Hands-on experience on Linked Data API

1. Brief Linked Data API presentation

The Linked Data API offered by the INGEOCLOUDS project offer diverse facilities for importing, exporting, querying and updating semantically described data (i.e. data described following a formal model like an ontology). Those data are usually interlinked to each other following the recently introduced Linked Open Data initiative of World Wide Web Consortium (W3C) and allowing for more complex reasoning by combining in new ways otherwise unrelated information, which refers to the same entities. The API also allows exporting geospatial data in various formats including the ability to create INSPIRE compliant formats. Finally, by referring to the API, these blueprints mainly target developers, but more people can benefit from it through understanding the described capabilities and engaging into proposing new applications based on it.

2. Where can I find online information about the API?

You can find online information on the wiki of the InGeoCloudS project at the following URL: http://www.ingeoclouds.eu/?q=wiki/linked-data-management-api as well as on the on-line, technical documentation of the API automatically produced by Enunciate tool at the following URL: https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/

3. What type of functionality is exposed by the API?

With the API, you can publish Linked Data either in a direct way through RDF descriptions or indirectly from your relational or XML-based database. You can query the data (with normal or geo-spatial queries) when they are published and you can of course update them. The updating can be performed directly or indirectly in case you are using the indirect importing mechanism for relational data. The Linked Data stored can be exported in various RDF format as well as on XML-based INSPIRE form. Results of Linked Data queries can also be transformed to quite well known feature representation formats, such as KML and GeoJSON, for visualization in maps.

4. Is it possible to combine the use of some API methods?

You can of course combine the use of some API methods, and the API was designed to offer this type of support. You can combine methods which produce query results (i.e., *ldquery* (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_ldquery.html) and *geoldquery* (*https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_GeoLinked%20Data%20Service.html#path__geold_geoldquery.html*)) with the method that transforms these results into feature collection representation formats (i.e., *geoldtransform* (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_GeoLinked%20Data%20Service.html#path__geold_geoldtransform.html)). You can also combine non-blocking method calls (i.e., by calling *ldimport* (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_ldimport.html)) with method calls that allow making enquiries (i.e., *import_status* (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_import_status.html)) for the results of the non-blocking method call.

5. Do I need a particular programming language to access the API?

To access the API, you can use any language as long as it is enabled to interact with a REST-based service (such as the LinkedData Management API) in the Internet. For instance, you can use Java or Javascript to access the API. Example java code snippets can be found at this document and at the following URL: http://www.ingeoclouds.eu/?q=wiki/linked-data-management-api#example.

6. Can I import RDF data in a non-blocking way?

Depending on the type of importing functionality requested, there can be parameters indicating whether the call will be blocking or not. In case of calling the direct importing API method (*ldimport*: https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_ldimport.html), you can set the method parameter *blocking* as false. In case of R2RML-based (indirect) importing, the call to the API method (*addR2RMLMappings*: https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_addR2RMLMappings.html) is non-blocking due to the nature of the requested functionality. In particular, the synchronization of the RDF Triple Store with the original Relational Database of the user usually takes significant time, so it is meaningless to block the user's program waiting for this synchronization to end. In case of XSL-based (indirect) importing, the API method (*addXSLMappings*: https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_addXSLMappings.html) was designed to be blocking as the time to transform XML data into RDF is usually small.

7. Do I have to create or process any special object to interact with the API?

In case of non-blocking (direct) import, an *ImportStatus* object (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/el_linkeddata_importStatus.html) will be returned through which (actually its importID field) you can enquire the status of the import request via the *import_status* method of the API. In case of the INSPIRE queries that are used to enquire the RDF TripleStore for particular geospatial themes, through the use of the *theme_query_info* method of the API, a QueryInfo object is returned which analyzes the query involved and its variables (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/el_ns0_queryInfo.html). This information is useful for constraining the data exported (via SPARQL Filter constraints on the variables) into the XML-based Inspire form through the *inspire_export* method of the API. A visual description of the objects can also be found at: http://www.ingeoclouds.eu/?q=wiki/linked-data-management-api#description. The objects can be returned in XML or JSON form, where the default is JSON. The user can of course indicate in his/her request to the respective method which form of the objects must be supported/returned (see also example of using Java to call *ldquery* API of the method in the documentation where the user sets the accepted type of results to be returned as "application/sparql-results+xml").

8. What types of feature collection representation formats are supported?

For the moment, the API supports the transformation of (Geo)SPARQL results in the following geospatial formats: KML, GeoJSON, Shape and GML. See also: https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_GeoLinked%20Data%20Service.html#path__geold_geoldtransform.html with respect to the exact media types to set in order to obtain feature collection descriptions in a specific format. An example java code fragment for calling the respective *geoldtransform* API method can be found at: http://www.ingeoclouds.eu/?q=wiki/linked-data-management-api#example.

9. What types of RDF formats are supported?

Both the *ldimport* and *transform* API methods support the following RDF formats (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_transform.html): Turtle, N3 and XML. On the other hand, the *ldexport* method supports a more extensive list of RDF formats (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_ldexport.html): Turtle, N3, Trix, X-Trig, X-Nquads, XML, X-Binary-RDF. Thus, there is a basic set of formats supported across all respective API methods as well as a more extensive set of formats supported by the LD exporting functionality of the API.

10. What types of geospatial functions and operators are supported?

Through the new version of Virtuoso (v.7), there are now two alternative ways to perform geospatial queries: (a) use plain SPARQL (through the *ldquery* API method (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_ldquery.html)) and exploit the SQL MM functions realized by Virtuoso and (b) for an almost full GeoSPARQL support, use the *geoldquery* API method (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_GeoLinked%20Data%20Service.html#path__geold_geoldquery.html) which exploits the underlying uSeekM framework. The geospatial functions supported by Virtuoso are analyzed in Section c.a while there is also an online URL at the Virtuoso web site (http://docs.openlinksw.com/virtuoso/rdfsparqlgeospat.html). The geospatial functions and operators supported by uSeekM are listed at the following URL: https://dev.opensahara.com/projects/useekm/wiki/GeoReference.

11. What are the themes currently supported by the Inspire exporting functionality of the API?

The themes currently supported are the following: "Borehole", "Water Level", "Chemical Analysis", "Lithology analysis", "Lithostratigraphy Analysis", "Landslide Events", "Landslide Feature Observations", "Landslide Observations", "Rainfall Data", "Earthquake Events", "Earthquake Observations", and "Earthquake Recordings". See also: https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_inspire_export.html

12. In what form do I get the results when calling the INSPIRE export methods?

You obtain back a zip file containing the results for all the themes requested. In case of calling the *inspire_export* method (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_inspire_export.html), then the zip file will contain XML files, each corresponding to one of the themes requested. Thus, if you ask for the themes of "Chemical Analysis" and "Borehole", then the zip file will comprise two XML files, one per each theme containing the respective results. In case of calling the *inspire_query_export* API method (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_inspire_query_export.html), then you obtain back a zip file comprising zipped XML files, each pertaining to a different theme related to the query posed to this method. Thus, based on the current example, the zip file will comprise two zipped XML files, one per each theme containing the respective results.

13. How do I know which themes are related to a particular query?

In case of calling the *inspire_query_export* method (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_inspire_query_export.html), the method itself discovers the themes that are related to the issued query and returns the respective XML-based theme-specific results (in a formed of a zipped file) in a zip file. In case of calling the *inspire_export* method (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_inspire_export.html), you decide which themes are related to your goal and you explicitly state them as input parameter (see *themes* method parameter) values in the respective method call.

14. How do I know which constraints to put to the inspire_export method?

You can first execute the *theme_query_info* API method (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_theme_query_info.html) which will indicate to you what is exactly the SPARQL query issued for a particular theme and which SPARQL variables can be constrained and in which way (the variables are analyzed in detail explaining to which type of information they map and what is their domain of values). Then, you can construct the respective SPARQL Filter constraints based on the desired SPARQ variables and the corresponding desired values from their domain and use these constraints as input to the inspire_export method (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_inspire_export.html) (see *constraints* method parameter) along with the respective value for the *themes* method parameter. A specific example can be seen in: (! link to respective java code fragment).

15. What types of formats are supported for the results of a (Geo)SPARQL query?

Both *ldquery* and *geoldquery* API methods support the following SPARQL results formats: CSV, TSV, XML and JSON (see also https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_ldquery.html). An example of how to set the desired results format is provided at: http://www.ingeoclouds.eu/?q=wiki/linked-data-management-api#example.

16. Can I avoid putting inline in the request a big file mapped to a particular API method parameter?

In many cases, you have the ability to provide a URL from which the file can be downloaded (obviously if it is available). The API methods that support this (see their technical description at https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/) are the following: *addR2RMLMappings* (see *mappingsURI* method parameter), *ldImport* (see *url* method parameter), *transform* (see *url* method parameter), and *geoldtransform* (see *url* method parameter).

17. Can I see the status of a blocking request?

Normally, when you call the *ldimport* method (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_ldimport.html) with the method parameter *blocking* as true, you get a *ImportStatus* object (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/el_linkeddata_importStatus.html) that indicates whether the import request was successfully performed or not. To this end, you have all the information provided in this way and there is no need to consult another method for obtaining it. In this way, the *import_status* method (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_import_status.html) returns *ImportStatus* information only for non-blocking *ldimport* method calls.

18. Can I restrain the number of results returned by a call to a query or exporting functionality?

There is usually a *maxRows* parameter in many API methods which return query or export results (see also technical description in https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/). These methods are the following: *ldquery* (see *maxrows* method parameter), *ldupdate* (see *maxRows* method parameter), *inspire_export* (see *maxRows* method parameter), *inspire_query_export* (see *maxRows* method parameter), *geoldquery* (see *maxrows* method parameter), *earthquakes* (see *limit* method parameter), and *landslides* (see *limit* method parameter).

Please note that when you are providing a (Geo)SPARQL query/update statement (e.g., for calling the methods ldquery, geoldquery, *ldupdate* or *inspire_query_export*) there is usually the option to include a *Limit* sub-statement which has the same effect as in providing a value for the respective parameter method. To this end, it is better to use one of the two result limitation alternatives and not both. The default behavior in case both alternatives are used is that the *Limit* sub-statement in the query/update is ignored.

19. What is the role of the earthquakes and landslides API methods?

These methods have been integrated in order to enable the integration of internal to the InGeoCloudS platform data with external data provided by various related web-based sources. The integration result is actually performed on the fly and not stored in the RDF Triple Store and can be returned in three different forms: XML, JSON and KML. The latter format is useful for visualization purposes and obviously can be exploited by prospective applications in order to not only exploit this specific integration functionality offered by the API but also visualize the respective results obtained. A demo application was constructed in this way, which can be accessed at the following URL:

http://ec2-46-137-153-13.eu-west-1.compute.amazonaws.com:8080/demo.html. The two API methods offer searching functionality for earthquakes and landslides, respectively, by exposing particular method parameters which map to the information of the GSOM model (see Deliverable D2.2 for an analysis of this model).

20. What does the error message "The request cannot be fulfilled due to bad syntax" actually mean mapped to error code 400?

This means that you have provided a query ((Geo)SPARQL) or update (SPARUL) statement which is wrong (syntactically incorrect). You will have to correct your query/update statement accordingly in order to avoid getting back this error message.

21. Why for some methods I am getting an Access Denied error message?

This means that this method is secured and you have to provide the respective authentication information (actually you need to provide an authentication token once you have successfully logged-in via SSO - see java code fragments in http://www.ingeoclouds.eu/?q=wiki/linked-data-management-api#example). If, however, you have provided the correct authentication token, then probably you are attempting to modify (via the direct/indirect import or update API methods) a LinkedData graph that you do not own. The methods whose access is secured are the following: *ldimport*, *ldupdate*, *addR2RMLMappings*, and *addXSLMappings*.

22. I am providing correct values for method parameters but still obtain an error message of the

form "The request cannot be fulfilled due to wrong parameter use". What is wrong?

This means that the combination of the parameter values is not allowed. For instance, for the *ldimport* method (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/resource_Linked%20Data%20Service.html#path__ld_ldimport.html), if you provide a specific URL for the imported RDF file to be downloaded, you should also specify the RDF format of this file (and not neglect to provide this method parameter). Please read carefully the documentation of each method as there is a description of which parameters require values to be set also for other parameters of the same method.

23. How do I know which are the default parameters for the API methods?

You just have to consult the technical documentation of the API method (by visiting https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/). In the Parameters table there is a column which indicates for each parameter row the default value for the respective parameter.

24. What does the error message "The requested resource could not be found" mean?

This means that either you have provided a URI or a URL value for a method parameter that does not map to an existing and available resource. The methods on which this error can appear are the following: ldimport, addR2RMLMappings, import_status, ldexport,

25. FIND GUIDANCE/HELP/SUPPORT

- If you encounter any problem with the LD Management API, then please first read carefully the FAQ (previous section) and the technical documentation of the API (https://portal.ingeoclouds.eu/ingeoclouds-api/linkeddata/)
- If you do not find any solution to your problem, then please visit the known bug list of the InGeoCloudS platform: http://www.ingeoclouds.eu/?q=wiki/getting-help#Known%20Bugs
- If your problem is not included in the known bug list, then probably a new bug has been identified. If so, please send us an email to ingeoclouds-support@ingeoclouds.eu or report this bug thanks to the feedback panel provided on the portal.
- If you feel that you need more specific and dedicated support and/or training, get in touch with InGeoCloudS team!