics' outcome to improve and enrich the ranking of the produced recommendations. Thus, we use this table field to associate and correlate the Data Mining and Analytics methods with the accompanied SREs
3. Description
    - Describes which Data Mining and Analytics techniques, implementation approach and metrics are used
4. Input
    - Input table field is used to give the reader an input that would be use by the Data Mining and Analytics algorithms
5. Output
    - Described output which would be given as output via the use of RESTful API responses, presented in the next sections.
6. Example
    - Demonstration of how each presented algorithm is implemented on example scenarios.

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

3.2 DATA ACQUISITION LAYER DERIVED INSIGHTS

DAI - 1.	Natural Language Processing on Twitter/Facebook Services
Presentation Layer View	Co-Creation Group Page
Associated SRE	SRE.1
Description	The relevant services for a co-creation group will be listed in the co-creation group page. The Natural Language Processing is then used to extract structured information from unstructured and/or semi-structured machine-readable information which are the Twitter tweets and Facebook posts.
Input	The information below is either taken from the GraphDB (KB1), by performing GraphQL, or from Elasticsearch, by performing RESTful API calls to the Elasticsearch API. • Co-Creation Group Data • Service Data (Twitter and Facebook)
Methodology	Based on the above input, we are using the Natural Language Processing to perform information extraction on the collected Twitter tweets and Facebook posts. More specifically, we are deploying information extraction to automatic extract structured information such as entities, relationships between entities and attributes describing entities from unstructured sources. We will use the following sub-modules to perform the Natural Language Processing, namely: • Fact Extraction • Entity Extraction • Sentiment Analysis • Polarity Granularity
Output	The DAI - 1 is available as a series of RESTful API calls that each performs one of the modules mentioned above. The complete list of RESTful API calls is available in section 5
Example	In a tweet or a facebook post, a named entity (location, organisation, etc.) can be identified and will be stored as structured data in the semantic repository.

Table 14 – DAI - 1. Natural Language Processing on Twitter/Facebook Services

DAI - 2.	Open Data → Data Analytics on Datasets based on pre-defined options
Presentation Layer View	Co-Creation Group Page
Associated SRE	SRE.1
Description	The portal allows the different stakeholders of the platform to add existing open data resources to a co-creation group. The process of obtaining the Open Data Resources is described in D4.1 - Data Standardization and Semantic Interoperability. Briefly, there is a set of RESTful API calls that the portal communicates with the help of an implemented library-connector. The RESTful API calls are deployed as part of the KB and the library-connector is a plug-n-play library that is part of the portal. The stakeholders of the portal are using the UI component of the library-connector to

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

	search on the pilot sites' open data portals for relevant open data resources. The retrieved information is then stored on the Semantic Repository and this information then becomes available for consumption and data mining and analysis.
Input	The information below is either taken from the GraphDB (KB1), by performing GraphQL, or from Elasticsearch, by performing RESTful API calls to the Elasticsearch API. • Co-Creation Group Data • Semantically Annotated Open Data Sources and Metadata
Methodology	Based on the above input, the consumed information from the D3 - Open Data – Statistical and Research Data Handler and D4 – Open Data – Governmental Data Handler are the metadata of an open data resource and the raw data of a resource. The supported raw datasets as already defined in D4.1, are .pdf, .csv, .txt and .xlsx. The raw data and their metadata are stored on both the semantic repository (KB1 - GraphDB) and on the Elasticsearch instance (more specifically on KB2.2 Analytical Data Repository). The metadata are stored on the Semantic Repository for quicker retrieval of the relevant Open Data Resources which is provided by the underlying semantic transformation layer. This is explained in previous deliverables. In addition, we are also performing some data analysis on the raw data such as standard deviation, min/max, sampling, aggregation, sum, mean, etc. This is provided in a set of RESTful API endpoints that are made available to the portal layer.
Output	The DAI - 2 is available as a series of RESTful API calls that each performs one of the modules mentioned above. The complete list of RESTful API calls is available in section 5
Example	A relevant Open Data Resource is being fetched using the set of RESTful API calls refined in section 5. This is provided by the semantic transformation layer by performing specific GraphQL queries on the layer. Then using the same RESTful API calls, we can select amongst the previously mentioned data analysis functions to perform analysis on the data. The analysis is then projected to the co-creation stakeholders and eventually help them obtain insights on underutilised services, etc.

Table 15 – DAI - 2. Open Data → Data Mining and Analytics on Datasets based on pre-defined options

3.3 Raw Data Derived Insights

RDI - 1.	Most Visited Categories and Associated Co-Creation Groups
Presentation Layer View	Search Page
Associated SRE	SRE.1
Description	The Most Visited Categories and Associated Co-Creation Groups are displayed in the Search Page of the SoCaTel portal. The top-N most visited and selected categories are displayed in the Search Page and they provide to a user with an information on what is being trending now in his area based his/her location, profession and spoken languages
Input	The information below is either taken from the GraphDB (KB1), by performing GraphQL, or from Elasticsearch, by performing RESTful API calls to the Elasticsearch API. • History Data

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

	• Co-Creation Group Data • User Data
Methodology	Based on the above input, we are using the User Data to derive all necessary information for a user based on his location, age group, profession, interests and skills and then we use this information to find similar co-creation groups that have the same characteristics. We then use the number of visits and joins of each co-creation group to extract each group's popularity ratio. The popularity ratio is then ranked based on the ratio and the results is then returned.
Output	The RDI.1 can be split to 6 different API calls and each is accommodating each used filter [age-range, location, profession, interests, skill, language]. Each output is a ranked list of co-creation groups that are given to the user as a potential suitable choice of co-creation groups to join based on location, age range, profession and spoken languages
Example	A user from Barcelona of age 30 which is a Social Worker and speaks English and French will be purposed to join one or more co-creation groups that takes place in Barcelona and it is about improving social wellbeing in Barcelona City, etc. This is purposed since similar users regarding this user's location, age range, profession and spoken language also participate in such co-creation groups.

Table 16 – RDI - 1. Most Visited Categories and Associated Co-Creation Groups

RDI - 2.	**Most Interacted Co-Creation Group – User Viewed/Joined a Group**
Presentation Layer View	Search Page
Associated SRE	SRE.1
Description	The Most Interacted Co-Creation Group insights are displayed in the Search Page of the SoCaTel portal. The top-N most joined groups are displayed in the Search Page and they provide to a user with the information on which co-creation group has the most interest over time.
Input	The information below is either taken from the GraphDB (KB1), by performing GraphQL, or from Elasticsearch, by performing RESTful API calls to the Elasticsearch API. • History Data
Methodology	Based on the above input, we are using the History Data to count how many users have joined each Co-Creation group over a period. The following periods will be used [just joined, last hour, last day, last week, last month]
Output	The RDI – 2 output is projected on the Search Page and the user can filter out the ranked results based on the time interval
Example	User selects a time period and a ranked recommendations list is projected to the screen providing information on which co-creation groups are trending based on groups' views and joins

Table 17 – RDI - 2. Most Interacted Co-Creation Group – User Viewed/Joined a Group

RDI - 3.	**Most Interacted Co-Creation Group – Comments/Likes → Traction**
Presentation Layer View	Search Page

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

Associated SRE	SRE.1
Description	The Most Interacted Co-Creation Group insight using the comments and likes provides an overview to the user, while using the Search Page of the SoCaTel portal on which co-creation group is attracting a lot of attention. The traction of a co-creation group is measured using the sum of exchanged messages over a period [last hour, last day, last week, last month].
Input	The information below is either taken from the GraphDB (KB1), by performing GraphQL, or from Elasticsearch, by performing RESTful API calls to the Elasticsearch API. • History Data • Group Data • Posts Data • Proposition Data • User Post Vote Data • User Post Proposition Vote Data
Methodology	Based on the Input above, we are using the History Data to count how many users have posted comments and users who have voted comments over a period. This projects the overall traction a co-creation group attracts, and it intrigues users to join a co-creation group and contribute to an already on-going discussion around a subject.
Output	The RDI – 3 output is projected on the Search Page and the user can filter out the ranked results based on the time interval
Example	User selects a time period and a ranked recommendations list is projected to the screen providing information on which co-creation groups are trending based on groups' comments and positive/negative votes

Table 18 – RDI - 3. Most Interacted Co-Creation Group – Comments/Likes - Traction

RDI - 4.	**Group Member Infographics based on Age Group, User Interests, etc.**
Presentation Layer View	Co-Creation Group Page
Associated SRE	SRE.1
Description	The Group Member Infographics are listed in a co-creation group and their purpose is to give the moderator and the users of a co-creation groups insights about the infographics of the participants.
Input	The information below is either taken from the GraphDB (KB1), by performing GraphQL, or from Elasticsearch, by performing RESTful API calls to the Elasticsearch API. • Group Data • User Data
Methodology	Based on the Input above, we are going to extract common users' interests, skills, language, age-group, etc. and create some statistics on the users. These statistics, in-time, will help the moderator to know the participating audience and potentially what is missing from the participants' list. This will aid the moderator to discover more

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

	assets to join a co-creation group to facilitate, enhance and enrich the co-creation process.
Output	The RDI – 4 output is projected on the Co-Creation Group page in the form of tables, pie charts, line-charts, etc.
Example	Co-Creation Group Users can observe a table with statistics about the co-creation group participants' age group, user interest, etc.

Table 19 – RDI - 4. Group Member Infographics based on Age Group, User Interests, etc.

RDI - 5. Natural Language Processing on User Posts (most popular words in groups/word-cloud, most active users, etc.)	
Presentation Layer View	Co-Creation Group Page
Associated SRE	SRE.1
Description	The Natural Language processing on User Posts adds an additional layer of Data Mining and Statistics to the user participants. The users that are participating to a co-creation group can obtain insights on the various mentioned topics, entities that are mentioned, most popular words in a form of word cloud. Moreover, the moderator and the rest of the users can get statistics like the most active users, the most active users per age-group, etc. The latter can lead to a top-commenter flag notation that can be used to assign to expert users a badge that indicate their high experience level in a co-creation group.
Input	The information below is either taken from the GraphDB (KB1), by performing GraphQL, or from Elasticsearch, by performing RESTful API calls to the Elasticsearch API. • History Data • Group Data • Posts Data • Proposition Data • User Post Vote Data • User Post Proposition Vote Data
Methodology	Based on the Input above, we collect information from the history data to extract and indicate the top commenters of a co-creation group by tracking when a user has commented a post or a reply to a co-creation group. Moreover, we are using the posts data to perform topic extraction and extract data mining and analytics to discover the most active users, the most active users per age-group, topics, entities. Some NLP techniques that will be used amongst others are the TF-IDF, Latent Semantic Indexing, etc.
Output	The RDI – 5 output is projected on the Co-Creation Group page in the form of tables, pie charts, line-charts, statistics, word-cloud, etc.
Example	Brief explanation was given on Methodology. An example can be, from a user post because the words "bus" and "elderly" are very common in a user's posts, we can conclude that the main concern of that user are these topics.

Table 20 – RDI - 5. Natural Language Processing on User Posts (most popular words in groups/word-cloud, most active users, etc.)

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

RDI - 6. Sentiment and Affective Analysis on User Posts	
Presentation Layer View	Co-Creation Group Page
Associated SRE	SRE.1
Description	The Sentiment and Affective analysis processing on User Posts adds an additional layer of Data Mining and Statistics to the user participants. The users that are participating to a co-creation group can obtain insights on the sentiment and affective analysis on what is being written to a co-creation group. This can help the users to get the overall users' sentiment polarity and subjectivity of a post and of the co-creation group in general. Additionally, it can provide insights to the moderator on which users are not behaving under the proper usage guidelines of the platform, which eventually will enable the moderator to ban them.
Input	The information below is either taken from the GraphDB (KB1), by performing GraphQL, or from Elasticsearch, by performing RESTful API calls to the Elasticsearch API. • History Data • Group Data • Posts Data • Proposition Data • User Post Vote Data • User Post Proposition Vote Data
Methodology	Based on the Input above, we collect information from the history, group, post, proposition, user post vote, user post proposition vote data to perform the Sentiment and Affective analysis so as to aggregate each posts' intermediate result to a unified sentiment and affective score that will indicate the over sentiment and affect of the entire co-creation group. We also perform the sentiment and affective analysis per each user to identify which users are misbehaving and we return this array of results to the moderator. We also extract the overall sentiment and affective analysis per age-group to extract the satisfaction of each participating age-group in the co-creation process
Output	The output consists of statistics, explained in the Description and Methodology, that are returned in a JSON format to the co-creation group and to the moderator. The results are shown in tabular form, line plots, charts, etc.
Example	Explained in the Description, Methodology and Output

Table 21 – RDI - 6. Sentiment and Affective Analysis on User Posts

3.4 SEMANTIC DATA DERIVED ANALYTICS

This section describes the insights that are generated by the data analytics module based on the semantic data available in the semantic repository (KB1) of the Knowledgebase. This repository contains data retrieved from the portal as well as from external sources, after being transformed into semantic format by the Semantic Pre-processing layer (S1). It integrates data coming from two main sources:

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

- Portal data: consists of all the data of the portal after being anonymized and sent to the Raw Data Repository (KB2.1).
- Data handlers output: consists of the data collected by the data handlers layer from the different types of external sources.

As such, this repository is an invaluable source of information that can be used to link the portal data with data coming from external sources and generate important recommendations and insights. The data analytics module accesses this data, stored in KB1, using a set of GraphQL API calls that are defined to facilitate the interaction with the repository. These API calls are fully defined and detailed in the Deliverable D4.4 – Data Access Interfaces.

The rest of this section explains all the Semantic-Data Derived Insights (SDIs) and how they can be useful for the users of the platform.

SDI - 1. Suggest services related to co-creation group topic	
Presentation Layer View	Co-Creation Group Page
Associated SRE	SRE.6
Description	This suggests services related to the topic(s) of each specific co-creation group. The aim behind this is to allow users to review existing services that might be relevant to the topic being discussed in the co-creation group, thus facilitating the process of studying existing alternatives when suggesting new services.
Input	The topic(s) of the co-creation group. This information is taken from the Raw Data repository (KB2.1), after being sent there from the portal side.
Methodology	When the details of a service are sent from the portal, they are transformed by the semantic pre-processing layer (S1) into a semantic format and then stored in the semantic repository (KB1). In particular, each service has a set of related "keywords" that define it. These keywords are transformed into semantic concepts and then the Linked Open Data Handler (D5) looks for synonyms, related concepts as well as translations to these keywords on external open data sources and stores them in KB1 as well. Once inside a co-creation group, the topic(s) of the group is used to look for similar services, both directly through the keywords of the service as well as through the synonyms and related concepts discovered by the Linked Open Data Handler (D5).
Output	List of suggested services related to the co-creation group topics.
Example	If a co-creation group has the topic "food delivery services", the data analytics module would use these words to look for services related to concepts "food delivery". Given that the knowledge base related concepts to synonyms as well, the returned services might not explicitly have these exact topics but rather might have keywords such as "nutrition", "home delivery" or "comida a domicilio" (meaning "food delivery" in Spanish).

Table 22 – SDI - 1. Suggest service paradigm within co-creation groups

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

SDI - 2.	Suggest social media accounts related to co-creation group topic
Presentation Layer View	Co-Creation Group Page
Associated SRE	SRE.5, SRE.6
Description	This suggests social media accounts related to the topic(s) of a particular co-creation group. The purpose of this is to allow users inside the group to view relevant content retrieved from the social media accounts of organizations registered on the platform.
Input	The topic(s) of the co-creation group. This information is taken from the Raw Data repository (KB2.1), after being sent there from the portal side.
Methodology	When an organisation registers a social media account (currently Twitter and Facebook accounts are supported), the Social Media Handlers (D2) retrieve posts and tweets from these registered accounts. The retrieved data is then sent to the semantic pre-processing layer (S1) which transforms the data into semantic format and stores it into the semantic repository (KB1). During this transformation process, the hashtags that appear inside the posts / tweets are transformed into semantic concepts. Afterwards, the Linked Open Data Handler (D5) looks for synonyms, related concepts as well as translations of these hashtags on external open data sources and stores them in KB1 as well. Once inside a co-creation group, the topic(s) of the group is used to look in KB1 for related social media accounts, both directly through the hashtags that appear in posts made by the account as well as through the synonyms and related concepts discovered by the Linked Open Data Handler (D5).
Output	List of social media accounts and posts related to the co-creation group's topics.
Example	A Tweet by a social media account containing the hashtag "#nutrition" would be suggested to a co-creation group with the topic "food delivery".

Table 20 – 0. Suggest service paradigm within co-creation groups

SDI - 3.	Suggest services related to co-creation group location
Presentation Layer View	Co-Creation Group Page
Associated SRE	SRE.4, SRE.5
Description	This suggests services related to the location of a particular co-creation group. Instead of basing the suggestions solely on the topics of the co-creation group, this module suggests services that are offered in the same locality of the group. These suggestions help users gain insights on how services are implemented within their desired location.
Input	The location of the co-creation group. This information is taken from the Raw Data repository (KB2.1), after being sent there from the portal side.
Methodology	When the details of a service are sent from the portal, they are transformed by the semantic pre-processing layer into a semantic format and then stored in the semantic repository (KB1). In particular, the location where the service is offered is also transformed and then the Linked Open Data Handler (D5) is used to retrieve additional information about the location from external sources (e.g. population, area, population density, etc.) and stores them in KB1 as well. Once inside a co-creation group, the location of the group is used to look for related services. Priority is given to services implemented directly within the same location of the co-creation group. However, in case no such services are found, the additional

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

	information of the location retrieved by D5 can be used to match the location of the co-creation group to locations with similar characteristics. The reasoning behind this is that, for instance, services have different requirements and are implemented differently in small towns compared to big cities. Thus, even if the service is implemented in another location, users can gain insights on how they can implement similar services in the location of the co-creation group.
Output	List of suggested services related to the co-creation group location.
Example	If a co-creation group has the location "Vilanova", the data analytics module looks for services implemented in the same locality. It also looks for services implemented in other similar localities, based on the location's area, population, etc. So for instance, a service implemented in "Tampere" is more related than one implemented in "Budapest", which is a much larger city.

Table 20 – SDI - 3. Suggest service paradigm within co-creation groups

SDI - 4. Suggest co-creation groups related to user interests	
Presentation Layer View	User Profile Page
Associated SRE	SRE.1
Description	This suggests co-creation groups related to the interests of a particular user. The aim behind this is to help the user in finding co-creation groups related to what they are interested in, as well as encourage them to engage with the platform in matters that interest them.
Input	The interests of the user, as declared by them when registering on the platform. This information is taken from the Raw Data repository (KB2.1), after being sent there from the portal side.
Methodology	When the details of a co-creation group are sent from the portal, they are transformed by the semantic pre-processing layer (S1) into a semantic format and then stored in the semantic repository (KB1). In particular, each co-creation group has one or more related topics that define it. These topics are transformed into semantic concepts and then the Linked Open Data Handler (D5) looks for synonyms, related concepts as well as translations to these topics on external open data sources and stores them in KB1 as well. When a user registers, their data is anonymized and sent to the Raw Data repository (KB1.2). Afterwards, the data analytics module uses the user's interests to find related co-creation groups, both directly through the topics of the group as well as through the synonyms and related concepts discovered by the Linked Open Data Handler (D5).
Output	List of suggested co-creation groups related to a user's interests.
Example	Co-creation groups about "disabled parking space" would be suggested to a user interested in "accessibility". This is due to the fact that the knowledge base contains synonym and related concepts to the topics of the co-creation groups, so "disable" is mapped with "accessibility".

Table 21 – SDI - 4. Inclusive Search of Co-Creation Groups

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

SDI - 5. Suggest co-creation groups related to user location	
Presentation Layer View	User Profile Page
Associated SRE	SRE.1, SRE.3
Description	This suggests co-creation groups that are active in the location of a particular user. Instead of basing the suggestions solely on the topics of the co-creation group, this module suggests groups that are offered in the same locality of the user. These suggestions help users gain insights on what co-creation groups are trending in their location.
Input	The location of the user. This information is taken from the Raw Data repository (KB2.1), after being sent there from the portal side.
Methodology	When the details of a co-creation group are sent from the portal, they are transformed by the semantic pre-processing layer (S1) into a semantic format and then stored in the semantic repository (KB1). In particular, the location of the group is also transformed and then the Linked Open Data Handler (D5) is used to retrieve additional information about the location from external sources (e.g. population, area, population density, etc.) and stores them in KB1 as well. When a user registers on the platform, their location is used to look for active co-creation group within the same location. Depending on the type of the user (user, organisation, moderator), more suggestions can be generated based on the additional data retrieved by D5. For instance, if the user is an organization, the additional information of the location retrieved by D5 can be used to match the location of the user to locations with similar characteristics. The reasoning behind this is that, for instance, co-creation groups active in locations with similar population, area, etc., might have relevant information for the organisation in order to see what users need in other similar localities.
Output	List of suggested co-creation group active in the user's location.
Example	The co-creation groups in "Vilanova" are suggested to a user based in the same locality. Furthermore, if the user is an organisation, groups in similar localities, based on the location's area, population, etc., are also suggested. So for instance, a co-creation group in "Tampere" is more related than one in "Budapest", which is a much larger city.

Table 23 – 0. Inclusive Search of Co-Creation Groups

4 SoCaTel Recommendation Engine

Recommendation Engine is a software that, uses available data, analyses them and produces suggestions regarding content on the website that might interest the user. A more detailed description and explanation of the different type of recommendations and how the work can be found in "D4.3 Context Aware Recommendation Engine".

The view of this document regarding the SoCaTel recommendation engine is technical, therefore referencing the D4.3 might be required for acquiring a clearer understanding on how these systems work and what is their algorithmic composition.

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

A thorough example of how the recommendation engine works along with input output and training dataset can be found in section 4.1.1.

Furthermore, on D4.3 there is section “4.3.2 Identification of Service Needs”, which gives a description of how the recommendation engine will be used in the SoCaTel platform in different “scenarios”. In this section, we describe how these are realised using our recommendation engine. All the details regarding the implementation of the parts of the SREs that required recommendations from the recommendation engine can be found in section 4.2.

4.1 RECOMMENDATION ENGINE INFRASTRUCTURE

While the full creation of a bespoke recommendation engine is possible, its optimisation is a complicated and time-consuming task. This is however not necessary since, customizable solutions in the form of tools and frameworks already exist that can cover the needs of the SoCaTel platform and there is no need to reinvent these. A very detailed description of these tools and frameworks can be found in D4.3 in “Section 3: Existing Technologies and Frameworks – Review”. As it was also mentioned in that section; after a close comparison between these existing tools, it was decided that PredictionIO was the best fit for tackling the SoCaTel platform challenges.

For a fast recap, PredictionIO is a deployable machine learning server, build on a state-of-the-art stack of open source material. PredictionIO requires additional software as dependencies, some of those are:

- Apache Hadoop
- Apache Spark
- Java SE Development Kit

Moreover, it requires one or more of the following data storages:

- PostgreSQL
- MySQL
- Apache HBase + Elasticsearch

The implementation of PredictionIO is described from Figure 2 found in the PredictionIO website[4]

In more detail:

1. **Import Events:** All events are collected from the application and sent either in real time or in a batch in the event server. Since in the recommendation engine

[4] PredictionIO Website: http://predictionio.apache.org/

This project has received funding from the European Union’s Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

scenarios, most of the time these datasets contain events (user bought an item, user rated a movie, user liked a comment) we can distinguish this data by calling them Event Data.

2. **Data Preparation and Training:** A data preparator performs feature selection on data if necessary and outputs prepared data. The prepared data are then used from the algorithm for training a model. This model is what will be used for performing recommendations.
3. **REST queries:**
    a. The Recommendation system is now ready to accept queries using REST API.
    b. The Recommendation system returns the predicted result in a JSON format.

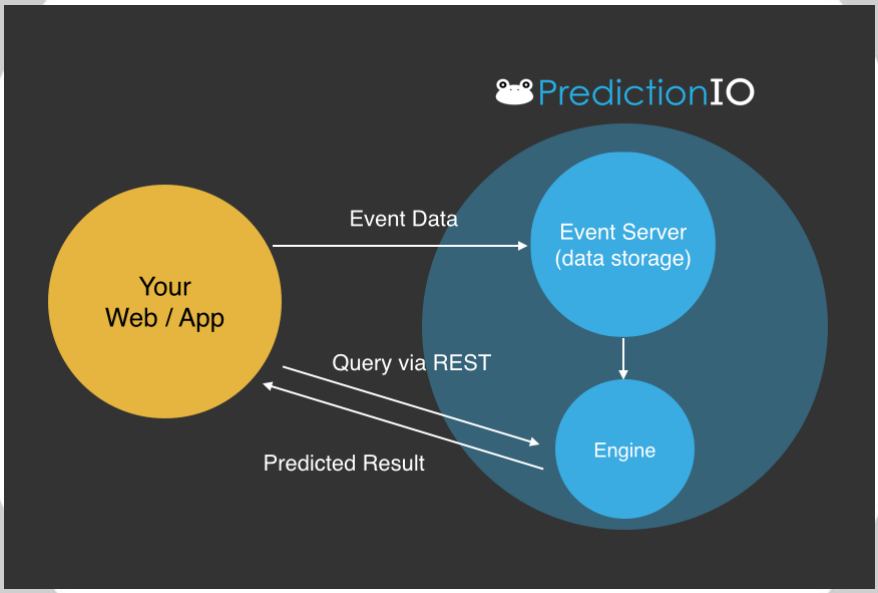

Figure 2 – PredictionIO architecture[5]

Another feature of PredictionIO that also helped on choosing it as the SoCaTel platform framework is the templates. Templates are open source implemented solutions, containing all the "ingredients" for creating machine learning systems. More specifically, there are templates for:

- Recommendation Systems
- Classification
- Regression
- Natural Language Processing
- Clustering

[5] PreditionIO – A Quick Intro: https://predictionio.apache.org/start/

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

- Similarity
- Other more specialised algorithms

Thus, PredictionIO can also help tackle other challenges identified in the platform.

4.1.1 RECOMMENDATION SCENARIO AND DATASET

For demonstrating the capabilities of PredictionIO lets consider the following scenario influenced from one of the SREs mentioned in D4.3:

> "The user has already joined a few co-creation groups of interest and has already interacted with some of them, left some that wasn't what expected and got heavily involved with others. The platform would like to further engage the user by recommending more co-creation groups that would be closer to the ones that got the user involved."

This type of recommendation is what is typically used in most of movie rating websites. A movie rating website requires a recommendation system for recommending movies to users. The movie recommendation should be specialised to the user, to be a movie that the user will enjoy. For the purpose of these scenarios, User-to-User collaborative recommendation algorithm is used.

User-to-User collaborative recommender system is a method of recommendation that correlates user A with other users based on similar behaviour (rating high the same movies) and produces recommendation based on what the other corelated users rated high that user A hasn't rated. The algorithm is more explained in detail in D4.3.

A main difference between the movies in the movie rating website and the co-creation group in SoCaTel platform is that co-creation group lack a defined rating system. Nevertheless, we can use the behaviour of the platform user from its activity in a co-creation group and thus create an internal platform rating system that automatically gives a rating to a group from a user based on that user's activity and behaviour. The activity is measured from:

- Likes.
- Replies.
- Comments.
- Popularity of comment (likes a posted comment receives).

An example of how this rating could work is:

- 1-Star (Very Negative): The User joins a group and without any interaction leaves the group.
- 2-Star (Negative): The User joins a group, interacts a bit and then leaves the group.
- 3-Star (Neutral): The User simply ignores a group and never joins it.

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

- 4-Star (Positive): The User joins a group and interacts a bit.
- 5-Star (Very Positive): The User joins a group and is heavily involved.

First, for the sake of the example, a fake dataset is taken. Suppose that this is a movie rating dataset for a movie recommendation system. In this dataset there are 3 values:

1. **The User ID:** This is what hypothetically will be the user id that will correspond to a user of the platform in the database.
2. **The Item ID:** In the example situation, the item id can correspond to a movie.
3. **User Rating**: This is a numerical value from 1-5 of how the user rated the movie (item).

Movie Rating Dataset	
Dataset (UserID::ItemID::Rating)	1::81::5 1::82::5 1::85::1 1::90::5 2::82::5 2::81::4 2::85::1 2::9::1 2::90::5 2::69:5 3::0::1 3::1::1 3::2::1 3::7::3 3:90:3 4::81::4 4::82::4 4::85::1 4::90::5 4::10::1 4::11::1 4::69::5 5::9::1 5::81::1 5::82::1 5::8::1

Table 24 – Example Event Data for a movie rating site

4.1.2 IMPORT EVENTS

In general, there are two different “schools” for importing data in the Event server. It is done either “one by one” (when the event occurs) or, it is stored in a file (i.e.

This project has received funding from the European Union’s Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

".txt" or ".json") and added as a "batch" at a time configured from the server. Both ways have positives and negatives, therefore each should of them is used in specific situations depending on the activity event that is required to be imported.

PredictionIO engine supports four different languages for adding the event data on the Event Server. They can be added using:

- Raw HTTP
- PHP SDK
- Python SDK
- Ruby SDK
- Java SDK

For our system we decided to use the Python SDK with some exceptions where Raw HTTP is used.

Table 25 demonstrates how an HTTP call can be used to import data in the Event server. In this code snippet, the HTTP call POSTs on the Event server (localhost:7070 by default) passing as variable the access key associated with the application. The call also defines that the event is of type "rate" performed by a "user" with id "u0" on an "item" with id "i0". Finally, the rating is imported as part of a variable called "properties" and in this situation the rating is "5".

Raw HTTP Code

$ curl -i -X POST http://localhost:7070/events.json?accessKey=$ACCESS_KEY \
-H "Content-Type: application/json" \
-d '{
"event" : "rate",
"entityType" : "user",
"entityId" : "u0",
"targetEntityType" : "item",
"targetEntityId" : "i0",
"properties" : {
"rating" : 5
}
"eventTime" : "2014-11-02T09:39:45.618-08:00"
}

Table 25 – HTTP Call to Import Event Data

For comparison with the HTTP call and demonstration of how the Python SDK works, Table 26 presents the python code for an import call, identical with the one presented in Table 25.

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

Python SDK Code
import predictionio client = predictionio.EventClient(access_key=<ACCESS KEY>, url=<URL OF EVENTSERVER>, threads=5, qsize=500) # A user rates an item client.create_event(event="rate", entity_type="user", entity_id=<USER ID>, target_entity_type="item", target_entity_id=<ITEM ID>, properties= { "rating" : float(<RATING>) })

Table 26 – Python SDK Event Data Import

The data are now ready for use by the PredictionIO engine. The following sub-section (4.1.3) describes in detail the steps followed by the engine.

4.1.3 DATA PREPARATION AND TRAINING

Following the PredictionIO architecture (Figure 2), the engine takes the data imported in the Event server and creates/trains our prediction model. More specifically, the engine follows the DASE architecture, taking its name from the components:

- Data (Data source and Data Preparator)
- Algorithm(s)
- Serving
- Evaluator

4.1.3.1 Data Source and Data Preparator

During data preparation, data is prepared by 2 components the:

- **Data Source:** Responsible for reading and selecting data from the Event Server, giving as output Training Data.

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

Data Source Code[6]

```
case class DataSourceParams(appName: String) extends Params

class DataSource(val dsp: DataSourceParams)
  extends PDataSource[TrainingData,
      EmptyEvaluationInfo, Query, EmptyActualResult] {

  @transient lazy val logger = Logger[this.type]

  def getRatings(sc: SparkContext): RDD[Rating] = {

    val eventsRDD: RDD[Event] = PEventStore.find(
      appName = dsp.appName,
      entityType = Some("user"),
      eventNames = Some(List("rate", "buy")), // read "rate" and "buy" event
      // targetEntityType is optional field of an event.
      targetEntityType = Some(Some("item")))(sc)

    val ratingsRDD: RDD[Rating] = eventsRDD.map { event =>
      val rating = try {
        val ratingValue: Double = event.event match {
          case "rate" => event.properties.get[Double]("rating")
          case "buy" => 4.0 // map buy event to rating value of 4
          case _ => throw new Exception(s"Unexpected event ${event} is read.")
        }
        // entityId and targetEntityId is String
        Rating(event.entityId,
          event.targetEntityId.get,
          ratingValue)
      } catch {
        case e: Exception => {
          logger.error(s"Cannot convert ${event} to Rating. Exception: ${e}.")
          throw e
        }
      }
      rating
    }.cache()

    ratingsRDD
  }

  override
  def readTraining(sc: SparkContext): TrainingData = {
    new TrainingData(getRatings(sc))
  }
}
```

Table 27 – Data Source Code

[6] PredictionIO DASE: https://predictionio.apache.org/templates/recommendation/dase/

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

- **Data Preparator:** Takes as input the Training Data and performs any necessary feature selection and data processing. This will output the Prepared Data, ready to be used in an algorithm.

Data Preparator Code[7]

```
class Preparator
  extends PPreparator[TrainingData, PreparedData] {

  def prepare(sc: SparkContext, trainingData: TrainingData): PreparedData = {
    new PreparedData(ratings = trainingData.ratings)
  }
}

class PreparedData(
  val ratings: RDD[Rating]
) extends Serializable
```

Table 28 – Data Preparator Code

4.1.3.2 Algorithms

The algorithmic part is mostly defined by the template used, although this is not always necessary. The algorithmic part uses the data extracted from the Data Preparation for performing training and preparing the predictions.
The algorithmic part usually uses two methods:

- **Train:** Used for training a predictive model
- **Predict:** Used when called from a query HTTP call. The default call should look like: http://localhost:8000/queries.json. The file *queries.json* should be a json file containing the query required for prediction.

Following up the example scenario from 4.1.1 with the movie rating, a query for movie recommendations should return to the user movies that users with similar likes and dislikes rated high but the user haven't yet rated (User Collaborative Recommendation). Therefore, an appropriate query to get this recommendation could look like Table 29, where "**user**" is the user id and "**num**" is the number of recommendations to be returned (number of recommended movies).

Query.json

```
{
"user": "0",
"num": "1"
}
```

Table 29 – Recommendation Query

[7] PredictionIO DASE: https://predictionio.apache.org/templates/recommendation/dase/

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

4.1.3.3 Serving

Serving part of the architecture is the combination of multiple predicted results (if more than one engine/algorithm is used) into one and the return as the final predicted result.

Table 30 demonstrates how the “serve” method of class “Serving” processes predicted result.

Result Serving Code

```
class Serving
  extends LServing[Query, PredictedResult] {

  override
  def serve(query: Query,
    predictedResults: Seq[PredictedResult]): PredictedResult = {
    predictedResults.head
  }
}
```

Table 30 – Serving Class

Concluding, the results will be served in json format. For the results presented in Table 31, we consider as dataset the event data presented in Table 24, as an algorithm we used User-to-User collaborative recommendation, and the query for retrieving the data is presented in Table 29.

Query Results

```
{
        "itemScores":
        [
                {
                "item": "itemid",
                "score":  "item_score"
                }
        ]
}
```

Table 31 – Query Results

4.1.3.4 Evaluation

The last part of DASE is evaluation. PredictionIO engine is embodied by a set of parameters. These parameters are the ones to determine which algorithm is used. Therefore, it is required to have a sort of testing to determine which are the best sets of parameters. According to PredictionIO documentation: “The

This project has received funding from the European Union’s Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

evaluation module streamlines the process of tuning the engine to the best parameter set and deploy it."[8].

PredictionIO offers an evaluation module, that allows the developer to streamline the process of testing many details in engine parameters and deploy the best one out, by using statistically thorough cross-validation methods.

The PredictionIO evaluation module orbits around two key components. The target which is the Engine itself and the Evaluator.

What the Evaluator does, is to join the sequence of Query, Predicted Result and Actual Result and evaluates the quality of the engine. A representation of this can be seen in Figure 3.

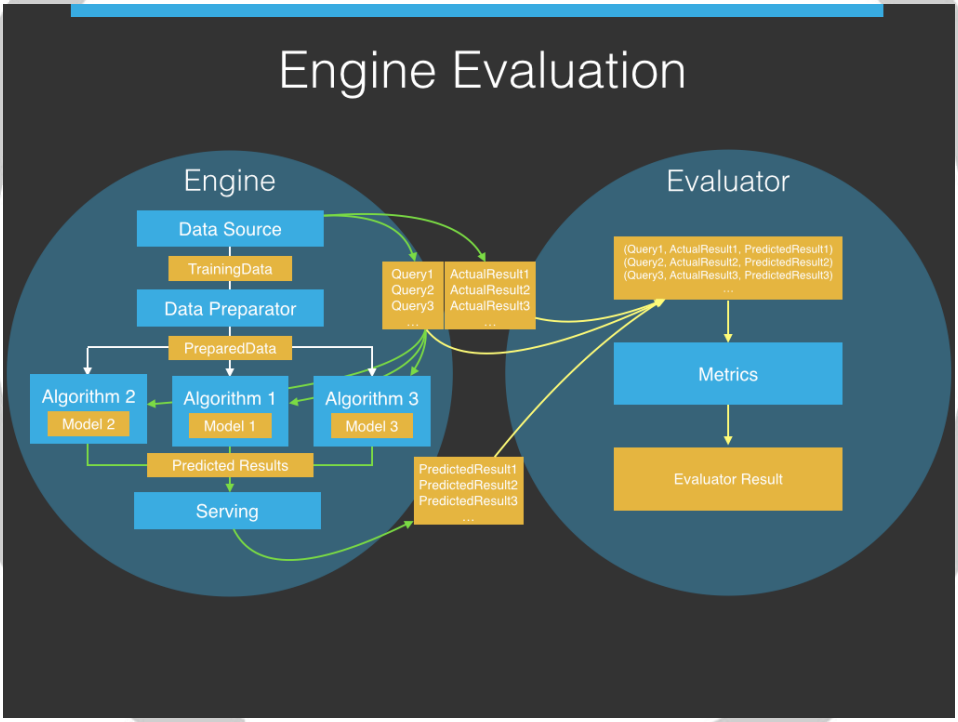

Figure 3 – Engine Evaluation

Calling the evaluation module requires two mandatory parameters.

1. Evaluation object
   - Tells PredictionIO which metric we use for the evaluation
2. Engine Parameters
   - A list of engine parameters to test the engine against.

[8] PredictionIO Evaluation Explained:
https://predictionio.apache.org/templates/recommendation/evaluation/

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

4.2 THE SOCATEL PLATFORM IMPLEMENTATION

Using as reference the examples demonstrated on Section 4.1, Section 4.2 describes how the PredictionIO framework is developed to be the SoCaTel platform recommendation engine. In this section, we describe how the SREs (detailed described on D 4.3) were implemented.

Following are the recommendation tables, presenting specific information for each recommendation (REC). The table rows are separated as:

1. Description
    - Describes which recommendation techniques, implementation approach and metrics are used
2. Associated SRE
    - The Associated SRE creates a symbolic link to the SREs defined in D4.3 – Context Aware Recommendation Engine defined recommendations. As it is already mentioned in this deliverable there are many recommendations which can be further improved and are satisfied with the use of Data Mining and Analytics. There are other SREs which use the Data Mining and Analytics' outcome to improve and enrich the ranking of the produced recommendations. Thus, we use this table field to associate and correlate the Data Mining and Analytics methods with the accompanied SREs
3. Dataset
    - Describes what are the requirements for the dataset, so it can train the recommendation model properly. The section also contains a small description regarding each attribute individually.
4. Input
    - Input table field is used to give the reader an input that would be use by the Recommendation Engine
5. Methodology
    - The name of algorithms used for producing the recommendation
6. Output
    - Described output which would be given as output via the use of RESTful API responses, presented in the next sections.
7. Scenarios

REC - 1. Recommendations to Users Based on Similar Users	
Description	Recommendations to Users Based on Similar Users correlates the subject user with other users based on similar preferences towards the same items. The system then identifies items that the subject user hasn't explore yet, but they were "liked" by the "similar" users and ranks them based on a score given by the "similar" users.
Associated SRE	SRE.3, SRE.4

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

Dataset	The dataset used for training this Recommender consists of three values. Two of type "ID" and one of type "integer". ➢ UserID: An id correlated with a user of the SoCaTel platform ➢ ItemID: An id correlated with an item of the platform (Co-Creation group, Service, etc.) ➢ Rating: A value defined by the activity of the user in regard to the item. The rating takes value from 1-5 where 1 is the lowest and suggests the user disliked the item and 5 the maximum which suggests that the user really liked the item. In the situations where the item doesn't have a rating field, the user behaviour can be used to extract a rating. For example: Entering a co-creation group and interacting with it (likes, comments, etc) then it can be considered as a high rating. Low rating can be considered behaviour where the user leaves a co-creation group with no interaction. Example: userid::itemid::rating – 123821::1890::4
Input Query	The request only requires a Userid and a value that corresponds to the number of recommendations to be returned Example: {"userid": "123821", "num": "2"}
Methodology	**"User-to-User" collaborative recommendation system**. (More details regarding the methodologies can be found in D 4.3).
Output	A JSON file with the recommended items and their rating value. Example: {["itemid": "4151", rating: "4.31321], ["itemid": "154", rating: "4.12142"]}
Scenarios	This recommender system is used in situations where there is a lot of user activity and interaction that can be corelated with other users with similar likes and dislikes. In SoCaTel platform, this is used for: ➢ Recommending co-creation group to a user ➢ Recommending service to a user

Table 32 – REC - 1. Recommendations to Users Based on Similar Users

REC - 2. Recommendations to Users Based on Similar Items	
Description	Recommendations to Users Based on Similar Items utilises a model of relative items. Thus, when subject user likes an item, the recommendation system proposes relative items.
Associated SRE	SRE.3, SRE.4, SRE.6
Dataset	The dataset used for training this Recommender consists of three values. Two of type "ID" and one of type "integer". ➢ UserID: An id correlated with a user of the SoCaTel platform ➢ ItemID: An id correlated with an item of the platform (Co-Creation group, Service, etc.) The dataset for REC2 is similar with the dataset used in REC1; the dataset needs to contain a user and an item that the user interacted with. The difference though, is that the dataset will be used to correlate items between them and thus create a new dataset with correlation between items. The rating value for ranking the recommendation is calculated by how often two items are correlated together.

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

	Example: userid::itemid
Input Query	The request only requires a Userid and a value that corresponds to the number of recommendations to be returned Example: {"userid": "123821", "num": "2"}
Methodology	"Item-to-Item" collaborative recommendation system. (More details regarding the methodologies can be found in D 4.3).
Output	A JSON file with the recommended items and their rating value. Example: {["itemid": "4151", rating: "4.31321], ["itemid": "154", rating: "4.12142"]}
Scenarios	This recommender system, due to the way it works, it doesn't require a lot of data to be able to perform well. Thus it is used early on the publication of the platform and until the number of entries on datasets increase or in situations where the amount of data for training are insufficient for other types of recommendations , this is used for: ➢ Recommending co-creation group to a user ➢ Recommending service to a user ➢ Recommending organizations to a group moderator user

Table 33 – REC - 2. Recommendations to Users Based on Similar Items

REC - 3. Recommendations to Users Based on Similar Users and Similar Items	
Description	Offer recommendations using a hybrid model, combining recommendations based on similar users (Rec – 1) and similar items (Rec – 2).
Associated SRE	SRE.3, SRE.4, SRE.6
Dataset	The dataset used for training this Recommender consists of three values. Two of type "ID" and one of type "integer". ➢ UserID: An id correlated with a user of the SoCaTel platform ➢ ItemID: An id correlated with an item of the platform (Co-Creation group, Service, etc.) ➢ Rating: A value defined by the activity of the user in regard to the item. The rating takes value from 1-5 where 1 is the lowest and suggests the user disliked the item and 5 the maximum which suggests that the user really liked the item. In the situations where the item doesn't have a rating field, the user behaviour can be used to extract a rating. For example: Entering a co-creation group and interacting with it (likes, comments, etc) then it can be considered as a high rating. Low rating can be considered behaviour where the user leaves a co-creation group with no interaction. Example: userid::itemid::rating – 123821::1890::4
Input Query	The request only requires a Userid and a value that corresponds to the number of recommendations to be returned Example: {"userid": "123821", "num": "2"}
Methodology	This can be considered a **hybrid collaborative recommendation system**, since it combines both "User-to-User" and "Item-to-Item" recommendations. (More details regarding the methodologies can be found in D 4.3).

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

Output	A JSON file with the recommended items and their rating value. Example: {["itemid": "4151", rating: "4.31321], ["itemid": "154", rating: "4.12142"]}
Scenarios	This recommender system is considered the most accurate between the collaborative models that is consisted of (User to User collaborative and Item to Item collaborative): ➢ Recommending co-creation group ➢ Recommending service to a user ➢ Recommending service paradigm within co-creation groups

Table 34 - REC - 3 Recommendations to Users Based on Similar Users and Similar Items

REC - 4. Recommendations to Users Based on Content	
Description	Offer recommendations to a User, based on items found to be similar with those of interest to the user.
Associated SRE	SRE.5, SRE.6, SRE.7
Dataset	The dataset used for training a Content Based recommendation engine needs to contain the item_id and a set of words as description for the item: ➢ ItemID: An id correlated with an item in the database. In our platform this item can be a co-creation group, a service or a resource. ➢ Words: A set of words (or tags) describing the item, gathered from content in the SoCaTel platform and from pages from related social media pages of Facebook and Twitter.
Input Query	
Methodology	**Content Based Recommendation System**. (More details regarding the methodologies can be found in D 4.3).
Output	A JSON file with the recommended items and their rating value. Example: {["itemid": "4151", rating: "4.31321], ["itemid": "154", rating: "4.12142"]}
Scenarios	This recommendation system uses natural language and semantically annotated data, to produce recommendations based on the content of an item. Thus, it should be used in situations where the rest of the recommendations cannot be used because of the importance of the content. These scenarios can be: ➢ Recommend organizations that work on related services to group moderator ➢ Suggest services to users of a co-creation group ➢ Suggest resources to users of a co-creation group

Table 35 – REC - 4. Recommendations to Users Based on Content

5 RESTful API Definitions

This chapter outlines and the basic set of RESTful API endpoints that the portal uses to interact with the KB and obtain insights from the Data Mining and Analytics Engine and recommendations from the Recommendation Engine.

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

The next section presents the derived RESTful endpoints in a tabular form by outlining the following information:

1. Type
    - Describes the type of the RESTful API Endpoint. The following options are defined:
        i. Data Mining and Analytics Insights
        ii. User to User Collaborative Recommendation
        iii. Item to Item Collaborative Recommendation
        iv. Hybrid Collaborative Recommendation
        v. Content Based Recommendation
2. Reference
    - Describes the
3. HTTP Method
    - Can either be [HTTP GET, HTTP POST, etc.]
4. URL Params
    - List of URL parameters that are send with the request to parameterise the input of the endpoint and obtain relevant addressed information
5. Output
    - Refers to the output already defined in the corresponding list of the following:
        i. DAI – Data Acquisition Layer Derived Insights
        ii. RDI – Raw Data Derived Insights
        iii. SDI – Semantic Data Derived Insights
        iv. REC – SoCatel Recommendation Engine Recommendations

5.1 Data Mining and Analytics Engine API Endpoints

All API calls are set agains endpoint https://socatel-rec.ozwillo-preprod.eu

5.1.1 Data Acquisition Layer Derived Insights API Endpoints

Type	Data Mining and Analytics Insights
Reference	DAI - 1
HTTP Method	GET
API Call	NaturalLangProc/api_endpoint/service_id?language

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

URL Params.	➢ api_endpoint: There are a series of supported API endpoints and more will be added in the future upon need. A few of them are: ○ fact_extraction ○ entity_extraction ○ relation_extraction ○ sentiment_analysis ○ polarity_granularity ➢ service_id ➢ language: For language specific groups, to identify the language request. Default is English
Output	As explained above in DAI - 1

Table 36 - API Definition - Services' Natural Language Processing

Type	Data Mining and Analytics Insights
Reference	DAI - 2
HTTP Method	GET
API Call	OpenData/search?text,language
URL Params.	➢ language: Language corresponds to the open data portal that the search will be conducted. ➢ Text: Actual text that will be used to search across the entire open data listing
Output	As explained above in DAI - 2

Table 37 - API Definition – Research, Governmental and Open Data Search Functionality

Type	Data Mining and Analytics Insights
Reference	DAI - 2
HTTP Method	POST
API Call	OpenData/save?open_data_id,language,operation
URL Params.	➢ open_data_id: Open Data unique Identification Number ➢ language: Language corresponds to the open data portal that the search will be conducted. ➢ operation: sum, standard deviation, min, max, etc. ** This REST endpoint will save the open data resource to the KB. The defined operation is applied on the obtained data and is then presented to a co-creation group page as data mining and analytics insights
Description	As explained above in DAI - 2

Table 38 - API Definition - Save Research, Governmental and Open Data

Type	Data Mining and Analytics Insights
Reference	DAI - 2
HTTP Method	GET
API Call	OpenData/get?open_data_id,language
URL Params.	➢ open_data_id: Open Data unique Identification Number

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

	➢ language: Language corresponds to the open data portal that the search will be conducted. ** This REST endpoint returns the actual open data resource and the applied outcome of the previously selected operation
Description	As explained above in DAI - 2

Table 39 - API Definition - Retrieve Research, Governmental and Open Data

5.1.2 RAW DATA DERIVED INSIGHTS API ENDPOINTS

Type	Data Mining and Analytics Insights
Reference	RDI - 1
HTTP Method	GET
API Call	MostVisitedCat_AssociatedGroups/
URL Params.	-
Description	As explained above in RDI - 1

Table 40 - API Definition - Most Visited Categories and Associated Co-Creation Groups

Type	Data Mining and Analytics Insights
Reference	RDI - 2
HTTP Method	GET
API Call	MostInteractedCoGroup_UserVJGroup/
URL Params.	-
Description	As explained above in RDI - 2

Table 41 - API Definition - Most Interacted Co-Creation Group – User Viewed/Joined a Group

Type	Data Mining and Analytics Insights
Reference	RDI - 3
HTTP Method	GET
API Call	MostInteractedCoGroup_Traction?co_group_id
URL Params.	➢ co_group_id: The unique identification number of a co-creation group
Description	As explained above in RDI - 3

Table 42 - API Definition - Most Interacted Co-Creation Group – Comments/Likes → Traction

Type	Data Mining and Analytics Insights
Reference	RDI - 4
HTTP Method	GET
API Call	GroupMember_Infographics/?co_group_id
URL Params.	➢ co_group_id: The unique identification number of a co-creation group
Description	As explained above in RDI - 4

Table 43 - API Definition - Group Member Infographics based on Age Group, User Interests, etc.

Type	Data Mining and Analytics Insights
Reference	RDI - 5
HTTP Method	GET
API Call	NLP_Data_Stats/?co_group_id
URL Params.	➢ co_group_id: The unique identification number of a co-creation group

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

Description	As explained above in RDI - 5

Table 44 - API Definition - Natural Language Processing on User Posts (most popular words in groups/word-cloud, most active users, etc.)

Type	Data Mining and Analytics Insights
Reference	0
HTTP Method	GET
API Call	SentimentAffectAnalysis/?co_group_id
URL Params.	➢ co_group_id: The unique identification number of a co-creation group
Description	As explained above in 0

Table 45 - API Definition - Sentiment and Affective Analysis on User Posts

5.1.3 Semantic Data Derived Insights API Endpoints

Type	Semantic Data Derived Analytics
Reference	SDI - 1
HTTP Method	GET
API Call	SemanticServiceTopicToCCgroup?group_id?topic
URL Params.	➢ group_id: An id corresponding to a co-creation group in the SoCaTel platform ➢ topic: A categorized tag, regarding an ítem (discussion, co-creation group, service).
Description	As explained above in SDI - 1

Table 46 - API Definition - Suggest services related to co-creation group topic

Type	Semantic Data Derived Analytics
Reference	SDI - 2
HTTP Method	GET
API Call	SemanticOSMAccountToCCgroup?group_id?topic
URL Params.	➢ group_id: An id corresponding to a co-creation group in the SoCaTel platform ➢ topic: A categorized tag, regarding an ítem (discussion, co-creation group, service).
Description	As explained above in SDI - 2

Table 47 - API Definition - Suggest social media accounts related to co-creation group topic

Type	Semantic Data Derived Analytics
Reference	SDI - 3
HTTP Method	GET
API Call	SemanticServiceToCCgroupLocation?group_id?topic?location
URL Params.	➢ group_id: An id corresponding to a co-creation group in the SoCaTel platform ➢ topic: A categorized tag, regarding an ítem (discussion, co-creation group, service).

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

	➢ location: A geographical location of a co-creation group or a service
Description	As explained above in SDI - 3

Table 48 - API Definition - Suggest services related to co-creation group location

Type	Semantic Data Derived Analytics
Reference	SDI - 4
HTTP Method	GET
API Call	SemanticCCgroupFromInterest?user_id
URL Params.	➢ user_id: Id that correlates to a SoCaTel platform user
Description	As explained above in SDI - 4

Table 49 - API Definition - Suggest co-creation groups related to user interests

Type	Semantic Data Derived Analytics
Reference	0
HTTP Method	GET
API Call	SemanticCCgroupFromLocation?user_id
URL Params.	➢ user_id: Id that correlates to a SoCaTel platform user
Description	As explained above in 0

Table 50 - API Definition - Suggest co-creation groups related to user location

5.2 SoCaTel Recommendation Engine API Endpoints

Type	User to User Collaborative Recommendation
Reference	REC - 1
HTTP Method	GET
API Call	Recommend_Collab_user/user_id?rec_nums?platform_screen&language”
URL Params.	➢ user_id: Id that correlates to a SoCaTel platform user. ➢ rec_nums: Number of items returned from recommendation (Top 3, Top 5 etc.) ➢ platform_screen: Id that specifies from which screen is the recommendation requested from. This will also define the type of “item” requested. ➢ language: For language specific groups, to identify the language request. Default is English
Output	As explained above REC - 1

Table 51 - API Definition - User to User Collaborative Recommendation

Type	Item to Item Collaborative Recommendation
Reference	REC - 2
HTTP Method	GET
API Call	Recommend_Collab_item/item_id?rec_nums?patform_screen&language”

This project has received funding from the European Union’s Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

URL Params.	➢ Ítem_id: Id that correlates to an item in a corresponding database (groups, services, scientific papers etc.) ➢ rec_nums: Number of items returned from recommendation (Top 3, Top 5 etc.) ➢ platform_screen: Id that specifies from which screen is the recommendation requested from. This will also define the type of "item" requested. ➢ language: For language specific groups, to identify the language request. Default is English
Output	As explained above REC - 2

Table 52 - API Definition - Item to Item Collaborative Recommendation

Type	Hybrid Collaborative Recommendation
Reference	REC - 3
HTTP Method	GET
API Call	Recommend_Hybrid/user_id?item_id?rec_nums?patform_screen&language"
URL Params.	➢ user_id: Id that correlates to a SoCaTel platform user. ➢ Ítem_id: Id that correlates to an item in a corresponding database (groups, services, scientific papers etc.) ➢ rec_nums: Number of items returned from recommendation (Top 3, Top 5 etc.) ➢ platform_screen: Id that specifies from which screen is the recommendation requested from. This help define the type of "item" requested and the type of user taking a recommendation (Group moderator for that group, group admin, service owner etc.). ➢ language: For language specific groups, to identify the language request. Default is English
Output	As explained above REC - 3REC - 1

Table 53 - API Definition – Hybrid Collaborative Recommendation

Type	Content Based Recommendation
Reference	REC - 4
HTTP Method	GET
API Call	Recommend_Content/user_id?rec_nums?patform_screen&language"
URL Params.	➢ user_id: Id that correlates to a SoCaTel platform user. ➢ rec_nums: Number of items returned from recommendation (Top 3, Top 5 etc.) ➢ platform_screen: Id that specifies from which screen is the recommendation requested from. This help define the type of "item" requested and the type of user taking a recommendation (Group moderator for that group, group admin, service owner etc.). ➢ language: For language specific groups, to identify the language request. Default is English
Output	As explained above in REC - 4

Table 54 - API Definition – Content Based Recommendation

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.

6 APPENDICES

6.1 APPENDIX 1 – TECHNICAL ARCHITECTURE OF THE SOCATEL CO-CREATION PLATFORM

Below is the "Technical architecture of the SoCaTel co-creation platform showing the interfaces, technologies and protocols used by the system component", as this was described in D3.2 – SoCaTel Platform Concrete Architectural Design

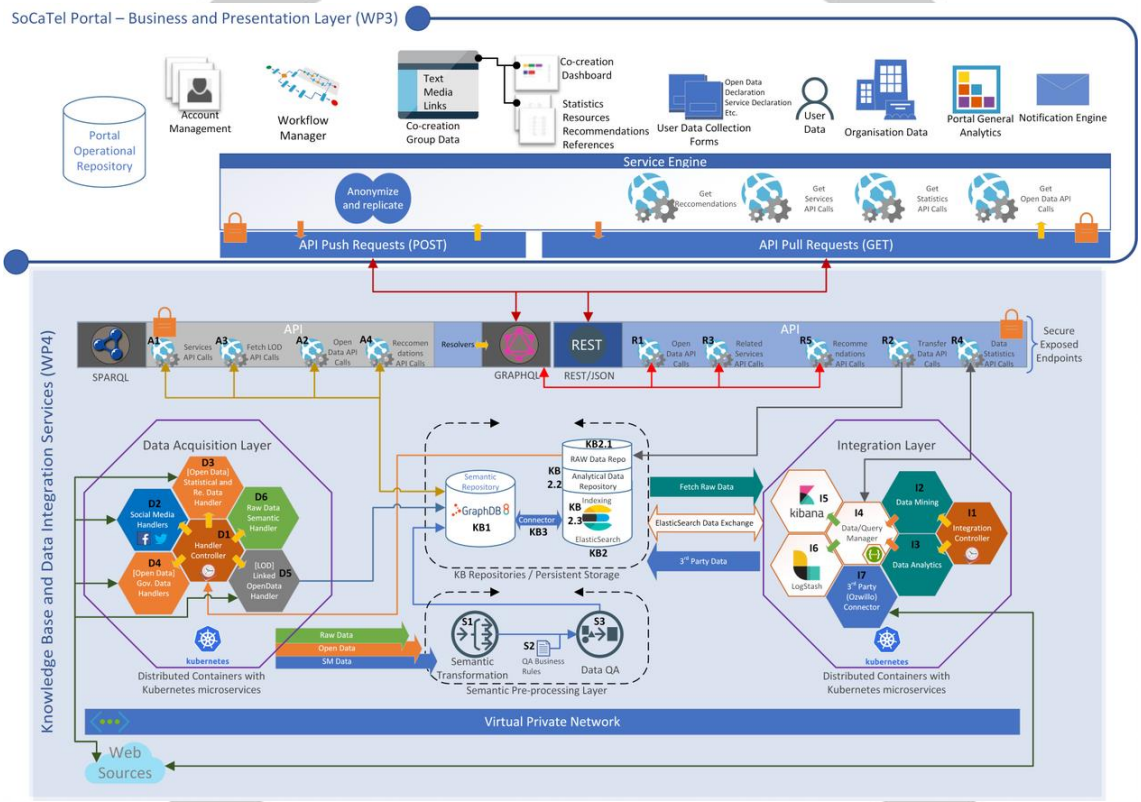

Figure 4 - Technical architecture of the SoCaTel co-creation platform showing the interfaces, technologies and protocols used by the system component

This project has received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Grant Agreement N° 769975.