Collaborative Project

LOD2 – Creating Knowledge out of Interlinked Data

Project Number: 257943 Start Date of Project: 01/09/2010 Duration: 48 months

Deliverable 8.1.1

Enterprise Requirements

Dissemination Level	Public
Due Date of Deliverable	Month 12, 31/08/2011
Actual Submission Date	[05/09/2011]
Work Package	WP 8, Use Case 2: LOD2 for Enterprise Data Web
Task	T 8.1
Type	Report
Approval Status	Approved
Version	1.0
Number of Pages	57
Filename	D8-1-1_Enterprise_Requirements.docx

Abstract:

This report describes the detailed specifications of the Enterprise use case following the preliminary one provided in the overall functional requirements of LOD2 in D1.1.

History

Project funded by the European Commission within the Seventh Framework Programme (2007 – 2013)

Version	Date	Reason	Revised by
0.0	13/05/2011	Creation	Amar-Djalil MEZAOUR
0.5	26/08/2011	First complete release	Amar-Djalil MEZAOUR
0.9	29/08/2011	Peer review	Tassilo PELLEGRINI
1.0	01/09/2011	Consolidation of peer review comments	Amar-Djalil MEZAOUR

Author List

Organisation	Name	Contact Information
EXALEAD	Amar-Djalil MEZAOUR	Amardjalil.mezaour@3ds.com
Open Link software	Hugh WILLIAMS	hwilliams@openlinksw.com
Semantic Web Company	Thomas SCHANDL	t.schandl@semantic-web.at
TenForce	Johan De SMEDT	johan.de-smedt@tenforce.com
TenForce	Bastian DEBLIECK	bastiaan.deblieck@tenforce.com

Executive Summary

D8.1.1 specifies the detailed functionalities and requirements of the enterprise use case of LOD2 project.

In this deliverable we introduce the application domain chosen for the enterprise use case. A scenario and corresponding requirements are provided. We define the functionalities and features to implement for the use case application. A preliminary technical architecture overview is also provided.

We will describe the used data model framework and the corresponding mapping strategies to RDF vocabularies.

Abbreviations and Acronyms

LOD	Linked Open Data
POC	Proof of Concept

Table of Contents

List of Figures

List of Tables

1.Introduction

1.1 Background

This document is written in the context of the activities of WP8 of the LOD2 project funded by the European Commission. It is a deliverable about the functional requirements of the application to be developed in WP8 for demonstrating the open data possibilities in an enterprise environment.

1.2 Purpose

Integrating open data, available on the web or in the enterprise cloud, in a business corporate application is one of the challenges that LOD2 project is targeting. WP8 activities aim at providing a concrete use case application for demonstrating the usefulness of integrating open data in existing enterprise workflows. Therefore, the intended application to implement for WP8 will be developed for the purpose of providing a proof of concept on the features that open data and the LOD paradigm could generate within an enterprise IT system. Industrialization and possible commercial exploitation of the previous POC can be envisaged if a significant impact can be shown from the tests and trials in real situations.

1.3 Goals

The aim of this document is to provide a vision about the requirements needed by the user community of the enterprise application we would like to develop in WP8. The specification task developed in this document is aiming at providing clear functionalities cartography of the application. This cartography is designed as a mapping between user requirements and available features in the components of the LOD2 stack. It is then straightforward to show the rate of the requirements covered by the existing LOD2 components and the effort to develop the features not already covered.

In this document, we present in details the enterprise application for demonstrating the integration of LOD standards and data in corporate workflow. We also introduce the cartography of the available components in LOD2 stack. We define the functional requirements for the enterprise use case and a mapping from the LOD2 stack components to these requirements.

This deliverable is organised as follows:

- In section 2, we introduce the enterprise use case application: “Active Hiring”. We also present an implementation architecture overview of the application.
- In section 3, we provide a detailed functional map of the LOD2 stack and components.
- In section 4, we show the mapping from the LOD2 components and the functional requirements.

2.Enterprise Use Case: “Active Hiring”

Hiring people for open job positions can be viewed as a workflow composed by different processes, each covering a set of recruiting. Thus, the workflow normally starts from the company that is hiring with a position opening. This opening is notified to the labour market through several channels: company web site (usually in jobs section), job boards (like monster for example)… A given number of applicants respond to the request. A preliminary selection step is done on the applicants by matching their skills to the job requirements and by applying, if necessary, tests procedures to assess their profiles. A scoring mark is computed for each applicant and a ranking is output at the end allowing a first pool of applicants to be preselected for further interview. Interviews are then achieved where each party (the recruiter and the candidate) can acquire additional knowledge about the hiring process, expectations, etc. Once a candidate is selected, the on boarding step starts by tailoring a labour contract, eventually adapting the job description. Then information about the candidate is added to the different HR systems of the recruiter and finally the contract is signed and becomes effective. With that step, the legal hiring process is completed.

Different implementations of the previous steps can be envisaged; each manipulating dedicated data models and custom dataflows. In fact, HR IT actors offer several custom and specific applications covering one or many recruiting steps. Unfortunately, making these applications interoperable in a global integrated environment is a difficult task that requires the development of wrappers and intermediate data models.

On the other hand, the market HR applications rely on formats that are poorly standardised and harmonized. Typically, the processed information is often expressed in free text with few or no references to standardized terminologies and vocabularies. This leads to ambiguity, unclear profiles vs. position requests understanding and inefficient recruiting dataflow management. For example, resumes’ entries, like experience and educational background, are expressed in free text that could usually be mapped to structured terminology.

Therefore, hiring candidates to a given position could become time and resource consuming and may require frequent manual human intervention. Starting from these observations, the LOD2 consortium is targeting to support the hiring process by providing a semantic dataflow for supporting the hiring process. This is achieved by integrating open standards and vocabularies from the LOD cloud with available semantic tools for supporting this integration process.

2.1 Overview

The application we intend to implement for WP8 consists of providing a semantic workflow to support the hiring and recruiting process. In the common requirements introduced in deliverable D1.1, we presented a generic workflow for the enterprise use case. This workflow shows the different processes involved in the implementation of an enterprise use case starting from the data acquisition until the final data delivery to the end users (see Figure 1: Enterprise use case workflow in page 12).

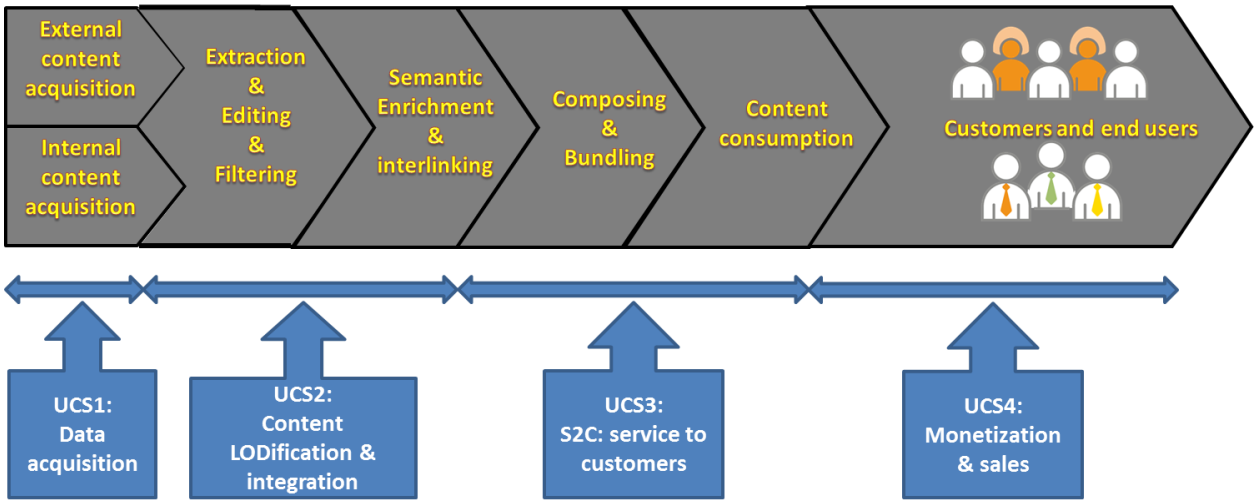


Figure 1: Enterprise use case workflow

The idea from this diagram is to build a semantic processing workflow from the different LOD2 consortium partners' components and resources and apply them on HR data. We strongly believe that the semantic possibilities of open data can enhance significantly the efficiency of the hiring process and thus the associated applications. This semantic workflow provides advanced semantic features like transformation of data to RDF-like formats, mapping data instances to standardized terminologies and vocabularies, extraction of concepts, interlinking data and concepts in a straightforward way. Better HR data retrieval can be expected and advanced queries also.

Another important idea of implementing LOD in an HR enterprise use case lies on the possibility to demonstrate the benefit of semantic data integration. In fact enterprise IT systems operate usually on a set of internal data that do not benefit from the available potential external sources in the web for example. Thus, an HR system can refer to data available in external sources (like web sites or specific partners' repositories and resources). For example, during the contract tailoring step, the hiring system can refer and point to legal data information about labour regulations and laws, social security information, etc. In this situation, our "Active hiring" use case will demonstrate the benefits of having interlinking tools and mechanisms for mapping candidate contract data to local regulations. This will be one of the intended data integration possibilities that we would like to show.

Remark: Note that the objective is not to develop a new integrated HR application for the full recruiting process but to provide tools and methodologies to facilitate the management of the hiring process and to provide rich semantic information about HR data.

In the following sections, we explain how this workflow is mapped to the "Active Hiring" use case by defining the associated functional requirements.

2.2 Stakeholders

- Exalead: corporate search software editor. Exalead is leading the WP8 activities for developing the business use case.

- NUI Galway: National University of Ireland, Galway (NUI, Galway) - Digital Enterprise Research Institute (DERI) is one of the main actors in research and development of semantic technologies in the world.
- TenForce: TenForce is a Belgian software company specialized in knowledge management combined with an in-depth expertise in emerging technologies.
- SWCG: Semantic Web Company (SWC), based in Vienna, is an SME, which offers consulting services in the fields of semantic web technologies, knowledge management systems and social software since 2005.
- WKD: Wolters Kluwer Deutschland GmbH (WKD) is part of Wolters Kluwer n.v., a leading global information services and publishing company.
- FUB: The Web-based Systems Group at Freie Universität Berlin explores technical and economic questions concerning the development of global, decentralized information environments. Its current focus lies on the publication and interlinking of structured data on the Web using Semantic Web technologies.
- OGL: OpenLink Software is an acclaimed technology innovator and leading vendor of industry standards compliant data access, integration, and management technology.

2.3 Roles

In this section, we present the different role models that are involved in the “Active Hiring” application. In the context of this document, a role model is a person entity that interacts with our “Active Hiring” application. Note that, the interaction with the “Active Hiring” application can be direct or indirect through third-party software component.

We classified our role models in three categories on the basis of their mode of interaction with the application: end users, advanced users (like administrators) and developers of third party software components using the application. The following diagram shows the interaction mode of each of the previous role models:

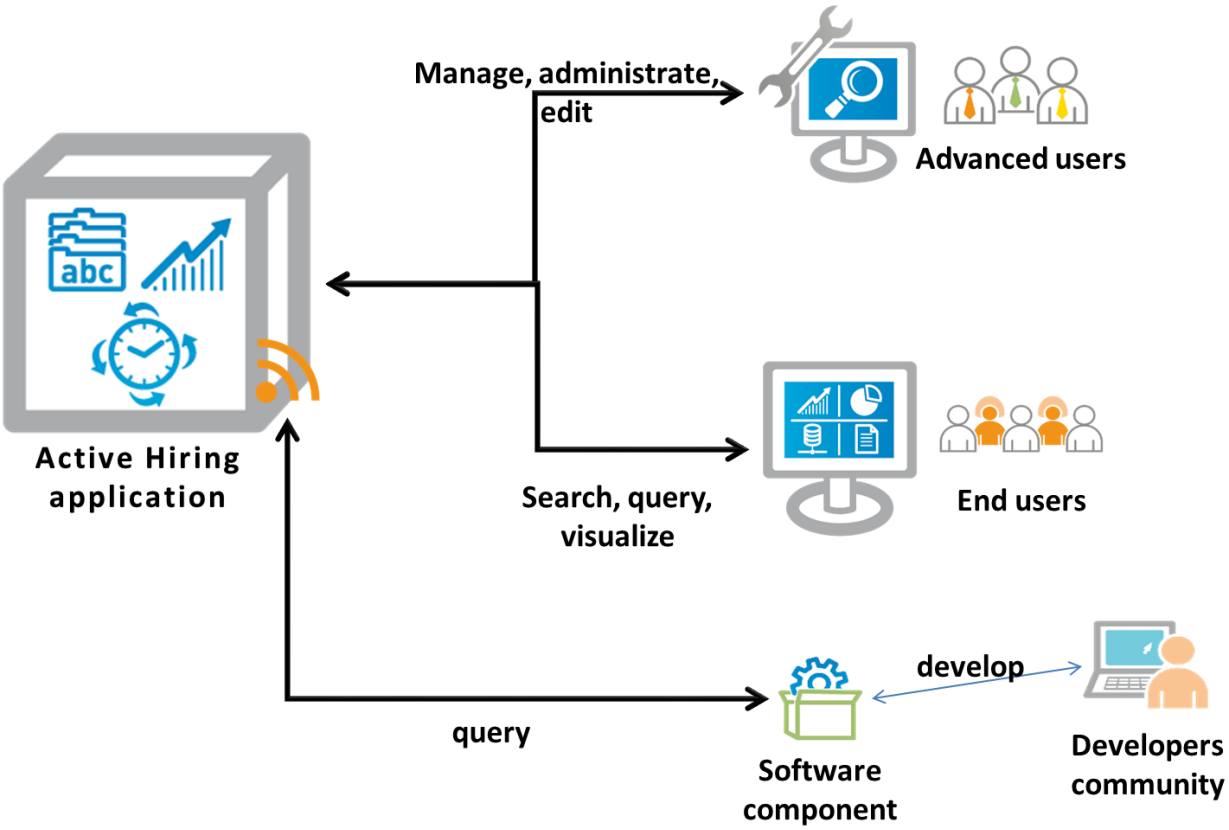


Figure 2: Active Hiring application role models

2.3.1 End users role models

ID	Denomination	Description
EU-1	Employee of HR department or HR responsible	Represents the recruiting party in the use case. She defines the recruiting strategy of a company by posting and notifying job offers, assessing candidates, tailoring contracts, etc.
EU-2	Head-hunter	Looks for appropriate candidates to companies
EU-3	A candidate (Anyone looking for a job)	Searches for a job meeting her qualifications

Table 1: End users role models

2.3.2 Advanced users role models

ID	Denomination	Description
AU-1	Expert in data acquisition	Defines, implements data acquisition procedures from various sources
AU-2	Expert in HR and recruiting	Suggests data sources, edits vocabularies and writes position descriptions…

AU-3	Expert in information extraction and semantic vocabularies	Defines extraction algorithms, maps vocabularies, develops semantic components, implements interlinking procedures …
AU-4	Expert in search	Defines the indexing/search implementations, querying modes…
AU-5	Product manager	Application promotion and marketing, definition of new functionalities and interfaces, definition of business cases

Table 2: Advanced users role models

2.3.3 Developers role models

ID	Denomination	Description
DU-1	Developers of services on top of “Active Hiring” application	Writes software components or services using exposed interfaces/services/endpoints of the application

Table 3: Developers role models

2.4 Scenarios

For the "Active Hiring" application, we identified several potential users from the previous role models that could use the platform in different situations. We present in this section some use scenarios from each category of users.

2.4.1 End users scenarios

2.4.1.1 EU-1 scenario: a recruiting company publishing and notifying a position opening

TheJobCompany is a company that is growing very fast and so recruits intensively. The HR manager Carla is interested in the "Active Hiring" platform in order to edit and publish the job openings available in the company. When logging into the platform, she selects the dedicated tab and gets the interface to enter the job opening description. In a similar way as the resume editing (see section 2.4.1.4), the interface proposes a set of fields to enter the different parts of a job position.

As Carla starts to fill in the fields, the system will assist her with an auto-complete functionality which suggests input values for strings she enters based on various thesauri for industry sectors, skills, etc. When she e.g. starts to enter the characters 'Jav' in the 'Required skills' field, the system can suggest Java, JavaScript, JSP, J2EE and the like. The suggestion of concepts like J2EE and JSP that do not explicitly have Java in their name is possible because in a taxonomy these concepts can be associated with several synonyms, abbreviations and misspellings. In a thesaurus they can also be related to each other. Because in the thesaurus JSP is associated with its synonym 'Java Server Pages' and because J2EE is related to Java the system can suggest them which helps to exhaustively describe a skill set and avoids inconsistent naming of skills.

When Carla selects the appropriate skills and other items from the system's suggestions, she also gets the benefit of additional semantic features: skills will be linked to their equivalent DBpedia resources, locations will be marked up with their appropriate geo coordinates, corresponding industry sectors can be automatically applied as classification of the job description, the job offer can be linked with relevant resources like related legal provisions, social benefits rules etc. These semantic annotations will also help in better indexing the job description.

2.4.1.2 EU-2 scenario: a recruiter searching for a candidate

Bob is an HR manager of MyITCompany and is responsible for hiring new developers in the R&D team. Bob has no official job opportunity opened but is interested in harvesting promising talents to discuss possible recruitment in case of agreement. Thus, Bob is interested in head hunting people having strong skills in SEO, JavaScript, Ajax and MVC frameworks. In particular, he is specially targeting people having worked for MyITCompany's competitors in the last 5 years in a specific region…

As usual, Bob logs into his favourite job board, monster for example, and starts browsing profiles. He first uses the keyword search by entering a list of relevant keyword queries. The job board returns too many potential candidates and none of them are relevant. He explained this by the fact that he could not express exactly what he has in mind using simple keyword queries. He then decided to browse the profiles by the provided categories, by region for example, and starts to read and validate manually the resulting CVs, which turn out to be thousands. He stopped very quickly as soon as he understood that the task could not be managed efficiently this way.

When talking about his difficulties with one of his colleagues in R&D who heard about LOD2 project, this colleague later suggested to Bob to use the "Active Hiring" application with its advanced search and querying features.

Bob now logs into the system and starts browsing the resumes catalogue using keyword queries. His first tentative query is not so different from the one he used in his favourite job board. The result interface provides interesting widgets that can help him specify his initial requirements.

From the map widget, he starts by delimiting the region that he is focusing on. On the remaining results, he executes a set of operations to filter resumes having annotations corresponding to a set of selected companies appearing in the profile of the candidate. Having a thesaurus of companies active in his industry sector the system can help him create a complete query: Once Bob enters or selects from auto-complete suggestions some companies that he wants to track the system can go one step further and suggest their parent companies, subsidiaries or other similar companies from their section. In the same way the system supports Bob while specifying required skills and other data items.

In many cases Bob can use the system's widgets and query assistants to create the queries he needs, for very specific search requirements, SPARQL is available to create custom semantic queries that Bob needs. As he is not familiar with SPARQL he asks his colleague to create SPARQL queries. Bob is very satisfied with the accuracy of the returned results. He asked then the R&D team to develop a connector that uses the "Active Hiring" platform services endpoints to feed an internal repository of potential promising talents to MyITCompany that can continually monitor selected job portals.

2.4.1.3 EU-3 scenario 1: a candidate searching for a job

Linda is a J2EE software architect that is looking for new job opportunities in the Financial IT consulting industry. She has in mind a set of criteria for her ideal job position. So, Linda wants a job meeting the following criteria:

- The hiring company is in the domain of IT for Financial markets
- The hiring company should have more than 500 employees and be traded
- The development requirements should meet her skills in development (J2EE, WebSphere, Hibernate and Spring framework)
- The hiring company should be reachable from her home in 30 minutes using public transport.

To express her requirements, she uses the query interface of the LOD2 "Active Hiring" job portal. She uses a keyword query field to express her complex query. This query field proposes advanced input features to help the user formulating her query. It queries the portal index to propose an automatic completion functionality based on the structure (schema) and the values of the indexed data. This functionality allows Linda to build advanced structured queries in the form of a logical combination of `<field:value>` queries.

Since Linda is a user with a significant expertise in manipulating structured and unstructured sources of information, she switches from the keyword queries to a SPARQL like interface for querying the portal in depth.

On the other hand, Linda is aware that LOD2 "active hiring" job portal is frequently visited by head-hunters that are looking for candidates. In order to maximize the chances of having an interview, she decided to post her resume on the portal. She therefore uses advanced semantic functionalities of the system to optimize her profile description before indexing. From a classical resume interface, Linda fills in the different fields (name, education background, previous experience, etc…) and at each step the system validates her input by recommending annotations and standard vocabularies enrichment. For example,

when Linda inputs her address, the system will suggest a geocoordinate that corresponds to her location. When she inputs her education background, the system will propose to add a specific category from a standard vocabulary of education levels to annotate it. When she enters or selects skills, the system can suggest further skills and competencies that are related to them, e.g. after entering WebSphere, the system can ask whether she has experience with *WebSphere Application Server, WebSphere Portal Server*, etc.

Instead of entering all her infos from scratch, she has also the possibility to upload one of her old CVs from which the system extracts the content and tries to find skills, companies and locations that it knows from a thesaurus in order to partially fill the resume form automatically.

2.4.1.4 EU-3 scenario 2: a candidate editing its resume

Paula is a tour guide and she is looking for job opportunities in US. First, she starts to create and edit a new CV containing her experience background and professional profile. She logs into the "Active Hiring" platform and selects the tab form "Resume Edit". This tab presents a basic HTML form interface for inputting resume data. She fills in the different parts of the form with her personal data and she gets some nice auto-completion features for filling common fields like location, industry domains and sectors, education levels, etc.

Meanwhile and based on Paula's input, the system computes progressively various semantic annotations suggestions to optimize Paula's resume. These annotations are submitted for validation to Paula. For example, the system proposes to disambiguate the locations (Tripoli: did you mean "Tripoli, Libya" or "Tripoli, Lebanon" or "Tripoli, Greece"). According to the selected choice the system adds the corresponding geo-coordinates. Vocabulary harmonization is also another feature that the system will provide to delimiter the free input text with standard concepts from domain taxonomies and terminologies. For example, educational level will be replaced by an entry from the ISCED[1] education terminology.

Paula is also much concerned with the optimisation of her resume to maximise the chance of having it viewed. She is interested with the feature of linking relevant concepts in her resume with online resources. For example, linking companies' names to their websites or relevant DBpedia concepts and categories will increase the understanding and findability of the resume.

Once the resume is completed, Paula would like to export it a standard format for publishing it into different platforms. She exports it in an XML format (HR-XML for example or Europass) and gets also an RDF file containing the semantic annotations and resources of her resume.

2.4.1.5 EU-3 scenario 3: an expert or candidate requesting legal support in Hiring context

Erik gives expert advice to contractors on legal and social issues for job-seekers in a globalised and multidisciplinary job market. To do this in a cost efficient way, he uses sector and country specific expert systems which provide him access to legislation and rulings that match the situation of his clients. A new German client Angela (a specialist in ecological urban planning) found a job opportunity in Brazil for a French company. She wants to know the available options to organize social security in the most effective way taking into account the conditions of the contract. Angela has completed 3.5 years of a 4 year contract with her current German company. The French/Brazilian contract is for a 2 year project with a possible 3 year extension. She must consider the options of moving with her family to Brazil. Erik uses his expert system to describe her career history and find relevant legislation, cases and articles in Germany (France and Brazil). With this information he prepares a dossier for Angela to discuss her options. This process could be automated and efficiently reproduced using the advanced querying interface on legal provisions.

[1] International Standard Classification of Education

This is enabled by having access to different legal provision resources that are fully tagged and enriched with appropriate semantic annotations.

2.4.2 Advanced users scenarios

2.4.2.1 AU-1 scenario: Data acquisition expert feeding the platform

Djalil is an expert in data acquisition. He is in charge of defining the acquisition procedures to feed the "Active Hiring" platform with the necessary data: resumes, job descriptions. To do so, he logs into the "Active Hiring" platform with special credentials that give him access to an admin interface where he can run his job. In this administration interface, he can specify the URLs of the sources that can be used to feed the platform index and RDF store.

Depending on how identified relevant sources expose data and depending also on how to reach it, the extraction process will require different more or less complex tools. For basic and simple acquisition of data, i.e. not requiring complex extraction features, the system will propose a basic interface for defining extraction rules. Such an interface contains fields where to specify the URLs of sources, the crawling rules to reach the relevant data and possibly simple extraction/filtering rules. For more complicated cases, Djalil will use advanced frameworks for scraping data. These frameworks will combine the use of XPath/XQuery queries with crawling rules to get the targeted data. Advanced scraping frameworks are available and under open source license like WebHarvest, ScraperWiki, Scrapy, etc… Djalil will rely on some of these frameworks to achieve complex scraping tasks.

Based on the previous tools and frameworks, Djalil will define pipelines for extracting, typing and streaming data into the RDF stores and text index of the "Active Hiring" platform. The main goal is to get resumes and job position descriptions from the web and have them structured, processed and turned into valuable RDF data.

2.4.2.2 AU-2 scenario: HR expert integrating HR resources

Bastian is an HR expert who has a set of HR resources to integrate in the "Active Hiring" platform. This set consists of domain specific resources and taxonomies that are used to support the semantic annotation and extraction process. To do so, Bastian uses the administrator interface to have access to an editing tool for taxonomies. With this tool, he can edit his resources; align concepts from different taxonomies and create links between them. Bastian is also interested in enriching his concepts with concepts coming from the web and more specifically from the LOD cloud via authority sources (DBpedia for example). Finally, Bastian uses different resources and taxonomies. Before he integrates them in the platform, Bastian is interested in checking their consistency and therefore uses the validation module with ontologies available in the "Active Hiring" platform.

2.4.2.3 AU-3 scenario: Semantic expert tuning the semantic features of the platform

Claudia is an expert in semantic and NLP technologies. She is in charge of defining the semantic extraction procedures of the "Active Hiring" platform. The interaction of Claudia with the platform consists of integrating linguistic resources into the system to be used by different LOD2 stack components. In a similar way as Bastian, Claudia requires an editing and visualisation tool to manage her resources and to define RDF mappings between concepts. On the other side, Claudia logs into the administration interface of the platform to build the NLP extraction process of the data been processed during the indexing process. She requires a pipeline editor to compose the different steps of the extraction.

2.4.2.4 AU-4 scenario: Search expert defining search strategy

Julien is an expert in search strategy for retrieving resumes and vacancies from the index. He logs into the administration interface and starts to define the search strategy. The search strategy consists of

defining the features of the query language. According to the indexed data, Julien defines the search facets and also the indexed fields that can be targeted during query time. This last feature will allow to simulate the SPARQL language behaviour of the text based search engine keywords queries.

Julien also uses this interface to manage the search REST API that is exposed to the developers' community. In this API, Julien build XML feeds to provide an access to the indexed data.

2.4.2.5 AU-5 scenario: Product manager

The product manager of the platform has a marketing role to promote the platform. She gathers the requirements of the customers that will define the features of the next releases of the "Active Hiring" system. She is also responsible for building POCs and demonstrators to promote the platform. For this reason, an appropriate packaging of the components will allow her to deploy the platform in other HR use cases like e.g. for sales purposes.

2.4.3 "Active-Hiring" developers community scenarios

2.4.3.1 DU-1 scenario 1: Sebastian scenario

Sebastian is a last year graduate student of a computer science in Leipzig University. For his graduation, he was asked to develop an innovative social platform. He heard about the LOD2 project and its "Active Hiring" platform. Sebastian decided then to develop a "linkedin" like social network based on the possible professional relationships that may exist between a candidate profile and recruiters. The social platform that Sebastian would like to build consists of making job seekers and recruiters meet and get in touch according to potential professional affinities. For example, if Linda was using Sebastian's social platform service, she could have in her wall a dashboard with different useful information for her job application: a map with relevant job offers surrounding and meeting her interests, relationship suggestions of recruiters that might be interested in her profile and more. To do so, Sebastian requires an automatic access to the "Active Hiring" platform through a REST API. This API allows Sebastian social platform to query the index and RDF store for getting candidates profiles/job offers: in a given geographic area, having special requirements… To get out RDF facts of the platform, the platform proposes a SPARQL endpoint that allows any kind of query to the RDF store of resumes and job offers. The responses from the API are in JSON or XML.

2.4.3.2 DU-2 scenario 2: JOBHub scenario

JOBHub is an important career and employment center that centralizes millions of people profiles and job offers freshly updated. In a common agreement, "Active Hiring" platform and JOBHub decided to collaborate closely by mutualising the resources they have in common. More specifically, "Active Hiring" is interested in having its index and RDF store updated by the valuable content (resumes and job offers) proposed by JOBHub. To do so, "Active Hiring" exposes a collaboration API that allows external partners to push directly new resumes and job offers inside the indexing and annotation pipeline of the platform.

2.5 Requirements from scenarios

Following the previous section, we present, in the following table, the list of the requirements highlighted by the different scenarios use cases. The presented requirements are described by an identifier (ID), a label, a description and the users requesting it.

ID	Requirement	Description	Impacted users

R1	Logging and authentication	Logging interface to use the "Active Hiring" platform	ALL
R2	Vacancies description input interface	Interface that allows HR managers and companies to push vacancies descriptions in the system	EU-1 EU-2
R3	Geolocalisation	Adds geo coordinates when detecting locations and addresses	EU-1 EU-2 EU-3
R4	Disambiguation of locations	Disambiguate between homonym locations	EU-1 EU-2 EU-3
R5	Taxonomy/ontology editor	Edits taxonomies and ontologies and provides visual features to view concepts and links among them. This feature includes also the possibility to build taxonomies and ontologies	AU-2 AU-3
R6	Extraction semantic concepts	Extracts taxonomy/ontology concepts from text data	EU-1 EU-2 EU-3 AU-3 AU-4 DU-1
R7	Semantic enrichment	Adds implicit concepts annotations to data (implicit means not explicitly present in data, i.e., inferred from taxonomy/ontologies)	EU-1 EU-2 EU-3 AU-3 AU-4 DU-1
R8	Concept linking	Adds RDF links (ex: `skos:exactmatch`) between extracted concepts and concepts from LOD cloud (dbpedia for example)	EU-1 EU-2 EU-3 AU-3 AU-4 DU-1
R9	Resource linking	Links data or document (job vacancy for example) to online or platform resources (ontologies, legal provisions, official websites…)	EU-1 EU-2 EU-3

			AU-3 AU-4 DU-1
R10	Ontology mapping and alignment	Aligns concepts from different taxonomies and ontologies to harmonize vocabularies	AU-2
R11	Map display and pinpointing	Renders a map of given regions and pin points job offers or resume in the map	EU-2 EU-3
R12	Graphical querying	Selects free or geometric surfaces from the map to bound the query to the selected area	EU-2 EU-3
R13	Faceted browsing and filtering	Filters query results according to displayed facets	EU-2 EU-3 AU-4
R14	Faceted search	Search that targets keywords on facets and makes it possible to combine different criteria. The criteria is a `<field:value>` query	EU-2 EU-3 AU-4 DU-1
R15	Sparql querying interface	This interface allows advanced users to express SPARQL queries directly submitted to the RDF store	EU-2 DU-1
R16	Sparql REST endpoint	This endpoint serves RDF triples responses from the RDF store to submitted SPARQL queries. It is intended for APIs and software interaction	DU-1
R17	Reasoning	This feature consists of appending additional RDF facts based on reasoning techniques to enrich data description. For example, add an RDF bag element to group individual skills of a candidate. This bag would then be mapped to “my skills”	EU-1 EU-2 EU-3
R18	Query completion	Query completion consist on suggesting values or even data structures in query formulation to assist the user typing his query	EU-2 EU-3 AU-4
R19	Resume editor	Resume editor is an interface that offers the possibility of inputting a user resume	EU-3
R20	Resume importer	This feature is a connector that loads an existing resume in the accepted formats of the platform (HR-XML, Europass)	EU-3 DU-2
R21	Resume exporter	This feature is a connector that generates a resume in the accepted formats of the platform (HR-XML and Europass)	EU-2 EU-3 DU-1

R22	Text input harmonizer	The harmonizer delimits the entered data by a given vocabulary from domain taxonomy.	EU-1 EU-3
R23	Domain taxonomies	Vocabularies that will be used for different tasks: annotation, suggestion, filtering, vocabulary harmonisation	AU-2
R24	RDF dump	Generation of RDF triples corresponding to the annotations produced by the platform for a giving resume or job vacancy	EU-2 DU-1
R25	Web scraper framework	Scraper that extracts information from HTML pages using XPath/XQuery queries	AU-1
R26	Information Extraction / NLP framework	Framework for achieving information extraction / NLP tasks. The extraction can cover named entities extraction, ontology driven extraction tasks…	AU-3
R27	Search design framework	Tools for defining search feeds and logics for querying indexed data	AU-4
R28	REST API for search hits	This API will be used to retrieve the hits of the index to a given query	EU-2 DU-1
R29	Data PUSH system	This API allows content providers to push their content into the “Active Hiring” platform	EU-1 DU-2
R30	Data models checking	This feature allows to check data models, ontologies to detect inconsistencies before integrating them into the platform.	AU-2
R31	Legal support of Hiring	Advanced querying feature to access legal provisions related to hiring process	EU-3

Table 4: List of "Active Hiring" requirements of features

3.LOD2 Stack components

LOD2 consortium is composed of 12 partners that bring their expertise, semantic resources and software components in the integrated project LDO2. In the following section, we describe the different components/resources provided by the WP8 partners and we provide a short table for each component resuming the major functionalities.

3.1 ULEI

3.1.1 DBpedia datasets and multi-domain ontology

DBpedia is a community effort to extract structured information from Wikipedia and to make this information available on the Web. It currently already contains a tremendous amount of valuable knowledge extracted from Wikipedia. The DBpedia knowledge base will be used for evaluation of LOD2's interlinking, fusing, aggregation and visualization components. The DBpedia multi-domain ontology will be used as background-knowledge and as an alignment and annotation ontology for LOD in general.

Feature ID	Description

F1	Domain datasets
F2	Domain ontology & taxonomies

Table 5: Features provided by DBpedia datasets

3.1.2 DL-Learner

The DL-Learner software learns concepts in Description Logics (DLs) from examples. Equivalently, it can be used to learn classes in OWL ontologies from selected objects. It extends Inductive Logic Programming to Descriptions Logics and the Semantic Web. The goal of DL-Learner is to provide a DL/OWL based machine learning tool to solve supervised learning tasks and support knowledge engineers in constructing knowledge and learning about the data they created.

Purposes of Class Expression Learning:

(1) Learn Definitions for Classes: Based on existing instances of an OWL class, DL-Learner can make suggestions for class definitions to be included as an owl:equivalentClass or rdfs:subClassOf axiom. As the algorithm is biased towards short and human readable definitions, a knowledge engineer can be supported when editing the TBox of an ontology.

(2) Find similar instances: DL-Learner's suggested class expressions can be used to find similar instances via retrieval (concept definitions as search). Scalable methods allow the generation of recommendations on the fly, e.g. in a web.

(3) Classify instances: The learned class descriptions can be used in a typical classification scenario, i.e. to decide for unknown instances whether they belong to a certain class. Common ILP benchmarks have been tested with DL-Learner. On the Carcinogenesis page, DL-Learner competes with other state-of-the-art ILP algorithms.

Feature ID	Description
F3	Learn concepts/classes definitions
F4	Map similar concepts
F5	Classification of unknown instances

Table 6: Features provided by DL-Learner

3.1.3 ORE

The ORE (Ontology Repair and Enrichment) tool allows for knowledge engineers to improve an OWL ontology by fixing inconsistencies and making suggestions for adding further axioms to it.

Ontology Debugging: ORE uses OWL reasoning to detect inconsistencies and satisfiable classes. State-of-the-art methods are then used to detect the most likely sources for the problems. In a simple process, the user can create a repair plan to resolve a problem, while maintaining full control over desired and undesired inferences.

Ontology Enrichment: ORE uses the DL-Learner framework to suggest definitions and super classes for existing classes in the knowledge base. This works if instance data is available and can be used to detect potential problems and harmonise schema and data in the knowledge base.

Feature ID	Description
F62	Ontology consistency validation

F63	Ontology enrichment

Table 7: Features provided by ORE

3.1.4 OntoWiki

OntoWiki is an agile tool for supporting knowledge engineering. It helps the editing, the authoring and the visualisation of semantic knowledge.

OntoWiki provides sophisticated means for navigating, visualising and authoring of RDF-based Knowledge Bases. It serves and consumes Linked Data and comprises a comprehensive middleware API for building custom Semantic Web applications. OntoWiki uses RDF in the first place to represent information and enables multiple views on data, such as tabular representations or maps. For machine consumption it supports various RDF serialisations as well as RDFa, Linked Data and SPARQL interfaces.

OntoWiki allows the definition of data models and corresponding views. It provides then a mechanism (RDFauthor mechanism) to automatically generate editable forms for these views and models. This allows OntoWiki users to edit the knowledge base even without being acquainted with the RDF, RDF-Schema or OWL data models. Following the Wiki-approach all changes are put under version control and can be easily rolled back.

Feature ID	Description
F64	Authoring tool for creating data models and views
F65	Authoring tool to generate interfaces and forms for data models

Table 8: Features provided by OntoWiki

3.2 CWI

3.2.1 MonetDB

MonetDB is an open-source high-performance database system that allows to store relational, XML and RDF data, downloadable from monetdb.cwi.nl. While being well-known for its columnar architecture and CPU-cache optimizing algorithms, the crucial aspect leveraged in the scope of this project is its unique run-time query optimization framework which provides a unique environment to crack the recursive-correlated-self-join queries caused by semantic web queries to triple stores.

Feature ID	Description
F6	XML store
F7	RDF store
F8	Optimized SQL engine

Table 9: Features provided by MonetDB

3.3 NUI Galway (DERI: Digital Enterprise Research Institute)

3.3.1 SIG.MA

http://Sig.ma is a tool to explore and leverage the Web of Data. At any time, information in Sigma is likely to come from multiple, unrelated Web sites - potentially any web site that embeds information in RDF, RDFa or Microformats (standards for the Web of Data).

Sig.ma can be used in 3 main ways:

(1) As a Web of Data browser: start from any entity and then click to another from the resulting page. Remember you are browsing a "network of mashups", quite a unique thing. It might be noisy but you can spot gems, e.g. interesting description differences in different sources.

(2) As an embeddable/linkable widget: create a Sigma, refine it and when you're ready to paste it around in emails and twits or embed it on your blog. Sigmas are "data live": if one of your selected sources updates its information, so will your Sigma be updated wherever it shows.

(3) As a semantic API: retrieve entity descriptions and specific properties. For example picture, phone@Giovanni Tummarello, ready to consume, in JSON, in RDF.

Feature ID	Description
F9	Index of RDF triples extracted from the web
F10	Text based query language on RDF data
F11	Query API of RDF triples index with multiple output format (rdf, json, atom, …)

Table 10: Features provided by Sig.ma

3.3.2 Sindice

Billions of pieces of metadata are on the Web today, with increasing uptake across the Internet from search engines to social sites to governments alike. The key technologies are RDF, RDFa and Microformats. Examples of such information types are contacts, events, social networks, web polls, reviews, and hundreds of other domain specific entities.

Sindice is a state of the art infrastructure to process, consolidate and query the Web of Data. Sindice collates these billions of pieces of metadata into a coherent umbrella of functionalities and services.

Feature ID	Description
F12	Index of RDF triples extracted from the web
F13	Text based query language on RDF data
F14	SPARQL endpoint to access indexed triples

Table 11: Features provided by Sindice

3.3.3 Sparallax

Sparallax is a faceted browsing interface for SPARQL endpoints, based on Freebase Parallax.

Feature ID	Description
F15	Faceted browsing interface on RDF triples hits

Table 12: Features provided by Sparallax

3.4 FUB

3.4.1 D2R Server

D2R Server is a tool for publishing the content of relational databases on the Semantic Web, a global information space consisting of linked data.

Data on the Semantic Web is modelled and represented in RDF. D2R Server uses a customizable D2RQ mapping to map database content into this format and allows the RDF data to be *browsed* and *searched* - the two main access paradigms to the Semantic Web.

D2R Server's **Linked Data interface** makes RDF descriptions of individual resources available over the HTTP protocol. An RDF description can be retrieved simply by accessing the resource's URI over the Web. Using a Semantic Web browser like Tabulator (slides) or Disco, you can follow links from one resource to the next, surfing the Web of Data.

The **SPARQL interface** enables applications to search and query the database using the SPARQL query language over the SPARQL protocol.

A traditional **HTML interface** offers access to the familiar Web browsers.

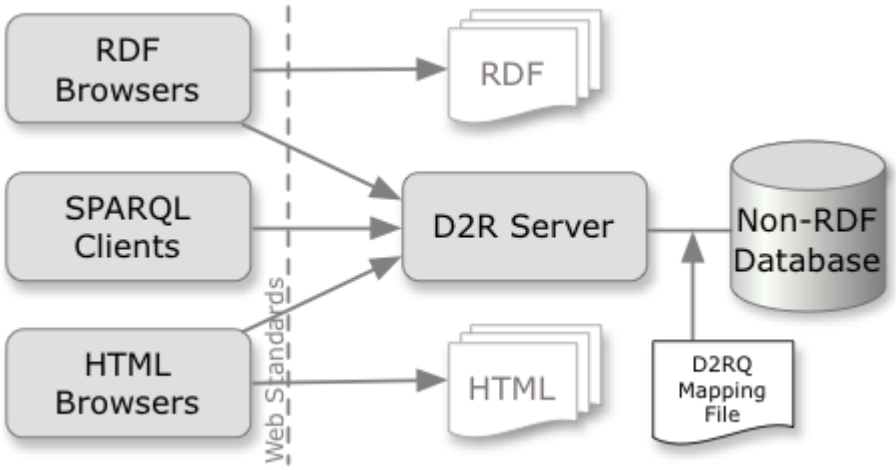


Figure 3: D2R Server architecture

Requests from the Web are rewritten into SQL queries via the mapping. This on-the-fly translation allows publishing of RDF from large live databases and eliminates the need for replicating the data into a dedicated RDF triple store.

Feature ID	Description
F16	Database mapper: relational schema converter into RDF
F17	DB tuples publisher on the web using RDF
F18	HTTP protocol navigation to provide HTML browsing and navigation of DB
F19	SPARQL engine with a SPARQL to SQL converter

Table 13: Features provided by D2R server

3.4.2 SEMMF

SemMF is a flexible framework for calculating semantic similarity between objects that are represented as arbitrary RDF graphs. The framework allows taxonomic and non-taxonomic concept matching techniques to be applied to selected object properties. Moreover, new concept matchers are easily integrated into SemMF by implementing a simple interface, thus making it applicable in a wide range of different use case scenarios.

Framework Architecture: The Matching Engine takes as input a query object and a collection of resource objects to be matched against the query object. Both are represented in RDF and may utilize different schema vocabularies. If they use concepts from a common taxonomy, an RDFS or OWL representation of this taxonomy has to be provided.

SemMF Engine is implemented in Java utilizing Jena2 Semantic Web Framework for accessing and querying of resource and query graphs as well as the underlying taxonomies.

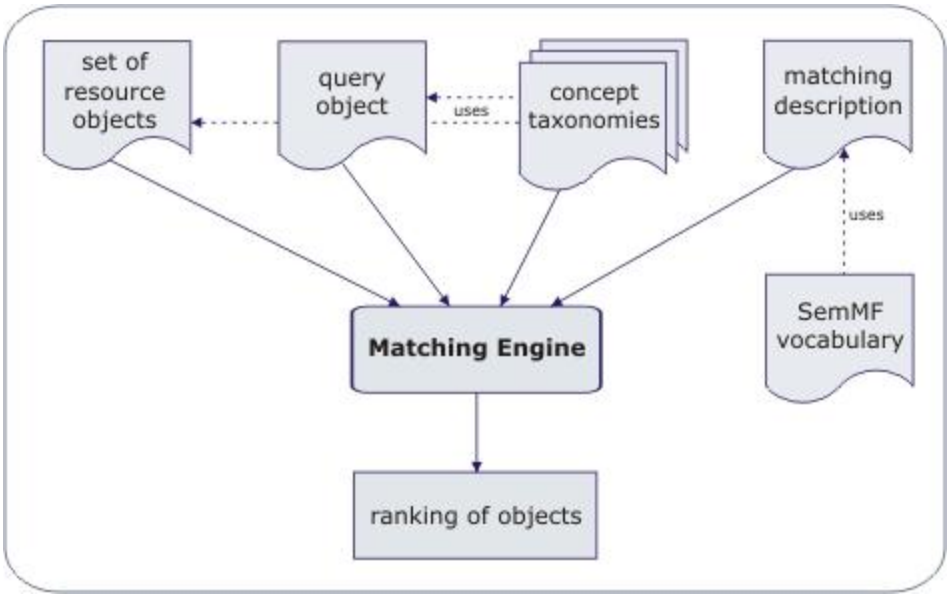


Figure 4: Architecture of SEMMF

Matching Description: In most cases not every object property is relevant for the similarity computation. For example, an object representing a certain product may contain manufacturer's phone number which may be irrelevant for comparing product's characteristics with customer's preferences (query object). Thus, each relevant property in the query RDF graph must be explicitly specified and mapped to the semantically corresponding property (i.e. holding the same kind of information, e.g. price information) in a resource RDF graph. Each mapping is assigned a concept matcher implementing a certain matching technique.

In a matching description the importance of each object property can be specified by assigning it a certain *weight*. Moreover, properties to be matched can be grouped into thematic clusters, for example all properties describing technical specification of a product. The property clustering enables to sort the matching result by cluster similarities. The matching description is represented in RDF using SemMF vocabulary provided with the framework.

Matching Process: Inside each thematic cluster the Engine calculates the similarity between each query property and the corresponding resource property. These similarities are multiplied by the indicated weights and summed up yielding the cluster similarity. All cluster similarities, in turn, multiplied by the specified cluster weights yield the object similarity.

However, if for a given query property value there is more than one semantically corresponding resource property value (e.g. a product may be available in different colors) the Engine chooses the one with the highest similarity.

Matching Result: The output of the Matching Engine is a ranking of objects by their similarity values. The Engine also provides a detailed description of the matching process (i.e. object property values, similarity values for all clusters and for each single object property within a cluster, associated weights, etc.) which can be used to generate explanations for the calculated object similarity.

Feature ID	Description
F20	Similarity matching between RDF graph concepts
F21	Clustering of RDF graphs concepts

Table 14: Features provided by SEMMF

3.4.3 Silk Framework

The Web of Data is built upon two simple ideas: First, to employ the RDF data model to publish structured data on the Web. Second, to set explicit RDF links between data items within different data sources. Background information about the Web of Data is found at the wiki pages of the W3C Linking Open Data community effort, in the overview article "Linked Data - The Story So Far" and in the tutorial on "How to publish Linked Data on the Web".

The *Silk Link Discovery Framework* supports data publishers in accomplishing the second task. Using the declarative *Silk - Link Specification Language* (Silk-LSL), developers can specify which types of RDF links should be discovered between data sources as well as which conditions data items must fulfill in order to be interlinked. These link conditions may combine various similarity metrics and can take the graph around a data item into account, which is addressed using an RDF path language. Silk accesses the data sources that should be interlinked via the SPARQL protocol and can thus be used against local as well as remote SPARQL endpoints.

The main features of the Silk link discovery engine are:

- Open source link discovery framework, running on all major platforms
- Support of RDF link generation (owl:sameAs links as well as other types)
- Flexible, declarative language for specifying link conditions
- Employment in distributed environments (by accessing local and remote SPARQL endpoints)
- Usable in situations where terms from different vocabularies are mixed and where no consistent RDFS or OWL schemata exist
- Scalability and high performance through efficient data handling (speedup factor of 20 compared to Silk 0.2):
  - Reduction of network load by caching and reusing of SPARQL result sets
  - Multi-threaded computation of the data item comparisons (3 million comparisons per minute on a Core2 Duo)
  - Optional blocking of data items

Feature ID	Description
F22	Linking concepts from different sources using RDF links like owl:sameAs for example
F23	Declarative language to define the validation procedure of links between concepts
F24	SPARQL connectors to Sparql endpoints

Table 15: Features provided by Silk framework

3.5 OGL

3.5.1 OpenLink Virtuoso Universal Server

3.5.1.1 General Purpose

Virtuoso is an innovative enterprise grade multi-model data server for agile enterprises & individuals. It delivers an unrivaled platform agnostic solution for data management, access, and integration. The unique hybrid server architecture of Virtuoso enables it to offer traditionally distinct server functionality within a single product offering that covers the following areas:

- Relational Data Management
- RDF Data Management
- XML Data Management
- Free Text Content Management & Full Text Indexing
- Document Web Server
- Linked Data Server
- Web Application Server
- Web Services Deployment (SOAP or REST)

3.5.1.2 Functional description

At core, Virtuoso is a high-performance object-relational SQL database. As a database, it provides transactions, a smart SQL compiler, powerful stored-procedure language with optional Java and .Net server-side hosting, hot backup, SQL-99 support and more. It has all major data-access interfaces, such as ODBC, JDBC, ADO .Net and OLE/DB.

Virtuoso has a built-in web server which can serve dynamic web pages written in Virtuoso's web language (VSP) as well as PHP, ASP .net and others. This same web server provides SOAP and REST access to Virtuoso stored procedures, supporting a broad set of WS protocols such as WS-Security, WS-Reliable Messaging and others. A BPEL4WS run time is also available as part of Virtuoso's SOA suite.

Virtuoso has a built-in WebDAV repository. This can host static and dynamic web content and optionally provides versioning. The WebDAV repository is tested to interoperate with WebDAV clients built into Windows XP, Mac OSX and others and makes Virtuoso a convenient and secure place for keeping one's files on the net. Further, Virtuoso provides automatic metadata extraction and full text searching for supported content types.

OpenLink Virtuoso supports SPARQL embedded into SQL for querying RDF data stored in Virtuoso's database. SPARQL benefits from low-level support in the engine itself, such as SPARQL-aware type-casting rules and a dedicated IRI data type. This is the newest and fastest developing area in Virtuoso.

Feature ID	Description
F7	RDF store
F6	XML store
F27	Sparql endpoint access
F28	WebDav repository to allow access in read/write mode to stored documents

F8	SQL engine
F29	API access to RDF store supporting .net, jdbc, odbc, OLD/DB…

Table 16: Features provided by Virtuoso

3.5.2 OpenLink Virtuoso Sponger: RDF Middleware

OpenLink Virtuoso's Sponger is an RDFizer providing middleware for creating RDF linked data from non-RDF data sources. The Sponger forms part of the extensible RDF framework built into Virtuoso Universal Server. A key component of the Sponger's pluggable architecture is its support for Sponger Cartridges, which themselves are comprised of an Entity Extractor and an Ontology Mapper. Virtuoso bundles numerous pre-written cartridges for RDF data extraction from a wide range of data sources. However, developers are free to develop their own custom cartridges.

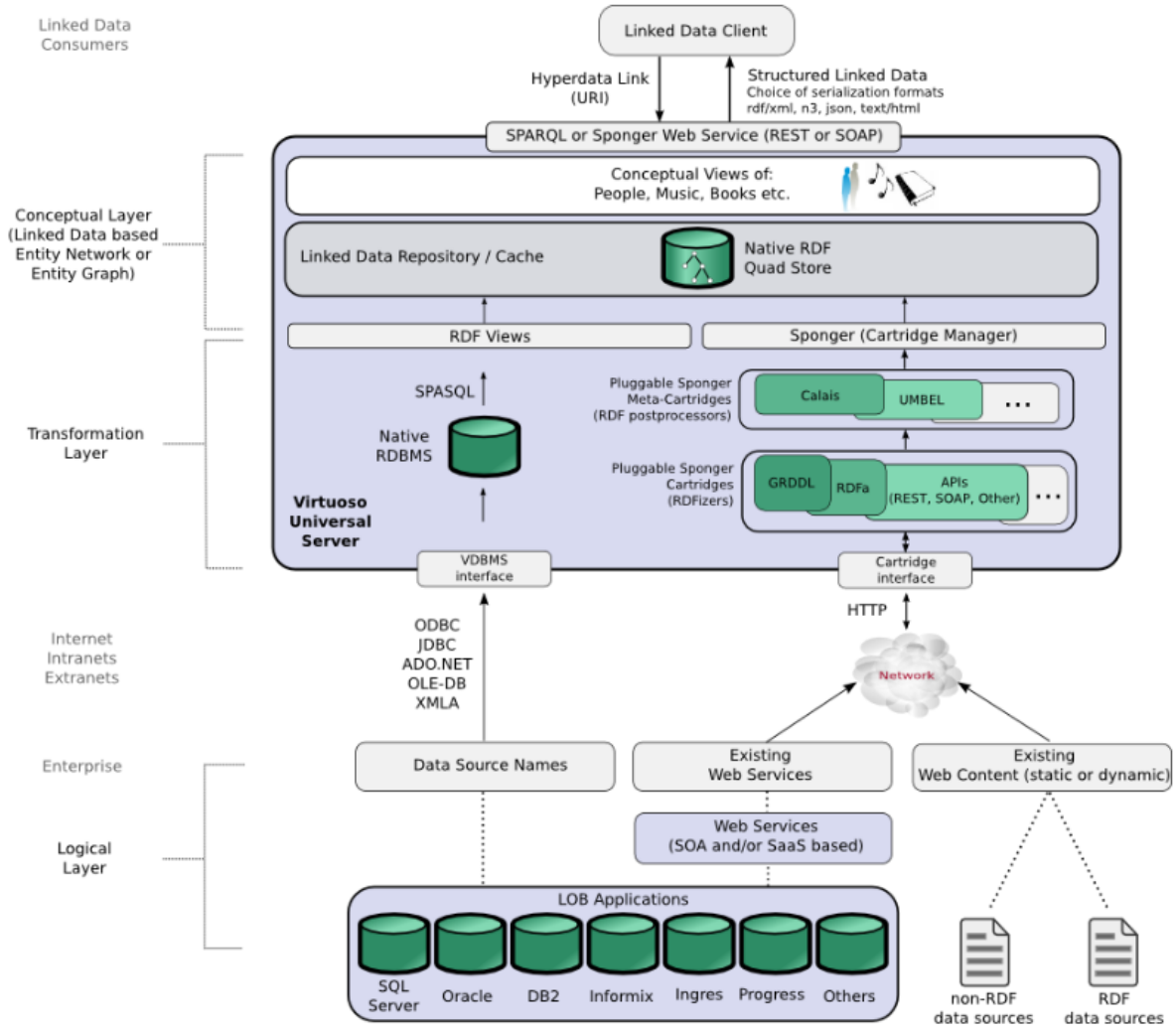


Figure 5: Sponger architecture

Feature ID	Description
F30	Framework for transforming flat or poorly structured content into RDF
F31	Support of NLP frameworks to achieve data extraction and transformation into RDF

Table 17: Features provided by Virtuoso Sponger

3.6 SWCG

3.6.1 PoolParty Thesaurus Manager (PPTM)

PoolParty Thesaurus Manager is a web-based thesaurus management system which is completely built on top of W3C's semantic web standards. In its core PoolParty uses RDF to represent SKOS and other vocabularies like Dublin Core or FOAF, therefore an RDF triple store is used as its technological basis.

Compared to other systems which still rely on relational databases PoolParty is ready to consume and to publish Linked Data out-of-the-box. Besides the possibility to publish any PoolParty based thesaurus via a Linked Data front-end, the system offers a SPARQL endpoint to execute queries over each thesaurus project. This technology can be used to integrate a thesaurus with other platforms (wikis, CMS, etc.) or search engines.

Thesauri in the age of the web most often should be engineered and maintained in a collaborative manner. PoolParty is fully web-based; administrators need only a web browser to do all typical CRUD operations like adding new concepts or relations. PoolParty also offers a wiki frontend for each thesaurus project so that more people can be involved in the thesaurus development process. Linking concepts is another flexible way to build thesauri in decentralised structures. Based on the linked data principles thesauri can be maintained at different places but still can be connected to each other indicating that several concepts are similar or even identical to each other.

PoolParty also makes use of existing Linked Data sources, e.g. concepts can be aligned and enriched with additional information from sources like Dbpedia, Sindice or others.

Feature ID	Description
F32	Taxonomy/Ontology editor
F37	Query auto completion
F38	Query expansion driven by thesauri concepts
F41	Taxonomy/ontology concepts mapping with concepts in LOD cloud

Table 18: Features provided by PoolParty Thesaurus Manager

3.6.2 PoolParty Extractor (PPX)

The PoolParty Extractor is an application for analyzing text, suggesting controlled and free tags for it and storing these tags + additional semantic information into an index.

PPX can retrieve text from different sources (like XHTML, HTML, PDF, plain text, RDF and connected systems like Drupal or Atlassian Confluence wiki) and do statistical analysis on it. As PPX creates an extraction model from a thesaurus, the suggested tags can be controlled semantic tags instead of conventional keyword tags. I.e. PPX checks whether the analysed text contains any words or phrases that match to any concept label stored in the loaded thesaurus. These concepts can be associated with synonyms, abbreviation, multilingual labels, etc., so that they make for much better tags and improve findability of tagged content.

Additionally concepts can be mapped to resources from the Linked Data cloud, where even more background knowledge (like longitude and latitude or category and type information from DBpedia or Yago) is available, thereby enriching the semantic fingerprint of a document tagged with such concepts and offering more possibilities for improved content recommendation and similarity calculations. The calculated tag suggestions can also be used to classify documents according to pre-defined categories.

The PoolParty Extractor uses a rule based system for performing sentence splitting, tokenization, stemming, and phrase construction on natural language documents. The words and the phrases extracted from the document are assigned with a relevancy score, and the extractions with the highest scores are chosen as keywords, key phrases to represent the input document. The scoring function of the extractor take the frequency and the position of the words into account (the more frequent a word is, the higher score it gets, and the words at the beginning of the text are considered more relevant, so get a higher score than words at the end of the document).

The extracted words and phrases are looked up in a thesaurus, after which a tag suggestions divided in concept tags and free tags can be presented to the user or automatically be associated with the document. All tags and their related info (labels, category information, Linked Data information, etc.) are stored in an index. Additionally any metadata from the original document (like creation date, author, abstract, etc.) and its full text can be stored in the index.

This index then can form a basis for services for

- retrieval of additional information from Linked Data resources,
- auto completion of user entered terms,
- facets provided from a thesaurus for search assistants or navigational aids
- moderated search or query expansion powered by the thesaurus

PPX should preferably be used with English language text.

Feature ID	Description
F33	Data extraction driven by an ontology/taxonomy
F34	Concept mapping between extracted data and concepts in LOD cloud
F35	Representative concepts and phrases extraction
F36	Scoring of representative concepts of a document

Table 19: Features provided by PoolParty Extractor

3.6.3 PoolParty Tag and Content Recommender

The PoolParty Tag and Content Recommender is an application built on top of the PoolParty Extractor. It is currently used to demonstrate the possibilities of semantic tagging and recommendations. PPX analyses the text and suggests extracted terms and phrases that are important inside these texts. These extracted terms are then used to calculate similarity with other documents.

The PoolParty Tag and Content Recommender can also be loaded with any SKOS thesaurus, which is then used as a basis for extracting concepts from a given text. This not only enables semantic tagging with concepts instead of free tags, it also gives the possibility of more fined grained tag recommendations and the system can even recommend tags that are not explicitly mentioned in the text, but that are related to terms from the text. Additionally the thesaurus concepts can be linked to the LOD cloud and have consequently much more metadata connected to them, which again can the used to provide better and more intelligent content recommendations.

From an end user's perspective the PoolParty - Tag & Content Recommender works like this:

- The user provides a text (either by entering the text into an editor or by retrieving the text from an URL or by clicking on a bookmarklet).
- PoolParty analyses the text.

- PoolParty suggests

  A) Tags from a thesaurus

  B) Tags from other words and phrases that appear to be important for the text

  C) Documents similar to the analysed text and

  D) Images from these documents.

Feature ID	Description
F35	Representative concepts and phrases extraction
F39	Document similarity
F33	Data extraction driven by an ontology/taxonomy
F40	Tag annotation
F42	Image suggestion from extracted concepts

Table 20: Features provided by PoolParty Tag and Content Recommender

3.6.4 DBpedia Extraction Framework

The DBpedia Extraction Framework takes several URIs for resources from DBpedia as input and follows their category links in order to build a hierarchy (or a graph) from it and subsequently create a SKOS thesaurus with this data.

Future development of this tool should include possibilities to allow the Extractor to

- exclude certain categories from the crawling process,
- use heuristics to identify viable links to be followed,
- use other Linked Data sources apart from DBpedia,
- fulfil other to be defined requirements from this work package.

Feature ID	Description
F43	Authoring tool for building SKOS thesauri from DBpedia

Table 21: Features provided by DBpedia Extraction Framework

3.7 TenForce

Tenforce is a service integrater. It has been working on several EU projects. One of these projects establishes the ESCO taxonomy as an extension to ISKO-88 and ISKO-08. ESCO is the EU taxonomy for Occupations, Skills and Competences

Feature ID	Description
F66	Domain taxonomy
F68	Integration services.

Table 22: Features provided by TenForce

3.8 Exalead

Exalead is a software editor of information search platforms. In LOD2 project, Exalead will bring its expertise via its search platforms: CloudView™. In the following, we describe in details the functionalities provided by CloudView™

3.8.1 CloudView ™ for SBA

Knowledge and experience are a company's most precious resources as they permit qualified and rapid decision-making. A smart solution for managing these resources should go beyond simply indexing and browsing data. This can be achieved by using semantic technologies to reconcile formats, structures and terminologies, and identify embedded meanings and relationships within and across resources.

With this respect, Exalead CloudView™ platform provides a uniform access to the information cloud of an enterprise. CloudView™ let business companies and institutions build advanced and complex search based applications (SBA). SBAs are being used for all applications in which information access plays a central role.

SBAs provide information access that is faster and far cheaper than relational database querying, because search engines are uniquely designed for fast information access ('read' operations) by vast numbers of users against massive data volumes. SBAs are being used for database applications including:

- Pure Information Access Applications: Online directories & classifieds, media portals, e-discovery, compliance, knowledge bases, etc.
- Information Access Plus Operational Reporting: Logistics (track and trace), quality control, network monitoring, survey banks, etc.
- Database-Centred Business Applications: Customer service/call centre, e-commerce, reservation systems, inventory management, etc.

SBAs are being used to bring new agility and expanded scope to enterprise applications like CRM, ERP, SCM, and Business Intelligence (BI) by enriching structured data with important emotive and qualitative data from 'unstructured' sources like email, blogs, chat, phone transcripts, and Web pages. In addition, because search engines can update large volumes of data in real-time even under heavy usage loads, they can significantly improve the operational efficiency.

3.8.1.1 CloudView™ Architecture:

The Exalead CloudView open architecture provides powerful APIs for connectors, analysis, document indexing, search query execution and platform monitoring and administration, so that Exalead CloudView can be easily integrated with other software components such as portals or Content Management Systems to deploy search based applications.

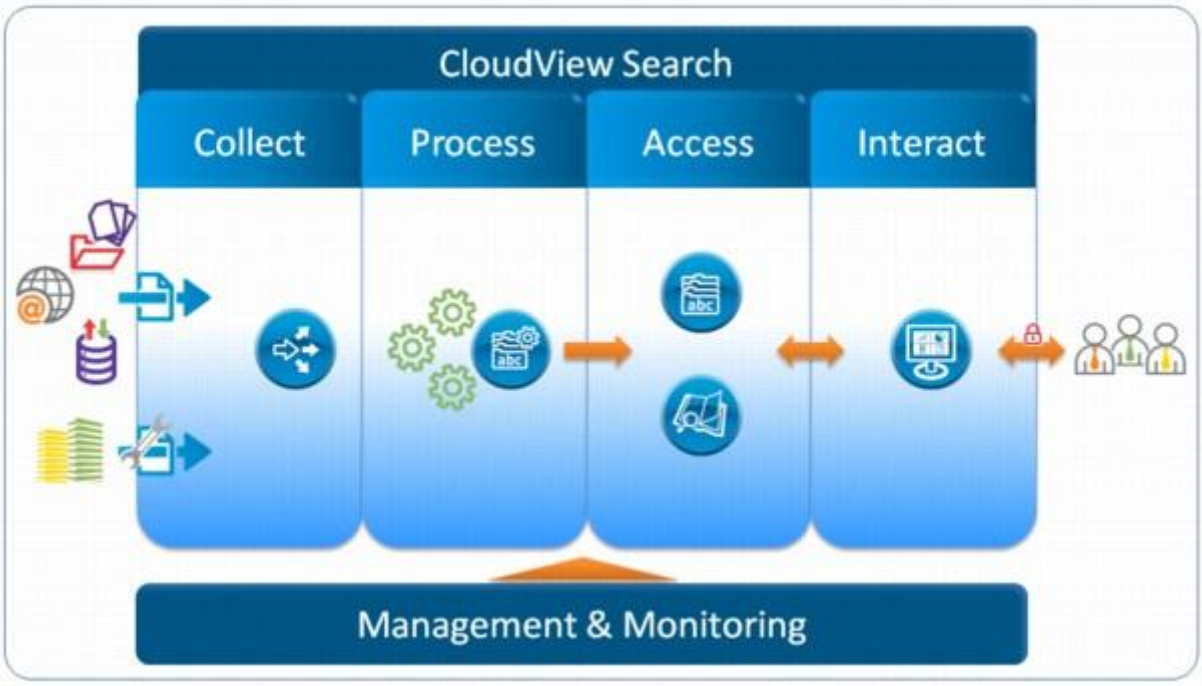


Figure 6: The Exalead CloudView architecture

The Exalead CloudView's modular architecture has been designed to be easily extended by external software components. This platform provides a comprehensive set of APIs allowing third-party applications to push documents into the index database, configure the analysis workflow and index build, send search queries and retrieve search results, monitor the application and services. The following diagram details the extension APIs available to Exalead CloudView 5.0:

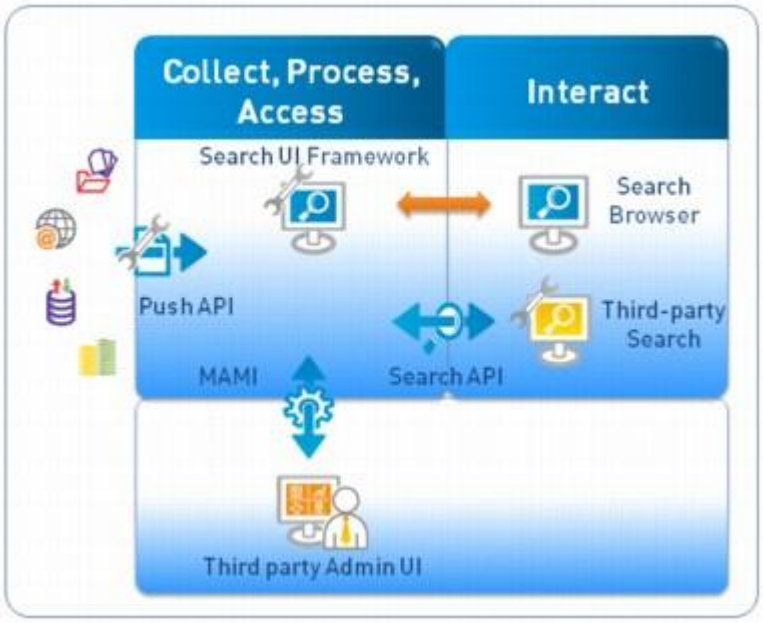


Figure 7: Exalead's extensible platform

Indexing documents

The Exalead CloudView Push API (PAPI) is a simple client/server API that allows an external application to feed the Exalead index with documents. Based on a simple HTTP/REST protocol, this API can be easily

used by any programming language. Higher level wrappers, such as Java, Python and C#, encapsulating the network protocol are also provided.

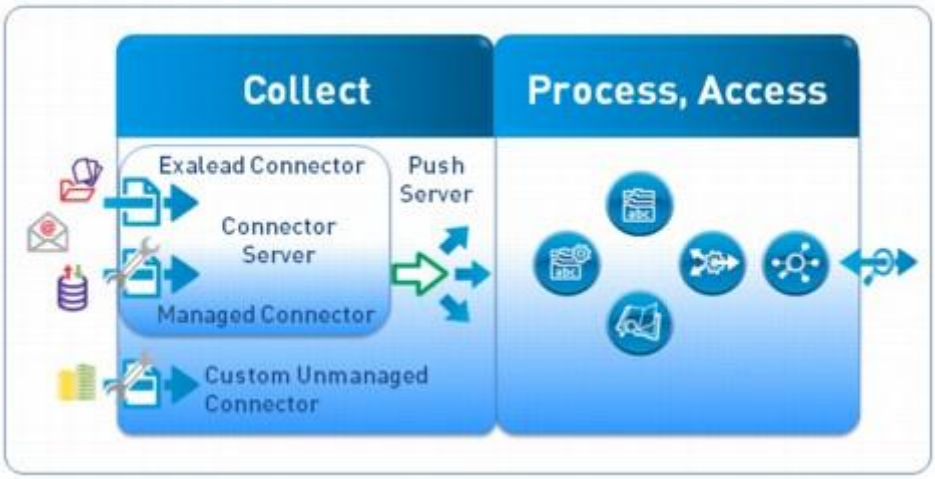


Figure 8: Push API Architecture

The Push API leverages the transactional features of the Exalead index build process to provide robust checkpoint operations, so that connectors using this API can be crash proof and guarantee the consistency of the indexed documents w.r.t. the current state of the document repositories.

The Push API supports the basic operations required to develop new connectors including:

- adding new documents
- updating and deleting existing documents
- retrieving the current list of indexed documents

The main components in the Push API architecture are:

- Push server
- Push client
- Push connector using the Push client
- Push connector using the Connector framework
- Document fetcher (for example, for thumbnails and previews) using the Push client

When a document is pushed to Exalead CloudView, it can contain one or more parts and associated metadata items. The connectors push documents to Exalead CloudView. A document is represented by the following fields:

- Document uri
- document stamp to indicate the version
- `<meta data name>, <value>` pairs
- parts containing the bytes from the document
- directives for data extraction

The first step is to transform the content of the document parts and the metadata items into internal items that can be processed by the document and semantic analysis processors. These items are called

document chunks. One or more document chunks are tagged as a context. The contexts created depend upon the extraction process. The context names are case sensitive. By default the extraction process creates two contexts (with one or more document chunks) text and title for each part, and a context for each metadata item.

Once the document has been represented as contexts with one or more document chunk, it can be processed by the analysis processors. The processors can perform one of the following:

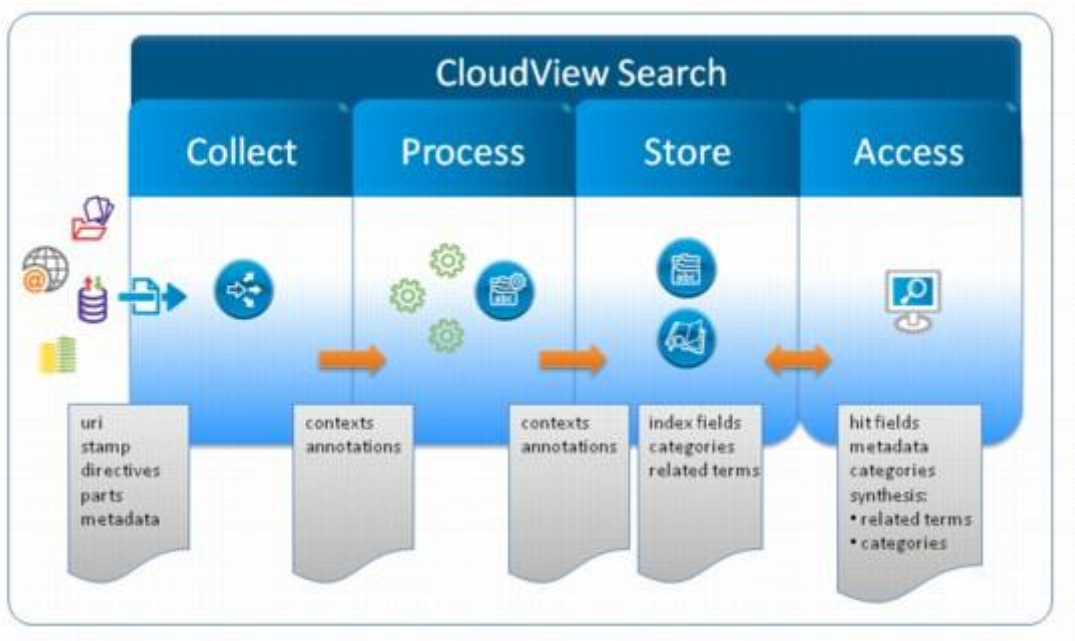


Figure 9: Document Lifecycle in Exalead PAPI

Exalead CloudView™ provides categorization functionalities to annotate indexed documents. Categorization or taxonomy defines how documents in the corpus (your unified information sources) will be annotated based on a document's author, last modification date, file format, language, from where it was crawled (its source), or by a category you define (for example, a specific fileshare directory or URL). These groups of categories form what is called query category groups.

Once these categories are defined, the appropriate categorization information is added to each document (if available). This allows Exalead CloudView to group together documents with similar contents allowing your users to further refine and explore related topics.

Document processing

When the connector pushes documents to Exalead CloudView they are processed by the Push server. This is the entry point in the workflow for document analysis and indexing. The diagram below illustrates the indexing chain:

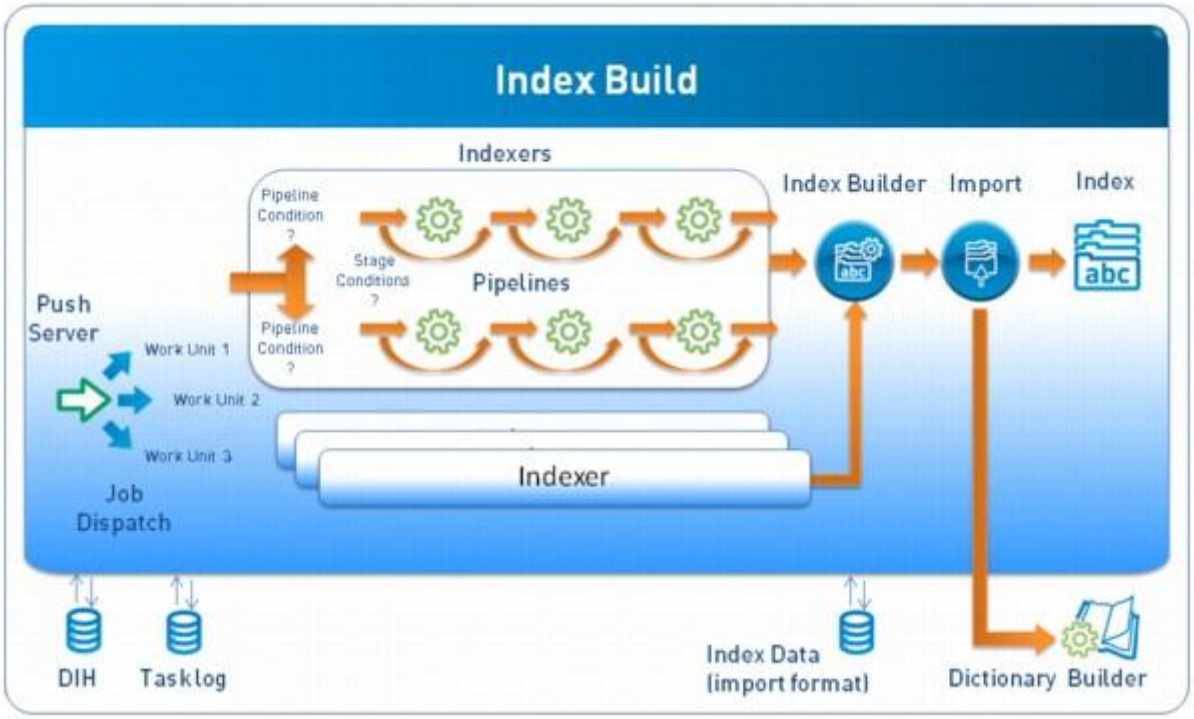


Figure 10: Indexing chain

The analysis and semantic processors provide the following functionality:

- Text extraction: handles a wide range of document formats (including PDF, Microsoft Office, and Open Office …) and uses document filters that are able to extract the text and metadata at a very high throughput.
- Document Processors: provides a set of processors that are able to automatically modify the content of the documents for indexing. For example, the document's language and type are automatically detected, and the result of this detection can be used as document categories. At the end of this phase the original document has been enriched with new or transformed content and annotations that can be mapped in the final phase to fields in the index.
- Semantic Analysis: provides a set of semantic filters that are able to detect related terms, perform semantic processing, categorization and ontology matching. At the end of this phase the original document has been enriched with new annotations that can be mapped in the final phase to fields in the index.
- Content and Annotation Mapping: the heterogeneous content and annotations, gathered from the different connectors or generated by the processors, need to be mapped to index fields that are common to all the data sources. For example, the notion of document date can be found in a LastModifiedTimeStamp field for a given document repository, whereas it is stored in the CreationDate field for another repository.

Searching the index

The Exalead CloudView Search API allows an external application to query the index. It provides all standard search features including search querying, faceted navigation, search profiles, spell checker and many more. In addition to that, the Search API allows the application to control the advanced query execution parameters such as the number of returned search results, the search time-out values, the

number of categories selected, and more. The Search API also gives control of the relevancy ranking parameters such as the sort criterion (relevancy, date, document size, etc.), the relative weight of the relevancy score values, and various ranking bonus options. In addition, Exalead CloudView's Search Manager can be used to configure query expansion processing and post-filtering processing.

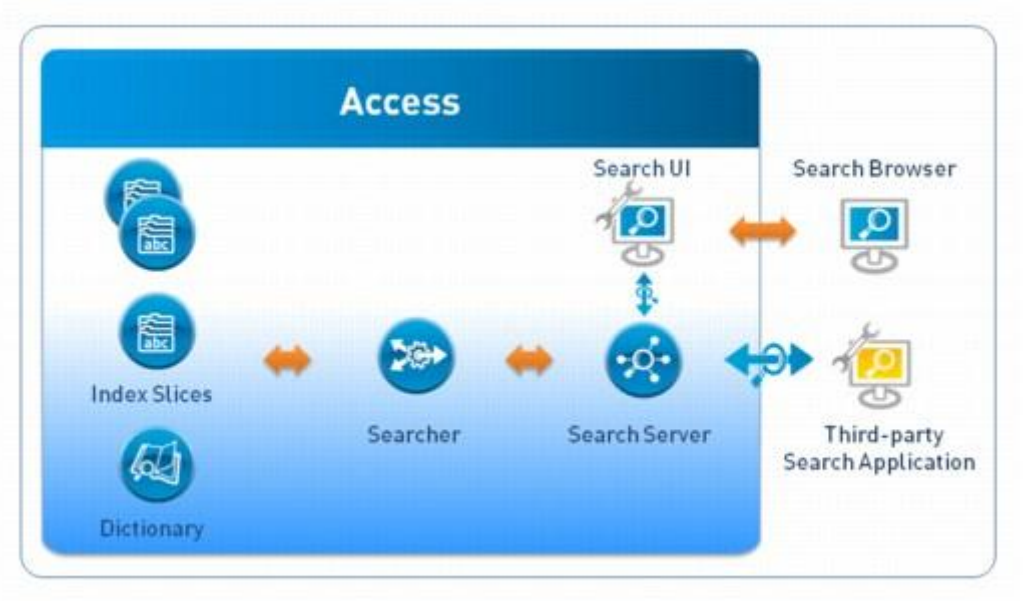


Figure 11: Search API

To be able to provide relevant search results even when the user's query is incomplete, misspelled or imprecise, the Exalead query language provides powerful word matching operators and fuzzy matching options, including:

- Approximate spelling (e.g. "exalaed" will match "exalead")
- Phonetic spelling (e.g. "exaleed" will match "exalead")
- Word truncations (e.g. "exal* " will match "Exalead", "exalid" and "exalted")
- Regular expressions (e.g. " /exa.ead/ will match "exalead" and "exahead")
- Semantic expansion (e.g. "plane" will match "aircraft")

The user can use arguments in the query to target certain fields or influence the use of linguistic resources. Similar to the indexing pipeline, when the Exalead search engine is processing a search query, it first parses the input text into individual words (tokenization), then identifies possible variants for these words (lemmatization).

All the possible matches for the query terms are searched within a dictionary built upon the corpus of known documents, so that the available vocabulary includes all the application-specific words such as people names, product names and technical terms.

The matching words found in that dictionary are associated with a numerical score representing the distance (lexically, phonetically or semantically) between them and the initial words typed in the user's query. This distance is taken into account in the computation of a document's relevancy score, so that the relative weight given to a word is higher when the word is closer to the original user's query.

Individual query terms can also be combined by using a comprehensive set of query operators, including boolean operators, word position operators, and parenthesized expressions. Finally, the system uses a built-in spell checker to verify the query input and offer the user alternative spellings as needed.

Monitoring and managing the platform

The Exalead CloudView MAMI exposes administration, inspection and operations through web services. These APIs give complete control of the product's configuration parameters, as well as status information concerning the platform's components. These APIs can be used by third-party applications embedding the Exalead CloudView OEM software as an OEM component.

The MAMI is accessible directly using SOAP, through a MAMI client that encapsulates SOAP or through the command line that exposes all operations for scripting or testing. The Exalead WSDL interfaces define the operations and messages for Exalead CloudView MAMI Web Services. The following diagram illustrates how a third-party application can be developed and deployed using the one or all of the web services managers:

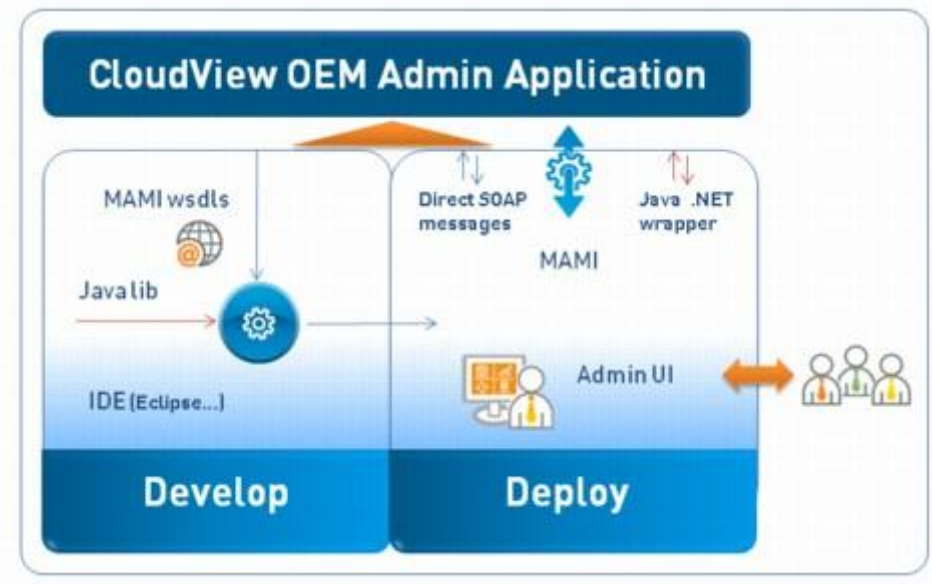


Figure 12: Example of the Exalead Web services architecture

A MAMI client is packaged with Exalead CloudView. It encapsulates all the SOAP envelopes as defined in the MAMI wsdl.

Exalead CloudView now provides a set of 360 modules on top of the platform; providing tools to help create Search Based Applications (SBA) and handle typed documents from the data collection step, to the Mashup interface design.

3.8.1.2 Presentation of the 360 modules

The 360° modules provides means for defining business rules for managing information from its indexing to its presentation. These rules include data formatting, typing and reconciliation, mashup design of data rendering interfaces, etc. The following are the modules included in the Exalead CloudView 360 package:

- Business Console: Designed to give control of search applications to business users, with features such as synonyms, search relevance tuning, content boosting or query reporting.
- Mashup Builder: Intuitive administration interface for 360 components, from data ontology configuration to a fully drag and drop Mashup builder
  - Mashup UI: Highly modular and easy to customize front-end delivered with a rich set of widgets to build rich mashups on top of a Mashup API.
  - Mashup API: Query time content federation engine. Allows fetching content from heterogeneous sources at query time to search across different corpus or enrich already fetched hits.
  - Administration APIs: A set of HTTP REST services to programmatically configure the 360 modules, apply partially or fully the configuration etc.
- Trusted queries: A next generation content suggestion engine that takes advantage of the data structure to help the user to build semi structured queries to find content.
- Data Ontology: High level data schema description. Provides an object oriented model for describing your data types, relationships and indexing strategies.
- Content Recommender: A powerful and intuitive Web console for applying e-business rules and triggers to query processing and the display of online advertising and editorial content.

When to use 360 modules?
Exalead CloudView 360™ is a good fit for the following use cases:

- Search Based Application: Web applications usually deals with different kind of typed objects that are all related to each other at query time to create rich mashups, which is exactly what CloudView 360™ is made for.
- Proof of concepts: Demonstrating the core of any customized application only requires a few minutes with CloudView™. The Mashup Builder makes it possible to gather your client's needs and sketch them right in front of him.
- Rich intranet portals: Just as for Search Based Applications, creating rich intranet portals that provides content from different sources in an elegant mashup is very easy with CloudView 360™. Nevertheless, please be careful as Security support in CloudView 360™ (UI) is still experimental and thus not delivered in the standard distribution.
- Business users control Search Applications: The Business Console is a module on top of the Exalead CloudView™ platform, providing tools to simplify the creation of Search Based Applications. The Business Console gives control of search applications to business users with features such as synonyms, search relevance tuning, content boosting or query reporting.

Feature ID	Description
F45	Text based index
F46	HTTP web crawler
F47	Database connector
F48	Named entities extraction and ontology/taxonomy concept extraction
F49	Integrated scraping tool
F50	XML connector to index xml documents
F51	REST Push API for remote indexing
F52	Monitoring API for integration requirements
F53	Monitoring interface for scraping
F54	Monitoring interface for crawling
F55	Monitoring interface for search logic
F56	Monitoring interface for index mappings, data models and indexing strategies
F57	REST/XML API for querying index
F58	Auto complete and trusted queries
F59	Faceted search
F60	Monitoring interface for building mashup apps and feeds from different apps
F61	Mapping features of documents to tags and annotations

Table 23: Features provided by Exalead CloudView platform

3.9 OKFN

OKFN manages CKAN, a data repository of open knowledge.

3.9.1 CKAN (Comprehensive Knowledge Archive Network)

CKAN is a registry or catalogue system for datasets or other "knowledge" resources. CKAN aims to make it easy to find, share and reuse open content and data, especially in ways that are machine automatable.

Feature ID	Description
F61	Datasets from the LOD cloud

Table 24: Features provided by CKAN

3.10 WKD

Wolters Kluwer Deutschland is a publisher of legislation, jurisprudence and related doctrine. One particular business area is Human Resources. For this domain, WKD provides content and taxonomies.

Feature ID	Description
F66	Domain Taxonomies

F67	Domain specific content

Table 25: Features provided by WKD

4.Functional requirements mapping to available LOD2 components

From the different users scenarios and the cartography of functionalities covered by the LOD2 stack, we present in the following section the mapping between the requested functionalities and the available LOD2 components. The aim is to provide a clear view of the covered requirements by the LOD2 components features and to highlight the requirements that will need specific developments and adaptations.

Requirements	Available feature	Corresponding component	Development status
R1	None	None	To be developed
R2	F64	Onto Wiki	Usable as it is
	F65	Onto Wiki	Usable as it is
R3	F34	PoolParty Extractor (PPX)	Requires configuration and minor developments
	F61	Exalead CloudView indexing mappings	Requires configuration and minor developments
	None	Geonames webservice[2] connector	To be developed
R4	None	None	To be developed if necessary
R5	F32	PoolParty Thesaurus Manager (PPTM)	Usable as it is
	F43	DBpedia extraction framework	?
	F61	CKAN dataset	Requires specific browsing and search
R6	F48	Exalead CloudView (Ontology Matcher + Query Matcher + Fast Rules modules)	Requires configuration
	F33	PoolParty Extractor (PPX) + PoolParty Tag and Content Recommender	Usable as it is
R7	F3	DL-Learner (class/concept definition learner)	?
	F5	DL-Learner (classification module)	?
	F48	CloudView (Fast Rules module)	Requires configuration and minor developments
R8	F4	DL-Learner	?

[2] http://www.geonames.org/export/web-services.html

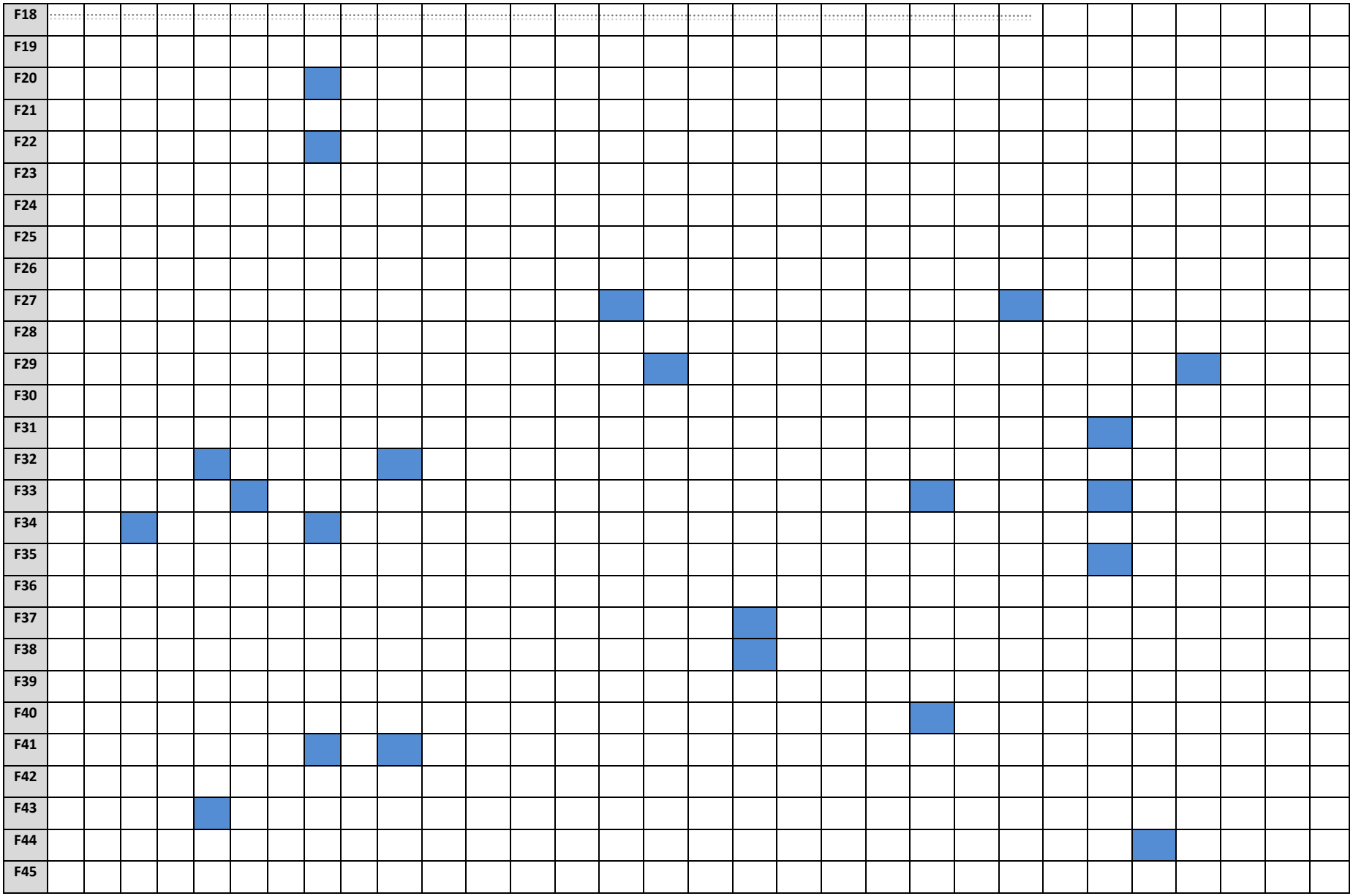
	F20	SEMMF	?
	F22	Silk framework	?
	F34	PoolParty Extractor	Requires adaptation and minor developments
	F41	PoolParty Thesaurus Manager	Usable as it is
R9	F61	CloudView + WKD resources	Requires adaptation
R10	F32	PoolParty Thesaurus Manager	Usable as it is
	F41	PoolParty Tag and Content Recommender	Requires configuration and minor developments
	F63	ORE	Usable as it is
R11	F60	CloudView 360° (using google maps)	Usable as it is
R12	F60	CloudView 360°	To be developed
R13	F15	Sparallax	?
	F59	CloudView (search interface)	Usable as it is
R14	F15	Sparallax	?
	F59	CloudView (query language)	Usable as it is
R15	F8	MonetDB	To be developed
	F14	Sindice	Usable as it is
	F27	Virtuoso Universal Server	Usable as it is
R16	F11	Sig.ma	Usable as it is
	F29	Virtuoso Universal Server	Usable as it is
R17	None	None	To be developed if necessary
R18	F37 & F38	PoolParty Thesaurus Manager	Requires configuration
	F58	CloudView (trusted queries)	Usable as it is
R19	None	None	To be developed
R20	None	None	To be developed
R21	None	None	To be developed
R22	F33	PoolParty Extractor	Requires configuration and minor developments
	F40	PoolParty Tag and Content Recommender	Requires adaptation and minor developments
R23	?	(WKD + TenForce)?	?
R24	F27	Virtuoso Universal Server	Usable as it is
R25	F49	CloudView (scraper module)	Requires adaptation and

			minor developments
R26	F31	Virtuoso Sponger	Requires adaptation and minor developments
	F33	PoolParty Extactor	Requires adaptation and minor developments
	F35	PoolParty Tag and Content Recommender	Requires adaptation and minor developments
	F48	CloudView (NLP pipeline module)	Usable as it is
R27	F55	CloudView (search logic interface)	Usable as it is
	F60	CloudView 360° (feed interface)	Usable as it is
	F44	TuQS	Requires adaptation and major developments
R28	F11	Sig.ma	Usable as it is
	F29	Virtuoso Universal Server	Usable as it is
	F57	Exalead Cloudview (REST search API)	Usable as it is
R29	F51	Exalead CloudView (Push API framework)	Usable as it is
R30	F62	ORE	Usable as it is
R31	F66	Legal domain content and taxonomies	Usable as it is
	F67	Legal domain content	Usable as it is
	F68	Integration modules	Usable as it is

Table 26: Mapping between requirements and covered features of LOD2 components

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20	R21	R22	R23	R24	R25	R26	R27	R28	R29	R30	R31
F1																															
F2																															
F3																															
F4																															
F5																															
F6																															
F7																															
F8																															
F9																															
F10																															
F11																															
F12																															
F13																															
F14																															
F15																															
F16																															
F17																															

Project funded by the European Commission within the Seventh Framework Programme (2007 – 2013)


F18
F19
F20
F21
F22
F23
F24
F25
F26
F27
F28
F29
F30
F31
F32
F33
F34
F35
F36
F37
F38
F39
F40
F41
F42
F43
F44
F45

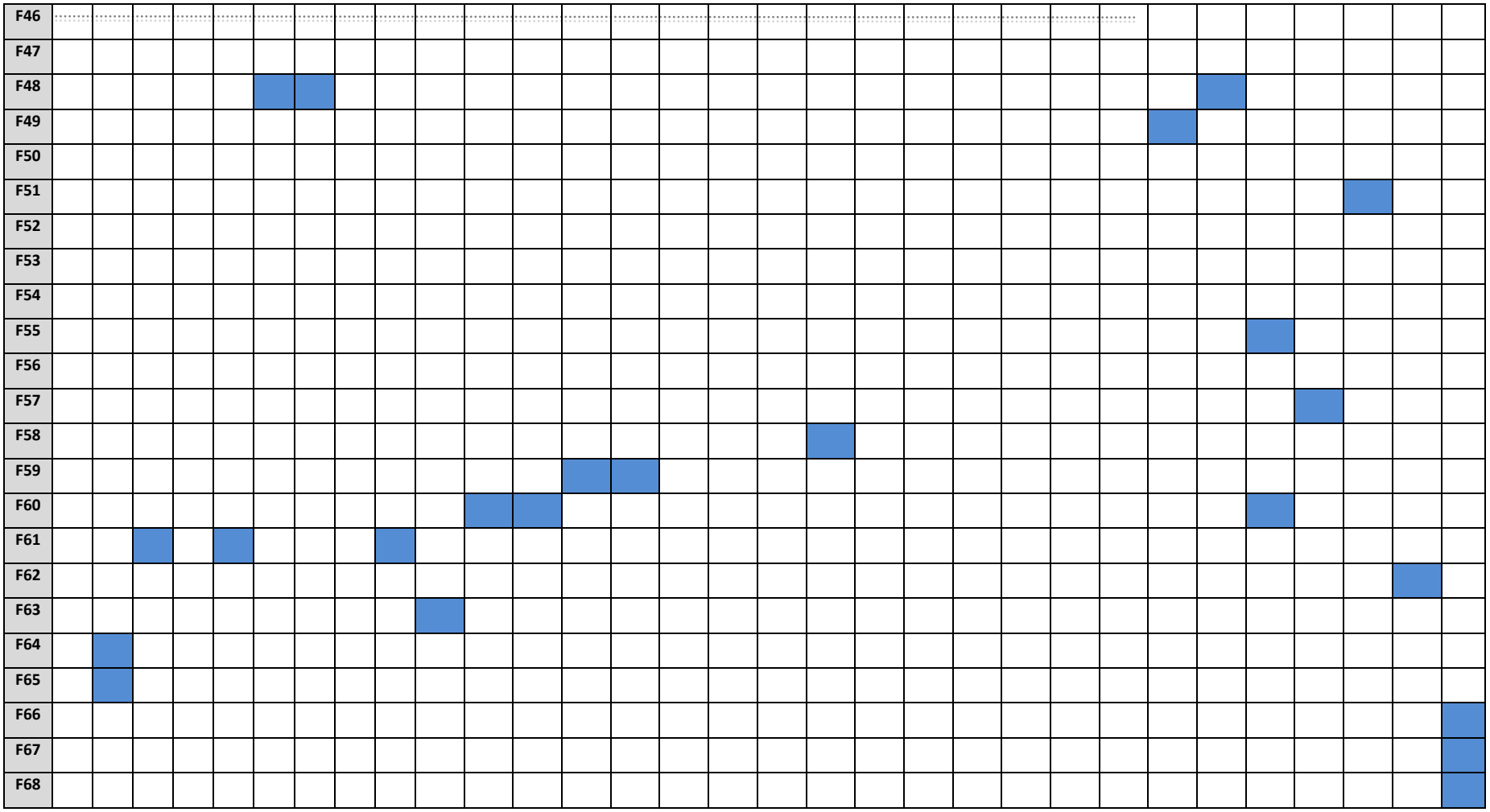


Table 27: Requirements coverage by LOD2 components features

5.Architecture

5.1 Data Model

"Active Hiring" platform manages essentially resumes and job vacancies descriptions. Few existing standards are available to represent and manage this information. HR-XML and Europass are two standards for representing resumes and job position openings descriptions. In the following sections, we provide a brief description of these formats.

5.1.1 HR-XML

HR-XML is an IT consortium of industrials to promote the use of xml standards and technologies in human resources industry. The consortium is composed of 52 members with major actors in HR applications like Oracle, SAP, Randstad US, CareerBuilder, Bundesanstal für Arbeit, First Advantage, ADP ...

HR-XML consortium focuses its efforts in developing interoperable standards for managing hiring/recruitment exchanges, payroll systems, talent & time management processes, healthcare benefits ... With respect to this, the consortium released a library called HR-XML that defines the xml framework targeted for the previous described use. HR-XML library is in its 3.1 release (with a 3.1.1 draft release in 26/04/2011).

HR-XML library is an xml-based framework for defining a set of common and standardized vocabularies necessary to make HR information systems interoperable. This framework is built on top of OAGIS (Open Application Group Integration Specification) framework, a standardized framework for business applications. In other words, HR-XML implements OAGIS vocabulary for HR domain and applications. The diagram below shows the different vertical framework applications that are built on the basis of OAGIS

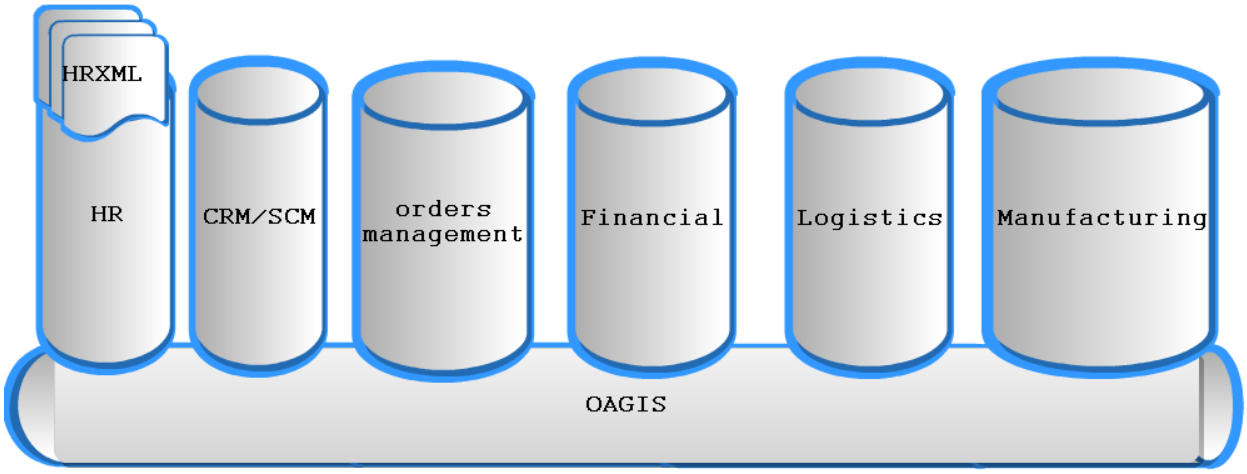


Figure 13: HRXML in the OAGIS framework

HR-XML library includes:

- XSD files representing the data model of the different resources and items involved in an HR application
- WSDL files specifying the services interfaces to implement for integrating different HR IS systems

In the HR domain, different actors can be involved through several HR applications. Each actor is associated to one or several roles depending on the goals they want to accomplish in the application. For example, in a classical recruiting process, three major actors and roles can be distinguished: the company recruiting (recruiting supplier), the job board assuring the interface between the company and the candidates (recruiting provider) and the candidate itself. Additional actors could be associated in the process like the assessment bodies for validating the candidate profile for example.

HR-XML framework covers 11 functional areas in the HR domain. Each functional area is associated with one or several packages containing WSDL interfaces to achieve the integration the services description describing the possible integration scenarios of HR applications related to:

5.1.1.1 Assessments:

This area covers different integration scenarios related to the evaluation process of a candidate to validate his profile and application. The package includes also the process of testing the candidate and validates his background.

5.1.1.2 Provisioning HR, Benefits, and Payroll Systems:

This area covers the communication of employee data from core HR systems to benefits and payroll service providers.

5.1.1.3 Talent Management Provisioning:

This area covers the management of the talent and competencies of employees.

5.1.1.4 Employee Performance Management:

This area covers the integration scenarios related to EPM (Employment Performance Management) systems. This is specifically targeted to manage employee development plans, results, objectives...

5.1.1.5 Recruiting:

HR-XML Consortium's very first standards project focused on the problem of sharing information about position openings and candidates among HR and recruiting systems. The package contains xml specifications of position openings and candidate resources like resume. A set of services are also specified to integrate job suppliers and job providers recruiting systems.

5.1.1.6 Savings Plan:

This area covers the HR-XML's Savings Plan Enrolment specification together with HR-XML's Payroll Instructions, Payroll Benefits Contributions, and Indicative Data specifications can be used in various combinations to support savings plan integration. The term "savings plan" is used broadly to refer to a category of programs under which participants accrue money in accounts typically funded through a mix of employee payroll deferrals and employer contributions.

5.1.1.7 Screening (Background Checks):

HR-XML Consortium's Screening specification defines a data format for requests to employment-screening service providers.

5.1.1.8 Staffing:

This area covers the specifications of staffing process (staffing resource, staffing order, staffing assignment…)

5.1.1.9 Stock:

To attract and retain employees, many employers offer employee stock plans. Two common types of plans are stock options and employee stock purchase plans (ESPP). The HR-XML Stock functional area was developed to manage the programs common in United States.

5.1.1.10 Time Card:

HR-XML's TimeCard allows the capture of time-worked data as well as information about associated expenses and allowances.

5.1.1.11 US Enrolment:

HR-XML's US Benefits Enrolment schema supports enrolment in "tier-based" benefit coverages (such as medical, dental, and vision) and in spending accounts (for example, flexible spending accounts or "FSAs"). As the name implies, the schema is designed around requirements principally found within the U.S. market.

5.1.2 Europass

Europass is a specification framework that is intended to help the European labour market to harmonize and understand the skills and qualifications of European citizens. The framework is designed as a portfolio of 5 documents defined and maintained by CEDEFOP (European centre for the development of Vocational Training). The 5 documents specifications are:

5.1.2.1 The curriculum vitae specification:

The curriculum vitae specification allows a candidate to describe her training aptitudes and skills acquired outside education, i.e, in a professional environment.

5.1.2.2 The language passport specification:

The language passport specification allows a candidate to describe her language skills with the support of a set of resources including predefined levels of language masteries.

5.1.2.3 The Europass mobility specification:

The Europass mobility specification is designed for recording skills acquired during a stay in another European country.

5.1.2.4 The certificate supplement specification:

The certificate supplement specification is designed to capture the skills acquired during training and certification programmes.

5.1.2.5 The diploma supplement specification:

The diploma supplement specification is developed for high graduates skills. It allows to describe the nature, level and the context of high graduation studies.

Since its launch in 2005, CEDEFOP claims 11.5 millions of CVs recorded in the Europass format.

5.2 Architecture

In this section, we present the architecture of the “Active Hiring” platform using different views according to the scenarios developed in section 2.4 on page.16. Each architecture show the LOD2 stack involved components in the use detailed in the corresponding scenario.

5.2.1 Active Hiring platform architecture for end user scenario:

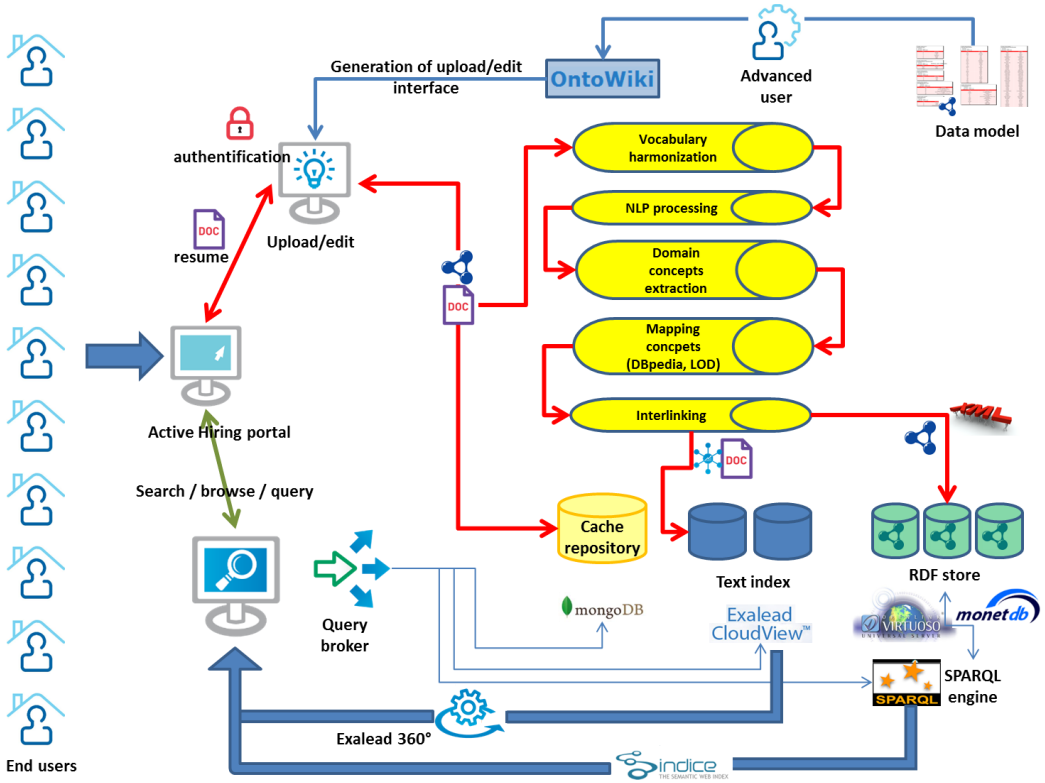


Figure 14: "Active Hiring" architecture for end users

Project funded by the European Commission within the Seventh Framework Programme (2007 – 2013)

Active Hiring platform architecture for advanced user scenario:

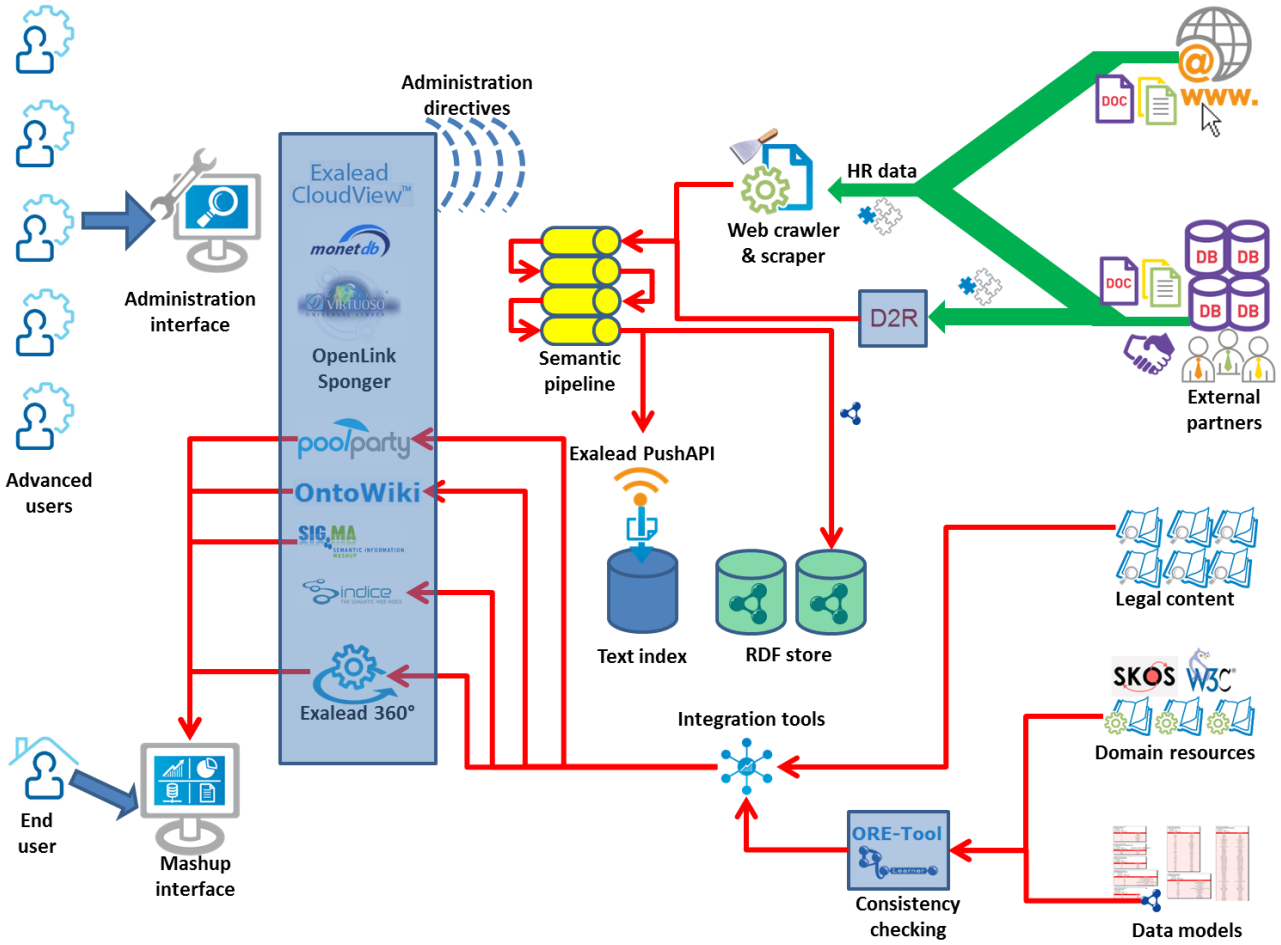


Figure 15: "Active Hiring" architecture for advanced users

5.2.2 Active Hiring platform architecture for developers' scenario:

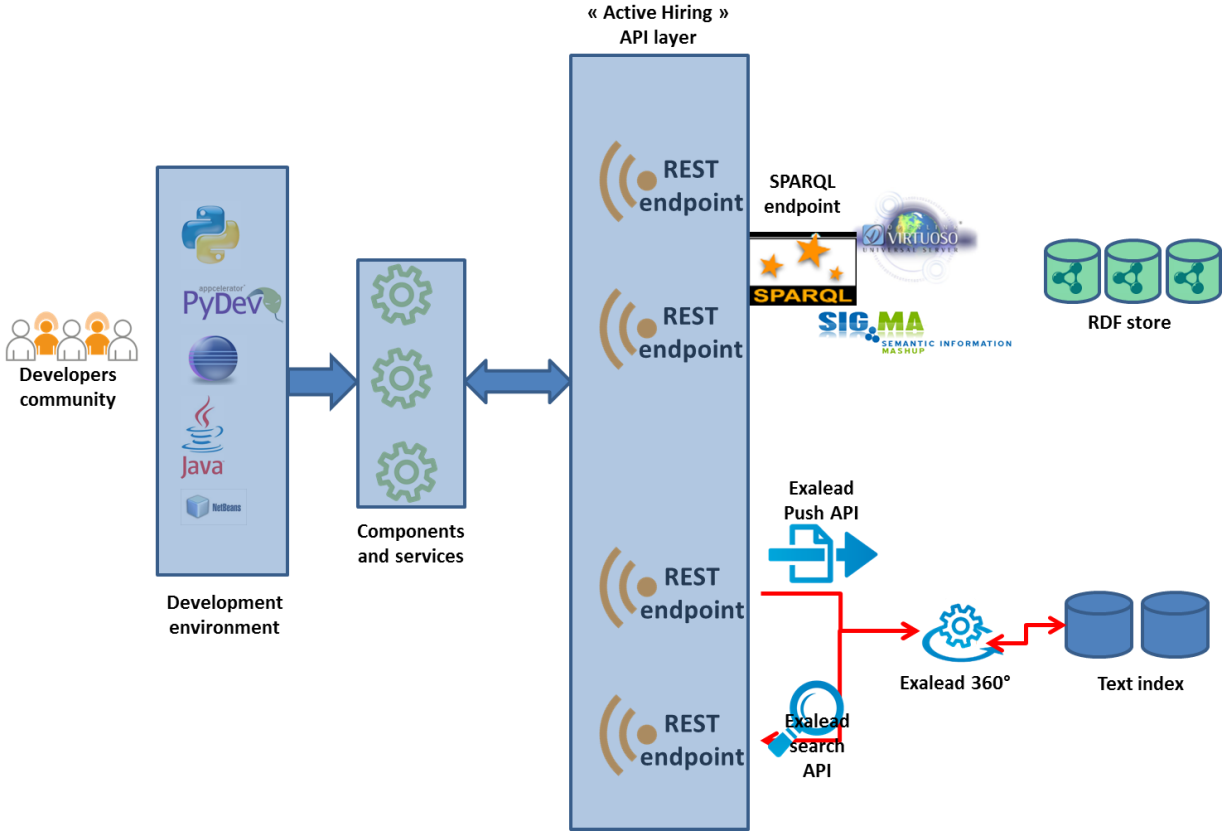


Figure 16: "Active Hiring" architecture for developers community

6.Conclusion

In this deliverable, we presented the functional requirements for implementing a business use case for data from the LOD cloud using real use situation of the targeted platform. We presented the features covered and provided by the different LOD2 stack components and provided a mapping between these features and the expressed requirements by the different use situations. Based on this mapping, we depicted an architecture of the system and the corresponding dataflow.

Project funded by the European Commission within the Seventh Framework Programme (2007 – 2013)