MULTISENSOR

Mining and Understanding of multilinguaL contenT for Intelligent Sentiment Enriched coNtext and Social Oriented inteRpretation

FP7-610411

D7.4
First Prototype

Dissemination level:	Public
Contractual date of delivery:	Month 18, April 30th, 2015
Actual date of delivery:	Month 19, May 16th, 2015
Workpackage:	WP7 System Development and Integration
Task:	T7.4 System development
Type:	Prototype
Approval Status:	Final Version
Version:	1.0
Number of pages:	58
Filename:	D7.4_FirstPrototype_2015-05-15_v1.0.pdf

Abstract

This document describes the technical components and infrastructure of the First Prototype for the MULTISENSOR platform.

It provides an overview of the integration status of the different components, the improvements of the three UC applications, and the hosting infrastructure.

The First Prototype is the new version of the platform that integrates most of the services as a baseline version according to the requirements in Milestone 3.

Co-funded by the European Union

History

Version	Date	Reason	Revised by
0.1	30/03/2015	Document initiation	E. Jamin
0.2	10/04/2015	Initial comments	S. Vrochidis, D. Liparas (CERTH)
0.3	15/04/2015	First draft	E. Jamin (EVERIS)
0.4	22/04/2015	Draft with integrated inputs	E. Jamin, A. Callabed, A. Mas (EVERIS), All partners
0.5	27/04/2015	Review ready version	A. Mas (EVERIS)
0.6	30/05/2015	Prototype review	D. Liparas (CERTH)
0.7	12/05/2015	Updated version	E. Jamin (EVERIS)
0.8	13/05/2015	Prototype and document review	D. Liparas (CERTH)
1.0	15/05/2015	Final version	E. Jamin (EVERIS)

Author list

Organization	Name	Contact Information
EVERIS	Emmanuel Jamin	ejacques@everis.com
EVERIS	Axel Callabed	acallabe@everis.com
EVERIS	Alan Mas	alan.mas.soro@everis.com
EVERIS	Enric Staromiejski	enric.staromiejski.torregrosa@everis.com
EVERIS	Marcel Risques	marcel.risques.andersen@everis.com
CERTH	Stefanos Vrochidis	stefanos@iti.gr
CERTH	Dimitrios Liparas	dliparas@iti.gr
CERTH	Ilias Gialampoukidis	heliasgj@iti.gr
PRESSRELATIONS	Leszek Blacha	leszek.blacha@pressrelations.de
LinguaTec	Reinhard Busch	r.busch@linguatec.de
UPF	Gerard Casamayor	gerard.casamayor@upf.edu
BM-Y!	Iris Miliaraki	irismili@yahoo-inc.com
BM-Y!	Ioannis Arapakis	arapakis@yahoo-inc.com
ONTO	Boyan Simeonov	boyan.simeonov@ontotext.com
ONTO	Vladimir Alexiev	vladimir.alexiev@ontotext.com

Executive Summary

D7.4 presents the First Prototype description as a first integrated functional version of the MULTISENSOR system. The first prototype shows all the main services integrated in the platform and most of them in the User interface (UI).

The 1st prototype meets the requirements of Milestone 3. Specifically three different applications have been implemented (one for each use case) integrating the following online services:

1. UC1: Journalism application: a) Profile service, b) Contributor analysis service, c) Semantic search service, d) Translations service, e) summarisation service, f) Content delivery service, g) Clustering&filtering service.
2. UC2: Media monitoring application: a) Profile service, b) Contributor analysis service, c) Semantic search service, d) Translation service, e) Summarisation service, f) Content delivery service, g) Clustering&filtering service.
3. UC3: SME internationalisation application: a) Profile service, b) Contributor analysis service, c) Semantic search service, d) Content delivery service, e) Clustering&filtering service, e) Reference data service, f) Decision support service.

This document provides a technical overview of the development and integration of the First Prototype. Then, it represents a technical reference for the D7.4 deliverable of the MULTISENSOR platform.

Abbreviations and Acronyms

CMR	Central Multimedia Repository
CNR	Central News Repository
CSV	Comma Separated Values
DB	DataBase
DBMS	DataBase Management System
EBS	Elastic Block Storage
EC2	Elastic Compute Cloud
ECU	Elastic Compute Unit
FP	First Prototype
FTP	File Transfer Protocol
FTS	Full-Text Search
HTTP	HyperText Transfer Protocol
JPEG	Joint Photographic Experts Group
JSON	JavaScript Object Notation
MAF	Multimedia Application Format
MPEG	Moving Picture Experts Group
NER	Named Entity Recognition
NFR	Non-functional Requirement
OP1	Operational Prototype 1
OPS	OPerationS repository
PPA	Personal Package Archive
OWL	Ontology Web Language
RDBMS	Relational DataBase Management System
RDF	Resource Definition Framework
REST	Representational State Transfer
SIMMO	Socially Interconnected and MultiMedia-enriched Object
SOA	Service Oriented Architecture
SPARQL	SPARQL Protocol And RDF Query Language
TSV	Tab Separated Values
UC	Use Case
UCS	Universal Character Set
UI	User Interface
UTF	UCS Transformation Format
W3C	World Wide Web Consortium
XML	eXtensible Markup Language

Table of Contents

1 INTRODUCTION

In D7.1, a general roadmap and technical vision for the implementation of the MULTISENSOR platform was established. The user and non-functional requirements in D8.2 and the technical vision were combined in D7.2 to define the global architecture of the system and its subsystems, workflows and interfaces. The "walking skeleton" for the technical roadmap laid out in the D7.1 is presented below:

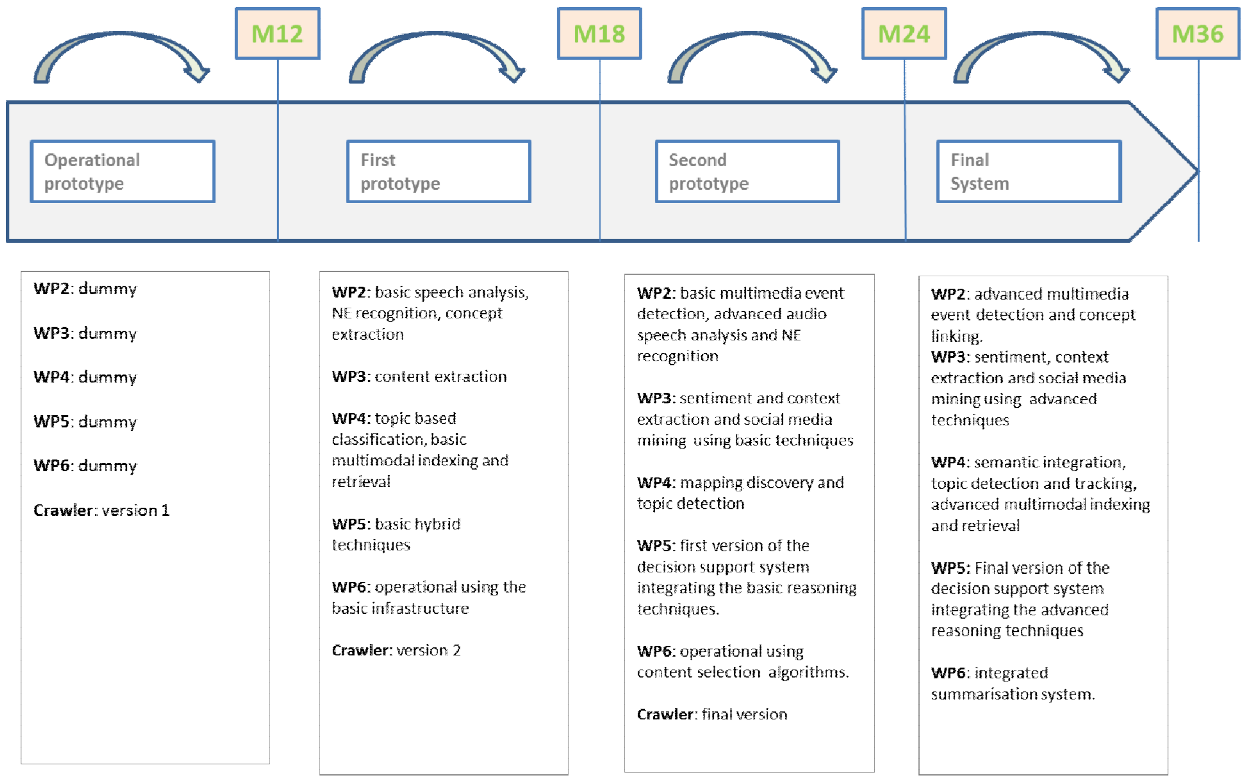

M12
M18
M24
M36
Operational prototype
First prototype
Second prototype
Final System
WP2: dummy
WP3: dummy
WP4: dummy
WP5: dummy
WP6: dummy
Crawler: version 1
WP2: basic speech analysis, NE recognition, concept extraction
WP3: content extraction
WP4: topic based classification, basic multimodal indexing and retrieval
WP5: basic hybrid techniques
WP6: operational using the basic infrastructure
Crawler: version 2
WP2: basic multimedia event detection, advanced audio speech analysis and NE recognition
WP3: sentiment and context extraction and social media mining using basic techniques
WP4: mapping discovery and topic detection
WP5: first version of the decision support system integrating the basic reasoning techniques.
WP6: operational using content selection algorithms.
Crawler: final version
WP2: advanced multimedia event detection and concept linking.
WP3: sentiment, context extraction and social media mining using advanced techniques
WP4: semantic integration, topic detection and tracking, advanced multimodal indexing and retrieval
WP5: Final version of the decision support system integrating the advanced reasoning techniques
WP6: integrated summarisation system.

Figure 1: Technical roadmap

In D7.3, a brief technical reference of the Operational Prototype was presented in the second milestone of the project MS2 (M12). The Operational prototype was a first rough UI for the platform and dummy implementations of the major services, processes and workflows.

The purpose of this document is to provide a brief technical reference for the D7.4 deliverable, which is the third milestone of the project MS3 (M18). D7.4 describes the implementation of the major services, processes and workflows of the First Prototype. Also, it contains the integration framework to connect the services implementations and their functional integration with the UI:

- **Section 2** contains a high-level technical overview of the First prototype.
- **Section 3** contains a description of the integration framework.
- **Section 4** contains a description of the demonstrator applications.
- **Section 5** provides a walk-through of the structure of the code, to assist in the navigation of the Subversion repository
- **Section 6** details the technical infrastructure hosting the prototype.
- **Section 7** contains links and details for accessing the demonstrator application for reviewers.

- **Section 8** presents a brief summary and conclusions.

2 FIRST PROTOTYPE ARCHITECTURE

2.1 Global view

The global architecture for the MULTISENSOR platform has been discussed deeply in D7.1 and D7.2.

The MULTISENSOR architecture is based on a SOA approach and encompasses two discrete modalities: offline asynchronous processing of harvested data (see D7.2, section 4.2.2), and synchronous retrieval, delivery and exploitation of the analytical data (see D7.2, section 4.2.3).

D7.4 explains the current status of the system in terms of repositories, services, processes and workflows of the First Prototype.

One of the most important efforts was focused on the services integration of the Offline modality to analyse the document and to populate the main repositories with analytic data. This allows delivering analytical data to the online services and improving the UI of the three UC applications.

2.1.1 Status of the Operational Prototype

In D7.3, the status of the Operational Prototype was presented. Here is a quick summary of the elements that were part of it.

- The first version of the cloud platform with the basic system configuration.
- Only the CNR and OPS repositories were available.
- A basic version of the Site collector (PR) was implemented.
- The main components were implemented as dummy services.
- The basic UI with dummy content were presented for the UC1 and UC3.

2.1.2 Objective of the First Prototype

The First Prototype is an improvement of the Operational Prototype. In this prototype version, most of the Online and Offline services are expected to be delivered as baseline versions. Here is the list of the goal to be achieved for the first SW development cycle of the project according to the requirements of Milestone 3:

1. Operational architecture including the initial module implementations:
    1. *content extraction*: basic speech analysis, NE recognition, concept extraction
    2. *user-centric content extraction*: context extraction
    3. *content integration and retrieval*: topic-based classification, basic multimodal indexing and retrieval
    4. *reasoner and decision support*: basic hybrid techniques
    5. *information production*: operational using the basic infrastructure.
2. Data wrappers and crawlers (second version)

2.1.3 Status of the First Prototype

The First Prototype is the complete integration of all the services (offline and online) and all the specific components (repositories and user interfaces). It is composed of the online and offline modalities.

The offline modality has to analyse all the information that is coming from the crawlers and is stored in the CNR repository. The Content Extraction Pipeline (see 3.2.1) is fully implemented and fully functional even if data models and validation have to be improved before populating the RDF repository. Also, the Content Alignment Pipeline (see section 3.2.2) is implemented and integrated as a baseline version. As shown on the Figure 2, only the Social Media Analysis (see section 3.2.3) is not provided due to technical issues with the Twitter API.

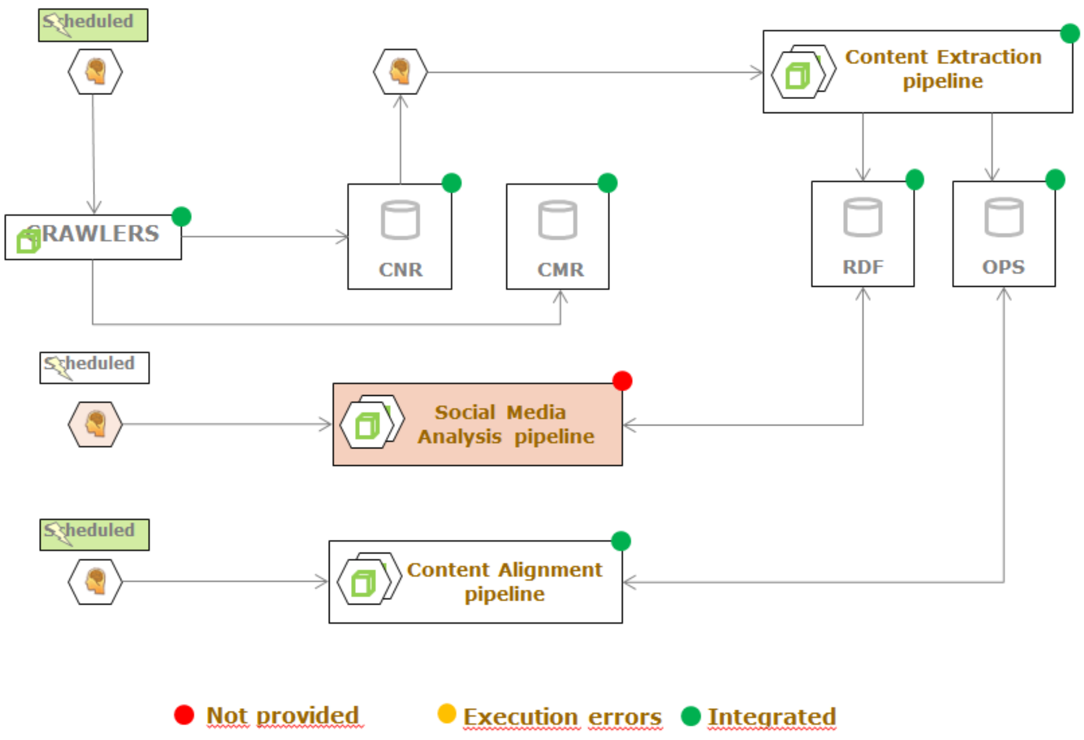


Figure 2: Offline modality of the First Prototype architecture

The online modality is the live connection between the user interfaces functionalities and the access to the knowledge available in the different repositories. As shown in the Figure 3, most of the services are integrated and connected with the repositories. As the RDF repository is not fully populated with the analytical data coming from the CEP, the index for the search functionality is still supported by the CNR (ElasticSearch).

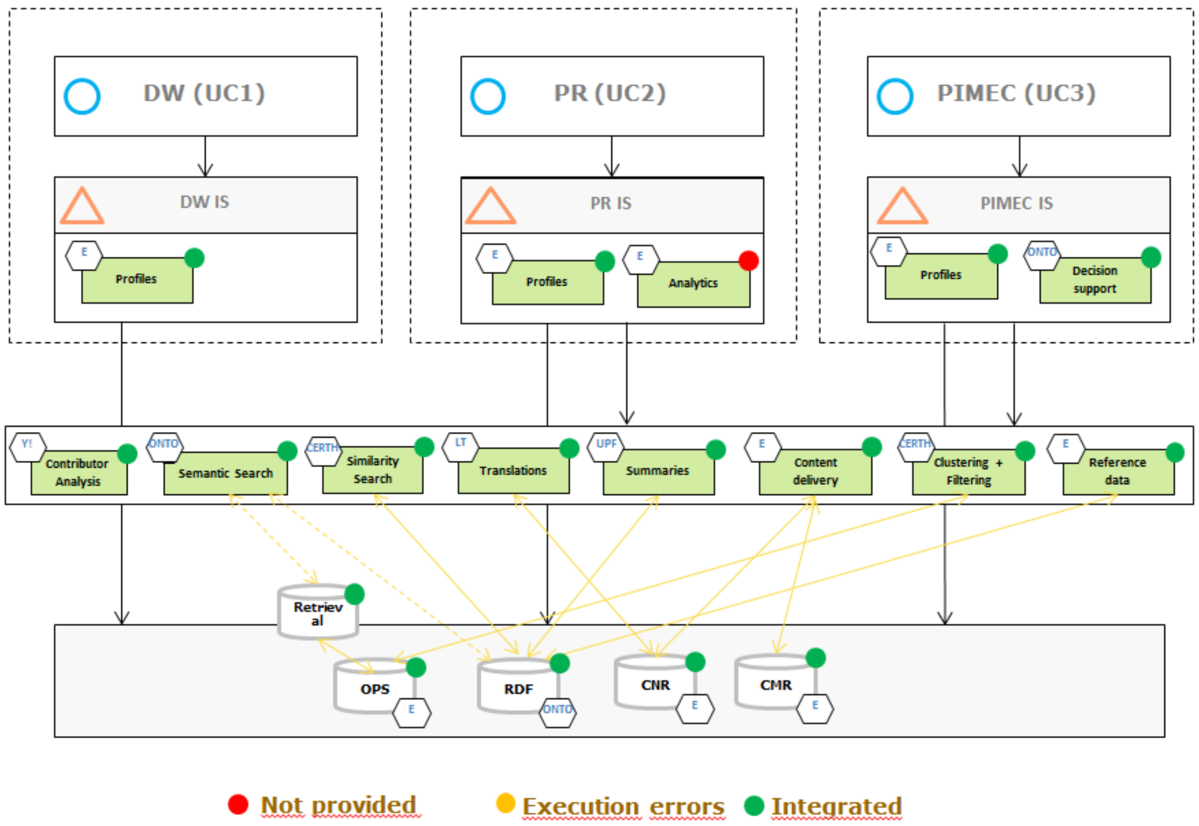


Figure 3: Online modality of the First Prototype architecture

The development phase of the First Prototype is a huge challenge to connect and integrate so many different services and technologies in the platform. Many technical issues regarding dependencies between those services and other libraries are addressed to obtain a stable and coherent integration and to start populating the repositories with analytical data.

In D7.4, the result of the First Prototype development is summarised as follows:

- The system was migrated in a most powerful server and the configuration has been improved.
- The four repositories of the Data Layer are in place (CNR, CMR, OPS and RDF).
- The two crawlers are implemented and are able to collect complementary documents (multimedia and social media).
- All the main services are implemented, only few are presented as a baseline version.
- The different services are integrated in the platform, i.e. they can interact between themselves, store and collect data from the different repositories.
- The CEP is fully functional but it is still under data models tuning to produce really high qualitative RDF content. After this step, the RDF repository will be fully populated with the CEP outputs.
- The UI for the three Use Cases are improved with the interaction of the available online services.

2.2 Configuration and supervision of the system

The system has been migrated on a more powerful server called Grinder (see section 6.1.3). Then the configuration and the supervision systems have been improved. The basic concepts were presented in D7.2 (see 4.1.4) and D7.3 (see 2.2).

As defined in the D7.2 document (see 4.2.2.1), the configuration of the server system is referenced in the bootstrap file. This file contains the main configuration to all the main components (repositories access) and the deployed services (available endpoints).

In the SOA approach, this file is deployed as a service to share all the services and repositories configuration and ensure a clear and common vision for the integration framework. Then, each service is connected to other services or repositories to exchange, store or collect data in the system.

The Supervisor system (see D7.2, section 4.2.2.1 and see D7.3, section 2.2) monitors and orchestrates execution of the different subsystems of the platform. It is responsible for scheduling background and offline processes, monitoring health of the system and providing other auxiliary services.

The supervisor has different tasks in charge,

1. it exposes the common tools to enable the SOA interoperability between all the services of the system
2. it orchestrates the execution of the different subsystems and
3. it monitors the health of the system and triggers specific services to maintain qualitatively the data collections in the different repositories.

For the First Prototype, it triggers 5 different jobs:

- Hosts a bootstrap service to provide all other services with the shared platform configuration.
- Runs the crawlers periodically to collect the documents and store them in the CNR repository.
    - The Site collector (PR crawler) is started every hour.
    - The Media collector (Yahoo! crawler) is started once per day.
- Initiates the Content Extraction Pipeline (CEP) on demand. The first test was initiating the CEP once a day for a pool of 20 documents.
- Initiates the Content Alignment Pipeline (CAP) on demand.
- Initiates the (SMAP) on demand.

2.3 Harvesting layer

The Harvesting Layer is responsible for capturing and aggregating multi-language, multi-source and multi-media content from a variety of sources into the platform, for analysis and exploitation (see D7.2, section 4.1.1).

This layer is composed by the following components:

- The crawlers (see section 2.3.1)
- The repositories (see section 2.3.2)

2.3.1 **Crawlers**

The operational prototype contained the first proof-of-concept implementation of the crawler engine and the PR crawler (see D7.2, section 4.2.2.2).

For the First Prototype, the two crawlers are implemented and integrated. The one that was already present in the Operational Prototype (Site collector) has been improved and consolidated. The other one (Media collector) has been newly implemented and integrated in the system.

2.3.1.1 **Media collector (PR)**

The First Prototype contains the full version of the site collector (the PR crawler) as defined in the D7.2 (see D7.2, section 4.2.2.2).

pressrelations aggregates news from media all over the world via proprietary crawling technologies and provides access to news items that have been preselected for topics relevant to the MULTISENSOR project (i.e. referring to the defined use cases).

News sites are crawled for selected keywords that are relevant to the three use cases. The keyword list has been created by the use case partners.

The Media Collector accesses the JSON-based-API at pressrelations and retrieves content incrementally as new articles are available.

For radio and TV articles no audio and video material is provided but textual information only. For internet articles links to multimedia material (images, videos, audio) are provided if it was possible to extract them with the PR crawlers. This is only the case for very few news sites due to the heterogeneous structure of the news sites that are being crawled.

Number of articles and kind of documents (monthly):

~ 110.000 internet items (news sites and social media)

~ 1-10 agency articles

~ 1-10 radio and TV articles (only up to September 2014)

~ 100 print articles

Previously, the Site collector collected a huge quantity of articles from the NewsRadar that were stored in the CNR. The multimedia resources available in those articles were not interpreted and stored in the system.

In this version, the crawler collects the news with different multimedia resources (images and videos) and permit to store them as multimedia files in the system. Images and videos are stored in the Content Multimedia Repository (CMR). Those multimedia resources are analysed by the Content Extraction Pipeline (see D7.2, section 4.2.2.3).

The site collector is configured with a list of documents to ask the NewsRadar platform to download new items. It assigns a unique ID to every item and stores it in the Central News Repository (CNR).

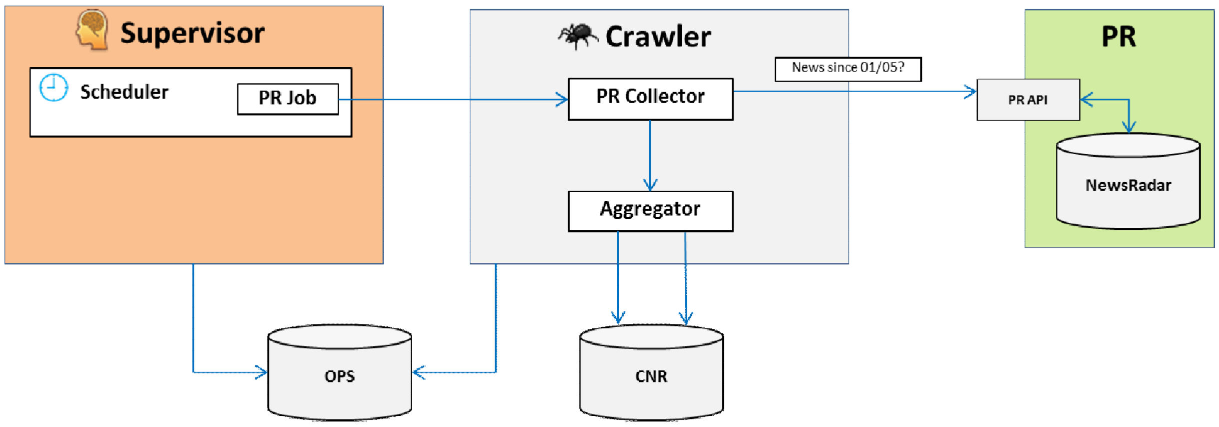


Figure 4: Media collector (PR)

Improvements of the PR API and the Site collector have been realised to collect more precisely the different fields required by the CEP services to analyse the data. Further development of the pressrelations crawlers and the pressrelations API is not foreseen.

2.3.1.2 Site & sociometrics collector (BM-Y!)

The Media collector (called the "Y Collector" for short) aggregates content crawled by the targeted crawler (see D7.2, section 4.2.2.2).

BM-Y! provides a crawler for collecting web pages given a seed list of news provider domains. The seed list is provided by the user partners and there can be a potentially different seed list for each use case scenario. The crawler will run on a daily basis collecting news articles from the respective media sources. In addition, BM-Y! also provided a sociometrics collector for Twitter, Linkedin, Reddit, Facebook, StumbleUpon, Delicious, Pinterest, and Google Plus metrics.

2.3.1.3 Technical description of the Site crawler:

The crawler is based on Apache Nutch (http://nutch.apache.org/). Nutch is a highly extensible and scalable open source web crawler software project. Nutch can run on a single machine, but gains a lot of its strength from running in a Hadoop cluster, which we currently allow. As a result, crawling is performed via MapReduce, where every MapReduce iteration has two steps, one for downloading the pages and the second for extracting the links.

The input to the crawler is a set of URLs that will serve as roots for the crawling process. The crawler by default uses a link depth of 2, meaning the number of "hops" a page can be away from the root (i.e., the URL of the source in the input seed list), where a "hop" means following a link on a page. However, depth can be parameterized allowing for crawling a larger or small set of pages.

After the crawling is completed, the set of crawled pages is first stored in Hbase which is the default storage system for the output of Nutch. Next, the HBase output is fetched and dumped to disk in JSON format.

All the functionality including the downloading and installation of the relevant components (i.e., Apache Nutch, Hadoop, HBase) is provided by the crawler script. The script has been already installed in a server provided by everis and is running periodically for performing crawling.

- **Input**: A list of URLs which will serve as seeds for crawling pages

- **Output**: A JSON object containing the following fields (this is in compliance with the PR crawler)
    - use_case, e.g., "UC1 - Household Appliances"
    - language, e.g., "en"
    - country, e.g., “DE”
    - crawled, i.e., the date crawled, e.g., "2014-06-17"
    - multimediaUrls, i.e., a set of multimedia URLs such as images included in the page
    - c_sourcecode, i.e., the HTML original source code
    - source, i.e., the seed that led to this page
    - date_timestamp, i.e., the date of the page
    - body, i.e., the textual content/body of the web page
    - url, i.e., the URL of the page
    - title, i.e., the title of the page

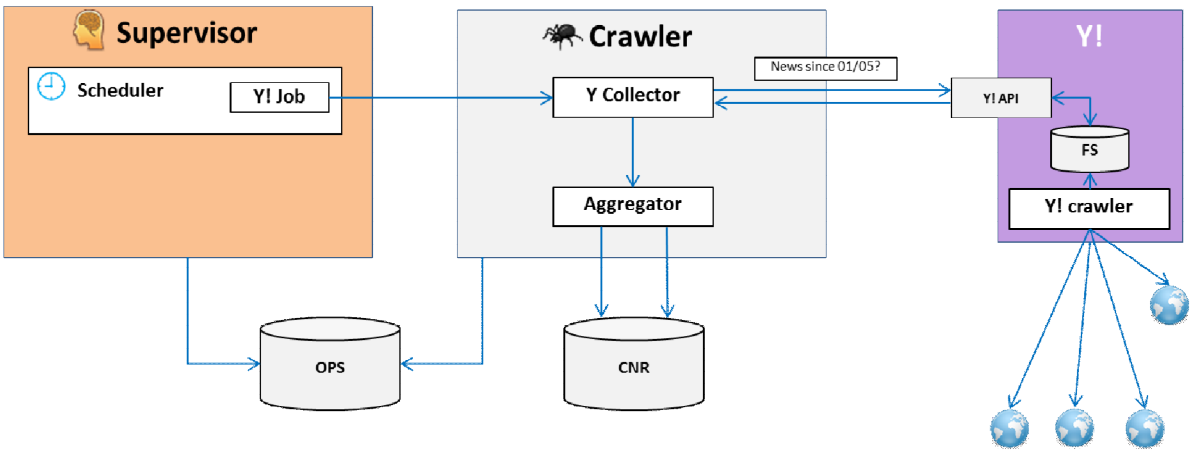


Figure 5: Site collector (Yahoo!)

2.3.1.4 **Technical description of the sociometrics collector:**

The sociometrics collector is a service for aggregating sociometric data in the form of time series. The service consists of a collection of wrappers for several social media platform API’s, and more specifically for (i) Twitter, (ii) Linkedin, (iii) Reddit, (iv) Facebook, (v) StumbleUpon, (vi) Delicious, (vii) Pinterest, and (viii) Google Plus.

The service is called in an iterative manner, and for each URI (associated to a news article) passed as input it performs in a multi-threaded manner HTTP calls to the specified social media platforms, receives the response of the calls, and returns the sociometric scores associated to the input URI as a JSON object. The service also retains a collection timestamp (generated once the collector is first instantiated), and a timestamp for each HTTP call made to the social media API’s.

- **Input**: A URI for a news article
- **Output**: A JSON object containing the sociometric counts assigned by the specified social media platforms

For example, given a news article with the following ID and URI:

id = “12345"; url = "http://www.linkedin.com";

the JSON output would be as follows:

{"twitter":[{"tweet_count":136789}],"linkedIn":[{"share_count":152542}],"_crawl_cycle_timestamp":1393607209433,"_id":"12345","_url":"http://www.linkedin.com","reddit":[{"comments_count":0,"downs_count":0,"ups_count":1,"score_count":1}],"facebook":[{"comment_count":1968,"like_count":4065,"share_count":17263}],"_url_timestamp":1393607209634,"stumbleUpon":[{"views_count":7089}],"delicious":[{"bookmarks_count":11944}],"pinterest":[{"shares_count":37}],"googlePlus":[{"shares_count":316424}]}

2.3.2 **Repositories**

The First Prototype provides four different repositories:

- The CNR to store all the crawled documents
- The CMR to store the multimedia elements related to the crawled documents
- The OPS to store the Operations information
- The RDF to store all the knowledge base produced by the offline modality and some relevant Linked Open Data datasets.

The CNR and OPS repositories were already available in the Operational Prototype. The update of these repositories is presented in the following sections (see 2.3.2.1 and 2.3.2.3).

The CMR and the RDF repositories have been implemented for the First Prototype. A basic description is provided with functional and technical details in the next sections.

In the First Prototype, the four repositories are implemented, fully functional and accessible through the local access in the server or the REST service endpoint. For each access method, other services are able to store, update or collect data.

2.3.2.1 **Central News Repository (CNR)**

The CNR stores all harvested news. The crawler stores news items (news articles, social media posts, etc.) as they become available, along with metadata (source, date, country, etc.). Unprocessed news can then be pulled by the analytic pipelines for processing.

The Central News Repository (see D7.2, section 4.2.4.1) is the raw storage dump for the Crawlers (Site and Media collectors). The CNR is an Elastic Search instance. This instance is used as a document pool to deliver the original documents in the Content Analysis Pipeline to be processed. Then, the CNR items are analysed by the CEP and the extracted/produced knowledge is stored in the RDF repository.

For each crawler a JSON API is provided with the structure of the document to be stored in the CNR (ElasticSearch). The news items (news articles, social media posts, etc.) are collected from the crawlers by calling the corresponding JSON API and become available in the CNR along with metadata (source, date, country, etc.). Then, the unprocessed news can be pulled by the analytic pipelines for processing.

Here is the list of the collected fields used for each crawler:

Name of the field	**Description of the field**	**Used by the Site collector**	**Used by the Media collector**
use_case	Name of the use case that the document refers to	X	X
Language	Language of the article	X	X

Country	Country from where comes from the article	X	X
_id	Identifier of the article in CNR	X	
Crawled	Date when the article was crawled.	X	X
multimediaUrls	List of the multimedia elements (url) available in the article	X	X
c_sourcecode	Content of the article in HTML format	X	X
Source	Media from where comes the articles	X	X
pr_summary	Summary provided with the article	X	
date_timestamp	Date of the article publication in the timestamp format	X	X
Body	Text of the articles	X	X
Feed	Name of the feed used to collect the article	X	
date	Date of the article publication	X	
article_id	Identifier of the article generated by the PR API	X	
pr_feed	Type of the feed used to collect the article (internet, press, files)	X	
title	Title of the articles	X	X

Table 1: List of the collected fields from the crawlers

2.3.2.2 **CMR Repository**

The Central Media Repository (CMR) is the storage of the source multimedia content (video, images and audio) collected by the harvester (see D7.2, section 4.2.4.2).

The CMR is built as a simple filesystem. Within the logic of the Crawler, a method is defined to operate asynchronous iterations over the “multimediaUrls” field. Inside this field there is a list of URLs that contain the related images, videos and audio files for a specific article retrieved by the Crawler. Because of this, customised folders are created for every article and save the multimedia content in that directory. The final goal of the CMR is to offer all that multimedia data to other services that could make used of it.

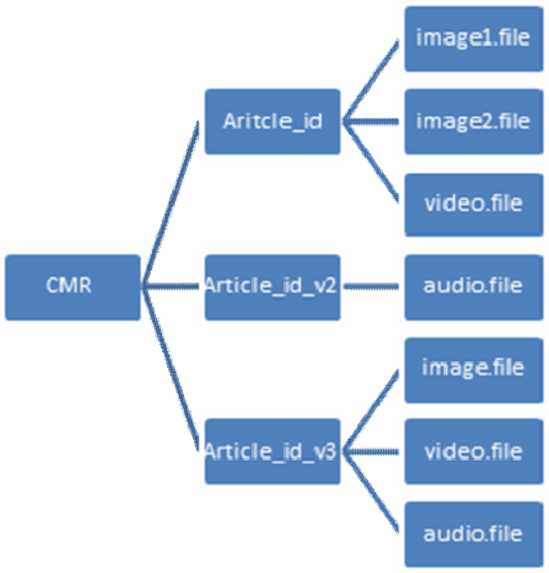


Figure 6: Folder structure to store the multimedia content in the CMR

At the end, the multimedia items are pulled and process by the analytic pipelines for processing. Audio files are extracted from the video and text is extracted from the audio file.

2.3.2.3 **OPS Repository**

The Operations Repository (OPS) provides fast, read/write structured data storage for any systems in the platform that require it (see D7.2, section 4.2.4.4).

The OPS is implemented as an instance of MongoDB and is used and different cases.

- It permits to store the information about the infrastructure, i.e. the list of the document to pull in the pipeline.
- Finally, the information related to the UC applications is also stored in OPS. The different credentials to login the web application, the user session and profile are stored in MongoDB as well.

2.3.2.4 **RDF Repository**

The RDF repository (RDF) holds all data in semantic format for the platform (see D7.2, section 4.2.4.3). This includes the ontologies and datasets used to model the data, as well as semantic annotations on the harvested SIMMOs. Also, many Linked Data datasets were loaded in GraphDB. This knowledge is used to search specific indicators and to reason on facts to enrich the search results and support the user decisions.

Ontotext provides GraphDB-Enterprise as a semantic data infrastructure layer for the purposes of MULTISENSOR. It is a high performance system implemented in Java, which support storing, querying and processing structured data formatted according to the RDF standards and is packaged as Storage and Inference Layer (SAIL) for the Sesame framework. It is based on the Ontotext's TRREE – native RDF rule-entailment engine.

This system is deployed and fully available on http://multisensor.ontotext.com.

The RDF repository is searchable via SPARQL or a textual mode. It will be populated by the different analytical pipelines:

- From the CEP: entities, context and linguistic knowledge
- From the CAP: topics and domain are aligned

- From the SMAP: the social metrics are formalised and related to the different sources of information
- From the Linked Open Data datasets: DBpedia[1], World Bank[2] and EuroStat[3] datasets have been uploaded in GraphDB.

Then, the knowledge available in the RDF repository is used for different purposes:

- The knowledge is collected, used or updated by some of the services of the offline modality to get knowledge about specific entities or semantic resources.
- The knowledge is used as an index for the search functionalities. Integrated with the ElasticSearch connector, GraphDB is used as a semantic search engine that can acts in a textual mode or in SPARQL mode.
- Finally, the Data wrappers are built on top of the RDF repository. Many resources identified in the Linked Open Data initiatives as relevant knowledge for the project has been selected and stored in the GraphDB. Then, some of the Online services can search and get the semantic resources to reason or display it directly in the UC applications.

2.4 Distillation layer

The Distillation Layer comprises three sets of processes which analyse the harvested information and generate actionable knowledge (INTEL related to the SIMMO) from it (see D7.2, section 4.1.2).

Each pipeline is used to analyse the harvested information and generate actionable knowledge. The three have different goals and are complementary:

- The Content Extraction Pipeline (CEP) to analyse the news, extract the relevant knowledge to create the index.
- The Content Alignment Pipeline (CAP) to analyses the existing data and performs reasoning inconsistencies and contradictory facts.
- The Social Media Analysis Pipeline (SMAP) to analyse the social network data mainly coming from Twitter and Facebook.

2.4.1 Content Extraction Pipeline (CEP)

The Content Extraction Pipeline analyses all the harvested news that are stored in the CNR.

The CEP chains execution of several language analysis services to generate intelligence from the contents of a text article or a video segment, from base syntactical analysis to sentiment analysis to automated clustering and classification of the contents. The multimedia content is separated in four modalities: text, image, video, and audio. The most important one is the textual. For the audio and video, the textual information is extracted and then the multimedia resources are described as a textual document.

All the documents collected by the crawlers are stored in the CNR as a specific JSON structure. According to the JSON API provided by the Site collector (PR crawler) and the

[1] DBpedia: http://wiki.dbpedia.org/Datasets

[2] http://worldbank.270a.info/.html

[3] http://eurostat.linked-statistics.org/

Media collector (Yahoo crawler), news are pulled in the CEP. Then, the different fields are analysed by the different services of the pipeline to extend and add the extracted knowledge in a more complex structure called WorkBook (see D7.2, section 4.2.2.3).

The WorkBook is also a JSON structure that is enriched by every service that receives it and sends it to the next service. The WorkBook is a JSON container that is composed of several fields, and one of those fields is the RDF content (available in the JSON-LD[4] format). The RDF content contains the different recognised entities and the linguistic description (NIF[5] representation) for each sentence.

Implementing the CEP, some specific needs appeared and implied to modify the structure of the pipeline.

- UPF Content extraction was separated in two different services (Concept extraction and the Relation extraction)
- The indexing service should be separated to take into account the JSON content of the SIMMO and the RDF content. Each part should be stored in different repositories.

Here is the new structure of the CEP:

[4] http://json-ld.org/

[5] http://persistence.uni-leipzig.org/nlp2rdf/

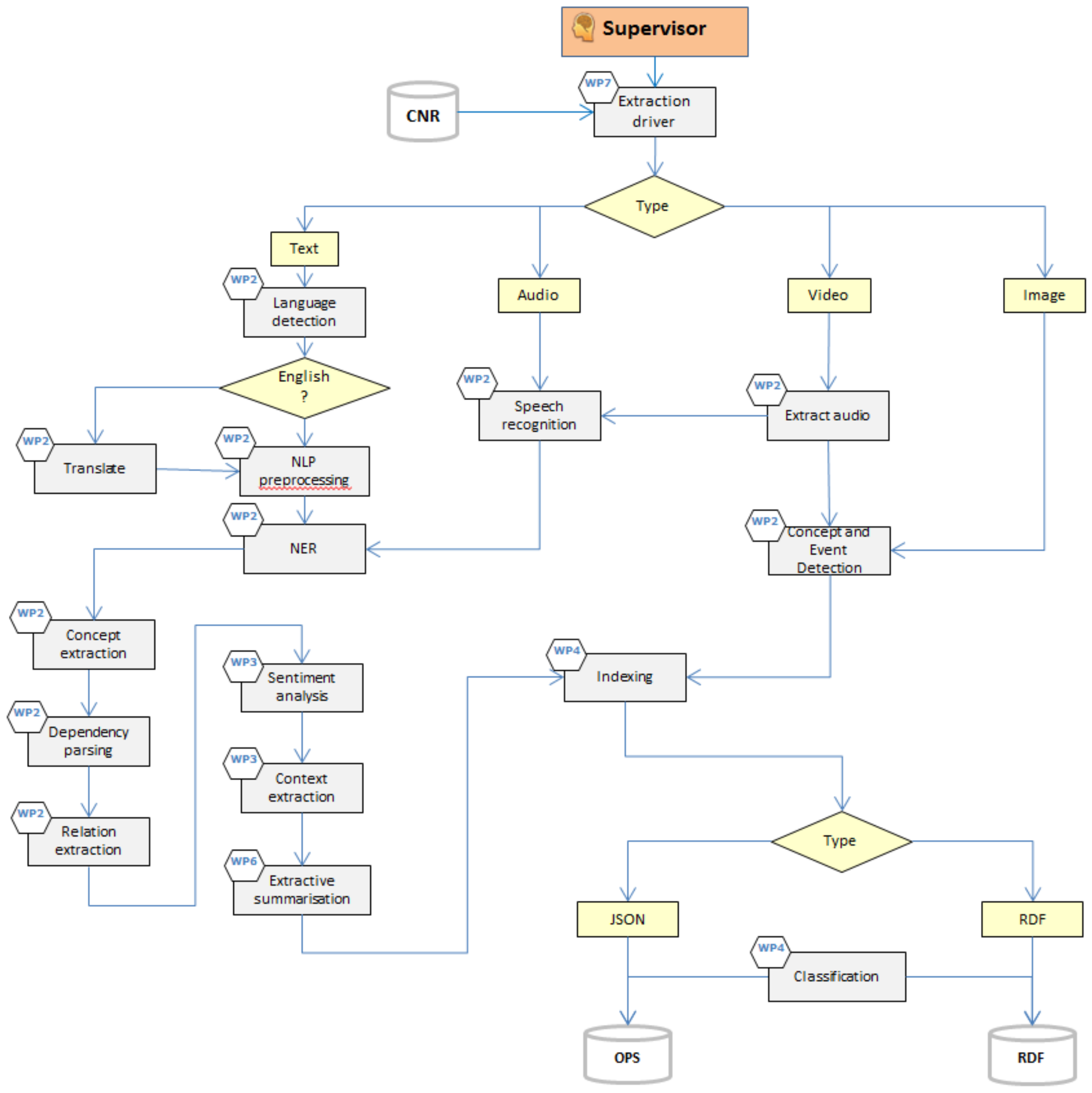


Figure 7: Content Extraction Pipeline of the First Prototype

2.4.2 **Content Alignment Pipeline (CAP)**

The Content Alignment Pipeline (CAP) includes the Content Alignment Service (see D7.1, section 3.3.4 and see D7.2, section 4.2.2.5). The Service takes as input a document id and returns a list of article ids that share similarities to the input document. It makes use of the RDF knowledge base content in order to identify similarities between content items. The CAP is user-initiated. The baseline version of the service implements a simple SPARQL query-based method.

The current version of CAP implements a baseline methodology where relevant documents are retrieved using a SPARQL queries to the Knowledge Base (RDF). A more advanced version is being implemented where it will identify semantic similarity between articles using content and structure (graph-based) similarity metrics.

2.4.3 Social Media Analysis (SMAP)

The Social Media Analysis pipeline (SMAP) is a set of processes related to analysis of social network data stored in the MULTISENSOR repositories. It is executed periodically in the background by the Supervisor.

The SMAP pipeline analyses mainly Twitter and Facebook data to detect trends, map network graphs of contributors, and track topics and controversies over time (see D7.2, section 4.2.2.4). It is not provided in the First Prototype.

2.5 Delivery layer

The Delivery Layer implements interfaces for users to interact with the platform. It is responsible for hosting applications and services that build value from the information in MULTISENSOR to implement the Use Cases.

2.5.1 Overview of the First Prototype applications

2.5.1.1 Online modality

In the Operational Prototype, the Online Services were only dummy services that presented some fake content directly in the user interface.

In the First Prototype, most of the Online Services are deployed as a baseline service and integrated in the UC application.

The Online Services are connected to the Harvesting repositories: the OPS repository that contains the JSON content of the SIMMO object and the RDF repository that contains the RDF content produced from the CEP and some Linked Open Data datasets.

As the RDF repository is not yet fully populated, the search functionality still remains connected with the CNR.

More details are provided in section 3 of the present document.

2.5.1.2 First Prototype applications

The First prototype provides three different UC applications: UC1 and UC3 applications were already presented in the deliverable D7.3 (see D7.3, section 3). The UC2 is a new UC application developed by PR.

The three UC applications integrate the online services that are deployed and connected with the different repositories (See the section 4 of the present document).

3 INTEGRATION FRAMEWORK

3.1 Approach

For the First Prototype, all the services and the repositories are deployed as a baseline service. The integration phase raised several technical issues related to the services compilation and execution in a common environment, and also to the dependencies between the services (especially for the CEP) and with other libraries.

All the services are uploaded in the SVN system. Then, everis tests all the services compilation and the execution of the tests in the local environment until being successful. After, the services are built as a war project and are deployed in the Grinder server supported by Jetty.

Finally, the war project for every service is uploaded in the server and deployed as a REST service. When the real services are deployed, they can be executed by the supervisor in the case of the offline modality or directly by the UC applications in the case of the online modality.

For the next iteration, the integration tasks will be automated thanks to a continuous integration system called Jenkins[6]. This tool permits to test automatically the compilation and the execution of the services, and enable their deployment only when the tests are successful.

3.2 Development status of the Offline modality

3.2.1 Content extraction pipeline (CEP)

For the First Prototype, a lot of efforts were provided to integrate the services of the Content Extraction Pipeline (CEP).

Due to the diversity of dependencies and the quantity of services to integrate, many technical issues were raised along the integration process. Also, a lot of integration iterations were applied to ensure the compliancy of the services output as a valid input for the next service. Each service consumes the output of the previous service, and thanks to the MSWorkBook class, the different fields of the JSON container are interpreted and can be extended properly with the analytic data generated by the current service.

In the First Prototype, the CEP is fully functional, this means all the services are integrated in the pipeline, each one extract and generate new knowledge that is added in the WorkBook. Now, before storing this knowledge in GraphDB and to exploit it efficiently, the RDF content should be validated according to the Linked Data recommendations. This work requires many iterations and test with GraphDB to ensure the CEP is producing the best index possible and the retrieval functionalities will be optimised. Only after this important phase, the GraphDB repository will be populated with all the knowledge extracted from all the CNR items.

In the following subsections, the integration description for all the services is provided.

[6] https://jenkins-ci.org/

3.2.1.1 Language detection

The language detection service recognises language of a text.

Service name	Language Identifier
SVN artefact / remote endpoint	http://services.linguatec.org/rest/lang/identify
Functional Description	Recognizes language of a text
Development status	Version 1.0, fully functional
Deployment status	Remotely
Integration status	Integrated
Integration issues / dependencies	None
Next steps	None

Table 2: Integration description of the Language identifier service

3.2.1.2 Translation

The Translation service receives as input text in source language and translates it into specified target language.

Service name	Machine Translation
SVN artefact	http://services.linguatec.org/rest/lang/translate
Functional Description	Receives as input text in source language and translates it into specified target language
Development status	Version 0.8, fully functional
Deployment status	Remotely deployed
Integration status	Integrated
Integration issues / dependencies	Baseline systems for English into French, Spanish, German
Next steps	Bulgarian translation will be added and language directions will be extended.

Table 3: Integration description of the Translation service

3.2.1.3 Named Entities recognition

Named Entities recognition is a service that recognises the names of persons, the locations, and the organisations & companies, as well as date, amount and measurements.

Service name	Named Entity Recognition
SVN artefact	http://services.linguatec.org/rest/ner/recognize
Development status	Version 0.8, fully functional

Deployment status	Remotely deployed
Integration status	Integrated,
Integration issues / dependencies	System for English
Next steps	Extension to other languages

Table 4: Integration description of the Name Entities Recognition service

3.2.1.4 Concept extraction

Concept extraction extends Named Entities Recognition with the detection and annotation of mentions to abstract concepts and other entities that are relevant to the domain in hand. A detection strategy is applied where corpora of texts belonging to each UC are analysed using statistical methods in order to obtain lists of potential terms used to refer to relevant concepts. When processing texts, the concept extraction service uses this list to detect potential terms and then attempts to link them to entries in domain-specific ontologies and datasets, disambiguating between multiple senses if necessary.

Service name	**Concept extraction service**
SVN artefact	ms-svc-extr
Development status	Version 0.2, baseline version
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	The current service implementation uses a third party library and database dump, BabelNet 2.5.1[7]. The database files, due to their large size, have not been uploaded to the SVN server and must therefore be deployed manually on the prototype server.
Next steps	The NIF/OLiA RDF annotations produced by this service must be validated by Ontotext. Concept extraction lacks term detection and disambiguation components. The first will be implemented using the third party TermRaider[8] GATE plugin. UPF is working on a graph-based algorithm for disambiguation candidate terms against BabelNet and other dataset senses.

Table 5: Integration description of the Concept extraction service

3.2.1.5 Dependency parsing

The dependency parsing service annotates texts with the syntactic structure of their sentences. Two different structures are annotated, (i) a surface syntactic structure indicating language-specific grammatical relations between all words in a sentence, and a (ii) deep syntactic structure with language-independent predicate-argument relations between content words. Both structures follow the dependency syntax formalism. While the surface

[7] http://babelnet.org/2.5.1/about

[8] https://gate.ac.uk/projects/arcomem/TermRaider.html

structure is equivalent to the output of existing dependency parsers, the deep structure is more "semantic" and hence more suitable for concept extraction.

Service name	**Dependency parsing service**
SVN artefact	ms-svc-dep
Development status	Version 1.0, fully functional (for English language).
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	The dependency parsing service depends on the annotations produced by the named entity recognition and concept extraction services. The surface parser uses language-specific models stored in files. These files, due to their large size, have not been uploaded to the svn server and must therefore be deployed manually on the prototype server.
Next steps	The NIF/OLiA RDF annotations produced by this service must be validated by Ontotext. The service will be extended in order to support additional languages.

Table 6: Integration description of the Dependency Parsing service

3.2.1.6 **Relation extraction**

The relation extraction service focuses on identifying and annotating relations between named entities and concepts annotated in the text. Two kind of relations are annotated, coreference relations and semantic relations indicated by linguistic predicates. Coreference relations link text fragments which denote the same entity or concept. Semantic relations link two or more entities or concepts involved in a common situation such as an event. Semantic relations are classified according to existing repositories of predicative senses (e.g. VerbNet, FrameNet).

Service name	**Relation extraction service**
SVN artefact	ms-svc-rel
Development status	Version 0.2, baseline version
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	The relation extraction service depends on the annotations produced by the named entity recognition, concept extraction and dependency parsing services. The service uses two third-party libraries: the Stanford CoreNLP library[9] for detection of coreference relations, and the Semafor[10]

[9] http://nlp.stanford.edu/software/corenlp.shtml

	disambiguation tool for detection on semantic relations and annotations with FrameNet senses. Both tools use large model sizes which have not been uploaded to the SVN server and must therefore be deployed manually on the prototype server.
Next steps	A deterministic tool for mapping deep dependency structures to semantic structures with VerbNet and FrameNet senses is being developed and will replace the Semafor third-party library.

Table 7: Integration description of the Relation extraction service

3.2.1.7 **Sentiment analysis**

The sentiment analysis service is responsible for detecting sentiments in English text. For a given piece of short text, the service generates a positive sentiment score [+1, +5] and a negative sentiment score [-1, -5] for each short text or sentence in the news article. From the positive and negative sentiment scores, it then computes the sentimentality of the text, as follows:

$$|score_{pos}| + |score_{neg}| - 2 = score_{sent}$$

and the polarity of the text as shown below:

$$score_{pos} + score_{neg} = score_{pol}$$

The sentiment analysis service is currently implementing the baseline SentiStrength tool. SentiStrength is a lexicon-based classifier that uses additional (non-lexical) linguistic information and rules to detect sentiment strength in short informal English text. The service accepts as input the textual content of a media item, from which it extracts the sentences and stores them in an object that implements the Interface <Collection>. It then iterates through the collection of sentences and computes the sentiment scores *pos*, *neg*, *sent*, and *pol* for each sentence separately.

The core of the SentiStrength algorithm is the sentiment word strength list. This is a collection of 298 positive terms and 465 negative terms classified as having either positive or negative sentiment strength with a value from 2 to 5. In addition, SentiStrength contains several key features, such as:

- A spelling correction algorithm
- A booster word list
- An idiom list
- A negating word list
- An emoticon list with polarities
- A functionality for handling repeated letters
- A functionality for handling repeated punctuation
- A training algorithm that optimises sentiment word strengths and potentially also changes polarity

The output of the service is the original list of sentences, as taken from the textual part of the media item, coupled with their sentiment scores. The output is in JSON format.

[10] http://www.ark.cs.cmu.edu/SEMAFOR/

The next steps involve the implementation of the first version of the module for sentiment analysis using the annotated in-domain, news corpus, which is built for evaluation and training purposes. In addition, the performance of the sentiment analysis module will be benchmarked against the baseline SentiStrength and evaluated using standard performance metrics like Precision, Recall, Accuracy, and F1-score.

Service name	**Sentiment Analysis**
SVN artefact	ms-svc-sa
Development status	Version 0.2, baseline version
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	Requires a NIF wrapper (NIFWrapper) class to access the RDF part of the article object.
Next steps	Implementation of the first version of the module for sentiment analysis.

Table 8: Integration description of the Sentiment analysis service

3.2.1.8 Context extraction

Given a media item, the context extraction service extracts or collects from the output of other services the following contextual features:

- Author: an entity responsible for the creation of the item content
- Source: an entity responsible for making the item available
- Title: a name given to the media item
- Keywords: a set of phrases describing the topic of the item
- Genre: the style or type of the item
- Category: a classification of the item according to its content
- Date: a date associated with the creation or availability of the item
- Location: a location indicating where the item was created
- Literary style: a metric of language formality
- Language: the language of the content of item

The context extraction service requires as input the textual content as well as the metadata stored in the html source of the media item. It also exploits the output of other services preceding its execution in the pipeline such as the language detection module for retrieving language information.

In its current status (version 1.0) it extracts either from the text or from the metadata the *author* or creator of the content item, a set of *keywords* characterizing the content item, the *literary style or formality level* of the content of the item and the *genre* of the item if found in the metadata. Features such as *date*, *location*, and *source* are provided by the crawler service. Specifically, date refers to the crawling date, location refers to the country associated with the media source from which the media item was crawled and source is the respective source defined in the seed list used for crawling. Similarly, *title* is also provided by the crawler and corresponds to the title associated with the out-link that pointed to the media item considered. Language is defined by the respective module and finally category will eventually be provided by the topic classification module.

With respect to the output produced by the context extraction service, we use the Dublin Core ontology that contains a number of metadata elements for classifying documents and other electronic resources. The vocabulary of Dublin Core can be used to express contextual features as follows: author is mapped to dc:creator, source is mapped to dc:source, title is mapped to dc:title, keywords are mapped to dc:subject, language is mapped to dc:language, genre is mapped to dc:type, and finally date is mapped to dc:date. Finally, additional features not supported directly by the schema, such as literary style, will be added by extending the schema.

The next steps concern an improvement of the extraction methods after checking their performance (in terms of precision and recall) on the set of media items stored in the news repository.

Service name	**Context extraction**
SVN artefact	ms-svc-context
Development status	Version 1.0, fully functional extracting a subset of the contextual features.
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	The output has to be validated.
Next steps	Improvement of performance in terms of precision (i.e., how accurate are the contextual features?) and in terms of recall (i.e., what is the coverage of the contextual features?)

Table 9: Integration description of the Context extraction service

3.2.1.9 **Extractive summary**

Extractive summarisation refers to the generation of summaries from texts by selecting sentences from the original summaries and composing a summary from these (unchanged) sentences. The language of the summary is the same as that of the original documents.

Service name	**Extractive summary**
SVN artefact	ms-svc-summ
Development status	Version 0.2, baseline version
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	The extractive summarisation functionality of this service uses a third party library, the GATE-based SUMMA framework[11].

[11] http://www.taln.upf.edu/pages/summa.upf/

Next steps	A first version of the extractive summarisation functionality has been developed and will be integrated and deployed in the following weeks.

Table 10: Integration description of the Extractive summary service

3.2.1.10 Indexing

In the indexing service, a multimedia data representation framework that allows for the efficient storage and retrieval of SIMMO objects (see 2.4.3) is developed. The service contains an indexing structure for holding and retrieving efficiently the multimodal entities of the multimedia information.

Service name	**Indexing**
SVN artefact	wp4/ms-svc-multimediaStructure
Development status	Version 1.0, fully functional
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	No integration issues Dependencies: MongoDB, Morphia framework
Next steps	If needed, a new version of the service will be implemented, capable of indexing more advanced output from the CEP. This will be implemented in T4.4.

Table 11: Integration description of the Indexing service

3.2.1.11 Classification

The category classification service receives as input the text body of a News Item that is retrieved from the CNR, extracts a set of textual features and utilizes a Random Forest classifier, in order to provide as output the category, to which the News Item belongs.

Service name	**Category classification**
SVN artefact	wp4/ms-svc-categoryClassification
Development status	Version 0.2, baseline version
Deployment status	Remotely deployed
Integration status	Integrated
Integration issues / dependencies	No integration issues Dependencies: R (statistical programming language) needs to be installed in the computer where the service runs
Next steps	Implementation of a fully functional version of the service that will make use of the multimodal features that are produced in WP2 (textual and visual concepts, name entities) and WP3 (sentiment, polarity and contextual information) and are stored in the indexing structure of the OPS repository. This version will be implemented in T4.1.

Table 12: Integration description of the Category classification service

3.2.1.12 Storing RDF

The RDF Storing service is designed to handle input in the form of SIMMO JSON objects, to parse it and to store it in the knowledge base in different context. It supports two different request methods: PUT and POST

The client should send the document ID as URL parameter. The body content of the request, which holds the SIMMO object should be named -"SIMMO". The SIMMO object should be in a valid JSON file. Otherwise the parsing procedure will fail and an error message will be returned to the client.

The parsing service store each SIMMO object in different context identified by the document id, so the retrieving of the documents at this moment is possible by SPARQL construct query and the id of the searched document. We can use the capabilities of the GraphDB Workbench and retrieve the document in all supported formats – JSON, XML, TSV, CSV.

Service name	**Storing RDF**
SVN artefact	http://multisensor.ontotext.com/cep
Development status	Version 1.0, fully functional
Deployment status	Remotely deployed
Integration status	Integrated
Integration issues / dependencies	No integration issues
Next steps	n/a

Table 13: Integration description of the Storing RDF service

3.2.1.13 Concept and Event detection

The concept and event detection service receives as input a multimedia file (i.e. image or video) and computes degrees of confidence for a predefined set of concepts. To this end, the service incorporates various procedures, such as video decoding (applicable for video files only), feature extraction and supervised classification. The video decoding procedure is responsible for extracting a predefined number of frames from a video file. The feature extraction step refers to the extraction of descriptors that describe visually images by capturing either global or local information out of the images. Finally, the classification step refers to the development of models used for classifying images or video frames to the set of predefined concepts/ categories.

Service name	**Concept and Event detection**
SVN artefact	wp2/ms-svc-conceptEventDetection
Development status	Version 1.0, fully functional
Deployment status	Remotely deployed
Integration status	Integrated
Integration issues / dependencies	No integration issues

	Dependencies: OpenCV and vlfeat libaries, ffmpeg, ffprobe
Next steps	Implementation of a version that will contain the advanced module of the service. The advanced module will be implemented in T2.6 and will offer object and event detection functionalities.

Table 14: Integration description of the Concept and Event detection service

3.2.1.14 **Speech recognition**

The Speech recognition service (see D7.1, section 3.1.4) extracts audio from the videos and recognizes text from audio files. Then, the human speeches are converted into texts that can be processed by the textual dimension of the CEP.

Service name	**Speech recognition**
SVN artefact	http://voicepro.linguatec.org/rest/documents/2075/
Development status	Version 0.8, fully functional
Deployment status	Remotely deployed
Integration status	Integrated
Integration issues / dependencies	For English
Next steps	Extension to German

Table 15: Integration description of the Speech recognition service

3.2.2 **Content Alignment Pipeline (CAP)**

The Content Alignment Pipeline includes the Content Alignment service which receives as input a document id and returns a list of document ids of documents with similar content to the input document. It uses the RDF knowledge repository to extract the documents' content and identify similarities between content. The CAP is initiated by the user when he/she selects a document from the user interface. The current service implements a baseline version which makes use of a SPARQL query-based method for retrieving similar content.

Service name	**Content Alignment**
SVN artefact	wp4/ms-svc-contentAlignment
Development status	Version 0.2, baseline version
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	No integration issues No dependencies from external APIs
Next steps	Implementation of a more sophisticated content alignment method which will replace the current SPARQL query-based method that will evaluate semantic similarity between documents using content and structure (graph-based) similarity metrics.

Table 16: Integration description of the Content alignment service

3.2.3 Social Media Analysis Pipeline (SMAP)

As described before, the Social Media Analysis Pipeline (SMAP) is not part of the First Prototype. Due to some problems to access to the Twitter data, the implementation is still in progress and will be integrated in the Second Prototype.

3.3 Development status of the Online modality

3.3.1 Business Shared Services

3.3.1.1 Content delivery

The Content delivery serves to get the information related to the predefined articles. It permits to display the different elements of the articles that are stored in the RDF repository (GraphDB). The different elements that are retrieved by the content delivery service are the body, the media, the summary, the list of the entities, the sentiment values, the translation and the content.

Service name	**Content delivery**
SVN artefact	ms-svc-delivery
Development status	Version 0.5, baseline
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	None
Next steps	As the knowledge base is not yet populated, the different entities are not available. The content delivery will be extended when the whole RDF content from the CEP will be stored in GraphDB.

Table 17: Integration description of the Content delivery service

3.3.1.2 Semantic search

The Semantic search service is integrated to the system and is implemented in two different scenarios:

- Implemented with the CNR ElasticSearch
- With the knowledge repository (RDF repository connected to a new ElasticSearch instance)

In the second prototype we aim at having a stable version of the ElasticSearch connector with GraphDB. When the new instance will be ready and the RDF data produced by CEP will be validated, the system will switch the search engine from the CNR to the RDF repository.

Service name	**Semantic search**
SVN artefact	CNR repository (ElasticSearch)
Development status	Version 0.7, temporary search engine

Deployment status	Remotely deployed
Integration status	Integrated
Integration issues / dependencies	No integration issues Dependencies:
Next steps	When the whole RDF content will be stored in GraphDB, the CNR engine will be replaced by the ElasticSearch connector integrated with GraphDB. Then, instead of searching in the crawled documents, the user will be able to query the knowledge base in textual mode.

Table 18: Integration description of the Semantic search service

3.3.1.3 **Clustering / Filtering**

The topic-event detection service receives as input the text body of a list of News Items retrieved from the CNR, extracts a set of textual features, makes use of a modified version of the DBSCAN clustering algorithm and provides as output a grouping of the list based on the existence or not of an unknown number of topics / events. Irrelevant News Items are marked as noise and they are not assigned to any of the discovered topics/events.

Service name	**Topic detection**
SVN artefact	wp4/ms-svc-topicDetection
Development status	Version 0.2, baseline version
Deployment status	Remotely deployed
Integration status	Integrated
Integration issues / dependencies	No integration issues Dependencies: R (statistical programming language) needs to be installed in the computer where the service runs
Next steps	Implementation of a fully functional version of the service that will make use of the multimodal features that are produced in WP2 (textual and visual concepts, name entities) and WP3 (sentiment, polarity and contextual information) and are stored in the indexing structure of the OPS repository. This version will be implemented in T4.1.

Table 19: Integration description of the Topic detection service

3.3.1.4 **Similarity search**

The similarity search service receives as input the id (Elastic Search id) of an article and outputs a list of 250 randomly selected article ids (dummy version).

Service name	**Similarity search**
SVN artefact	wp4/ms-svc-similaritySearch
Development status	Version 0.1, dummy
Deployment status	Deployed

Integration status	Integrated
Integration issues / dependencies	No integration issues No dependencies
Next steps	Implementation of a fully functional version of the service that will output a list of the most similar articles (ids) to the input article (id). To this end, the training and integration of a suitable ranking / retrieval algorithm (which will be implemented in T4.4) is required.

Table 20: Integration description of the Similarity search service

3.3.1.5 Machine Translation

The Machine translation is a service that translates a text in a specific language into another language. It receives as input the text, the source language and the target language. Then it returns as output the translated text.

Service name	**Machine Translation**
SVN artefact	http://services.linguatec.org/rest/lang/translate
Functional Description	Receives as input text in source language and translates it into specified target language
Development status	Version 0.8, fully functional
Deployment status	Remotely deployed
Integration status	Integrated
Integration issues / dependencies	Baseline systems for English into French, Spanish, German
Next steps	Bulgarian translation will be added and language directions will be extended.

Table 21: Integration description of the Machine translation service

3.3.1.6 Abstractive summary

Abstractive summarisation refers to the generation of multilingual summaries from data using natural language generation methods. The abstractive summarisation service starts from a user query and an RDF graph generated in response to the same query by the decision support service, and it returns a textual summary in the requested language that verbalises in a coherent and grammatically correct way the most relevant contents in the graph.

Service name	**Abstractive summarisation service**
SVN artefact	ms-svc-summ
Development status	Version 0.1, dummy
Deployment status	Not deployed
Integration status	Not integrated
Integration issues /	The abstractive summarisation functionality of this service

dependencies	depends on having a functional decision support service.
Next steps	A first version of the abstractive summarisation will be deployed and integrated into the second prototype.

Table 22: Integration description of the Abstractive summarisation service

3.3.1.7 **Contributor analysis**

The contributor analysis module receives as input a twitter handle (e.g., "@barackobama") and given this handle it extracts a list of descriptors capturing its influence. Specifically, the service allows a user with a legitimate Twitter application and user authentication keys to crawl the profile of particular users and compute basic statistics on network and retweeting influence.

Instead of giving as input a specific twitter handle, the service can work alternatively given a specific search key as input, e.g., "Barack Obama". Given this search key, the service retrieves the top 10 relevant Twitter accounts with this string and proceeds as before with each of them.

The service has been fully developed and deployed. Since this an online service operating given a specific user input (i.e., a specific Twitter handle or a search key for retrieving relevant twitter handles), the service is integrated but with specific limitations. Depending of the Twitter API, uses of this services is allowed to check 3 accounts per hour.

Service name	**Contributor analysis**
SVN artefact	ms-svc-contributorAnalysis
Development status	Version 1.0, fully functional
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	This service is independent of the rest of the pipeline, however it requires input from the end user or another relevant service if applicable.
Next steps	n/a

Table 23: Integration description of the Contributor analysis service

3.3.2 **Other Online Services**

3.3.2.1 **User profile**

The User Profile is the service that allows the system to manage the user accounts with credentials for the system registration, and the creation of the specific profile to the different queries done by the end-user.

Registration, login, logout, and profile updates are enabled by this service. It is implemented as a generic service that is used by the 3 UC applications. All the profile data are stored in the OPS repository (MongoDB).

The API for the service is defined as followed:

PATH	METHOD	ACTIONS	PARAMETERS	RESPONSE

/login	GET POST	Retrieves login view where to login and authenticate (Google). Validation of inputs against DB. Redirects options should be decided internally for UC2 (PR).	Request, email, password, callback	JSON type (user information)
/register	POST	Insert user's details on database. MongoDB's methods to save and insert new users and their fields. For UC2 decide redirect view (home for example).	Request params such as username, email and password. (object_id automatic generated when added).	JSON type (user information)
/home	GET POST DELETE	Home page view rendered. Able to save searches as an object getting name and filtering fields, adding them to the DB. Visually displaying list of favorites. Possible to manage (update/delete)saved ones.	Requested params: profile_name keywords country language media_source	JSON type (profile data)

Table 24: API of the Profile service

The Profile service model is implemented with the npm module of node.js called passport[12] and further dependencies to safely define local login strategies as well as Google account login approach.

Service name	**User Profile**
SVN artefact	ms-svc-profile
Development status	Version 1.0, fully functional
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	No integration issues. Based on the npm module of node.js called "passport".
Next steps	The SPARQL queries will be created to collect all the indicators defined in the document D3.2.

Table 25: Integration description of the User Profile service

3.3.2.2 **Reference Data**

Specific to the Use Case 3, the Reference Data service permits to collect information about different indicators such as economic, political, or encyclopaedic ones. Different datasets of Linked Open Data have been uploaded in the RDF repository (GraphDB) to provide such a

[12] https://www.npmjs.com/package/passport

kind of knowledge. For example, the DBpedia, the World Bank and the EuroStat datasets have been loaded in GraphDB and are available in the platform through the SPARQL endpoint (http://multisensor.ontotext.com/sparql). Then, the Reference Data service can query the knowledge base and get the specific facts related to country and the product the end user is looking for.

Some SPARQL query template have been created to collect the relevant indicators. Just after, an example of the SPARQL query template is presented. This is used to get the GDP indicators related to the predefined country and during the predefined period:

SPARQL Query Template to get the GDP growth indicator

```
PREFIX country: <http://worldbank.270a.info/classification/country/>
PREFIX indicator: <http://worldbank.270a.info/classification/indicator/>
PREFIX property: <http://worldbank.270a.info/property/>
PREFIX qb: <http://purl.org/linked-data/cube#>
PREFIX sdmx-dimension: <http://purl.org/linked-data/sdmx/2009/dimension#>
PREFIX sdmx-measure: <http://purl.org/linked-data/sdmx/2009/measure#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT ?observation ?countryURI ?refPeriodURI ?obsValue ?refPeriod {
        ?observation a qb:Observation ;
                property:indicator indicator:SL.UEM.TOTL.ZS ;
                sdmx-dimension:refArea ?countryURI ;
                sdmx-dimension:refPeriod ?refPeriodURI ;
                sdmx-measure:obsValue ?obsValue.

        BIND(SUBSTR(STR(?refPeriodURI), 38, 4) AS ?refPeriod)
        BIND(xsd:integer(?refPeriod) as ?d)
        FILTER (?d > 2005 && ?d <= 2012)
        FILTER (?countryURI = country:ES)
}
```

Table 26: SPARQL Query Template to get the GDP growth indicator

Service name	**Reference data**
SVN artefact	ms-svc-refdata
Development status	Version 0.5, baseline version
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	No integration issues The Reference data is fully dependent of the RDF repository and the different datasets available there.
Next steps	The SPARQL queries will be created to collect all the indicators defined in the document D3.2.

Table 27: Integration description of the Reference data service

3.3.2.3 **Decision support**

For the first prototype Onto will deliver first version of decision support system. This first version is represented as SPARQL queries to compare specific geopolitical indicators like GDP rates of a country, corruption levels etc. Such kind of information will help in the process of decision making. We also added support for Google charts, so the data can be easily visualised using different kind of charts depending on the specific use case. These SPARQL queries can be executed remotely and also can be used as a query template. The identification of the specific query is done by ID. We also support variable binding, so the users can change some parameters to the query and retrieve different results.

This service is deployed and fully available on **http://multisensor.ontotext.com**.

Service name	**Decision support**
SVN artefact	ms-svc-decsupport
Development status	Version 0.5, baseline version
Deployment status	Deployed
Integration status	Integrated
Integration issues / dependencies	No integration issues The Reference data is fully dependent of the RDF repository and the different datasets available there.
Next steps	Extend the SPARQL queries.

Table 28: Integration description of the Decision support service

The previous online services are integrated in the three UC application. In the next section, the description of the UC applications is provided regarding the new functionalities.

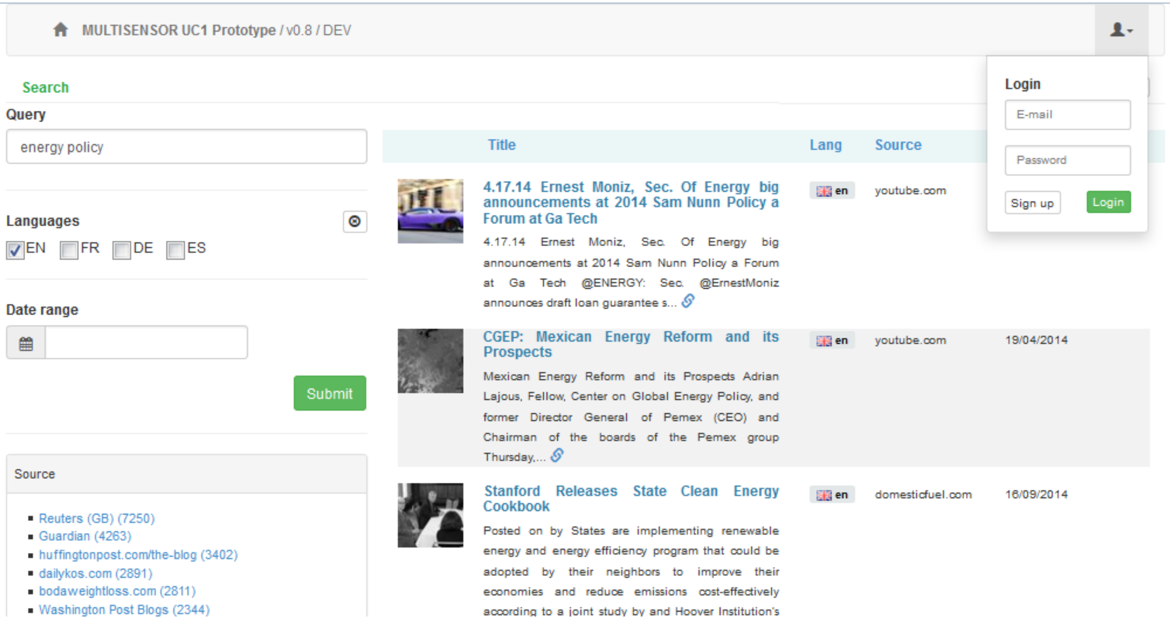
4 PROTOTYPE APPLICATIONS

4.1 UC1: Journalism Use Case

The UC1 application (see D7.1, section 4.1) is an application that should support media professionals (e.g. journalist, media expert) to find relevant information in different formats, coming from different sources, and according the social activities that were produced around.

The description of the user interfaces have been provided in the D7.3 deliverable (see D7.3, section 3.1). The interfaces are similar but instead of displaying fake content, the online services that are using the analytic data stored in the repositories (OPS or RDF) are exploited. Here, only the UI improvements are presented.

The User profile service is integrated in the UC1 application. The end user can register, login and logout from the system.



MULTISENSOR UC1 Prototype / v0.8 / DEV

Search

Query: energy policy

Languages: [✓ EN] FR DE ES

Date range

Submit

Source
- Reuters (GB) (7250)
- Guardian (4263)
- huffingtonpost.com/the-blog (3402)
- dailykos.com (2891)
- bodaweightloss.com (2811)
- Washington Post Blogs (2344)

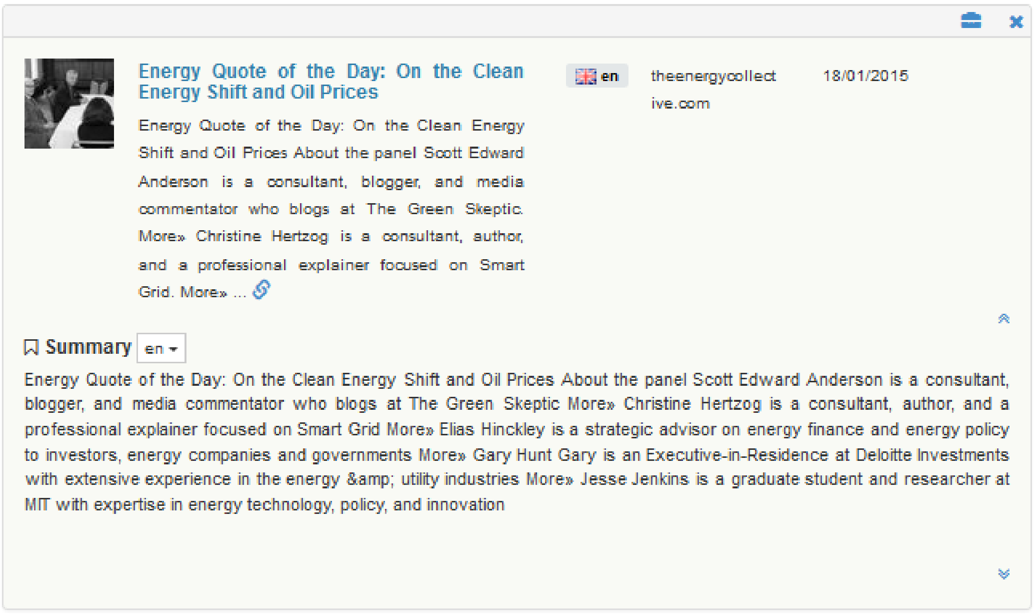
Title	Lang	Source	
4.17.14 Ernest Moniz, Sec. Of Energy big announcements at 2014 Sam Nunn Policy a Forum at Ga Tech 4.17.14 Ernest Moniz, Sec. Of Energy big announcements at 2014 Sam Nunn Policy a Forum at Ga Tech @ENERGY: Sec. @ErnestMoniz announces draft loan guarantee s...	en	youtube.com	
CGEP: Mexican Energy Reform and its Prospects Mexican Energy Reform and its Prospects Adrian Lajous, Fellow, Center on Global Energy Policy, and former Director General of Pemex (CEO) and Chairman of the boards of the Pemex group Thursday,...	en	youtube.com	19/04/2014
Stanford Releases State Clean Energy Cookbook Posted on by States are implementing renewable energy and energy efficiency program that could be adopted by their neighbors to improve their economies and reduce emissions cost-effectively according to a joint study by and Hoover Institution's	en	domesticfuel.com	16/09/2014

Login
E-mail
Password
Sign up
Login

Figure 8: Widget for user registration and login

As for the Operational Prototype, the CNR operates as the search engine. The user can search for specific topics in the full textual mode and in different languages. Most relevant articles are retrieved and the user can browse them, get specific information on them, and do a specific selection.

When the user is registered in the application, he is able to select the relevant articles. On the right top, by clicking on the folder icon, it permits to add the selected document in the specified folder.



Energy Quote of the Day: On the Clean Energy Shift and Oil Prices

Energy Quote of the Day: On the Clean Energy Shift and Oil Prices About the panel Scott Edward Anderson is a consultant, blogger, and media commentator who blogs at The Green Skeptic. More» Christine Hertzog is a consultant, author, and a professional explainer focused on Smart Grid. More» ...

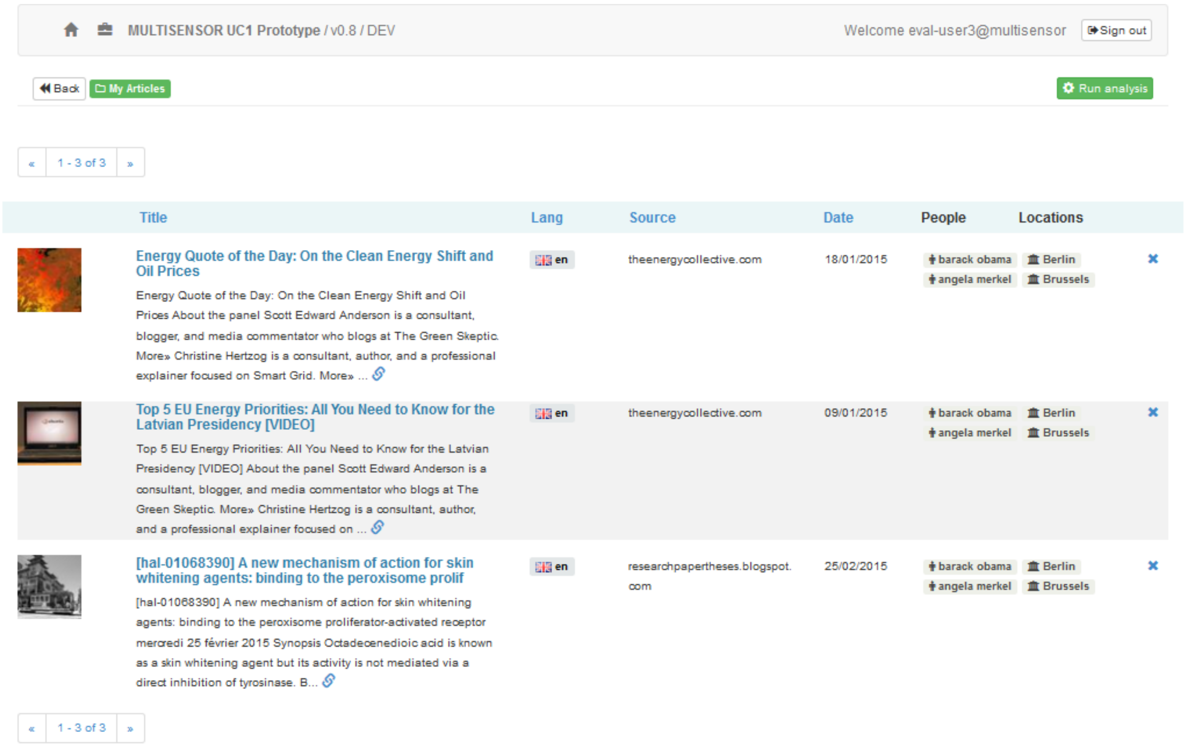
en | theenergycollective.com | 18/01/2015

Summary en

Energy Quote of the Day: On the Clean Energy Shift and Oil Prices About the panel Scott Edward Anderson is a consultant, blogger, and media commentator who blogs at The Green Skeptic More» Christine Hertzog is a consultant, author, and a professional explainer focused on Smart Grid More» Elias Hinckley is a strategic advisor on energy finance and energy policy to investors, energy companies and governments More» Gary Hunt Gary is an Executive-in-Residence at Deloitte Investments with extensive experience in the energy &amp; utility industries More» Jesse Jenkins is a graduate student and researcher at MIT with expertise in energy technology, policy, and innovation

Figure 9: Add the selected article in the folder

By selecting the “My Findings” button, the user has access to the list of the articles available in its folder.



MULTISENSOR UC1 Prototype / v0.8 / DEV   Welcome eval-user3@multisensor   Sign out

Back   My Articles   Run analysis

« 1 - 3 of 3 »

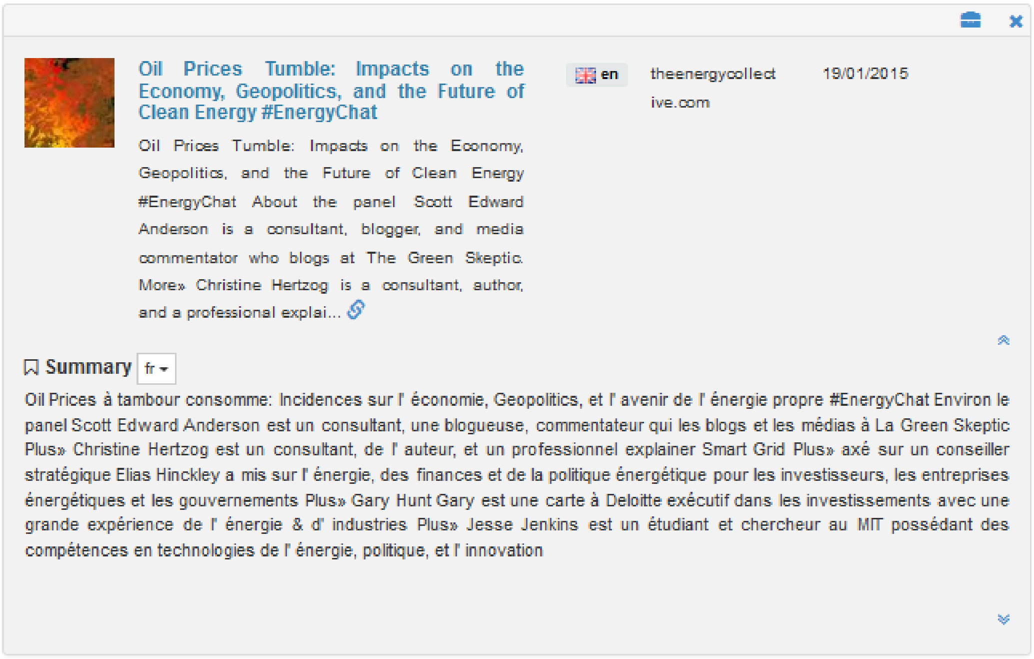
Title	Lang	Source	Date	People	Locations
Energy Quote of the Day: On the Clean Energy Shift and Oil Prices Energy Quote of the Day: On the Clean Energy Shift and Oil Prices About the panel Scott Edward Anderson is a consultant, blogger, and media commentator who blogs at The Green Skeptic. More» Christine Hertzog is a consultant, author, and a professional explainer focused on Smart Grid. More» ...	en	theenergycollective.com	18/01/2015	barack obama, angela merkel	Berlin, Brussels
Top 5 EU Energy Priorities: All You Need to Know for the Latvian Presidency [VIDEO] Top 5 EU Energy Priorities: All You Need to Know for the Latvian Presidency [VIDEO] About the panel Scott Edward Anderson is a consultant, blogger, and media commentator who blogs at The Green Skeptic. More» Christine Hertzog is a consultant, author, and a professional explainer focused on ...	en	theenergycollective.com	09/01/2015	barack obama, angela merkel	Berlin, Brussels
[hal-01068390] A new mechanism of action for skin whitening agents: binding to the peroxisome prolif [hal-01068390] A new mechanism of action for skin whitening agents: binding to the peroxisome proliferator-activated receptor mercredi 25 février 2015 Synopsis Octadecenedioic acid is known as a skin whitening agent but its activity is not mediated via a direct inhibition of tyrosinase. B...	en	researchpapertheses.blogspot.com	25/02/2015	barack obama, angela merkel	Berlin, Brussels

« 1 - 3 of 3 »

Figure 10: List of the selected article in the folder

When the user select an article in the results list, the summary is displayed for English articles.

Then, the user has the possibility to ask the summary translation in three different languages (French, Spanish and German).



Oil Prices Tumble: Impacts on the Economy, Geopolitics, and the Future of Clean Energy #EnergyChat

en theenergycollective.com 19/01/2015

Oil Prices Tumble: Impacts on the Economy, Geopolitics, and the Future of Clean Energy #EnergyChat About the panel Scott Edward Anderson is a consultant, blogger, and media commentator who blogs at The Green Skeptic. More» Christine Hertzog is a consultant, author, and a professional explai...

Summary fr

Oil Prices à tambour consomme: Incidences sur l' économie, Geopolitics, et l' avenir de l' énergie propre #EnergyChat Environ le panel Scott Edward Anderson est un consultant, une blogueuse, commentateur qui les blogs et les médias à La Green Skeptic Plus» Christine Hertzog est un consultant, de l' auteur, et un professionnel explainer Smart Grid Plus» axé sur un conseiller stratégique Elias Hinckley a mis sur l' énergie, des finances et de la politique énergétique pour les investisseurs, les entreprises énergétiques et les gouvernements Plus» Gary Hunt Gary est une carte à Deloitte exécutif dans les investissements avec une grande expérience de l' énergie & d' industries Plus» Jesse Jenkins est un étudiant et chercheur au MIT possédant des compétences en technologies de l' énergie, politique, et l' innovation

Figure 11: Display of the summary translated in French

Here is the summary of the integration of the Online Services with the UC1 application.

- <u>Profile service:</u> Integrated (fully for user and profile specific functionalities except for saving relevance and accessing relevant articles)
- <u>Contributor analysis service:</u> integrated as an API that provides the most popular accounts related to a Twitter handle.
- <u>Semantic search service:</u> FTS functionality is implemented by calling elastic search API. Facets are currently faked and need to be switched to the corresponding information as soon as articles are enriched. Also disambiguation will be implemented as soon as the service is fully deployed.
- <u>Similarity search service:</u> not integrated because it is provided as a dummy service.
- <u>Translations service:</u> Translation implemented for the summary of the article.
- <u>Summarisation service:</u> Extractive summarisation is implemented but only dummy data is displayed at this moment. Abstractive summarisation is not due for this prototype.
- <u>Content delivery service</u>: Service is implemented to access additional data for single articles.
- <u>Clustering& Filtering service:</u> Clustering service is functional. The filtering service is partially implemented (a filter for relevant articles is still missing).
- <u>Reference data service:</u> This service is not applicable for UC1.

4.2 UC2: Media Monitoring Use Case

The profile serve will be finalised for the next prototype with functionalities to save the UI language and relevance of articles.

In the search service facets need to be implemented as well as search disambiguation.

The summary service will return actual data instead of dummy data as soon as articles are enriched with information in the offline pipeline.

For the contributor analysis service an API has not been provided yet in order to implement the contributor analysis feature in the UI. Precise functionalities have to be defined for the use case.

The semantic search service will be used for auto-completion and search suggestions as soon as it is integrated. Also disambiguation will then be implemented. Facets are currently dummy within the use case application.

Abstractive summaries are planned for future prototypes.

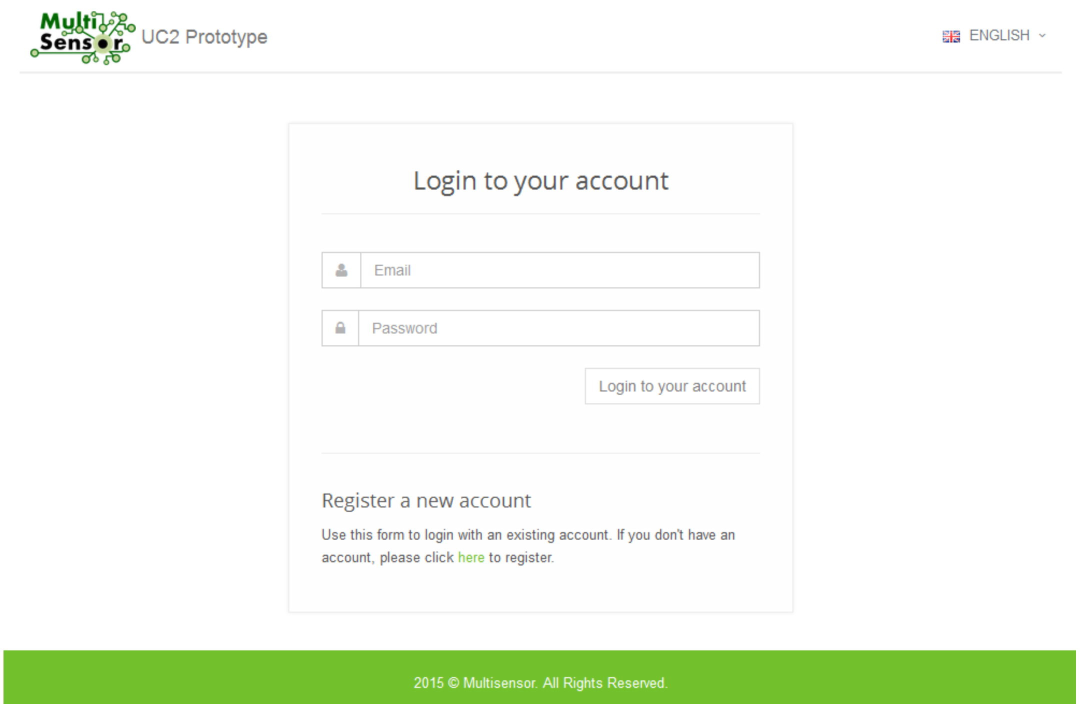
Cluster facets are foreseen but not implemented yet.

Filtering service for filter relevant/irrelevant articles needs to be implemented.

The media monitoring use case application is focused on replicating of a media monitoring professional's workflow for execution of an analysis as previously mentioned in D7.2.

All searches and analysis are user-specific, therefore authentication has been set up prior to using the application.

First the user has to login into the MULTISENSOR system:

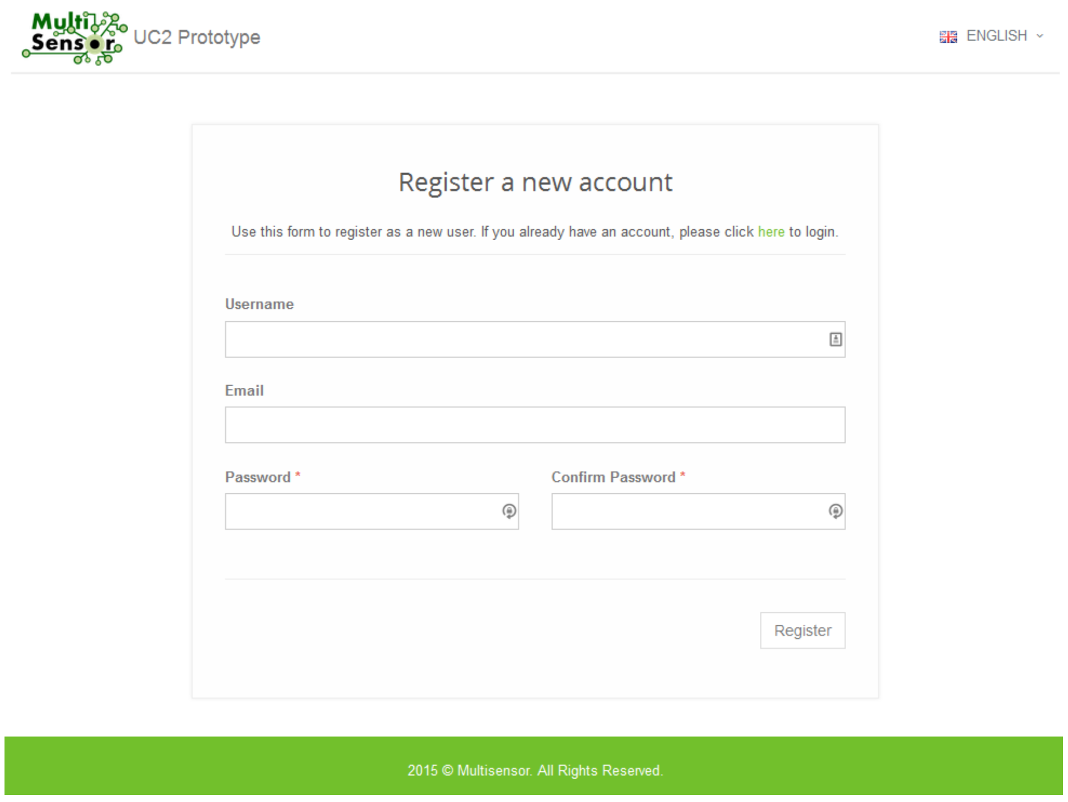


Use this form to login with an existing account. If you don't have an account, please click here to register.

2015 © Multisensor. All Rights Reserved.

Figure 12: Login screenshot

He/She needs to register if he/she is not registered yet:


UC2 Prototype

ENGLISH

Register a new account

Use this form to register as a new user. If you already have an account, please click here to login.

Username

Email

Password *

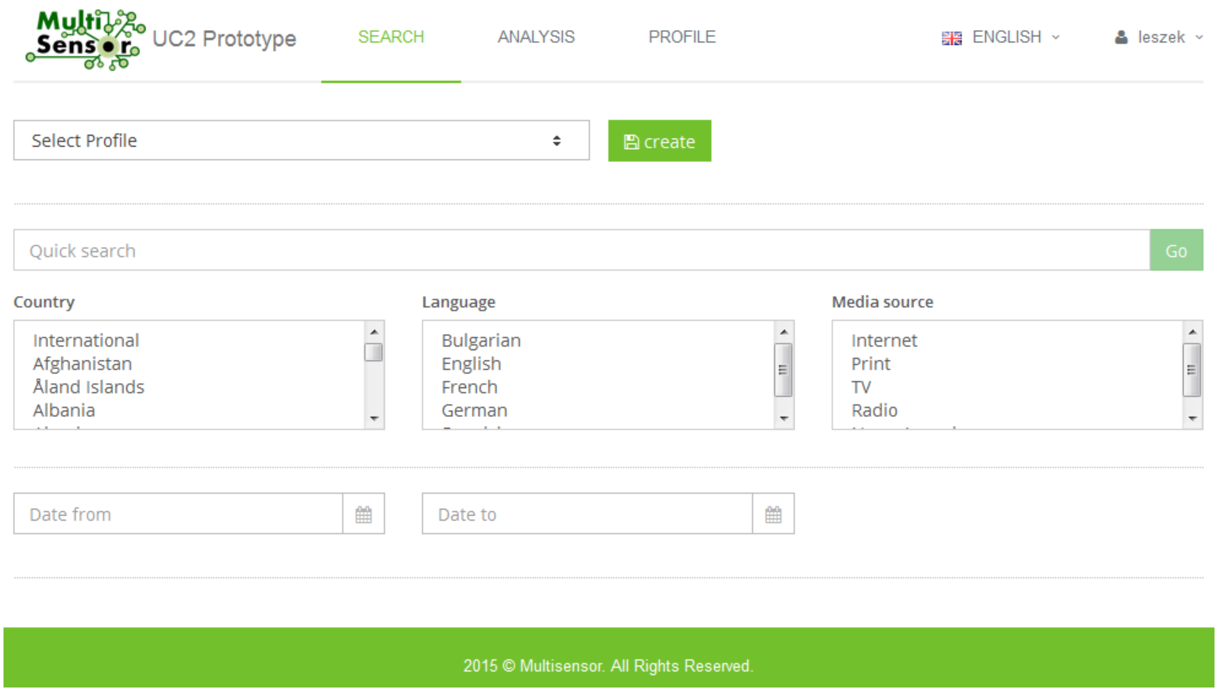
Confirm Password *

Register

2015 © Multisensor. All Rights Reserved.

Figure 13: Registration screenshot

After login the user is presented with a search mask to query a specific search.


UC2 Prototype | SEARCH | ANALYSIS | PROFILE | ENGLISH | leszek

Select Profile | create

Quick search | Go

Country: International, Afghanistan, Åland Islands, Albania

Language: Bulgarian, English, French, German

Media source: Internet, Print, TV, Radio

Date from | Date to

2015 © Multisensor. All Rights Reserved.

Figure 14: Search functionality

The search fields can be pre-populated by selecting an already existing profile in the upper dropdown-box.

Alternatively search terms and additional filters can be set without selecting a profile.

If a profile has not been selected the user has the chance to save his search settings as a new profile from within the search section.

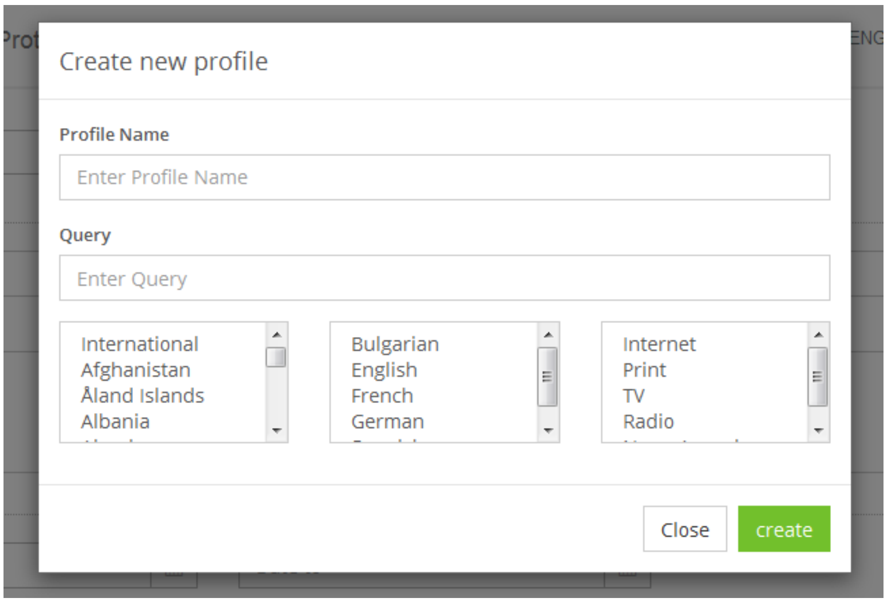


Figure 15: Page to create the new Profile

After a search query has been triggered, the user is presented with results which grouped by topics.

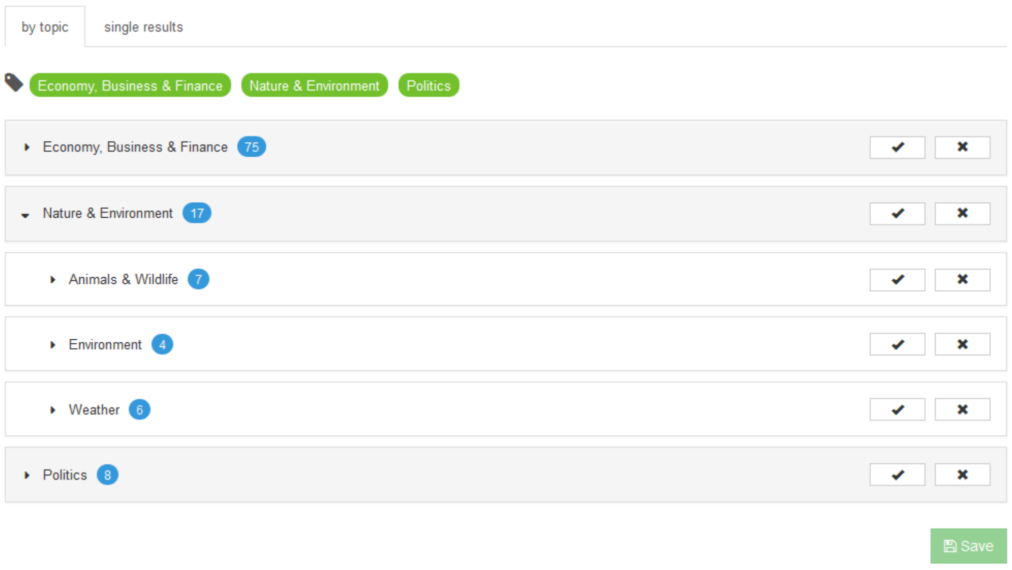


Figure 16: Search results grouped by topics

(Currently all articles are returned matching the query. In the next prototype only articles that have not been marked as relevant/irrelevant are returned. Relevance of articles will be stored per profile.) The ability to mark articles as relevant or irrelevant has been implemented in the UI in the upper right corner of the topic. In order to gather more information about the articles within a topic a tag cloud for the topic can be viewed.

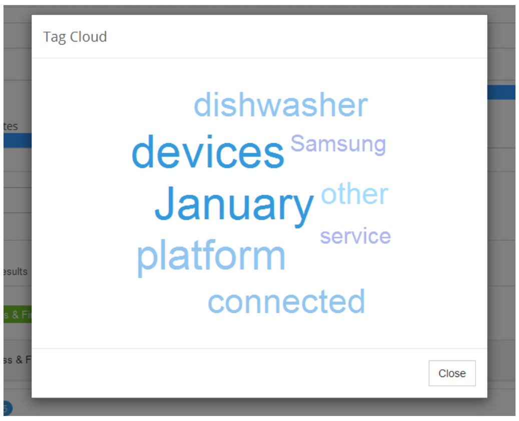
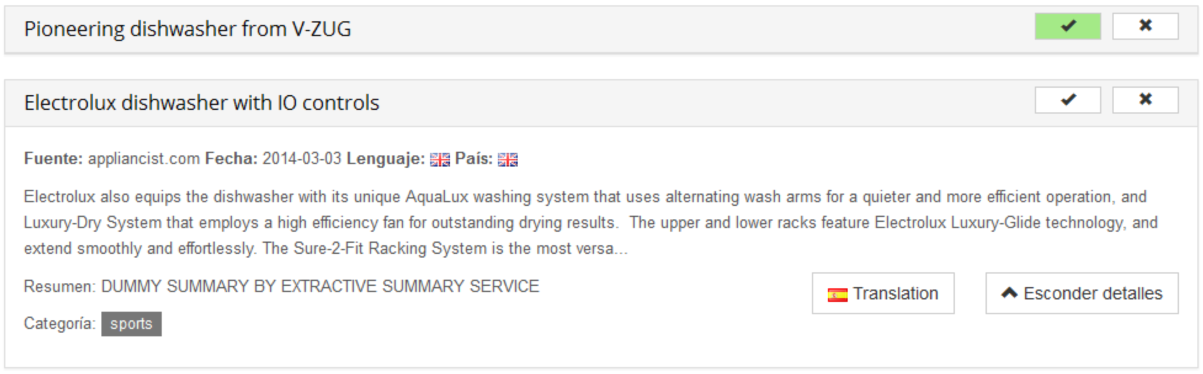


Figure 17: Tag cloud

Saving whether an article is relevant will be implemented in the next prototype.

A second detailed view of articles is implemented as well. This supports the user in checking for relevance of an article. After marking the article, it is reduced to the headline. Additionally more details can be displayed per article by using the “Show Details” button. Currently a dummy summary and a dummy category are displayed. Entities will be displayed as soon as articles are enriched with entity information. Social network information is planned for the prototype, which is due in October 2015.

Also a function to translate the article is integrated. A button to call the translation service is displayed if article language and user interface language differ, e.g. n user with a Spanish UI can translate all non-Spanish articles into Spanish. Obviously Spanish articles should not be translated into Spanish.


Pioneering dishwasher from V-ZUG

Electrolux dishwasher with IO controls

Fuente: appliancist.com **Fecha:** 2014-03-03 **Lenguaje:** **País:**

Electrolux also equips the dishwasher with its unique AquaLux washing system that uses alternating wash arms for a quieter and more efficient operation, and Luxury-Dry System that employs a high efficiency fan for outstanding drying results. The upper and lower racks feature Electrolux Luxury-Glide technology, and extend smoothly and effortlessly. The Sure-2-Fit Racking System is the most versa...

Resumen: DUMMY SUMMARY BY EXTRACTIVE SUMMARY SERVICE

Categoría: sports

Figure 18: List of the search results

For displaying and analysing results for a client the user can switch to the “Analysis” section. Here he also has the opportunity to pre-select a saved profile and to pre-populate the search boxes. After querying a search only articles that have been marked as relevant previously will be returned. The results are displayed in different charts (e.g. by country, by sources...) and the user also has the possibility to change chart contents (either the content that is displayed or the number of chart bars/pie slices).

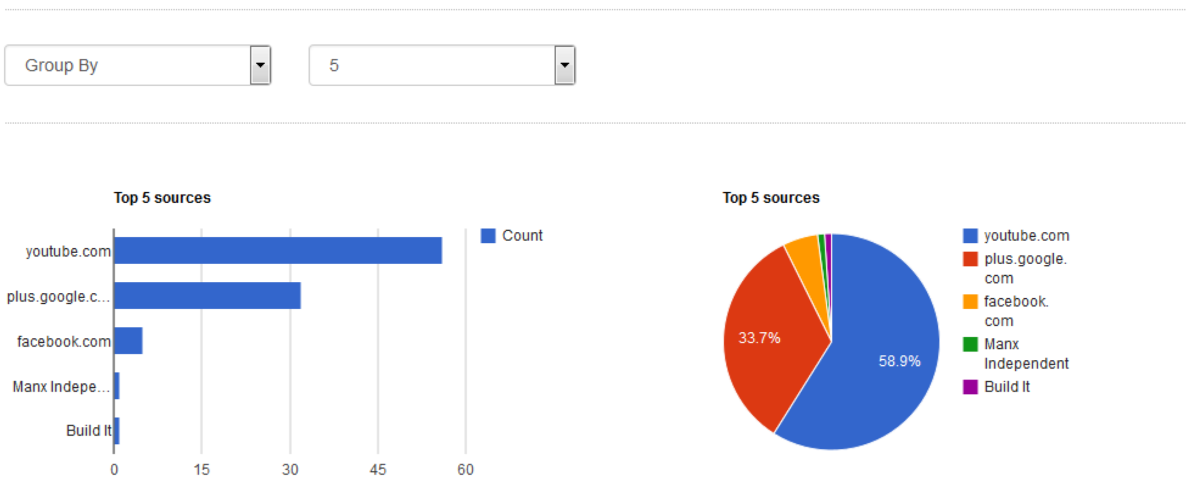


Figure 19: View of the analysis results

Also a selection of the most current articles is displayed.

In addition, a tag cloud is planned and the displaying of social media networks if twitter articles are included in the result set. Significant contributor scores are foreseen as soon as the services are provided.

Within the "Profile" section the user can edit his profiles that have previously been stored.

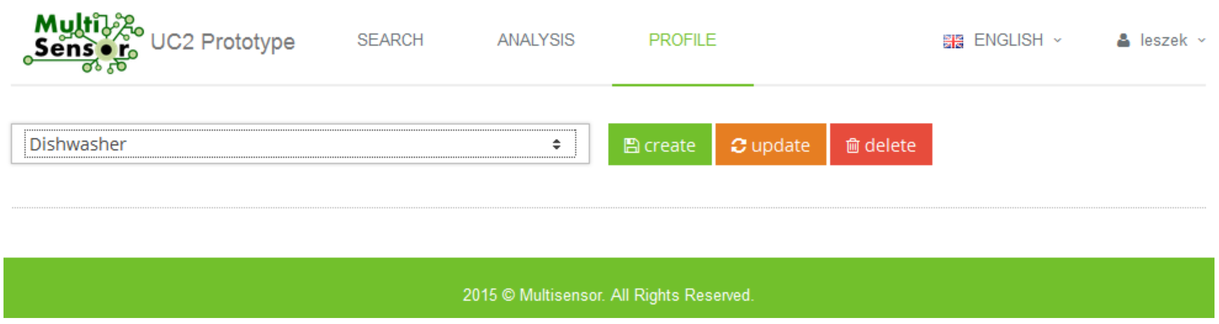


Figure 20: Page to edit the profiles

Functionalities to create, update and delete profiles have been fully implemented.

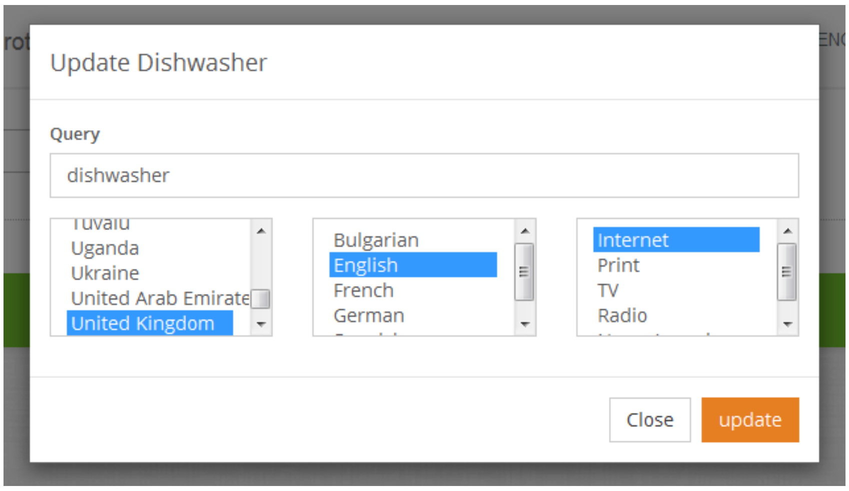


Figure 21: Page to update the profiles

Here is the summary of the integration of the Online Services with the UC2 application.

- Profile service: Has been fully implemented regarding user and profile functionalities except saving the user interface language. Functionality to store relevance of articles needs to be implemented as well in the near future.
- Contributor analysis service: precise functionalities have to be defined for the use case.
- Semantic search service: The full text search capabilities of ElasticSearch have been implemented for keyword based search. Facets are currently faked and need to be switched to the corresponding information as soon as articles are enriched. Also disambiguation will be implemented as soon as the service is fully deployed.
- Similarity search service: The usage of this service needs further elaboration according to D8.2 in order to implement a function in the UC2 prototype.
- Translation service: It is implemented in the user interface for single articles.
- Summarisation service: Extractive summarisation is implemented and integrated. Abstractive summarisation is not due for this prototype.
- Content delivery service: The service is implemented to access additional data for single articles (e.g. extractive summary, entities (currently dummy), categories (currently dummy)...).
- Clustering & Filtering service: Filtering service is implemented partially. Still missing is a filter for relevant articles.
- Reference data service: This service is not applicable for UC2.
- Decision support service: This service is not applicable for UC2.

The profile serve will be finalised for the next prototype with functionalities to save the UI language and relevance of articles.

In the search service facets need to be implemented as well as search disambiguation.

The summary service will return actual data instead of dummy data as soon as articles are enriched with information in the offline pipeline.

For the contributor analysis service an API has not been provided yet in order to implement the contributor analysis feature in the UI. Precise functionalities have to be defined for the use case.

The semantic search service will be used for auto-completion and search suggestions as soon as it is integrated. Also disambiguation will then be implemented. Facets are currently dummy within the use case application.

Abstractive summaries are planned for future prototypes.

Cluster facets are foreseen but not implemented yet.

Filtering service for filter relevant/irrelevant articles needs to be implemented yet.

4.3 UC3: SME internationalisation Use Case

The UC3 application (see D7.1, section 4.2) is an application that should the SMEs to start a process of internationalisation with any kind of products. Relevant information related to the countries, the economic situation of the market, the legal information, etc. should be retrieved easily to support the decision making.

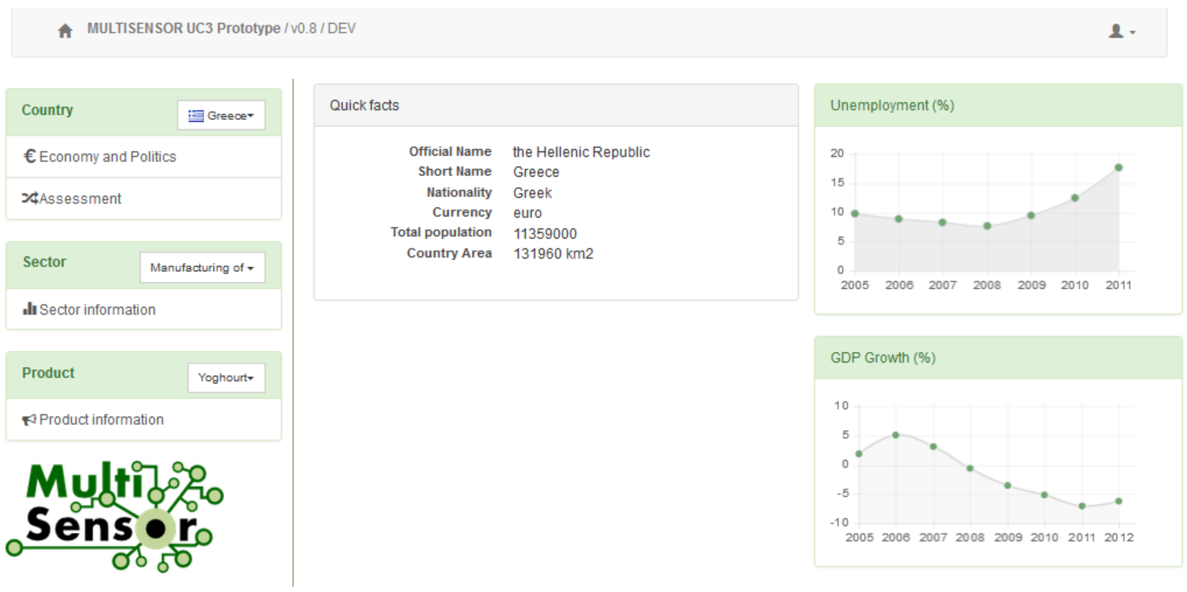
As the UC1, the description of the user interfaces have been provided in the D7.3 deliverable (see D7.3, section 3.2). The interfaces are similar but instead of displaying fake content, the online services that are using the analytic data stored in the repositories (OPS or RDF) are exploited. Here, only the UI improvements are presented.

The user can search by three different criteria: by country, by sector and by product. For the country search and for the product search, the system can present specific facts and retrieval results.

In the case of the country search, the system can collect some basic indicators in the "Economy and Politics" category and some other indicators in the "Assessment" category.

For the "Economy and Politics" category, the application collect some quick facts about the selected country from the DBpedia dataset that is stored in GraphDB.

Also, more indicators are displayed in a graphical representation containing the GDP growth and the public debt of the country (see Figure 19). The percentage of the unemployment and GDP growth per year are displayed in a graphic representation. These indicators are collected from the World Bank dataset (also stored in GraphDB).

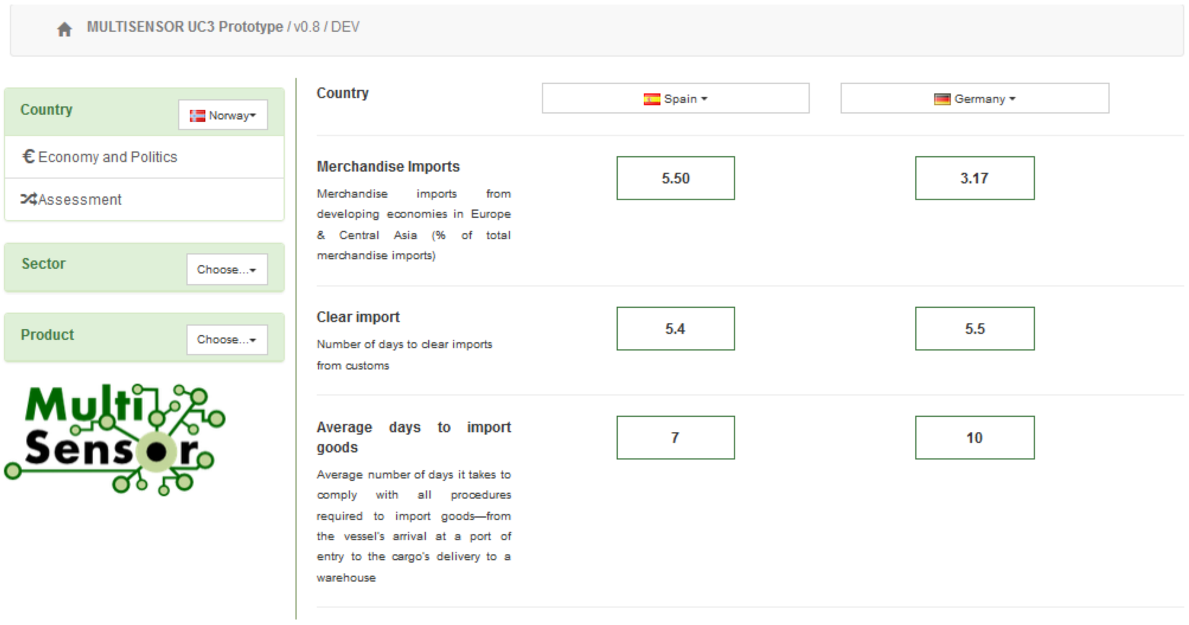


Official Name	the Hellenic Republic
Short Name	Greece
Nationality	Greek
Currency	euro
Total population	11359000
Country Area	131960 km2

Figure 22: Display of the country quick facts and main indicators

In the case of the "Assessment" category, a number of indicators are compared, in order to support the decision of the end user. In the FP, three main indicators are compared:

- Merchandise Imports: merchandise imports from developing economies in Europe & Central Asia (% of total merchandise imports)
- Days to clear import: number of days to clear imports from customs
- Average days to import goods: average number of days it takes to comply with all procedures required to import goods—from the vessel's arrival at a port of entry to the cargo's delivery to a warehouse



MULTISENSOR UC3 Prototype / v0.8 / DEV

Country Norway

€ Economy and Politics

Assessment

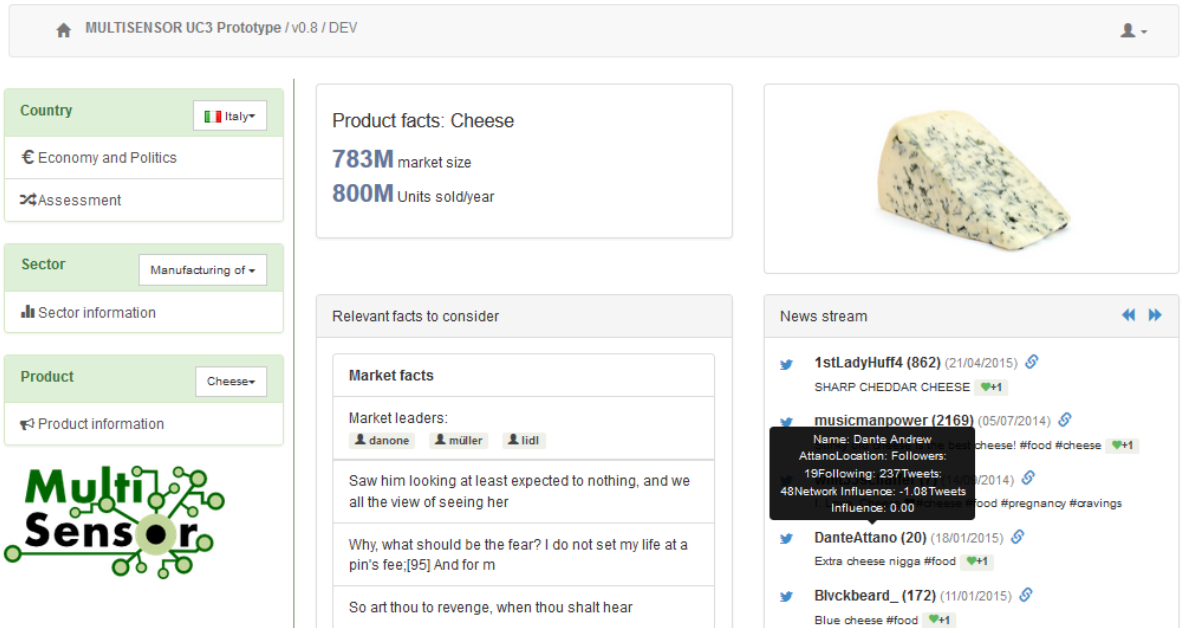
Sector Choose...

Product Choose...

Country	Spain	Germany
Merchandise Imports Merchandise imports from developing economies in Europe & Central Asia (% of total merchandise imports)	5.50	3.17
Clear import Number of days to clear imports from customs	5.4	5.5
Average days to import goods Average number of days it takes to comply with all procedures required to import goods—from the vessel's arrival at a port of entry to the cargo's delivery to a warehouse	7	10

Figure 23: Display the indicators comparison for the decision support

In the case of the Product search, some news stream are collected from social networks. For each one, thanks to the Contributor analysis, the user access to the information related to the twitter account by clicking on it.



MULTISENSOR UC3 Prototype / v0.8 / DEV

Country Italy

€ Economy and Politics

Assessment

Sector Manufacturing of

Sector information

Product Cheese

Product information

Product facts: Cheese

783M market size

800M Units sold/year

Relevant facts to consider

Market facts

Market leaders:
danone müller lidl

Saw him looking at least expected to nothing, and we all the view of seeing her

Why, what should be the fear? I do not set my life at a pin's fee;[95] And for m

So art thou to revenge, when thou shalt hear

News stream

1stLadyHuff4 (862) (21/04/2015)
SHARP CHEDDAR CHEESE +1

musicmanpower (2169) (05/07/2014)
cheese! #food #cheese +1

Name: Dante Andrew AttanoLocation: Followers: 19Following: 237Tweets: 48Network Influence: -1.08Tweets Influence: 0.00

(14/09/2014)
#cheese #food #pregnancy #cravings

DanteAttano (20) (18/01/2015)
Extra cheese nigga #food +1

Blvckbeard_ (172) (11/01/2015)
Blue cheese #food +1

Figure 24: Display information from the contributor analysis

Here is the summary of the integration of the Online Services with the UC3 application.

- Profile service: as the user profile is not relevant for this use case, it is not yet integrated.
- Contributor analysis service: integrated as an API that provides the most popular accounts related to a Twitter handle.
- Semantic search service: The full text search capabilities of ElasticSearch have been implemented for keyword based search. Facets are currently faked and need to be

switched to the corresponding information as soon as articles are enriched. Also disambiguation will be implemented as soon as the service is fully deployed.

- Similarity search service: not integrated because it is provided as a dummy service.
- Translations service: not integrated because no specific text is yet presented.
- Summarisation service: not integrated because no specific text is yet presented.
- Content delivery service: the service is implemented to access additional data for single articles.
- Clustering + Filtering service: Filtering service is implemented partially. Still missing is a filter for relevant articles.
- Reference data service: integrated as a data wrapper that collects the data from the Linked Open Data datasets uploaded in GraphDB. Results are obtained by querying the SPARQL endpoint with query templates.
- Decision support service: integrated as a SPARQL query to compare specific indicators between two countries.

For the First Prototype, most of the Online services are integrated in the UC applications. Some functionalities are still in a basic version because of the qualitative work to store relevant knowledge in the RDF repository is still in progress. Whether this task finished, the new search engine functionality will be enabled thanks to the ElasticSearch connector with GraphDB provided by OntoText.

5 CODE ORGANISATION

5.1 Source tree layout (D7.3 updates)

All the MULTISENSOR code and related artefacts are kept in a Subversion[13] repository in everis' premises and organised on a per-Work Package basis. The root of the source tree is located at **https://quark.everis.com/svn/MULTISENSOR/trunk**.

A breakdown of the layout of the repository follows.

- **wp1**: WP1 artefacts

- **wp2:** WP2 artefacts
  - **ms-svc-dep**: Dependency Parsing Service, Maven package (see D7.1, section 3.1.2).
  - **ms-svc-extr**: Concept Extraction Service, Maven package (see D7.1, section 3.1.2).
  - **ms-svc-rel**: Relation Extraction Service, Maven package (see D7.1, section 3.1.2).
  - **nifutils**: Library to manage NIF formats, Maven package (complementary tool for the Concept extraction, Dependency parsing and the Relation extraction).
  - **ms-svc-topicDetection**: Concept and Event Detection Service, Maven package (see D7.1, section 3.1.5).

- **wp3**: WP3 artefacts
  - **ms-svc-context**: Context Extraction Service, Maven package (see D7.1, section 3.2.1).
  - **ms-svc-contributorAnalysis**: Social Graph Service (see D7.2, pp. 42).
  - **ms-svc-sa**: Sentiment Analysis service, Maven package (see D7.1, section 3.2.2).

- **wp4:** WP4 artefacts
  - **ms-svc-categoryClassification**: Category Classification Service, Maven package (see D7.1, section 3.3.2).
  - **ms-svc-contentAlignment**: code related to Content Alignment pipeline (see D7.2, section 4.2.2.5).
  - **ms-svc-multimediaStructure**: indexing Service (see D7.2, section 4.2.2.5).
  - **ms-svc-similaritySearch**: Similarity Search Service (see D7.2, section 4.2.3.1).

- **wp5:** WP5 artefacts
  - **ms-svc-decsupport**: UC3 Decision Support Service, Maven package (see D7.2, section 4.2.3.4).

- **wp6:** WP6 artefacts

[13] See https://subversion.apache.org/

    - **ms-vc-summ**: Summarisation service, Maven package (see D7.2, section 4.2.3.1).

- **wp7:** WP7 artefacts
    - **crawler**: Crawler engine (see D7.2, section 4.2.2.2).
    - **ms-common**: Shared Java library and services for services.
    - **ms-crawler-socialmedia**: Yahoo! Crawler, Maven package (see D7.2, pp. 32).
    - **ms-js-common**: Shared Node.js modules and utilities.
    - **ms-parent**: Parent Maven package for all MULTISENSOR packages.
    - **ms-svc-cdelivery**: Content Delivery service, Maven package (see D7.2, section 4.2.3.1).
    - **ms-svc-refdata**: UC3 Reference Data service, Maven package (see D7.2, pp. 47).
    - **supervisor**: Supervisor Node.js (see D7.2, section 4.2.2.1).
    - **uc**: Use Case portals
        - **uc1**: UC1 Node.js application and related artefacts
        - **uc2**: UC2 Angular JS application and related artefacts
        - **uc3**: UC3 Node.js application and related artefacts
        - **uclib**: Shared Node.js modules and libraries for UC applications.

- **wp8**: WP8 artefacts

- **wp9**: WP9 artefacts

5.2 Packaging (D7.3 updates)

5.2.1 Java modules

All Java modules are packaged as Maven[14] artefacts for automated build, test and deployment capabilities.

In order to keep dependency management in check and ensure consistent use of package and library versions, all packages in the MULTISENSOR platform use a parent package, **wp7/ms-parent**. This package provides versions for common dependencies and specifies shared build properties.

Additionally, a transversal module, **wp7/ms-common**, provides shared features for all services. This includes constants, common classes and interfaces, access to shared resources, wrappers to access common services, and more. All services must depend on this package.

Most notably, the ms-common package contains a Bootstrap class, which calls the supervisor to bootstrap into the platform, retrieving the shared configuration for coordination with the rest of services.

[14] See http://maven.apache.org

5.2.2 Node.js modules

The Node.js modules are built as self-contained applications. They all have a package.json file which describes their dependencies and allows using npm[15] to download and install them.

A special module, **ms-js-common**, contains shared modules across the rest of Node.js applications.

[15] See **https://www.npmjs.org/**

6 INFRASTRUCTURE

The FP1 is running in Amazon EC2 cloud infrastructure provisioned by everis. Rationale and plans for scaling and provisioning is discussed in D7.2, section 5.

6.1 Current farm (D7.3 updates)

All servers run Ubuntu Linux 14.04.1 LTS ("Trusty") on x64 architecture. Ubuntu is hugely popular and as such, Personal Package Archives (PPAs) and vendor repositories are readily available providing very recent versions of core packages of MULTISENSOR (mongodb, elasticsearch, nodejs).

6.1.1 Mscrawler1

The mscrawler1 hosts the Yahoo! Crawler described in D7.2, pp.32. This is a small server dedicated to crawling targeted sites using a combination of Hadoop, Nutch and HBase. In the First Prototype, it has been successfully deployed but not yet integrated with the main platform.

The crawler1 has the following specs:

- 1x x64 core (2 ECUs).
- 3.75 GB RAM.
- 32 GB local SSD storage (ext4).
- 100GB EBS SSD storage (ext4).

6.1.2 Msgrinder1

An extra server, called **msgrinder1**, has already been commissioned for use in the platform, but has been integrated with the prototype. It is now hosting the Content Extraction Pipeline Services, the repositories and the three UC applications.

The grinder1 has the following specs:

- 16x x64 core (52 ECUs).
- 122 GB RAM.
- 300 GB local SSD storage (xfs).
- 100 GB EBS SSD storage (ext4).

7 DEMONSTRATOR URLS AND INFORMATION

The following URLs can be used to access the different applications and the code of the MULTISENSOR First Prototype:

UC1 Application: http://grinder1.multisensorproject.eu/uc1/

UC2 Application: http:// grinder1.multisensorproject.eu/uc2/

UC3 Application: http:// grinder1.multisensorproject.eu/uc3/

SVN repository: **https://quark.everis.com/svn/MULTISENSOR/trunk/**

(Credentials for access to the environments will be provided privately by other channels).

8 SUMMARY AND CONCLUSIONS

In D7.4 was presented the status of the First Prototype. It explains the current status of the system in terms of repositories, services, processes and workflows of the First Prototype. Also, it includes some updates regarding the architecture and the integration of all the Offline and Online services. This can be summarised as follows:

- The system was migrated in a most important server and the configuration has been improved.
- The four repositories of the Data Layer are in place
- The two crawlers are now implemented and could collect complementary documents (multimedia and social media)
- All the main services are implemented, only few are presented as a baseline version.
- The different services are integrated in the platform, i.e. they can interact between themselves, store and collect data from the different repositories.
- The RDF repository is not fully populated with the extracted knowledge produced by the CEP. For this reason, the search functionality is still provided by the CNR instead of GraphDB.
- And the UI for the three Use Cases are improved with the interaction of the available online services.

The development iteration of the First Prototype was a huge challenge to connect and integrate such different services in the platform. Many technical issues regarding dependencies between those services and other libraries were addressed to obtain a stable and coherent integration and to start populating the repositories with analytical data.

Then, the most important work was focused on the services integration of the Offline modality to analyse the news collected and stored in the CNR in order to populate the main repositories (OPS and RDF) with analytic data. This allows to aliment the online services with real data and improve the UI of the three UC applications.

Starting with the data population of the repositories, it seems the big challenge for the next iteration is to populate GraphDB well-tuned RDF content to enable powerful search functionalities. Another challenge will be to control the data quality and maintain properly the different repositories. Then, most of the Online services will be considerably improved to exploit correctly the knowledge base and enrich significantly the user experience.