Collaborative Project

GeoKnow - Making the Web an Exploratory for Geospatial Knowledge

Project Number: 318159 **Start Date of Project:** 2012/12/01 **Duration:** 36 months

Deliverable 1.4.2

Intermediate Release of the GeoKnow Generator

Dissemination Level	Public
Due Date of Deliverable	Month 24, 30/11/2014
Actual Submission Date	Month 24, 30/11/2014
Work Package	WP1
Task	T1.4
Type	Prototype
Approval Status	Approved
Version	1.0
Number of Pages	30
Filename	D1.4.2 Intermediate release of the GeoKnow Generator.pdf

Abstract: This document reports the development of the GeoKnow Generator carried out in the second year of this project. Major features were developed such as user and graph access management, and the processes dashboard. Improvements on existing components and integration of new software components were also performed.

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

History

Version	Date	Reason	Revised by
0.1	05/11/2014	Initial draft of this document for internal contributions	Alejandra Garcia Rojas M.
0.2	18/11/2014	Reviewed and Provided Introduction and Conclutions	Daniel Hladky
0.5	15/11/2014	Added Screenshot and	Alejandra Garcia Rojas M.
0.3	24/11/2014	Internal and Peer review	Giorgos Giannopoulos
0.5	28/11/2014	Address peer review comments	Alejandra Garcia Rojas M.
1.0	28/11/2013	Approval	Daniel Hladky

Author List

Organization	Name	Contact Information
Ontos	Alejandra Garcia Rojas M.	alejandra.garciarojas@ontos.com
Ontos	Daniel Hladky	daniel.hladky@ontos.com

Executive Summary

The GeoKnow Generator is a stack of software tools that aim to support the creation and management of geospatial linked data. With this objective, GeoKnow collected a list of user requirements in the first year of the project, and started the development of several software tools that aims at satisfying those requirements. This deliverable presents efforts performed during the second year of GeoKnow by developing, improving and distributing software tools. The new features of the GeoKnow Generator Workbench, the integration Workbench of GeoKnow, are described in this document. And finally the contribution to the maintenance of the Linked Data Stack, a common software repository for the different EU projects.

Table of Contents

List of Figures

List of Tables

1 Introduction

The main contribution of GeoKnow resides in providing a stack of software components that give support to the creation of a geospatial dimension on the web of data, taking into consideration the size that this kind of data may imply. The development and distribution status of GeoKnow components are presented in this deliverable, however the benchmarking and performance reports are presented in D1.3.3 Continuous Report on Performance Evaluation due M24.

The task 1.4, follows the development of the GeoKnow software, and aligns them against the user requirements collected in the first months of this project[1]. We make sure that these components are accessible for any person interested on using them. As described in the previous version of this deliverable, GeoKnow is active contributor and maintainer of the Linked Data Stack[2], our distribution platform.

To demonstrate GeoKnow components and to fulfil a set of user requirements, we are also developing a Workbench that integrates different components to support users in the tasks of generating Linked data out of spatial data. The first prototype of the Workbench, delivered in M12, provided an initial prototype where several tools could be used for transforming, authoring, interlinking and visualising spatial data as linked data. In this deliverable we present the intermediate and public release of the GeoKnow Generator Workbench which implements most of the user requirements and brings this platform closer to achieve its objetives.

This document is organised as follows. Section 2 presents an overview of the user requirements with a traceability matrix that maps the component tools developed within GeoKnow to fulfil such requirements. Section 3 presents the development carried during the second year for the GeoKnow Generator Workbench as the integration component of the Generator. This section also describes the current installation and configuration procedures. In Section 4 the Linked Data Stack, is presented with the GeoKnow contributions as part of task 1.4. Finally, the conclusions and the planned roadmap for next year is presented in 5.

[1] http://svn.aksw.org/projects/GeoKnow/Deliverables/D1.1.1/D1.1.1_Initial_Common_Requirements_Specification.pdf

[2] http://stack.linkeddata.org/

2 Software Requirements Specification Review

For reporting second activities we provide a summary of user requirements and concerning software components that have being developed or improved in order to satisfy initial user requirements.

Some user requirements concern specific component tools such as Facete, FAGI, etc. and have been tackled in the different work packages. Whereas other requirements concern the Generator Workbench as an integration of components tools. The following subsections will present a traceability matrix of requirements and the respective components that satisfy them. A complete list of User Requirements is presented in the Appendix A for reader's reference.

2.1 Component Tool Requirements traceability matrix

In table 1 we present the list of released components and ongoing developments within GeoKnow. Each component is linked to the requirement they aim at satisfying and the release/development status. In short, GeoKnow released 4 new components this year (second year) and 6 new versions of existing components[3]. This table also presents some components that were not developed within GeoKnow but that are used to meet specific requirements.

Table 1: Generator components requirements traceability matrix

Component	WP	Description	User Req. Mapping	Released
Deer (previously GeoLift)	3	Framework to enrich geographic and other content in the Semantic Web	FR-U4, FR-S9	Released second year
GeoLiftService	1	A web service for GeoLift	IR-B1	Released first year
TripleGeo	3	Tool for converting data in geospatial formats into RDF triples	FR-B5	Released first year
TripleGeoService	3	A web service for TripleGeo	IR-B1	Released first year
LIMES	3	Link discovery framework	FR-U4, FR-S9	Released first year
Mappify	4	Js snippet generator	FR-D3	Released first year
OntoWiki	LOD2	Authoring tool of semantic content, with an inline editing mode for editing RDF content, similar	FR-U3, FR-B9, SR-B5	External tool
Rsine	LOD2	Subscription and notification service	FR-D5	External tool
Virtuoso	2	RDF Server	FR-S1, PR-U3	New version released second year
Facete2	4	Geospatial RDF browser	FR-D1	Version 2.0 released in second year

[3]Considering that Deer is a new version of GeoLift

Sparqlify		RDB-RDF mapper	FR-S9	New version released second year
LimesService	1	A web service for Limes	IR-B1	New version released second year
FAGI-gis	3	Tool for fusing geospatial features of RDF data	FR-U4, FR-U8	Released second year
RDF Data Cube Validation Tool	3	Quality Analysis and Repair of RDF Data Cubes	FR-E2, FR-U8	Released second year
ESTA-LD	4	Exploratory Spatiotemporal Analysis	FR-E1, FR-D1	Released second year
GEM	4	Mobile spatial-semantic visualization, exploration and authoring tool	FR-D1	Released second year.
FAGI-tr	3	Tool for aligning RDF Vocabularies	FR-U4, FR-U8	To be released in the third year

2.2 GeoKnow Generator Workbench System Requirements

A significant part of the GeoKnow Generator development was dedicated to the Workbench, which is a integration software tool aiming to be a common access point to the different components previously listed, and to provide functionality on top of them. The design of Workbench is based on list of system requirements derived from the D1.4.1 Initial prototype of the GeoKnow Generator. Table 2 presents the status of those requirements, identifying the year when they were implemented in the Workbench. NA status stands for requirements that have Not being Addressed yet.

Table 2: Generator Workbench System Requirements

Id	Description	User Req. Mapping	Status
1	The systems shall provide preconfigured access to all datasets described in the DoW	DR-D6	Initial Protptype can be preconfigured with information of existing Endpoints
2	The system shall be able to access public RDF data sources: SPARQLendpoint, ckan, LinkedGeoData, etc	FR-U3, FR-S9, FR-B5	The Initial Protptype can access and extract data from external endpoints
3	The system shall use be able to access internal data sources: SPARQLendpoint, sftp, databases, RDF files, CSV files	FR-U3, FR-S9, FR-B5	The Initial Protptype can access internal endpoints, databaes and the user can import and RDF files from the file system or from a URL

4	The system shall be able to convert CSV and RDB data to RDF	FR-U3, FR-S9, FR-B5, DR-B10	The Initial Protptype integrated Sparlqify
5	The system shall provide the ability of importing subsets of data from endpoints/graphs	FR-U6, FR-B8	The Initial prototype provided Import functionality from RDF endpoints using CONSTRUCT query
7	The system shall allow storing the resulting dataset into a specific graph	SR-B5	The initial prototype provided this feature to TripleGeo, Limes, and Geolift Services
8	The system shall produce a log of each of the tasks performed	FR-U2	The first release provides status of processing tasks in the Dashboard
9	The system shall produce a log of each of the tasks performed with a given pattern and in the different levels	FR-U2	NA
10	The system shall indicate the provenance of the resulting data using PROV-O and/or pav-ontology for instance	DR-U1, DR-U2	NA
11	The System shall provide a super user for administration pruposes	FR-B6	First releas provide admin user
12	Indicate the status of a given dataset (availability)	FR-B7	NA
13	The system shall allow administration of datasets to to define access roles	SR-B4, FR-B6	Firs release provides role management features
14	The system shall support secure communication	SR-B6	NA
15	User shall be able to login to the system support some SSO Authentication	FR-B1 SR-S2 FR-B3	First release provide support basic authentication
16	Users shall be able to assign following right access to data sources: no-acces, read-only or read+write (for instance) to specific users	FR-B6, SR-B4	In the fist release users can define public or private datasets
17	The system shall provide a dashboard with status of processing tasks and being able to do some execution operations	FR-R1, FR-R2	First release provides a dashboard for executing and monitoring tasks

Previous tables presented User Requirements traceability matrix among the GeoKnow Generator components. The requirements were extended with comments in the first review meeting and with the extension of the DoW. We can summarise that, in these 2 years, out of 56 reviewed requirements, 37 requirements have being tackled at least by one of the components developed in GeoKnow including the Workbench; and 19 are still unmet and to be reviewed in the coming year.

Next section describes in detail the development carried out this second year in the Generator Workbench, which is now officially the first public release.

3 GeoKnow Generator Workbench

3.1 Second Year Achievements

The development carried out this second year is basically focused in the enhancement of management features of the generator such as user authentication and management, data access control and process status monitoring. In this section we describe and discuss the implementation of such features.

3.1.1 User Authentication and Management

According to user requirements the GeoKnow Generator shall provide a Single Sign On (SSO) functionality. Since the Generator Workbench is the entry point of several software tools, this SSO has being implemented using basic authentication. This authentication corresponds to the use of the Workbench and the access to use different tools via the Workbench, but some tools may require a second layer of authentication, for instance Virtuoso. For the release version, part of the configuration of Workbench includes to provide a user name, password and email to create an administrator user at setup stage, which will have all rights to configure the Workbench. The administrator can create users for the Workbench using a UI in the front-end. However, users can also register themselves by using an email based registration system. For this purpose the Workbench has to be configured before start up by providing an email account and email server to be used. More configuration details are described in Section 3.2.2. Besides User management, we have also implemented role management functionality. The administrator can create different roles and assign to those roles specific access to the different tools. Section 3.2.4 presents the actual interface of these management features.

3.1.2 Data Access Management

Having the possibility of handling private and public data together is one of the most important user requirements for the system. To achieve this, we make use of Virtuoso security graph[4]. For the Workbench users to have access to the RDFStore access control, the Workbench creates a user in the RDF Store which is associated to the Workbench user. Figure 1, presents the authentication/authorisation schema between the user, the Workbench and Virtuoso Store.

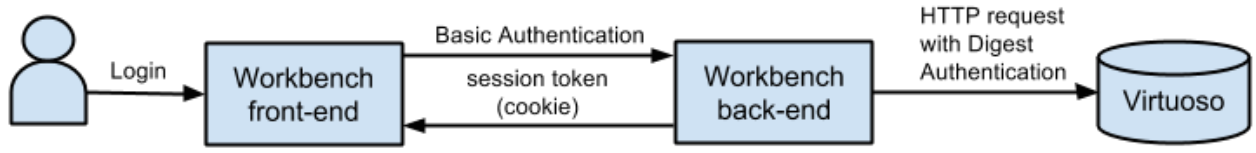

Figure 1: Workbench Authentication Sketch

Virtuoso access control functionality allows managing access to different users at graph level and at group of graphs level. In the Workbench, by default, Virtuoso is configured to have no access to any graph, so accesible graphs have to be set explicitly public by the user. Virtuoso control access to graphs is specified by four bits of an integer "permission bit-mask", plain old UNIX style:

- Bit 1 permits read access.
- Bit 2 permits write access via SPARUL and it's basically useless without bit 1 set.

[4] http://docs.openlinksw.com/virtuoso/rdfgraphsecurity.html

- Bit 4 permits write access via "RDF sponge" methods and it's basically useless without bits 1 and 2 set.
- Bit 8 allows to obtain list of members of graph group; an IRI can be used as graph IRI and as graph group IRI at the same time so bit 8 can be freely combined with any of bits 1, 2 or 4.

These access control are specified by the users for each graph created in the Workbench.

One of challenges in adapting access control to graphs was the possibility to allow components to have access to private user graphs and, thus, to be able to process data (read or write). The ideal scenario would be that the tools could provide some support for authentication (i.e. Jena authentication[5]). However, this is not the case for all the tools. To solve this issue, a proxy endpoint service was added to the Workbench for allowing component tools to perform SPARQL queries.

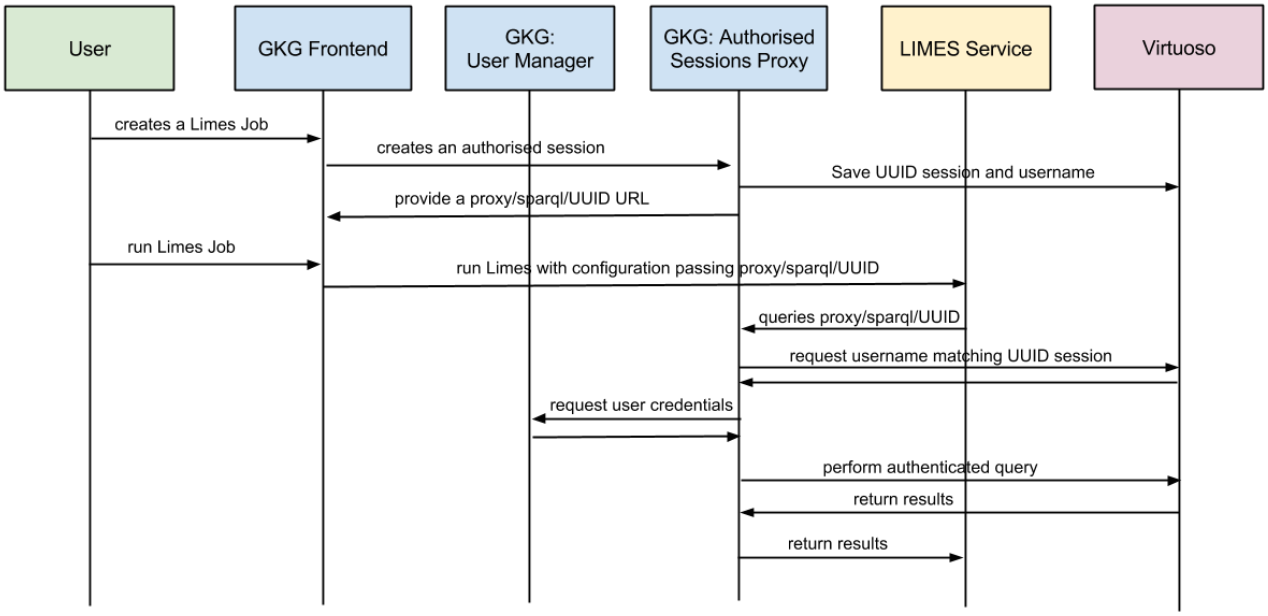

Figure 2: Workbench Authentication Sketch

At the moment the user has configured a process for a component (Limes, TripleGeo, etc.), the Workbench replaces all URL endpoints that correspond to the backend Virtuoso store (e.g http://10.0.0.8:8890/auth-sparql) with a URL that points to the proxy (e.g http://generator.geoknow.eu/rest/session/1e4d6600-6e52-11e4-9803-0800200c9a66). This URL includes a UUID that will represent an authorised session for the tool created by the user to perform read/write tasks. Therefore, the component will use this proxied URL as an endpoint to perform required SPARQL queries. The proxy matches that UUID with the user that created a session for the tool, and retrieves the corresponding credentials for Virtuoso. Figure 2 presents a sequence diagram to depict the creation process of an authorised session and how the component tool uses the proxy to perform SPARQL queries. Limes Service is not aware of any authentication, it just performs required queries to the proxied URL, which is in charge of authenticating the user. The way of executing a job is abstracted for better describing the authorised session functionality. Jobs executions are part of the Dashboard processing feature and is described in the next section.

Finally, for securing the access to the proxy, a Cross-origin resource sharing (CORS) is enabled, so the proxy service is allowed to serve requests only from address where components have being installed. These URLs are provided in the configuration of the Workbench (see section 3.2.2).

[5] http://jena.apache.org/documentation/query/http-auth.html

3.1.3 Processing Dashboard

A dashboard for the Workbench has two main purposes. First it aims at providing feedback to the user about processes that he may have started. Second, it allows users to configure which LD lifecycle processing they want to execute on their data and assign the batch processing tasks to the Dashboard, without further interaction being required. For this release only the first objective is provided.

A deep research was done this year for choosing the most adequate tool for implementing such a dashboard. Among Extract-Transform-Load (ETL) frameworks, Unifiedviews[6], recently released this year, is a Linked Data processing framework created under the EU project COSMODE[7]. Unifiedviews has being deployed using components developed in other EU projects such as LOD2[8]. Implementing a Data Processing Units is required in order to work with a new software component, which results in a tightly coupled architecture. This tool is an integration platform itself and basically at the same level as our Workbench, with no authentication or authorisation features; however, it represents a relevant reference point for the GeoKnow.

Therefore, we also investigated software tools that could be integrated in the Workbench. The Web Services Business Process Execution Language, commonly known as BPEL, is an OASIS[9] standard executable language for specifying actions within business processes with web services. BPEL tools were conceptually good candidates for the dashboard because they encapsulated the concept of Web Services and provided support for workflow creation. In practice, these tools support mainly SOAP protocols and, in theory, RESTful protocol too. However, RESTful features are still unstable within the processing engines available in the web i.e. Apache ODE[10]. Batch processing tools, such Spring Batch[11] where another possible candidate. Spring batch is an implementation of JSR-352, a Java-based Batch Application for the Java Platform (JSR 352), which provides support for defining, implementing, and running batch jobs. Within the JSR 352, jobs are tasks that can be executed without user interaction. The batch framework is composed of a job specification language based on XML, a Java API, and a batch runtime.

Spring batch is mainly file oriented batch processing tool, but it is also possible to wrap a Web Service within the batch implementation. Therefore, we adapted the Workbench to use Spring batch for the dashboard, with the objective of being able to replace it with a BPEL tool whenever a stable implementation is available.

Spring batch provides a web-based interface called spring-batch-admin[12] which provides information corresponding to the Jobs configurations, job executions and executions status. The Workbench uses a adapted version of spring-batch-admin available at: https://github.com/GeoKnow/spring-batch-admin. This version contains the corresponding implementation for wrapping web services: the ServiceTasklet. This class is an implementation of a Tasklet concept[13] , which is different to Chunk-oriented processing because it expects to have only one single process. The ServiceTasklet receives as parameters the basic information to perform HTTP requests. Thus is a generic implementation that can be used to call any Web Service, and then used for executing all processes within the Workbench.

As described in the sequence diagram presented in Figure 3, when the user configures a Job using the Interface provided in the tools (i.e. Limes, Geolift, TripleGeo), the Workbench is in charge of creating XML definitions of batch Jobs and of registering them to the batch-admin service. The job's metadata is stored in a specific graph (jobsGraph) in the RDF store. Therefore, the user can execute the job in the Dashboard interface,

[6] http://www.unifiedviews.eu/
[7] http://www.comsode.eu/
[8] http://lod2.eu/
[9] http://en.wikipedia.org/wiki/OASIS_(organization)
[10] http://ode.apache.org/extensions/restful-bpel-part-i.html
[11] http://projects.spring.io/spring-batch/
[12] https://github.com/spring-projects/spring-batch-admin
[13] http://docs.spring.io/spring-batch/trunk/reference/html/configureStep.html

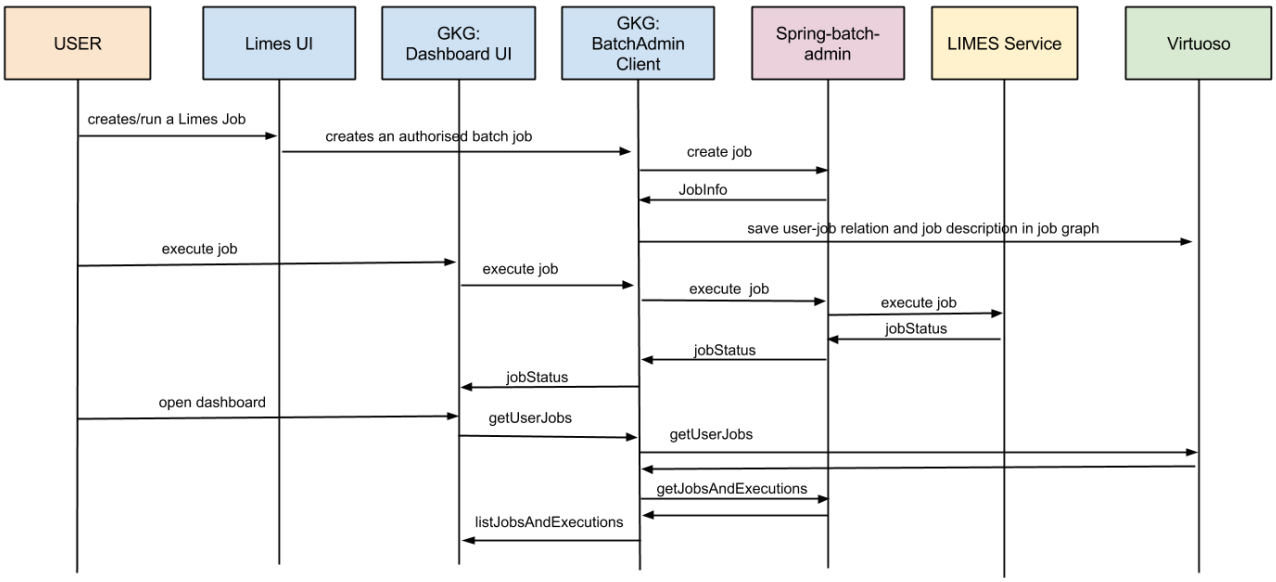

Figure 3: Sequence diagram for User's Dashboard

which will be executed in the Spring-Batch-Admin to monitor its execution. Finally the user can visualise configured jobs and the status of executions in the dashboard. In section 3.2.5, a screenshot of the dashboard is presented.

This section provided a detailed description of the development carried out this year for the GeoKnow Generator Workbench. Next section will present the actual status of the Workbench for this first release.

3.2 First Release

The first release of the Generator Workbench delivered in this report integrates most of the tools developed within GeoKnow. For review purposes we provide a online demo at http://generator.geoknow.eu:8080/generator/. The user/password of administrator users are admin/demo2014. However, it is also possible to register with a new user, it is only required a email registration[14].

The Workbench also provides Single Sign On functionality, user and role management and data access control for the different users. Figure 4 presents the actual architecture of the Workbench. The Workbench is comprised of a front-end and back-end implementations. The front-end provides GUIs for software components where a REST API is available (LimesService, GeoLiftService and TripleGeoService). Components that provide their own GUI, are integrated using containers (FAGI-gis, OntoWiki, Mappify, Sparqlify and Virtuoso SPARQL query interface). The front-end also provides GUIs for the administrative features like users and roles management, data source management and graphs management, as well as the Dashboard GUI. The Dashboard provides a visual feedback to the user with the registered jobs and the status of executions. The Workbench back-end provides REST interfaces for management of users, roles, graphs, datasources and batch jobs, for retrieving the system configuration, and for importing RDF data. All system information is stored in Virtuoso RDF store.

The Workbench is distributed as a component in the Linked Data Stack[15]. All information to get started is provided here: http://stack.linkeddata.org/documentation/users-guide/geoknow-generator/ and described in detail next.

[14]For demo purposes all new users are administrators. Personal information such as the e-mail provided to register, will not be used by GeoKnow and will be deleted at some point

[15]http://stack.linkeddata.org/

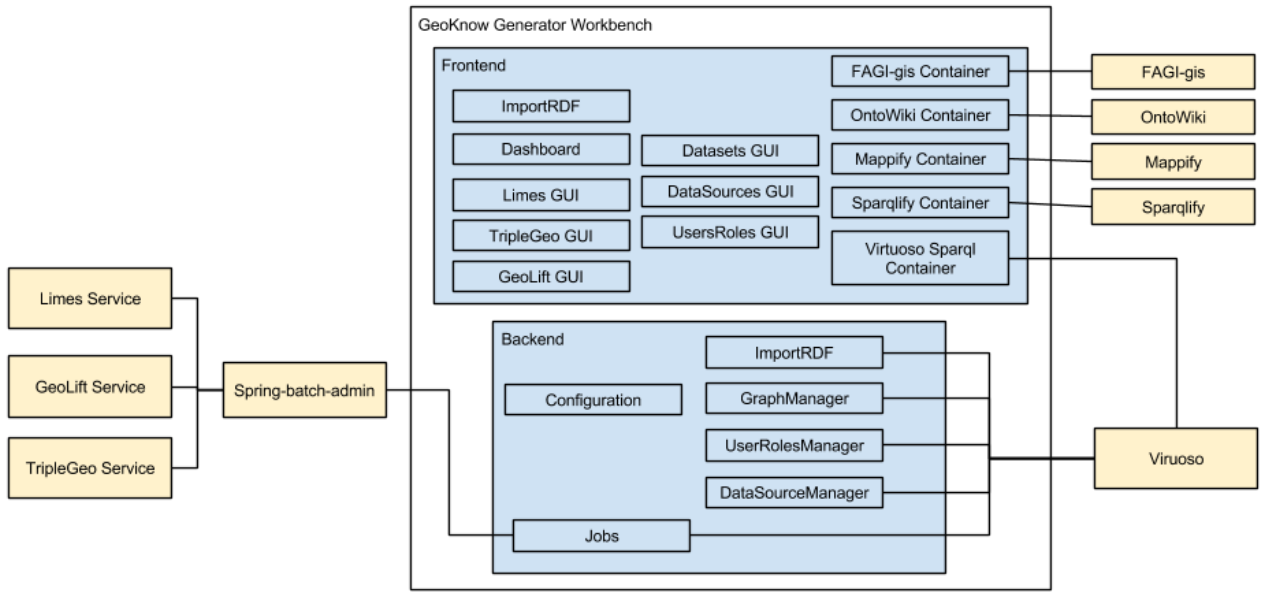

Figure 4: GeoKnow Generator Workbench Architecture

3.2.1 Virtual Machine

The GeoKnow Generator Workbench provides a preconfigured demonstrator of the first prototype. An image of Ubuntu 12.04 in OVF format can be downloaded here: http://generator.geoknow.eu/geovm/generator.ova. This VM can be installed in several is a virtualisation software such as VMWare or VirtualBox.

3.2.2 Debian Installation and Configuration

The installation of the Workbench can be done using the Debian package, which includes a local pre-configured installation and relieves the user from following all configuration steps.

To install the generator from the command line you may type:

```
sudo apt-get install geoknow-generator-ui
```

This will install the Generator Workbench on a Tomcat7 (which will be installed if not already in the system). Currently the Generator Workbench does not include any component yet, and the user has to install each component separately.

The Stack components that are integrated and can be installed are:

```
sudo apt-get install virtuoso-opensource
sudo apt-get install ontowiki-virtuoso
sudo apt-get install facete2
sudo apt-get install sparqlify
sudo apt-get install mappify
sudo apt-get install geolift-service
sudo apt-get install limes-service
sudo apt-get install triplegeo-service
sudo apt-get install fagi-gis
```

After the installation the GeoKnow Generator demo and components should be visible at

http://localhost:8080/generator, and the default user/password to login as administrator is admin/admin.

3.2.3 Installation from source

The GeoKnow Generator Workbench source code is available in GitHub:

`https://github.com/GeoKnow/GeoKnowGeneratorUI`

The code can be cloned or downloaded from this repository. To configure the Workbench it is required to have preinstalled Virtuoso 7.x and the Database Administrator (DBA) credentials. Then, the user have to proceed with the following configuration steps.

3.2.3.1 Framework configuration template

Virtuoso user information needs to be provided into the configuration file. This file is located in `main/src/resources/framework-configuration-template.ttl`. The following code listing shows where to add this information:

```
:VirtuosoConductor
  a  lds:StorageService ;
  lds:password "***DBA_USER***"^^xsd:string ;
  lds:serviceUrl <http://***VIRTUOSO_IP***:8890/conductor> ;
  lds:connectionString "jdbc:virtuoso://***VIRTUOSO_IP***:1111/"^^xsd:
     string;
  lds:user   "***DBA_PASSWORD***"^^xsd:string .
```

This file also contains information for the user that will be created as a system user in the Virtuoso store. This user is required for managing control access to users and graphs. This information is described under the `VirtuosoAuthSPARQLEndpoint` resource in the configuration file presented in the following listing:

```
:VirtuosoAuthSPARQLEndpoint
  a       lds:SecuredSPARQLEndPointService, void:Dataset , gkg:
     SPARQLEndpoint , gkg:DataSource ;
  rdfs:label  "Workbench Authenticated Endpoint" ;
  lds:password "generator"^^xsd:string ;
  void:sparqlEndpoint <http://***VIRTUOSO_IP***:8890/sparql-auth> ;
  lds:serviceUrl <http://***VIRTUOSO_IP***:8890/sparql-auth> ;
  lds:user   "generator"^^xsd:string .
```

The configuration file also contains the system graphs description required to save information for the different functionalities. These are:

settingsGraph : contains configuration information such as named graphs and components descriptions

initialSettingsGraph : is a settings template provided for the creation of new users

accountsGraph : contains the information of users

groupsGraph : contains metadata for group graphs

jobsGraph : contains information about the configured jobs and owners

sessionsGraph : contains the created authorised sessions and owners relation

Finally, in this configuration file an administrator user name and password has to be configured by providing a valid email:

```
:admin
  a gkg:Account;
  foaf:accountName "admin"^^xsd:string;
  lds:password "admin"^^xsd:string;
  foaf:mbox <mailto:***REMOVED***>;
  gkg:Role gkg:Administrator.
```

3.2.3.2 Component configuration

The configuration of components follows the same logic presented in the previous deliverable, which is using the Linked Data Stack Integration schema[16]. If the user needs to modify the URL where component services are installed, he/she has to modify the URLs in the file located in `main/src/resources/framework-components.ttl`. An example of the Mappify service configuration is presented next:

```
:Mappify
      a       lds:StackComponent ;
      rdfs:label "Mappify"^^xsd:string ;
      lds:providesService :MappifyService ;
      lds:version "0.9.10-1"^^xsd:string ;
      foaf:homepage <https://github.com/GeoKnow/Mappify> .
:MappifyService
      a       lds:ExplorationService ;
      lds:serviceUrl <http://localhost:8080/mappify2/> .
```

3.2.4 Post-installation Settings

Most of the configurable settings in the Workbench were presented in the initial prototype. This first release includes the enhanced Graph Management with authorisation access, and the User and Role Management. The following section describes configurable settings of the Workbench which can be navigated to by clicking on *Settings* on the top navigation bar.

3.2.4.1 Data Sources

In the *Data Sources* settings has remained the same since the initial prototype. In this interface users can manage SPARQL endpoints and Relational Databases connection info that can be used by certain components within the Workbench.

3.2.4.2 Datasets

In the *Datasets* settings section it is possible to create graphs and graph groups within the graph store. These are necessary for organising and storing triples.

[16] https://github.com/LinkedDataStack/ldis-schema/blob/master/ldsi-schema.ttl

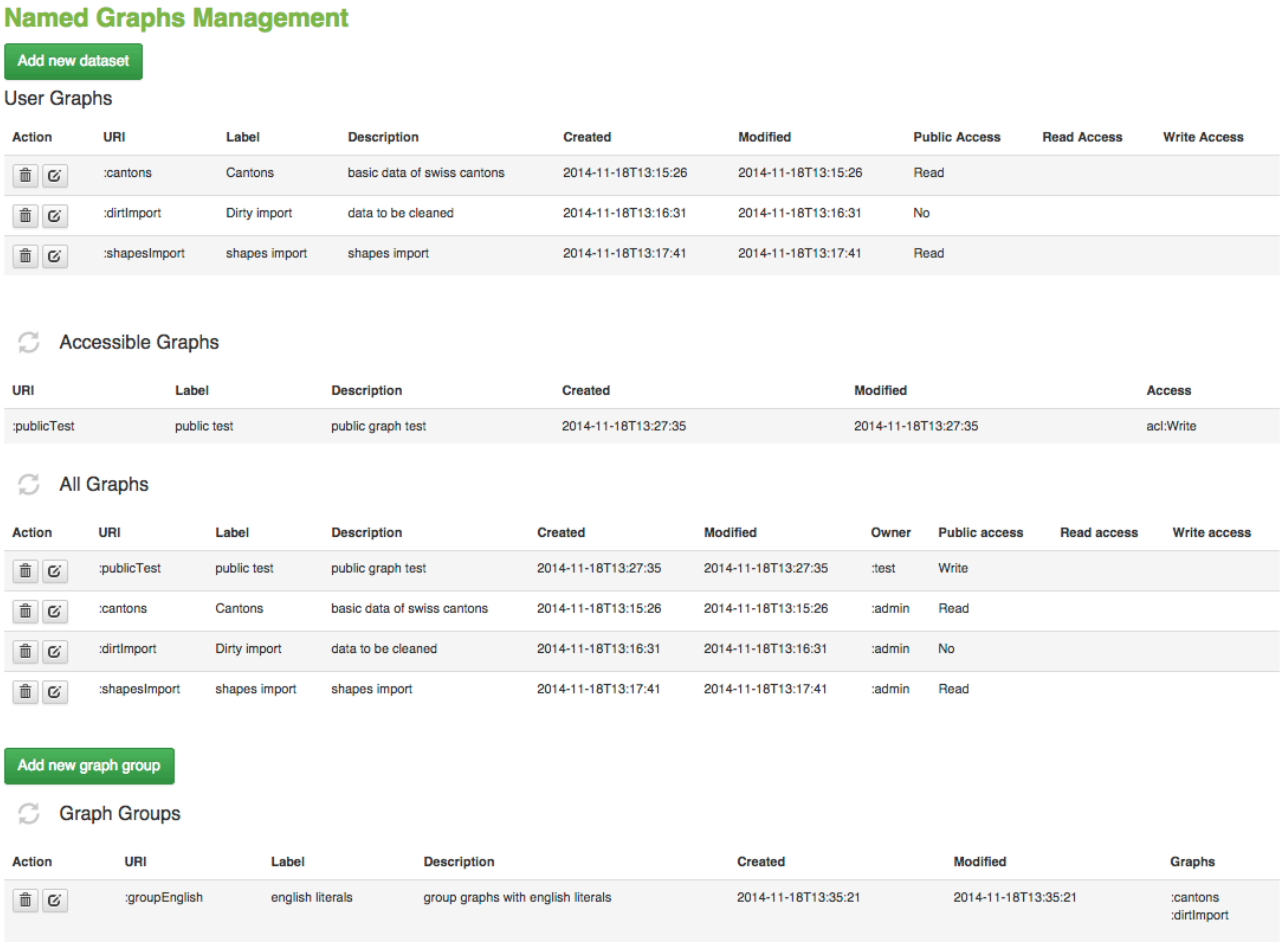

Figure 5: Datasets GUI

In the Graph management interface (Figure 5) the user can visualise, add, edit or delete a Dataset or Graph. For creating a new graph (see Figure 6), the user needs to provide a name, label and description for identification purposes. The user can also define the access control of the graph. By selecting *No, Read or Write* access from the *Access* dropdown menu, access for any user with access to the SPARQL endpoint for the graph store (i.e. the public if the SPARQL endpoint is available over the internet) can be configured. It is further possible to control access to graphs for registered users of the Workbench. Various configurations are possible, for example, a graph could be configured with *Read* access for all users with access to the SPARQL endpoint, while another registered user could be granted *Read* and *Write* access.

It is also possible to add Graph Groups to organise datasets into groups, as depicted in the Figure 5 . This Graph groping functionality, based on Virtuoso, allows to the user to perform SELECT queries for a list of named graphs (those provided in the group), and can be quite useful.

3.2.4.3 Components

It is currently not possible to directly manage integrated components within the Workbench, and it should be provided at installation phase as described in the previous section. The *Components* page just shows which components are integrated and how they are configured.

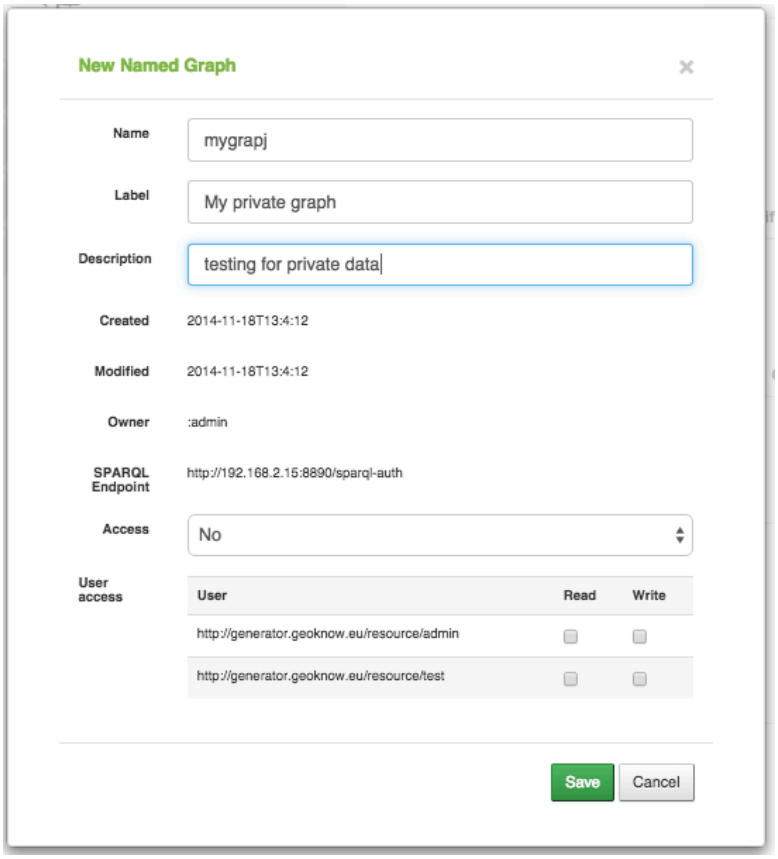

Figure 6: Creating a new graph

3.2.4.4 Users

The *Users and Roles Management* interface, depicted in Figure 7, can only be accessed by an administrator. This allows to define kinds of roles for registered users and configure role for non-registered users of the Workbench. The administrator can define the components that can accessed and executed by users with given role.

A *Default* role can also be defined, which is the role that will be given to new users. The *Not logged in user* option defines what role a user that has no user account will be assigned.

By default the new role has no access rights for any components. The access rights can be set by placing ticks next to the relevant component in the columns for each user type.

3.2.5 Generator Workbench

The Workbench section provides access to components in the left menu. This menu has remained the same as the initial prototype. Therefore, is not worth to describe in detail all components again, this is already available in D1.4.1 Initial protptype of the GeoKnow Generator[17]. The landing page of the Workbench section is the Dashboard, depicted in Figure 8. In this page the user can see at first sight all posted jobs for processing. Then, he can navigate into each job by clicking on it, and visualise the status of processes. When the user clicks

[17] http://svn.aksw.org/projects/GeoKnow/Deliverables/D1.4.1/D1.4.1_Initial_protptype_of_the_GeoKnow_Generator.pdf

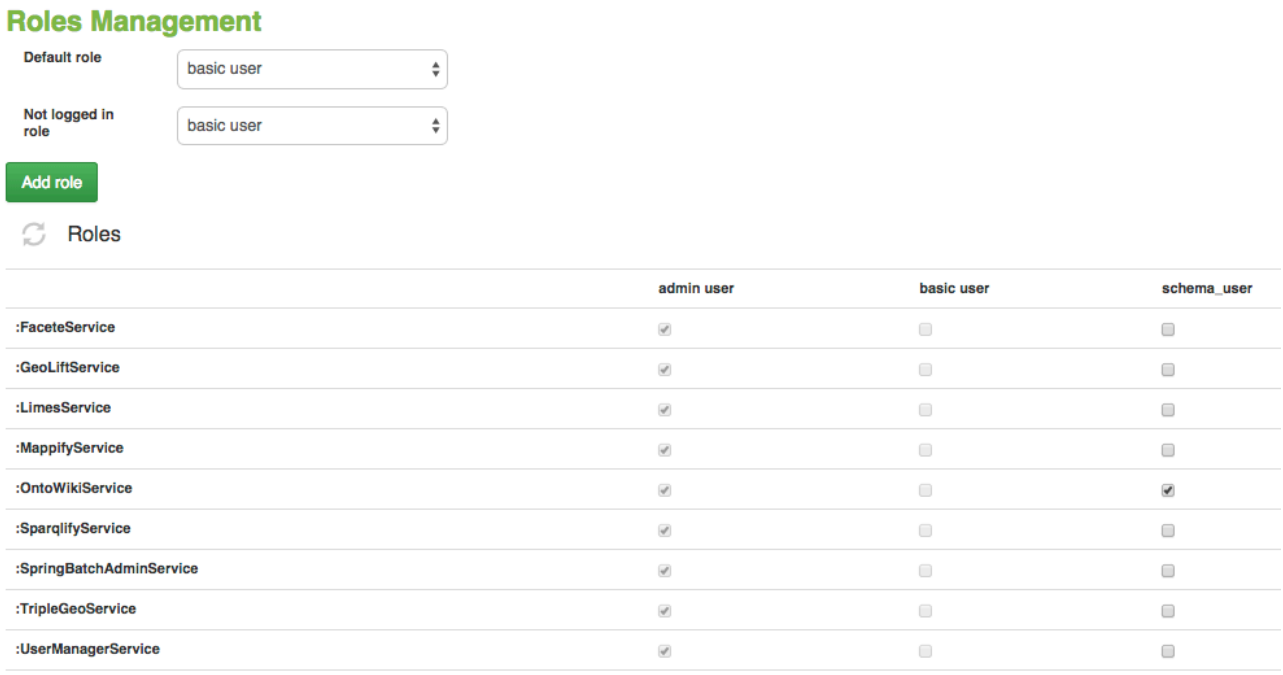

Figure 7: Users and roles management GUI

on a job, a drill down navigation will open that show the actions that can be performed (run/delete/refresh) and the status of the different executions.

To be able to register jobs, we have made available a RESTful interface that can be used by the different components. It is required that the user is authenticated in order to interact with this interface. Table 3 describes the available REST services for batch jobs. The list of services are also available trough the Web Application Description Language (WADL) in the generator demo at http://generator.geoknow.eu:8080/generator/rest/application.wadl.

We have implemented for the LIMES service the possibility of adding jobs when a Limes configuration is created. Thus, once the user has finished to configure a Limes job, he can create the Job, and which will be registered and accessible for launching in the Dashboard. Figure 9 presents the dialog window the user gets for registering a job. He has to provide a unique identifier (the system already provides a suggestion) and the description of the job.

For the moment the batch processing is limited to create one-step jobs. However, with spring-batch, it is possible to create jobs of several steps, for instance a processing workflow could be defined by importing RDF data, transforming some shape to RDF format, linking with public datasets, etc. To be able to implement such workflow we still need to create the corresponding user interfaces. Therefore, for the next year we plan to extend the Dashboard to allow users to configure simple workflows of several steps.

3.2.6 REST services

The GeoKnow Generator Workbench have being complemented with RESTful API, to facilitate the interaction with other components. Table 3 presents the list of services currently available. These services cover the authorised sessions and the batch jobs. However this API has to be enhanced to provide authentication and graph management too.

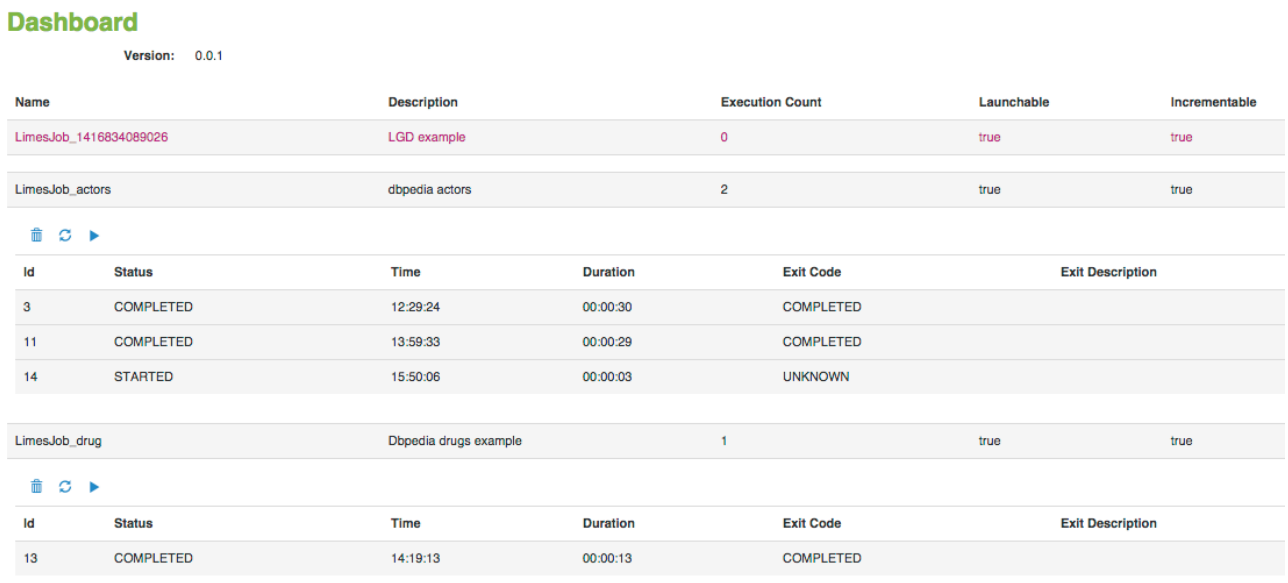

Figure 8: Dashboard GUI

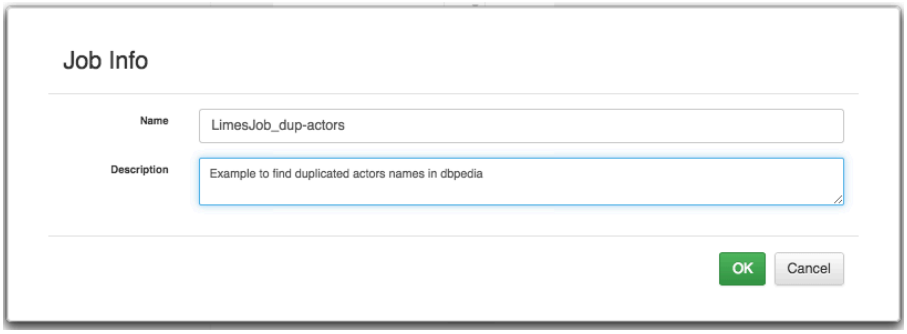

Figure 9: Registering a Job GUI

Table 3: Rest services for batch jobs

Path	Method	Description
rest\config	GET	Provides the configuration information of the Workbench
rest\jobs	GET	Retrieves all the jobs of the registered user.
rest\jobs\{jobName}	GET	Retrieves the job description and its executions
rest\jobs\{jobName}	DELETE	Deletes a job
rest\jobs\{jobName}	PUT	Registers a new job with the description passed in the body object
rest\jobs\{jobName}\run	POST	Executes a job
rest\session	PUT	Creates a new authorised session with the credentials of the user that is currently registered

rest\session\{UUID}	POST	Proxies a SPARQL Endpoint post request
rest\session\{UUID}	GET	Proxies a SPARQL Endpoint get request
rest\session\{UUID}	DELETE	Deletes the authorised session

3.3 Documentation

The documentation has being created for the Workbench users and it is available here: http://stack.linkeddata.org/documentation/users-guide/geoknow-generator/. This page has the link to three video tutorials of the Workbench and some components: Limes, TripeGeo, Sparqlify and GeoLift. These video tutorials are in a dedicated Youtube channel: https://www.youtube.com/playlist?list=PLXOobQ2x425GbU6oP-8PHgNhv-_EeTqNA

This section has described the main features developed for the first release of the GeoKnow Generator Workbench. Next section describes the Linked Data Stack as part of the work performed in this task with the objective of distributing the software developed within the project.

4 Linked Data Stack

One of the tasks GeoKnow team was committed last year, was the support in the maintenance of the Linked Data Stack (LDStack) on behalf of the LOD2 project[18], which ended in September 2014. An official hand-over was announced during the Semantics Conference[19], where one of the GeoKnow representatives assisted to the *Linked Data Stack: Tools and Apps* talk given by LOD2 team. The support to the LDStack was also announced in our blog[20].

This section provides activities around the Linked Data Stack performed this year.

4.1 Releases

The main contribution to the LDStack is the publication of new releases in the repositories.

Table 4 presents the Debian packages uploaded by GeoKnow, as in M24.

Table 4: Component distributions within GeoKnow

Component	Version	Observations
virtuoso-opnesource	7.1.0	Virtuoso 7.1.0 includes improvements in the Engine (SQL Relational Tables and RDF Property/Predicate Graphs); Geo-Spatial support; SPARQL compiler; Jena and Sesame provider performance; JDBC Driver; Conductor CA root certificate management; WebDAV; and the Faceted Browser.
FAGI-gis	1.1+rev0	FAGI aims to provide data fusing on geometries of linked entities. This latest version provides several optimisations that increased the scalability and efficiency. It also provides a map-based interface for facilitating the fusion actions through visualisation and filtering of linked entities.
linkedgeodata	0.4.2	The LinkedGeoData package contains scripts and mapping files for converting spatial data from non-RDF (currently relational) sources to RDF. OpenStreetMap is so far the best covered data source. Recently, initial support for GADM and Natural Earth were added. • Added an alternative *lgd* load script which improves throughput by inserting data into a different schema first followed by a conversion step. • Optimized export scripts by using parallel version of pbzip. • Added rdfs:isDefinedBy triples providing licence information for each resource.

[18] http://lod2.eu/
[19] http://www.semantics.cc/www.semantics.cc/index.html
[20] http://blog.geoknow.eu/the-linked-data-stack/

facete2-tomcat7	0.0.1	Facete2 is a web application for exploring (spatial) data in SPARQL endpoints. It features faceted browsing, auto detection of relations to spatial data, export, and customization of which data to show. • Context menus are now available in the result view enabling one to conveniently visit resources in other browser tabs, create facet constraints from selected items and copy values into the clipboard. • Improved Facete's autodetection when counting facets is infeasible because of the size of the data • Suggestions of resources related to the facet selection that can be shown on the map are now sortable by the length of the corresponding property path.
facete2-tomcat-common	0.0.1	This package is a helper package and is mainly responsible for the Facete database setup. There were no significant changes.
sparqlify-tomcat7	0.6.13	This package provides a web admin interface for Sparqlify. The system supports running different mappings simultaneously under different context paths. Minor user interface improvements.
sparqlify-tomcat-common	0.6.13	This package is a helper package and is mainly responsible for the Sparqlify database setup. There were no significant changes.
sparqlify-cli	0.6.13	This package provides a command line interface for Sparqlify. Sparqlify is an advanced scalable SPARQL-to-SQL rewriter and the main engine for the LinkedGeoData project. • Fixed some bugs that caused the generation of invalid SQL. • Added improvements for aggregate functions that make Sparqlify work with Facete. • Added initial support for Oracle 11g database.
limes-service	0.3.1	Limes-services updated to the latest LIMES library. The main enhancement this year was refactoring the service to provide RESTful interface.
geoknow-generator-ui	1.1.0	First public release of the GeoKnow Generator Workbench extends the initial prototype by including user and role management, graph access control management, processing monitoring within a dashboard.

Upcoming packages programmed to make available in the Debian repository by M26 are presented in Table 5.

Table 5: Upcoming distributions within GeoKnow

Component	Version	Observations
Deer	0.0.1	GeoLift has been renamed to DEER[21]. The functionalities provided in GeoLift have been generalised to not only support geospatial, but generally structured data.
RDF Data Cube Validation Tool	0.0.1	Validation tool aims to ensure the quality of statistical datasets. It is based primarily on the integrity constraints defined by the RDF Data Cube vocabulary, and it can be used to detect violations of the integrity constraints, identify violating resources, and fix detected issues. Furthermore, to ensure the proper use of vocabularies other than the RDF Data Cube vocabulary, it relies on RDFUnit. It can be configured to work any SPARQL endpoint, which needs to be writeable in order to perform fix operations. However, if this is not the case, user is provided with the SPARQL Update query that provides the fix, so that it can be executed manually. Main purpose of the tool within the GeoKnow project is to ensure the quality of input data that is to be processed and visualized with ESTA-LD.
ESTA-LD	0.0.1	ESTA-LD (Exploratory Spatio-Temporal Analysis for Linked Data) is a software tool for interactive analysis, visualisation and drill-down style online access to highly granular spatio-temporal data. It is aimed for business users interested in spatial analysis of different socio-economic indicators modelled using W3C standard vocabularies (RDF Data Cube, eGovernment) and LinkedData principles. The data retrieved from a specified SPARQL endpoint is visualized on the choropleth map thus allowing the user to study the indicators (stored in a single or multiple datasets) across a geographic area. The geographic data (such as region borders), originally created from shape files, is stored in GeoJSON format. Different user interaction and filtering options are foreseen. Currently supported filtering options are selection of indicator under study, selection of the area of interest on the geographical map, selecting geo granularity level, selecting values from the hierarchical dimensions modelled with the SKOS vocabulary, and selecting the chart type. The results of the spatio-temporal analysis are visualized using Highcharts library.
spring-batch-admin-geoknow	0.0.1	The spring-batch-admin-geoknow is the first version of batch processing component that functions as the backend of the Workbench's.

[21] https://github.com/GeoKnow/DEER

4.2 Source Repository

For handing over purposes and future management of the LDStack repositories and website, we have created the Linked Data Stack as organisation in GitHub: https://github.com/LinkedDataStack.

4.3 Website

For dissemination purposes, GeoKnow developed in the first year a project-independent website for publishing information about activities around the Linked Data Stack[22]. This year we have provided maintenance to the website by including the following content:

A dynamic Debian repository component list.
A rss-like publications of releases for the Debian repository was implemented for been able to show on the website the actual list of components in the stack. Previously, this list was hand written and needed to be updated every time the repository was updated.

A Maven Repository RSS reader.
To provide an up-to-date list of libraries in the Maven repository, the website reads the RSS available in Sonatype Nexus repository manager used for some of the components distributed in the stack.

Developer recommendations
With the development of the Generator Workbench we have experienced a big challenge integrating such heterogeneous software components. However, we still believe that some harmonisation can be achieved if component developers can commit at following some guidelines. Therefore, we have added in the documentation of the LDStack, a list of recommendations that could make integration work easier:

1. To accept a SPARQL Endpoint Input:(if required) SPARQL Endpoints can be provided to the component where data can be read. And even better to accept FROM Graph, thus the component can read from specific graphs.
2. To accept a SPARQL Endpoint Output:(if required) SPARQL Endpoints where data can be written, and even better, output results into a specific graph
3. Support SPARQL HTTP Authentication: Provide support for authentication methods, so the component is able to read private graphs.
4. Add provenance information into a graph using, for instance, the PROV-O ontology[23]
5. Provide a RESTful API Interface: the easiest way to integrate tools
6. Provide an exhaustive documentation to your Debian/control and Changelog files of the Debian packages.

Some of these recommendations were perviously defined as requirements, however it is really hard to know to which components these requirements can apply, and also to make sure if component developers are really following them.

[22] http://lod2.eu/
[23] http://www.w3.org/TR/prov-o/

5 Conclusions and Future Work

This report presented the second year activities performed for the GeoKnow Generator. We reviewed software tools developed in GeoKnow to meet user requirements and the GeoKnow Generator Workbench system requirements. We also presented the the first release of the GeoKnow Generator. The general functionality of the GeoKnow Generator was extended by an user management environment and authentication system. This allows to set detailed views on what a user is allowed to do. The concept of private and public graphs was incorporated to support the user management and roles. The current version is available as Debian package, as source code on the git and as a pre-installed system on a virtual machine. By this way the GeoKnow team provides an easy access to the tool set.

Along with the Workbench distribution, components developed within GeoKnow have been distributed within a common distribution platform for Linked Data software tools used by other EU projects such as LOD2. In this second year of the project, GeoKnow is engaged to continue giving maintenance to this distribution platform and continue software distribution in the same manner. Contribution to LDStack has been done by improving the website and updating the repositories with new components and new versions of existing ones. The collaboration with the LOD2 EU project was intensified and in September 2014 a hand-over from LOD2 to GeoKnow was coordinated in order to guarantee the successful continuation of the linked data stack. The launch of the GeoKnow generator will be promoted using the social channels, workshops and conferences.

In year 3 we plan to enhance the dashboard, monitoring and work flow environment, provide more guidance on integrating tools that are developed by the external community and to assure the seamless integration of new components. We will seek new collaborations with other projects as in order to guarantee the continuation of the LDStack after the official end of the GeoKnow research project. Last but not least, we are confident that first exploitations of the stack will be done and the team will provide support to those new users within the available resources.

Appendices

A GeoKnow Generator Requirements

Each requirement has an identifier where the first group of two letters stands for the requirement type. Some requirements were grouped together because of their similarity. Requirement identifier is composed firstly with a type of requirement:

FR Functional Requirement

PR Performance Requirement

SR Security Requirement

IR Interfase Requirement

DR Data Requirement

The second couple of characters identifies each requirements define the source of the requirement:

U Unister Use Case

B Brox Use Case

S Online Survey

D Description of Work document of GeoKnow (DoW)

E Description of Work Extended document of GeoKnow (DoW)

R Requirement from the Review meeting 2014

Table 6: User Requirements

Req. IDs	Priority	Short Description
FR-U6	High	GeoKnow Generator has to be able to process subsets of data sets.
FR-B8	High	Users can select a subset of the configured Data Sources for each context.
DR-U1	High	GeoKnow Generator has to keep track of the coarse-grained provenance of a data set.
DR-U2	Medium	GeoKnow Generator has to keep track of the fine-grained provenance of single data items.
IR-B1	High	GeoKnow Generator functionality can be accessed via REST API for easy
DR-B8	High	Contexts can be defined and exported as an XML structure.

FR-B1	High	GeoKnow Generator has to provide Single Sign On (SSO) compatibility
SR-S2	Low	SSO and extra functionalities for authenticated users
FR-B3	High	Administrators can connect authentification systems using a web-based admin console
FR-B6	High	Administrators can manage the Data Sources and define access roles with respective access properties on the managed Data Sources.
FR-B4	High	Users can configure views and save them within a context in a web-based dashboard application.
FR-B7	High	GeoKnow indicates data source accessibility status in the dashboard.
IR-B3	High	GeoKnow has to provide a web-based admin console to administrators.
SR-B4	High	GeoKnow has to provide access control for public, protected, and private data.
SR-B6	High	All GeoKnow components have to support secure authorization and communication.
SR-B5	High	In the ontology editor, ontology and customer data must be separated based on roles.
FR-U2	High	Functional Requirement: GeoKnow Generator has to log every processing step (mapping, matching, transformation, merging,etc.), so the processing can be monitored.”
DR-D6	High	Present all datasets listed in the DoW, support cases in which data continuously changes as well as bulk loading scenarios
FR-U1	High	For any entity, GeoKnow Generator has to identify all base knowledge entities with a geographical relation to it, such as ””entirely located in””, ””partially located in””, ””border on””, and ””nearby”
FR-S1	Low	Advanced spatial query capabilities
FR-S4	Low	Export/Import data, also for a region of interest
FR-U10	Medium	GeoKnow Generator has to preprocess data so they are more applicable for search strategies.
PR-U3	High	The information created by GeoKnow Generator has to enable search query response times of at most one second for typical queries.
PR-U5	Medium	The GeoKnow Generator has to process input parameters for generating strategic information in reasonable time.
PR-U4	Medium	The GeoKnow Generator has to handle analytical queries for a reasonable user base in reasonable time.
FR-U3	High	GeoKnow Generator has to integrate knowledge from various public and internal sources in RDF.
FR-S9	Low	Merging of private and public data pools to be used together

FR-B5	High	GeoKnow has to integrate internal (private) and external (public) data
FR-U7	Medium	GeoKnow Generator has to integrate uncertain knowledge extracted from unstructured sources.
FR-U4	High	GeoKnow Generator has to provide sophisticated resource matching methods based on multiple features for invalidating errors in original datasets.
FR-U5	High	GeoKnow Generator has to enable quality assurance by providing metrics.
FR-U8	Medium	GeoKnow Generator has to be able to identify conflicts and provide solutions for solving them. Conflicts could be consistency issues expressed as rules, such as "No two distinct entities should have the same polygon". The solution may be (semi-) automatic or manual, depending on the configuration.
FR-B2	High	Logistics users will be empowered to view supply chain flows in close to real time and drill down into the linked information layers on demand.
IR-S6	Low	Data reduction for different zoomlevels
IR-S10	Low	Visualising large number of markers, plotting polygons
FR-S7	Low	Improved building trace / area measurement tools in OSM
FR-S8	Low	Visual analysis where I could intuitively select features and analyse them without having to dig into datasets
FR-D1	High	Possibility to browse data in the GUI
FR-D2	High	chosing between map providers (e.g. OpenStreetMap, Google Maps) will require no coding
FR-D3	High	A widget library, based on the use-cases, with some widgets (e.g. for place searches, ratings, images and facts) being usable across multiple domains.
FR-D5	High	Subscription and notification service enable users to subscribe to arbitrary information adhering to certain spatial or semantically defined filter criteria. In a second development iteration, we will extend to subscription and notification service for GeoKnow with an API adhering to the Open-social standard
FR-D4	High	GeoKnow has to integrate SMILA as a data source.
DR-U6	Medium	The GeoKnow Generator has to be able to extract general and user-specific information from social networks.
FR-S3	Low	device compatible, usable in mobiles
IR-B2	High	EDI messages are received via Secure File Transfer Protocol (SFTP).Properties for SFTP server and key management have to be part of the GeoKnow Admin Console.
FR-B9	High	Ontology editor users can extend and exchange the mapping ontology.
DR-B10	High	An ontology based data model and automatic data converters will be used to integrate new data types and new message formats without a software update

FR-U9	Medium	In addition to rarely changing base information (FR-U1), GeoKnow Generator has to support highly dynamic information.
DR-B9	High	GeoKnow has to provide an offline cache and conflict resolution options.
PR-S5	Low	Scalability: be able to scale operations across several GIS engines
FR-E1	High	GeoKnow Generator has to support statistical and temporal geospatial information management including advanced analysis of spatio-temporal data.
FR-E2	High	GeoKnow Generator has to integrate quality analysis services.
FR-R1	High	Dashboard to check which process is running, how much time left etc
FR-R2	High	Dashboard: defines settings, configs etc and define which jobs have to be executed in which sequenze and create a bundle (batch job) and launch for execution
FR-R3	Medium	Authentication: a extended Version for multiple login options such us WebID, G+, Facebook etc
FR-R4	Medium	Login user activities. Maybe couple it with the authentication module