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: D.4.3

Context Aware Recommendation Engine

By CyRIC

Due date of deliverable: 31/05/2019

Actual submission date: 31/05/2019

Responsible WP: WP4

WP responsible partner: CyRIC

Deliverable responsible partner: CyRIC

Revision: V 1.0

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

AUTHORS

Author	Institution	Contact (e-mail, phone)
Loizos Koukoumas	CyRIC	l.koukoumas@cyric.eu
Rafael Constantinou	CyRIC	r.constantinou@cyric.eu
Demetris Antoniades	CyRIC	d.antoniades@cyric.eu
Rizk Allah Touma	Everis	rizk.allah.touma@everis.com

DOCUMENT CONTROL

Document version	Date	Change
V 0.1	22/04/2019	Table of Contents
V 0.2	15/05/2019	First Draft
V 0.3	25/05/2019	Final Draft
V 1.0	28/05/2019	Completed Version for Review
V2.0	30/05/2019	Reviewed Version

VALIDATION

Reviewers		Validation date
Everis		
Ozwillo	Anthony Schoofs	29/05/2019

DOCUMENT DATA

Keywords	SoCaTel, Co-design, co-creation platform
Contact	Name: Pantelis Nicolaou Tel: +357 22 282828 E-mail: p.nicolaou@cyric.eu
Delivery date	31/05/2019

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.

Table of Contents

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.

Index of Tables

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

Index of Figures

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

1 Executive Summary

This document focuses on the SoCaTel Context-Aware Recommendation Engine. A service provided by the SoCaTel Knowledge Base and Data Integration Services layer, and which aims at assisting the user during the co-creation process. The service utilizes data (internal and external) collected as part of the Knowledge base processes and through their semantic translation into the SoCaTel defined ontologies, which facilitates context aware recommendations. The recommendations to be provided by the SoCaTel platform, during user interactions, aim at aiding the user identify co-creation groups of their interest as either service provider or service user. Additionally, after joining a co-creation group, the recommendation aims at providing to the user all relevant information around the scope of the group, thus aiding a more informed co-creation process. Such recommendations consider not only the preferences and past actions of the user, but also, their current status and activity in order to provide the most relevant information.

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

2 Introduction

SoCaTel is a Long-Term Care services co-creation platform, with purpose the creation of a space where individuals with different profiles and backgrounds can work together for the improvement of existing or creation of new long-term care services. The power of this space lies in the differentiation between its users. The different needs and ideas that people with different expertise, from different working backgrounds and age groups bring together to tackle challenges.

This raw power though, needs to be harnessed with dexterity in order to have the optimal results. Therefore, the inspirer of a service challenge can create a group discussion for people to join and exchange their opinion regarding the service from their point of view. The group character is defined from its title and the different keyword and tags given from the group creator during the group creation phase. The group creator can then invite individuals that have relative experience or relate to the service to join the discussion and give their ideas and feedback. Moreover, these keywords and tags can then work as an indication for those who were not initially invited, that the subject of the group is of their interest and join the conversation.

Furthermore, with all these individuals participating and even competing for the best idea as solution to the service, the need to ensure quality of input is necessary. Therefore, the platform provides a mechanism to help participants state their opinions based on facts and that gives evidence to the other discussion participants about the truth of a statement. Thus, SoCaTel co-creation platform supports a knowledge base, designed to complement the platform. Using this knowledge base, a user can source and provide external insights regarding a topic or a discussion and thus improve or diminish the gravity of a statement.

Given the nature of the project, there is a vast amount of information expected to be concentrated in the platform. However, this can prove to be an issue when trying to find group discussions or other individuals to invite for participation. Thus, the need arises for a medium that can “guide” the system users towards co-creation groups that might be of their interests or suggest to group creators users that might be productive towards the theme of the group or even suggest to individuals services with negative

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

feedback, in order to be improved or get recreated.

This document presents the principles behind the SoCaTel Context-Aware Recommendation engine; a system that perceives contextual information and categorises it as contextual entities and then combines it with the result of processing data through a computational method. The outcome of this engine recommends to a platform user, items (i.e. other users, groups services) that are closely related to the user profile and interests, as well as to the activities of the user within the platform. Such recommendations can be:

- User looking for Groups -> Recommend Groups.
- User created a Group -> Recommend other Users to invite to the Group.
- User is looking for Service -> Recommend other Services.
- User is looking for Resources -> Recommend Resources.

To fulfil these goals, the SoCaTel recommendation engine takes also into consideration the current context of the user, in regard to her interaction with the platform and relative to where recommendations will appear to the user. For example:

- User home screen -> Groups and Services recommendations.
- Group screen -> Services, organisations and resources correlated with the group topic.

Lastly, to ensure that recommendations are suitable for each situation, the recommendation will take into consideration the profile-related context of the user "asking" for" the recommendation. For example:

- A user speaks only the English language -> Do not recommend groups that the discussion is in the Greek language.
- Group discussion is addressed to the residents of a specific village -> Do not recommend this group to outsiders.

Consequently, this document is a study around the most popular recommendation engines and their implementation, what context offers to a recommendation engine and finally how the SoCaTel Recommendation Engine (SRE) is implemented.

A more detailed definition and explanation of how a recommendation engine works

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

follows in Section 2. The section also includes the definitions of the most widely used of the algorithms and techniques in the field of recommender systems, the different types of recommendation systems, their implementation steps and finally a comparison between these recommendation engines.

Since the subject of recommendation systems has been vastly used from the industry leaders (Amazon, Google, YouTube, Facebook) and even created sponsorships for improved systems (Netflix [1]), frameworks that implement most of the state-of-the-art recommender systems already exist. Section 3 provides a review of these systems, detailing what they offer and how compatible these are with the rest of the SoCaTel platform needs.

Finally, on Section 4 all this information collected from the previous sections gets into a pragmatic context, of how these types of recommendation systems can get into the context of the SoCaTel platform and create an engine; the SoCaTel Recommendation Engine.

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

3 Context Aware Recommendation Engine Definitions

Recommender systems are a relatively new concept, starting as an independent research topic in the mid-90s. Researches and practitioners tried to solve recommendation problems that explicitly relied on the notion of ratings as a way to collect user preferences, thus, the first recommender or recommendation systems were created. These software tools suggest information that might be of interest to the end user, taking into account data collected on that specific user, such as the user's preferences, actions, tasks and contextual information. Usually, these software systems are computational methods, estimating a rating for an item towards a user based on ratings given by this user to other items, ratings given to this item by other users and possibly some other more general information like demographics and item characteristics.

There is a great variety of areas and many examples of how recommender systems are used, whether this is about shopping preferences (Amazon, eBay), recommended videos to watch (YouTube), or recommend music and playlists to listen to (Spotify, YouTube). These systems can either operate using a single input like music or even combine multiple inputs across the inter-connected platforms like news, books and search queries. For example, watching a Star Wars movie might trigger recommendations for Star Wars music on a music site and Star Wars merchandise on a shopping site.

Consequently, a context aware recommendation engine can be defined as a recommendation engine that takes into consideration a context correlated directly with the result. For example:

Let's consider a scenario in which a recommendation engine would normally indicate that the most well-suited vacation site for a user is Cyprus, because of:

a) budget preferences,

b) weather preference,

c) proximity to home location,

d) destination popularity,

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

e) due to similar user preferences etc.

However, because the user additionally indicated that they are looking for a vacation during the months of January-February, the context aware recommendation engine may avoid a Cyprus recommendation due to a reduced score in the output rankings and point to a different destination such as the Canary Islands.

In the remaining of this section we first provide a detailed definition of context and how this is utilized in recommendation systems and applications. Following, we provide a comprehensive list of types and algorithms used to implement different types of recommendation engines, along with the definition of these types. Lastly, the section compares the recommendation engines in the form of tables and provides a conclusion from that comparison.

3.1 Context Definition

Context, as described from literature definitions [2] is "… any information that can be used, to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application including the user and applications themselves".

A good classification can help us understand a variety of contexts efficiently. Table 1 summarizes the most well-known forms of context classifications as mentioned in [3]. Each table row presents a different form of context, detailing the different categories that fall under that form. Note that forms are not mutually exclusive. This can help recognize the type of a given context before using it.

As one may note the multifaceted concept of context can create a bit of a "chaos" towards providing an explicit definition. The following subsection presents a "taxonomy of contexts", as introduced by Dourish [4] with the goal to bring "order" to this diversity.

Context Categorization	Observation Aspects and Context Features	Example
Who, When, Where, What, Why and How	Analysis of the environment from different points of view. Intuitive to understand the story behind a recommendation	**Life assistance of elderly** Peter, 8 o'clock, garden, picking fruit, apples are ripe, with a ladder.

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

Physical/Virtual	**Physical**: Differentiation according to sources: sensors, GPS locations etc. **Virtual**: Use of the analysed data – outcome of semantics, NLP datasets.	**Rehabilitation** **Physical**: heart rate **Virtual**: patient's medical history from database
Static/dynamic	**Static**: Observation over time Dynamic: Adaptive to changes Intuitive to learn and understand.	**Plant inspection** **Static**: the place where a tree grows **Dynamic**: the aspect of the tree due to the current season
Direct/Indirect	Differentiation through obtainment complexity: **Direct**: Knowledge acquired from the information gained from the data as they are **Indirect**: Knowledge gained from the context, acquired from the process of data (ontologies etc.)	**Birthday** **Direct**: actual date is the birthday **Indirect**: which birthday is it and does this mean something (e.g., 50th Birthday)
Sensed, Combined, Inferred and Learned	Differentiation of the way the data are collected and what they represent **Sensed**: Raw data metrics **Combined**: Combine raw data to gain extra information **Inferred**: Use data to create conclusions and rules **Learned**: Gain conclusions through feedback after each use iteration	**Navigation** **Sensed**: proximity to an object **Combined**: speed and direction of motion **Inferred**: check distance (rules) **Learned**: compare with similar situations

Table 1 – Context Categorization [3]

3.1.1 Taxonomy of contexts

Dourish [4] introduces "taxonomy of contexts" to bring "order" to the context definition diversity by classifying the contexts into representational and interactional views. Representational view contexts are defined with a predefined set of observable elements that do not change significantly over time. Hence these attributes, generally referred as "contextual attributes", can be easily identified, captured and used within context-aware applications. Contrary to this, interactional view context is often hidden since it collects context from the user behaviour.

According to Dourish [4] context aware application in practice, should follow an implementation that uses the human computer interaction and the interaction of the environment that surrounds the user, collects information and creates or identifies an analogous context. To achieve this, contextual information needs to be found and collected. The main ways for obtaining contextual information are categorized as [5] :

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

- **Explicitly:** By directly asking relevant people or using other sources of contextual information. For example, ask a person that joins a website to answer some specific questions that would provide context or tag some of her actions with predefined or free text tags.
- **Implicitly:** Gathering information from the data or the environment, such as timestamp of a transaction, location of a user detected from a post location.
- **Inferring:** Gather information from statistical or data mining methods. For example, a cable TV company identify who is the person watching TV (husband, wife, son, daughter, etc.) based on the TV programs watched the channels visited or identifying the sentiment (opinion) of the user about the topic through Natural Language Processing.

Following, Section 3.1.2 contains a description of how contextual attributes are chosen, referencing state-of-the-art methods for finding the contextual attributes of a subject. In the same section we also adapt the context definition and investigate approaches around the distinctive needs of recommender systems. Definition and more details regarding the Recommendation systems can be found in Section 3.3.

3.1.2 Contextual Attributes

Adomavicius et al. at [2] and [6] proposed a methodology for deciding the contextual attributes to be used in a recommendation system. Initially, a wide range of contextual attributes should be carefully chosen by domain experts. These attributes will aid and focus the data collection process. After and during the data collection, which also should include item ratings given by other users along with the contextual information, appropriate statistical tests should be applied, to identify which of the chosen (and collected) contextual attributes are truly significant in the sense that they indeed affect the item we are testing.

Yanlin & Yoneo [7] on the other hand introduced a framework (on 2007) with application in the e-learning context. This framework categorizes context elements in three fundamental components:

- Knowledge Context (Knowledge): Knowledge context refers to the objects of learning efforts and the relation amongst them and the complications they might enfold in the platform, like knowledge backgrounds, needs and interests. It exists in two forms: explicit and tacit [7].
- On the co-creation settings, information and communication technologies

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

(explicit knowledge) help co-creating users explicate their ideas. Tacit knowledge on the other hand can be defined as the co-creating experience that is often exchanged through joint activities (being together, spending time together, exchanging messages).

- Social Context (Human): The social context is directly linked to the human dimension of the co-creation platform and is defined from the human as the subject of the platform, human-based social network and human-based social culture. The social context deals with the social, cultural, psychological and emotional influences, such as social-cultural backgrounds, social distances, social roles/responsibilities and social prestige or status in the co-creation context.
- Technical Context (Technology): Technical context is defined from the technology that is used to overcome the obstacle of time or space, that might arise during the co-creation process. Technical context refers to factors that might change the influence of the co-creation platform such as technical media, participant's preferences, skills or time proximity.

$$R = User\ X\ Item\ X\ Context \rightarrow Rating$$

Equation 1 User, Item and context combined for calculating the rating

Considering the above definitions and approaches, a recommendation (R) in SoCaTel is the combination of User, Item and Context attributes that results in an item rating (see equation above). In the context of SoCaTel these attributes are defined as:

- Users: The people who use the platform for realising their ideas and services or provide comments and feedback or even offer their expertise and background for co-creating a service and implementing an idea.
- Services: Existing services, added in the platform from the service owners.
- Groups: Groups for discussing services and/or ideas. Users can open groups, join and invite other users
- Furthermore, the contextual information might consist of the following types:
- Location: Where the user lives, is the location relevant to the proposed topic
- Language: Language can be a barrier in groups co-creation

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

Additional features will be defined when platform operates. Moreover, as mentioned above, for a better approach on finding the context attributes, it will be necessary to get a feedback from field experts regarding each subject category.

3.2 Techniques and Algorithms Related to Recommendation Systems

This section acts as a reference point towards the definition of the methods, algorithms and techniques used by most of the recommendation systems that will be described in Section 3.3. The information provided here might prove necessary for further understanding the recommender systems.

A reference of what is used in each recommendation system is presented in Table **2**.

3.2.1 Similarity Metrics

3.2.1.1 Euclidean Distance

Euclidean distance is the most common method used for measuring the distance between two objects. Sometimes known also as the Pythagorean metric, this method finds the distance by calculating the root of square differences between coordinates of a pair of objects.

The Euclidean formula is:

$$d = \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2}$$

Equation 2 – Euclidean Distance

3.2.1.2 Cosine Similarity

Mathematically, it measures the cosine of the angle between two vectors projected in a multi-dimensional space. In our context, a multi-dimensional space where each dimension corresponds to a word in the document, the cosine similarity captures the angle of the documents. When measured, the smaller the angle between them the higher the similarity.

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

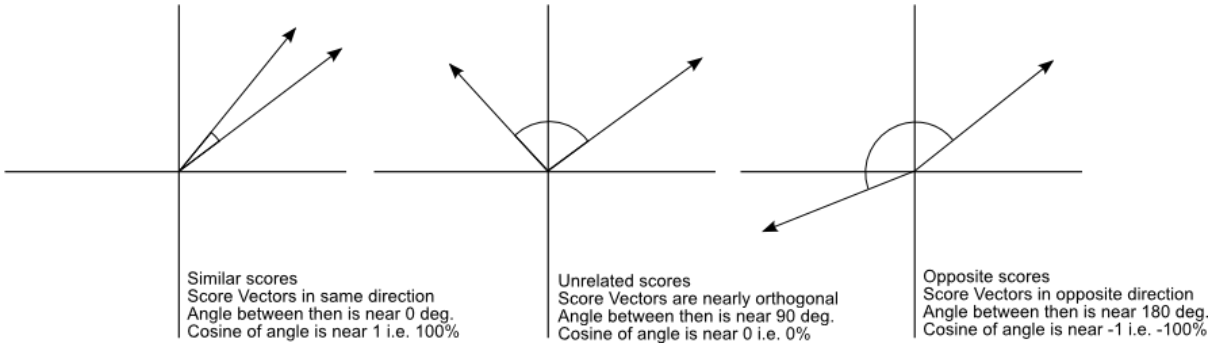

Figure 1 – Cosine Similarity Representation [8]

Figure 1 represents a diagram of how cosine similarity Is translated in the 3D space. In the figure, the words “hi”, “hello” and “world” represent the axis. Then the sentences “Hello, World!” and “Hi, world!” are placed in the diagram. To find the relation between the two sentences, we have to calculate the angle between them. The smaller the angle the more similar are the sentences.

3.2.1.3 Pearson Correlation

Pearson Correlation product-moment coefficient (its full name) calculates the strength of a linear association between two variables. In more detail, it draws a line of best fit between the data of two variables and the Pearson correlation coefficient “r” indicates how far away all these data points are to this line of best fit. As an example in **Figure 2**, we can see that the graphs with the greatest value in “r”, have the value of the variables closer. Furthermore, on the second row of graphs, we can see that there is a decline of values on the *y* axis and thus the “r” takes negative value.

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

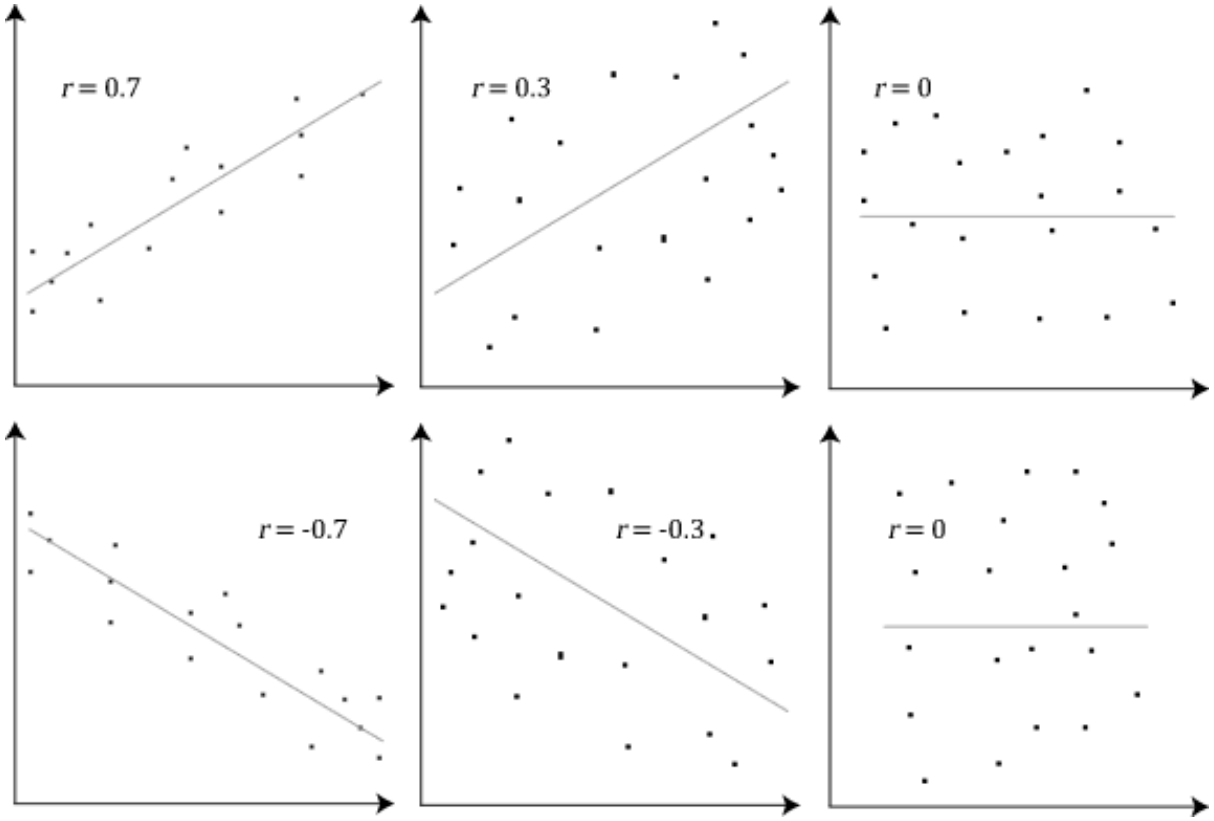

Figure 2 – Pearson Correlation Coefficient Measure [9]

The equation of Pearson Correlation can be defined as:

$$r = \frac{\sum(x - m_x)(y - m_y)}{\sqrt{\sum(x - mx)^2 \sum(y - my)^2}}$$

$$where\ m_x\ and\ m_y\ are\ the\ means\ of\ x\ and\ y\ variables$$

Equation 3 - Pearson Correlation Function

3.2.1.4 Jaccard Similarity

Jaccard Similarity Index (or Jaccard Similarity Coefficient) [10] is a similarity function that compares members of two datasets to find which members of the dataset are shared and which are distinct. The measure ranges from 0% to 100%, the higher the percentage, the more similar the two populations.

The equation of Jaccard Similarity can be defined as:

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

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

Equation 4 – Jaccard (Tanimoto) Distance

Let's take as an example two datasets:

- A = {0, 1, 2, 5, 6}
- B = {0, 2, 3, 4, 5, 7, 9}

J(A, B) = |A∩B| / |A∪B| = |{0,2,5}| / |{0,1,2,3,4,5,6,7,9}| = 3/9 = 0.33

3.2.1.5 Rocchio's Algorithm

Rocchio's Algorithm [11] applies the relevance feedback technique [12], which helps users improve queries based on the results of their previous searches.

> *Rocchio's algorithm represents documents as vectors, so that documents with similar content have similar vectors. Each component of such a vector corresponds to a term in the document, typically a word. The weight of each component is computed using the TF-IDF term weighting scheme. Learning is achieved by combining document vectors (of positive and negative examples) into a prototype vector for each class in the set of classes C. To classify a new document d, the similarity between the prototype vectors and the corresponding document vector representing d are calculated for each class (for example by using the cosine similarity measure), then d is assigned to the class whose document vector has the highest similarity value.*

3.2.2 Natural Language Processing Algorithms and Methods

3.2.2.1 TF-IDF

TF-IDF is one of the most popular and state-of-the art Natural Language Processing (NLP) algorithms. TF-IDF is categorised as one of the Content-Based Filtering (CBF) techniques. It stands for Term Frequency – Inverse Document Frequency (thus TF-IDF) and it is used to reflect the importance of a keyword to a document in a collection of corpuses.

> *TF is calculated by the frequency of a word in each document in the corpus. In other words, it is the ratio of word occurrences compared to the total number of words in the document. The equation of TF can be defined as:*

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

$$TFij = \frac{fi,j}{max\ fz,j}$$

Equation 5 - TF Calculation

Since a keyword might appear in many documents, therefore with TF, keywords are not very useful for finding the relevance between documents [13]. Thus, the Inverse Document Frequency (IDF) is also used, which captures terms that occur rarely in a document set, but are important, by diminishing the weight of terms that are popular but common in all documents.

$$IDF_i = log\left(\frac{N}{n_i}\right)$$

Equation 6 IDF equation

Combining these two we come up with the TF-IDF scored (w). In other words, the product of TF and IDF:

$$w_{i,j} = TF_{i,j} \times IDF_i$$

Equation 7 TF-IDF product

3.2.2.2 Latent Semantic Indexing

Latent Semantic Indexing is a retrieval technique that uses the structure of Latent Semantic Analysis [14] (LSA). LSA is a natural language processing (NLP) technique that specializes in correlating sets of documents and the terms they contain by producing a set of concepts related to the documents and terms. The specialisation of LSI rests on its ability to correctly match queries to documents of similar topical meaning when query and document use different words.

An example scenario that LSA and LSI would be useful, would be situations where synonym terms or expressions would be used for the same meaning. Terms like “laptop” and “notebook”, “monitor” and “screen”, with a TF-IDF (3.2.2.1) these words would be assigned as different. LSA can identify their similarity and therefore enhance the results that TF-IDF would return.

3.2.3 Classification

Classification is defined as the technique that decides, how much a data point (item) is or isn’t part of a class (specific category), or how much it does or doesn’t have an attribute [15]. For example, the spam detection in emails is a problem that can be

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

clearly solved with classification, since there are only two classes. Either spam or not spam.

Moreover, a classifier utilizes some training data to understand how input variables relate to the class. In this scenario, known spam and non-spam emails will be used for training. Onward, once the classifier is trained, it can be used to detect an unknown email.

Classification is considered as a supervised learning algorithm, where the machine is trained with known provided input data.

Classifiers are separated in two main categories [16]:

- Lazy Learners: Classifies testing data based on most related training data.
- Eager Learners: Construct classification model to cover the entire instance space (cover all scenarios).

There are many different classification algorithms available, each with its positives and negatives depending on the scenario of use. Two of the most popular state-of-the-art algorithms are Naïve Bayes and Decision Trees that are described below.

3.2.3.1 Naïve Bayes

The Naïve Bayes classifier is one of the most widely used machine learning algorithm because of its simplicity and ease of implementation. It belongs to the family of Bayesian Classifiers. Bayesian Classifiers construct their models based on experience which is used as training data.

> *Naïve Bayes classifiers assume strong, or naïve, independence between attributes of data points. They are popular for being used in spam filters, text analysis and even medical diagnosis. The Bayesian theorem is expressed as:*

$$P(c|d) = \frac{P(c)P(d|c)}{P(d)}$$

Equation 8 Naive Bayes Equation

3.2.3.2 Decision Trees

Decision tree is a decision helper tool that uses a tree-like graph or model of decisions and their possible results, including event outcomes, resource costs and utility. A

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

decision tree is a flowchart-like structure and each internal node represents a “test” on an attribute (true or false), each branch represents the outcome of a test and each leaf node represents a class label (decision taken after computing all attributes). For each decision, the tree branch tested expands, containing new child branches (therefore new tests) until a leaf is reached. Tree based algorithms are useful in machine learning and are considered as one of the best and most used in supervised learning methods. A simple example of a decision tree used to recommend the most suitable type of vehicle to a person is depicted in **Figure 3**.

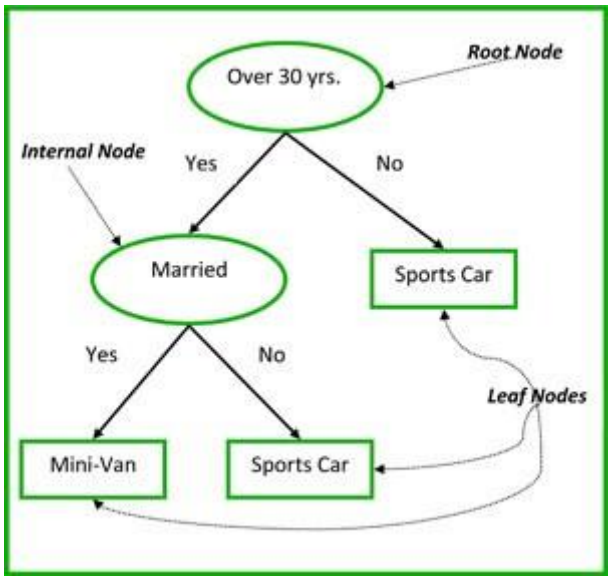

Figure 3 – Decision Tree for choosing a car [17]

3.2.4 Clustering

Clustering is the process of dividing the population of data points into a number of groups based on a similarity criterion. Thus, the data points in the same group are similar to other data points in the same group and dissimilar to other data points in other groups. Clustering is used to find meaningful structure, explanatory underlying process or generative features in a set of unlabelled data. There are different types of clustering [18], divided by the approach they follow like:

1. Density-Based Methods: Clusters are created only in heavily concentrated areas and are separated from each other by sparser areas.
2. Hierarchical Based Methods: Cluster form tree structures based on hierarchy. New clusters are formed using the previously formed cluster.

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

3. Partitioning Methods: Clusters are created by constructing k number of partitions (k<=number of objects) into k number of clusters. Then each object is tested and then assigned to the partition that is most similar [19].
4. Grid-based Methods: Data space is separated in a finite number of cells, forming a grid-like structure. These cells form the clusters.

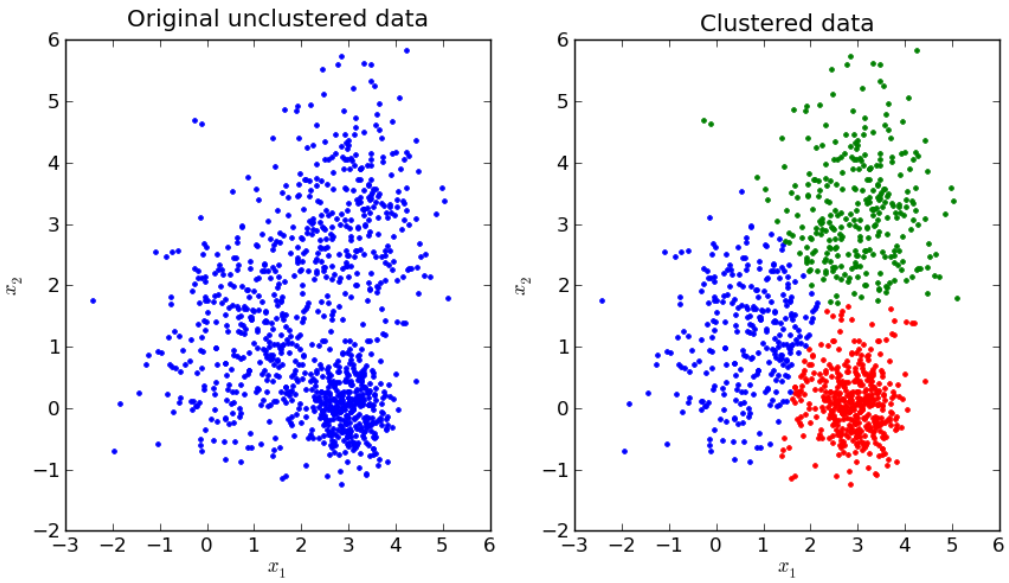

Figure 4 – Datapoints turn into clusters [20]

One important issue with clustering is the definition of similarity between objects, so that clusters can be formed. A common way to measure similarity is using a distance function like “Euclidean distance”, defined in Section 3.2.1.1. Distance functions return lower value for pairs of objects that are more similar to one another.

3.2.5 Multi-attribute utility theory (MAUT)

MAUT is a quantitative and systematic method that aims at modelling decision-making that involves multiple interdependent objectives based on uncertainty and preference analysis [21]. In other words, MAUT is a structured methodology that is designed to take into consideration multiple objectives and handle the trade-off in performance among them [22].

As a good example of MAUT implementation we can take its first application. MAUT methodology was involved in deciding alternative locations for building a new airport in Mexico City (in 1970). The multiple interdependent objectives were:

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

- The cost
- The capacity
- The access time to the airport
- Safety
- Social disruption
- Noise Pollution

Then, using a utility theory equation on the different candidate locations, an appropriate location was decided.

3.2.6 Implementation Approaches

There are two approaches when implementing a recommendation system. The fundamentals of these approaches are mostly based on whether the results are expected to be time efficient or they need to be as accurate as possible. Following are the definitions of two approaches. Heuristic (fast but not optimal) and Model-based approach (intelligent automated process to generate optimal solution).

Furthermore, **Table 2** presents the algorithms and methods necessary for implementing each recommendation system with each approach.

3.2.6.1 Heuristic Approach

Heuristic in general is defined as a problem-solving approach that utilises a practical process (“best practice”) that is fast and efficient in solving a specific problem and achieving immediate results. This is achieved by trading optimality, completeness accuracy or precision for speed [23].

3.2.6.2 Model Approach

Model approach [24] or Conceptual Model, embodies an intelligent, automated process to generate an optimal solution to a particular problem, taking decisions based on specific performance indicators. This approach aims at achieving the optimal solution in regard to performance towards all performance indicators.

There are two types of models. A model can be either:

- Descriptive: Helps in understanding underlying process or behaviour

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

- Predictive: Set of rules that predicts an unseen value based on other known values.

These models are trained from various machine learning algorithms that aim to adapt the model towards a better performing solution.

3.3 Types of Recommendation Systems

Since the challenges that a recommendation engine might have to face might be unique between them, there are also different types of how to construct a recommendation engine to tackle efficiently each challenge. Some of these methods associate a user with other users based on their similarity and recommend what those other users liked (Collaborative-based), while others are more focused on the content an individual user interacted with and recommend similar content (Content-based). This section describes some of the most popular methods and reports their pros and cons.

3.3.1 Collaborative-based Recommendation Systems

A Collaborative-based recommendation system focuses on analysing the behaviour information of a user, activities or preferences. It correlates them with other users with similar behaviour and based on what other topics those other users like or do, offers recommendations. This method is efficient in accurately recommending complex correlations between entities without having to understand the entire context of both entities, e.g. movies recommendations without the need of “recognizing” what the movie is about, since it does not rely on analysing the content but the user preferences. Nevertheless, collaborative filtering strongly follows the belief that people who agreed in the past will agree in the future, and they will like similar kinds of things as they liked in the past. Take as an example that a person A likes topics 1, 2, 3 and person B likes topics 2, 3, 4. Then this system will recommend topic 4 to person A and topic 1 to person B.

Furthermore, this filter can be specialised on the type of data it correlates as follow:

- **User-to-User Collaborative filtering:** Here, the effort is to search for lookalike users and recommend to them other groups or users based on what his/her

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

lookalike has chosen. This algorithm is very effective but takes a lot of time and resources, due to the need of calculating similarities between all pairs of users. So, for big base platforms, this algorithm is considered hard to put in place.

- **Item-to-Item Collaborative filtering**: It is very similar to the previous algorithm, but instead of finding a customer look alike, we try finding item look alike. Once the item look alike matrix is created, the algorithm can easily recommend alike topics to a user who has interacted with any topic in a group. This algorithm requires far fewer resources than user-user collaborative filtering. Hence, for a new user, the algorithm takes far less time than user-user collaborative as we do not need all similarity scores between users. Amazon uses this approach in its recommendation engine to show related products which boost sales.
- **Multimodal:** As one of the latest algorithms, "Multimodal" collaborative filtering is considered a revolutionary recommender[1]. It takes as data all sorts of information that might be considered as an indicator of a user taste.

3.3.2 Content-Based Recommendation Systems

Content-based recommendation systems focus on correlating the "content" that describes an item with the "content" of other items the user previously liked. These systems implement content-based filtering (CBF) recommendations. Furthermore, Pazzani in [25] and [26] claims that content based methods are based on both the item and a profile of the user's preferences. The following steps showcase how a simple content-based recommender system can be built:

1. The candidate items are described by keywords obtained from an algorithm like TF-IDF (3.2.2.1), which will return the words with the most weight in the item analysed. It is assumed that since these words hold most of the weight of the item description, they can describe the item to the core of its definition. These words are then usually stored in a vector, to be compared.
2. A user profile is built, that states what items the user likes, by analysing the activity history and the ratings (likes, upvotes) the user gave. These items are also analysed with the same algorithm (TF-IDF for example), thus their meaning will be also stored in another vector.
3. Using the above-mentioned vectors, a scoring heuristic can be used (like Cosine Similarity) to find the common distance between the two vectors which

[1] Recommender Overview: https://mahout.apache.org/docs/latest/algorithms/recommenders/

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

are basically the user preferences history compared against item description

Consequently, this filter is based on the idea that if the user likes a specific item, she will also like a 'similar' item.

3.3.3 Hybrid Recommendation Systems

Many modern recommender systems now are using a hybrid approach, combining Collaborative Filtering with Content-Based filtering, and sometimes even adding up other approaches [27]. The need for the combination of these filters was born from certain limitations the content-based and collaborative systems have [28] [29] [30]. According to Gediminas et al. [27] there are few ways to combine collaborative and content-based methods into a hybrid recommender system, and they can be summarized into:

1. Combining the predictions of collaborative and content-based filters after they are calculated separately. This can be done in two ways:
    a. By combining the outputs obtained from individual recommender systems into a final recommendation by using either:
        i. Linear combination of ratings [30].
        ii. Voting Scheme [31].
    b. Use individual recommender in any given moment based on a recommendation "quality" metric depending on what recommendation is requested.
2. Add content-based characteristics in a collaborative-based approach. As mentioned by implementors of this type of recommenders, in [28] and [31] both the traditional collaborative techniques and the content-based user profiles are maintained. Then using these content-based user profiles combined with the collaborative techniques the similarity between two users is calculated.
3. Add collaborative-based characteristics in a content-based approach. The usual approach to implementing this type of hybrid recommender is to use a dimensionality reduction technique on a group of content-based profiles. An example would be the use of latent semantic indexing (LSI) to create a collaborative view of a collection of user profiles, where these profiles are then

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

represented as term vectors. For example, the implementation of [32] presents an improvement in performance compared with a pure content-based approach.

4. Construct a general model that combines both the content-based and the collaborative-based characteristics and then calculate the predictions. There are a few methods for creating this model, using some of the state-of-the-art methods like:
    a. probabilistic latent semantic analysis [14],
    b. Bayesian mixed effects regression models [33] [34]
    c. Estimating ratings from known data [34]
    d. Augment hybrid recommendation systems with knowledge-based techniques [35]

Netflix is a good example of the use of hybrid recommender systems. The website makes recommendations by comparing the watching and searching habits of similar users (i.e., collaborative filtering) as well as by offering movies that share characteristics with films that a user has rated highly (i.e. content-based filtering).

Empirical comparison of the performance of hybrid recommender systems as reported by [27] [28] [31] [32] demonstrate improved accuracy compared to the pure collaborative and content-based methods.

3.3.4 Knowledge-Based Recommendation System

Knowledge-based Recommender systems, as also the Utility-based systems described below, do not attempt to build long-term generalisations about their users but built recommendations on an evaluation of the match between user needs and available options.

This recommender system attempts to suggest items to a user based on “inferences” about user’s needs and preferences, something that almost all recommendation systems do on some level. What differentiates the Knowledge-based system from the rest, is the fact that this system has a functional knowledge about how a particular item meets a particular user’s need and can therefore reason about the relationship between a need and a possible recommendation [35]. Nevertheless, the term is broad

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

and refers to many kinds of systems. The one common theme that unites all knowledge-based systems is an attempt to represent knowledge explicitly and a reasoning system that allows it to derive new knowledge. Thus, a knowledge-based system has two distinguishing features: a knowledge base and an inference engine.

The first part, the knowledge base, contains the facts about a situation. Often these are represented in some form of ontology. This is where the references of the recommender reside, thus creating a conclusion based on evidence and reasoning.

The second part, the inference engine, allows new knowledge to be inferred and consequently be added in the knowledge base. Most commonly, it can take the form of IF-THEN rules coupled with forward or backward chaining approaches.

As a relative example we can consider Google, which builds a user profile from any knowledge structure such as the query a user formulated while doing a search. Moreover, the knowledge used by a knowledge-based recommender can take many forms. For example, Google, using information collected from the connection of links between web pages, builds inferences regarding the popularity and the authorisation value of a page.

3.3.5 Utility-Based Recommendation Systems

Utility-based recommender systems make suggestions based on the calculation of the match between a user's need and the set of options available. More precisely, the Utility-based recommender computes the utility of each object compared to the user needs. Thus, the core of the success of this recommender lies on creating a proper utility function for each user. Various techniques exist on creating such an implementation.

Guttman [36] suggests a method called "Tête-à-Tête", an agent-mediated comparison system applied as a shopping system, that allows users to consider more dimensions than the price when buying an item by "asking" the customer two questions : "What to buy ?" and "Who to buy from ?".

Deng Feng [21] on the other hand, suggests an implicit utility and genetic algorithm implementation that self improves its recommender system on each iteration.

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

According to [21], the genetic algorithm approach offers a more accurate, customer satisfactory approach compared to its predecessors.

Due to the functionality and the algorithms used for Knowledge and Utility-based systems, some relevant work considers Utility-based systems to be a specific case of a Knowledge-based system [37] where elsewhere they are considered a recommendation system category of its own [38] [39]. This discussion is out of the scope of this document and furthermore this deliverable. Nevertheless, since all the references are important for the establishment of solid conclusions, sometimes these terms will be referred to independently whereas sometimes it will be referred as "Knowledge/Utility-based".

3.3.6 Ontology-Based Recommendation Systems

Ontology-based recommenders are knowledge-based recommendation systems that use ontology for knowledge representation. Similarly to their knowledge-based parents, ontology-based recommenders do no experience most of the problems associated with conventional recommendation systems such as cold-start problem or rating sparsity [40]. This is because ontology-based recommenders rely more on domain knowledge rather than ratings [41].

The use of ontology to represent data in a recommender's knowledge base has been proven to improve the overall quality of the recommendations [42] [43] [44]. This is because using ontologies brings many advantages to the recommender system, which can be summarized as follows:

1. Integrating multi-source and heterogeneous data: in order to obtain enough domain knowledge to perform accurate and personalized recommendations, many ontology-based systems rely on data retrieved from several different, heterogeneous data sources. Before being able to take advantage of this information to perform the recommendation, the recommender must then integrate the data coming from the different sources into one common format. In this sense, ontologies are commonly used in data integration tasks due to their high conceptualization, expressiveness and ability to reconcile the heterogeneities between the different sources [45] [46]. Some examples of

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

ontology-based recommendation systems that use ontologies for this purpose include [47] [48].

2. Built-in support for semantics and reasoning rules: ontologies use standard representation languages such as Web Ontology Language (OWL) and Resource Description Framework (RDF), to represent knowledge. These languages include predefined, built-in mechanisms to represent semantics and reasoning rules that allow the processing of the knowledge and the discovery of additional implicit knowledge. For instance, using an ontology we can differentiate the recommendations based on different topics (e.g. sports, music, food) to understand that two people that like the same music and sports may not share the same taste in food [5]. Similarly, concepts inside a specific topic can be organized in hierarchical schemes that allow the system to measure similarities and provide recommendations at different levels of granularity (e.g. dance music is a subtype of electronic music, skiing is a subtype of winter sports, etc.). Many recommendation systems that take advantage of these capabilities have been researched and implemented, such as [40] [49] [42] [50].
3. Straightforward accessibility to external knowledge; given that ontologies are a cornerstone of the Semantic Web and Linked Open Data projects, a plethora of structured freely available data is available in the form of ontologies. These external sources contain extensive knowledge in many domains that is readily available to be exploited by a recommendation system to improve the quality of the recommendations. For instance, Wordnet[2] is a lexical ontology that can be used to find synonyms, antonyms and definitions of concepts used by a recommendation system. These synonyms allow us to discover new relations with other concepts that might not be explicitly defined in the system's ontology and use these new relations when offering recommendations. Another example is using DBPedia[3] to discover additional information about a specific location (e.g. population, official language) and use that information to recommend locations similar to the ones in which the user is interested. Some examples of

[2] https://wordnet.princeton.edu/
[3] https://wiki.dbpedia.org/

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

recommendation systems that use this type of external information are [47] [49].

3.3.7 Recommendation Systems Comparison

Due to the numerous possible approaches regarding a recommendation system, it is considered necessary to create a comparison section, in order to clearly represent the capabilities of each type of recommendation and how similar it is with other recommender systems. The tables in this section attempt to prepare the definitions for the implementation phase, describing a more practical representation of the algorithms. Additionally, this section describes how methods and algorithms described in Section 3.2 relate to the recommender systems.

This is described throughout **Table 2**. The table is separate in rows, referring to the types of recommendation systems. Additionally, the table columns compare the two ways that the recommender systems can be implemented. Thus, consequently its cells are filled with information regarding the methods and algorithms that are used in each approach.

Recommendation Approach	**Recommendation Technique**	
	Heuristic-based	Model-based
Content-based	Commonly used techniques: • TF-IDF (information retrieval) • Clustering Representative research examples: • Lang 1995 • Balabovic & Shoham 1997 • Pazzani & Billsus 1997	Commonly used techniques: • Bayesian classifiers • Clustering • Decision trees • Artificial neural networks Representative research examples: • Pazzani & Billsus 1997 • Mooney et al. 1998 • Mooney & Roy 1999 • Billsus & Pazzani 1999, 2000 • Zhang et al. 2002
Collaborative	Commonly used techniques: • Nearest neighbour (cosine, correlation) • Clustering • Graph theory Representative research examples: • Resnick et al. 1994 • Hill et al. 1995 • Shardanand & Maes 1995 • Breese et al. 1998 • Nakamura & Abe 1998	Commonly used techniques: • Bayesian networks • Clustering • Artificial neural networks • Linear regression • Probabilistic models Representative research examples: • Billsus & Pazzani 1999, 2000 • Breese et al. 1998 • Ungar & Foster 1998 • Chien & George 1999

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

	• Aggarwal et al. 1999 • Delgado & Ishii 1999 • Pennock & Horwitz 1999 • Sarwar et al. 2001	• Getoor & Sahami 1999 • Pennock & Horwitz 1999 • Goldberg et al. 2001 • Kumar et al. 2001 • Pavlov & Pennock 2002 • Shani et al. 2002 • Yu et al. 2002, 2004 • Hofmann 2003, 2004 • Marlin 2003 • Si & Jin 2003
Hybrid	Combining content-based and collaborative components using: • Linear combination of predicted ratings • Various voting schemes • Incorporating one component as a part of the heuristic for the other Representative research examples: • Balabonovic & Shoham 1997 • Claypool et al. 1999 • Good et al. 1999 • Pazzani 1999 • Billsus & Pazzani 2000 • Tran & Cohen 2000 • Melville et al. 2002	Combining content-based and collaborative components by: • Incorporating one component as a part of the model for the other • Building one unifying model Representative research examples: • Basu et al. 1999 • Condliff et al. 1999 • Soboroff & Nicholas 1999 • Ansari et al. 2000 • Popescul et al. 2001 • Schein et al. 2002
Knowledge/Utility-Based	Commonly Used Techniques: • TF-IDF • Cosine Similarity	Commonly Used Techniques: • Genetic Algorithm • MAUT

Table 2 – Algorithms used on recommendation systems

Table 3 presents a comparison of the fundamental functionalities of the different recommendation systems. The columns of the table refer to:

- What background the recommender requires in order to function -> Background
- What input does the recommender requires in order to function -> Input

What is the process and output of each of the recommender systems -> Process

Technique	Background	Input	Process
Collaborative	Ratings from set of users of items in Items	Ratings from User of items in Set of Items	Identify users in sets of Users similar to user and calculate ranking of items based on their ratings on items

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

Content-based	Features of items in set of Items	User's ratings of items in Set of Items	Generate a classifier that fits user's rating behaviour and compare it with an Item
Utility-based	Features of items in Set of Items	A description of user preferences on a Set of Items	Apply a utility function to the items and determine item's rank.
Knowledge-based	Features of items in Set of Items. Knowledge of how the items respond to any of the user needs	A vector of user needs or interests	Infer a match between item and user's need

Table 3 – Recommendation Techniques (excluding Hybrid[4])

Consequently, the final table attempts to compare what are the positives and negatives of each of the recommender systems. Therefore **Table 4** lists the fields in which each recommendation technique excels and where it struggles to give optimal results.

Technique	Positives	Negatives
Collaborative filtering (CF)	1. Can identify items from different categories 2. No need of background knowledge regarding items or users 3. Adaptive: Improves over time 4. Suggestive feedback (Learning)	1. New user data "Cold Start[5]" 2. New item data "Cold Start" 3. "Grey sheep" problem 4. Result quality depends on large "history" dataset
Content-based (CN)	1. No need of background Knowledge regarding items or users 2. Adaptive: Improves over time 3. Suggestive feedback (Learning)	1. New user data "Cold Start" 2. Result quality depends on large "history" dataset 3. Stability vs Plasticity issues
Hybrid (CF + CN)	1. No need of background Knowledge regarding items or users 2. Adaptive: Improves over time	1. New user data "Cold Start" 2. Result quality depends on large "history" dataset

[4] Hybrid in the context is identical with the Knowledge-based recommendation system.
[5] Cold Start: Lack of personalized recommendations for users due to the lack of data.

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

	3. Suggestive feedback (Learning)	
Utility-based	1. No "Cold start "issues 2. Responsive on changes of preferences	1. User needs to input utility function (Hard for the user) 2. No suggestive feedback (Doesn't learn)
Knowledge-Based	1. No "Cold start" issues 2. Responsive on changes of preferences	1. No suggestive feedback (Doesn't learn) 2. Requires knowledge engineering

Table 4 – Recommendation Techniques pros and cons

As can be indicated from **Table 4**, each recommendation system has strong points, that can produce recommendations with a highly accuracy potentials but all of them in certain scenarios struggle to unleash their full potential. Therefore, the SoCaTel Recommendation Engine is planned to provide means to implement all recommender system solutions, thus, each solution will be implemented on a situation where it will excel that most.

This might not have been possible, if it weren't for the recommendation system available frameworks and technologies. As will be seen in section 4, most of the available frameworks and technologies have the capabilities to implement more than one (1) of these recommender systems.

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

4 Existing Technologies and Frameworks – Review

4.1 Introduction

The various tools and frameworks described in the sections below (4.2 through to 4.6) were evaluated for the particular needs of the SoCaTel recommendation, as these have been described in Deliverable 3.1 and Deliverable 3.2.

4.2 Mahout Framework

Apache Mahout[6] is an open source tool that provides a distributed algebra framework which is widely used by the research community for performing mathematically expressive and exhaustive Scala DSL (Domain-specific language). This is designed to let mathematicians, statisticians and data scientists quickly run their experiments and implement their own algorithms.

Using Mahout, any user can seamlessly run any experiments and Machine Learning algorithms without being aware of the underlying infrastructure. This enables them to create infrastructure-agnostic algorithms that could easily scale in different infrastructure set-ups and different supporting architectures (CPU/GPU/CUDA cores). As more back-end power is added this ensures the "write once, run everywhere"[7] moto of Mahout.

Mahout runs coupled with a Hadoop[8] Infrastructure as its background heavy-processing data management system that can handle and process huge volumes of data which are often in an unstructured form.

Mahout, alongside Hadoop, provides an environment to create and utilize existing scalable machine learning algorithms such as Classification Algorithms, Clustering Algorithms and Recommender Systems.

Recommendation filters, which are used for calculating the similarity between entities,

[6] Apache Mahout: https://mahout.apache.org/
[7] Apache Scala & Spark Bindings: http://mahout.apache.org/users/sparkbindings/home.html
[8] Apache Hadoop: https://hadoop.apache.org/

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

which are provided by Mahout are:

1. Euclidean (see 3.2.1.1)
2. Pearson Correlation **(**see 3.2.1.3)
3. Jaccard Similarity (see 3.2.1.4)

Some of the classification, clustering algorithmic approaches offered by Mahout are:

1. Distributed Hadoop MapReduce Jobs for Mining Tasks on large volumes of data
2. K-Means
3. Fuzzy K-Means
4. Canopy
5. Dirichlet
6. Mean-Shift
7. Distributed Naïve Bayes Classification
8. Complementary Naïve Bayes Classification
9. Fitness Function for evolutionary programming
10. Vector and Matrix libraries for Vector Multiplication, Matrix Content-Based Similarity Matrix, etc.

Using the above as a readily available functionality, what we aim to do next is to feed these with the correct input and then Mahout takes over. It will then use its infrastructure, regardless of this being a single, parallel, or a cluster of nodes, to calculate and produce recommendation according to a need. This is achieved in a relatively easy way due to its vast recognisability amongst the community that generated a large corpus of online documentation and examples[9].

4.3 PredictionIO

Apache PredictionIO[10] is an open source Machine Learning Server built on top of a state-of-the-art open source stack for developers and data scientists to create predictive engines for any machine learning task. The library is available with a variety of SDKs (Software Development Kits) in many programming languages such as Java, PHP, Python, Ruby and many other community powered SDKs.

[9] Mahout Recommender Overview: https://mahout.apache.org/docs/latest/algorithms/recommenders/
[10] Apache PredictionIO: 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.

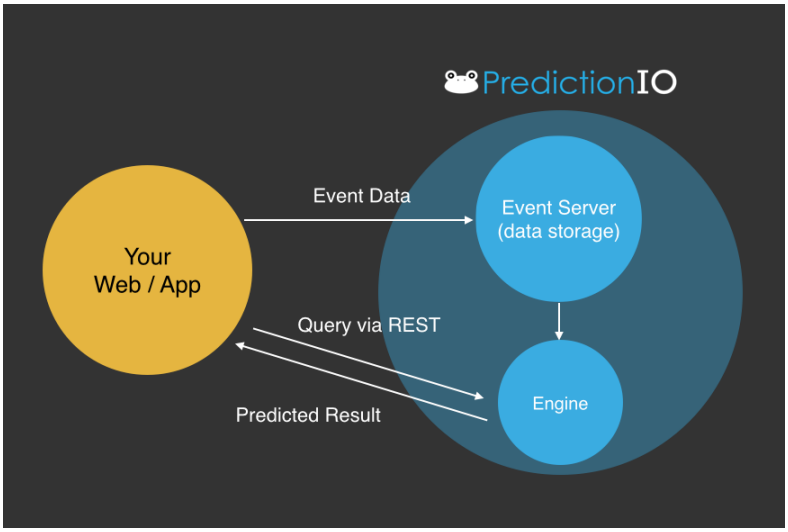

Figure 5 – PredictionIO - Quick Intro[11]

According to the PredictionIO – Quick Intro[11] and **Figure 5** above, the PredictionIO Platform mainly consists of the following components which are briefly explained below:

1. PredictionIO platform – open source machine learning stack for building, evaluating and deploying Engines with machine learning algorithms.
2. Template Gallery – engine templates for different type of machine learning applications. The Template Gallery, despite not being shown in the referenced Figure, is a repository and provides a wide range of algorithms that are then deployed as standalone Engines. The template repository is explained below.
3. Event Server – open source machine learning analytics layer for unifying events from multiple platforms
    a. The Event server continuously acts as a simulation stream mechanism for data collection from multiple data sources that originate from multiple web applications, mobile agents, etc. The collect of data from multiple sources, as the ones defined before, takes place with the use of EventAPI[12]. The data that are streamed into the PredictionIO platform are data that are Event Data shown in the figure.
    b. The purpose of the Event Server is using this data to build predictive model(s) with one or many algorithms. Thus, the Event Server knows the underlying deployed Engines and which types of data these engines subscribed to. As soon as specific Event Data of a particular type is being streamed into the Event Server, this data is then being sent to the many Engines for training, testing and validations.

[11] Apache PredictionIO – Quick Intro: https://predictionio.apache.org/start/
[12] Apache PredictionIO – EventAPI: https://predictionio.apache.org/datacollection/eventapi/

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

c. After the training, explained above the, these Engine(s) listen to queries from the multiple web applications, mobile agents, etc. and respond with predicted results in real time.

An Engine is generally responsible for making predictions as it contains one or more ready to use machine learning algorithms. For providing this the PredictionIO offers a wide range of templates[21] which are the following:

1. Recommenders
    a. User-based Personalized Recommendations
    b. Item-to-Item Similar Item Recommendations
    c. Viewed this bought that item-based cross-actions recommendations
    d. Popular Items and User-Defined Rankings
    e. Item-Set recommendations for complementary purchases or shopping cars
    f. Hybrid Collaborative Filtering and content-based recommendations
2. Classifications
    a. LingPipe[13]
    b. Lead Scoring
    c. Text Classification (Spark MLLib's Multinomial Naive Bayes)
    d. Churn Prediction - H2O Sparkling Water – Deep Learning Algorithm
    e. Classification Deeplearning4j[14]
    f. Probabilistic Classifier (Logistic Regression w/ LBFGS)[15]
    g. Document Classification with OpenNLP[16]
    h. Circuit End Use Classification
    i. GBRT_Classification[17]
    j. MLlib-Decision-Trees-Template[18]
    k. Classification with MultiLayerNetwork[19]
    l. Deeplearning4j RNTN[20]
    m. classifier-kafka-streaming-template
    n. Sentiment Analysis - Bag of Words Model
    o. Classification template for Iris

Using PredictionIO, we can quickly build and deploy several machine learning components as web-services on a production scale environment with multiple

[13] LingPipe Classification Algorithm: http://alias-i.com/lingpipe/
[14] PredictionIO-template-classification-dl4j: https://bit.ly/2VYKgDt
[15] Classification Engine Template: https://bit.ly/2K6blNT
[16] OpenNLP Text Category Classifier Engine Template: https://bit.ly/2WmmTTA
[17] PredictionIO-GBRT-Classification: https://bit.ly/2MgvYJY
[18] PredictionIO-MLlib-Decision-Trees-Template: https://bit.ly/2WtYgEI
[19] PredictionIO Classification Engine Template with MultiLayerNetwork from Deeplearning4j: https://bit.ly/2K1l4ov
[20] PredictionIO - Deeplearning4j RNTN: https://bit.ly/2M651bz

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

customisable templates[21] in an Engine form as already explained above. This is one of the most important advantages that the PredictionIO offers. The above adds the necessary expendability that our recommender system needs to have in terms of being easily deployable, highly available and maintainable and having a degree of automation.

4.4 LensKit Toolkit

LensKit[22,23] [51] is a set of Python tools for experimenting with and studying recommender systems. It provides support for training, running, and evaluating recommender algorithms in a flexible fashion suitable for research and education. LensKit is an Anaconda[24] library distribution.

In addition, LensKit provides Batch-Running Recommenders[25] for multiple users or user-item pairs. There is also the option of Scripting Evaluation that evaluates multiple algorithms or algorithm variants simultaneously, across multiple datasets.

The algorithms that LensKit provides for filtering and recommendations are the following[26]:

1. Basic and Utility Algorithms
    a. Personalized Mean Rating Prediction: A user-item bias rating prediction algorithm
    b. Fallback Predictor: Simple hybrid that takes a list of composite algorithms, and uses the first one to return a result to predict the rating for each item
    c. Memorized Predictor: Primarily useful for test cases. It memorizes a set of rating predictions and returns them
2. k-NN Collaborative Filtering
    a. Item-based k-NN: Item-item nearest-neighbour collaborative filtering with ratings
    b. User-based k-NN: User-user nearest-neighbour collaborative filtering with ratings

[21] Engine Template Gallery: http://predictionio.apache.org/gallery/template-gallery/
[22] LensKit: https://lenskit.org/
[23] LensKit Documentation: https://lkpy.lenskit.org/en/stable/
[24] Anaconda: https://www.anaconda.com/
[25] Batch-Running Recommenders: https://lkpy.lenskit.org/en/stable/batch.html
[26] LensKit Algorithms: https://lkpy.lenskit.org/en/stable/basic.html

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

3. Classic Matrix Factorization
4. Hierarchical Poisson Factorization.

4.5 Surprise

Surprise[27] is a Python Scikits (SciPy Toolkits, add-on packages for SciPy[28], which is an open-source software for mathematics, science and engineering), for building and analysing recommender systems. Surprise has a documentation on how to train and test the recommender systems' algorithms and can be found here[29].

Surprise offers the following list of ready-to-use prediction algorithms with some of the most common examples of each category and many similarity functions, such as:

1. Prediction Algorithms
    a. NormalPredicator: Algorithm predicting a random rating based on the distribution of the training set, which is assumed to be normal.
    b. BaselineOnly: Algorithm predicting the baseline estimate for given user and item.
    c. SlopeOne: A simple yet accurate collaborative filtering algorithm.
    d. CoClustering: A collaborative filtering algorithm based on co-clustering.
2. Neighbourhood methods
    a. KNNBasic: A basic collaborative filtering algorithm.
    b. KNNWithMeans: A basic collaborative filtering algorithm, taking into account the mean ratings of each user.
    c. KNNWithZScore: A basic collaborative filtering algorithm, taking into account the z-score normalization of each user.
    d. KNNBaseline: A basic collaborative filtering algorithm considering a baseline rating.
3. Matrix Factorization-based methods
    a. SVD: The famous SVD algorithm, as popularized by Simon Funk during the Netflix Prize.

[27] Surprise: http://surpriselib.com/
[28] SciPy.org: https://www.scipy.org/
[29] Surprise' documentation: https://surprise.readthedocs.io/en/stable/index.html

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

b. SVD++: The SVD++ algorithm, an extension of SVD considering implicit ratings.
c. NMF: A collaborative filtering algorithm based on Non-negative Matrix Factorization.

4. Similarity Functions:
   a. Cosine
   b. Mean Squared Difference
   c. Pearson
   d. Pearson Baseline

Surprise does not offer any infrastructure components. It works as a library component that can be inserted in a Python script. It is a developer's responsibility to load the data and prepare the input that the Surprise consumes

4.6 Racoon Recommendation Engine

Racoon Recommendation Engine[30] is a collaborative filtering-based recommendation engine. It mainly consists of a NodeJS and a Redis server. The engine uses the Jaccard coefficient to identify the similarity between users and then the k-NN neighbours' algorithm to create recommendations. This is a rather simplistic library that can be deployed as a service to any infrastructure set up.

4.7 Technologies and Frameworks Comparison

The table below presents an overview of all the tools that were mentioned in the previous sections. These tools are examined in accordance to the architecture purposed on ***D3.2 SoCaTel Platform Concrete Architecture Design***, thus the main concerns are the ones described in the table.

It is of a great importance to choose and evaluate tools on which level they are Containerized-ready. If not, we need to evaluate at this point what is the effort of attempting an in-house transforming these tools to containers. In addition, an

[30] Racoon Recommendation Engine: https://github.com/guymorita/recommendationRaccoon

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

important role plays the Data Storage that each tool requires. The Data Storage is already defined in D3.2 and this deviating from the initial plan is not considered a viable solution.

The table below presents our research on the aspects explained above. The last row projects the Adaptability based on Partners' Knowledge, which defines the level of difficulty that corresponds in adapting a technology/tool. The latter is measured in a scale of 1-5 with 1 being the Less Confident of adapting and 5 being the Highest Confident for adaptation

The Combined that relates to the Hybrid Recommendation Techniques, describes that in order to create a Hybrid Recommendation a Collaborative and/or Content based output is required which is often derived from the same tool.

The Utility-based/Knowledge-Based recommendations are not fulfilled from the candidate tools and frameworks that were examined above. At this stage we are evaluating the option of creating these Recommendations techniques from scratch in an in-house environment that will allow us to test their performance and efficacy.

Tool/Framework		**Mahout**	**Prediction IO**	**LensKit Toolkit**	**Surprise Scikit**	**Racoon**
Open source		X	X	X	X	X
Documentation		X	X	X	X	X
Containerised		X	X		X	X
Recommendation Technique	*Collaborative filtering (CF)*	X	X	X	X	X
	Content-based (CN)	X	X	X	X	-
	Hybrid (CF + CN)	Combined	Combined	Combined	Combined	-
	Utility-based/ Knowledge-Based	*	*	*	*	*

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

Programming Languages	Java, Scala, Spark	**REST API**, Python, PHP, Ruby, Java	Python	Python	NodeJs Redis
Data Storage	Hadoop	HBase, Elastic search	-	-	
Adaptability based on Partners' Knowledge [1 Less Confident-5 Highest Confident]	1	5	3	4	3

Table 5 – Recommender Tools/Framework Comparison

4.8 Conclusion

The main idea behind the SoCaTel Recommender System, *hereafter known as SRE,* is to create a recommendation engine based on existing reusable tools and functionality. The purpose of this is fulfilled using a combination of the tools described above.

The most important aspects that help us choose which of the tools and/or frameworks will be used are the following:

1. *Compatible Data Storage* with the SoCaTel infrastructure, defined in D3.2 SoCaTel Platform Concrete Architecture Design.
2. *Adaptability,* that will provide an abstract way of deploying several algorithmic components for performing filtering all types of recommendations outlined in the previous sections. The adaptability also ensures that any implementation will not be bound to specific programming languages and thus programmers and developers will freely use any programming language if the implementation is deployable in a containerized environment as this is defined in D3.2 SoCaTel Platform Concrete Architecture Design.

For the reasons mentioned above we have concluded the following by considering each of the tools' advantages and disadvantages:

- PredictionIO is a strong candidate of a framework to use, as it encapsulate all algorithmic approaches presented in Chapter 3 - Context Aware

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

Recommendation Engine Definitions, including the technics and algorithms and the types of recommendation systems.

- Surprise Scikit is also a suitable tool since it is a simple python library which, given the data in a correct pre-defined tool format, can provide extensible algorithms that we can use for the types of recommendation that our system will provide. The format is nothing more than the data input that each algorithm defines which can either be an array of objects, vectors, matrixes, etc.

In conclusion:

- ✓ PredictionIO is selected as the main recommendation framework for generating all recommendations which are mentioned later in the document (Section - 5.3.2).
- ✓ The motivation behind PredictionIO is the extendibility, adaptability and reusability of the framework for creating and providing an engine in a containerised environment that will have all the encapsulated algorithms in a service-oriented deployment that the SoCaTel portal will use to absorb recommendations. The above, as previously mentioned, is fully compatible with the SoCaTel infrastructure, defined in D3.2 SoCaTel Platform Concrete Architecture Design.
- ✓ Any recommendation technique, similarity function, etc., which is not provided by PredictionIO will be customizable in a template form and will also be deployed as an Engine to PredictionIO such as the Surprise Scikit mentioned above.

For all the above we will perform a thorough test and hands-on experimentation to make sure that all functional requirements defined in D1.1 and D3.1 along with all technical and infrastructure requirements defined in D3.2 are fulfilled.

The applicability of the SRE along with what recommendation cases are covered by the SRE is presented in Chapter 5 - Implementation.

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

5 Implementation

5.1 Introduction

Section 5.2, will provide an overview of the existing SoCaTel Knowledge Base and how it relates to the semantic ontology from which the SRE will obtain data. In addition, Section 0 describes the component architecture of the SRE by providing an in-depth analysis on which underlying recommendations will be available. The SRE recommendations are explained with respective purpose, input, methodology, recommendation filter and an output example for each methodology. The chapter concludes with Section 5.4 that outlines the SRE Engine Technical Documentation and describes the distinct components of it.

5.2 SoCaTel Knowledge Base

The recommendation system of SoCaTel uses data retrieved from many types of external and internal sources to perform high-quality, personalized recommendations. The external sources can be split into the following main categories (note that the external data sources used in SoCaTel, along with the data handlers responsible for each of them within the Data Acquisition Layer, are fully documented in the deliverable D4.1):

1. Social Media sources: particularly social media accounts (e.g. Facebook pages and groups, Twitter feeds, etc.) of Long-Term Care (LTC) providers,
2. Open Data sources: consist of datasets published following the CKAN open data standard,
3. Linked Open Data sources: consist of sources available in the Linked Open Data project that contain semantic data in the form of RDF / OWL.

Additionally, the SoCaTel recommendation system uses anonymized SoCaTel platform data, which describe the interactions of the user with the platform and are used to identify user preferences. Moreover, user provided profile information (such as location, language, expertise etc.) are used to further enhance the recommendations.

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

Thus, in order to facilitate the recommendation process and integrate the data retrieved from these sources, the SoCaTel recommender system will use a custom-designed ontology. The ontology was designed not only considering the integration requirements, but also the advantages that come with using an ontology-based recommendation system (see Section 2.2.2.6).

Once the data is mapped to this ontology and stored into the knowledge base, it can be exploited by the recommendation system through the SPARQL and GraphQL APIs, which form part of the Integration Layer as detailed in deliverable D3.2. By doing so, the recommender has access to the data integrated from the different external sources through a single API, without needing the details of the data's original source or format.

5.2.1 SoCaTel External Sources Ontology

In Figure 6 we show the design of the SoCaTel external sources ontology. Note that this ontology contains many concepts and relations defined by other ontologies (such as SIOC, GN and SKOS) which can also be used to map the output of the data handlers. The main concepts and relations defined directly in this ontology are the following:

- **socatel:ExternalSource:** the core concept of the ontology. It is an abstract concept that represents all external sources used in SoCaTel. It has many properties representing various pieces of information common to all the sources (e.g. title, description, language, location, web link, etc). It also has several subclasses which correspond to different types of sources, namely *socatel:LODSource, socatel:OpenDataSource* and *socatel:SocialMediaSource*.
- **socatel:topic:** a relation that defines one or more topics related to an External Source. An external ontology specialized in classification schemes and taxonomies, SKOS[31], was used to represent the topics and, if needed, organize them into taxonomies with relations between the different topics (e.g. sub-topic, parent topic, equivalent topic, etc.).
- **socatel:LODSource:** a subclass of *socatel:ExternalSource* used to represent

[31] https://www.w3.org/2004/02/skos/

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

Linked Open Data sources.

- **socatel:OpenDataSource:** a subclass of *socatel:ExternalSource* used to represent Open Data sources. Using *owl:equivalentClass*, it is declared as an equivalent class of *dcat:Dataset* from the DCAT ontology, which can be used to represent additional information about open data sources.
- **socatel:SocialMediaSource:** a subclass of *socatel:ExternalSource* used to represent Social Media sources. It has in turn several subclasses that represent the different types of social media sources used (i.e. pages, account, post). Each subclass is declared as equivalent to different classes from the SIOC ontology[32], which can be used to represent additional information about the sources.
- **socatel:Page:** a subclass of *socatel:SocialMediaSource* used to represent Facebook pages.
- **socatel:Account:** a subclass of *socatel:SocialMediaSource* used to represent Twitter accounts.
- **socatel:Post:** a subclass of *socatel:SocialMediaSource* used to represent a post on a Facebook page or group or a tweet retrieved from a Twitter account.
- **socatel:fact*:*** a high-level property used to represent additional information retrieved from Linked Open Data sources. Each property leads to a *socatel:Fact* concept, which in turn has a *rdfs: label* that describes the fact and an *rdf:value* that has the value of the fact. This property was introduced to cope with the variety of data that can be retrieved from LOD sources while avoiding creating different concepts for each piece of retrieved information.

[32] http://rdfs.org/sioc/spec/

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

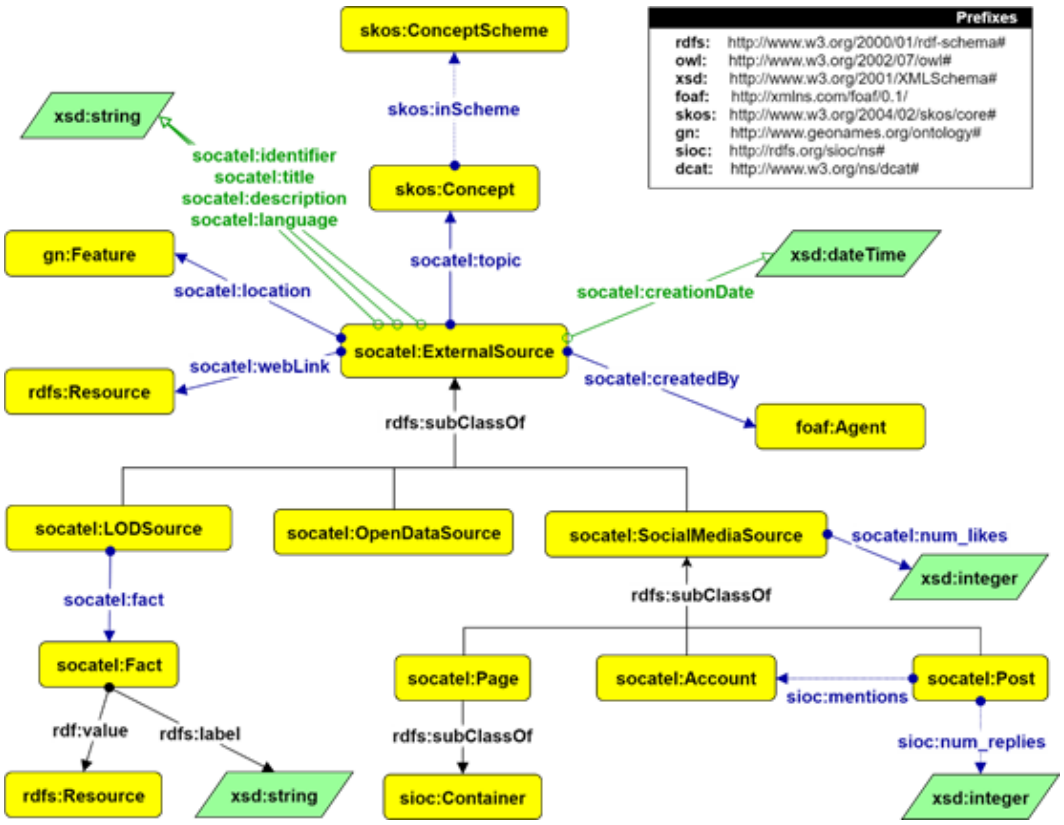

Figure 6 – SoCaTel external sources ontology

5.2.2 Mapping of external sources with the ontology

The Semantic Pre-processing layer, detailed in deliverable D4.1, is responsible for converting the data retrieved by the Data Handlers from the external sources. This layer aims to automatically map the structure of the incoming data with the SoCaTel external sources ontology. Based on this mapping, the data will be converted to RDF in order to be stored in the knowledge base and later exploited by the SRE to provide recommendations.

The tables below show the mappings of the main concepts of the ontology with the output of each Data Handler responsible for a specific type of external sources used, namely Social Media (Facebook and Twitter), Open Data and Linked Open Data Handlers. The format of the data retrieved from each of these sources is detailed along with examples in deliverable D4.1.

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

5.2.2.1 Twitter Data Handler

Twitter Handler Output	SoCaTel Ontology
Root Object	socatel:Post
→ **id**	→ socatel:identifier
→ **full_text**	→ socatel:description
→ **created_at**	→ socatel:creationDate
→ **lang**	→ socatel:language
→ **place** → → **name** → → **full_name** → → **country_code** → **geo** → → **long** → → **lat**	→ socatel:location → gn:Feature → → gn:name → → gn:alternateName → → gn:country_code → → gn:long → → gn:lat
→ **entities** → **hashtags []**	→ socatel:topic → skos:Concept []
→ **entities** → **user_mentions[]** → → **screen_name** → → **name**	→ sioc:mentions → sioc:UserAccount[] → → sioc:name → → sioc_n:fullName
→ **entities** → **urls []**	→ socatel:webLink []
→ **user:** → → **id** → → **name** → → **description** → → **expanded_url** → → **location** → → **lang** → → **followers_count**	→ socatel:createdBy → socatel:Account → → socatel:identifier → → socatel:title → → socatel:description → → socatel:webLink → → socatel:location → → socatel:language → → socatel:num_likes
→ **retweeted_status**	*Same mapping as the Root Object*
→ **retweet_count**	→ socatel:num_replies
→ **favorite_count**	→ socatel:num_likes

Table 6 – Twitter Data Handler

5.2.2.2 Facebook Data Handler

The format of the Facebook Data Handler's output varies depending on whether it retrieves a Facebook page or a specific Facebook post. It is also important to note that the Facebook API does not include a language attribute for any of its output formats. As this is an essential attribute for the recommender, one solution would be to automatically detect the language of Facebook posts or pages in order to create the *socatel:language* attribute.

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

Facebook Handler Output (Page)	SoCaTel Ontology
Root Object	socatel:Page
→ id	→ socatel:identifier
→ name	→ socatel:title
→ about	→ socatel:description
→ category_list []	→ socatel:topic → skos:Concept []
→ link	→ socatel:webLink
→ website	→ socatel:webLink
→ location	→ socatel:location → gn:Feature
→ posts [] / visitor_posts [] **→ → created_time** **→ → message** **→ → story** **→ → id**	→ sioc:container_of → sioc:Post [] → → sioc:created → → sioc:content → → sioc:content → → sioc:id
→ fan_count	→ socatel:num_likes

Table 7 – Facebook Data Handler (Page)

Facebook Handler Output (Post)	SoCaTel Ontology
Root Object	socatel:Post
→ id	→ socatel:identifier
→ message	→ socatel:description
→ created_time	→ socatel:creationDate
→ from	→ socatel:createdBy → foaf:Agent
→ permalink_url	→ socatel:webLink
→ coordinates	→ socatel:location → gn:Feature
→ likes [] / reactions [] **→ → id** **→ → name**	← sioc:likes ← socatel:Account → → socatel:identifier → → socatel:title
→ comments []	→ sioc:has_reply → socatel:Post
→ likes[].size + reactions.size[]	→ socatel:num_likes
→ comments[].size	→ sioc:num_replies

Table 8 – Facebook Data Handler (Post)

5.2.2.3 Open Data Handler

Open Data Handler Output	SoCaTel Ontology
Root Object	socatel:OpenDataSource
→ author	→ createdBy → foaf:Agent
→ → autrhor_email	→ → foaf:mbox

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

→ **id**	→ socatel:identifier
→ **issued**	→ socatel:creationDate
→ **language**	→ socatel:language
→ **title**	→ socatel:title
→ **notes**	→ socatel:description
→ **owner_org**	→ socatel:createdBy → foaf:Agent
→ **spatial_other**	→ socatel:location
→ **theme**	→ socatel:topic → skos:Concept
→ **url**	→ socatel:webLink
→ **tags []**	→ socatel:topic → skos:Concept []

Table 9 – Open Data Handler

5.2.2.4 Linked Open Data Handler

The Linked Open Data Handler is different than the other handlers in the sense that the data it provides does not conform to a predefined format. Instead, each LOD source has its own ontology and thus having a one-to-one direct mapping with the SoCaTel external sources ontology is not practical. For this reason, the data retrieved from Linked Open Data sources will be mapped to the generic *socatel:Fact* concept and the essential metadata about the source will be created and mapped to the ontology's main properties.

Linked Open Data Handler Output	SoCaTel Ontology
LOD source	socatel:LODSource
→ **URL**	→ socatel:webLink
→ **Retrieval date**	→ socatel:creationDate
→ **Topic used to query source []**	→ socatel:topic → skos:Concept []
→ **Location used to query source**	→ socatel:location → gn:Feature []
→ **Triples retrieved from source []**	→ socatel:fact → socatel:Fact → → rdfs:label → property name (*triple predicate)* → → rdf:value → property value (*triple object)*

Table 10 – Linked Open Data Handler

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

5.3 Component Architecture

5.3.1 SoCaTel Recommendation Engine – Architectural Component

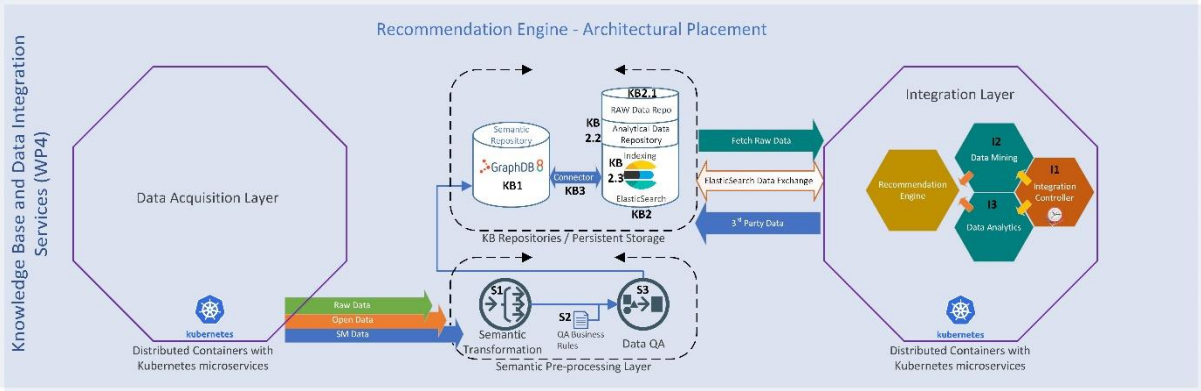

Figure 7 – Architecture Extract - Recommendation Engine Architectural Placement (Figure in Deliverable 3.2)

Figure 7 outlines the architectural placement of the SoCaTel Recommendation Engine (SRE) within the overall architecture of SoCaTel Knowledge Base and Data Integration Services (WP4) which is already outlined in D3.2 SoCaTel Platform Concrete Architecture Design. The SRE is part of the Integration Layer. The data that is being used by the SRE are provided by the Data Mining, Data Analytics and from the Knowledge Base (KB) in either semantic and/or raw form (directly obtained from the Elasticsearch Database storage engine). The SRE then projects its recommendations in a form of a microservice WebAPI to the SoCaTel platform in the following recommendation service types:

1. On-Demand Recommendations: Recommendations of this type include cases where a platform event will initiate a request to the Knowledge base backend component to fetch any recommendation regarding a user or one or more co-creation groups' user participants.
2. Periodic-Run Recommendations: Such recommendations are the ones generated in a periodic manner and their aim is to inform system administrators, moderators, co-creation group creators, platform users on various users or services that would improve the overall participation and engagement of the platform. An example would be a recommendation that could inform users to create co-creation groups based on under-utilized services [SRE.6]. Such

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

recommendations run in a period manner and can be displayed to users in an email thread or platform notifications.

3. System Recommendations: This type of recommendations can be addressed to a co-creation moderator to indicate potential users that could join a specific co-creation group based on their expertise (so that they can be invited) and also on the other hand to indicate users that potentially may need to be removed from one or more co-creation groups, or even from the platform, by analysing their written content and/or by obtaining other users' complaints. This last portion is different from a direct user report of abusive behaviour as it is system induced.

5.3.2 Identification of service needs

Service needs are derived from *D1.1 – Co-design of a multi-stakeholder co-creation platform for better access to long-term care services* and D3.1 Platform interaction plan and SoCaTel reference model.

The above deliverables are highlighting the important suggestions that the Knowledge base (WP4) should provide to the platform for enriching the usability and functionality of the SoCaTel Portal. The content of the Portal will be enriched using the SRE. The use of SRE will offer the following recommendations and suggestions:

SRE.1	**Suggest to User one or more co-creation groups to join based on user profile**
Purpose	The SRE will suggest to a user a co-creation group to join based on a user's profile which consists of the following main characteristics which is defined by the following input.
Recommendation Service Type	Periodic-Run
Input	Below is every information is maintained for a user in the SoCaTel platform and this is being used as an input to the methodology explained next: Location (e.g. Area/Municipality, City, Town, Country, etc.) Date of Birth (using an age scale e.g. ages from 18-25, 25-35, 35-50, 50-65, >65) Profession (e.g. Field of Profession, Actual Profession e.g. Social Service Worker, Programmer, Organization Owner, etc.) Interests (defined on user registration)

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

	Skills (defined on user registration) Spoken Languages Participating Co-Creation Groups The input is consumed from the semantic ontologies (KB1) and/or the Elasticsearch repository (KB3.2).
Methodology	A recommendation per user would be to find all different content explained above, within existing co-creation groups. This will match user profiles with the existing information taken from one or more co-creation groups: Location → $User_i$ is in the same location as all users within a $Co\text{-}CreationGroup_g$ Date of Birth → $User_i$ *belongs to the same date range* as all users within a $Co\text{-}CreationGroup_g$ Profession → $User_i$ *has the same profession* as all users within a $Co\text{-}CreationGroup_g$ Interests→ $User_i$ *has the same interests* as all users within a $Co\text{-}CreationGroup_g$ Skills→ $User_i$ *has the same set of skillsets* as all users within a $Co\text{-}CreationGroup_g$ Language → $User_i$ *speaks the same languages* as all users within a $Co\text{-}CreationGroup_g$ We can use a combination of filters to improve the overall ranking that we will purpose new co-creation groups.
Recommendation Filter	For all the above input we will not be using any specific Recommendation Filter for some of the filters outlined above such as [Location, Profession, Date of Birth, Profession, Language] but instead we will run basic queries and join operations on the Knowledgebase repositories. On the other hand, for filters such as [Interest and Skills] we can use a Content-based recommendation to find groups similar to $User_i$ based on e the interests and skills of existing groups members of a $Co\text{-}CreationGroup_g$
Output Example	Output consists of a ranked list of co-creation groups that are given to a user as a potential suitable choice of co-creation group to join based on location, age range, profession and spoken languages e.g. 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 11 – SRE.1 Suggest to User one or more co-creation groups to join based on user profile

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

SRE.3	Suggest to User one or more co-creation groups to join based on similar co-creation groups
Purpose	The SRE will suggest to a user one/more co-creation group(s) to join based on the existing co-creation groups that the user already participates.
Recommendation Service Type	On-Demand
Input	The input of these recommendations is i) co-creation group's metadata and fields which are preserved by the platform, ii) exchanged written content within one or more co-creation groups and iii) *hybrid recommendation* approaches that involve both the previous. The input is then used for clustering, classification, etc. suitable and usable for each methodology and recommendation filter. We will construct vectors and/or signatures and any other form which is suitable for comparing a $Group_i$ and a $Group_j$
Methodology	The methodology which is followed in this case involves the following covered scenarios. *Using the co-creation group's metadata and fields [keywords, initiator, co-creation topics, location, etc.]* we will construct vectors and/or signatures, etc. that will enable us to *perform similarity operations on these structures* and at the end to group using clustering similar groups. This is an *item-to-item collaborative filtering* Using the semantic ontologies, we will attempt to perform *Content-Based filtering* on what is being written in one or more co-creation groups. We will deploy *NLP, semantic analysis, etc.* to everything is written in one or more co-creation groups and we will attempt to find similar groups. These similar groups, based on the existing groups that a $User_i$ is participating, will be ranked in accordance to their similarity. *Multimodal Collaborative filtering* involve a mixture of both the two methodologies explained above (and/or SRE.1, SRE.2, etc.) and their aim is to improve the overall ranking result.
Recommendation Filter	For all the above Input we will be using *Item-to-Item collaborative filtering methods.* We will attempt an item-based-vector-representation and/or signature-vectors by calculating using multiple similarity functions, such as one of the following (Jaccard Similarity, Euclidean Distances, k-NN, Multi-Variate Distance Functions, TF-IDF, MinHashing, Bloom Filters, etc.). Similarity is then used as an input to the clustering/classification, clustering matrix filtering, etc. Results are then ranked using the various methodologies applied and explained in the previous section.
Output Example	A detailed output is shown below with explanatory examples. Let us assume that groups are represented in a clustering matrix vector using their metadata representator, initiator, co-creation topic and location accordingly. By applying an n-to-n similarity operation between the various groups we can calculate which groups are like one another. We can then find similar groups in comparison with

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

	existing groups that $User_i$ participates to. We can also deploy a ranking method that will consider the SRE.1 and SRE.2 outcome (similar users that participates to the newly identified similar co-creation groups) Using the semantic ontology, we can extract everything is ever written in one or more co-creation groups and we can apply either the bag-of-words, TF-IDF, etc approaches to represent the content of one or more co-creation groups. Next, we can correlate and calculate the content with other co-creation groups' content resulting in finding similar co-creation groups to recommend. Same as before, we can deploy a ranking method that will consider the SRE [1-2] outcome (similar users that participates to the newly identified similar co-creation groups). An example of a Multimodal Collaborative filtering could initially take into consideration any of the above *item-to-item collaborative filtering* along with any of the SER1 and SRE2 *user-to-user collaborative filtering.*

Table 12 – SRE.3 Suggest to User one or more co-creation groups to join based on similar co-creation groups

SRE.4	**Suggest Services relevant to a user**
Purpose	The SRE will suggest to a user services that need improvement based on his/her defined characteristics. An example of these characteristics is his/her location, language, profession, etc.
Recommendation Service Type	Periodic-Run
Input	The input consists of every known service to the semantic ontology. The services are being given as an input to an information retrieval/extraction engine and via the use of NLP, Semantic and Affective analysis, etc tools, it is extracted on whether a service has a low overall rating and/or satisfaction.
Methodology	The input described above is then being given to a *content based collaborative filtering algorithm* that will rank and filter out based on a user's collaborative approach which of the services would be purposed to a user as a new suggested co-creation group initiation topic.
Recommendation Filter	The methodology explained below will be used with a Content-Based Collaborative filtering to find and suggest to a user new co-creation group to be created. In addition, we will be using *User-to-User collaborative filtering methods.* We will attempt a user-based-vector-representation and/or signature-vectors by calculating using multiple similarity functions, such as one of the following (Jaccard Similarity, Euclidean Distances, k-NN, Multi-Variate Distance Functions, TF-IDF, Min-Hashing, Bloom Filters, etc.). Similarity is then used as an input to the clustering/classification, clustering matrix filtering, etc. This can be also taken from SRE [1-3]. This is then used for suggesting a user on which other users to include in one or more co-creation groups that

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

	will add value to the discussion.
Example	An example of the above methodology is to scan the existing semantic ontology and discover amongst the existing declared services of an organisation that "dog-walking" services in Barcelona that are under-utilized. Therefore, we will recommend to a $User_i$ which is a resident in the same city and that s/he is a programmer that these services need improvement. Whereas, a $User_k$ which is a social worker in Barcelona, we will also suggest him/her to join in this group and co-create to contribute to the overall good of his/her region. Both these recommendations are considering specific context, which is the user's location, profession, etc. and also a service's content which is derived by applying information retrieval techniques on the content of these services from the semantic ontology.

Table 13 – SRE.4 Suggest Services relevant to a user

SRE.5	**Suggest to group moderator organizations that work on related services with regards the group topic and discussion**
Purpose	The SRE will suggest to a group moderator related services that are similar to a group's topics and discussion. The organization's topics of interest are declared during the organization registration phase and are also extracted from the various co-creation groups that an organization participates. Services, similarly, are also declared by an organization during registration, along with their respective characteristics.
Recommendation Service Type	System
Input	The input consists of the following: Organizations' topics of interest, location, declared services, etc. are represented in a form that can be used to discover similarities with other services and/or groups and services. Co-creation groups replies, post, user interaction, etc. are represented in a form that can be used to discover similarities with other organizations and services.
Methodology	The input described above is then used to discover similarities between of the following pairs: Related Services – Co-Creation Group (*Content-Based Filtering* to match content and other services). In detail the content of a co-creation topic is parsed and analysed with various methods such as NLP, information retrieval/data extraction techniques, etc and a signature, vector representation, etc is being constructed. Same is applied regarding an organization's registered services using Content-Based filtering. The final stage of this recommendation is for the approach to identify and purpose similar services which are applicable to the co-creation group and suggest an organization (*that the service belongs to*) to participate to the co-creation group. Related Services *(Content-Based Filtering*), *for identifying similar services with the ones closer to a group.* Same as above, we will

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

	perform a content-based filtering amongst existing services and rank the organizations based on their participation to similar co-creation groups. The outcome can even be clustered and use distance similarity functions to purpose the closer organizations to the existing organizations already participating in the co-creation group of interest *Item-to-Item Collaborative Filtering* between existing Services of Organization and existing services of one or more co-creation groups. Using this filter, we can purpose new services that are a potential match and added value to one or more co-creation groups based on their interest.
Recommendation Filter	The methodology explained below will be used with a Content-Based Collaborative filtering to correlate one or more co-creation groups with existing services. The algorithm will then purpose to the moderator organizations that work on related services with regards to the group topic and discussion
Example	An example of the above recommendation is using one or more co-creation groups that the main topic of discussion is that "in my town (Barcelona) there is no dog walking service": Perform text mining/analytics to understand and extract more on the topic of interest. Create a signature and/or similarity matrix and perform a Content-Based Filtering with the existing declared services to identify services that are useful to such discussion topics. The recommendation will provide to the moderator with existing services that provide such/or similar solutions Like the above we will also offer the same suggestions to the moderator with a minor difference. We will identify other similar co-creation groups with the same discussion topics, and we will purpose services that also participate to these co-creation groups. We will rank the outcome based on which organizations are participating to the co-creation group that the recommendation is about. We will collect using an item-to-item collaborative filtering approach similar co-creation groups and we will extract what services are being used. The main motivation behind this recommendation is that by identifying similar co-creation groups we are also identifying similar services or services which are most probably used to serve and potentially solve the same problem. Such suggestions are equally important recommendations to similar co-creation groups

Table 14 – SRE.5 Suggest to group moderator organization that work on related services with regards the group topic and discussion

SRE.6	Suggest service paradigm within co-creation groups
Purpose	Suggest service paradigm within co-creation groups will periodically run on the entire service listing in the SoCaTel knowledge base and perform a content-based filtering analysis and item-to-item filtering analysis with clustering. This will create clusters of common services based on specific context which can be the language, service topic,

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

	service rating, etc
Recommendation Service Type	Periodic-Run
Input	The input, as described above is the Services' content and metadata information which is stored in the Knowledgebase of the SoCaTel platform.
Methodology	The information is then transformed into vectors and/or signatures or in any other form. The signature-based form is then clustered based on specific context which can be the language, service topic, service rating, etc. The data are then being clustered into clusters that represent services having common context such as language, interest, same user feedback, etc. Moreover, a *Content-Based Filtering* identifies services that are like one-another. This can be done using many recommendation approaches such as bag of words, TF-IDF, etc. Last, a *Hybrid Recommendation Filtering* approach that combines both the outcome of the two above methodologies with a content-based analysis on one or more co-creation groups will suggest service paradigms that are related within one or more co-creation groups based on their successfulness or unsuccessfulness rate.
Recommendation Filter	Recommendation filters, as already explained in the sections above, mainly involves *Content-Based filtering*.
Example	An example of the above recommendation is one or more co-creation groups that the main topic of discussion is that "in my town (Barcelona) there is no dog walking service". We will perform a Content-Based analysis on everything written in the co-creation group and we will cross-validate it with every service known to the knowledgebase that is also about this topic. Upon each service a Content-Based analysis is performed so as to identify which service is more closely related to this co-creation group in accordance to their crowd-acceptance value.

Table 15 – SRE.6 Suggest service paradigm within co-creation groups

SRE.7	**Suggest Resources to co-creation groups**
Purpose	The purpose of this recommendation is to propose resources [Research and Statistical Data, Governmental Data, Linked Open Data] relevant to what one or more co-creation groups topic is about. The latter is derived from what is being written in the co-creation group
Recommendation Service Type	Periodic-Run + System
Input	The input of this recommendation is taken from the semantic ontologies and it is the content of the co-creation group.
Methodology	The content is then being analysed using text processing techniques such as NLP, text mining, information retrieval, etc in which the main topics of interest are extracted. These topics of interest reflect what is being mentioned in one or more co-creation groups. This information in addition to a context, such as location, language, etc. is being used

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

	to retrieve relevant resources to the context. These resources are enhancing the co-creation process and are helping the co-creation group's participants to fully utilise every bit of information the platform is aware and co-create using every tool necessary. These are particularly helpful in the ideation phase where the users ideally could contribute the most having in front of them the full picture.
Recommendation Filter	Initially for the above input we will be using *Content-Based Recommendation Filter* to identify similarities between co-creation groups' and resources' content. Using this, we will identify similar services towards a co-creation group's content.
Example	An example of the above recommendation is one or more co-creation groups that the main topic of discussion is that "in my town (Barcelona) there is no dog walking service". The recommendation would suggest to the users that participate to the co-creation groups every resource which is relevant to the "dog walking". Such services include underutilized services and/or services that work in other cities to act as an example, etc.

Table 16 – SRE.7 Suggest Resources to one or more co-creation groups

5.4 SoCaTel Recommendation Engine Technical Documentation

5.4.1 General Description

This section is a detailed description of the components that the SoCatel Recommendation Engine consists of. The SRE is inhered from several components, some of which are already part of the SoCaTel platform architecture as can be seen in **Figure 7** above. The general categorization of the components can be defined as:

- Detect context filters based on request -> Resolution of Requested Recommendation
- Data query from database (WP4 platform Acquisition Layer, Data Acquisition, Data Mining components) -> Data Cleansing
- Data Analytics (WP4 platform component I3) -> Information Retrieval
- Prepare data for the corresponding recommendation system -> Data Representation/Transformation
- Use the appropriate recommendation system -> Filters
- Calculate similarities between recommendations -> Similarity Calculation
- Get previous recommendation responses -> User Feedback
- Recommendation results returned -> Recommendation(s)

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

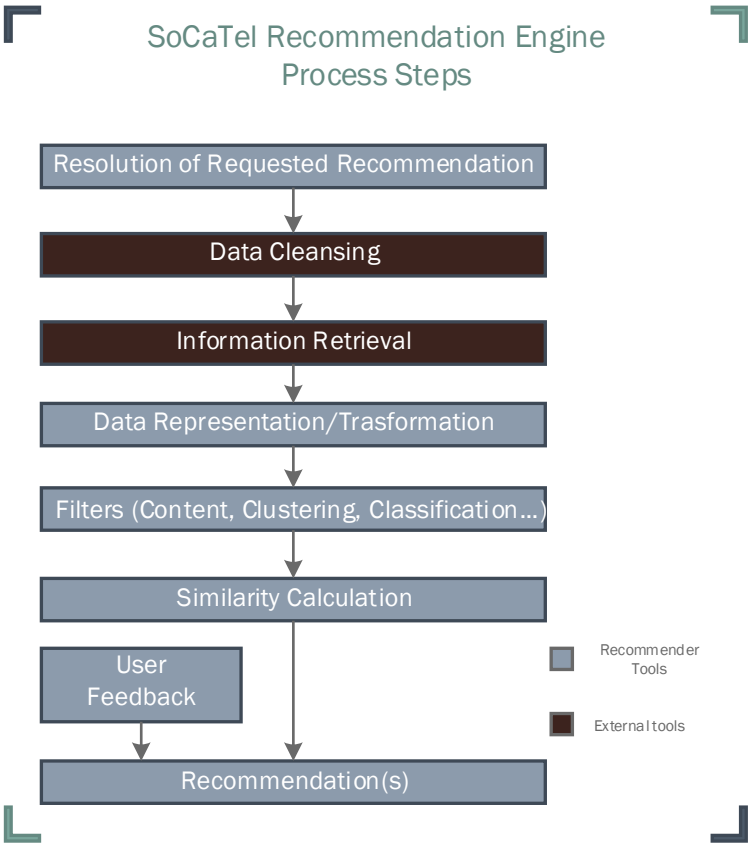

Figure 8 – Recommendation Engine Steps

Based on this general categorization, Figure 8 represents a more analytical representation of each component and further describes the steps that the recommendation engine will follow.

5.4.1.1 Resolution of requested Recommendation

This component is a reference to the initial call of the recommendation engine. There are a few different factors that play a role on which type of recommendation system should be used but also the contexts which need to be considered when making the weights for the recommendations. These factors can be defined as:

- The platform screen that the user is making the request
- The role of the user that is making the request (group moderator, group user)
- The type of recommendation that the user is requesting based on the two previous points

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

A more detailed description of these scenarios can be viewed in the section 3.3 above.

5.4.1.2 Data Cleansing and Information Retrieval

These steps are general components of the platform. They represent the way that the data acquired are reduced to the data that are necessary for the recommendations. The raw data along with the statistical and analysed data will be retrieved from specialised databases.

5.4.1.3 Data representation/Transformation

Here, the gathered data are transformed in a representative form, making them ready for the next step (filtering). The data will be transformed in the required structure (based on the ontologies defined), so they can match the expected input form of the methods that are going to be used.

5.4.1.4 Filters and Similarity Calculations

This is where the core functionality of the recommendation engine lies. Where most of the methods, algorithms and recommendation systems -described on Chapter 3 - are utilised. For each scenario that requires the use of the recommendation engine, a predefined set of methods is to be used, consisting of the appropriate recommendation system and methods to be used, making sure that the optimal results are acquired on each situation.

5.4.1.5 User Feedback

User Feedback is the feedback acquired directly from user actions regarding an item of a previous recommendation. The feedback will act as a rating weight on creating specific recommendation systems that make use of the user feedback. User feedback can be in the form of:

- The user upvoting/downvoting (Like or unlike) the recommendation item
- The recommendation item was chosen from a user during a previous recommendation

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

- The traffic (a sum of views, likes, discussions) around the recommendation item.

5.4.1.6 Recommendations

The recommendations are returned in a form of ranking, from the best matching to the least matching recommended item. Furthermore, the choice of a recommended item will be stored along with any possible rating the user gives to the item for future use in the User Feedback section described above (5.4.1.5).

5.4.2 Context Recommendation Engine Structure

This section makes a more detailed definition and a more specific focus on the recommendation engine structure as can be seen in **Figure 9**. The following subsections will each have a more in-depth approach about the processes the recommendation engine will implement and details regarding the route that the data will follow, from their query and retrieval of the data from the databases, to their built as recommendations.

Figure 9 and the rest of Section 5.4.2 comes to complement **Figure 8** and Section 5.4.1. This is achieved by giving a more detailed reasoning of each process described on **Figure 8** in the form of components to be used in 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.

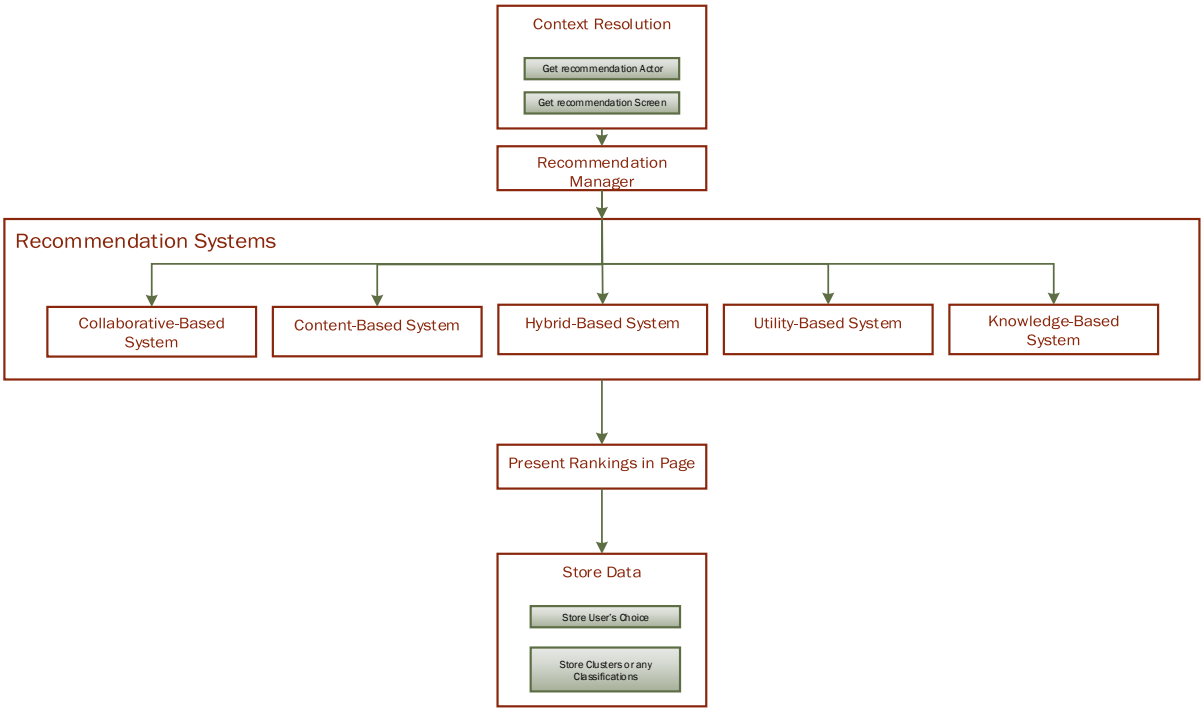

Figure 9 – Recommendation Engine Structure

5.4.2.1 Context Resolution

Part of giving a good answer is having a good understanding of the question. This initial step takes into consideration data regarding the context of the request, ensuring that a "correct" question was set for the system to answer. In other words, this section is responsible for feeding the system with information regarding:

- Who is the user making the request?
- What role is making the request? (Group creator, group moderator, simple user)
- From which part of the platform is the request coming from? (Group, user home, service search, etc.)

This component corresponds to the "Resolution of Requested Recommendation" in the processes diagram on **Figure 8**.

5.4.2.2 Recommendation Manager

Recommendation Manager is responsible for taking into consideration the results of context resolution (see above section) and deciding which set of methods are to be used for optimal results.

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

The set of methods are predefined according to the possible use case scenarios as they are detailed described in section 5.3.2.

The “Recommendation Manager” describes what part of “Filters” (**Figure 8**) process will be the most well suited for resolving the recommendation as documented from “Context Resolution” section

5.4.2.3 Recommendation Systems

Eventually the recommendation manager will have to choose which recommendation system or which combination of recommendation systems is to be used. Each recommendation has its own advantages and disadvantages regarding either the amount of data required to work, or the type of recommendations they have to calculate. Section 3.3 gives a more detailed description of the recommender systems and their advantages and disadvantages.

Another part to take into consideration when choosing a recommender system is how feasible this recommender systems are to implement. Section 4 gives a documentation on existing and ready (up to a point) to use frameworks. The ideal scenario would be a full implementation of all described recommender systems.

This component describes the processes “Filters” and “Similarity Calculation” in **Figure 8** and what possible recommendation systems available can achieve these processes. More detailed description regarding these processes can be found in Section 5.4.1.4.

5.4.2.4 Presentation

Regardless of what is the outcome of the recommender system, the output needs to take into consideration the rank of the item from previous users, from their feedback and if this item how many times this item was chosen when recommended. This type of ranking will act as a secondary recommendation ranking and will not in any case override any results that the recommendation engine will return.

The recommendations will be feed in the appropriate component that is responsible for presenting the recommendations.

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

5.4.2.5 Data Storage

The choice of recommended item, together with any other feedback given about the recommended items will be stored as a User Feedback dataset, to be used for future recommendations or for any other recommendation systems that require the use of user feedback.

The User Feedback is used from the recommendation engine as a “supervision” mechanism, that the recommendations presented to the users are indeed of their interests. Consequently, this can also be used as a feedback about the specific recommendation system and what can be done to be improved. The use of User Feedback in recommendations is more detailed described in Section 5.4.1.5.

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

6 Conclusion

This document focused on the SoCaTel Context-Aware Recommendation Engine. A service provided by the SoCaTel Knowledge Base and Data Integration Services layer and aims at assisting the user during the co-creation process. The service will utilize all data (internal and external) collected by the project and through their semantic translation into the SoCaTel defined ontologies will facilitate context aware recommendations. Such recommendations will take into account not only the preferences and past actions of the user, but also, it's current status and activity in order to provide the most relevant information.

As described in the document, the recommendation engine will provide a wide range of recommendation targeting all types of SoCaTel platform users and allowing them to streamline the Long-Term Care Service co-creation process.

The document presents a brief introduction to recommendation systems and context awareness, as a mean to introduce the SoCaTel Context-Aware Recommendation Engine. The engine will utilize state-of-the-art recommendation frameworks in order to implement the different recommendation systems needed for the SoCaTel co-creation flow.

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

7 Bibliography

[1] S. Lohr, “The New York Times,” New York Times, 21 September 2009. [Online]. Available: https://www.nytimes.com/2009/09/22/technology/internet/22netflix.html. [Accessed 27 May 2019].

[2] A. Gediminas, M. Bamshad, R. Francesco and T. Alex, “Context-Aware Recommender Systems,” *AI MAGAZINE,* pp. 67-80, FALL 2011.

[3] L. Xin, E. Martina, M. Jose-Fernan and R. Gregorio, “Context Aware Middleware Architectures: Survey and Challenges,” *Sensors,* vol. 15, pp. 20571-20607, 2015.

[4] D. Paul, “What we talk about when we talk about context,” *Personal and Ubiquitous Computing,* vol. 8, no. 1, pp. 19-30, 2004.

[5] F. Alexander, J. Michael, N. Gerald, R. Florian, R. Stefan and S. Martin, “Basic Approaches in Recommendation Systems,” in *Basic Approaches in Recommendation Systems, Recommendation Systems in Software Engineering*, Graz, Springer, 2014, pp. 15-37.

[6] A. Gediminas, S. Ramesh, S. Shahana and T. Alexander, “Incorporating Contextual Information in Recommender Systems Using a Multidimensional Approach,” *ACM Transactions on Information Systems (TOIS),* vol. 23, no. 1, pp. 103-145, 2005.

[7] Y. Zheng and Y. Yoneo, “A framework of context-awareness support for peer recommendation in the e-learning context,” *British Journal of Educational Technology,* vol. 38, no. 2, pp. 197-210, 2007.

[8] S. P. Christian, “Terra Incogniuta,” 12 September 2013. [Online]. Available: http://blog.christianperone.com/2013/09/machine-learning-cosine-similarity-for-vector-space-models-part-iii/. [Accessed 12 May 2019].

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

[9] Statistics, Laerd, "Laerd Statistics," Laerd Statistics, [Online]. Available: https://statistics.laerd.com/statistical-guides/pearson-correlation-coefficient-statistical-guide.php. [Accessed 26 May 2019].

[10] Wikipedia, "Wikipedia," Wikipedia, The Free Encyclopedia, [Online]. Available: https://en.wikipedia.org/wiki/Jaccard_index. [Accessed 30 May 2019].

[11] J. R. J, "Relevance Feedback in Information Retrieval," in *SMART Retrieval System - Experiments in Automatic Document Processing*, Prentice Hall, 1971, p. chapter 14.

[12] R. S and W. S, "Threshold Setting in Adaptive Filtering," *J. Documentation,* vol. 56, pp. 312-331, 2000.

[13] A. S. George, T. Aimilia and A. P. George, "Recommender Systems Review: Types, Techniques and Applications," in *Encyclopedia of Information Science and Technology (Third Edition)*, 2015, pp. 329-339.

[14] H. Thomas, "Probabilistic Latent Semantic Analysis," in *Uncertainty in Artificial Intelligence*, Stockholm, Sweden, 1999.

[15] O. Sean, A. Robin, D. Ted and F. Ellen, Mahout In Action, New York: Manning Publications Co., 2012.

[16] S. Asiri, "Towards Data Science," Medium, 11 June 2018. [Online]. Available: https://towardsdatascience.com/machine-learning-classifiers-a5cc4e1b0623. [Accessed 27 May 2019].

[17] R. S. Brid, "Medium," Medium Corporation, 26 October 2018. [Online]. Available: https://medium.com/greyatom/decision-trees-a-simple-way-to-visualize-a-decision-dc506a403aeb. [Accessed 20 May 2019].

[18] S. Priy, "Geeksforgeeks," [Online]. Available: https://www.geeksforgeeks.org/clustering-in-machine-learning/. [Accessed 21 May 2019].

[19] S. J. Swarndeep and P. Sharnil, "An Overview of Partitioning Algorithms in

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

Clustering Techniques," *International Journal of Advanced Research in Computer Engineering & Technology,* vol. 5, no. 6, pp. 1943-1946, 2016.

[20] N. Viswarupan, "Towards Data Science," Medium.com, 10 July 2017. [Online]. Available: https://towardsdatascience.com/k-means-data-clustering-bce3335d2203. [Accessed 27 May 2019].

[21] F. Deng, "Utility-based Recommender Systems Using Implicit Utility and Genetic Algorithm," in *International Conference on Mechatronics, Electronic, Industrial and Control Engineering*, 2015.

[22] H. S. O. Research, "HSOR.org," High School Operations Research, [Online]. Available: http://www.hsor.org/what_is_or.cfm?name=mutli-attribute_utility_theory. [Accessed 23 May 2019].

[23] 101computing.net, "101computing.net," 5 February 2018. [Online]. Available: https://www.101computing.net/heuristic-approaches-to-problem-solving/. [Accessed 28 May 2019].

[24] S. Pawel and W. Andre, "Modeling in the Context of Computer Science - A Methodological Approach," *Studies in Logic, Grammar and Rhetoric,* vol. 20, no. 33, pp. 155-179, 2010.

[25] J. P. Michael and B. Daniel, "Content-Based Recommendation Systems," in *The Adaptive Web*, Berlin, Springer, 2007, pp. 325-341.

[26] P. Michael and D. Billsus, "Learning and Revising User Profiles: The Identification of Interesting Web Sites," in *Machine Learning 27*, Netherlands, Kluwer Academic Publishers, 1997, pp. 313-331.

[27] A. Gediminas and T. Alexander, "Toward the Next Generation of Recommender Systems: A survey of the State-of-the-Art and Possible Extensions," *IEEE Transactions on Knowledge and Data Engineering,* vol. 17, no. 6, pp. 734-749, 2005.

[28] B. Marko and S. Yoav, "Fab: Content-based, Collaborative Recommendation,"

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

Communications of the ACM, vol. 40, no. 3, pp. 66-72, 1997.

[29] B. Chumki, H. Haym and C. William, “Recommendation as Classification: Using Social and Content-Based Information in Recommendation,” in *AAAI-98 Proceedings*, 1998.

[30] C. Mark, G. Anuja, M. Tim, M. Pavel, N. Dmitry and S. Matthew, “Combining Content-Based and Collaborative Filters in an Online Newspaper,” in *ACM SIGIR Workshop*, Berkeley CA, USA, 1999.

[31] P. Michael, “A Framework for collaborative, Content-based and demographic filtering,” *Artificial Intelligence Rev.,* pp. 393-408, December 1999.

[32] S. Ian and K. N. Charles, “Combining Content and Collaboration in Text Filtering,” in *IJCAI'99*, Stockholm, Sweden, 1999.

[33] K. C. Michelle, D. L. David, M. David and P. Christian, “Bayesian Mixed-Effects Models for Recommender Systems,” in *ACM SIGIR Workshop Recommender Systems: Algorithms and Evaluation*, 1999.

[34] A. Asim, E. Skander and K. Rajeev, “Internet Recommendation Systems,” *J. Marketing Research,* vol. 37, no. 3, pp. 363-375, 2000.

[35] B. Robin, “Knowledge-based recommender systems,” *Encyclopedia of Library and Information Systems,* vol. 69, no. 32, 2000.

[36] H. G. Robert, “Merchant Differentiation through Integrative Negotiation in Agent-Mediated Electronic Commerce,” Massachusetts Institute of Technology, Massachusetts, 1998.

[37] C. A. Charu, “Content-Based Recommender Systems,” in *Recommender Systems The Textbook*, New York, Springer, 2016, pp. 139-166.

[38] A. G. Sielis, C. Mettouris, A. G. Papadopoulos, A. Tzanavari, M. G. R. Dols and Q. Siebers, “A Context Aware Recommender System for Creativity Support Tools,” *Journal of Universal Computer Science,* vol. 17, no. 12, 2011.

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

[39] B. Robin, “Hybrid Recommender Systems: Survey and Experiments,” *User Modeling and User-Adapted Interaction,* vol. 12, no. 4, pp. 331-370, 2002.

[40] X. Zhao, Z. Niu, K. Wang, K. Niu and Z. Liu, “Improving Top-N Recommendation Performance Using Missing Data,” *Mathematical Problems in Engineering,* vol. 1, no. 13, 2015.

[41] J. Tarus, Z. Niu and G. Mustafa, “Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning,” *Artificial Intelligence Review,* vol. 50, no. 21, 2018.

[42] W. IJntema, F. Goossen, F. Frasincar and F. Hogenboom, “Ontology-based news recommendation,” in *Proceedings of the 2010 EDBT/ICDT Workshops (EDBT '10)*, 2010.

[43] S.-T. Yuan and C. Cheng, “Ontology-based personalized couple clustering for heterogeneous product recommendation in mobile marketing,” *Expert Systems with Applications,* vol. 24, no. 4, pp. 461-476, 2004.

[44] A. C. M. Costa, R. Guizzardi, G. Guizzardi and G. P. Filho, “COReS: Context-aware, Ontology-based Recommender system for Service recommendation,” in *Proceedings of Workshop on Ubiquitous Mobile Information and Collaboration Systems (UMICS'07)*, 2007.

[45] I. F Cruz, H. Xiao and A. Lab, “The Role of ontologies in data integration,” *Journal of Engineering Intelligent Systems,* vol. 13, 2005.

[46] R. Touma, O. Romero and P. Jovanovic, “Supporting Data Integration Tasks with Semi-Automatic Ontology Construction,” in *Proceedings of the ACM Eighteenth International Workshop on Data Warehousing and OLAP (DOLAP '15)*, 2015.

[47] J. Ge, Z. Chen, J. Peng and T. Li, “An ontology-based method for personalized recommendation,” in *Proceedings of the 11th International Conference on Cognitive Informatics and Cognitive Computing*, 2012.

[48] Y. Z., N. Y., J. S., K. S. and M. K., “Ontology-Based Semantic Recommendation

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

for Context-Aware E-Learning," *Ubiquitous Intelligence and Computing,* vol. 4611, 2007.

[49] M. S.E., D. R. D. and S. N.R., "Ontology-based Recommender Systems," in *Handbook on Ontologies. International Handbooks on Information Systems*, Springer, 2004.

[50] N. Yanes, S. B. Sassi and H. H. B. Ghezala, "Ontology-based recommender system for COTS components," *Journal of Systems and Software,* vol. 132, pp. 283-297, 2017.

[51] Ekstrand, Michael D and Ludwig, Michael and Konstan, Joseph A and Riedl, John T, "Rethinking the Recommender Research Ecosystem: Reproducibility, Openness, and {LensKit}," in *RecSys '11 - Proceedings of the Fifth {ACM} Conference on Recommender Systems*, 2011.

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