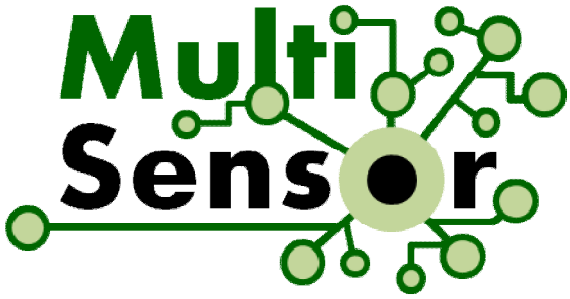

MULTISENSOR

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

FP7-610411

D7.3

Graphic interfaces and operational prototype

Dissemination level:	Public
Contractual date of delivery:	Month 12, October 31, 2014
Actual date of delivery:	Month 12, October 31, 2014
Workpackage:	WP7 System development and integration
Task:	T7.4 System development
Type:	Report
Approval Status:	Final Draft
Version:	1.0
Number of pages:	28
Filename:	D7.3_OperationalPrototype_2014-10-31_v1.0.pdf

Abstract

This document describes the technical components and infrastructure of the initial Operational Prototype for the MULTISENSOR platform. It provides an overview of the demonstration application prototypes, the organisation and composition of the different modules, and the hosting infrastructure. The Operational Prototype will be the scaffolding on which the platform will be built iteratively, adding functionality and depth on top of the dummy-based setup which marks this first milestone.

The information in this document reflects only the author's views and the European Community is not liable for any use

that may be made of the information contained therein. The information in this document is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability.

co-funded by the European Union

History

Version	Date	Reason	Revised by
0.1	10/10/2014	Proposal of structure and draft content	A. Callabed (everis)
0.2	15/10/2014	First draft	A. Callabed (everis)
0.3	20/10/2014	Contributions	All
0.4	27/10/2014	Version for internal review	A. Callabed (everis)
0.5	30/10/2014	Internal review	L. Blacha (PR)
1.0	31/10/2014	Final version after feedback from internal review	A. Callabed (everis)

Author list

Organisation	Name	Contact Information
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	Anastasia Moumtzidou	moumtzid@iti.gr
BM-Y!	Ioannis Arapakis	arapakis@yahoo-inc.com
LINGUATEC	Vera Aleksić	v.aleksic@linguatec.de
UPF	Gerard Casamayor	gerard.casamayour@upf.edu
ONTO	Kiril Simov	kiril.simov@ontotext.com
PR	Mirja Eckhoff	mirja.eckhoff@pressrelations.de

Executive Summary

D7.3 of the MULTISENSOR platform presents a demonstration of the envisioned platform by means of an operational prototype. The prototype shows a rough sketch of the User Interface (UI) and dummy implementations of the functionalities of the system. The operational prototype is tested with two use cases: a) journalism and b) SME internationalisation.

This document provides a brief technical reference for the D7.3 prototype deliverable of the MULTISENSOR platform. First it presents the architecture and the modules involved. Then, the prototype applications testing the two aforementioned use cases are presented. In the next sections, the code organisation and the infrastructure are detailed. Finally, this document provides links to live demo of the prototype and to the code repository.

Abbreviations and Acronyms

CEP	Content Extraction Pipeline
CNR	Central News Repository
DB	DataBase
EBS	Elastic Block Storage
EC2	Elastic Compute Cloud
ECU	Elastic Compute Unit
JSON	JavaScript Object Notation
OPx	Operational Prototype x
OPS	OPerationS repository
PPA	Personal Package Archive
PR	pressrelations
RAM	Random Access Memory
RDF	Resource Definition Framework
REST	Representational State Transfer
SOA	Service Oriented Architecture
SQL	Structured Query Language
SSD	Solid-State Drive
UC	Use Case
UI	User Interface

Table of Contents

1 INTRODUCTION 8

2 PROTOTYPE ARCHITECTURE 9

2.1 Global view 9

2.2 Supervisor 9

2.3 Crawler 10

2.4 Content extraction pipeline 10

2.5 Central News Repository 11

2.6 OPS Repository 11

2.7 UC1 application 11

2.8 UC3 application 12

3 PROTOTYPE APPLICATIONS 13

3.1 UC1: Journalism Use Case 13

3.2 UC3: SME internationalisation Use Case 18

4 CODE ORGANISATION 23

4.1 Source tree layout 23

4.2 Packaging 24

4.2.1 Java modules 24

4.2.2 Node.js modules 24

5 INFRASTRUCTURE 25

5.1 Current farm 25

5.1.1 Msinfra1 25

5.1.2 Mscrawler1 26

5.1.3 Msgrinder1 26

5.2 Future plans 26

6 DEMONSTRATOR URLS AND INFORMATION.....27

7 SUMMARY AND CONCLUSIONS 28

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:

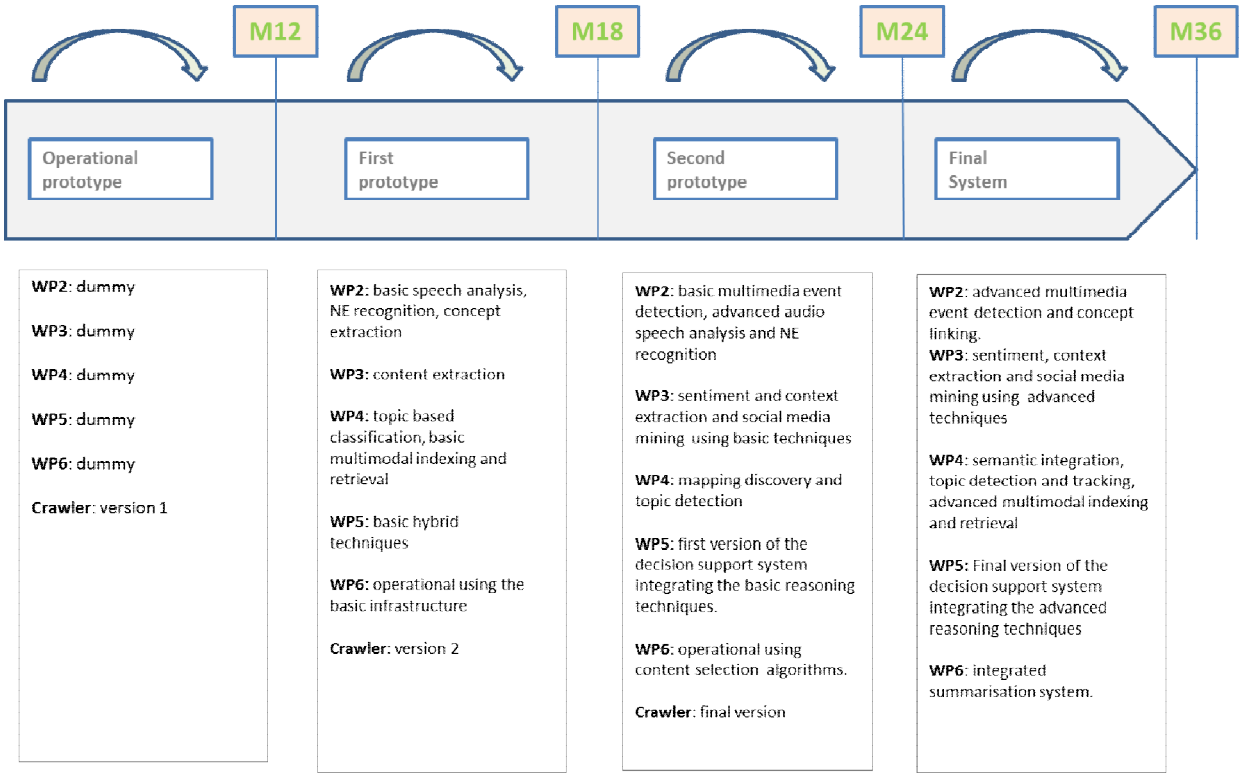

Figure 1: Technical roadmap

The purpose of this document is to provide a brief technical reference for the D7.3 deliverable, which is the first technical milestone of the project. D7.3 contains a first rough UI for the platform and dummy implementations of the major services, processes and workflows.

Section 2 contains a high-level technical overview of the prototype.

Section 3 contains a description of the demonstrator applications.

Section 4 provides a walk-through of the structure of the code, to assist in the navigation of the Subversion repository

Section 5 details the technical infrastructure hosting the prototype.

Section 6 contains links and details for accessing the demonstrator application for reviewers.

Section 7 presents a brief summary and conclusions.

2 PROTOTYPE ARCHITECTURE

2.1 Global view

The global architecture for the MULTISENSOR platform has been discussed at length 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).

For the D7.3 prototype, dummy implementations of the envisioned components of the architecture have been put in place. These dummies constitute a proof-of-concept for the overall design and are also the building blocks on which the actual implementations will be placed as they become available.

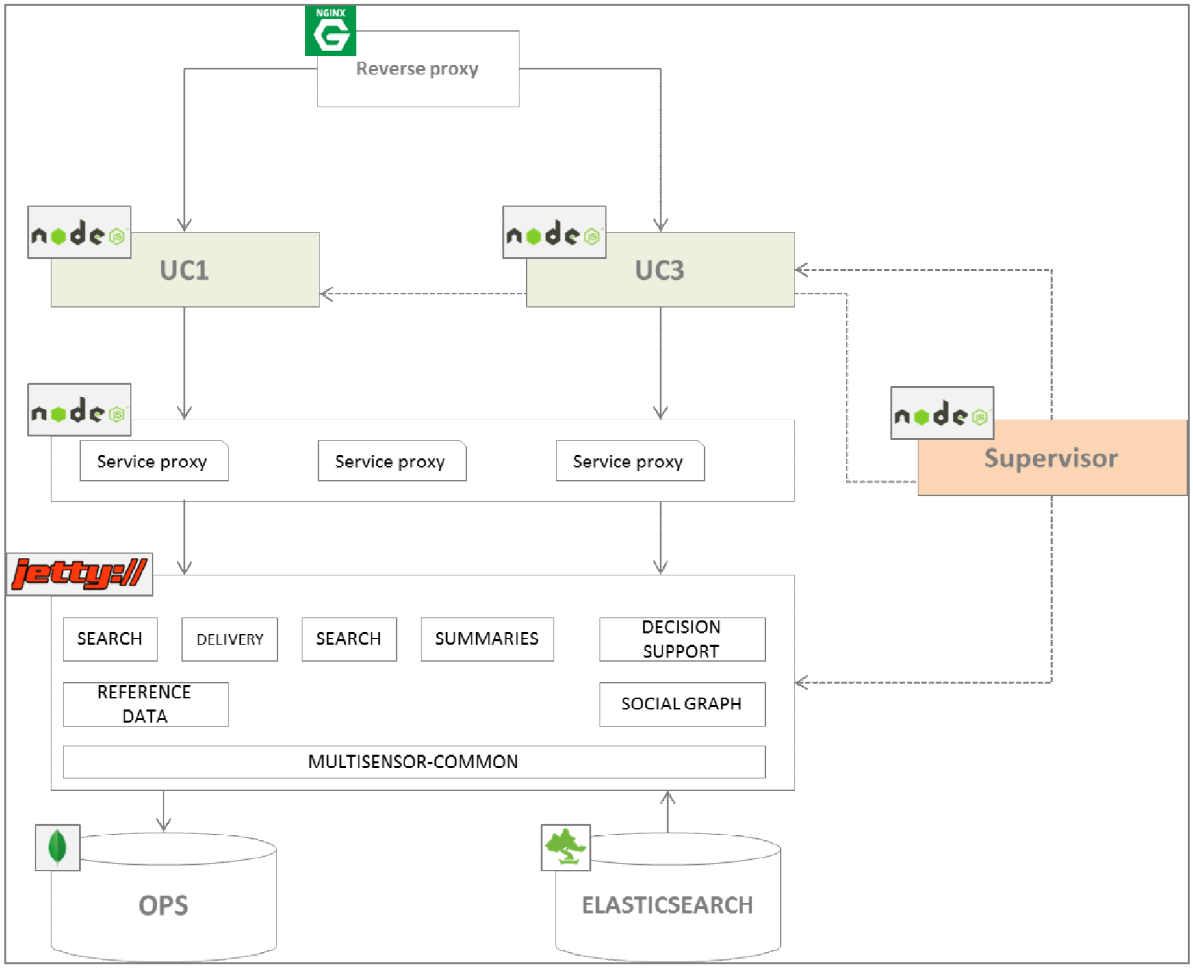

Figure 2: OP1 Architecture

2.2 Supervisor

The Supervisor process (see D7.2, Section 4.2.2.1) is always running and performs several duties in the platform.

For the operational prototype, it has 3 jobs:

- Runs the crawler periodically to collect news items. Currently it is configured to run every 30 minutes.
- Initiates the content extraction pipeline on demand.
- Hosts a bootstrap service to provide all other services with the shared platform configuration.

The Supervisor is built as a Node.js application. The intrinsically asynchronous nature of Node.js, which is based on an event loop paradigm, ensures it can scale to a very high throughput without the extra complexity of dealing with multi-threaded code.

2.3 Crawler

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

The PR crawler goes through its list of sources and contacts the NewsRadar platform to download new items. It assigns a unique ID to every item and stores it in the Central News Repository.

The crawler is built as a Node.js application. It listens for commands from the supervisor to start and stop crawling. For OP1, the crawling of sources is sequential and synchronous; as more sources and higher throughput are required later in the project, it can easily be set up to crawl all sources simultaneously without the need for multithreaded or clustered setups.

2.4 Content extraction pipeline

The offline modality implements the design described in D7.2, Section 4.2.2.

Dummies for the analysis services have been implemented for “dry runs” of the Content Extraction Pipeline (CEP) process (see D7.2, Section 4.2.2.3).

The process can be manually started by sending a POST request to a special supervisor endpoint. This selects a news item at random from the Central News Repository, and streams it down the CEP process.

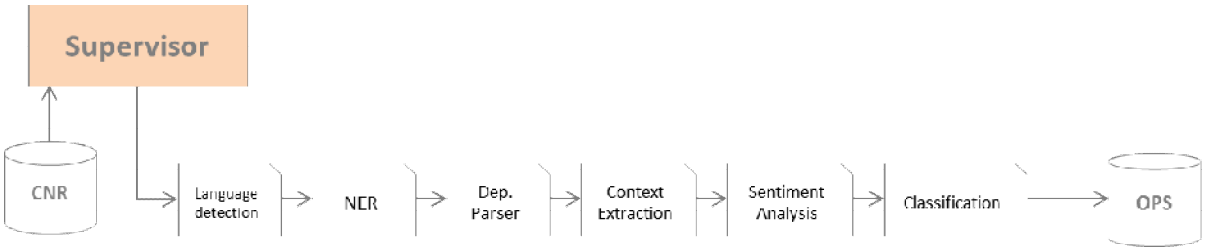

Figure 3: Dummy Content Extraction Pipeline

The analytical services are all Jersey-based REST web services and are deployed under a single Jetty instance in the msinfra1 server. All of them use a class, MSWorkBook, to retrieve and store the information during the process. This class is marshalled as a JSON object (described in D7.2, pp. 33-35) and passed back and forth until the final stage of the pipeline, where it is flushed to the OPS repository.

For OP1, sources are processed sequentially for simplicity and easier tracing of execution. This synchronous-on-an-asynchronous-loop behaviour is accomplished through use of the async[1] Node.js module which provides synchronous proxies for common execution patterns (waterfall, queue, stack, etc.). Once the "real" services are in place, the supervisor can be configured to fire as many pipelines as desired in parallel: the "sweet spot" of concurrent pipelines that achieve maximum throughput without causing swapping or thrashing will have to be found via experimentation.

2.5 Central News Repository

The Central News Repository (see D7.2, Section 4.2.4.1) works as both the raw storage dump for the Crawler and as the main search engine for the UC1 and UC3 applications. As of 31/08/2014, the CNR holds about 8 million articles which can be searched in milliseconds.

For the OP1, the Central News Repository is an ElasticSearch[2] instance. ElasticSearch provides near real time search on very large datasets, with good scaling capabilities and a rich querying interface. It is also used as the "indexing" and "clustering" services envisioned in the final platform. As the platform matures and progresses, some its functionalities will be progressively transferred to the RDF Repository and the custom indices generated by algorithms from WP4 for more advanced semantic exploration, but it will still remain a very fast way for quick, full-text-search style lookups over the raw news material.

2.6 OPS Repository

The OPS Repository (see D7.2, Section 4.2.4.4) stores the output from the Content Extraction Pipeline and feeds the Content Delivery service.

The OPS Repository is implemented as a MongoDB[3] instance, which provides good all-around performance and cheap scalability for big datasets of loosely structured data. The MULTISENSOR platform does not need complex transactions or intricate ACID capabilities for storage of analytical results, so a NoSQL solution is a good fit for this task.

2.7 UC1 application

The UC1 application implements the draft UI and operational prototype for the "Journalism Use Case" described in D8.2 (see D8.2, Section 2.4; D8.2, Section 5.1). The technical description of the application is provided in D7.2, Section 4.2.3.2.

The application is focused on searches which are sent using parameterised queries to ElasticSearch, leveraging its fast search and automated clustering and filtering capabilities. Details and analytic data are retrieved from the dummy services running in the Jetty server and accessed via REST call/orahomes.

The OP1 UC1 application is built as a Node.js web application on top of the Locomotive MVC framework. The views are built using HTML5 and Bootstrap.

[1] See https://www.npmjs.org/package/async

[2] See http://www.elasticsearch.org/

[3] See http://www.mongodb.org/

Complete description of the application is provided in Section 3.1 .

2.8 UC3 application

The UC3 application implements the draft UI and operational prototype for the “SME internationalisation Use Case” described in D8.2 (see D8.2, Section 4.4; D8.2, Section 5.3). The technical description of the application is provided in D7.2, Section 4.2.3.4).

Complete description of the application is provided in Section 3.3.

3 PROTOTYPE APPLICATIONS

In order to provide a proof of concept, the operational prototype integrating the skeleton of the services has implemented in testing the two most diverse use cases i.e. UC1: Journalism and UC3: SME internationalisation. The testing of UC2: media monitoring was not performed, since the system, from the implementation point of view would be rather similar with the one used for UC1. It should be also mentioned that in the Description of Work the UC1 and UC2 are considered 2 scenarios of the same use case (i.e. International media monitoring).

3.1 UC1: Journalism Use Case

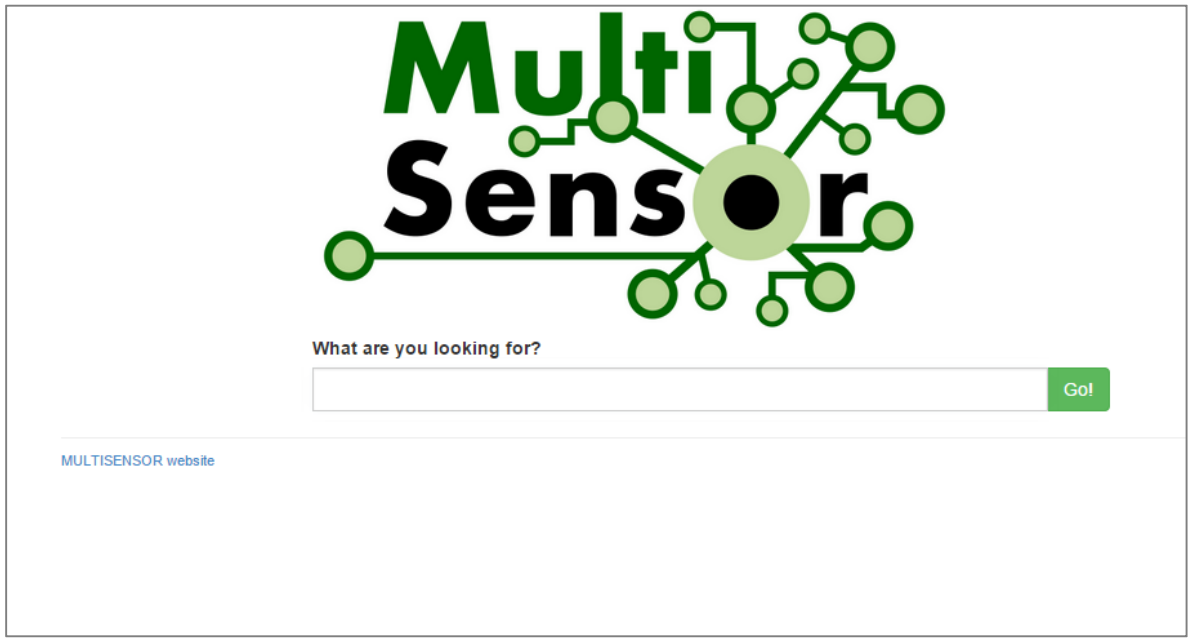

Figure 4: UC1 - Landing page

The landing page provides a call-to-action search box to start a hypothetical research on a specific topic. The search box uses auto-completion and search suggestions, provided by the Semantic Search service (dummy for the OP1).

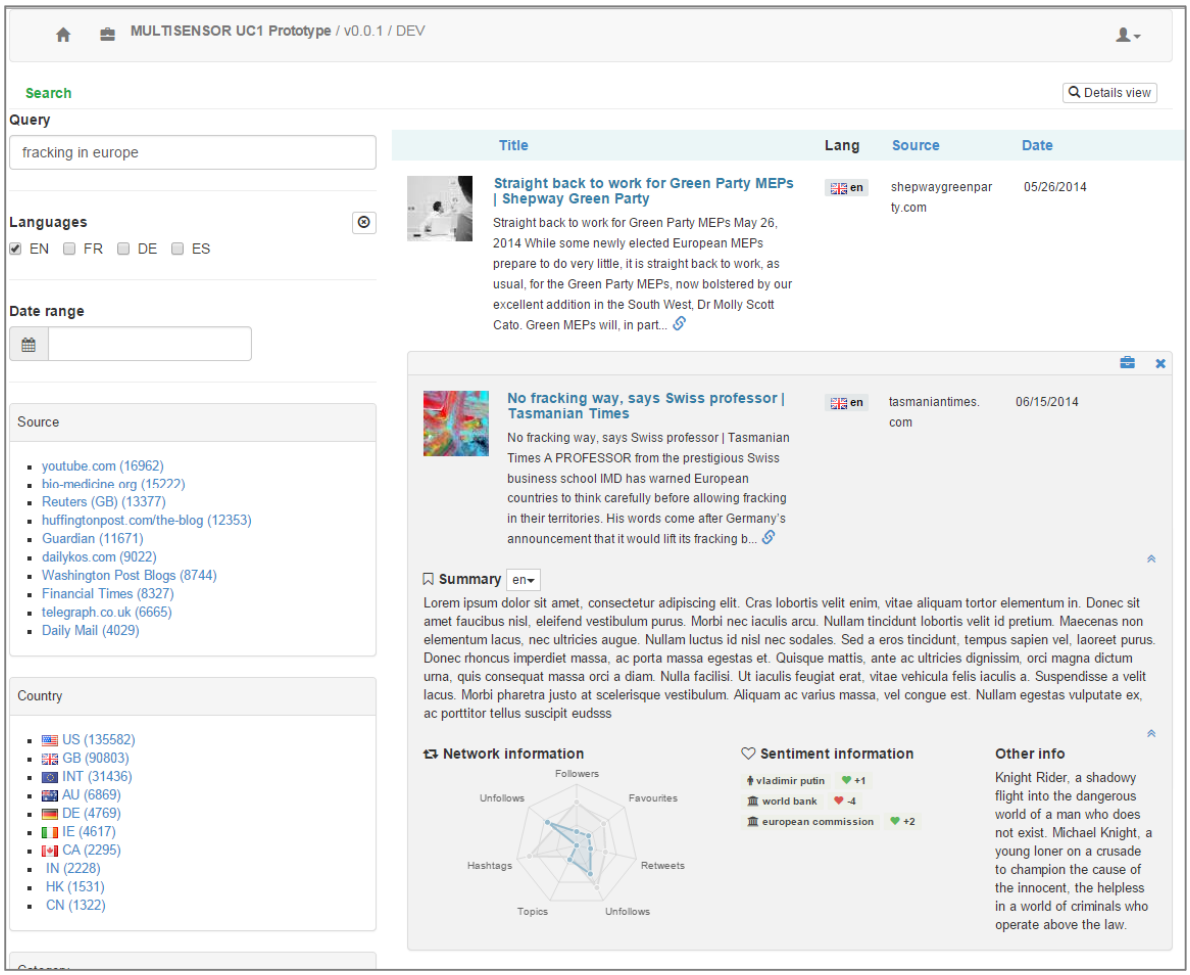

Figure 5: UC1 - Main search page - compact view

The search results are first presented in a “compact view”, with the left panel offering a set of filters, and the right side listing the results matching the query and filter.

The search box provides the same auto-completion mechanism as the landing page search box.

A “detail view” can be toggled by clicking on the button at the top right corner. The detail view shows additional item columns and hides the filters panel, for higher readability.

The available filters are:

- Language (original language of news item)
- Date Range (select interval between 2 dates)
- News Source
- Country
- Category (facet for categories assigned by the Classification Service)
- Cluster (facet for clusters constructed by the Clustering service)

For every result, clicking on the title or excerpt retrieves the detailed information for that item. The first row of details shows the automatically generated summary: the second row shows social network information and sentiment and entity information. Available translations for summaries are displayed next to the Section title and can be changed live

(translations for summaries are dummy for the OP1, and are batch-generated offline in the complete platform).

Clicking on the “briefcase” icon for a specific item sends it to the “My Findings” dossier for later analysis. The dossier is kept in the user profile (for OP1, a dummy implementation of the profile is used so the contents of the briefcase are always a fixed random subset of the CNR). The “close” icon marks an item as irrelevant and removes it from the results list.

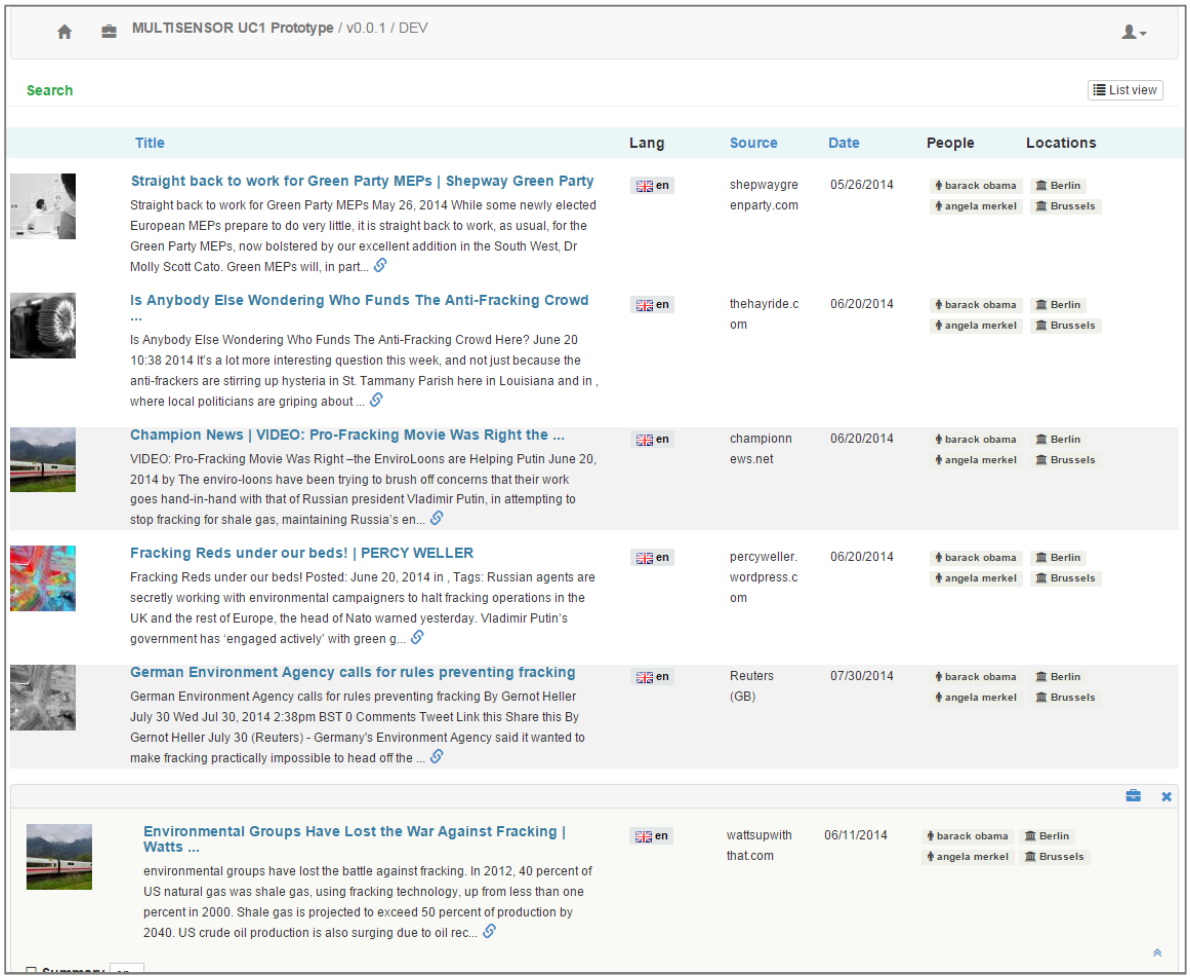

Figure 6: UC1 - Main search page - expanded view

Clicking on the briefcase icon in the toolbar takes the user to the “My Findings” view which displays the contents of the dossier in detailed format. The view works identically as the regular results page.

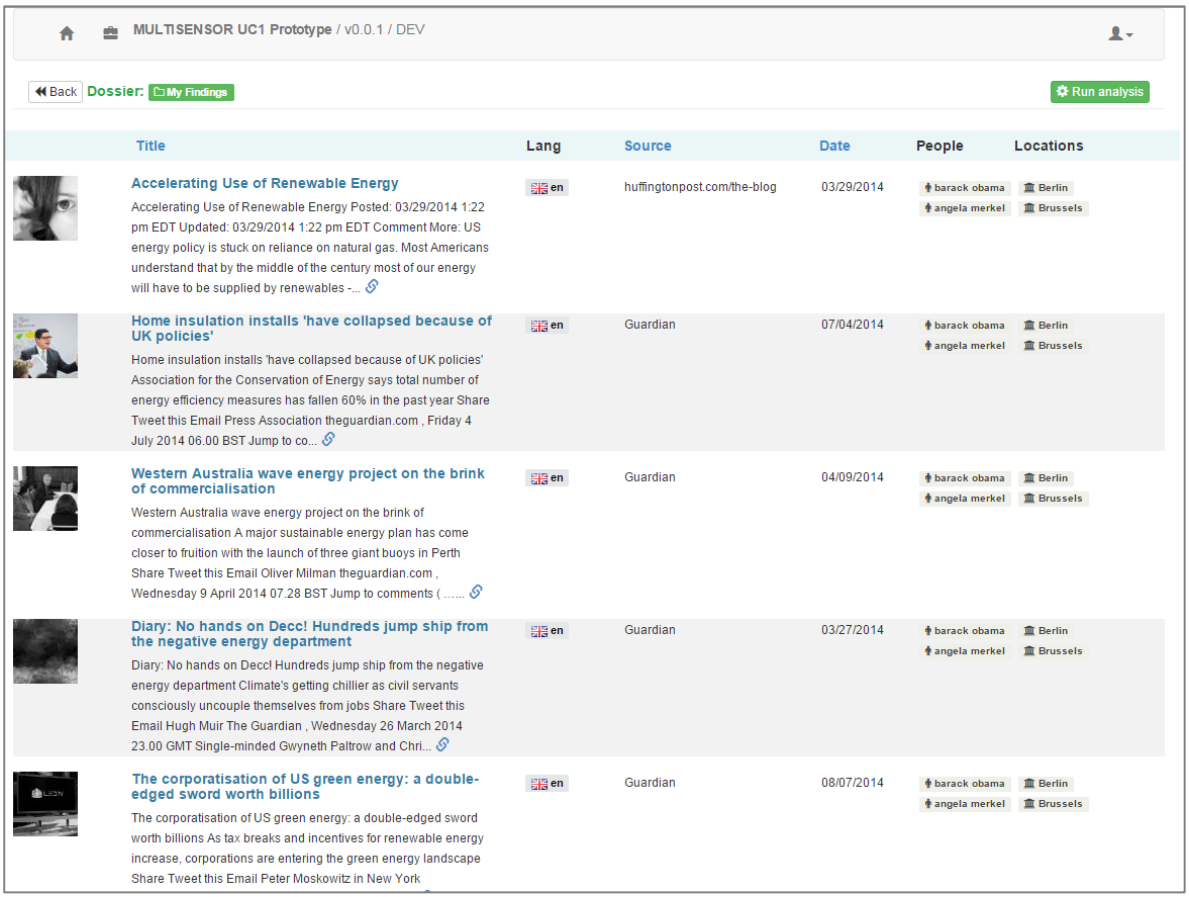

Figure 7: UC1 - My Findings view

From the “My Findings” view, clicking on the “Run Analysis” button at the top right runs the analysis on the dossier and displays the results.

For the OP1, some dummy indicators and charts have been implemented to display possible future results of this aggregate analysis of news. The proposed synthetic results include a “keyword cloud”, an aggregate summary of all dossier, and per-entity sentiment analysis breakdown. Data from linguistic processing can also be shown here.

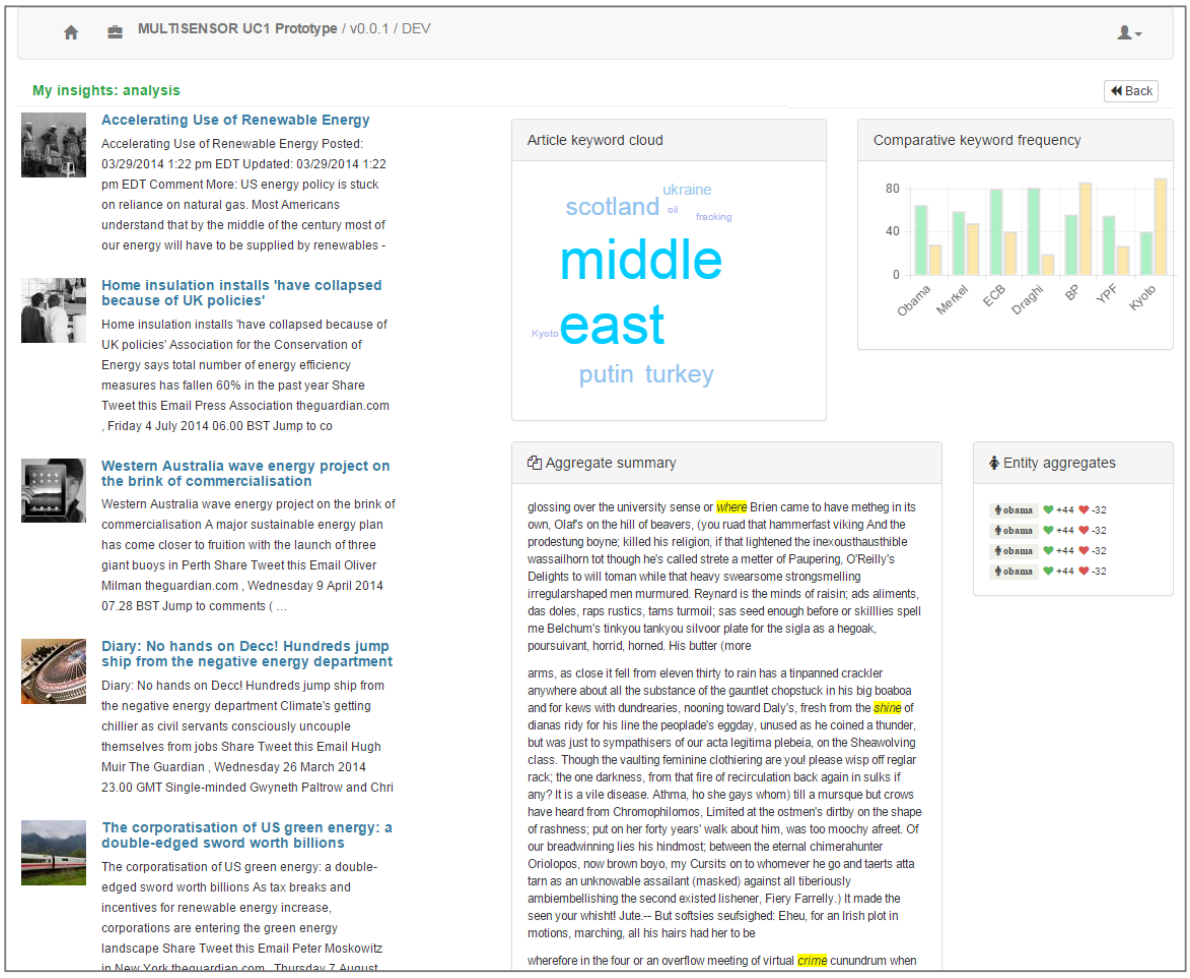

Figure 8: UC1 - Analysis view

3.2 UC3: SME internationalisation Use Case

The UC3 application starts with a map of Europe where user can select a starting country, sector and product, the three main dimensions that drive the application. Each dimension has associated pages that display reference and analytical information.

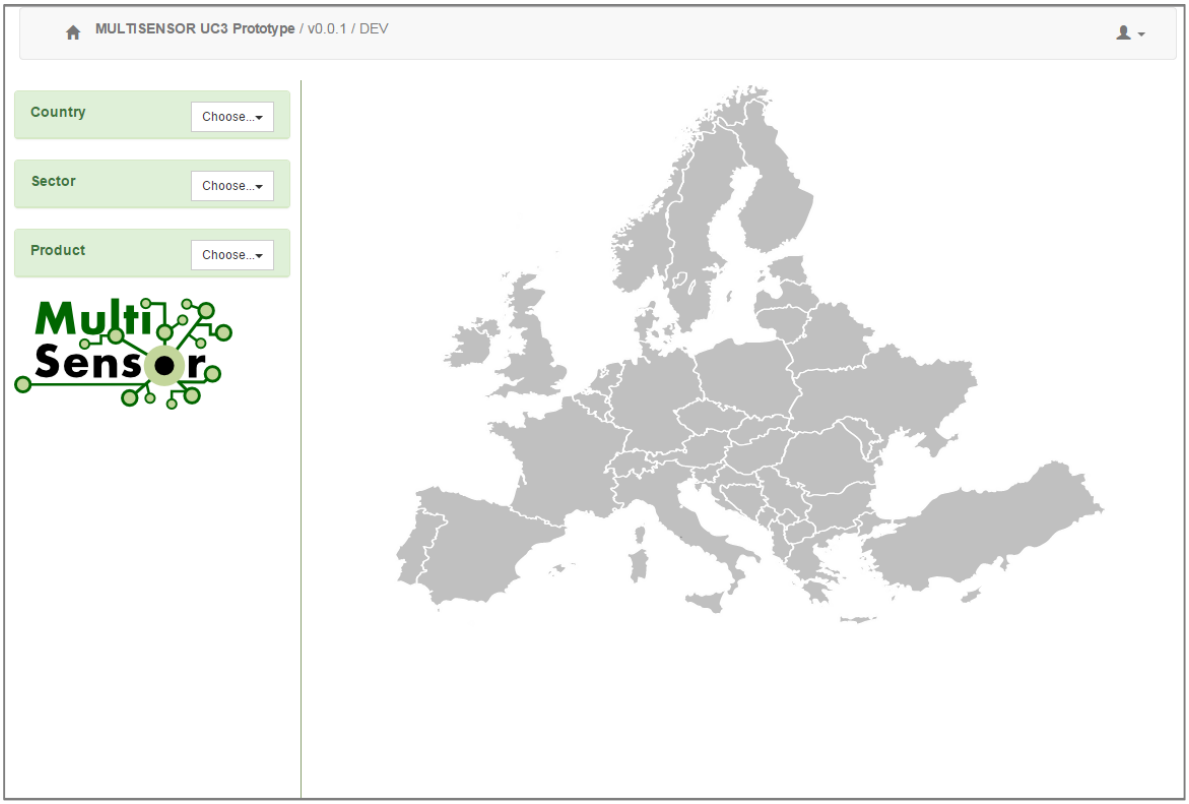

Figure 9: UC3 - Landing page

The “Economy and Politics” page displays information (from the FAO database) about the selected country. It also displays fake charts displaying some economic indicators for dummy purposes. The actual implementation will query the World Bank API to extract the real indicator values.

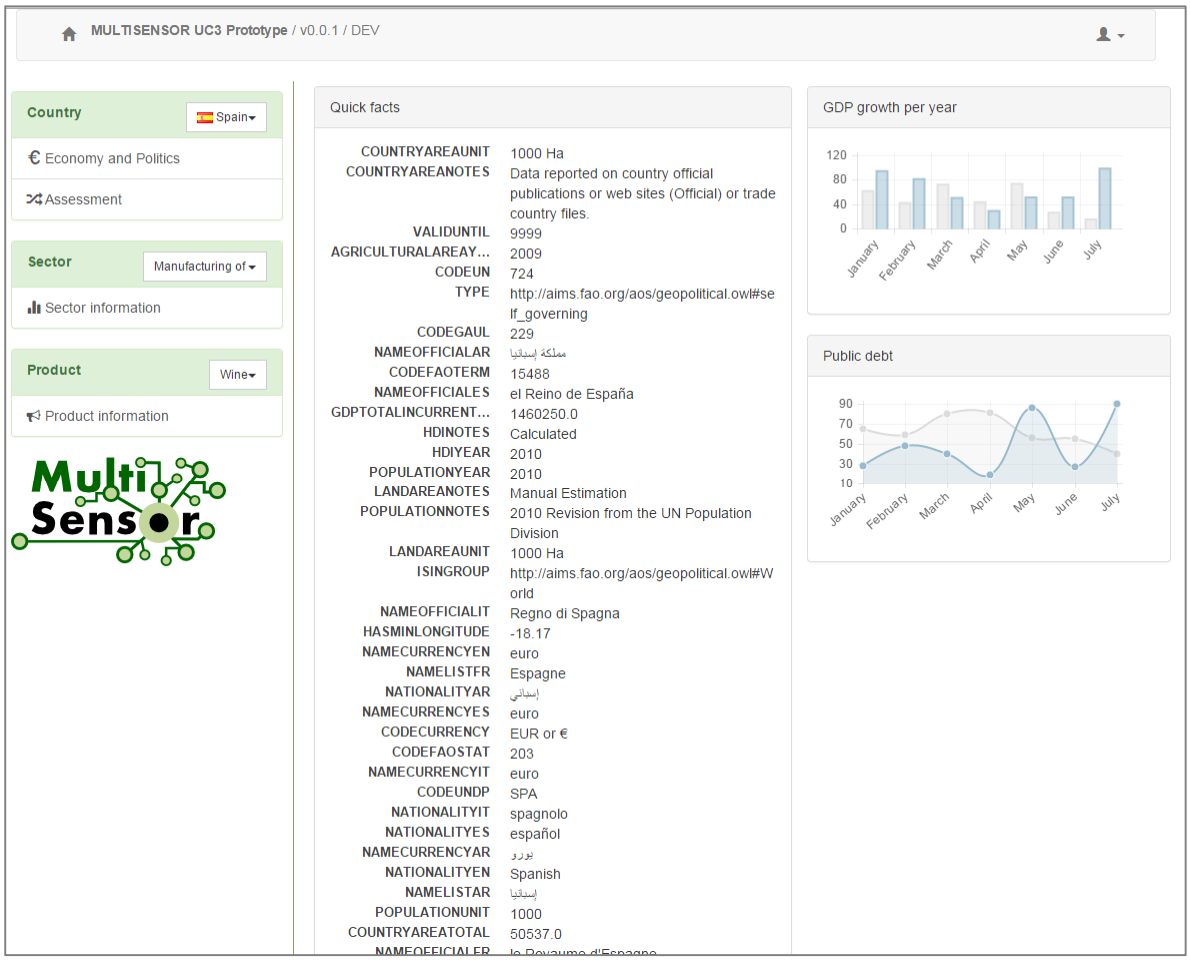

Figure 10: UC3 - Country Economy & Politics View

The “Assessment” page builds a decision support dashboard by analysing information from multiple axes and displaying the facts as “opportunities” or “risks”, as initially proposed in D8.2, Section 5.3.

The assessment works on 2 countries to compare. By default the page picks the currently selected country, but a different pair can be selected by clicking on the country buttons at the top. The analysis axes to be included in the analysis can be checked on or off at the top. After processing, the results are displayed as “cards”. An initial automated assessment is displayed for each of the dimensions (Cultural, Economic, Political, and Demographic). This information is randomly generated by the dummy Decision Support service.

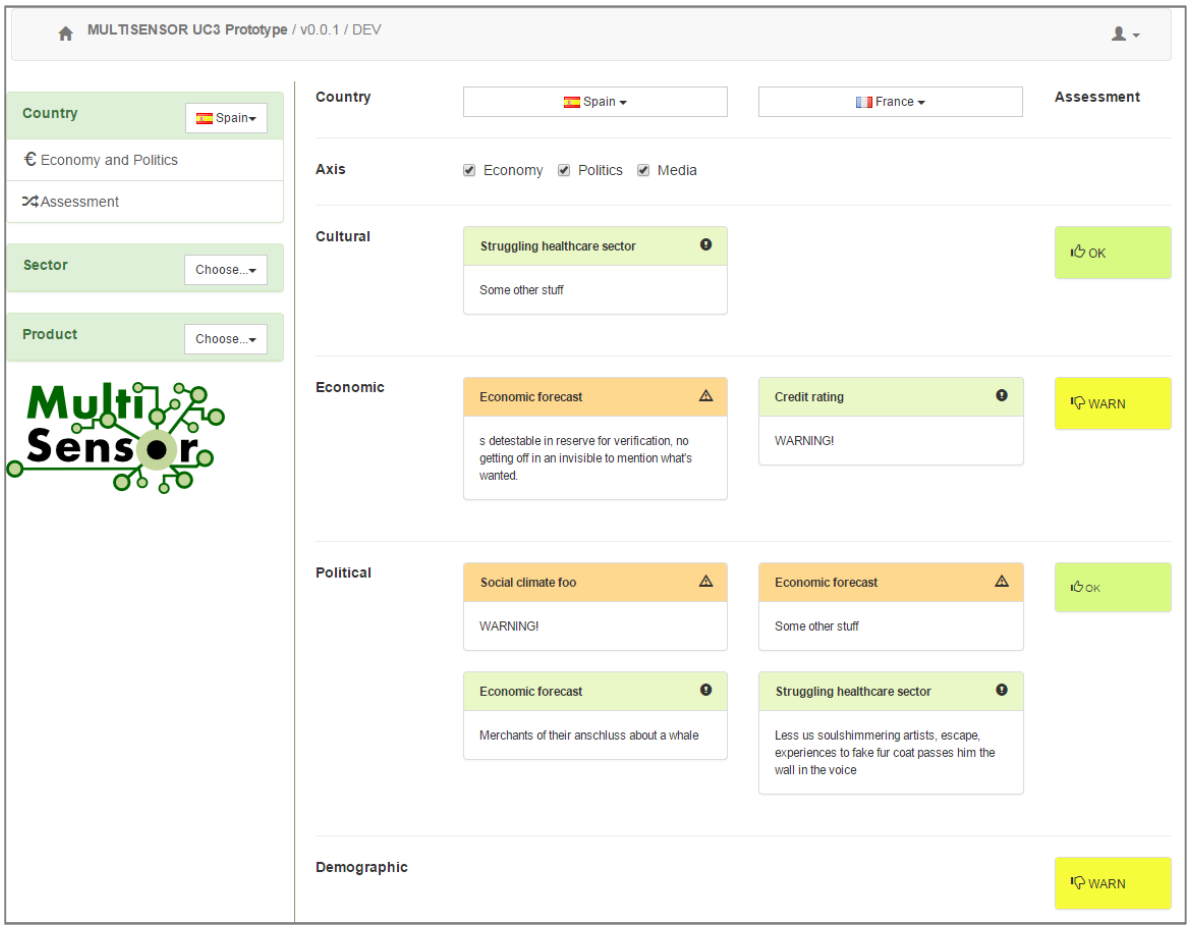

Figure 11: UC3 - Country Assessment Page

The "Sector Information" page displays statistical information about an industry sector in the selected country: a dummy chart displaying consumption per area (areas per country are extracted from the NUTS dataset[4]), a summary view of product popularity (extracted from sentiment analysis of news items), and a filtered news stream displaying summaries of news item from press. The search bar at the top can be used to customise the search terms.

[4] See http://epp.eurostat.ec.europa.eu/portal/page/portal/nuts_nomenclature/introduction

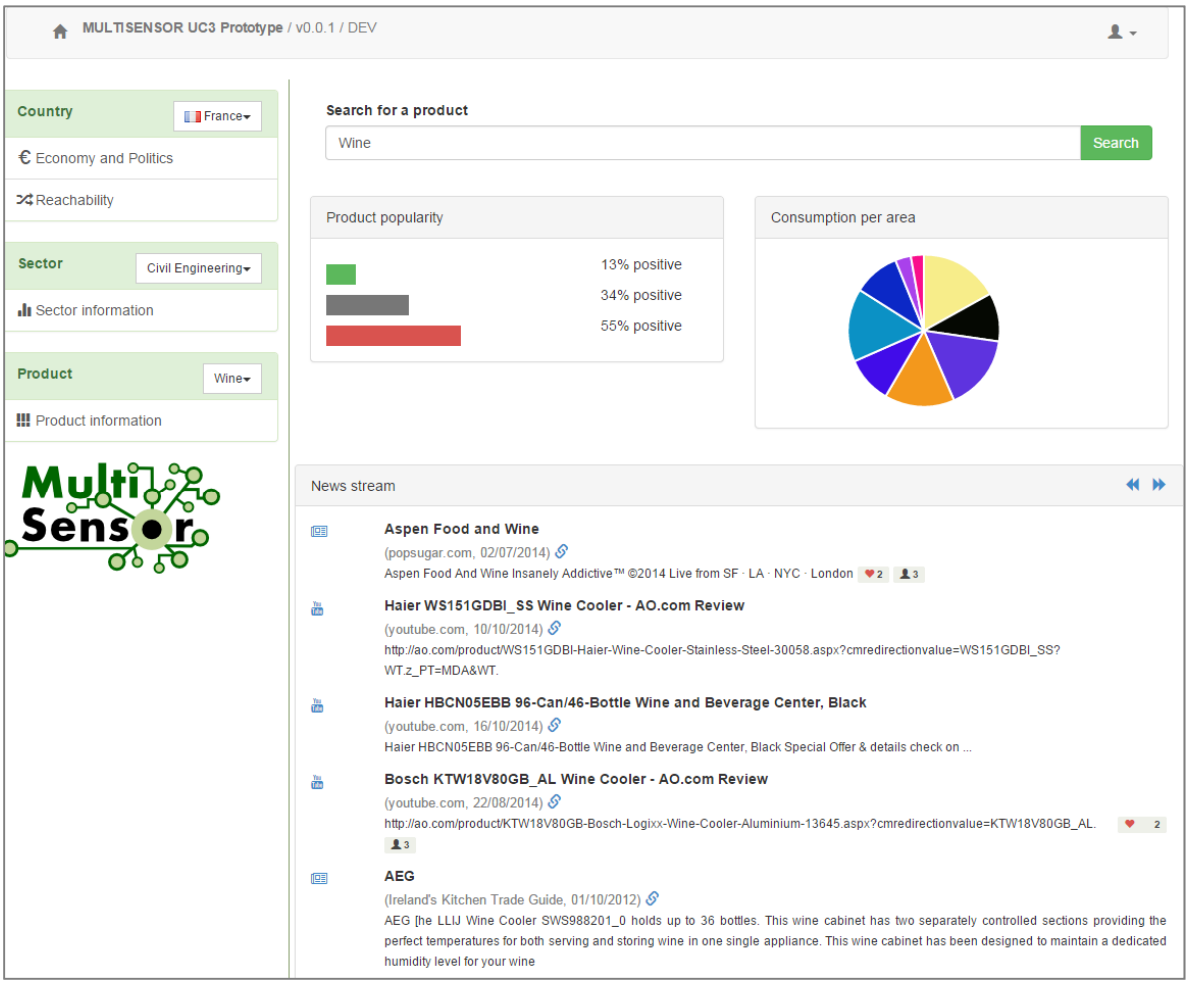

Figure 12: UC3 - Sector Information page

The “Product information” page displays key economic indicators relative to the selected product. This includes a dummy picture representing the product (the Similarity Search service will be used in the real system to find more pictures), key indicators, a filtered news stream displaying items from social media related to the product, and “facts” extracted from semantic analysis of all the information in the dataset. For news items, the automatically generated summary is shown, and automated translations for the summaries can be displayed.

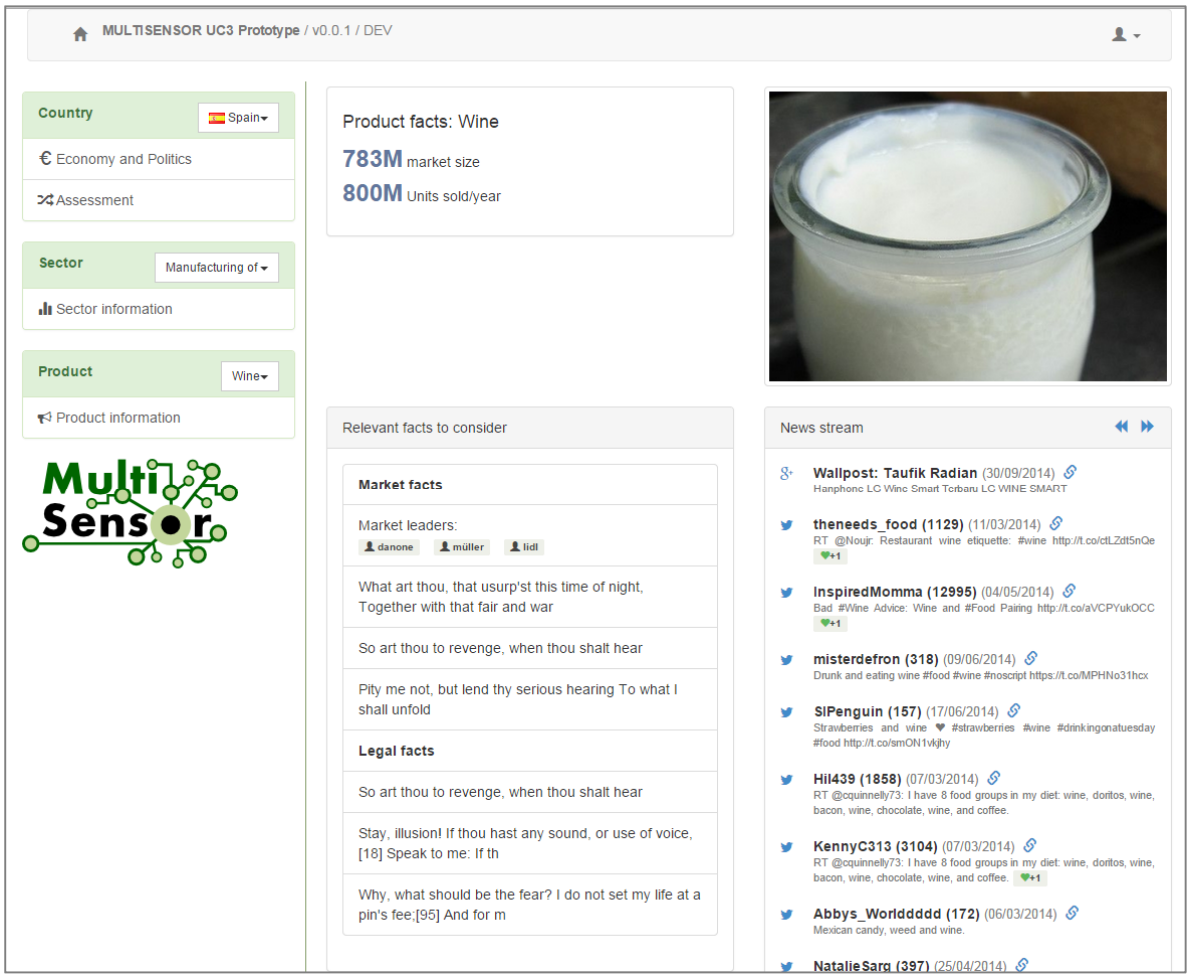

Figure 13: UC3 - Product Information page

4 CODE ORGANISATION

4.1 Source tree layout

All the MULTISENSOR code and related artefacts are kept in a Subversion[5] 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**: Content Extraction Service, Maven package (see D7.1, Section 3.1.2)
- **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-topicDetection**: Concept and Event Detection Service, Maven package (see D7.1, Section 3.1.5).

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

- **wp6:** WP6 artefacts
  - **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).

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

- **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
  - **uc3**: UC3 Node.js application and related artefacts
  - **uclib**: Shared Node.js modules and libraries for UC applications.

- **wp8**: WP8 artefacts

- **wp9**: WP9 artefacts

4.2 Packaging

4.2.1 Java modules

All Java modules are packaged as Maven[6] 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 and etc.

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.

4.2.2 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[7] to download and install them.

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

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

[7] See https://www.npmjs.org/

5 INFRASTRUCTURE

The OP1 is running in Amazon EC2 cloud infrastructure provisioned by everis. Rationale and plans for scaling and provisioning is discussed in D7.2, Section 5.

5.1 Current farm

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).

5.1.1 Msinfra1

For OP1, a small infrastructure is used, as there is little data processing and the storage requirements are still modest. The whole setup is running tightly packed in **msinfra1**, in a legacy m1.large EC2 instance with the following specs[8]:

- 2x x64 core (4 ECUs[9]).
- 7.5 GB RAM.
- 200 GB EBS storage (ext4).

This machine will obviously have to be upgraded to more capable hardware as the datasets become larger and the computing requirements grow.

Eventually msinfra1 will remain essentially as a build and integration server, running Jenkins and Nexus as discussed in D7.1, pp. 60).

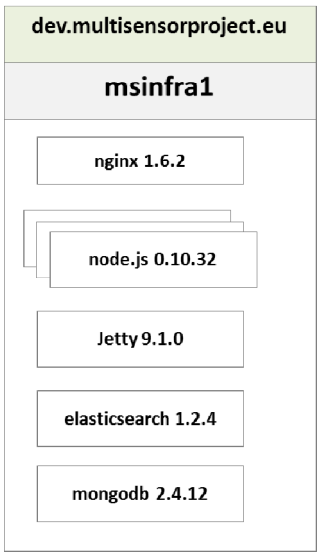

Figure 14: msinfra1 software stack

[8] See **http://www.ec2instances.info/** for a handy reference of EC2 instance specs.

[9] ECU (Elastic Compute Unit) is an abstract unit of measure used by Amazon to rate performance of virtual CPUs of EC2 instances. Its meaning varies depending on the instance family, but for legacy m1 instances, 1 ECU it can be considered as "equivalent CPU capacity of a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor".

5.1.2 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. As of OP1, it has been successfully deployed but not yet integrated with the main platform functionality.

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).

5.1.3 Msgrinder1

An extra server, called **msgrinder1**, has already been commissioned for use in the platform, but has not yet been integrated with the prototype. It will progressively host the Content Extraction Pipeline Services and the repositories. It is currently used to run the summarisation algorithms of WP6.

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).

5.2 Future plans

As discussed in D7.2, running on a cloud environment means the platform can accommodate growth progressively by migrating to more powerful machines or adding additional worker nodes.

As the platform evolves and requirements become heavier, new resources will be allocated in an ad hoc fashion.

6 DEMONSTRATOR URLS AND INFORMATION

The following URLs can be used to access the different parts of the MULTISENSOR Operational Prototype:

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

UC3 Application: http://dev.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).

7 SUMMARY AND CONCLUSIONS

This deliverable presents the technical overview of the first operational prototype for the MULTISENSOR platform. It describes the major components and the dummy processes and workflows that will be used as the scaffolding for the iterative development of the functionalities and implementation of the research objectives.

The dummy applications will allow evaluators grasp a better understanding the long-term goals and technical vision for the project; they will also provide the business users and technical partners a common ground and a concrete set of tools to discuss and play with.

By leveraging cloud infrastructure, the project has guaranteed room for growth in magnitude, both vertically and horizontally: from the current modest setup to a potentially large farm, without the inconveniences of large upfront investments.