Semantic Web 0 (2018) 1–55
IOS Press

Survey of Tools for Linked Data Consumption

Jakub Klímek [a,*] Petr Škoda [a] Martin Nečaský [a]
[a] *Faculty of Mathematics and Physics, Charles University in Prague*
Malostranské náměstí 25, 118 00 Praha 1, Czech Republic
E-mail: klimek@ksi.mff.cuni.cz, skoda@ksi.mff.cuni.cz, necasky@ksi.mff.cuni.cz

Abstract. There is lots of data published as Linked (Open) Data (LOD/LD). At the same time, there is also a multitude of tools for publication of LD. However, potential LD consumers still have difficulty discovering, accessing and exploiting LD. This is because compared to consumption of traditional data formats such as XML and CSV files, there is a distinct lack of tools for consumption of LD. The promoters of LD use the well-known 5-star Open Data deployment scheme to suggest that consumption of LD is a better experience once the consumer knows RDF and related technologies. This suggestion, however, falls short when the consumers search for an appropriate tooling support for LD consumption. In this paper we define a LD consumption process. Based on this process and current literature, we define a set of 34 requirements a hypothetical Linked Data Consumption Platform (LDCP) should ideally fulfill. We cover those requirements with a set of 94 evaluation criteria. We survey 110 tools identified as potential candidates for an LDCP, eliminating them in 3 rounds until 16 candidates for remain. We evaluate the 16 candidates using our 94 criteria. Based on this evaluation we show which parts of the LD consumption process are covered by the 16 candidates. Finally, we identify 8 tools which satisfy our requirements on being a LDCP. We also show that there are important LD consumption steps which are not sufficiently covered by existing tools. The authors of LDCP implementations may use our paper to decide about directions of future development of their tools. The paper can also be used as an introductory text to LD consumption.

Keywords: Linked Data, RDF, dataset, consumption, discovery, visualization, tool, survey

1. Introduction

The openness of data is often measured using the now widely accepted and promoted 5-star Open Data deployment scheme introduced by Sir Tim Berners-Lee[1]. The individual stars are awarded for compliance with the following rules (citing the web):

1* make your stuff available on the Web (whatever format) under an open license
2* make it available as structured data (e.g., Excel instead of image scan of a table)
3* make it available in a non-proprietary open format (e.g., CSV instead of Excel)
4* use URIs to denote things, so that people can point at your stuff
5* link your data to other data to provide context

The last star is also denoted as Linked Open Data (LD/LOD). More and more data publishers are convinced that the additional effort put into the last 2 stars, i.e. LD in RDF as opposed to CSV files, will bring the promised benefits to the users of their data. The publishers rightfully expect that the users will appreciate the 5-star data much more than, e.g., the 3-star CSV files given that the proper publication of 5-star data is considerably harder and more costly to achieve. In particular, they expect that the users will appreciate benefits such as better described and understandable data and its schema, safer data integration thanks to the global character of the IRIs and shared vocabularies,

*Corresponding author. E-mail: klimek@ksi.mff.cuni.cz.
[1] http://5stardata.info

better reuse of tools, more context thanks to the links and the ability to (re)use only parts of the data.

Open Data users are familiar with 3-star publication formats and principles but it is hard to encourage them to use 5-star data. Often they do not know LD technologies, most notably RDF model, its serialization formats and SPARQL. They refuse to learn them because they do not see the difference between 3-star and 5-star representations. For example, our initiative *OpenData.cz* has re-published 77 2-star and 3-star datasets published by Czech public bodies in 5-star representation[2]. Moreover, as a part of the EU funded project *COMSODE*, we participated on re-publishing other datasets published by Czech, Slovak, Italian and Dutch public bodies and several pan-European datasets in 5-star representation[3]. We also helped two Czech governmental bodies to publish their Open Data as 5-star data (the Czech Social Security Administration[4][75] and the Ministry of Finance[5]). We presented the publication results to Open Data consumers at several community events. Their feedback was that they would like to use the data but they need a consumption tool which does not require them to know the LD technologies. Only after being able to work with LD in this way and see the benefits on real data of their interest they will consider learning LD technologies to use the data in an advanced way.

There are also other resources reporting that learning LD technologies is a problem. For example, [121] describes a toolset for accessing LD by users without the knowledge of LD technologies. A recent blog post *"Why should you learn SPARQL? Wikidata!"*[6] shows an interesting view of learning SPARQL presented by one of Open Data users. This blog post also mentions a widely known problem of quality of LD and availability and performance of services providing LD. These problems are problems of Open Data in general as reported, e.g., in [85]. However, they seem to be even more serious in case of LD. For example, the SPARQLES service[7] shows that only 38 % of known services providing LD are available[8].

In this paper we do not aim at surveying and evaluating the problems with data availability, performance and quality as it is a problem of its own. We are interested in the other problem which is a question whether there are software tools or platforms available which support Open Data users to consume LD without the knowledge of LD technologies. It is not so important whether such platform is a single software tool or a set of compatible and reusable software tools integrated together. The need for building such a platform is also identified in [121].

We call such a hypothetical platform a *Linked Data Consumption Platform* (*LDCP*). In the broadest sense, an LDCP should support users in discovering, loading and processing LD datasets with the aim of getting some output useful for users without the knowledge of LD technologies. However, it would not be practical to require each LDCP to support all these tasks. In the minimal sense, an LDCP should have the following properties:

1. It is able to load a given LD dataset or its part.
2. Its user interface does not presume the knowledge of LD technologies so that users without such knowledge can use the tool. Most notably this includes the knowledge of the RDF data model, its serializations and the SPARQL query language.
3. It provides some non-RDF output the users can further work with. The output can be a visualization or a machine-readable output in a non-RDF format, e.g. a CSV file.

The contribution of this paper is a survey of existing software tools presented in recent scientific literature which can serve as an LDCP at least in this minimal sense. The tools are evaluated using a set of requirements described in the paper.

In the first part of the paper, we describe the LD consumption process and we identify requirements which specify how individual activities in the LD consumption process could be supported by an LDCP. The requirements are identified based on existing W3C Recommendations and research areas and approaches published in recent scientific literature which is related to some parts of the LD consumption process. Therefore, the individual proposed requirements are not novel. Each requirement either comes from an existing W3C Recommendation and/or the possibility of fulfilling the requirement is described in the literature. For each requirement we define a set of criteria to be used to decide whether a given software tool fulfills the requirement or not. In total, the paper presents 34 requirements covered by 94 evaluation criteria. The requirements and their evaluation criteria constitute a framework we use to evaluate the tools.

[2] https://linked.opendata.cz
[3] http://data.comsode.eu/
[4] https://data.cssz.cz/sparql
[5] http://cedropendata.mfcr.cz/c3lod/cedr/sparql
[6] https://longair.net/blog/2017/11/29/sparql-wikidata/
[7] http://sparqles.ai.wu.ac.at/
[8] Checked on 09.03.2018

In the second part of the paper we present our contribution – a survey of existing software tools which can be used as LDCPs. To identify LDCP candidates we do a systematic review of recent scientific literature. We search for papers which describe software tools for LD consumption, visualization, discovery, exploration, analysis or processing. For each candidate we search for its on-line instance (or demo) or we install our own instance and we try to load our testing LD datasets to the instance. If we are successful the candidate is selected for further evaluation. As a result, we select 16 software tools. Each tool is evaluated using the whole set of criteria to determine the fulfilled requirements introduced in the first part of the paper. The individual requirements are neither disqualifying nor intended to select the best tool for LD consumption. They represent different LD consumption features. We show how each considered tool supports these features.

We distinguish two kinds of users in the evaluation. First, there are 3-star data consumers who are not familiar with LD technologies. These users initially motivated our work. We call them *non-LD experts*. Second, there are experts in LD principles and related technologies. We call them *LD experts*. When evaluating a tool we distinguish whether the tool is suitable for a non-LD expert. We consider a tool as suitable for a non-LD expert when its user interface does not presume the knowledge of LD technologies. The evaluation is, however, not a user based study and, therefore, it does not evaluate the usability of the tools beyond distinguishing the LD expert and non-LD expert usage.

Based on the evaluation we show tools which are LDCPs – either in the minimal sense specified above or with some additional features represented by our requirements. Moreover, we show tools which offer something relevant for LD consumption in terms of our requirements even though they provide only RDF output. These tools do not fulfill condition 3 of the minimal LDCP defined above. However, their RDF output can be used as an RDF input for any LDCP and therefore, they can be integrated with any LDCP and extend its LD consumption features.

This paper is based on our introductory study of the problem [78]. Since then, we have better defined requirements, better defined survey methodology and we survey considerably more tools which are also more recent, which we present here.

This paper is structured as follows. In Section 2 we provide a motivating example of a user scenario in which journalists want to consume LD and expect to have a platform that will enable them to enjoy the promised LD benefits. In Section 3 we describe our evaluation methodology. In Section 4 we define the LD consumption process, describe the identified requirements and cover them with evaluation criteria. In Section 5 we identify existing tools and evaluate them using the evaluation criteria. In Section 6 we survey related studies of LD consumption and in Section 7 we conclude.

2. Motivating Example

To motivate our work, let us suppose a team of data journalists. For their article they need to collect data about the population of cities in Europe and display an informative map. The intended audience are statisticians, who are also used to CSV, so the journalists also want to publish the underlying data for them as CSV files attached to the articles.

The journalists are used to working with 3-star open data, which means that they have to know where to find the necessary data sources, e.g. population statistics and descriptive data about cities including their location, they have to download them, integrate them and process them to get the required output, i.e. a visualization and a CSV file.

There are several examples of such articles available on-line. For example, there is a pan-European *European Data Journalism Network*[9] where data journalists from different publishing houses in Europe publish their articles based on different data sources. As an example there is an article titled "Xenophobia in European cities"[10] which exploits Open Data about European cities.

The journalists now want to use LD because they heard that it is the highest possible level of openness according to the 5-star Open Data deployment scheme. They expect the experience of working with LD will be somehow better than the experience of working with 3-star data. Naturally, they expect a tool they can use to work with such data and benefit from LD principles. On the other hand, they expect that they will be able to use the tool even without the knowledge of LD technologies such as RDF serialization formats or SPARQL. Such user expectations are described also in the literature. For example, [121] presents a scenario where non-LD experts search and analyze LD indexed in Open Data

[9] https://www.europeandatajournalism.eu/

[10] https://www.europeandatajournalism.eu/eng/News/Data-news/Xenophobia-in-European-cities

portals such as the Open Data portal of EU commission[11] or similar. In our recent work [97], we present similar scenarios of searching, analyzing and visualizing datasets by non-LD experts published by Czech public bodies. A user based study [122] presents how data consumers from the tourism domain (both non-LD experts as well as LD experts) can benefit from LD when working with different cross-domain datasets.

In the rest of this section, we show how a LD consumption tool may enable our journalists to benefit from LD at different consumption steps without the prerequisite of knowing LD technologies.

First, the journalists need to find data about cities in Europe, i.e. their description and location so that they can be placed on a map, and data about their population. Here, LD can help already. Recently, a number of national open data catalogs emerged and are being harvested by the European data portal[12]. It utilizes the DCAT-AP v1.1[13] vocabulary for metadata about datasets and exposes it in a SPARQL endpoint. The endpoint can be queried for, e.g. keywords (city) and formats (Turtle, JSON-LD, etc.). LDCP could therefore support loading and querying of dataset metadata from DCAT-AP enabled catalogs and even contain well-known catalogs, such as the European data portal, preset so that the user can search for datasets immediately. It can than provide a search user interface for non-LD experts on top of the collected DCAT-AP metadata.

After identifying candidate datasets the journalists need to choose the ones that contain information needed for their goal. A helpful functionality of LDCP could be various kinds of summaries of the candidate datasets and characterizations based on both the metadata of the dataset and the actual data in the dataset. Of course we need to suppose that the metadata loaded from a catalog contain correct access information for each candidate dataset. For example, for each candidate dataset, LDCP could offer human-readable labels and descriptions or resources from vocabularies used, numbers of instances of individual classes and their interlinks, previews tailored to the vocabularies used or to the current use case (visual or textual, readable to non-LD experts), datasets linked form the candidate, datasets linking to the candidate, spatial and time coverage, etc. Using this information, the journalists would be able to choose a set of datasets containing the required information more easily.

Another feature that could help them at this time is recommendation of related datasets. Based on available information about datasets already selected LDCP could suggest similar datasets. Given a set of datasets to be integrated, a useful feature would be the ability to analyze those datasets in order to find out whether they have a non-empty intersection, e.g. whether the population information specified in one dataset actually links to the cities and their locations present in another dataset.

There could be an interoperability issue between selected datasets caused by the existence of multiple vocabularies describing the same domain. In our example, it could be that the dataset containing locations of the cities could have the geocoordinates described using the *schema:GeoCoordinates*[14] class from the Schema.org vocabulary but the user needs another representation, e.g., the *geo:Point*[15] class of the WGS84 Geo Positioning vocabulary. Since both of these vocabularies are well-known and registered in Linked Open Vocabularies (LOV)[16], LDCP could contain components for data transformation between them and offer them to the user automatically.

Our journalists now have to choose the entities and properties from those datasets that are needed for their goal. This could be done in a graphical way used e.g. by graphical SPARQL query builders such as SPARQLGraph [127], but not necessarily complicated by the full expressiveness of SPARQL.

Finally, the data should be prepared for further processing, and the journalists need to decide, in which data formats they will publish the results of their work. Since their readers, given the focus of the article, will typically be non-LD experts, a suitable format is one of the 3-star data ones, i.e. XML or CSV, so that readers can open the data in, e.g. their favorite spreadsheet editor. This format is also suitable for people simply used to using legacy tools. This would require assisted export to tabular data in CSV, tree data in XML, or perhaps generic graph data, e.g. in CSV for Gephi [15]. Another option is to publish the data as-is in the platform, i.e. in 5-star RDF. This would be the preferred way for experts, who could then use their favorite RDF processing tools, and for users with access to a LDCP, which, as we show in this survey, is still far from avail-

[11] http://open-data.europa.eu

[12] https://www.europeandataportal.eu/

[13] https://joinup.ec.europa.eu/asset/dcat_application_profile/asset_release/dcat-ap-v11

[14] http://schema.org/GeoCoordinates

[15] http://www.w3.org/2003/01/geo/wgs84_pos#Point

[16] http://lov.okfn.org/

able. This could enable the readers to exploit, e.g. RDF enabled visualizations such as in LinkedPipes Visualization [74]. Naturally, the journalists may decide to publish both versions of their data to satisfy the widest possible audience.

Users such as our journalists will never be able to exploit these features only using basic technologies like IRI dereferencing and SPARQL. They need a software tool, an LDCP, which will help them to use and to benefit from these features even without the knowledge of LD technologies.

3. Survey methodology

The goal of the paper is to evaluate tools classified as LDCP candidates using a predefined list of requirements. In this section we describe our methodology for identification of the requirements, identification of the tools to be evaluated, for conducting their evaluation and for aggregation of their scores.

3.1. Requirements identification

To identify requirements to be used, we first define the LD consumption process based one existing literature which also introduces a similar process. For each step of the process we identify requirements that we see as crucial for LDCP. We focus on requirements which can be satisfied by exploiting the benefits of consuming LD compared to consuming 3-star data. We then cover each requirement with a set of evaluation criteria which we later use to evaluate existing tools and decide if and to which extend they fulfill the requirement.

We identified each requirement on the base of the following existing resources:

1. An existing software tool which satisfies the requirement.
2. A technical specification (e.g., W3C Recommendation) which defines the requirement.
3. A published research paper which at least theoretically proposes a method fulfilling the requirement.
4. A published research paper which expresses the need for such a requirement.

These resources are evidence that requirements are meaningful and relevant. For each of the introduced requirements it must hold that it is supported by at least one of these types of evidence. The evidence identified to support a requirement by one author was then verified by another author, and possible conflicts were addressed in a discussion leading to a consensus.

In Section 4, we describe the evidence for each identified requirement. The evidence is not a complete survey of software tools, technical specifications and literature supporting the requirement. It only describes some relevant resources which support the requirement. If there is available a research paper surveying resources relevant to the requirement, we refer to it in the evidence.

3.2. Tools selection and evaluation

In this section, we describe the publication venues investigated, search engines used and the method used to identify tools which we classify as LDCP candidates, i.e. tools described in recent scientific literature as potentially relevant to LD consumption. Then we describe the method for selecting final candidates from the potential ones and the tool evaluation methodology.

3.2.1. Publication venues investigated

We have identified the most relevant venues where publications about relevant tools could appear. For each venue we investigated its main sessions and especially the posters and demo tracks. We also included the workshops listed below, other workshops were not investigated. We investigated publications since 2013 (published till July 13th 2017). The relevant venues were:

- World Wide Web Conference (WWW) 2013 - 2017
- International Semantic Web Conference (ISWC) 2013 - 2016
- Extended Semantic Web Conference (ESWC) 2013 - 2017
- SEMANTiCS 2013 - 2016
- OnTheMove Federated Conferences (OTM) 2013 - 2016
- International Conference on Web Engineering (ICWE) 2013 - 2016
- Linked Data on the Web workshop (LDOW) 2013 - 2017
- International Workshop on Consuming Linked Data (COLD) 2013 - 2016
- International Conference on Information Integration and Web-based Applications & Services (iiWAS) 2013 - 2016

In addition, we investigated the following journals for papers no older than 2013, the newest ones being accepted for publication on July 13th 2017:

- Semantic Web journal (SWJ)

– Journal of Web Semantics (JWS)

Based on titles and abstracts, two of the authors independently went through the investigated venues and picked potentially interesting candidate papers. Then they read the full text of the candidate papers. In the full text they searched for any mentions of an existing tool which could be used for any of the identified requirements. This resulted in 82 tools identified as potential candidates.

3.2.2. Search engines used

In addition to going through the above mentioned publication venues, we used the Scopus search engine[17] to query for more potential candidates using the following query:

For keywords, we used `TITLE-ABS-KEY((`linked data `OR` semantic data `OR` rdf `OR` software `OR` tool `OR` toolkit `OR` platform `OR` application `) AND (` consumption `OR` visualization `OR` discovery `OR` exploration `OR` analysis `OR` processing `))`. For limiting the publication year, we used `PUBYEAR > 2012 AND PUBYEAR < 2018` and for limiting the subject we used `LIMIT-TO ( SUBJAREA, "COMP" ))`.

The query returned 1416 candidate papers. Two of the authors independently went through the search result and identified additional potentially interesting candidate papers based on their title and abstract. In the full text of the candidate papers they searched for mentions of relevant tools. This resulted into 28 additional tools. All conflicts between the authors in this phase were resolved by a discussion leading to a consensus.

3.2.3. Selection of candidates

The list of potential candidates was iteratively filtered in 3 elimination rounds using preliminary criteria described below, to limit the number of tools to be later evaluated according to all our 94 criteria from Section 4. The list of all tools which passed round 1, and their progress through rounds 2 and 3 can be seen in Table 9, Table 10 and Table 11 in Appendix B.

First elimination round In the first elimination round, we eliminated potential candidates based on the following exclusion criteria, using the formula `C1 OR (C2 and C3)`:

1. `C1`: based on its available description, it does not use LD
2. `C2`: source code is not available
3. `C3`: does not have a freely accessible demo, i.e. it is online or downloadable without registration or request

Out of the 110 potential candidates, 65 were left after the first elimination round.

Second elimination round In the second elimination round, we eliminated potential candidates based on the following exclusion criteria, using the formula `C4 OR C5`:

1. `C4`: despite initial assessment based on titles and abstracts, it is clear from the text of the paper or the description on the web that the tool is not related to LD consumption
2. `C5`: cannot be installed or the demo does not work

Out of the 65 potential candidates, 40 were left after the second elimination round. For each of the 40 candidates we had its running instance available which was tested in the following elimination rounds. These criteria may seem too restrictive, however, it is crucial for the evaluated tools to be easily and freely usable or installable, e.g. without registration, so that they can be assessed and used without the need for interaction with their maintainers.

Third elimination round In the third elimination round, we eliminated potential candidates based on the following exclusion criteria, using the formula `C6 AND C7 AND C8`:

1. `C6`: cannot load data from provided RDF file on the Web
2. `C7`: cannot load data from provided SPARQL endpoint
3. `C8`: cannot load data from provided RDF file using file upload

For testing `C6`, we used two prepared RDF files[18][19]. Their content is identical, they differ only in the used RDF serialization format. The content consists of approx. 1 million RDF triples. For testing `C7`, we used our SPARQL endpoint[20]. It contains 260 RDF graphs with over 610 million RDF triples. Note that we do not assume that the tool will try to copy all the data in the endpoint. Instead, we assume that the tool will query the endpoint, avoiding issues which may be caused by the amount of data. For testing `C8`, we used the same files as for `C6`. Two of the evaluated tools were not able

[17] `https://www.scopus.com`

[18] `https://linked.opendata.cz/soubor/ovm/ovm.trig`

[19] `https://linked.opendata.cz/soubor/ovm/ovm.ttl`

[20] `https://linked.opendata.cz/sparql`

to load our provided files, but were able to load different files. For the purpose of our survey we consider those able to load RDF.

Out of the 40 potential candidates, 16 were left after the third elimination round.

3.3. Candidate evaluation methodology

The 16 candidates were evaluated thoroughly based on all 94 of our criteria from Section 4. The evaluation results are presented in Section 5.

Each criterion was evaluated from the point of view of a non-LD expert and then from the point of view of a LD expert according to the scenario, which can be seen in Appendix C. When the scenario was passed without expert knowledge of LD technologies, we assigned "✓✓" to the tool for this criterion. If the scenario could be passed only with expert knowledge of LD technologies, we assigned "✓" to the tool for this criterion. Finally, if the scenario could not be done in any way, we assigned "–" to the tool for this criterion.

When evaluating a tool given a criterion and its scenario, we first studied the user interface of the tool for any indication of support for that scenario in that tool. Next, we searched for evidence of support for that criterion in the available tool documentation, i.e. articles and web. Only after we failed to find any mention of the desired functionality this way, we said that the tool failed the criterion.

For criteria dealing with dataset search, when the tool allowed us to search for individual RDF resources and we were able to identify the datasets the found resources belonged to, we considered the criteria passed.

3.4. Tool score aggregation and interpretation

Once a tool is evaluated, it has a non-LD expert pass "✓✓", LD expert pass "✓" or a fail "–" mark assigned for each criterion. These detailed results can be seen in Table 6, Table 7 and Table 8 in Appendix A. Next, its *requirement score* is computed as percentage of passed criteria of each requirement. These can be seen in Table 3 for non-LD experts and in Table 4 for LD experts in Section 5. Finally, its *requirement group score* for each group of requirements is computed as an average of scores of requirements in the group. These can be seen in Table 1 for non-LD experts and in Table 2 for LD experts in Section 5. The purpose of the score aggregation is to show the focus of each particular tool regarding the steps of the LD consumption process, both for non-LD and LD experts.

4. Requirements and evaluation criteria

We define the following steps of LD consumption process:

1. discovery of candidate datasets available on the web on the base of a given user's intent and selection of datasets needed to fulfill the intent,
2. extraction of data from the relevant datasets,
3. manipulation of the extracted data, e.g. cleansing, linking, fusion, and structural and semantic transformation,
4. output of the resulting data to a required format.

Similar processes are considered in the related literature. For example, in [38] the authors describe the following process:

(a) correctly identify relevant resources,
(b) extract data and metadata that meet the user's context and the requirements of their task or end goal,
(c) pre-process this data to feed into the visualization tool available and the analysis required,
(d) carry out initial exploratory analysis, using analytical and/or mining models.

Compared to our definition of the data consumption process, the process proposed in [38] concentrates more on data visualization and analysis. Nevertheless, it is comparable to ours. Step (a) corresponds to our step 1. Step (b) corresponds to our steps 1 and 2. Step (c) corresponds to our steps 3 and 4. Finally, step (d) is not covered by our process. We consider that the user works with the data as described in (d). However, this is outside of LDCP as it is performed by the user in an external tool which is not part of LDCP. Therefore, we do not evaluate this step.

There is also non-scientific literature describing the LD consumption process. In [16], the authors specify the following process:

(a) specify concrete use cases,
(b) evaluate relevant data sets,
(c) check the respective licenses,
(d) create consumption patterns,
(e) manage alignment, caching and updating mechanisms,
(f) create mash ups, GUIs, services and applications on top of the data

This process can also be aligned with ours. Steps (a) and (b) correspond to our step 1 - in (a) the users formulate their intent, in (b) they search for relevant datasets. Step (c) is a part of our step 3 (see Section 4.2.4). In

step (d) the users identify and select relevant data from the datasets which corresponds to our step 2. Step (e) is a part of our step 3. Step (f) contains our step 4 and extends it with the actual creation of a service on top of the data. We expect that the users work with the data created in step 4 of our process in the same way as in step (f) from [16] but it is out of the scope of our process.

Our described process is generic and any data consumption process may be described in these steps, not only the LD consumption process. However, an LDCP may benefit from the LD principles in each step.

In the rest of this section, we describe the requirements for each of the described steps. In this paper, we aim explicitly on the requirements which may somehow benefit from LD principles and related technologies. For each requirement, we further specify evaluation criteria. The requirements are split into 5 requirement groups.

1. Dataset discovery and selection - techniques allowing users to actually find LD needed to fulfill their intent are detailed in Section 4.1.
2. Data manipulation - techniques supporting users in manipulating the found LD are described in Section 4.2.
3. Data visualization - techniques allowing users to visualize LD are identified in Section 4.3
4. Data output - techniques allowing users to output the result of consumption process in other tools, both LD and non-LD, are identified in Section 4.4
5. Developer and community support - properties allowing users to share plugins and projects created in the tools and properties allowing developers to integrate and reuse the tools are specified in Section 4.5.

Therefore, there are no efficiency or performance requirements specified here. Once enough tools implement at least the minimal requirements identified here, their efficiency and performance may be evaluated.

We also do not evaluate how well a certain criterion is implemented regarding the user interface of the tool. What we evaluate is simply whether or not the tool implements at least some approach to address the criterion and whether LD expert knowledge is required or not.

We identified 34 requirements based on the existing tools and literature, split further into 94 fine grained criteria.

4.1. Dataset discovery

The consumption process starts with the user's intent to discover datasets published on the web which contain the required data. Therefore, an LDCP should be able to gather metadata about available datasets and provide a search user interface on top of the metadata.

4.1.1. Gathering metadata

The primary goal of dataset discovery is to provide the users with information about existence of datasets which correspond to their intent. The provided information should consist of metadata characterizing the datasets, such as title, description, temporal and spatial information, URL where it can be accessed, etc. In this section, we define requirements on how an LDCP should gather such metadata.

The most natural way to gather such information is to load dataset metadata from existing data catalogs. They usually provide a machine readable API, which an LDCP can use. Currently, the most wide-spread open data catalog API is the proprietary non-RDF *CKAN API*[21]. For example, open data catalogs implemented by CKAN[22] or DKAN[23] open-source platforms support this API.

Besides the proprietary CKAN API metadata representation, more or less standardized RDF vocabularies for representing dataset metadata exist. First, there is the DCAT [49] vocabulary, a W3C Recommendation, and DCAT-AP v1.1, the DCAT application profile for European data catalogs recommended by European Commission, providing the vocabulary support for dataset metadata. There are existing data catalogs utilizing these vocabularies for providing access to dataset metadata. Examples of such data catalogs are the European Data Portal[24], integrating dataset metadata from various national data catalogs and the European Union Open Data Portal[25] providing information about the datasets from the institutions and other bodies of the European Union. Besides DCAT, there is also the VoID [2] vocabulary, a W3C interest group note for representation of metadata about LD datasets. The metadata using these vocabularies is then accessible by the usual LD ways, i.e. in a data dump, in a SPARQL endpoint or using IRI dereferencing.

[21] http://docs.ckan.org/en/latest/api/
[22] https://ckan.org/
[23] http://getdkan.com/
[24] https://www.europeandataportal.eu
[25] https://open-data.europa.eu

A LDCP should therefore support loading dataset metadata using a data catalog API and/or the LD principles and vocabularies. This leads us to our first requirement, Req. 1.

Requirement 1 (Catalog support) *Support for loading dataset metadata from wide-spread catalogs like CKAN and from standardized metadata in DCAT, DCAT-AP and VoID using dereferencable dataset IRIs, data dumps and SPARQL endpoints which provide the metadata.*

Example 1.1 (CKAN API) *Given a URL of a CKAN instance, e.g.* `https://datahub.io`*, one can access the list of datasets and then the details about each of the found datasets by using the* package_list *and* package_show *functions of the CKAN Action API*[26]*, respectively.*

```
{
"success": true,
"result": [
  "0000-0003-4469-8298",
  "0000-0003-4469-8298-1",
  "0000-0003-4469-8298-2"
  ]
}
```

The detailed information about a dataset typically contains, among others, a link for downloading the data and information about its data format, i.e. a MIME-type, in case of downloadable files.

```
{
 "description": "Download",
 "format": "application/n-triples",
 "url": "http.../all_languages.tar",
 "resource_type": "file"
}
```

Therefore, LDCP can consume such API in a way which allows it to directly access the cataloged data without unnecessary user interaction. SPARQL endpoints can be discovered in a similar fashion.

Example 1.2 (DCAT-AP) *In the European Union, the DCAT-AP standard for data portals prescribes that dataset metadata should be represented in RDF, and lists classes and properties to be used. Therefore, LDCP can consume an RDF metadata dump or access a SPARQL endpoint, e.g. the European Data Portal endpoint*[27]*, to get the data access information in a standardized way.*

Criterion 1.1 (CKAN API) *The evaluated tool is able to load dataset metadata from a CKAN API.*

Criterion 1.2 (RDF metadata in dump) *The evaluated tool can exploit VoID, DCAT, DCAT-AP and/or Schema.org dataset metadata stored in RDF dumps.*

Criterion 1.3 (Metadata in SPARQL endpoint) *The evaluated tool can exploit VoID, DCAT, DCAT-AP and/or Schema.org dataset metadata stored in SPARQL endpoints.*

Criterion 1.4 (Metadata from IRI dereferencing) *The evaluated tool can dereference a* `dcat:Dataset` *or* `void:Dataset` *IRI to get the corresponding VoID, DCAT, DCAT-AP and/or Schema.org dataset metadata.*

The previous requirement assumes that metadata about datasets already exists provided through APIs of existing data catalogs. However, there is a more advanced way of gathering metadata. It is through implementation or support of a custom crawling and indexing service not necessarily relying solely on metadata found in data catalogs. Such a service can build its own index of datasets comprising automatically computed metadata based on the content of the datasets using so called *dataset profiling* techniques [47]. The computed metadata of a dataset is then called a *dataset profile*. Profiles may encompass structure of the datasets, i.e. the used classes and predicates [33], semantically related labels [117] or topics which characterize the content [1] where a topic is a resource from a well-known and highly reused LD data source, e.g. DBpedia [83] or Wikidata[28]. There are also approaches to clustering resources in a dataset. A cluster is formed by similar resources and characterization of these clusters may be included in the profile. [47] provides a detailed survey of existing RDF dataset profiling methods.

Examples of services which build dataset profiles automatically from the content of the datasets are Sindice [103] or LODStats [12]. The resulting Req. 2 can be viewed as a parallel to the development witnessed in the Web of Documents where first, there were web page catalogs and later came full-text search engines such as Google.

[26] `http://docs.ckan.org/en/latest/api/`

[27] `https://www.europeandataportal.eu/sparql`

[28] `https://www.wikidata.org`

Requirement 2 (Catalog-free discovery) *Support for automated extraction of dataset profiles from the web based on the dataset content, without using a catalog.*

Example 2.1 (Sindice) *Sindice [103] was a LD indexing service which then allowed to search the index data using the Sig.MA tool, which facilitated discovering the data sources of the search results. The service is, however, currently unavailable.*

Criterion 2.1 (Dataset profiling) *The evaluated tool is able to discover datasets without a catalog and compute their profile of any kind.*

4.1.2. Search user interface

The ability of the users to precisely express their intent strongly depends on the provided user interface (UI). In this section, we define requirements on the query language the users may use to express their intent and the assistance provided by LDCP to the users when writing search queries. The way how search queries are evaluated by LDCP is discussed later in Section 4.1.3.

Most users are familiar with UIs provided by dominating dataset discovery services - open data catalogs (e.g., CKAN). They enable the users to express their intent as a list of keywords or key phrases. The users may also restrict the initial list of datasets discovered using keywords with facets. A facet is associated with a given metadata property, e.g. *publisher* or *date of publication*, and enables the users to restrict the list only to datasets having given values of this property. The provided facets are determined by the structure of the collected metadata. Alternatively, a UI may provide a search form structured according to the structure of collected metadata about datasets.

These features are independent of the methods of gathering the metadata described in Section 4.1.1 and they can be offered without building on top of the LD principles. However, there exist many approaches in the literature which show how the UI may benefit from LD principles to better support the user. Even though they are primarily intended for querying the content of datasets, they could be directly used by LDCP for dataset discovery. We show examples of these works in the rest of this section and formulate requirements on LDCP based on them.

Supported query language. As argued in [131], expressing the user's intent with a keyword-based query language may be inaccurate when searching for data. It shows that building the user search interface on top of LD principles may help with increasing accuracy of the queries. Theoretically, it would be possible to offer SPARQL, a query language for RDF data, as a search query language. However, this language is too complex and hard to use for common users [25], including our intended users such as data journalists.

There are two approaches to this problem in the literature. The first approach tries to provide the users with the power of SPARQL while hiding its complexity and enables them to express their intent in a natural language. The query is then translated to a SPARQL expression. For example, in [139], the authors parse a natural language expression to a generic domain independent SPARQL template which captures its semantic structure and then they identify domain specific concepts, i.e. classes, predicates, resources, combining NLP methods and statistics. In [45], the authors propose a natural language syntactical structure called Normalized Query Structure and show how a natural language query can be translated to this normalized structure and from here to SPARQL. In SimplePARQL [43], parts of the SPARQL query can be written as keywords, which are later rewritten into pure SPARQL.

The second approach offers search query languages which do not aim to be as expressive as SPARQL, but they somehow improve the expressive power of the keywords based languages. They achieve this by extending them with additional concepts. For example, [131] proposes a search query language which supports so called predicate-keyword pairs where a keyword may be complemented with RDF predicates which are related to the keywords in the searched datasets. In [54], the authors propose a novel query language NautiLOD which enables to specify navigation paths which are then searched in the datasets. A navigation path expression is simpler and easier to specify than a SPARQL query even though NautiLOD navigation paths are more powerful then SPARQL navigation paths.

Requirement 3 (Search query language) *Provide a user friendly search interface.*

Example 3.1 (Sparklis) *Sparklis[53]*[30] *provides a user interface which supports a restricted english as a search query language. A screenshot is displayed in Figure 1 with the query Give me every food that is the industry of Coffee Republic. The query expression must use labels of classes, predicates and individuals present in the queried dataset.*

[30] http://www.irisa.fr/LIS/ferre/sparklis/

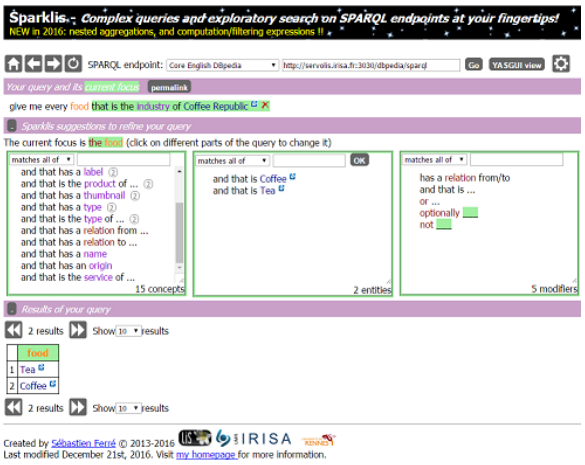

Fig. 1. SPARKLIS[29] user interface for formulating SPARQL queries

Criterion 3.1 (Structured search form) *The evaluated tool provides a form structured according to the structure of metadata considered by the tool where the user's intent can be expressed by restricting possible values in metadata profiles of searched datasets.*

Criterion 3.2 (Fulltext search) *The evaluated tool provides an input field where the user's intent can be expressed as a set of keywords or key phrases.*

Criterion 3.3 (Natural language based intent expression) *The evaluated tool supports expressing user's intent in a natural language.*

Criterion 3.4 (Formal language based intent expression) *The evaluated tool supports expressing user's intent in a formal language less complex than SPARQL.*

Assistance provided to the user when writing the query. Most users are familiar with behavior of common search engines which auto-suggest parts of queries to assist the users when expressing their intent. Also, facets may be considered as a way of assisting the users when formulating their intent. Similar assistance can also be provided in the scenario of dataset discovery. For example, [131] auto-suggests predicates relevant for a given keyword in the searched datasets. In [129], the authors show how it is possible to assist users when formulating a natural language query by auto suggesting the query not based on query logs (which is a method used by conventional web search engines) but based on the predicates and classes in the searched RDF datasets. In [53], auto-suggestions are provided by the tool automatically, based on a chosen SPARQL endpoint with a knowledge base. Faceted search over LD is studied in [7] where the authors present a tool which generates facets automatically based on the content of indexed datasets. A detailed survey of faceted exploration of LD is provided in [138].

Requirement 4 (Search query formulation assistance) *Assist the user when formulating search query with auto-suggestion of parts of the query or with facets derived automatically from the content of indexed datasets.*

Example 4.1 (Search query formulation assistance) *The user interface provided by Sparklis (see Figure 1) assists the user with formulating graph pattern expressed in a restricted natural english by providing labels of classes, labels of their individuals as well as their predicates.*

Criterion 4.1 (Seach query automated suggestions) *The evaluated tool assists the user when formulating search query with automated suggestions of parts of the query.*

Criterion 4.2 (Faceted search) *The evaluated tool assists the user when formulating search query with facets.*

4.1.3. Search query evaluation

In this section we define requirements on how LDCP evaluates a user's intent expressed as a search query using the provided search user interface. Evaluation means that LDCP identifies a list of datasets corresponding to the intent. The simplest solution is provided by the existing open data catalogs. Keywords or other metadata values filled in a metadata search form are matched against metadata records. Further restrictions expressed by the user using facets are evaluated by simple filtering of the metadata records of the datasets.

The current literature shows how this basic evaluation can be extended on the base of LD. The first possibility is to interpret the semantics of keywords or key phrases in the search query against given knowledge bases. The keywords are represented with semantically corresponding resources, i.e. individuals, concepts, classes or properties, from the given knowledge bases, possibly including their context. In [25], the authors first discover the semantics of entered keywords by consulting a given pool of ontologies to find a syntactical match of keywords and labels of resources in these ontologies. Then they automatically formulate a formal query based on the semantics. In [151], the authors discover semantics of keywords by matching

them to extraction ontologies, i.e. ontologies which are augmented linguistically to enable information extraction from texts. LDCP may use these approaches in two ways - it may add semantically similar concepts to the search query or it may add semantically related ones as discussed in [150]. Both possibilities mean that the query provided by the user is expanded. Therefore, we speak about *query expansion*. Query expansion may improve precision and recall if considered carefully which is studied in [82].

For query expansion, LDCP may also benefit from approaches for computing semantic relatedness of resources. For example, [42] shows how a semantic relatedness may be computed in a scalable way based on graph distance between two resources (average length of paths connecting the resources in a graph).

Requirement 5 (Search query expansion) *Support for expanding search queries with additional semantically related resources.*

Example 5.1 (Search query expansion) *Using a knowledge base such as Wikidata*[31]*, a LDCP may recognize terms in the search query as concepts from the knowledge base. For instance, it may recognize the term* Czech Republic *as a country*[32]*. It may have several query expansion rules encoded, such as* when a country is identified in the intent it may be expanded with contained administrative territorial entities[33]*. When a user searches for datasets with demography of the Czech Republic, such query expansion would mean that also datasets with demography of administrative entities of the Czech Republic would be searched for.*

Criterion 5.1 (Search query expansion) *The evaluated tool is able to expand a search query provided by a user with additional resources which are semantically related to the resources in the original search query.*

Another possibility is to extend the discovered list of datasets. It means that after the search query is evaluated (either expanded or not) the resulting list of discovered datasets is extended with additional semantically related datasets. By the term "semantically related" we mean various kinds of semantic relationships among datasets. A simpler solution is to search for additional datasets which provide statements about the same resources, use the same vocabularies or link their resources to the resources in the already discovered datasets. These relationships are explicitly specified as RDF statements in the datasets. A more complex solution is to derive semantic relationships among datasets based on the similarity of their content [147].

LDCP may also profit from approaches which recommend datasets for linking based on dataset similarity. For example, [48] introduces a dataset recommendation method based on cosine similarity of sets of concepts present in datasets. Similarly, the authors of [89] recommend datasets based on the similarity of labels present in datasets. Having discovered a dataset, LDCP may use such approach to recommend other semantically similar datasets. Instead of recommending them for linking, they could be recommended as additional datasets possibly interesting for the user.

We may also get back to simpler solutions and consider metadata profiles which may also contain some explicitly expressed semantic relationships between datasets. E.g., DCAT-AP discussed in Section 4.1.1 enables data publishers to specify semantic relationships between datasets in their metadata descriptions.

The discovery of related datasets is understood as one of the key features LDCP should have for example in [81], where the authors show how a possibility of recommending additional semantically related datasets may improve the quality of scientific research in domains such as medicine or environmental research by supporting interdisciplinary research.

Requirement 6 (Search result extension) *Support for recommendation of additional datasets semantically related to already discovered datasets.*

Example 6.1 (Search result extension) *DCAT-AP considers the following predicates from the Dublin Core Vocabulary which can be used to express semantic relationships between datasets:* `hasVersion` *(a dataset is a version of another dataset),* `source` *(a dataset is a source of data for another dataset), and* `relation` *(a dataset is somehow related to another dataset). These relationships define a context which can be used to extend the set of discovered datasets. A LDCP may provide additional datasets to a discovered one which contain other versions of the discovered dataset, other datasets with the content derived from the discovered one, etc.*

Criterion 6.1 (Search result extension) *The evaluated tool is able to extend the list of datasets discovered by evaluating a search query provided by a user with additional semantically related datasets.*

[31] `http://www.wikidata.org`
[32] `https://www.wikidata.org/wiki/Q213`
[33] `https://www.wikidata.org/wiki/Property:P150`

4.1.4. Displaying search results

LDCP needs to display the discovered list of datasets to the user. For this, it needs to order the discovered list and display some information about each dataset. The current dataset discovery solutions, i.e. data catalogs, order the discovered list according to the keywords based similarity of the intent of the user, and metadata descriptions of the discovered datasets. Usually, they display an ordered list of datasets with basic metadata such as the name, description, publication date, and formats in which the dataset is available.

However, there are more advanced ways of displaying the search results which can be built on top of LD.

First, discovered datasets may be ranked and their list may be ordered according to this ranking. For example, in [41], the authors modify a well-known PageRank algorithm for ranking datasets. Dataset ranking can also be based on their quality scores surveyed in [149]. In [82], the authors also propose an algorithm based on PageRank but they moreover consider the relevance to the intent of the user. A review of ranking approaches is available in [66].

Requirement 7 (Dataset ranking) *Support for dataset ranking based on their content, including quality, context such as provenance and links to other datasets, and the intent of the user.*

Example 7.1 (Dataset ranking) *For example, a LDCP may combine a PageRank score of a dataset with the relevance of publishers of datasets where datasets published by public authorities are ranked higher than datasets published by other organizations or individuals.*

Criterion 7.1 (Content-based dataset ranking) *The evaluated tool supports ranking of datasets based on their content.*

Criterion 7.2 (Context-based dataset ranking) *The evaluated tool supports ranking of datasets based on their context.*

Criterion 7.3 (Intent-based dataset ranking) *The evaluated tool supports ranking of datasets based on the intent of the user.*

Second, it is possible to show a (visual) preview or summary of the discovered datasets [20] to help the user to select the appropriate datasets. For example, a timeline showing the datasets according to their creation dates, the dates of their last modification, their accrual periodicities or their combination. Other types of previews are a map showing their spatial coverage, a word cloud diagram from their keywords, etc.

The techniques which would allow a LDCP to create such previews of the datasets include support for the appropriate RDF vocabularies. Some of the LD vocabularies recently became W3C Recommendations[34], which are de facto standards, and as such should be supported in LDCP. These include the Simple Knowledge Organization System (SKOS) [17], The Organization Ontology (ORG) [113], the Data Catalog Vocabulary (DCAT) and The RDF Data Cube Vocabulary (DCV) [114]. Having the knowledge of such well-known vocabularies, LDCP can generate vocabulary specific preview of each of the candidate datasets, giving the user more information for his decision making. There are also many well-known vocabularies which are not W3C Recommendations. Some of them are on the W3C track as Group notes, e.g. VoID [2] used to describe RDF datasets and vCard Ontology [92] for contact information. Others are registered in Linked Open Vocabularies [141], e.g. Dublin Core [39], FOAF [28], GoodRelations [62] and Schema.org[35]. These vocabularies are also popular as shown by the LODStats service [12][36].

Requirement 8 (Preview based on vocabularies) *Support for dataset preview based on well-known vocabularies used to represent the dataset content or metadata.*

Example 8.1 (Preview using the RDF Data Cube Vocabulary) *Given a statistical dataset modeled using the RDF Data Cube Vocabulary, a dataset preview (Req. 8) could contain a list of dimensions, measures and attributes or a list of concepts represented by them, based on its Data Structure Definition (DSD). In addition, the number of observations could be displayed and, in the case of simpler data cubes such as time series, a graphical line graph or bar chart can be shown automatically, as in LinkedPipes Visualization [73].*

Criterion 8.1 (Preview – W3C vocabularies) *The evaluated tool provides a dataset preview based on used vocabularies that are W3C Recommendations.*

Criterion 8.2 (Preview – LOV vocabularies) *The evaluated tool provides a dataset preview based on used well-known vocabularies registered at Linked Open Vocabularies.*

[34] https://www.w3.org/standards/techs/rdfvocabs#w3c_all

[35] http://schema.org

[36] http://stats.lod2.eu/

Criterion 8.3 (Preview metadata) *The evaluated tool previews dataset description and statistics metadata recorded in DCAT, DCAT-AP, Schema.org and/or VoID.*

The user may also require a preview compiled not based on descriptive and statistical metadata provided by the publisher about the dataset but based on metadata automatically computed from the dataset content. This metadata include descriptive metadata (e.g., important keywords or spatial and temporal coverage), statistical metadata (e.g., number of resources or numbers of instances of different classes) and schema. It it important to note that schemas are often very complex and their summaries are more useful for users. [137,105] propose techniques for schema summarization which create a subset of a given dataset schema which contains only the most important nodes and edges. Therefore, it is a reasonable requirement that LDCP should be able to generate metadata independently of what is contained in the manually created metadata records.

Requirement 9 (Data preview) *Provide dataset description, statistics and schema based on automatic querying the actual content of the dataset.*

Example 9.1 (Data preview) *A preview of data based on the data itself instead of relying on the metadata can be seen in the LODStats [12] dataset. From statistics such as the number of triples, entities, classes, languages and class hierarchy depth, one can get a better information contributing to the decision on whether or not to use such a dataset.*

Example 9.2 (Schema extraction using SPARQL) *For RDF datasets, the list of used vocabularies can be attached using the* `void:vocabulary`[37] *predicate. Their specifications can also be attached to RDF distributions of datasets using the* `dcterms:conformsTo` *predicate as specified in DCAT-AP. Either way, even if the vocabularies are indeed specified in the metadata, the dataset itself may use only few properties and classes from them, not all covered by the vocabularies. Therefore, a useful feature of a LDCP would be extraction of the actual schema of the data, i.e. used classes and predicates connecting their instances shown in a diagram. This can be done by querying the data using quite simple SPARQL queries such as this one for classes and numbers of their instances:*

```
SELECT ?class (COUNT(?s) as ?size)
WHERE {?s a ?class}
ORDER BY DESC(?size)
```

and this one for predicates among classes with number of their occurences:

```
SELECT ?c1 ?c2
(MIN(?p) as ?predicate)
(COUNT(?p) AS ?num)
WHERE {
?s1 a ?c1 .
?s2 a ?c2 .
?s1 ?p ?s2 .
}
GROUP BY ?c1 ?c2
ORDER BY DESC(?num)
```

Criterion 9.1 (Preview data) *The evaluated tool provides dataset description and statistics based on automatic querying the actual data.*

Criterion 9.2 (Schema extraction) *The evaluated tool provides schema extracted automatically from the given dataset.*

Besides metadata, statistics and used vocabularies there are many other criteria regarding data quality that should also be presented to the user to support his decision making. We do not aim at specific quality indicators. The following requirement related to data quality is generic and it considers any meaningful LD quality indicator. There is a recent survey of such indicators [149] and techniques for their computation.

Requirement 10 (Quality indicators) *Provide quality measurements based on LD quality indicators.*

Example 10.1 (Checking of availability) *One of the quality dimensions surveyed in [149] is* availability. *A LDCP could support its users by periodically checking the availability of known datasets by trying to access dumps or by querying SPARQL endpoints and dereferencing IRIs. Data gathered this way could be presented e.g. as availability percentage. The SPARQLES service [142]* [38] *performs such measurements periodically for known SPARQL endpoints.*

Criterion 10.1 (Quality indicators) *Provide quality measurements based on LD quality indicators.*

[37] `https://www.w3.org/TR/void/#vocabularies`

[38] `http://sparqles.ai.wu.ac.at/`

4.2. Data manipulation

A common problem of LD tools out there is their limited support for standards of representation and access to LD. LDCP should support existing serialization standards on its input such as Turtle and support only loading the input in different forms (e.g., dump files, SPARQL endpoints, etc.). In this section, we motivate and define requirements on data input, its transformations, provenance and licensing support, with emphasis on existing web standards.

4.2.1. Data input

LDCP should support all the standard ways of accessing LD so that there are no unnecessary constraints on the data to be consumed. These include IRI dereferencing (Req. 11), SPARQL endpoint querying (Req. 12 and the ability to load RDF dumps in all standard RDF 1.1 serializations[39] (Req. 13), i.e. RDF/XML, Turtle, TriG, N-Triples, N-Quads, RDFa and JSON-LD. This functionality can be achieved relatively easily, e.g. by integrating Eclipse RDF4J[40] or Apache Jena[41] libraries.

There are at least three ways of how LDCP should support IRI dereferencing. First, LDCP should be able to simply access a URL and download a file in any of the standard RDF serializations, which we deal with later in Req. 13. Second, LDCP should be able to use HTTP Content Negotiation[42] to specify the request for RDF data even though the URL yields an HTML page for web browsers, and be able to work with the result, e.g. to dereference related entities, a.k.a. the "follow your nose" principle[43].

Requirement 11 (IRI dereferencing) *Ability to load RDF data by dereferencing IRIs using HTTP content negotiation and accepting all RDF 1.1 serializations.*

Example 11.1 (Usage of IRI dereferencing) *One of the possible data inputs can be a single RDF resource, e.g. the representation of Prague in DBpedia, for which the user might want to monitor, some property, e.g. population. The easiest way to do that is to simply access the resource IRI, to support following HTTP redirects and to ask for RDF data in our favorite RDF serialization (Req. 11). This process can be then repeated as necessary. It is an equivalent of running, e.g.:* `curl -L -H "Accept: text/turtle" http://dbpedia.org/resource/Prague`

Criterion 11.1 (IRI dereferencing) *Ability to load RDF data by dereferencing IRIs using HTTP content negotiation.*

Criterion 11.2 (Crawling) *Ability to apply the* follow your nose *principle*

Publishing data via a SPARQL endpoint is considered the most valuable way of publishing LD for the consumers. It provides instant access to the computing power of the publishers infrastructure and the consumer does not have to work with the data at all to be able to query it. Publishing data this way also enables users to do federated queries [112]. Therefore, this should be the easiest way of getting data using a LDCP. On the other hand, public SPARQL endpoints typically implement various limitations on the number of returned results or query execution time, which make it hard to get, e.g. a whole dataset this way. These limits, however, can be tackled using various techniques such as paging.

Requirement 12 (RDF input using SPARQL) *Ability to load data from a SPARQL endpoint.*

Example 12.1 (Accessing the DBpedia SPARQL endpoint) *Since DBpedia is the focal point of the whole LOD cloud, it is often used as a linking target to provide more context to published data. In addition to IRI dereferencing, a LDCP can use the DBpedia SPARQL endpoint to, e.g., suggest more related data of the same type, etc.*

Criterion 12.1 (SPARQL: querying) *Ability to query a given SPARQL endpoint using a SPARQL query or a set of queries provided by the tool.*

Criterion 12.2 (SPARQL: named graphs) *Awareness of named graphs in a SPARQL endpoint and ability to work with them.*

Criterion 12.3 (SPARQL: custom query) *Ability to extract data from a SPARQL endpoint using a custom SPARQL query.*

Criterion 12.4 (SPARQL: authentication) *Ability to access SPARQL endpoints which require authentication.*

[39] https://www.w3.org/TR/rdf11-new/#section-serializations
[40] http://rdf4j.org/
[41] https://jena.apache.org/
[42] https://www.w3.org/Protocols/rfc2616/rfc2616-sec12.html
[43] http://patterns.dataincubator.org/book/follow-your-nose.html

Criterion 12.5 (SPARQL: advanced download) *Ability to download a dataset from a SPARQL endpoint even though it is too big to download using the standard* `CONSTRUCT WHERE {?s ?p ?o}`*, e.g. by paging or similar techniques.*

Requirement 13 (RDF dump load) *Ability to load data from an RDF dump in all standard RDF serializations, including tabular data with CSVW mapping and relational databases with R2RML mapping.*

The previous techniques such as SPARQL querying and IRI dereferencing relied on the fact that the target service was able to provide the desired RDF data serialization. However, when there are RDF dumps already serialized, to be downloaded from the web, LDCP needs to support all standardized serializations itself, as it cannot rely on any service for translation.

Example 13.1 (RDF quads dump load) *One of the use cases for named graphs in RDF already adopted by LD publishers is to separate different datasets in one SPARQL endpoint. It is then quite natural for them to provide data dumps in an RDF quads enabled format such as N-Quads or TriG, e.g. statistical datasets from the Czech Social Security Administration*[44]*. This also eases the loading of such data to a quad-store - the user does not have to specify the target graph to load to. Since there are multiple open-source libraries and quad-stores supporting RDF 1.1 serializations, this should not be a problem and LDCP should support all of them.*

Criterion 13.1 (RDF/XML file load) *Ability to load data from an RDF/XML file, both from a URL and locally.*

Criterion 13.2 (Turtle file load) *Ability to load data from a Turtle file, both from a URL and locally.*

Criterion 13.3 (N-Triples file load) *Ability to load data from an N-Triples file, both from a URL and locally.*

Criterion 13.4 (JSON-LD file load) *Ability to load data from a JSON-LD file, both from a URL and locally.*

Criterion 13.5 (RDFa file load) *Ability to load RDF data from an RDFa annotated file, both from a URL and locally.*

Criterion 13.6 (N-Quads file load) *Ability to load data from an N-Quads file, both from a URL and locally.*

Criterion 13.7 (TriG file load) *Ability to load data from a TriG file, both from a URL and locally.*

Criterion 13.8 (Handling of bigger files) *Ability to process bigger RDF dumps without loading them as RDF models. These techniques can be found in RDFpro [35] and RDFSlice [90].*

Criterion 13.9 (CSV on the Web load) *Ability to load data from a CSV file described by the CSV on the Web (CSVW) metadata, both from a URL and locally.*

Criterion 13.10 (Direct mapping load) *Ability to load data from a relational database using Direct mapping [22].*

Criterion 13.11 (R2RML load) *Ability to load data from a relational database using R2RML mapping.*

Besides these traditional ways of getting LD, LDCP should support the Linked Data Platform [9] specification for getting resources from containers. These can be, e.g. notifications according to the Linked Data Notifications W3C Recommendation [29,30].

Requirement 14 (Linked Data Platform input) *Ability to load data from the Linked Data Platform compliant servers.*

Example 14.1 (Loading data from Linked Data Platform) *The Linked Data Platform (LDP) specifications defines (among others) containers*[45] *as lists of entities. Therefore, a LDCP should be aware of this and support the handling of LDP containers, e.g. by listing them and their entities to be used in the data consumption process. An example of a LDP compliant implementation is Apache Marmotta*[46]*.*

Criterion 14.1 (Linked Data Platform containers) *Ability to load items from the Linked Data Platform containers.*

In addition to be able to load data in all standardized ways, LDCP should also provide a way of saying that the data source should be periodically monitored for changes, e.g. like in the Dynamic Linked Data Observatory [67]. This includes a specification of the period-

[44] `https://data.cssz.cz/dump/nove-priznane-duchody-dle-vyse-duchodu.trig`

[45] `https://www.w3.org/TR/ldp/#ldpc`

[46] `http://marmotta.apache.org/platform/ldp-module.html`

icity of the checks and the specification of actions to be taken when a change is detected. The actions could vary from a simple user notification to triggering a whole pipeline of automated data transformations. This way, the users can see what is new in the data used for their goals and whether the data is simply updated or needs their attention due to more complex changes. A short survey of existing approaches to change monitoring is provided in [47], Section 3.7.

Requirement 15 (Monitoring of changes in input data) *Ability to periodically monitor a data source and trigger actions when the data source changes.*

Example 15.1 (Keeping data up to date) *The EU Metadata Registry (MDR)*[47] *publishes various codelists to be reused across the EU and they are updated almost weekly. A LDCP should allow the users to register a dataset to be downloaded regularly (or on demand). The interval can be set manually, or automatically from the* `dcterms:accrualPeriodicity` *property in DCAT metadata. Or, LDCP may be notified by MDR when a change occurs according to the Linked Data Notification protocol.*

Criterion 15.1 (Monitoring of input changes: polling) *Ability to periodically monitor a data source and trigger actions when the data source changes.*

Criterion 15.2 (Monitoring of input changes: subscription) *Ability to subscribe to a data source, e.g. by WebSub [57], formerly known as PubSubHubbub, and trigger actions when the data source changes.*

When working with real world data sources, it may be necessary to keep track of versions of the input data which can change in time or even become unavailable. When more complex changes happen in the data, the whole consumption pipeline could be broken. Therefore, a version management mechanism should be available in LDCP so that the user can work with a fixed snapshot of the input data and see what has changed and when. Such a mechanism could also be automated. For example, if a consuming client holds a replica of a source dataset, changes in the source may be propagated to the replica as shown in [51], creating a new version. Updates may also be triggered by events like Git commits, as can be seen in [8].

Another type of version management is support for handling a data source, which itself is available in multiple versions at one time, e.g. using the Memento [40] protocol.

Requirement 16 (Version management) *Ability to maintain multiple versions of data from selected data sources.*

Criterion 16.1 (Manual versioning support) *Ability to keep multiple versions of input data, the versions are created, deleted and switched manually.*

Criterion 16.2 (Automated versioning support) *Ability to keep multiple versions of input data, with support for automated versioning actions such as automatic synchronization of remote datasets.*

Criterion 16.3 (Source versioning support) *Ability to exploit the existence of multiple versions of a dataset at its source, e.g. by supporting the Memento protocol.*

4.2.2. Analysis of semantic relationships

The requirements discussed in the previous sections support the users in discovering individual datasets isolated from one another. However, the users usually need to work with them as with one integrated graph of RDF data. Therefore, the users expect that the datasets are semantically related with each other and that they can use these semantic relationships for the integration. If a dataset is not in a required relationship with other discovered datasets the users need to know this so that they can omit the dataset from the further processing. We have briefly discussed possible kinds of semantic relationships in Req. 6. However, while Req. 6 only required LDCP to show datasets which are somehow semantically related to the selected one, now the users expect a deeper analysis of the relationships.

Each of the existing semantic relationships among discovered datasets should be presented to the users with a description of the relationship, i.e. the kind of the relationship and its deeper characteristics. The descriptions will help the users to understand the relationships and decide whether they are relevant for the integration of the datasets or not.

When the datasets share resources or their resources are linked, the deeper characteristics may involve the information about the classes of the resources, the ratio of the number of the shared or linked resources to the total number of the resources belonging to these classes, compliance of the datasets in the statements provided about the shared resources, etc. There are approaches which provide explanations of such relationships to users. For example, [107][108] propose

[47] `http://publications.europa.eu/mdr/`

a method for generating explanations of relationships between entities in a form of most informative paths between entities. The approach can be extended to explaining relationships among datasets which contain related entities. In [128], the authors present another approach to explaining relationships between datasets on the base of so called *relatedness cores* which are dense sub-graphs that have strong relationships with resources from related datasets.

When the datasets neither share the resources nor there are links among them, the semantic relationship may be given by the fact that they share some parts of vocabularies, i.e. classes or predicates. Such relationships may be deduced from metadata collected by services such as LODStats [12].

Requirement 17 (Semantic relationship analysis) *Provide characteristics of existing semantic relationships among datasets which are important for the user to be able to decide about their possible integration.*

Example 17.1 (LODVader) *LODVader [99] shows relationships among given datasets, demonstrated on the LOD cloud. Instead of relying on existing metadata in VoID, it generates its own based on the actual data. It is able to present relationships based on links between resources as well as shared vocabularies. Unfortunately, the software seems abandoned and the demo works no longer.*

Criterion 17.1 (Shared resources analysis) *Provide characteristics of existing semantic relationships among datasets based on resources shared by the datasets.*

Criterion 17.2 (Link analysis) *Provide characteristics of existing semantic relationships among datasets based on links among resources.*

Criterion 17.3 (Shared vocabularies analysis) *Provide characteristics of existing semantic relationships among datasets based on shared vocabularies or their parts.*

As a supplemental requirement to the previous one is to support methods for automated or semi-automated deduction of semantic relationships among datasets. These methods may be useful when semantic relationships cannot be directly discovered based on statements present in the datasets but new statements must be deduced from the existing ones. There are tools like SILK [145], SLINT [100] or SERIMI [6] which support so called *link discovery* which means deducing new statements linking resources of, usually, two datasets. In theory, deduction of new linking statements is usually done using various similarity measures [135][94]. In [98] is a recent comprehensive survey of link discovery frameworks. Another family of tools supports so called *ontology matching* [104] which is a process of deducing mappings between two vocabularies. These methods may help LDCP to provide the user with more semantic relationships.

Requirement 18 (Semantic relationship deduction) *Provide support for automated or semi-automated deduction of semantic relationships among datasets.*

Example 18.1 (Link discovery using SILK) *A classic example of adding semantic relations to existing data is link discovery. Let us, for instance, consider a dataset with useful information about all Czech addresses, towns, etc. called RÚIAN. Data on the web, however, typically links to DBpedia representations of Czech cities and regions. In order to be able to use information both from the LOD cloud and from RÚIAN, the data consumer needs to map, e.g., the Czech towns in the Czech dataset to the Czech towns in DBpedia, based on their names and based on regions in which they are located. This is a task well suited for a link discovery tool such as SILK, and a LDCP should be able to support this process.*

Criterion 18.1 (Link discovery) *Provide support for automated or semi-automated deduction of links among datasets.*

Criterion 18.2 (Ontology alignment) *Provide support for automated or semi-automated ontology alignment.*

4.2.3. Semantic transformations

The user now needs to process the discovered datasets. As we discuss later in Section 4.4, this means either visualizing the datasets or exporting them with the purpose of processing in an external tool. In general, it means to transform the datasets from their original form to another one, which is necessary for such processing. We can consider transformations, which transform RDF representation of the datasets to another RDF representation or to other data models such as relational, tree, etc. In this section, we discuss the RDF to RDF transformations. The transformations to other data models are considered as outputs of the process and are covered in Section 4.4. A transformation involves various data manipulation steps which should be supported by LDCP.

First, LDCP must deal with the fact that there exist different vocabularies which model a similar part of reality. We say that such datasets are *semantically overlapping*. Let us note that a semantic overlap is the kind of semantic relationship discussed in Req. 17. Therefore, it often happens that the vocabulary used in the dataset is different from the one required for the further processing. In such situations it is necessary to transform the dataset so that it corresponds to the required vocabulary. The transformation can be based on an ontological mapping between both vocabularies or it can be based on a transformation script specifically created for these two vocabularies. This requirement is realistic when we consider well-known vocabularies such as FOAF. Schema.org, GoodRelations, WGS84_pos, and others registered in LOV.

Requirement 19 (Transformations based on vocabularies) *Provide transformations between semantically overlapping vocabularies.*

Example 19.1 (GoodRelations to Schema.org transformation) *In the process of development of the Schema.org vocabulary, a decision was made to import classes and properties of the GoodRelations vocabulary used for e-commerce into the Schema.org namespace. This decision is questionable and controversial*[48] *in the light of best practices for vocabulary creation. These suggest reusing existing parts of vocabularies instead of creating identical classes and properties or importing them into another IRI namespace. Nevertheless, the fact remains that some datasets might be using GoodRelations*[49] *and other datasets might be using Schema.org*[50] *for, e.g., the specification of opening hours of a point of sales. This is a perfect opportunity for a transformation script or component which would transform data modeled according to one of those ways to the other kind, if necessary (Req. 19).*

Criterion 19.1 (Vocabulary mapping based transformations) *Provide transformations between semantically overlapping vocabularies based on their mapping, e.g. using OWL.*

Criterion 19.2 (Vocabulary-based SPARQL transformations) *Provide transformations between semantically overlapping vocabularies, based on predefined SPARQL transformations.*

Another often required kind of data manipulation is inference. Inference can be used when additional knowledge about the concepts defined by the vocabulary is provided which can be used to infer new statements. Such knowledge is expressed in a form of so called *inference rules*. Semantic mappings between vocabularies mentioned above also form a kind of inference rules, but here we consider them in their broader sense. A specific kind of inference rules allows one to specify that two resources are the same real-world entity, i.e. the `owl:sameAs` predicate. Having such inference rule means that statements about both resources need to be fused together. Fusion does not mean simply putting the statements together but also identification of conflicting statements and their resolution [93][79].

Requirement 20 (Inference and resource fusion) *Support inference based on inference rules encoded in vocabularies or ontologies of discovered datasets.*

Example 20.1 (Inference) *Many RDF vocabularies are defined with some inference rules in place. An example of inference which would be beneficial to support in an LDCP can be the SKOS vocabulary with its* `skos:broader` *and* `skos:narrower` *properties. According to the SKOS reference, these properties are inverse properties*[51]*. For instance, a dataset may contain triples such as* `ex:a skos:broader ex:b`*, whereas, the output is required to contain* `ex:b skos:narrower ex:a` *because of limitations of another component. These inverse triples can be inferred and materialized by LDCP, based on the knowledge of SKOS.*

Example 20.2 (`owl:sameAs` fusion) *Let us have two datasets we already mentioned in Example 18.1, Czech towns from DBpedia and the Czech registry containing information about all addresses and, among others, cities of the Czech Republic and let us consider we linked the corresponding entities using the* `owl:sameAs` *links. Now if we are preparing a dataset for, e.g. visualization to a user, it might be more fitting to fuse the data about each city from both datasets than to leave the data with the linking. Disjunctive facts about one city are easy to fuse. We choose the IRI of one or the other as the target subject and copy all predicates and objects (including blank nodes) using this subject. What may require assistance are resulting conflicting statements such as different city populations. LDCP should*

[48] `http://wiki.goodrelations-vocabulary.org/GoodRelations_and_schema.org`

[49] `http://purl.org/goodrelations/v1#OpeningHoursSpecification`

[50] `http://schema.org/OpeningHoursSpecification`

[51] `https://www.w3.org/TR/skos-reference/#L2055`

support the user in this process by, at least, providing a list of conflicts and options of solving them.

Criterion 20.1 (RDFS inference) *Support inference based on inference rules encoded in the RDFS vocabulary.*

Criterion 20.2 (Resource fusion) *Support fusion of statements about the same resources from different datasets.*

Criterion 20.3 (OWL inference) *Support inference based on inference rules encoded OWL ontologies.*

Besides transformations, the user will typically need the standard operations for data selection and projection. LDCP can assist the user with specification of these operations by providing graphical representation of the required data subset, which may correspond to a graphical representation of a SPARQL query like in SPARQLGraph [127] and similar approaches.

Requirement 21 (Selection and projection) *Support assisted graphical selection and projection of data.*

Example 21.1 (Data language selection) *One type of selection may be limiting the dataset to a single language. For example, when working with the EU Metadata Registry (MDR), codelists are usually provided in all EU languages. Having the data in multiple languages may, however, not be necessary. A LDCP may support common selection techniques without requiring the user to write SPARQL queries.*

Criterion 21.1 (Assisted selection) *Support assisted graphical selection of data.*

Example 21.2 (Data projection) *An example of a projection may be picking only data which is relevant for a given use case, while omitting other properties of represented resources. The user may again be assisted by a graphical overview of the dataset with possibility to select relevant entities, their properties and links.*

Criterion 21.2 (Assisted projection) *Support assisted graphical projection of data.*

Besides the above described specific data manipulation requirements, the user may need to define own specific transformations expressed in SPARQL.

Requirement 22 (Custom transformations) *Support custom transformations expressed in SPARQL.*

Criterion 22.1 (Custom transformations) *Support custom transformations expressed in SPARQL.*

Having the support for all kinds of data manipulations described above it is necessary to be able to combine them together so that the discovered datasets are transformed to the required form. The user may be required to create such data manipulation pipeline manually. However, we might also expect that LDCP compiles such pipeline automatically and the user only checks the result and adjusts it when necessary.

Requirement 23 (Automated data manipulation) *Support automated compilation of data manipulation pipelines and enable their validation and manual adjustment by users.*

Example 23.1 (Automated data manipulation) *A LDCP may be aware of what the data in each step of the consumption process looks like even as the process is being designed, e.g. by performing background operations on data samples, and it may recommend possible or probable next steps. Using the same principles, the data transformation pipelines may be even generated automatically, given that the requirements on the final data structure are known and the source data is accessible. This can be seen in e.g. LinkedPipes Visualization [74], where given an RDF data source, a list of possible visualizations formed by data transformation pipelines, is offered automatically, based on known vocabularies and input and output descriptions of the data processing components.*

Criterion 23.1 (Automated data manipulation) *Support automated compilation of data manipulation pipelines and enable their validation and manual adjustment by users.*

4.2.4. Provenance and license management

For the output data to be credible and reusable, detailed provenance information should be available capturing every step of the data processing task, starting from the origin of the data to the final transformation step and data export or visualization. There is The PROV Ontology [124], a W3C Recommendation that can be used to capture provenance information in RDF.

Requirement 24 (Provenance) *Support for processing provenance information expressed using a standardized vocabulary.*

Example 24.1 (Provenance) *When the goal of the LD consumption process is to gather data, combine it and pass it to another tool for processing, it might be useful to store information about this process. Specifically, the consumers of the transformed data might be interested in where the data comes from, how it was combined, what selection and projection was done, etc. This information can be stored as an accompanying manifest using the PROV Ontology and may look like this (taken directly from the W3C Recommendation):*

```
ex:aggregationActivity
   a prov:Activity;
   prov:wasAssociatedWith ex:derek;
   prov:used ex:crimeData;
   prov:used ex:nationalRegionsList;
.
```

Criterion 24.1 (Provenance input) *Ability to exploit structured provenance information accompanying input data.*

Criterion 24.2 (Provenance output) *Provide provenance information throughout the data processing pipeline using a standardized vocabulary.*

Part of metadata of a dataset should be, at least in case of open data, the information about the license under which the data is published, since that is what the first star of open data is for. A common problem when integrating data from various data sources is license management, i.e. determining whether the licenses of two data sources are compatible and what license to apply to the resulting data. This problem gets even more complicated when dealing with data published in different countries. A nice illustration of the problem is given as Table 3 in ODI's Licence Compatibility[52] page. Also, the European Data Portal has its Licence Assistant[53] which encompasses an even larger list of open data licenses. Issues with (linked) open data licensing are also identified in [50], where the authors also present the License Compositor tool. Finally, LIVE [59] is a tool for verification of license compatibility of two LD datasets.

Requirement 25 (License management) *Support license management by tracking licenses of original data sources, checking their compatibility when data is integrated and helping with determining the resulting license of republished data.*

Example 25.1 (License management) *Based on open data license compatibility information such as the ODI's Licence Compatibility table, a LDCP should advise its user on whether two data sources can be combined or not and what is the license of the result. The licenses of individual datasets can be obtained from its DCAT metadata, as in, e.g. the Czech National Open Data catalog[54]. For datasets without a specific license, a warning should be displayed. For datasets with known licenses, e.g. a dataset published using CC-BY and another one published using CC-BY-SA, the CC-BY-SA minimum license should be suggested. The legal environment of open data is quite complex and obscure even to developers who are LD experts, so a support like this, preferably even region specific, is necessary.*

Criterion 25.1 (License awareness) *Awareness of dataset licenses, e.g. gathered from metadata, and their presentation to the user.*

Criterion 25.2 (License management) *Awareness of the license combination problem and assistance in determining the resulting license when combining datasets.*

4.3. Data visualization

The goal of using LDCP is either to produce data for processing in other software or to produce visualizations of discovered data sets. Given the graph nature of RDF, the simplest visualization is a graph visualization of the data, reflecting its structure in RDF. Another type of visualizations are manually specified using a mapping of the RDF data to the inputs of visualization plugins, as in LinkDaViz [134]. Finally, visualizations can be automated, based on correct usage of vocabularies and a library of visualization tools as in the LinkedPipes Visualization (LP-VIZ) [73], which offers them for DCV, SKOS hierarchies and Schema.org GeoCoordinates on Google Maps.

Requirement 26 (Visualization) *Offer visualizations of LD based on its graph structure, manual mapping from the data to the visualizations, or using automated methods for visualization.*

[52] https://github.com/theodi/open-data-licensing/blob/master/guides/licence-compatibility.md

[53] https://www.europeandataportal.eu/en/licence-assistant

[54] https://nkod.opendata.cz/sparql

Example 26.1 (Manual visualization in LinkDaViz) *An example of manual visualization is LinkDaViz. The user first selects the dataset to be visualized and then selects specific classes and properties from the dataset. Based on the selected properties and their data types, visualization types are recommended and more can be configured manually to get a visualization such as a map, a bar chart, etc.*

Example 26.2 (Automated visualization in LP-VIZ) *In contrast to manual visualization in Example 26.1, in LinkedPipes Visualization the user points to the dataset and the tool by itself, automatically, takes a look at what data is present in the dataset. It then offers possible visualizations, based on the found, well-known vocabularies supported by components registered in the tool.*

Criterion 26.1 (Graph visualization) *Offer automated visualizations of LD based on well-known vocabularies.*

Criterion 26.2 (Manual visualization) *Offer visualizations of LD based on manual mapping from the data to the visualizations.*

Criterion 26.3 (Vocabulary-based visualization) *Offer automated visualizations of LD based on well-known vocabularies.*

4.4. Data output

The other possible goal of our user is to output raw data for further processing. The output data can be in various formats, the most popular may include RDF for LD, which can be exported in various standardized ways. The first means of RDF data output is to a dump file using standardized serializations. The inability to create a dump in all standardized formats (only some) will be classified as partial coverage of the requirement.

Requirement 27 (RDF dump output) *Ability to export RDF data to a dump file in all standard RDF serializations.*

Example 27.1 (RDF dump output) *Various tools may need various RDF serializations to be able to process the output data. The main problem which can be seen quite often is that many applications consuming RDF only support triple formats and not quad formats. Therefore, a LDCP should support output in all standard RDF 1.1 serializations, which can be easily supported by using an existing library such as Apache Jena or Eclipse RDF4J.*

Criterion 27.1 (RDF/XML file dump) *Ability to dump data to an RDF/XML file.*

Criterion 27.2 (Turtle file dump) *Ability to dump data to a Turtle file.*

Criterion 27.3 (N-Triples file dump) *Ability to dump data to an N-Triples file.*

Criterion 27.4 (JSON-LD file dump) *Ability to dump data to a JSON-LD file.*

Criterion 27.5 (RDFa file dump) *Ability to dump RDF data to an RDFa annotated file.*

Criterion 27.6 (N-Quads file dump) *Ability to dump data to an N-Quads file.*

Criterion 27.7 (TriG file dump) *Ability to dump data to a TriG file.*

Another way of outputting RDF data is directly into a triplestore, using one of two standard ways, either the SPARQL Update query [109] or using the SPARQL Graph Store HTTP Protocol [102].

Requirement 28 (RDF output using SPARQL) *Ability to load RDF data to a SPARQL endpoint using SPARQL Update or SPARQL Graph Store HTTP protocol.*

Example 28.1 (Loading data using the SPARQL Graph Store HTTP Protocol) *For larger datasets, where whole named graphs are manipulated, it is simpler to use the SPARQL Graph Store HTTP Protocol to load data. Basically, a data dump is sent over HTTP to the server instead of being inserted by a SPARQL query. However, despite being a W3C Recommendation, the implementation of the protocol differs among quad stores. Therefore, besides the endpoint URL, authentication options and the target named graph, a LDCP should be also aware of the specific implementation of the endpoint. This can be seen e.g. in LinkedPipes ETL's Graph Store Protocol component*[55].

Criterion 28.1 (SPARQL Update support) *Ability to load RDF data to a SPARQL endpoint using SPARQL Update.*

Criterion 28.2 (SPARQL: named graphs support) *Awareness of named graphs in a SPARQL endpoint and ability to load data to them.*

[55] https://etl.linkedpipes.com/components/l-graphstoreprotocol

Criterion 28.3 (SPARQL: authentication support) *Ability to access SPARQL Update endpoints which require authentication.*

Criterion 28.4 (SPARQL Graph Store HTTP Protocol output) *Ability to load RDF data to a SPARQL endpoint using the SPARQL Graph Store HTTP Protocol.*

Lastly, LDCP should be able to write RDF data to Linked Data Platform compliant servers. This may also include support for W3C Recommendations building on top of the LDP, such as Linked Data Notifications [30] for sending notifications to inboxes on the web.

Requirement 29 (Linked Data Platform output) *Ability to write data to Linked Data Platform compliant servers.*

Example 29.1 (Loading data to Linked Data Platform) *The Linked Data Platform (LDP) specifications defines (among others) how new entities should be added to existing containers*[56]*. Therefore, a LDCP should be aware of this and support the handling of LDP containers. A specific use case may be sending messages according to the Linked Data Notifications W3C Recommendation.*

Criterion 29.1 (Linked Data Platform Containers Output) *Ability to upload data by posting new entities to Linked Data Platform Containers.*

Besides being able to output RDF, a perhaps even more common use case of LDCP will be to prepare LD to be processed locally by other tools, which do not know LD. One choice for non-LD data export is a CSV for tabular data. The usual way of getting CSV files out of RDF data is by using the SPARQL SELECT queries. Such CSV files can now be supplied by additional metadata in JSON-LD according to the Model for Tabular Data and Metadata on the Web (CSVW) [132] W3C Recommendation. Besides tabular data, tree-like data are also popular among 3-star data formats. These include XML and JSON, both of which the user can get, when a standardized RDF serialization in one of those formats, i.e. RDF/XML and JSON-LD, is enough. This is already covered by Req. 27. However, for such data to be usable by other tools, it will probably have to be transformed to the JSON or XML format accepted by the tools and this is something that LDCP should also support. For advanced graph visualizations it may be useful to export the data as graph data e.g. for Gephi [15]. Majority of people today do not know SPARQL and do not want to learn it. However, LDCP should still allow them to work with LD and use the data in their own tools. Therefore, LDCP should not only allow to export 3* data, but it should support doing so in a user friendly way, with predefined output formats for external tools and, ideally, predefined transformations from vocabularies used in the tool's domain. This observation comes from a recent publication of a Czech registry of subsidies, which was published as LD only, with N-Triples dumps, dereferencable IRIs and a public SPARQL endpoint[57]. This step, while being extremely interesting to semantic web researchers, was received negatively by the Czech open data community, which demanded CSV or JSON files and was not willing to learn RDF and SPARQL. This motivates Req. 30 and especially C 30.5.

Requirement 30 (Non-RDF data output) *Ability to export non-RDF data such as CSV with CSVW metadata, XML, JSON or Graph data for visualizations or further processing in external tools.*

Example 30.1 (Gephi support) *To support graph visualization in Gephi, which can effectively visualize large data graphs, LDCP should support output consumable by Gephi. Technically, this is a pair of CSV files with a specific structure. LDCP could include a wizard for creation of such files.*

Criterion 30.1 (CSV file output) *Ability to export data as CSV files.*

Criterion 30.2 (CSV on the Web output) *Ability to export data as CSV files accompanied by the CSV on the Web JSON-LD descriptor.*

Criterion 30.3 (XML file output) *Ability to export data as arbitrarily shaped XML files.*

Criterion 30.4 (JSON file output) *Ability to export data as arbitrarily shaped JSON files.*

Criterion 30.5 (Output for specific third-party tools) *Ability to export data for visualization in specific third-party tools.*

[56] `https://www.w3.org/TR/ldp/#ldpc`

[57] `http://cedr.mfcr.cz/cedr3internetv419/OpenData/DocumentationPage.aspx` (in Czech only)

4.5. Developer and community support

For most of our requirements there already is a tool that satisfies them. The problem why there is no platform using these tools is caused by their incompatibility and the large effort needed to do so. LDCP does not have to be a monolithic platform and can consist of multitude of integrated tools. However, the tools need to support easy integration, which motivates the next requirements. Each part of LDCP and LDCP itself should use the same API and configuration which it exposes for others to use. The API should again be based on standards, i.e. REST or SOAP based, and the configuration should be consistent with the rest of the data processing, which means in RDF with a defined vocabulary.

Requirement 31 (API) *Offer an easy to use, well documented API (REST-like or SOAP based) for all important operations.*

Example 31.1 (API) *For instance, LinkedPipes ETL offers REST-like API used by its own frontend to perform operations like running a transformation, creating a transformation pipeline, etc. The API is documented by OpenAPI*[58] *(f.k.a. Swagger) to help developers with its reuse. A LDCP should also have an API for all operations so that its parts are easily reusable and easily integrated with other software.*

Criterion 31.1 (Existence of API) *Ability to accept commands via an open API.*

Criterion 31.2 (API documentation) *Existence of a documentation of the API of the tool, such as OpenAPI.*

Criterion 31.3 (Full API coverage) *Ability to accept all commands via an open API.*

Requirement 32 (RDF configuration) *Offer RDF configuration where applicable.*

Example 32.1 (RDF configuration) *All configuration properties and projects of a LDCP should be also stored, or at least accessible, in RDF. This is so that this data can be read, managed and created by third-party RDF tools and also shared easily on the Web, as any other data. For instance, LinkedPipes ETL stores its transformation pipelines in RDF and makes their IRIs dereferencable, which eases their sharing and offers wider options for their editing.*

Criterion 32.1 (Configuration in open format) *Configuration stored and accepted in an open format such as JSON or XML.*

Criterion 32.2 (Configuration in RDF) *Configuration stored and accepted in RDF.*

An important quality of a tool available on the web is the nature of its community. This includes the size of the community formed around that tool, its activities and repositories for sharing tool plugins and data processes created in the tool. The community should be active and maintain the tool, or at least share new plugins or new data processes.

Since one of the ideas of LD is distribution of effort, the data processes defined in LDCP should themselves be shareable and described by an RDF vocabulary as it is with, e.g. the Linked Data Visualization Model (LDVM) pipelines [71]. For this, a community repository for sharing of LDCP plugins, i.e. reusable pieces of software, which can extend the functionality of the LDCP, like visualization components from Req. 26 and vocabulary-based transformers from Req. 19 should be present and identified for the LDCP.

In addition, the data transformation processes defined using the LDCP, i.e. the LDCP process configuration, should also be easily shareable with the community, facilitating effort distribution. An example of such a shared project could be a configuration allowing users to extract subsets of DBpedia as CSV files, etc.

Requirement 33 (Tool community) *An active community should exist around the tool in the form of repositories and contributors.*

Example 33.1 (Sharing on GitHub) *Git is a very popular tool for sharing of and collaboration on software and data, with GitHub as one of its most popular repository implementations. To spark the network effect, a LDCP should have a catalog of plugins and reusable projects, on which the community can collaborate. Since projects themselves should be stored as RDF (Req. 32), they can be easily shared on, e.g. GitHub. This can be illustrated by, e.g. LinkedPipes ETL, which has its repository of plugins*[59] *and reusable pipeline fragments*[60] *which accompany component documentation.*

[58] https://github.com/OAI/OpenAPI-Specification

[59] https://github.com/linkedpipes/etl/tree/develop/plugins

[60] https://github.com/linkedpipes/etl/tree/gh-pages/assets/pipelines

Criterion 33.1 (Repository for source code) *Existence of an openly accessible repository where the source code of the tool is developed.*

Criterion 33.2 (Repository for plugins) *Existence of an openly accessible repository of plugins such as GitHub.*

Criterion 33.3 (Repository for data processes) *Existence of an openly accessible repository of data processes such as GitHub.*

Criterion 33.4 (Community activity) *There is at least 2 commits from 2 distinct contributors in the last year to any of the tool related repositories.*

When the whole process of data gathering, processing and output is described in LDCP, it may be useful to expose the resulting transformation process as a web service, which can be directly consumed by e.g. web applications. This could facilitate live data views or customizable caching and update strategies.

Requirement 34 (Deployment of services) *Offer deployment of the data transformation process as a web service.*

Example 34.1 (Encapsulated SPARQL queries) *A very simple example of Req. 34 could be a SPARQL query, possibly with parameters, encapsulated as a web service. This is a common way of helping non-RDF applications consume live RDF data without even knowing it. However, the process behind the web service can be much more complex, and a LDCP could offer creation of such services to its users.*

Criterion 34.1 (Deployment of services) *Ability to deploy the data transformation process as a web service.*

5. Tools evaluation and description

In this section, we provide the evaluation results of 16 tools classified as LDCP according to the methodology described in Section 3.2. We evaluated all 94 criteria for each of the 16 tools. The detailed results of the evaluation of each of the 16 tools can be seen in Table 6, Table 7 and Table 8 in Appendix A. Next, as detailed in Section 3.4, its *requirement score* is computed as a percentage of passed criteria of each requirement. These can be seen in Table 3 for non-LD experts and in Table 4 for LD experts. Finally, its *requirement group score* for each group of requirements defined in Section 4 is computed as an average of scores of requirements in the group. These can be seen in Table 1 for non-LD experts and in Table 2 for LD experts.

Tool	Dataset discovery	Data manipulation	Data visualization	Data output	Developer and community support	Average
LP-VIZ		9%	33%		13%	**11%**
V2		1%	33%			**7%**
VISU						**0%**
Sp	8%	1%	67%			**15%**
YAS	5%		67%		13%	**17%**
UV		5%		31%	48%	**17%**
LP-ETL		4%		31%	52%	**18%**
OC	8%	4%	33%	25%	6%	**15%**
DL		5%		23%	13%	**8%**
LAUN		4%		4%	6%	**3%**
LDR		1%	67%		6%	**15%**
SF	10%	2%			13%	**5%**
ELD		1%	33%		6%	**8%**
PEPE		1%			6%	**2%**
FAGI		2%	33%	6%	13%	**11%**
RDFPRO		4%		28%	13%	**9%**

Table 1

Requirement group scores of evaluated candidates for non-LD experts. Empty cells represent 0%.

In this section, we first briefly describe each evaluated tool and we point out to fulfilled requirements. Then we discuss the evaluation results and identify tools which can be considered as a LDCP because they satisfy at least the minimal conditions described in the introduction. Not all evaluated tools meet even the minimal conditions. However, it does not mean that they are useless for LD consumption. We also discuss the evaluation results for these tools in the text.

5.1. Description of evaluated tools

In this section, we introduce the 16 individual tools which we evaluated using our 94 criteria.

Tool	Dataset discovery	Data manipulation	Data visualization	Data output	Developer and community support	Average
LP-VIZ		10%	33%		13%	**11%**
V2	5%	4%	33%			**8%**
VISU	5%	3%	33%	10%		**10%**
Sp	13%	1%	67%			**16%**
YAS	5%	1%	67%	5%	13%	**18%**
UV		17%		55%	48%	**24%**
LP-ETL		17%		65%	94%	**35%**
OC	23%	17%	33%	30%	6%	**22%**
DL		22%	33%	23%	13%	**18%**
LAUN	5%	4%		4%	6%	**4%**
LDR		3%	67%		6%	**15%**
SF	10%	2%			13%	**5%**
ELD		3%	33%		6%	**8%**
PEPE	5%	1%			6%	**3%**
FAGI		5%	33%	13%	13%	**13%**
RDFPRO	5%	17%		28%	25%	**15%**

Table 2

Requirement group scores of evaluated candidates for LD experts. Empty cells represent 0%.

5.1.1. LinkedPipes Visualization

LinkedPipes Visualization [74] (LP-VIZ) is a tool providing automated visualizations of LD, fulfilling Req. 8 Preview based on vocabularies and Req. 26 Visualization. It queries the input data given as an RDF dump or as a SPARQL endpoint for known vocabularies, and based on the result, it offers appropriate visualizations. These include an interactive RDF data cube visualization, various types of SKOS hierarchy visualizations and a map based visualization (see Figure 2), which can be further filtered based on concepts connected to the points on the map. In addition, in the RDF data cube visualization it also employs Req. 11 IRI dereferencing to search for labels of entities.

As to the other parts of the LD consumption process, no requirements were satisfied in the Data output and Developer and community support groups.

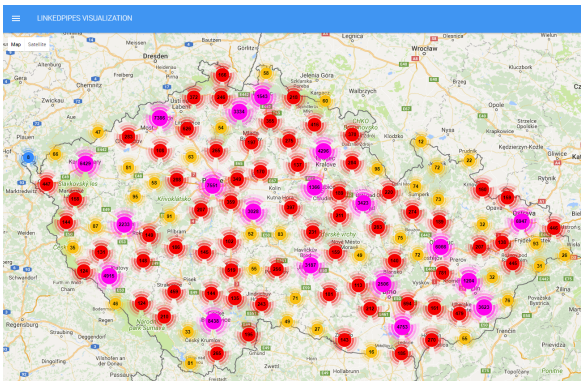

Fig. 2. LinkedPipes Visualization - Points on a map

5.1.2. V2

V2 [3] is an on-line demo for visualization of vocabularies used in RDF data. In the first step the user provides a list of SPARQL endpoints. V2 executes a set of predefined SPARQL queries to analyze the vocabularies used in the data and creates the dataset preview.

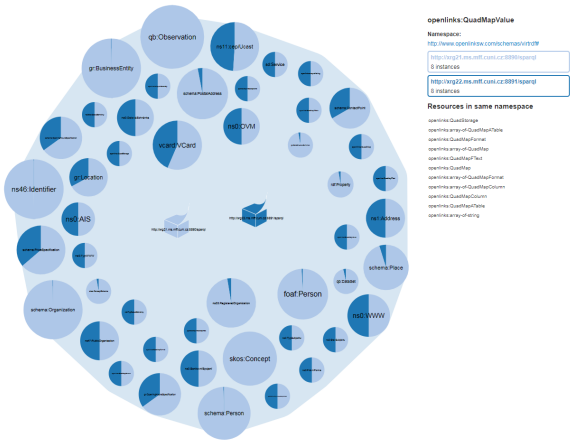

Fig. 3. V2 visualization

On the top level of the preview, the used classes are presented (see Figure 3). The user can select a class to see the information about the number of instances and about predicates used. V2 is simple, nevertheless it can be used to address the Dataset discovery part of the LD consumption process, specifically, it covers Data preview.

5.1.3. VISU

VISUalization Playground [3] (VISU) is an on-line service for visualization of SPARQL results. It uses predefined or user provided SPARQL SELECT queries to retrieve data from given SPARQL endpoints. Then

Requirement	LP-VIZ	V2	VISU	Sp	YAS	UV	LP-ETL	OC	DL	LAUN	LDR	SF	ELD	PEPE	FAGI	RDFPRO
Dataset discovery: Gathering metadata																
Catalog support								25%								
Catalog-free discovery																
Dataset discovery: Search user interface																
Search query language				25%				50%				50%				
Search query formulation assistance				50%	50%							50%				
Dataset discovery: Search query evaluation																
Search query expansion																
Search result extension																
Dataset discovery: Displaying search results																
Dataset ranking																
Preview based on vocabularies																
Data preview																
Quality indicators																
Data manipulation: Data input																
IRI dereferencing																
RDF input using SPARQL	20%	20%		20%							20%		20%	20%	20%	
RDF dump load	9%					73%	64%	55%	73%	55%		36%			9%	64%
Linked Data Platform input																
Monitoring of changes in input data																
Version management																
Data manipulation: Analysis of semantic relationships																
Semantic relationship analysis																
Semantic relationship deduction																
Data manipulation: Semantic transformations																
Transformations based on vocabularies																
Inference and resource fusion																
Selection and projection																
Custom transformations																
Automated data manipulation	100%															
Data manipulation: Provenance and license management																
Provenance																
License management																
Data visualization																
Visualization	33%	33%		67%	67%			33%			67%		33%		33%	
Data output																
RDF dump output						100%	100%	100%	71%	14%						86%
RDF output using SPARQL						25%	25%								25%	25%
Linked Data Platform output																
Non-RDF data output									20%							
Developer and community support																
API						67%	33%									
RDF configuration						50%										
Tool community	50%				50%	75%	75%	25%	50%	25%	25%	50%	25%	25%	50%	50%
Deployment of services							100%									
Average	**6%**	**2%**	**0%**	**5%**	**5%**	**11%**	**12%**	**8%**	**6%**	**3%**	**3%**	**5%**	**2%**	**1%**	**4%**	**7%**

Table 3

Requirement scores of evaluated candidates from the point of view of non-LD experts. Empty cells represent 0%. Requirements are grouped according to sub-sections of Section 4

it uses Google Charts Editor to create a wide range of visualizations (see Figure 4) based on the output table, including line charts, tree charts or map visualizations.

The main focus of VISU is at Data visualization.

5.1.4. Sparklis

Sparklis [53] (Sp) is an on-line Linked Open Data exploration tool that does not require the user to have knowledge of SPARQL. The user can specify a

Requirement	LP-VIZ	V2	VISU	Sp	YAS	UV	LP-ETL	OC	DL	LAUN	LDR	SF	ELD	PEPE	FAGI	RDFPRO
Dataset discovery: Gathering metadata																
Catalog support								25%								
Catalog-free discovery																
Dataset discovery: Search user interface																
Search query language				25%				50%				50%				
Search query formulation assistance				50%	50%			50%				50%				
Dataset discovery: Search query evaluation																
Search query expansion																
Search result extension																
Dataset discovery: Displaying search results																
Dataset ranking																
Preview based on vocabularies																
Data preview		50%	50%	50%				100%		50%				50%		50%
Quality indicators																
Data manipulation: Data input																
IRI dereferencing																
RDF input using SPARQL	40%	20%	40%	20%	20%	80%	80%	40%	60%		40%		40%	20%	40%	20%
RDF dump load	9%					73%	73%	64%	73%	55%		36%			9%	64%
Linked Data Platform input																
Monitoring of changes in input data																
Version management																
Data manipulation: Analysis of semantic relationships																
Semantic relationship analysis		33%														
Semantic relationship deduction								50%	50%							
Data manipulation: Semantic transformations																
Transformations based on vocabularies									50%							
Inference and resource fusion															33%	67%
Selection and projection																
Custom transformations						100%	100%	100%	100%							100%
Automated data manipulation	100%															
Data manipulation: Provenance and license management																
Provenance																
License management																
Data visualization																
Visualization	33%	33%	33%	67%	67%			33%	33%		67%		33%		33%	
Data output																
RDF dump output						100%	100%	100%	71%	14%						86%
RDF output using SPARQL						100%	100%								50%	25%
Linked Data Platform output																
Non-RDF data output			40%		20%	20%	60%	20%	20%							
Developer and community support																
API						67%	100%									
RDF configuration						50%	100%									50%
Tool community	50%				50%	75%	75%	25%	50%	25%	25%	50%	25%	25%	50%	50%
Deployment of services							100%									
Average	**7**%	**4**%	**5**%	**6**%	**6**%	**20**%	**26**%	**19**%	**15**%	**4**%	**4**%	**5**%	**3**%	**3**%	**6**%	**15**%

Table 4

Requirement scores of evaluated candidates from the point of view of LD experts. Empty cells represent 0%. Requirements are grouped according to sub-sections of Section 4

SPARQL endpoint and explore its contents using so called *focus*, which represents the user's intent. Its iterative refinement corresponds to the exploration process. The *focus* is defined using three types of constrains: concept, entity and modifier. The concept constraint allows the user to define subject type(s). The entity constrain restricts the focus to a single entity. Modifiers represent logical operations and filters.

In each step of the focus refinement process, the user is guided by suggestions for all three types of constrains.

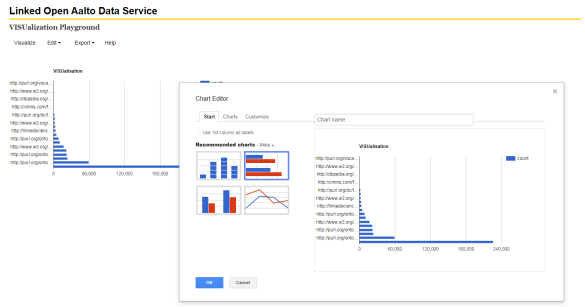

Fig. 4. VISU visualization

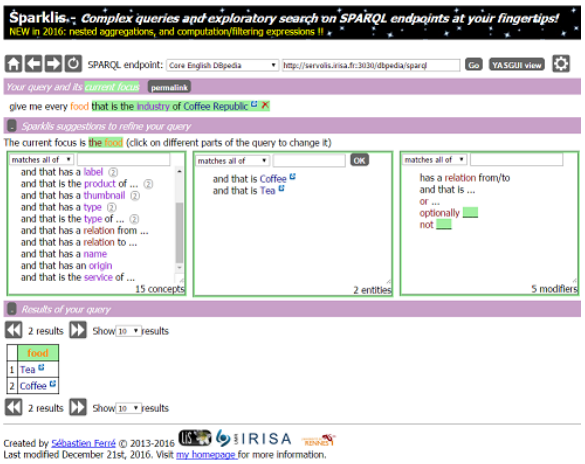

Fig. 5. Sparklis user interface for data exploration

Sparklis offers an on–the–fly data visualization in a form of a table or, in case of geolocalized data, a map. A query corresponding to the current focus is accessible in the integrated YASGUI tool, which provides additional options for visualization and data export.

From the point of view of the LD Consumption process, Sparklis covers mainly Dataset discovery and Data manipulation.

5.1.5. *YASGUI*

Yet Another SPARQL GUI [115] (YAS) is a web-based SPARQL query editor. As such it offers the ability for the user to specify a SPARQL query without knowing the precise syntax. Once the query is executed YASGUI provides different visualizations, such as table, pivot table, Google Chart and Map visualizations, and export of SPARQL SELECT results to a CSV file.

The aim of YASGUI is not to be used as a standalone tool, but rather be integrated as a component. Still, it covers several areas of the LD consumption process,

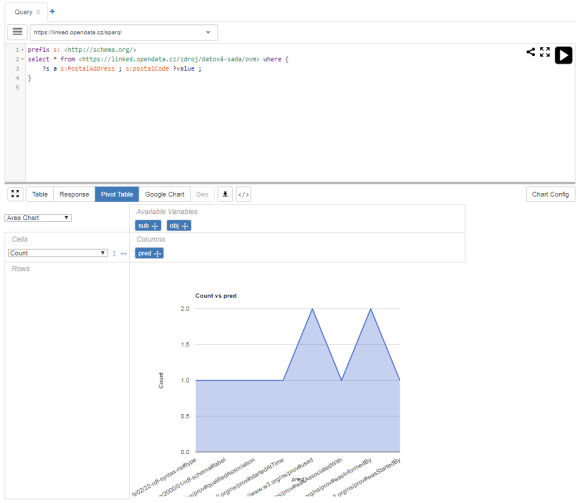

Fig. 6. YASGUI SPARQL query builder interface

mainly Data manipulation, Data visualization and Data output.

It is currently used in multiple LD based tools[61], e.g. Apache Jena Fuseki, Eclipse RDF4J and OntoText GraphDB triplestores.

5.1.6. *UnifiedViews*

UnifiedViews [80] (UV) is a predecessor of LinkedPipes ETL, which means that it is also an open–source tool for design and execution of ETL pipelines (see Figure 7), and it is also focused on RDF and linked data and features a library of reusable components. Its requirements coverage is therefore a subset of the coverage of LinkedPipes ETL.

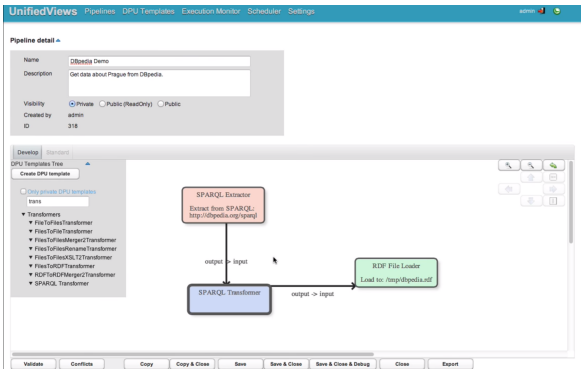

Fig. 7. UnifiedViews transformation pipeline

While it can be successfully used for LD publication, which is its primary purpose, its usage for LD consump-

[61] https://github.com/OpenTriply/YASGUI.YASQE

tion is limited to expert users only, due to its lack of proper documentation, incomplete API coverage and proprietary configuration formats.

5.1.7. LinkedPipes ETL

LinkedPipes ETL [77] (LP-ETL) is an open–source tool for design and execution of ETL (Extract, Transform, Load) pipelines (see Figure 8), with a library of more than 60 reusable components which can be used in the pipelines and which very well support the Data manipulation and Data output requirement groups. The tool is primarily focused on working with RDF data and features a thorough documentation, including how-tos and tutorials for typical LD ETL scenarios. LP-ETL also helps the user, who still has to be a LD expert though, to build a data transformation pipeline using smart suggestions of components to use. LinkedPipes ETL is developed as a replacement for UnifiedViews.

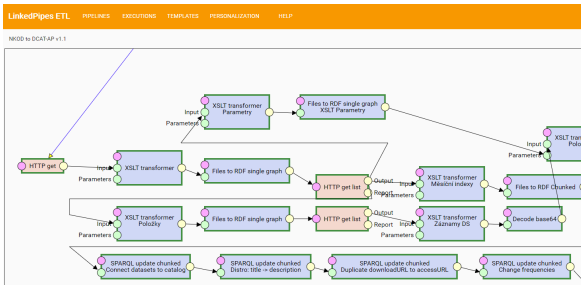

Fig. 8. LinkedPipes ETL transformation pipeline

Regarding the LD Consumption process, the most notable features are its capability of working with larger data using so called data chunking approach [76] satisfying C 12.5 SPARQL: advanced download. In the Developer and community support requirements group, the satisfaction of C 31.3 Full API coverage and C 32.2 Configuration in RDF means that the tool can be easily integrated with other systems.

LinkedPipes ETL does not support the user in Dataset discovery or Data visualization.

5.1.8. OpenCube

OpenCube [68] (OC) is an open source project, which supports publication of statistical data using the RDF Data Cube Vocabulary and also provides visualizations and data analytics on top of the statistical cubes. OpenCube wraps another tool, the Information Workbench [60], and thus it has all the functionality provided by Information Workbench (see Figure 9).

OpenCube also integrates several other tools such as R and Grafter. The created visualization can be embedded into an HTML page as an interactive script.

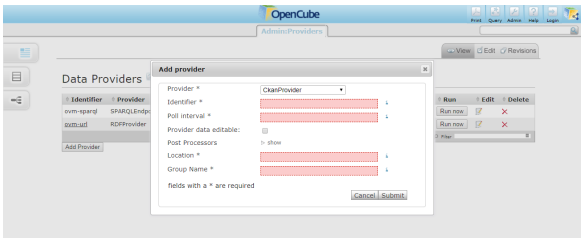

Fig. 9. OpenCube user interface in Information Workbench

5.1.9. Datalift

Datalift [52] (DL) aims at semantic lifting of non-RDF data, but it can be also used to consume RDF data. It uses projects to group different data sources together and it allows users to perform transformations such as SPARQL CONSTRUCT, String to URI, IRI translation and Silk link discovery, supporting Data manipulation. Finally, it provides visualizations and RDF and CSV data export, fulfilling Req. 26 Visualization, Req. 27 RDF dump output, and C 30.1 CSV file output, respectively.

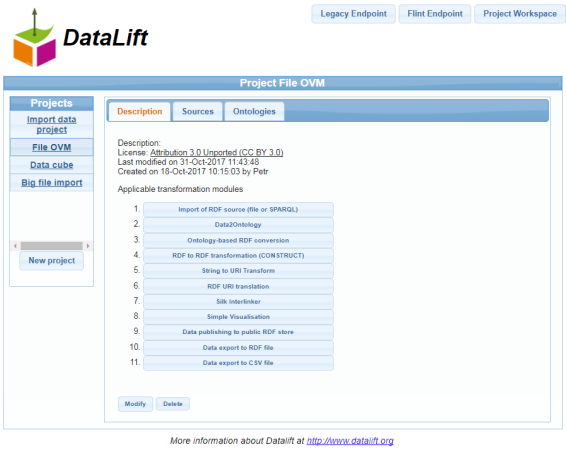

Fig. 10. Datalift user interface

The web-based user interface is simple and user friendly (see Figure 10). Datalift provides out of the box manual visualization with support for several chart types like line chart, table chart, bar chart, timeline, tree map, geographical map, etc.

5.1.10. LODLaundromat

LODLaundromat [19] (LAUN) is an open–source service, with web user interface (see Figure 11), designed to provide access to Linked Open Data. It aims to tackle the problem of data quality by converting the data into a single format, removing duplicates and blank nodes and by fixing syntax errors.

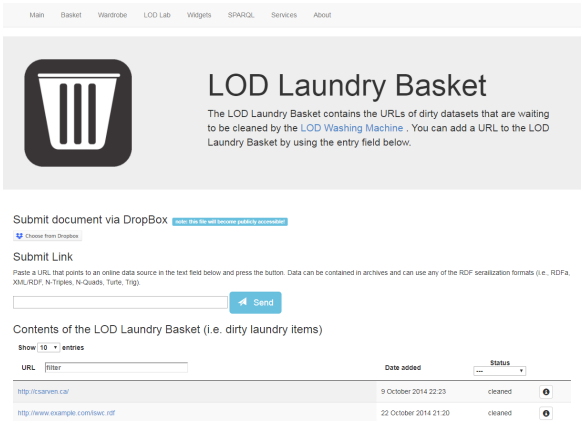

Fig. 11. LOD Laundromat data input user interface

Users can submit data via custom URLs for processing and cleaning. Once the data is clean, it can be downloaded or queried using a linked data fragments interface. LODLaundromat has strong support in the Data input functionality. As data output, it only supports C 27.3 N-Triples file dump.

5.1.11. LD-R

Linked Data Reactor [70] (LDR) is a framework consisting of user interface components for Linked Data applications. It is built to follow the Flux[62] architecture using ReactJS[63]. When installed, LD-R can also be used as a web-based application (see Figure 12). It allows users to list resources, filter them, edit them and visualize them. Most of this functionality, however, requires changing the configuration and by so requires the user to have expert knowledge of LD.

Nevertheless, once set up, most of the functionality is non-LD expert friendly. LD-R mainly covers requirements in the Data input and Data visualization sections.

5.1.12. SemFacet

SemFacet [7] (SF) is an open–source tool implementing semantic facet based search system. The tool has a web-based user interface (see Figure 13) and it is designed solely for the purpose of searching, with focus on faceted search. User can import custom data and an OWL ontology. SemFacet then utilizes the provided ontology to enhance the facet based search.

As the main focus of SemFacet is search, it offers functionality mainly in the area of Data input and it fulfills C 4.2 Faceted search.

[62] https://facebook.github.io/flux/
[63] https://reactjs.org/

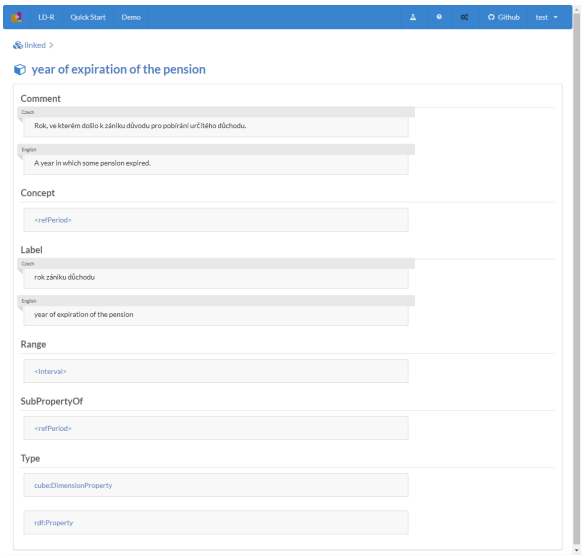

Fig. 12. LD-R user interface

Fig. 13. SemFacet user interface

5.1.13. ESTA-LD

ESTA-LD [146] (ELD) is a web–based application for viewing RDF data cubes with spatial and temporal data. ESTA-LD automatically searches the user provided SPARQL endpoint and provides the user with selection of graphs and datasets found in the endpoint. The user can also select measures, temporal and spatial information that is visualized (see Figure 14).

Most of the ESTA-LD functionality is domain specific, focused on data cubes and spatial and temporal data. From the point of view of an LDCP it supports the user in the area of Data input and it fulfills C 26.3 Vocabulary-based visualization.

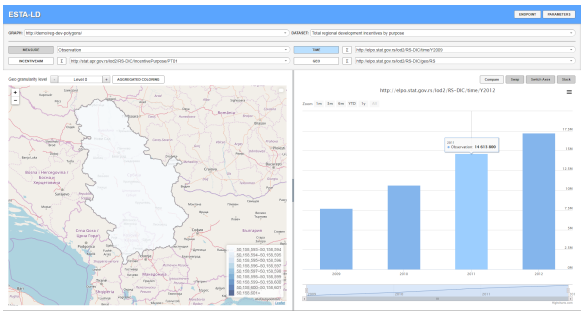

Fig. 14. ESTA-LD user interface

5.1.14. *PepeSearch*

PepeSearch [143] (PEPE) is an open–source tool for LD exploration. It consists of two main components: a tool for preprocessing data in a given SPARQL endpoint and a web–based application (see Figure 15). After preprocessing the data with the tool, the user can add it to the web–based application and explore it. In the exploration phase the user can select classes, properties and relations that are visualized in the form of an HTML table.

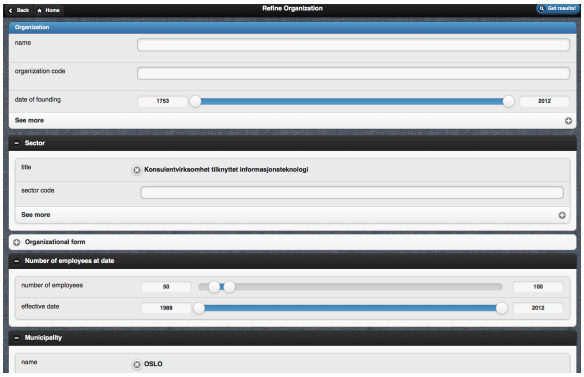

Fig. 15. PepeSearch user interface

The web–application mostly fulfills C 9.2 Schema extraction. The schema is created by the preprocessing tool, that provides the application with the Data input functionality.

5.1.15. *FAGI-gis*

FAGI-gis [58] (FAGI) is an open–source tool for fusion of RDF spatial data supported by a W3C community and business group[64]. The tool can import data from SPARQL endpoints using graphs to distinguish input datasets (see Figure 16). It relies on the user to provide a mapping (linking), according to which the data fusion will be performed. After the data and links are loaded into the tool, the user can manipulate the spatial information of entities on a map and specify the fusion conditions on RDF properties. The resulting fused data can be uploaded to a SPARQL endpoint.

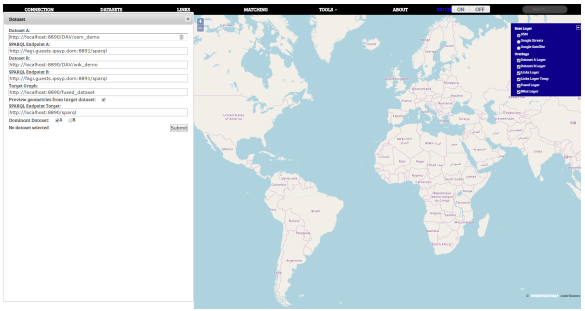

Fig. 16. FAGI-gis user interface

In terms of the identified LDCP requirements, FAGi-gis supports the areas of Data input, Data visualization and Data output, and implements C 20.2 Resource fusion.

5.1.16. *RDFpro*

RDFpro [35] (RDFPRO) is an open–source tool for transformation of LD. The transformation process is described in a script and can be executed using a command line or web-based interface (see Figure 17).

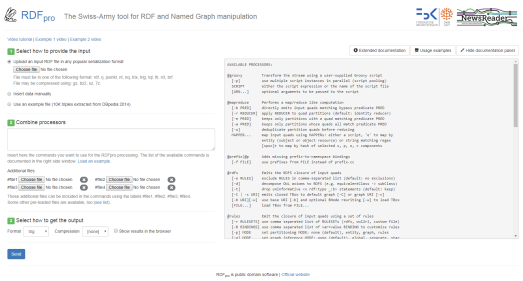

Fig. 17. RDFpro user interface

The main advantage of RDFpro is the performance which can be archived during the LD transformation satisfying C 13.8 Handling of bigger files. RDFpro also supports the user mainly in Data input, Semantic transformations and Data output.

5.2. *Discussion*

As we have said earlier in the introduction, we show in this section which of the evaluated tools are LDCP -

[64] https://www.w3.org/community/geosemweb

Requirement	LP-VIZ	V2	VISU	Sp	YAS	UV	LP-ETL	OC	DL	LAUN	LDR	SF	ELD	PEPE	FAGI	RDFPRO
RDF input using SPARQL	✓✓	✓✓	✓	✓✓	✓	✓	✓	✓	✓	–	✓✓	–	✓✓	✓✓	✓✓	✓
RDF dump load	✓✓	–	–	–	–	✓✓	✓✓	✓✓	✓✓	✓✓	–	✓✓	–	–	✓✓	✓✓
Visualization	✓✓	✓✓	✓	✓✓	✓✓	–	–	✓✓	✓	–	✓✓	–	✓✓	–	✓✓	–
Non-RDF data output	–	–	✓	–	✓	✓	✓	✓	✓✓	–	–	–	–	–	–	–
RDF dump output	–	–	–	–	–	✓✓	✓✓	✓✓	✓✓	✓✓	–	–	–	–	–	✓✓
RDF output using SPARQL	–	–	–	–	–	✓✓	✓✓	–	–	–	–	–	–	–	✓✓	✓✓

Table 5

Requirement fulfillment for LDCP specific requirements. ✓means fulfillment for experts, ✓✓means fulfillment for non-LD experts.

either in the minimal sense specified in the introduction or with some additional features relevant for the consumption process. We introduced three conditions each LDCP must fulfill.

First, an LDCP must support loading LD either from a SPARQL endpoint or a dump file. In terms of our requirements it means that the tool must fulfill the requirement RDF input using SPARQL (i.e. loading LD from a SPARQL endpoint) or the requirement RDF dump load (loading LD from a dump file). We do not consider the other requirements describing possible ways of providing input data, i.e. the requirements IRI dereferencing and Linked Data Platform input, because Table 3 and Table 4 show that none of the evaluated tools fulfills them.

Second, the tool must be able to provide the loaded data on output either in a form of a visualization or represented in some non-RDF data format. In terms of our requirements it means that the tool must fulfill the requirement Visualization (i.e. a visualization of the data) or the requirement Non-RDF data output (i.e. output of data in a non-RDF format).

Third, non-LD experts must be able to use both features without any knowledge of LD technologies. In the introduction, we have explained that this means that the tool provides a user interface which does not require the users to know the RDF model, its serializations or the SPARQL query language.

We are therefore interested in tools which fulfill these three conditions. We denote classify tools as LDCPs. However, the tools which fulfill only the first and the second condition are also important. They support the basic LD consumption activities but only LD experts can use them because the tools assume the knowledge of the RDF data model, its serializations or the SPARQL query language.

Table 5 shows evaluation of the above mentioned requirements important for identifying LDCPs. For each requirement and evaluated tool it shows the maximum result achieved for any of the criteria defined for the requirement. The detailed evaluation of individual criteria is presented in Table 6, Table 7 and Table 8 in Appendix A. The value ✓✓means that at least one criterion defined for the requirement is satisfied by the tool in a way usable for non-LD experts. The value ✓means that at least one criterion defined for the requirement is satisfied by the tool in a way usable for LD experts but there is no criterion satisfied in a way usable for non-LD experts. Therefore, the evaluation presented in Table 5 shows that

- the tools LinkedPipes Visualization, V2, Sparklis, OpenCube, LD-R, ESTA-LD and FAGI-gis are LDCPs because they enable non-LD experts to load LD and visualize them,
- the tool Datalift is an LDCP because it enables non-LD experts to load LD and provides an output in a non-RDF data format, and
- the tools VISU, YASGUI, UnifiedViews and LinkedPipes ETL are not LDCPs but they can be used for LD consumption by LD experts. These tools support the minimal features for LD consumption. However, they provide these features only for LD experts.

Table 3 and Table 4 show that some of these tools also support additional features represented by the other requirements. Table 3 shows which additional requirements are fulfilled to which extent by the tools from the point of view of non-LD experts. Table 4 shows this from the point of view of LD experts.

Table 5 shows that the rest of the evaluated tools, i.e. LODLaundromat, SemFacet, PepeSearch and RDFpro cannot be considered as LDCPs because they do not provide neither a visualization nor a non-RDF representation on the output. However, Table 5 shows that LODLaundromat and RDFpro provide an RDF output in a form of a dump. Moreover, RDFpro provides also its RDF output in a SPARQL endpoint. Therefore, both tools can be combined with the LDCPs mentioned above because their RDF output may serve as input for

the LDCPs. Also the tools UnifiedViews and LinkedPipes ETL, which are classified above as LD consumption tools for LD experts, provide an RDF output and, therefore, can be combined with the LDCPs. And also the LDCPs OpenCube, Datalift and FAGI-gis provide an RDF output so it is possible to combine them with other LDCPs.

Figure 18 summarizes the above discussion. The tools displayed in green boxes were classified as LDCPs. The tools in blue boxes were classified as tools which support LD experts in LD consumption but they are not suitable for non-LD experts and, therefore, they are not LDCPs. The tools in gray boxes provide some additional features which are relevant for LD consumption. As shown in Table 5, they provide some features suitable even for non-LD experts. There are two groups of tools displayed in Figure 18. The tools in the group on the left hand side provide an RDF output which can be used as an RDF input for the tools in the same group or in the group on the right hand side. This is depicted by the connecting arrows. It shows how one tool can extend features of another tool.

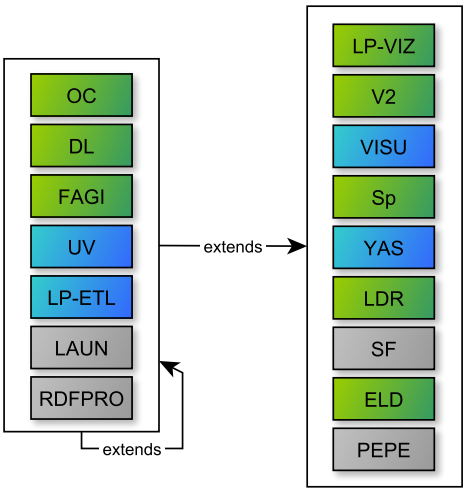

Fig. 18. Classification of evaluated tools: (green) tools classified as LDCPs, (blue) tools supporting LD experts in LD consumption and (gray) tools providing additional LD consumption features. The arrows show that tools in their source provide an RDF output which can be consumed as an input of tools in their target.

In total, we identified 8 LDCPs which is not a small number. Moreover, all evaluated tools have significant added value for LD consumers. In summary, all evaluated tools enable users to load LD and perform some useful actions, at least to preview the loaded LD in case of PepeSearch. There are tools providing automated visualization of published data (LinkedPipes Visualization), manual visualization of published data (V2, VISU), exploration of given data (Sparklis), SPARQL query building capabilities (YASGUI) and then there are tools mainly focused on publishing LD, which can also be used for LD consumption (UnifiedViews, LinkedPipes ETL, OpenCube, Datalift).

However, the evaluation presented in Table 3 and Table 4 also shows that there is a lack of tools covering certain steps of the LD consumption process even for LD experts. In the category of Dataset discovery, there are no tools providing dataset search capabilities built on top of LD principles which is demonstrated by the fact that no tool fulfills requirements Catalog-free discovery, Search query expansion and Search result extension. The existing tools are limited to requiring a link to an RDF dump or a SPARQL endpoint from their users as an input, which could, ideally, be provided by the dataset discovery techniques. OpenCube supports dataset discovery but only on top of CKAN APIs which is a proprietary non-LD API.

The evaluated tools do not inform users about quality of LD as there are no tools fulfilling the requirement Quality indicators. No tools support users in management of provenance information and licensing which can be seen from the evaluation of the requirements Provenance and License management. The tools do not support users in change and version management which is shown by requirements Monitoring of changes in input data and Version management not fulfilled by any of the tools. These parts of the LD consumption process are well covered by existing W3C standards and literature as we have shown in Section 4 but this theory is not reflected by the tools.

In the category of Developer and community support, 8 of the 16 evaluated tools passed 33.4 Community activity, the rest is not actively maintained. However, most of the tools, 13, have their source code in a publicly accessible repository. What could be improved is the presence of well–known repositories for plug-ins and projects, where most of the evaluated tools do not have one, and which could boost community adoption.

If we restrict ourselves only to the non-LD expert view presented in Table 3 we must conclude that the current tools support non-LD experts insufficiently in the LD consumption activities, with only basic search interfaces and visualization capabilities. This enables non-LD experts to consume LD in a limited way, i.e. load given LD and visualize them somehow. However, the tools do not provide the non-LD experts with features which would enable them to exploit the full range

of benefits of LD. They still have the only possibility which is learning LD technologies.

6. Related work

In this section we go over related work in the sense of similar studies of LD consumption possibilities. We structure related work to two parts. First, there are existing surveys of research areas which correspond to some parts of our LD consumption process. We summarize them in Section 6.1. We recommend them to our readers as more complete overviews of existing research literature in these concrete areas. For each survey we show related requirements from Section 4. However, not all areas related to LD consumption are covered by surveys. In this paper we do not aim at doing a complete survey of research literature for these areas. We only describe selected works in these areas or W3C Recommendations to support our introduced requirements.

Second, there are works which are surveys of software tools which can be used for LD consumption. These tools are close to our survey presented in this paper. We summarize them in Section 6.2 and we compare them to ours.

6.1. Complementary surveys

The Survey on Linked Data Exploration Systems [87] provides an overview of 16 existing tools in three categories, classified according to 16 criteria. The first category represents the LD browsers, the second category represents domain-specific and cross-domain recommenders, which should be considered for integration into LDCP regarding e.g. our Req. 6 Search result extension. The third category represents the exploratory search systems (ESS) which could be considered for LDCP regarding Req. 2 Catalog-free discovery. A recent survey on RDF Dataset Profiling [47] surveys methods and tools for LOD dataset profiling. The survey identifies and compares 20 existing profiling tools which could be considered for LDCP regarding Req. 2 Catalog-free discovery. The surveyed systems help to find the data the user is looking for. However, in the scope of LDCP, this is only the beginning of the consumption, which is typically followed by processing of the found data leading to a custom visualization or data in a format for further processing.

In [130] authors introduce a system which provides a visual interface to assist users in defining SPARQL queries. The paper compares 9 existing software tools which provide such assistance to the users. These tools could be integrated into LDCP to cover Req. 3 Search query language and Req. 4 Search query formulation assistance.

A recent survey on Ontology Matching [104] indicates there is a growing interest in the field, which could prove very useful when integrated into LDCP, covering Req. 17 Semantic relationship analysis and Req. 18 Semantic relationship deduction.

A survey on Quality Assessment for Linked Data [149] defines 18 quality dimensions such as availability, consistency and completeness and 69 finer-grained metrics. The authors also analyze 30 core approaches to LD quality and 12 tools, which creates a comprehensive base of quality indicators to be used to cover Req. 10 Quality indicators. One of the quality dimensions deals with licensing, which can be a good base for Req. 25 License management, which is, so far, left uncovered.

A survey on Linked Data Visualization [37] defines 18 criteria and surveys 15 LD browsers, which could be used to cover the discovery Req. 2 Catalog-free discovery and the visualization Req. 26 Visualization.

In Exploring User and System Requirements of Linked Data Visualization through a Visual Dashboard Approach [91] the authors perform a focus group study. They aim for determining the users' needs and system requirements for visualizing LD using the dashboard approach. The authors utilize their Points of View (.views.) framework for various parallel visualizations of LD. They also emphasize the heterogeneity of LD and the need for highly customized visualizations for different kinds of LD, which supports our Req. 8 Preview based on vocabularies, and Req. 26 Visualization.

6.2. Surveys of LD consumption

Close to our presented survey is a recent survey on The Use of Software Tools in Linked Data Publication and Consumption [14]. The authors assume a LOD publication process as well as a LOD consumption process, both taken from existing literature. They do a systematic literature review to identify existing software tools for LOD publication and consumption. They identify 9 software tools for publication and consumption of LOD. Their survey differs from ours in many aspects. First, 9 evaluated tools were published between years 2007 - 2012, and one is from 2014. We evaluate tools from 2013 or newer. Second, the authors evaluated how each tool supports individual steps in both processes. However, evaluation criteria are not clear. The paper

does not present any requirements nor criteria, only steps of the processes. It is not clear how the authors evaluated whether a given tools supports a given step. The only detail present is evaluation of supported RDF serialization formats. Therefore, our study is more detailed and precise. Third, they evaluate the tools only based on the papers where the tools are described. It is not described whether tools were installed and used. In our study, this is one of the selection criteria. Therefore, our work is more useful for the community not only because it is more recent, detailed and precise but also because it informs the readers about the tools which can be actually used and which requirements they really cover in their run-time.

Another close paper is our own study [78] conducted at the end of 2015 which contains a brief survey on the same topic. The novel survey presented in this paper is its significant extension. We describe the LD consumption process and compare it to the existing literature. We revise the evaluation requirements, put them into the context of the LD consumption process and make their evaluation more precise and repeatable by introducing exact evaluation criteria. Both the requirements and the criteria are now supported by existing literature and W3C Recommendations. In [78] the evaluated tools were selected based on a quick literature survey while in this paper we selected the tools based on a systematic, deep and repeatable survey. The evaluation results are presented in a bigger detail. Last but not least, the evaluated tools in this paper are more recent and we recently (Nov 2017) checked their availability.

7. Conclusions

In this paper, we identified the need for a software tool which would provide access to Linked Data to consumers who are experts in data consumption or processing but not experts on LD technologies. We call such software tool a *Linked Data Consumption Platform* (*LDCP*) and the target consumers *non-LD experts*. We motivated our work by a task given to data journalists who need to write an article about population of European cities and who want to attach data in a CSV file to the article. Based on the current literature and W3C standards related to LD consumption we described the LD consumption process, and in turn we defined 34 requirements on tools implementing it, which were further split into 94 evaluation criteria. Then we performed a survey of 110 existing tools, which after 3 elimination rounds ended in 16 tools which we evaluated using all criteria. We evaluated the tools from the point of view of non-LD experts as well as LD experts.

We identified 8 software tools which can be classified as a LDCP, i.e. a tool which enables non-LD experts to consume LD at least in a minimal sense. This minimal sense means that the tool has the ability to load selected LD and get them back in a form of data visualization or a representation in a non-RDF format, e.g., CSV. Moreover, we have shown that the rest of the evaluated tools, even not classified as LDCPs, support either LD experts in LD consumption or both kinds of users in certain parts of the LD consumption process. We have also shown that the tools can be combined together because many of them provide RDF output which can be used as an input by other tools.

On the other hand, we have also shown that there are still significant steps in LD consumption which are covered by the existing tools insufficiently. Basic LD consumption activities are already supported by the existing tools. However, non-LD experts still cannot exploit all benefits of LD principles and technologies. This poses a problem for LD adoption, as consumers often still find consuming 3-star data such as CSV files more comfortable than working with LD. At the same time, the potential LD publishers have a hard time convincing their superiors to invest the additional effort into achieving LD publication, when there are no immediately appreciable benefits of doing so.

Without a proper LDCP, even experts willing to learn LD technologies are having a hard time seeing the benefits in processing RDF files using SPARQL. To them, it is still easier to process CSV files using their own scripts or their favorite tabular data processor.

Our survey may serve as an overview of expected functionality for anyone who would like to build a tool or integrate more existing tools for LD consumption or selected parts of the consumption process. It may also serve as a starting point for anyone interested in what benefits to expect when publishing data as LD.

As to the future research directions, possibilities of integration of individual tools into bigger platforms should be investigated. Our survey shows that on one hand, there is enough research and specifications to make LD at least as usable to non-LD experts as, e.g. CSV files, but on the other hand there is lack of implementation of these features in present tools. These should implement the advanced features exploiting the LD ecosystem while shielding the non-LD experts from the technical details using a suitable user interface.

Acknowledgments

This work was supported in part by the Czech Science Foundation (GAČR), grant number 16-09713S and in part by the project SVV 260451.

References

[1] A. Abele. Linked data profiling: Identifying the domain of datasets based on data content and metadata. In J. Bourdeau, J. Hendler, R. Nkambou, I. Horrocks, and B. Y. Zhao, editors, *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11-15, 2016, Companion Volume*, pages 287–291. ACM, 2016.

[2] K. Alexander, J. Zhao, R. Cyganiak, and M. Hausenblas. Describing Linked Datasets with the VoID Vocabulary. W3C note, W3C, Mar. 2011. http://www.w3.org/TR/2011/NOTE-void-20110303/.

[3] M. Alonen, T. Kauppinen, O. Suominen, and E. Hyvönen. Exploring the Linked University Data with Visualization Tools. In P. Cimiano, M. Fernández, V. López, S. Schlobach, and J. Völker, editors, *The Semantic Web: ESWC 2013 Satellite Events - ESWC 2013 Satellite Events, Montpellier, France, May 26-30, 2013, Revised Selected Papers*, volume 7955 of *Lecture Notes in Computer Science*, pages 204–208. Springer, 2013.

[4] Andreas Steigmiller and Thorsten Liebig and Birte Glimm. Konclude: System description. *Web Semantics: Science, Services and Agents on the World Wide Web*, 27:78 – 85, 2014. Semantic Web Challenge 2013.

[5] P. Arapov, M. Buffa, and A. Ben Othmane. Semantic Mashup with the Online IDE WikiNEXT. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14 Companion, pages 119–122, New York, NY, USA, 2014. ACM.

[6] S. Araújo, J. Hidders, D. Schwabe, and A. P. de Vries. SERIMI - resource description similarity, RDF instance matching and interlinking. In P. Shvaiko, J. Euzenat, T. Heath, C. Quix, M. Mao, and I. F. Cruz, editors, *Proceedings of the 6th International Workshop on Ontology Matching, Bonn, Germany, October 24, 2011*, volume 814 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2011.

[7] M. Arenas, B. Cuenca Grau, E. Kharlamov, S. Marciuska, D. Zheleznyakov, and E. Jimenez-Ruiz. SemFacet: Semantic Faceted Search over Yago. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14 Companion, pages 123–126, New York, NY, USA, 2014. ACM.

[8] N. Arndt, N. Radtke, and M. Martin. Distributed Collaboration on RDF Datasets Using Git: Towards the Quit Store. In *12th International Conference on Semantic Systems Proceedings (SEMANTiCS 2016)*, SEMANTiCS '16, Leipzig, Germany, Sept. 2016.

[9] J. Arwe, S. Speicher, and A. Malhotra. Linked Data Platform 1.0. W3C recommendation, W3C, Feb. 2015. http://www.w3.org/TR/2015/REC-ldp-20150226/.

[10] Assaf, Ahmad and Troncy, Raphaël and Senart, Aline. Roomba: An Extensible Framework to Validate and Build Dataset Profiles. In Gandon, Fabien and Guéret, Christophe and Villata, Serena and Breslin, John and Faron-Zucker, Catherine and Zimmermann, Antoine, editor, *The Semantic Web: ESWC 2015 Satellite Events: ESWC 2015 Satellite Events, Portorož, Slovenia, May 31 – June 4, 2015, Revised Selected Papers*, pages 325–339, Cham, 2015. Springer International Publishing.

[11] J. Attard, F. Orlandi, and S. Auer. ExConQuer Framework - Softening RDF Data to Enhance Linked Data Reuse. In Villata et al. [144].

[12] S. Auer, J. Demter, M. Martin, and J. Lehmann. LODStats - An Extensible Framework for High-Performance Dataset Analytics. In A. ten Teije, J. Völker, S. Handschuh, H. Stuckenschmidt, M. d'Aquin, A. Nikolov, N. Aussenac-Gilles, and N. Hernandez, editors, *Knowledge Engineering and Knowledge Management - 18th International Conference, EKAW 2012, Galway City, Ireland, October 8-12, 2012. Proceedings*, volume 7603 of *Lecture Notes in Computer Science*, pages 353–362. Springer, 2012.

[13] B. Balis, T. Grabiec, and M. Bubak. Domain-Driven Visual Query Formulation over RDF Data Sets. In R. Wyrzykowski, J. J. Dongarra, K. Karczewski, and J. Wasniewski, editors, *Parallel Processing and Applied Mathematics - 10th International Conference, PPAM 2013, Warsaw, Poland, September 8-11, 2013, Revised Selected Papers, Part I*, volume 8384 of *Lecture Notes in Computer Science*, pages 293–301. Springer, 2013.

[14] A. Barbosa, I. I. Bittencourt, S. W. M. Siqueira, R. de Amorim Silva, and I. Calado. The use of software tools in linked data publication and consumption: A systematic literature review. *Int. J. Semantic Web Inf. Syst.*, 13(4):68–88, 2017.

[15] M. Bastian, S. Heymann, and M. Jacomy. Gephi: An Open Source Software for Exploring and Manipulating Networks. 2009.

[16] F. Bauer and M. Kaltenböck. Linked open data: The essentials. *Edition mono/monochrom, Vienna*, 2011.

[17] S. Bechhofer and A. Miles. SKOS Simple Knowledge Organization System Reference. W3C recommendation, W3C, Aug. 2009. http://www.w3.org/TR/2009/REC-skos-reference-20090818/.

[18] Beck, Nicolas and Scheglmann, Stefan and Gottron, Thomas. *LinDA: A Service Infrastructure for Linked Data Analysis and Provision of Data Statistics*, pages 225–230. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[19] W. Beek, L. Rietveld, H. R. Bazoobandi, J. Wielemaker, and S. Schlobach. LOD Laundromat: A Uniform Way of Publishing Other People's Dirty Data. In P. Mika, T. Tudorache, A. Bernstein, C. Welty, C. A. Knoblock, D. Vrandecic, P. T. Groth, N. F. Noy, K. Janowicz, and C. A. Goble, editors, *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*, volume 8796 of *Lecture Notes in Computer Science*, pages 213–228. Springer, 2014.

[20] F. Benedetti, S. Bergamasch, and L. Po. A visual summary for linked open data sources. In *Proceedings of the 2014 International Conference on Posters & Demonstrations Track - Volume 1272*, ISWC-PD'14, pages 173–176, Aachen, Germany, Germany, 2014. CEUR-WS.org.

[21] F. Benedetti, S. Bergamaschi, and L. Po. LODeX: A Tool for Visual Querying Linked Open Data. In Villata et al. [144].

[22] A. Bertails, M. Arenas, E. Prud'hommeaux, and J. Sequeda. A Direct Mapping of Relational Data to RDF. W3C recommendation, W3C, Sept. 2012. http://www.w3.org/TR/2012/REC-

rdb-direct-mapping-20120927/.

[23] N. Bikakis, G. Papastefanatos, M. Skourla, and T. Sellis. A hierarchical aggregation framework for efficient multilevel visual exploration and analysis. *Semantic Web*, 8(1):139–179, 2017.

[24] C. Bizer, S. Auer, T. Berners-Lee, and T. Heath, editors. *Proceedings of the Workshop on Linked Data on the Web, LDOW 2015, co-located with the 24th International World Wide Web Conference (WWW 2015), Florence, Italy, May 19th, 2015*, volume 1409 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2015.

[25] C. Bobed and E. Mena. Querygen: Semantic interpretation of keyword queries over heterogeneous information systems. *Information Sciences*, 329:412 – 433, 2016. Special issue on Discovery Science.

[26] P. Bottoni and M. Ceriani. SWOWS and dynamic queries to build browsing applications on linked data. In *Proceedings: DMS 2014 - 20th International Conference on Distributed Multimedia Systems*, pages 121–127, 2014.

[27] A. Bozzon, P. Cudré-Mauroux, and C. Pautasso, editors. *Web Engineering - 16th International Conference, ICWE 2016, Lugano, Switzerland, June 6-9, 2016. Proceedings*, volume 9671 of *Lecture Notes in Computer Science*. Springer, 2016.

[28] D. Brickley and L. Miller. The Friend Of A Friend (FOAF) Vocabulary Specification, November 2007.

[29] S. Capadisli and A. Guy. Linked Data Notifications. W3C recommendation, W3C, May 2017. https://www.w3.org/TR/2017/REC-ldn-20170502/.

[30] S. Capadisli, A. Guy, C. Lange, S. Auer, A. V. Sambra, and T. Berners-Lee. Linked Data Notifications: A Resource-Centric Communication Protocol. In E. Blomqvist, D. Maynard, A. Gangemi, R. Hoekstra, P. Hitzler, and O. Hartig, editors, *The Semantic Web - 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 - June 1, 2017, Proceedings, Part I*, volume 10249 of *Lecture Notes in Computer Science*, pages 537–553, 2017.

[31] Charalampos Nikolaou and Kallirroi Dogani and Konstantina Bereta and George Garbis and Manos Karpathiotakis and Kostis Kyzirakos and Manolis Koubarakis. Sextant: Visualizing time-evolving linked geospatial data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 35:35 – 52, 2015. Geospatial Semantics.

[32] R. Chawuthai and H. Takeda. RDF Graph Visualization by Interpreting Linked Data as Knowledge. In G. Qi, K. Kozaki, J. Z. Pan, and S. Yu, editors, *Semantic Technology - 5th Joint International Conference, JIST 2015, Yichang, China, November 11-13, 2015, Revised Selected Papers*, volume 9544 of *Lecture Notes in Computer Science*, pages 23–39. Springer, 2015.

[33] K. Christodoulou, N. W. Paton, and A. A. A. Fernandes. Structure inference for linked data sources using clustering. *Trans. Large-Scale Data- and Knowledge-Centered Systems*, 19:1–25, 2015.

[34] D. Collarana, C. Lange, and S. Auer. Fuhsen: A platform for federated, rdf-based hybrid search. In *Proceedings of the 25th International Conference Companion on World Wide Web*, WWW '16 Companion, pages 171–174, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee.

[35] F. Corcoglioniti, A. P. Aprosio, and M. Rospocher. Demonstrating the Power of Streaming and Sorting for Non-distributed RDF Processing: RDFpro. In *Proceedings of the ISWC 2015 Posters & Demonstrations Track co-located with the 14th International Semantic Web Conference (ISWC-2015), Bethlehem, PA, USA, October 11, 2015.*, volume 1486 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2015.

[36] L. Costabello and F. Gandon. Adaptive Presentation of Linked Data on Mobile. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14 Companion, pages 243–244, New York, NY, USA, 2014. ACM.

[37] A. Dadzie and M. Rowe. Approaches to visualising linked data: A survey. *Semantic Web*, 2(2):89–124, 2011.

[38] A.-S. Dadzie and E. Pietriga. Visualisation of linked data–reprise. *Semantic Web*, (Preprint):1–21, 2016.

[39] DCMI Usage Board. DCMI Metadata Terms. DCMI recommendation, Dublin Core Metadata Initiative, December 2006. Published online on December 18th, 2006 at `http://dublincore.org/documents/2006/12/18/dcmi-terms/`.

[40] H. V. de Sompel, M. Nelson, and R. Sanderson. HTTP Framework for Time-Based Access to Resource States – Memento. RFC 7089, RFC Editor, December 2013.

[41] R. Delbru, N. Toupikov, M. Catasta, G. Tummarello, and S. Decker. *Hierarchical Link Analysis for Ranking Web Data*, pages 225–239. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[42] D. Diefenbach, R. Usbeck, K. D. Singh, and P. Maret. A Scalable Approach for Computing Semantic Relatedness Using Semantic Web Data. In *Proceedings of the 6th International Conference on Web Intelligence, Mining and Semantics*, WIMS '16, pages 20:1–20:9, New York, NY, USA, 2016. ACM.

[43] S. Djebali and T. Raimbault. SimplePARQL: A New Approach Using Keywords over SPARQL to Query the Web of Data. In *Proceedings of the 11th International Conference on Semantic Systems*, SEMANTICS '15, pages 188–191, New York, NY, USA, 2015. ACM.

[44] B. Do, P. Wetz, E. Kiesling, P. R. Aryan, T. Trinh, and A. M. Tjoa. StatSpace: A Unified Platform for Statistical Data Exploration. In C. Debruyne, H. Panetto, R. Meersman, T. S. Dillon, eva Kühn, D. O'Sullivan, and C. A. Ardagna, editors, *On the Move to Meaningful Internet Systems: OTM 2016 Conferences - Confederated International Conferences: CoopIS, C&TC, and ODBASE 2016, Rhodes, Greece, October 24-28, 2016, Proceedings*, volume 10033 of *Lecture Notes in Computer Science*, pages 792–809, 2016.

[45] M. Dubey, S. Dasgupta, A. Sharma, K. Höffner, and J. Lehmann. *AskNow: A Framework for Natural Language Query Formalization in SPARQL*, pages 300–316. Springer International Publishing, Cham, 2016.

[46] M. Dudáš, V. Svátek, and J. Mynarz. Dataset Summary Visualization with LODSight. In Gandon et al. [56], pages 36–40.

[47] M. B. Ellefi, Z. Bellahsene, J. Breslin, E. Demidova, S. Dietze, J. Szymanski, and K. Todorov. RDF Dataset Profiling - a Survey of Features, Methods, Vocabularies and Applications. *To appear in The Semantic Web Journal*.

[48] M. B. Ellefi, Z. Bellahsene, S. Dietze, and K. Todorov. Dataset Recommendation for Data Linking: An Intensional Approach. In Sack et al. [123], pages 36–51.

[49] J. Erickson and F. Maali. Data Catalog Vocabulary (DCAT). W3C recommendation, W3C, Jan. 2014. http://www.w3.org/TR/2014/REC-vocab-dcat-20140116/.

[50] I. Ermilov and T. Pellegrini. Data licensing on the cloud: empirical insights and implications for linked data. In *Proceedings of the 11th International Conference on Semantic Systems, SEMANTICS 2015, Vienna, Austria, September 15-17, 2015*, pages 153–156, 2015.

[51] S. Faisal, K. M. Endris, S. Shekarpour, S. Auer, and M. Vidal. Co-evolution of RDF Datasets. In Bozzon et al. [27], pages 225–243.

[52] A. Ferraram, A. Nikolov, and F. Scharffe. Data linking for the semantic web. *Semantic Web: Ontology and Knowledge Base Enabled Tools, Services, and Applications*, 169:326, 2013.

[53] S. Ferré. Sparklis: An expressive query builder for SPARQL endpoints with guidance in natural language. *Semantic Web*, 8(3):405–418, 2017.

[54] V. Fionda, G. Pirrò, and C. Gutierrez. NautiLOD: A Formal Language for the Web of Data Graph. *ACM Trans. Web*, 9(1):5:1–5:43, Jan. 2015.

[55] A. Flores, M. Vidal, and G. Palma. Graphium Chrysalis: Exploiting Graph Database Engines to Analyze RDF Graphs. In Presutti et al. [111], pages 326–331.

[56] F. Gandon, C. Guéret, S. Villata, J. G. Breslin, C. Faron-Zucker, and A. Zimmermann, editors. *The Semantic Web: ESWC 2015 Satellite Events - ESWC 2015 Satellite Events Portorož, Slovenia, May 31 - June 4, 2015, Revised Selected Papers*, volume 9341 of *Lecture Notes in Computer Science*. Springer, 2015.

[57] J. Genestoux and A. Parecki. WebSub. W3C recommendation, W3C, Jan. 2018. https://www.w3.org/TR/2018/REC-websub-20180123/.

[58] G. Giannopoulos, N. Vitsas, N. Karagiannakis, D. Skoutas, and S. Athanasiou. FAGI-gis: A Tool for Fusing Geospatial RDF Data. In Gandon et al. [56], pages 51–57.

[59] G. Governatori, H. Lam, A. Rotolo, S. Villata, G. A. Atemezing, and F. L. Gandon. LIVE: a tool for checking licenses compatibility between vocabularies and data. In *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014.*, pages 77–80, 2014.

[60] P. Haase, M. Schmidt, and A. Schwarte. The information workbench as a self-service platform for linked data applications. In *Proceedings of the Second International Conference on Consuming Linked Data - Volume 782*, COLD'11, pages 119–124, Aachen, Germany, Germany, 2010. CEUR-WS.org.

[61] A. Hasnain, Q. Mehmood, S. S. e Zainab, and A. Hogan. SPORTAL: searching for public SPARQL endpoints. In *Proceedings of the ISWC 2016 Posters & Demonstrations Track co-located with 15th International Semantic Web Conference (ISWC 2016), Kobe, Japan, October 19, 2016.*, 2016.

[62] M. Hepp. GoodRelations: An Ontology for Describing Products and Services Offers on the Web. In *Proceedings of the 16th International Conference on Knowledge Engineering: Practice and Patterns*, EKAW '08, pages 329–346, Berlin, Heidelberg, 2008. Springer-Verlag.

[63] P. Heyvaert, P. Colpaert, R. Verborgh, E. Mannens, and R. V. de Walle. Merging and Enriching DCAT Feeds to Improve Discoverability of Datasets. In Gandon et al. [56], pages 67–71.

[64] F. Ilievski, W. Beek, M. van Erp, L. Rietveld, and S. Schlobach. LOTUS: Linked Open Text UnleaShed. In O. Hartig, J. F. Sequeda, and A. Hogan, editors, *Proceedings of the 6th International Workshop on Consuming Linked Data co-located with 14th International Semantic Web Conference (ISWC 2015), Bethlehem, Pennsylvania, US, October 12th, 2015.*, volume 1426 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2015.

[65] A. Jentzsch, C. Dullweber, P. Troiano, and F. Naumann. Exploring linked data graph structures. In *Proceedings of the ISWC 2015 Posters & Demonstrations Track co-located with the 14th International Semantic Web Conference (ISWC-2015), Bethlehem, PA, USA, October 11, 2015.*, 2015.

[66] V. Jindal, S. Bawa, and S. Batra. A review of ranking approaches for semantic search on web. *Information Processing & Management*, 50(2):416 – 425, 2014.

[67] T. Käfer, J. Umbrich, A. Hogan, and A. Polleres. DyLDO: Towards a Dynamic Linked Data Observatory. In C. Bizer, T. Heath, T. Berners-Lee, and M. Hausenblas, editors, *WWW2012 Workshop on Linked Data on the Web, Lyon, France, 16 April, 2012*, volume 937 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

[68] E. Kalampokis, A. Nikolov, P. Haase, R. Cyganiak, A. Stasiewicz, A. Karamanou, M. Zotou, D. Zeginis, E. Tambouris, and K. A. Tarabanis. Exploiting Linked Data Cubes with OpenCube Toolkit. In *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014.*, pages 137–140, 2014.

[69] M. Kaschesky and L. Selmi. Fusepool R5 Linked Data Framework: Concepts, Methodologies, and Tools for Linked Data. In *Proceedings of the 14th Annual International Conference on Digital Government Research*, dg.o '13, pages 156–165, New York, NY, USA, 2013. ACM.

[70] A. Khalili, A. Loizou, and F. van Harmelen. Adaptive linked data-driven web components: Building flexible and reusable semantic web interfaces - building flexible and reusable semantic web interfaces. In Sack et al. [123], pages 677–692.

[71] J. Klímek and J. Helmich. Vocabulary for Linked Data Visualization Model. In *Proceedings of the Dateso 2015 Annual International Workshop on DAtabases, TExts, Specifications and Objects, Nepřívěc u Sobotky, Jičín, Czech Republic, April 14, 2015.*, pages 28–39, 2015.

[72] J. Klímek, J. Helmich, and M. Nečaský. Payola: Collaborative Linked Data Analysis and Visualization Framework. In *10th Extended Semantic Web Conference (ESWC 2013)*, pages 147–151. Springer, 2013.

[73] J. Klímek, J. Helmich, and M. Nečaský. Use Cases for Linked Data Visualization Model. In Bizer et al. [24].

[74] J. Klímek, J. Helmich, and M. Nečaský. LinkedPipes Visualization: Simple Useful Linked Data Visualization Use Cases. In *The Semantic Web - ESWC 2016 Satellite Events, Heraklion, Crete, Greece, May 29 - June 2, 2016, Revised Selected Papers*, pages 112–117, 2016.

[75] J. Klímek, J. Kučera, M. Nečaský, and D. Chlapek. Publication and usage of official czech pension statistics linked open data. *Journal of Web Semantics*, 48:1 – 21, 2018.

[76] J. Klímek and P. Škoda. Speeding up Publication of Linked Data Using Data Chunking in LinkedPipes ETL. In H. Panetto, C. Debruyne, W. Gaaloul, M. P. Papazoglou, A. Paschke, C. A. Ardagna, and R. Meersman, editors, *On the Move to Meaningful Internet Systems. OTM 2017 Conferences - Confederated International Conferences: CoopIS, C&TC, and ODBASE 2017, Rhodes, Greece, October 23-27, 2017, Proceedings, Part II*, volume 10574 of *Lecture Notes in Computer Science*, pages

144–160. Springer, 2017.

[77] J. Klímek, P. Škoda, and M. Nečaský. LinkedPipes ETL: Evolved Linked Data Preparation. In *The Semantic Web - ESWC 2016 Satellite Events, Heraklion, Crete, Greece, May 29 - June 2, 2016, Revised Selected Papers*, pages 95–100, 2016.

[78] J. Klímek, P. Škoda, and M. Nečaský. Requirements on Linked Data Consumption Platform. In S. Auer, T. Berners-Lee, C. Bizer, and T. Heath, editors, *Proceedings of the Workshop on Linked Data on the Web, LDOW 2016, co-located with 25th International World Wide Web Conference (WWW 2016)*, volume 1593 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2016.

[79] T. Knap, J. Michelfeit, and M. Necaský. Linked Open Data Aggregation: Conflict Resolution and Aggregate Quality. In X. Bai, F. Belli, E. Bertino, C. K. Chang, A. Elçi, C. C. Seceleanu, H. Xie, and M. Zulkernine, editors, *36th Annual IEEE Computer Software and Applications Conference Workshops, COMPSAC 2012, Izmir, Turkey, July 16-20, 2012*, pages 106–111. IEEE Computer Society, 2012.

[80] T. Knap, P. Škoda, J. Klímek, and M. Nečaský. UnifiedViews: Towards ETL Tool for Simple yet Powerfull RDF Data Management. In *Proceedings of the Dateso 2015 Annual International Workshop on DAtabases, TExts, Specifications and Objects, Nepřívěc u Sobotky, Jičín, Czech Republic, April 14, 2015.*, pages 111–120, 2015.

[81] A. Lausch, A. Schmidt, and L. Tischendorf. Data mining and linked open data – new perspectives for data analysis in environmental research. *Ecological Modelling*, 295:5 – 17, 2015. Use of ecological indicators in models.

[82] J. Lee, J.-K. Min, A. Oh, and C.-W. Chung. Effective ranking and search techniques for Web resources considering semantic relationships. *Information Processing & Management*, 50(1):132 – 155, 2014.

[83] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195, 2015.

[84] M. Luggen, A. Gschwend, B. Anrig, and P. Cudré-Mauroux. Uduvudu: a Graph-Aware and Adaptive UI Engine for Linked Data. In Bizer et al. [24].

[85] D. Lämmerhirt, M. Rubinstein, and O. Montiel. The state of open government data in 2017. *Open Knowledge International*, 2017.

[86] E. Mäkelä. Aether - Generating and Viewing Extended VoID Statistical Descriptions of RDF Datasets. In Presutti et al. [111], pages 429–433.

[87] N. Marie and F. L. Gandon. Survey of Linked Data Based Exploration Systems. In D. Thakker, D. Schwabe, K. Kozaki, R. Garcia, C. Dijkshoorn, and R. Mizoguchi, editors, *Proceedings of the 3rd International Workshop on Intelligent Exploration of Semantic Data (IESD 2014) co-located with the 13th International Semantic Web Conference (ISWC 2014), Riva del Garda, Italy, October 20, 2014.*, volume 1279 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2014.

[88] M. Martin, K. Abicht, C. Stadler, A.-C. Ngonga Ngomo, T. Soru, and S. Auer. CubeViz: Exploration and Visualization of Statistical Linked Data. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, pages 219–222, New York, NY, USA, 2015. ACM.

[89] Y. C. Martins, F. F. da Mota, and M. C. Cavalcanti. *DSCrank: A Method for Selection and Ranking of Datasets*, pages 333–344. Springer International Publishing, Cham, 2016.

[90] E. Marx, S. Shekarpour, S. Auer, and A.-C. N. Ngomo. Large-Scale RDF Dataset Slicing. In *Proceedings of the 2013 IEEE Seventh International Conference on Semantic Computing*, ICSC '13, pages 228–235, Washington, DC, USA, 2013. IEEE Computer Society.

[91] S. Mazumdar, D. Petrelli, and F. Ciravegna. Exploring User and System Requirements of Linked Data Visualization Through a Visual Dashboard Approach. *Semantic Web*, 5(3):203–220, July 2014.

[92] J. McKinney and R. Iannella. vCard Ontology - for describing People and Organizations. W3C note, W3C, May 2014. http://www.w3.org/TR/2014/NOTE-vcard-rdf-20140522/.

[93] P. N. Mendes, H. Mühleisen, and C. Bizer. Sieve: Linked Data Quality Assessment and Fusion. In *2nd International Workshop on Linked Web Data Management (LWDM 2012) at the 15th International Conference on Extending Database Technology, EDBT 2012*, page to appear, March 2012.

[94] R. Meymandpour and J. G. Davis. A semantic similarity measure for linked data: An information content-based approach. *Knowledge-Based Systems*, 109:276 – 293, 2016.

[95] A. Micsik, Z. Tóth, and S. Turbucz. LODmilla: Shared Visualization of Linked Open Data. In L. Bolikowski, V. Casarosa, P. Goodale, N. Houssos, P. Manghi, and J. Schirrwagen, editors, *Theory and Practice of Digital Libraries - TPDL 2013 Selected Workshops - LCPD 2013, SUEDL 2013, DataCur 2013, Held in Valletta, Malta, September 22-26, 2013. Revised Selected Papers*, volume 416 of *Communications in Computer and Information Science*, pages 89–100. Springer, 2013.

[96] Mohsen Taheriyan and Craig A. Knoblock and Pedro Szekely and José Luis Ambite. Learning the semantics of structured data sources. *Web Semantics: Science, Services and Agents on the World Wide Web* , 37–38:152 – 169, 2016.

[97] M. Nečaský, J. Klímek, and P. Škoda. Practical use cases for linked open data in egovernment demonstrated on the czech republic. In A. Kő and E. Francesconi, editors, *Electronic Government and the Information Systems Perspective*, pages 80–96, Cham, 2017. Springer International Publishing.

[98] M. Nentwig, M. Hartung, A. N. Ngomo, and E. Rahm. A survey of current Link Discovery frameworks. *Semantic Web*, 8(3):419–436, 2017.

[99] C. B. Neto, K. Müller, M. Brümmer, D. Kontokostas, and S. Hellmann. LODVader: An Interface to LOD Visualization, Analytics and DiscovERy in Real-time. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11-15, 2016, Companion Volume*, pages 163–166, 2016.

[100] K. Nguyen, R. Ichise, and B. Le. Slint: a schema-independent linked data interlinking system. *Ontology Matching*, page 1, 2012.

[101] A. G. Nuzzolese, V. Presutti, A. Gangemi, S. Peroni, and P. Ciancarini. Aemoo: Linked Data exploration based on Knowledge Patterns. *Semantic Web*, 8(1):87–112, 2017.

[102] C. Ogbuji. SPARQL 1.1 Graph Store HTTP Protocol. W3C recommendation, W3C, Mar. 2013. http://www.w3.org/TR/2013/REC-sparql11-http-rdf-update-20130321/.

[103] E. Oren, R. Delbru, M. Catasta, R. Cyganiak, H. Stenzhorn, and G. Tummarello. Sindice.com: a document-oriented lookup

index for open linked data. *IJMSO*, 3(1):37–52, 2008.

[104] L. Otero-Cerdeira, F. J. Rodríguez-Martínez, and A. Gómez-Rodríguez. Ontology matching: A literature review. *Expert Systems with Applications*, 42(2):949 – 971, 2015.

[105] A. Pappas, G. Troullinou, G. Roussakis, H. Kondylakis, and D. Plexousakis. *Exploring Importance Measures for Summarizing RDF/S KBs*, pages 387–403. Springer International Publishing, Cham, 2017.

[106] Pierfrancesco Bellini and Paolo Nesi and Alessandro Venturi. Linked open graph: Browsing multiple SPARQL entry points to build your own LOD views. *Journal of Visual Languages & Computing*, 25(6):703 – 716, 2014. Distributed Multimedia Systems DMS2014 Part I.

[107] G. Pirrò. Explaining and suggesting relatedness in knowledge graphs. In M. Arenas, Ó. Corcho, E. Simperl, M. Strohmaier, M. d'Aquin, K. Srinivas, P. T. Groth, M. Dumontier, J. Heflin, K. Thirunarayan, and S. Staab, editors, *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*, volume 9366 of *Lecture Notes in Computer Science*, pages 622–639. Springer, 2015.

[108] G. Pirrò and A. Cuzzocrea. RECAP: Building Relatedness Explanations on the Web. In *Proceedings of the 25th International Conference Companion on World Wide Web*, WWW '16 Companion, pages 235–238, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee.

[109] A. Polleres, P. Gearon, and A. Passant. SPARQL 1.1 Update. W3C recommendation, W3C, Mar. 2013. http://www.w3.org/TR/2013/REC-sparql11-update-20130321/.

[110] J. Polowinski. Towards RVL: A Declarative Language for Visualizing RDFS/OWL Data. In *Proceedings of the 3rd International Conference on Web Intelligence, Mining and Semantics*, WIMS '13, pages 38:1–38:11, New York, NY, USA, 2013. ACM.

[111] V. Presutti, E. Blomqvist, R. Troncy, H. Sack, I. Papadakis, and A. Tordai, editors. *The Semantic Web: ESWC 2014 Satellite Events - ESWC 2014 Satellite Events, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, volume 8798 of *Lecture Notes in Computer Science*. Springer, 2014.

[112] E. Prud'hommeaux and C. B. Aranda. SPARQL 1.1 federated query. W3C recommendation, W3C, Mar. 2013. http://www.w3.org/TR/2013/REC-sparql11-federated-query-20130321/.

[113] D. Reynolds. The Organization Ontology. W3C recommendation, W3C, Jan. 2014. http://www.w3.org/TR/2014/REC-vocab-org-20140116/.

[114] D. Reynolds and R. Cyganiak. The RDF Data Cube Vocabulary. W3C recommendation, W3C, Jan. 2014. http://www.w3.org/TR/2014/REC-vocab-data-cube-20140116/.

[115] L. Rietveld and R. Hoekstra. The YASGUI family of SPARQL clients. *Semantic Web*, 8(3):373–383, 2017.

[116] P. Ristoski and H. Paulheim. Visual Analysis of Statistical Data on Maps Using Linked Open Data. In Gandon et al. [56], pages 138–143.

[117] M. Röder, A. N. Ngomo, I. Ermilov, and A. Both. Detecting similar linked datasets using topic modelling. In Sack et al. [123], pages 3–19.

[118] M. Röder, A. N. Ngomo, I. Ermilov, and A. Both. Detecting Similar Linked Datasets Using Topic Modelling. In Sack et al. [123], pages 3–19.

[119] D. Roman, N. Nikolov, A. Pultier, D. Sukhobok, B. Elvesæter, A. Berre, X. Ye, M. Dimitrov, A. Simov, M. Zarev, R. Moynihan, B. Roberts, I. Berlocher, S.-H. Kim, T. Lee, A. Smith, and T. H. and. DataGraft: One-Stop-Shop for Open Data Management. *Semantic Web*.

[120] Y. Roussakis, I. Chrysakis, K. Stefanidis, and G. Flouris. D2V: A tool for defining, detecting and visualizing changes on the data web. In Villata et al. [144].

[121] V. Sabol, G. Tschinkel, E. E. Veas, P. H"ofler, B. Mutlu, and M. Granitzer. Discovery and Visual Analysis of Linked Data for Humans. In *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*, pages 309–324, 2014.

[122] M. Sabou, I. Onder, A. M. P. Brasoveanu, and A. Scharl. Towards cross-domain data analytics in tourism: a linked data based approach. *Information Technology & Tourism*, 16(1):71–101, Mar 2016.

[123] H. Sack, E. Blomqvist, M. d'Aquin, C. Ghidini, S. P. Ponzetto, and C. Lange, editors. *The Semantic Web. Latest Advances and New Domains - 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Proceedings*, volume 9678 of *Lecture Notes in Computer Science*. Springer, 2016.

[124] S. Sahoo, T. Lebo, and D. McGuinness. PROV-O: The PROV Ontology. W3C recommendation, W3C, Apr. 2013. http://www.w3.org/TR/2013/REC-prov-o-20130430/.

[125] S. Scheider, A. Degbelo, R. Lemmens, C. van Elzakker, P. Zimmerhof, N. Kostic, J. Jones, and G. Banhatti. Exploratory querying of SPARQL endpoints in space and time. *Semantic Web*, 8(1):65–86, 2017.

[126] K. Schlegel, T. Weißgerber, F. Stegmaier, C. Seifert, M. Granitzer, and H. Kosch. Balloon Synopsis: A Modern Node-Centric RDF Viewer and Browser for the Web. In Presutti et al. [111], pages 249–253.

[127] D. Schweiger, Z. Trajanoski, and S. Pabinger. SPARQLGraph: a web-based platform for graphically querying biological Semantic Web databases. *BMC Bioinformatics*, 15(1):1–5, 2014.

[128] S. Seufert, K. Berberich, S. J. Bedathur, S. K. Kondreddi, P. Ernst, and G. Weikum. ESPRESSO: explaining relationships between entity sets. In S. Mukhopadhyay, C. Zhai, E. Bertino, F. Crestani, J. Mostafa, J. Tang, L. Si, X. Zhou, Y. Chang, Y. Li, and P. Sondhi, editors, *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*, pages 1311–1320. ACM, 2016.

[129] D. Song, F. Schilder, C. Smiley, C. Brew, T. Zielund, H. Bretz, R. Martin, C. Dale, J. Duprey, T. Miller, and J. Harrison. *TR Discover: A Natural Language Interface for Querying and Analyzing Interlinked Datasets*, pages 21–37. Springer International Publishing, Cham, 2015.

[130] A. Soylu, E. Kharlamov, D. Zheleznyakov, E. J. Ruiz, M. Giese, M. G. Skjæveland, D. Hovland, R. Schlatte, S. Brandt, H. Lie, et al. OptiqueVQS: a Visual query system over ontologies for industry. *To appear in The Semantic Web Journal*, 2017.

[131] M. Teng and G. Zhu. Interactive search over web scale rdf data using predicates as constraints. *Journal of Intelligent*

Information Systems, 44(3):381–395, 2015.

[132] J. Tennison and G. Kellogg. Model for Tabular Data and Metadata on the Web. W3C recommendation, W3C, Dec. 2015. http://www.w3.org/TR/2015/REC-tabular-data-model-20151217/.

[133] A. Thalhammer, N. Lasierra, and A. Rettinger. LinkSUM: Using Link Analysis to Summarize Entity Data. In Bozzon et al. [27], pages 244–261.

[134] K. Thellmann, M. Galkin, F. Orlandi, and S. Auer. LinkDaViz – Automatic Binding of Linked Data to Visualizations. In *The Semantic Web - ISWC 2015*, volume 9366 of *Lecture Notes in Computer Science*, pages 147–162. Springer International Publishing, 2015.

[135] I. Traverso-Ribón, G. Palma, A. Flores, and M.-E. Vidal. *Considering Semantics on the Discovery of Relations in Knowledge Graphs*, pages 666–680. Springer International Publishing, Cham, 2016.

[136] T. Trinh, B. Do, P. Wetz, A. Anjomshoaa, E. Kiesling, and A. M. Tjoa. Open Mashup Platform - A Smart Data Exploration Environment. In *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014.*, pages 53–56, 2014.

[137] G. Troullinou, H. Kondylakis, E. Daskalaki, and D. Plexousakis. *RDF Digest: Efficient Summarization of RDF/S KBs*, pages 119–134. Springer International Publishing, Cham, 2015.

[138] Y. Tzitzikas, N. Manolis, and P. Papadakos. Faceted exploration of RDF/S datasets: a survey. *Journal of Intelligent Information Systems*, pages 1–36, 2016.

[139] C. Unger, L. Bühmann, J. Lehmann, A.-C. Ngonga Ngomo, D. Gerber, and P. Cimiano. Template-based question answering over rdf data. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 639–648, New York, NY, USA, 2012. ACM.

[140] Valeria Fionda and Claudio Gutierrez and Giuseppe Pirrò. The swget portal: Navigating and acting on the web of linked data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 26:29 – 35, 2014. .

[141] P. Vandenbussche and B. Vatant. Linked Open Vocabularies. *ERCIM News*, 2014(96), 2014.

[142] P.-Y. Vandenbussche, J. Umbrich, L. Matteis, A. Hogan, and C. Buil-Aranda. SPARQLES: Monitoring public SPARQL endpoints. *Semantic Web*, 8(6):1049–1065, 2017.

[143] G. Vega-Gorgojo, L. Slaughter, M. Giese, S. Heggestøyl, J. W. Klüwer, and A. Waaler. PepeSearch: Easy to Use and Easy to Install Semantic Data Search. In H. Sack, G. Rizzo, N. Steinmetz, D. Mladenic, S. Auer, and C. Lange, editors, *The Semantic Web - ESWC 2016 Satellite Events, Heraklion, Crete, Greece, May 29 - June 2, 2016, Revised Selected Papers*, volume 9989 of *Lecture Notes in Computer Science*, pages 146–150, 2016.

[144] S. Villata, J. Z. Pan, and M. Dragoni, editors. *Proceedings of the ISWC 2015 Posters & Demonstrations Track co-located with the 14th International Semantic Web Conference (ISWC-2015), Bethlehem, PA, USA, October 11, 2015*, volume 1486 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2015.

[145] J. Volz, C. Bizer, M. Gaedke, and G. Kobilarov. Discovering and maintaining links on the web of data. In A. Bernstein, D. Karger, T. Heath, L. Feigenbaum, D. Maynard, E. Motta, and K. Thirunarayan, editors, *The Semantic Web - ISWC 2009*, volume 5823 of *Lecture Notes in Computer Science*, pages 650–665. Springer Berlin Heidelberg, 2009.

[146] Vuk Mijović and Valentina Janev and Dejan Paunović and Sanja Vraneš. Exploratory spatio-temporal analysis of linked statistical data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 41:1 – 8, 2016.

[147] A. Wagner, P. Haase, A. Rettinger, and H. Lamm. Discovering Related Data Sources in Data-Portals. In *Proceedings of the 1st International Workshop on Semantic Statistics (SemStats)*, number 1549 in CEUR Workshop Proceedings, Aachen, 2013.

[148] S. Yumusak, E. Dogdu, H. Kodaz, A. Kamilaris, and P.-Y. Vandenbussche. SpEnD: Linked data SPARQL endpoints discovery using search engines. *IEICE Transactions on Information and Systems*, E100D(4):758–767, 2017.

[149] A. Zaveri, A. Rula, A. Maurino, R. Pietrobon, J. Lehmann, and S. Auer. Quality assessment for Linked Data: A Survey. *Semantic Web*, 7(1):63–93, 2016.

[150] G. Zhu and C. A. Iglesias. Computing Semantic Similarity of Concepts in Knowledge Graphs. *IEEE Transactions on Knowledge and Data Engineering*, PP(99):1–1, 2016.

[151] A. J. Zitzelberger, D. W. Embley, S. W. Liddle, and D. T. Scott. Hykss: Hybrid keyword and semantic search. *Journal on Data Semantics*, 4(4):213–229, 2015.

Appendix

A. Tools evaluated using all criteria

Tool/Criterion	LP-VIZ	V2	VISU	Sp	YAS	UV	LP-ETL	OC	DL	LAUN	LDR	SF	ELD	PEPE	FAGI	RDFPRO
1.1 CKAN API	–	–	–	–	–	–	–	✓✓	–	–	–	–	–	–	–	–
1.2 RDF metadata in dump	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
1.3 Metadata in SPARQL endpoint	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
1.4 Metadata from IRI dereferencing	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
2.1 Dataset profiling	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
3.1 Structured search form	–	–	–	✓✓	–	–	–	✓✓	–	–	–	✓✓	–	–	–	–
3.2 Fulltext search	–	–	–	–	–	–	–	✓✓	–	–	–	✓✓	–	–	–	–
3.3 Natural language based intent expression	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
3.4 Formal language based intent expression	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
4.1 Seach query automated suggestions	–	–	–	✓✓	✓✓	–	–	✓	–	–	–	–	–	–	–	–
4.2 Faceted search	–	–	–	–	–	–	–	–	–	–	–	✓✓	–	–	–	–
5.1 Search query expansion	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
6.1 Search result extension	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
7.1 Content-based dataset ranking	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
7.2 Context-based dataset ranking	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
7.3 Intent-based dataset ranking	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
8.1 Preview – W3C vocabularies	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
8.2 Preview – LOV vocabularies	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
8.3 Preview metadata	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
9.1 Preview data	–	–	✓	–	–	–	–	✓	–	✓	–	–	–	–	–	✓
9.2 Schema extraction	–	✓	–	✓	–	–	–	✓	–	–	–	–	–	✓	–	–
10.1 Quality indicators	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
11.1 IRI dereferencing	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
11.2 Crawling	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
12.1 SPARQL: querying	✓✓	✓✓	✓	✓✓	–	–	–	–	–	–	✓✓	–	✓✓	✓✓	✓✓	–
12.2 SPARQL: named graphs	✓	–	–	–	–	✓	✓	–	✓	–	–	–	✓	–	✓	–
12.3 SPARQL: custom query	–	–	✓	–	✓	✓	✓	✓	✓	–	✓	–	–	–	–	✓
12.4 SPARQL: authentication	–	–	–	–	–	✓	✓	✓	✓	–	–	–	–	–	–	–
12.5 SPARQL: advanced download	–	–	–	–	–	✓	✓	–	–	–	–	–	–	–	–	–

Table 6

Tools evaluated using all criteria - part 1 - Req. 1 Catalog support – Req. 12 RDF input using SPARQL

Tool/Criterion	LP-VIZ	V2	VISU	Sp	YAS	UV	LP-ETL	OC	DL	LAUN	LDR	SF	ELD	PEPE	FAGI	RDFPRO
13.1 RDF/XML file load	–	–	–	–	–	✓✓	✓✓	✓✓	✓✓	✓✓	–	–	–	–	–	✓✓
13.2 Turtle file load	✓✓	–	–	–	–	✓✓	✓✓	✓✓	✓✓	✓✓	–	✓✓	–	–	–	✓✓
13.3 N-Triples file load	–	–	–	–	–	✓✓	✓✓	✓✓	✓✓	✓✓	–	✓✓	–	–	✓✓	✓✓
13.4 JSON-LD file load	–	–	–	–	–	✓✓	✓✓	–	✓✓	–	–	–	–	–	–	✓✓
13.5 RDFa file load	–	–	–	–	–	✓✓	✓✓	✓✓	✓✓	✓✓	–	–	–	–	–	–
13.6 N-Quads file load	–	–	–	–	–	✓✓	✓✓	✓✓	✓✓	✓✓	–	✓✓	–	–	–	✓✓
13.7 TriG file load	–	–	–	–	–	✓✓	✓✓	✓✓	✓✓	✓✓	–	✓✓	–	–	–	✓✓
13.8 Handling of bigger files	–	–	–	–	–	–	✓	–	–	–	–	–	–	–	–	✓✓
13.9 CSV on the Web load	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
13.10 Direct mapping load	–	–	–	–	–	✓✓	–	–	✓✓	–	–	–	–	–	–	–
13.11 R2RML load	–	–	–	–	–	–	–	✓	–	–	–	–	–	–	–	–
14.1 Linked Data Platform containers	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
15.1 Monitoring of input changes: polling	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
15.2 Monitoring of input changes: subscription	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
16.1 Manual versioning support	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
16.2 Automated versioning support	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
16.3 Source versioning support	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
17.1 Shared resources analysis	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
17.2 Link analysis	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
17.3 Shared vocabularies analysis	–	✓	–	–	–	–	–	–	–	–	–	–	–	–	–	–
18.1 Link discovery	–	–	–	–	–	–	–	✓	✓	–	–	–	–	–	–	–
18.2 Ontology alignment	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
19.1 Vocabulary mapping based transformations	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
19.2 Vocabulary-based SPARQL transformations	–	–	–	–	–	–	–	–	✓	–	–	–	–	–	–	–
20.1 RDFS inference	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	✓
20.2 Resource fusion	–	–	–	–	–	–	–	–	–	–	–	–	–	–	✓	–
20.3 OWL inference	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	✓
21.1 Assisted selection	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
21.2 Assisted projection	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
22.1 Custom transformations	–	–	–	–	–	✓	✓	✓	✓	–	–	–	–	–	–	✓
23.1 Automated data manipulation	✓✓	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
24.1 Provenance input	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
24.2 Provenance output	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
25.1 License awareness	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
25.2 License management	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Table 7

Tools evaluated using all criteria - part 2 - Req. 13 RDF dump load – Req. 25 License management

Tool/Criterion	LP-VIZ	V2	VISU	Sp	YAS	UV	LP-ETL	OC	DL	LAUN	LDR	SF	ELD	PEPE	FAGI	RDFPRO
26.1 Graph visualization	–	✓✓	–	–	–	–	–	–	–	–	✓✓	–	–	–	–	–
26.2 Manual visualization	–	–	✓	✓✓	✓✓	–	–	✓✓	✓	–	–	–	–	–	–	–
26.3 Vocabulary-based visualization	✓✓	–	–	✓✓	✓✓	–	–	–	–	–	✓✓	–	✓✓	–	✓✓	–
27.1 RDF/XML file dump	–	–	–	–	–	✓✓	✓✓	✓✓	✓✓	–	–	–	–	–	–	✓✓
27.2 Turtle file dump	–	–	–	–	–	✓✓	✓✓	✓✓	✓✓	–	–	–	–	–	–	✓✓
27.3 N-Triples file dump	–	–	–	–	–	✓✓	✓✓	✓✓	✓✓	✓✓	–	–	–	–	–	✓✓
27.4 JSON-LD file dump	–	–	–	–	–	✓✓	✓✓	✓✓	–	–	–	–	–	–	–	✓✓
27.5 RDFa file dump	–	–	–	–	–	✓✓	✓✓	✓✓	–	–	–	–	–	–	–	–
27.6 N-Quads file dump	–	–	–	–	–	✓✓	✓✓	✓✓	✓✓	–	–	–	–	–	–	✓✓
27.7 TriG file dump	–	–	–	–	–	✓✓	✓✓	✓✓	✓✓	–	–	–	–	–	–	✓✓
28.1 SPARQL Update support	–	–	–	–	–	✓✓	✓✓	–	–	–	–	–	–	–	✓✓	✓✓
28.2 SPARQL: named graphs support	–	–	–	–	–	✓	✓	–	–	–	–	–	–	–	✓	–
28.3 SPARQL: authentication support	–	–	–	–	–	✓	✓	–	–	–	–	–	–	–	–	–
28.4 SPARQL Graph Store HTTP Protocol output	–	–	–	–	–	✓	✓	–	–	–	–	–	–	–	–	–
29.1 Linked Data Platform Containers Output	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
30.1 CSV file output	–	–	✓	–	✓	✓	✓	✓	✓✓	–	–	–	–	–	–	–
30.2 CSV on the Web output	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
30.3 XML file output	–	–	–	–	–	–	✓	–	–	–	–	–	–	–	–	–
30.4 JSON file output	–	–	–	–	–	–	✓	–	–	–	–	–	–	–	–	–
30.5 Output for specific third-party tools	–	–	✓	–	–	–	–	–	–	–	–	–	–	–	–	–
31.1 Existence of API	–	–	–	–	–	✓✓	✓	–	–	–	–	–	–	–	–	–
31.2 API documentation	–	–	–	–	–	✓✓	✓✓	–	–	–	–	–	–	–	–	–
31.3 Full API coverage	–	–	–	–	–	–	✓	–	–	–	–	–	–	–	–	–
32.1 Configuration in open format	–	–	–	–	–	✓✓	✓	–	–	–	–	–	–	–	–	✓
32.2 Configuration in RDF	–	–	–	–	–	–	✓	–	–	–	–	–	–	–	–	–
33.1 Repository for source code	✓✓	–	–	–	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓	✓✓
33.2 Repository for plugins	–	–	–	–	–	✓✓	✓✓	–	–	–	–	–	–	–	–	–
33.3 Repository for data processes	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
33.4 Community activity	✓✓	–	–	–	✓✓	✓✓	✓✓	–	✓✓	–	–	✓✓	–	–	✓✓	✓✓
34.1 Deployment of services	–	–	–	–	–	–	✓✓	–	–	–	–	–	–	–	–	–
Total for LD experts	**7**	**4**	**6**	**6**	**7**	**31**	**36**	**27**	**23**	**9**	**5**	**9**	**4**	**3**	**9**	**22**
Total for non-LD experts	**6**	**2**	**0**	**5**	**5**	**22**	**20**	**18**	**16**	**8**	**4**	**9**	**3**	**2**	**6**	**16**

Table 8

Tools evaluated using all criteria - part 3 - Req. 26 Visualization – Req. 34 Deployment of services

B. Elimination rounds for potential candidates

- *U* in round 3 means that loading of the test RDF data into the tool is not supported by the tool, e.g. the tool has a fixed set of datasets to work with
- *E* in round 3 means that there was an error loading the test RDF data into the tool

Tools/Round	2	3
[101] Aemoo [65]	✓	U
[86] Aether [66]	✓	E
[126] CODE Balloon [67]	✓	U
[36] Prissma Browser [68]	✗	
[88] CubeViz [69]	✓	E
[120] D2V [70]	✗	
[119] DataGraft [71]	✗	
[52] Datalift [72]	✓	✓
[63] DCAT merger [73]	✓	E
[121] CODE DoSeR [74]	✗	
[121] CODE Enrichment Service and Annotator Tool [75]	✗	
[146] ESTA-LD [76]	✓	✓
[11] ExConQuer	✗	
[58] FAGI-gis [77]	✓	✓
[34] FuhSen [78]	✓	U
[69] Fusepool [79]	✓	E
[55] Graphium [80]	✗	
[96] Karma [81]	✗	
[4] Konclude [82]	✗	
[70] LD-R [83]	✓	✓

Table 9
All evaluated tools which passed elimination round 1 - part 1

Tools/Round	2	3
[73] LDVMi [84]	✗	
[18] LinDA [85]	✓	E
[134] LinDA LinkDaViz [86]	✗	
[136] LinkedWidgets [87]	✓	U
[133] LinkSUM [88]	✓	U
[21] LODeX [89]	✗	
[19] LODLaundromat [90]	✓	✓
[95] LODmilla [91]	✗	
[46] LODSight [92]	✓	U
[99] LODVader [93]	✗	
[106] LOG (linked-open-graph) [94]	✓	E
[64] LOTUS [95]	✓	E
[77] LP-ETL [96]	✓	✓
[74] LP-VIZ [97]	✓	✓
[68] OpenCube [98]	✓	✓
[72] Payola [99]	✗	
[143] PepeSearch [100]	✓	✓
[65] ProLOD++ [101]	✗	
[13] QUaTRO [102]	✗	
[121] CODE Query Wizard [103]	✗	
[23] rdf:SynopsViz [104]	✓	E
[32] RDF4U [105]	✓	U
[35] RDFpro [106]	✓	✓

Table 10
All evaluated tools which passed elimination round 1 - part 2

[65] http://wit.istc.cnr.it/aemoo/
[66] http://demo.seco.tkk.fi/aether/
[67] http://schlegel.github.io/balloon/index.html
[68] http://wimmics.inria.fr/projects/prissma/
[69] http://cubeviz.aksw.org/
[70] http://www.ics.forth.gr/isl/D2VSystem
[71] https://github.com/datagraft
[72] http://datalift.org/
[73] https://github.com/pheyvaer/dcat-merger
[74] https://code.know-center.tugraz.at/search
[75] https://code.know-center.tugraz.at/search
[76] https://github.com/GeoKnow/ESTA-LD
[77] https://github.com/SLIPO-EU/FAGI-gis
[78] https://github.com/LiDaKrA/FuhSen
[79] https://github.com/fusepoolP3
[80] https://github.com/afloresv/Graphium
[81] http://usc-isi-i2.github.io/karma/
[82] http://www.derivo.de/en/produkte/konclude.html
[83] http://ld-r.org/
[84] https://github.com/ldvm/LDVMi
[85] http://linda.west.uni-koblenz.de/
[86] http://eis.iai.uni-bonn.de/Projects/LinkDaViz.html
[87] http://linkedwidgets.org/
[88] http://km.aifb.kit.edu/services/link/
[89] http://dbgroup.unimo.it/lodex
[90] https://github.com/LOD-Laundromat
[91] http://lodmilla.sztaki.hu/lodmilla/
[92] http://lod2-dev.vse.cz/lodsight-v2
[93] https://github.com/AKSW/LODVader/
[94] http://log.disit.org/service/
[95] https://github.com/filipdbrsk/LOTUS_Search
[96] https://github.com/linkedpipes/etl
[97] https://github.com/ldvm/LDVMi
[98] http://opencube-toolkit.eu/
[99] https://github.com/payola/Payola
[100] https://github.com/guiveg/pepesearch
[101] https://www.hpi.uni-potsdam.de/naumann/sites/prolod++
[102] http://dice.cyfronet.pl/products/quatro
[103] https://code.know-center.tugraz.at/search
[104] http://synopsviz.imis.athena-innovation.gr/
[105] http://rc.lodac.nii.ac.jp/rdf4u/
[106] http://rdfpro.fbk.eu/
[107] https://github.com/ahmadassaf/OpenData-Checker

Tools/Round	2	3
[10] Roomba [107]	✗	
[110] RVL [108]	✓	U
[7] SemFacet [109]	✓	✓
[31] Sextant [110]	✓	U
[53] Sparklis [111]	✓	✓
[148] SpEnD [112]	✓	E
[125] SPEX [113]	✓	E
[61] SPORTAL [114]	✓	E
[44] StatSpace [115]	✓	U
[36] Prissma Studio [116]	✓	U
[140] SWGET [117]	✗	
[26] SWOWS [118]	✗	
[118] Tapioca [119]	✗	
[84] Uduvudu [120]	✓	U
[80] UnifiedViews [121]	✓	✓
[3] V2 [122]	✓	✓
[116] ViCoMap [123]	✓	E
[3] VISU [124]	✓	✓
[121] CODE Visualisation Wizard [125]	✗	
[5] WikiNext [126]	✗	
[18] LinDA Workbench [127]	✗	
[115] YASGUI [128]	✓	✓

Table 11
All evaluated tools which passed elimination round 1 - part 3

[108] https://github.com/janpolowinski/rvl
[109] https://github.com/semfacet
[110] http://sextant.di.uoa.gr/
[111] http://www.irisa.fr/LIS/ferre/sparklis/
[112] https://github.com/semihyumusak/SpEnD
[113] https://github.com/lodum/SPEX
[114] https://github.com/SAliHasnain/sparqlautodescription
[115] http://statspace.linkedwidgets.org/
[116] http://wimmics.inria.fr/projects/prissma/
[117] https://swget.wordpress.com/
[118] http://www.swows.org
[119] http://aksw.org/Projects/Tapioca.html
[120] https://github.com/uduvudu/uduvudu/
[121] https://github.com/unifiedviews
[122] http://data.aalto.fi/V2
[123] http://vicomap.informatik.uni-mannheim.de/
[124] http://data.aalto.fi/visu
[125] https://code.know-center.tugraz.at/search
[126] https://github.com/pavel-arapov/wikinext
[127] http://linda.west.uni-koblenz.de/
[128] https://github.com/OpenTriply/YASGUI

C. Criteria evaluation scenarios

In this section we describe in more detail the evaluation scenarios for our 94 criteria for tool evaluation. Each scenario is described by one or more *PASS* conditions and optional *Prerequisites* and examples of passing or failing the conditions. It is enough for the tool to pass one *PASS* condition to pass the criterion.

Criterion 1.1 CKAN API
Pass The tool can load metadata from a CKAN catalog using its API.
Pass example Load metadata from CKAN catalog with URL `https://linked.opendata.cz` using its API.

Criterion 1.2 RDF metadata in dump
Pass The tool is able to load VoID, DCAT, DCAT-AP or Schema.org dataset metadata from a file or a URL.
Pass example The tool loads DCAT-AP metadata from a file or a URL, e.g. `https://nkod.opendata.cz/soubor/nkod.trig`, and lets the user see the datasets in the tool.

Criterion 1.3 Metadata in SPARQL endpoint
Pass The tool is able to load VoID, DCAT, DCAT-AP or Schema.org dataset metadata from a SPARQL endpoint.
Pass example The tool can import DCAT-AP metadata from `https://www.europeandataportal.eu/sparql` and show the user a list of datasets, which the user can use in the tool.

Criterion 1.4 Metadata from IRI dereferencing
Pass The tool is able to load VoID, DCAT, DCAT-AP or Schema.org dataset metadata by dereferencing an IRI of a metadata resource.
Pass example The tool can dereference IRI of `dcat:Dataset` or `void:Dataset` resource, e.g. `https://nkod.opendata.cz/zdroj/datová-sada/247025777`, show the user information about the dataset and allow the user to use the dataset.

Criterion 2.1 Dataset profiling
Pass The tool is able to discover datasets on the web and compute a profile of any kind. The tool informs the user about the discovered datasets.
Pass example Given a URL of an RDF resource the tool can identify related resources and enables the user to use all of them as a dataset.
Fail example The tool can only load datasets from an external catalog (not necessarily CKAN or DKAN).

Criterion 3.1 Structured search form
Prerequisite The tool offers search functionality.
Pass The tool provides a structured search form to enable the user to restrict metadata values of the searched datasets. The tool presents the result of the search operation to the user.

Criterion 3.2 Fulltext search
Pass The tool provides an input field where the user can specify a full-text query to search for datasets. The tool presents the result of the search operation to the user.

Criterion 3.3 Natural language based intent expression
Pass The tool provides an input field where the user can specify a query in a natural language to search for datasets. The tool presents the result of the search operation to the user.

Criterion 3.4 Formal language based intent expression
Pass The tool provides an input field where the user can specify a query in a formal language to search for datasets. The tool presents the result of the search operation to the user.

Criterion 4.1 Seach query automated suggestions
Prerequisite 3.1 Structured search form or 3.2 Fulltext search or 3.3 Natural language based intent expression or 3.4 Formal language based intent expression
Pass When the user types a search query, the tool automatically suggests some parts of the search query.
Pass example The tool can automatically suggest predicates or classes based on prefixes when the user types a search query expressed in a formal query language.
Pass example The tool can automatically suggest parts of sentences when the user types a search query expressed in a natural language.

Criterion 4.2 Faceted search
Prerequisite The tool allows users to search for datasets.
Pass The tool shows filters (facets) the user can use to (re)formulate the search query.
Pass example The user can specify time/spatial coverage of a dataset.

Pass example The tool provides the user with a list of all values for a certain predicate and lets the user to select the required values.

Fail example The tool provides only a visualisation with filters that can be used to restrict the values being visualized, outside of the context of datasets search.

Criterion 5.1 Search query expansion

Prerequisite 3.1 Structured search form or 3.2 Full-text search or 3.3 Natural language based intent expression or 3.4 Formal language based intent expression

Pass The tool expands the search query provided by the user with additional resources (e.g., concepts, properties, classes).

Pass example The tool provides a check-box which enables the user to switch on the query expansion.

Pass example Before or after evaluating the search query, the tool shows the expanded version of the query and it informs the user that it is an expanded version of the original query provided by the user.

Criterion 6.1 Search result extension

Prerequisite 3.1 Structured search form or 3.2 Full-text search or 3.3 Natural language based intent expression or 3.4 Formal language based intent expression

Pass The tool extends the list of datasets which is the result of evaluating the given search query with additional semantically related datasets and it informs the user about the extension.

Pass example After the result of evaluation of the given search query is displayed by the tool, the user can click on a button to see additional semantically related results.

Pass example The tool displays a list of additional semantically related datasets below the list of datasets discovered by evaluating search query. Both lists are visually separated from each other in some way.

Fail example The tool only implements a "show more" button that shows additional items (pagination) from the result set.

Criterion 7.1 Content-based dataset ranking

Prerequisite The tool allows users to search for datasets.

Pass The tool orders the list of discovered datasets by a metric based on their metadata (e.g., publisher, update periodicity, size) or content (e.g., different content quality metrics).

Criterion 7.2 Context-based dataset ranking

Prerequisite The tool allows users to search for datasets.

Pass The tool orders the list of discovered datasets by a metric based on their context which is formed by linked datasets.

Criterion 7.3 Intent-based dataset ranking

Prerequisite The tool allows users to search for datasets.

Pass The tool orders the list of discovered datasets by a metric based on the structural and/or semantic similarity or graph distance of their content or context (i.e. linked datasets) with resources in the search query.

Criterion 8.1 Preview – W3C vocabularies

Prerequisite The tool allows users to search for datasets.

Pass The tool offers the user additional information in the search result for each discovered dataset which is a visual or textual preview of the dataset. The preview is extracted from a part of the content of the dataset which is represented according to the specification of well-known W3C vocabularies

Pass example For a dataset containing a SKOS hierarchy, the tool can show information about the SKOS hierarchy.

Criterion 8.2 Preview – LOV vocabularies

Prerequisite The tool allows users to search for datasets.

Pass The tool offers the user additional information in the search result for each discovered dataset which is a visual or textual preview of the dataset. The preview is based on vocabularies listed in LOV, similarly to the previous requirement.

Pass example For a dataset containing Schema.org Geocoordinates, the user can click on a button and see an image on a map.

Criterion 8.3 Preview metadata

Prerequisite The tool allows users to search for datasets.

Pass In the query results view the user can see additional information for each dataset created based on the dataset metadata.

Criterion 9.1 Preview data

Pass The tool can compute statistics (number of triples, etc..) or description of the dataset.

Criterion 9.2 Schema extraction

Pass The tool can extract and visualize a schema used in a dataset.
Pass example The tool can visualise schema extracted from the data.

Criterion 10.1 Quality indicators
Pass The tool is able to provide user with any quality indicators.
Pass example The tool shows that a dataset is available in 80% of time.

Criterion 11.1 IRI dereferencing
Pass The tool can request content from given IRI using HTTP content negotiation.
Pass example The tool can download representation of Prague from `http://dbpedia.org/resource/Prague`.
Fail example The tool can only download an RDF file from a server using simple HTTP GET.

Criterion 11.2 Crawling
Prerequisite 11.1 IRI dereferencing
Pass Given an IRI of an RDF resource the tool can download not only the given resource but also related resources.
Pass example Given the resource `http://dbpedia.org/resource/Prague` the tool can find and download additional information about referenced resources.

Criterion 12.1 SPARQL: querying
Pass Given a SPARQL endpoint the tool can query it using predefined queries and process the results.

Criterion 12.2 SPARQL: named graphs
Pass The tool enables the user to specify named graph present in the queried SPARQL endpoint by means other that specifying it in the user provided query.
Pass example The tool allows the user to select or provide a named graph that is used to evaluate the query.

Criterion 12.3 SPARQL: custom query
Pass User can provide custom SPARQL queries.

Criterion 12.4 SPARQL: authentication
Pass User can provide credentials that are used for authentication/authorization on SPARQL endpoint.

Criterion 12.5 SPARQL: advanced download
Pass The tool explicitly offers support for dealing with large SPARQL results set that can not be downloaded in a regular way due to endpoint limitations.
Pass example The tool supports pagination.
Pass example The tool supports the Virtuoso scrollable cursor technique.

Criterion 13.1 RDF/XML file load
Pass The tool can load RDF data from a URL or a local file in the RDF/XML format.

Criterion 13.2 Turtle file load
Pass The tool can load RDF data from a URL or a local file in the Turtle format.

Criterion 13.3 N-Triples file load
Pass The tool can load RDF data from a URL or a local file in the N-Triples format.

Criterion 13.4 JSON-LD file load
Pass The tool can load RDF data from a URL or a local file in the JSON-LD format.

Criterion 13.5 RDFa file load
Pass The tool can load RDF data from a URL or a local file containing RDFa annotations.

Criterion 13.6 N-Quads file load
Pass The tool can load RDF data from a URL or a local file in the N-Quads format.

Criterion 13.7 TriG file load
Pass The tool can load RDF data from a URL or a local file in the TriG format.

Criterion 13.8 Handling of bigger files
Pass The tool implements a strategy for processing of bigger RDF files.
Pass example The tool implements streaming as a form of RDF processing.
Fail example The tool can only handle loading of bigger files related to the size of the available memory.

Criterion 13.9 CSV on the Web load
Pass The tool can load an uploaded CSV file accompanied by a CSVW JSON-LD descriptor.
Pass The tool can load data from a given URL of a CSVW JSON-LD descriptor that contains a reference to the CSV data file.
Pass The tool can load a CSV file from a given URL assuming the server also provides the CSVW JSON-LD descriptor in the location described by the CSV on the Web specification.

Criterion 13.10 Direct mapping load

Pass Using the Direct mapping specification the tool can load data directly from a relational database.

Criterion 13.11 R2RML load
Pass Using R2RML mapping the tool can load data directly from a relational database.

Criterion 14.1 Linked Data Platform containers
Pass The user can provide a URL of an LDP container and the tool can extract data from it. The tool provides the user with loading options related to LDP containers.
Pass example The tool offers an interactive wizard guiding user through process of data extraction.
Fail example The tool simply downloads the content of a given URL without knowing that it represents the LDP container.

Criterion 15.1 Monitoring of input changes: polling
Pass The tool can periodically check for changes in the data source. If there are changes, it can trigger an action.
Pass example The tool can periodically use HTTP HEAD request using a given URL to detect changes and download data if any change occurs.
Fail example The tool only provides support for periodical downloading of the dataset.

Criterion 15.2 Monitoring of input changes: subscription
Pass The tool can subscribe to a service (data source, etc.) to receive notifications about data updates and trigger actions.
Pass example The tool can subscribe to a data source using WebSub, and download new data whenever it is notified.

Criterion 16.1 Manual versioning support
Pass The tool can store multiple versions of the same dataset. The user can work with any of them.

Criterion 16.2 Automated versioning support
Pass The tool can automatically synchronize versions of a dataset with the data source.
Pass example The tool can watch a Git repository containing RDF files. Upon change in the repository the tool can pull the new version of the data.
Fail example The tool can only periodically delete and download the dataset to keep it updated to latest version.

Criterion 16.3 Source versioning support
Pass The tool supports a protocol for asking for a data from a specified time instant/range.
Pass example The tool supports the Memento protocol.

Criterion 17.1 Shared resources analysis
Pass The tool can perform an analysis of shared resources among datasets.
Pass example The tool can compute the number of shared resources among datasets.

Criterion 17.2 Link analysis
Pass The tool can perform an analysis of links among datasets.
Pass example The tool can compute the number of links among datasets.

Criterion 17.3 Shared vocabularies analysis
Pass The tool can perform an analysis of vocabularies shared among datasets.
Pass example The tool can show the names of shared vocabularies.
Pass example The tool can show the shared predicates and classes among datasets.

Criterion 18.1 Link discovery
Pass The tool provides support for (semi-)automated link discovery.
Pass example Given two datasets the tool can discover and save new links between them.
Pass example The tool embeds Silk.

Criterion 18.2 Ontology alignment
Pass Given a source and a target vocabulary or ontology the tool can discover and save a mapping between them.

Criterion 19.1 Vocabulary mapping based transformations
Pass The tool can transform data to another vocabulary using a mapping (for example OWL, R2R).

Criterion 19.2 Vocabulary-based SPARQL transformations
Pass The tool can transform data to another vocabulary using a predefined SPARQL query.

Criterion 20.1 RDFS inference
Pass Support inference based on inference rules encoded in the RDFS vocabulary.

Criterion 20.2 Resource fusion
Pass Given the links between resources the tool can perform resource fusion and provide support for solving conflicts.
Fail example The tool can only rename resources in one datasets so the they match the ones from the other dataset.

Fail example The tool can only copy all predicates and values from one resource to another.

Criterion 20.3 OWL inference
Pass Support inference based on inference rules described in OWL ontologies.

Criterion 21.1 Assisted selection
Pass The tool provides a graphical interface for selection of data (filtering of resources using values) which the user can use to transform RDF data.
Pass example The tool assists the user with formulating a SPARQL query in with a graphical query builder.
Fail example The tool only provides a query editor in the form of a text area.
Fail example The tool provides only a visualisations with filters, i.e. filters can not be used to transform data.

Criterion 21.2 Assisted projection
Pass The tool provides a graphical interface for data projection (selecting properties) which the user can use to transform RDF data.
Pass example The tool assists the user with formulating a SPARQL query in with a graphical query builder.
Fail example The tool only provides a query editor in the form of a text area.
Fail example The tool provides only a visualisations with filters, i.e. filters can not be used to transform data.

Criterion 22.1 Custom transformations
Pass The user can transform data using custom transformations.
Pass example User can transform data using a SPARQL query.
Pass example User can transform data using a tool-specific language.

Criterion 23.1 Automated data manipulation
Pass The tool provides the user with possible data transformations that were automatically generated or discovered.
Pass example Given input data the tool can generate transformations pipelines.

Criterion 24.1 Provenance input
Pass The tool is able to load the provenance information and employ it in some way.
Pass example The tool can load and utilize provenance information using the PROV-O ontology.

Criterion 24.2 Provenance output
Prerequisite The tool supports data output functionality.
Pass Together with the data the tool can output provenance information using RDF and the PROV-O ontology or similar.

Criterion 25.1 License awareness
Pass The tool is aware of licensing information present in dataset metadata and allows the user to see it.

Criterion 25.2 License management
Prerequisite 25.1 License awareness
Pass The tool can assists the user with combining datasets with different licenses.
Pass example The tool notifies the user when datasets with different licenses are used together and lets the user select the license of the output.

Criterion 26.1 Graph visualization
Pass The tool can visualize the graph structure of RDF data.

Criterion 26.2 Manual visualization
Pass The tool allows the user to graphically visualize selected datasets. The tool requires user assistance to select properties, data sources, visualization types etc.
Pass example The tool provides the user with a map visualization, however, the user must provide the tool with identification of labels, longitude and latitude properties and only then the tool can visualize the resources.

Criterion 26.3 Vocabulary-based visualization
Pass The tool can explore datasets and offer the user an automatically generated visualization. The user does not have to provide any additional information besides which dataset to visualize.

Criterion 27.1 RDF/XML file dump
Pass The tool allows the user to output data in the RDF/XML format.

Criterion 27.2 Turtle file dump
Pass The tool allows the user to output data in the Turtle format.

Criterion 27.3 N-Triples file dump
Pass The tool allows the user to output data in the N-Triples format.

Criterion 27.4 JSON-LD file dump

Pass The tool allows the user to output data in the JSON-LD format.

Criterion 27.5 RDFa file dump
Pass The tool allows the user to output data as an RDFa annotated file.

Criterion 27.6 N-Quads file dump
Pass The tool allows the user to output data in the N-Quads format.

Criterion 27.7 TriG file dump
Pass The tool allows the user to output data in the TriG format.

Criterion 28.1 SPARQL Update support
Pass The tool can output a selected dataset to a remote SPARQL endpoint using SPARQL Update.

Criterion 28.2 SPARQL: named graphs support
Prerequisite 28.1 SPARQL Update support
Pass The user can specify the target named graph for SPARQL Update queries.

Criterion 28.3 SPARQL: authentication support
Prerequisite 28.1 SPARQL Update support
Pass The tool allows the user to provide credentials for access to a remote SPARQL endpoint.

Criterion 28.4 SPARQL Graph Store HTTP Protocol output
Pass The tool can output a selected dataset to a remote endpoint using the Graph Store HTTP Protocol.

Criterion 29.1 Linked Data Platform Containers Output
Pass The tool can output a selected dataset to a LDP container. The tool must allow the user to utilize the features of LDP when loading the data.

Criterion 30.1 CSV file output
Pass The tool can output data in the CSV format.

Criterion 30.2 CSV on the Web output
Pass The tool can output data in the CSV format along with its CSV on the Web JSON-LD descriptor.

Criterion 30.3 XML file output
Pass The tool can output data in the XML format.

Criterion 30.4 JSON file output
Pass The tool can output data in the JSON format.

Criterion 30.5 Output for specific third-party tools
Pass The tool can output data in a way that the data can be loaded in a specific third-party software.
Pass example The tool enables the user to export data as a specific CSV file with graph data for Gephi.
Pass example The tool enables the user to export data specifically for Fuseki instead of a generic SPARQL endpoint.

Criterion 31.1 Existence of API
Pass The tool has an API which enables other tools to work with it.
Pass example Human designed RESTful API.
Fail example Generated HTTP API used for communication with a client part of the tool, e.g. as in Vaadin.

Criterion 31.2 API documentation
Prerequisite 31.1 Existence of API
Pass There is a documentation for the API.

Criterion 31.3 Full API coverage
Prerequisite 31.1 Existence of API
Pass All the actions that can be performed by user interaction with the tool can also be performed throughout the API calls.

Criterion 32.1 Configuration in open format
Pass The configuration of the tool, e.g. the projects it saves, is in an open format.

Criterion 32.2 Configuration in RDF
Pass The tool, e.g. the projects it saves, can be configured using the RDF format.

Criterion 33.1 Repository for source code
Pass There is a publicly available repository with the source code of the tool.

Criterion 33.2 Repository for plugins
Prerequisite The tool supports plug-ins.
Pass There is a publicly available plug-in repository.
Pass example The tool supports a plug-in repository/-marketplace from where the user can download and install new plugins.
Pass example Git repository for plug-ins is mentioned in the documentation or referenced from the tool.

Criterion 33.3 Repository for data processes
Prerequisite There are projects, e.g. pipelines or workflows, created in the tool.
Pass There is a publicly available projects repository.
Pass example The tool supports a project repository/-marketplace from where the user can download or where the user can upload projects.

Pass example Git repository for projects is mentioned in the documentation or referenced from the tool.

Criterion 33.4 Community activity
Pass There were at least 2 commits from 2 persons in the last year across all the relevant repositories.

Criterion 34.1 Deployment of services
Prerequisite There are projects, e.g. pipelines or workflows, created in the tool.
Pass The transformation pipeline or workflow in the tool can be exposed as a web service.
Fail example A simple SPARQL query editor, where the query is part of a URL, does not fulfill this criterion.