FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Comparison and Implementation of LOD Frameworks

Finding evaluation and application of implicit existing best practice frameworks

BACHELORARBEIT

zur Erlangung des akademischen Grades

Bachelor of Science

im Rahmen des Studiums

Software und Information Engineering

eingereicht von

Lukas Baronyai
Matrikelnummer 1326526

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Pretitle Forename Surname, Posttitle
Mitwirkung: Pretitle Forename Surname, Posttitle
Pretitle Forename Surname, Posttitle
Pretitle Forename Surname, Posttitle

Wien, 6. Februar 2017

Lukas Baronyai	Forename Surname

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Comparison and Implementation of LOD Frameworks

Finding evaluation and application of implicit existing best practice frameworks

BACHELOR'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science

in

Software and Information Engineering

by

Lukas Baronyai
Registration Number 1326526

to the Faculty of Informatics

at the TU Wien

Advisor: Pretitle Forename Surname, Posttitle
Assistance: Pretitle Forename Surname, Posttitle
Pretitle Forename Surname, Posttitle
Pretitle Forename Surname, Posttitle

Vienna, 6th February, 2017

Lukas Baronyai	Forename Surname

Erklärung zur Verfassung der Arbeit

Lukas Baronyai
Längenfeldgasse 28/8/5, 1120 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 6. Februar 2017

Lukas Baronyai

Acknowledgements

Enter your text here.

Abstract

Enter your text here.

Contents

CHAPTER 1

Introduction

Enter your text here.

1.1 Research Question

RQ: *How do common LOD-frameworks compare against each?*

1. **RQ1:** What are existing frameworks?
2. **RQ2:** What are criteria to compare frameworks?
3. **RQ2:** How do they compare against each other?

1.2 Objective

1.3 Methodology

1.4 Structure of this Paper

CHAPTER 2

State Of The Art (RQ1)

In order to compare frameworks an understanding of existing frameworks is necessary. This section will look at existing frameworks, what kind of frameworks they are, which of them can be used for this paper and which must be excluded. Furthermore, this section aims to understand how frameworks look like and will examine the architecture of them.

2.1 Architectures Of Frameworks

In this subsection the paper will look into three proposed models how frameworks (and/or implemented LD-applications) should look like. There are many other existing architectures and ongoing projects exposing data as Linked (Open) Data, this paper will use the following as representation of them.

2.1.1 Euclid Project

The EUCLID project [1](EdUcational Curriculum for the usage of Linked Data) was founded under the *Seventh Framework Programme of Research and Technological Development*, a funding program of the European Union/European Commission for 2007-2013 [2,3].

[1] [13]
[2] [14]
[3] EUCLID in the CORDIS database: `http://cordis.europa.eu/project/rcn/103709_en.html`

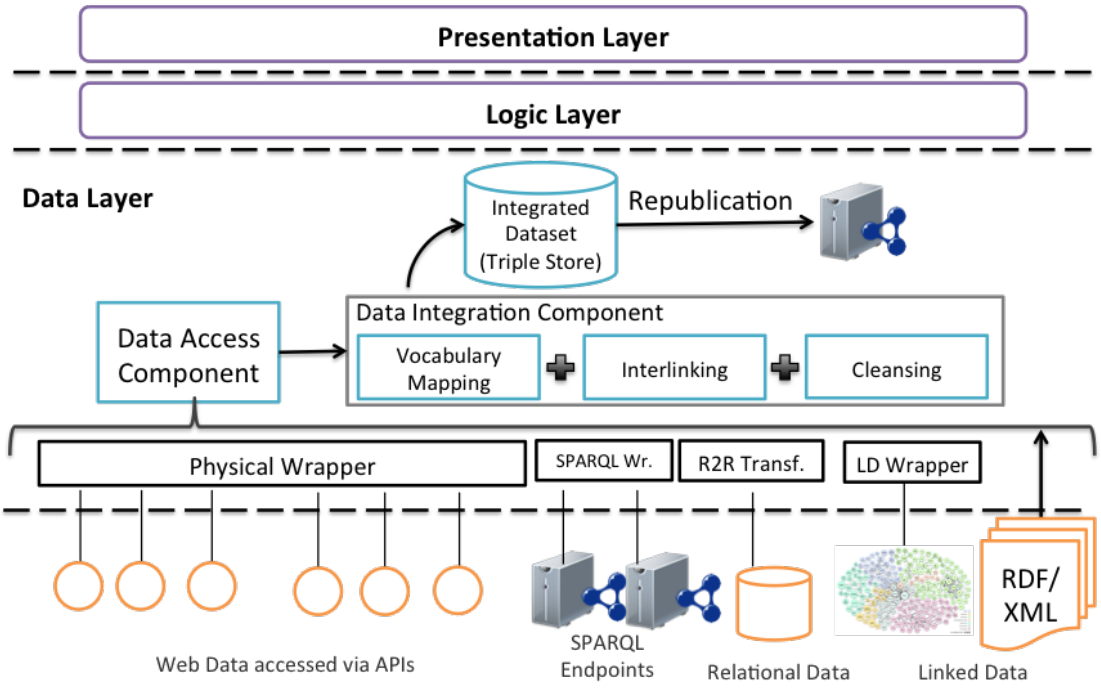

Figure 2.1: General EUCLID architecture

Aim of the project was (and still is) to gather existing knowledge and expertise of *"researchers, technology enthusiasts and early adopters in various European Member States"* and provide that accumulated as educational resources to enable the full benefit of L(O)D for European businesses. The project built upon a consortium experienced in *"over 20 LD projects with over 40 companies and public offices in more than 10 countries"* [15]

The outcome of this project is a a range of learning materials, fragmented into modules, and eLearning distribution channels. Overall there are six modules:

1. **Introduction and Application Scenarios** The introduction provides the knowledge to understand, *what* Linked Data are, the main principles, the standards and the required technologies. Further, an overview how to publish and to consume the data is given.

2. **Querying Linked Data** This chapter mainly describes SPARQL and how to use it for querying and updating.

3. **Providing Linked Data** This module deals with the production and exposure of Linked data, using the tools as R2RML (for relational databases), Open Refine (for spreadsheets), GATECloud (for natural language) and Silk (for interlinkage between datasets, see section 2.2.4 for details about this tool)

4. **Interaction with Linked Data** The projects describes in this chapter, how to explore Linked Data, using visualization tools, semantic browsers and applications, introducing search options like faceted search, concept-based search and hybrid search.

5. **Creating Linked Data Applications** This module describes how to build a Linked Data Application, which technologies to use and how to integrate common Web APIs.

6. **Scaling up** Finally this chapter examines the main issues of scalability regarding Linked Open Data and describes the relationship to Big Data.

For this paper module 3 and 5 [4] are the most interesting. Module 3 describes some useful technologies for various steps on the way of exposing L(O)D, but module 5 introduce a high level architecture and some patterns, how a L(O)D application might look like (see [16] for details). In detail, they provide a three-tier architecture (see figure 2.1 and three architecture patterns.

The architecture is very generic and consists of the classic three tiers: presentation, logic and data, each independent to the overlaying tier. Since the presentation and logic layer does not concern the actual publishing of the data, the data layer is the interesting one here. The layer consists of the *Data Access Component*, which represents the access to different data types like relational data or other Web APIs and transforms the data to RDF, the *Data Integration Component*, which does the vocabulary mapping and interlinking for the cleansing in order to e.g. identify and fix ambiguities in resource names, and finally the *Triple Store*, holding the integrated dataset for exposing it to the web.

The mentioned patterns to use for implementations are:

- **Crawling pattern** Used for loading the data in advance and storing them in a triple store, increasing the efficiency of data access. In exchange, the data might not be up to date when accessed

- **On-The-Fly Dereferencing Pattern** Meaning that the URIs are dereferenced when the application need to access the data. This pattern provides up to date data but for the cost of performance when dereference many URIs.

- **(Federated) Query Pattern** Describing the use of complex queries on a fix set of data sources, enabling to work with current data directly retrieved from the sources. The pattern offers an access up-to-date data with adequate response time in specific situations but for the cost of the complex problem to find optimal queries.

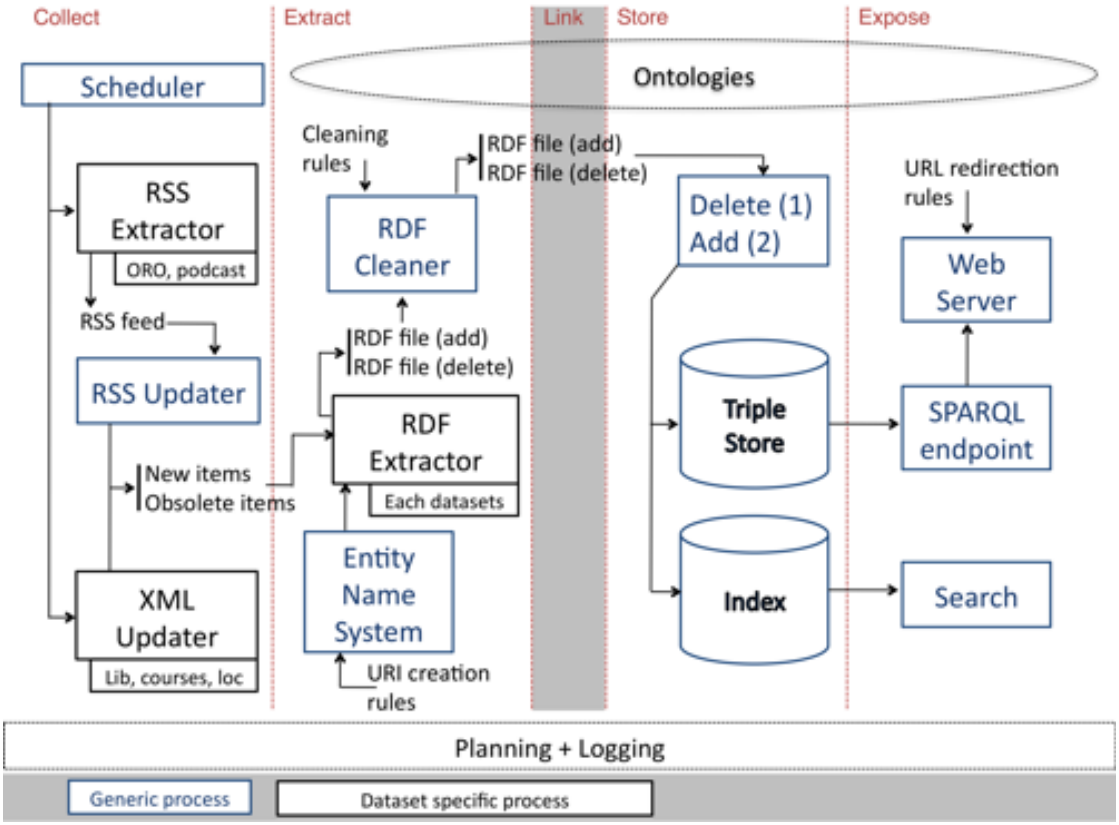

Figure 2.2: LUCERO work flow & architecture

2.1.2 LUCERO

The LUCERO project ("Linking University Content for Education and Research Online") [5] was a project at the Open University, aiming to *"scope, prototype, pilot and evaluate reusable, cost-effective solutions relying on the linked dataprinciples and technologies for exposing and connecting educational and research content"*. It was founded for one year by the JISC Information Environment 2011 Programme under the call Deposit of research outputs and Exposing digital content for education and research. [17]

The projects connected with other organizations through LinkedUniversities.org [6] to gather common issues and practices. The outcome was the first university linked data platform, `http://data.open.ac.uk/`, with a lot of impact on The Open University and the education community.

Looking at the architecture in figure 2.2 comparing to the Euclid architecture seen in the

[4] [16]

[5] The code is available in the Google Code Archive: `https://code.google.com/archive/p/luceroproject/wikis/StepByStepDocumentation.wiki`

[6] `http://linkeduniversities.org/`

previous section, there are quite a lot of similarities. Both have components for accessing different kinds of data, here called *Extractors*, for cleaning the data, here called *Cleaner*, and a Triple Store, holding the data available. The lanes "Collect", "Extract", "Link" and "Store" can be seen as the data layer from the classic three-tier architecture, the "Expose" lane as the logic and presentation layer.

Both using the crawling pattern to extract, map and store the data in a Linked Data format instead of transforming them for every request.

TABLOID

One of the outcomes next to the LOD application itself was the Tabloid ("Toolkit ABout Linked Open Institutional Data"), *"a toolkit intended to help institutions and developers to both publish and consume linked data"*. It contains work-flows, documentations, examples and tools [18] trying to address different roles such as managers, developers and users. Tabloid try to help people to understand LD, what can be done with it and give advice on a technical perspective, how to publish and consume LD, providing at the same time a detailed and generic way.

2.1.3 Linked Data book

Another big effort ampng many others of describing LD in general, how to publish and consume them and how to implement applications was done by the book "Linked Data: Evolving the Web into a Global Data Space" by Heath and Bizer [1], which received a lot of attention.

The book aims in general to give a basic understanding of LD and describing publication and consumption of LD. They providing advices and best practices, including architectures approaches, identifying the right set of URIs and vocabulary and much more. They also described an architecture, to be seen in figure 2.3

Next to patterns they also provide a general workflow for LD publishing, see figure 2.3. But comparing to the introduced architectures in the previous sections, the workflow has a different approach: instead of holding the data in a Triple Store, the workflow access and transforms the raw data on-the-fly for every request.

Next to this workflow, the book also provides various "recipes" for publishing LD and one of them is also to hold the data in a triple store as shown by Euclid and LUCERO. Furthermore the book provides a guide for the D2R-Server, which will be described in section 2.2.1.

2.2 Frameworks

2.2.1 D2RQ Platform

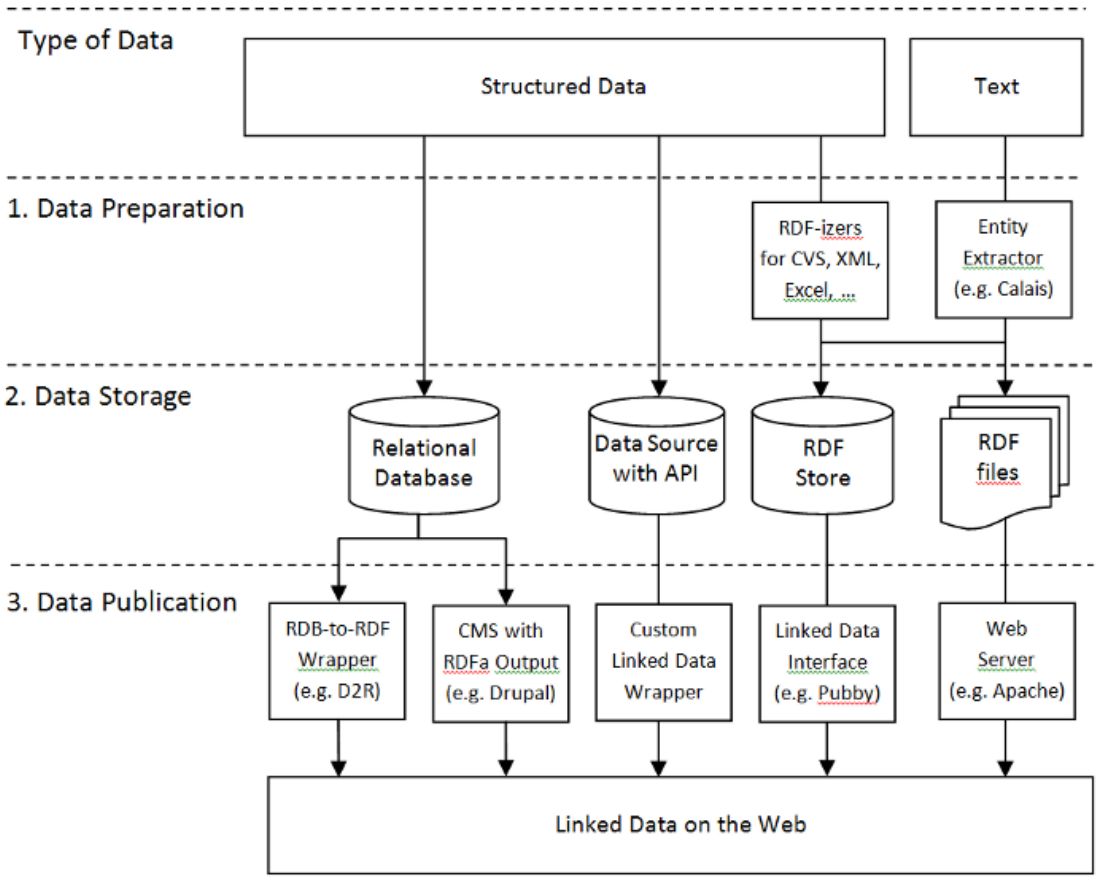

Figure 2.3: Linked Data Publishing Options and Workflows according to the LD book

NOTE: The last update on the D2RQ platform was in 2012 (version 0.8.1) and on the D2R Server in 2009 (version 0.7)

The D2RQ platform [7] was introduced by the Free University of Berlin and provides a database-to-RDF mapping. It is licensed under the terms of the GNU General Public License.

To map a relational database the platform provides a declarative mapping language, expressed in RDF, which is then be used to provide access to the database in the following, read-only, ways: [5]

- **RDF dumps**
- **RDF APIs**

[7] `http://d2rq.org`

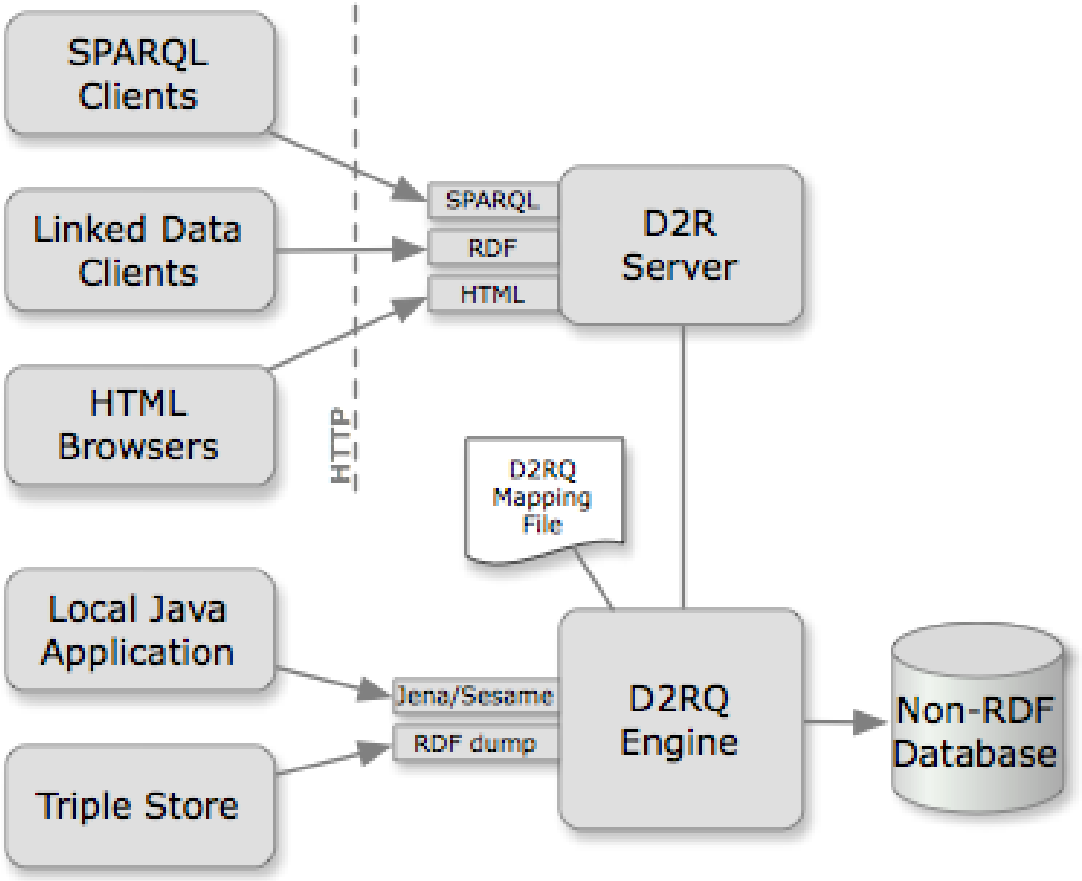

Figure 2.4: D2R Server architecture

- **SPARQL endpoint** (D2R Server)
- **Linked Data**
- **HTML view** (D2R Server)

For an overview of the framework structure see figure 2.4.

D2R Server

Part of the platform is the D2R Server [8], which provides the public access to the platform over SPARQL and HTML, publishing it to the semantic web. More concrete, the server provides a dereferencing interface, for HTTP request dereferencing, and a SPARQL interface.

[8] `http://d2rq.org/d2r-server`

The server uses the mentioned **On-The-Fly Dereferencing Pattern** and does not provide a triple store, therefore it may be not has as good performance than tools with a triple store, although the team made a great effort to improve it.

Part of the server is also a tool which generates automatically a corresponding mapping and RDF vocabulary for an existing table structure, using table names as class names and column names as property names. The generated mapping file can then be customised. [6]

The following applications are examples using D2R-Server:

- DBLP Bibliography (University of Hannover) [9]
- DBtune (University of London) [10]
- Database of the Nobel Prize [11]

2.2.2 Information Workbench

The Information Workbench [12] is a high customisable tool to support the building of Linked Data applications, from basic data integration up to rich UI and visualisations. The tool is developed by fluidOps and is published as Community Edition free available and under an Open Source License with a limited selection of capablities and only for non-productive use (educational use, testing, development). The enterprise edition is also available but not for free.

The workbench consists of four layers (see figure 2.5 for an overview): [7] [8]

- **Persistence** Using so-called *providers*, the layers offer capabilities to integrate and convert data from different data source and stores them in a central triple store. Alternatively it also supports virtualised integration of local and public Linked Data sources using a *federation layer*.
- **Platform** On top of the persistence layer the core Platform layer a selection of modules and functionalities covering generic needs of Linked Data applications, the most important are a *Semantic Wiki & Widget Engin*e, an *User Management & Access Control*, a *Search & Analytics Engine* and a *Workflow Engine*.
- **SDK** To support customised applications the workbench provides a SDK (Solution Development Kit) for developers to build domain specific applications, including *extensible data providers*, *data management facilities*, modified *ontologies*, *templates*, *widgets* and different APIs for extensive *system configuration*, *rules* and *workflows*.

[9] http://dblp.uni-trier.de/
[10] http://dbtune.org/
[11] http://data.nobelprize.org/
[12] https://www.fluidops.com/en/products/information_workbench/

- **Solution** On top of all layer stands the final solution, the application itself, which is either directly deployed through a RESTful API or over a zipped file for other installation approaches.

The resulting application is again customisable by widget and different views, enabling data exploration and visualisation.

2.2.3 Synth

2.2.4 Silk - The Linked Data Integration Framework

`http://silkframework.org/` [2] [9] [10]

2.2.5 LDIF – Linked Data Integration Framework

`http://ldif.wbsg.de/` [11] [12]

2.3 Excluded Tools and Projects

2.3.1 LD-Patterns

The Linked Data Patterns book by Dodds and Davis (see [3]) tried to give an overview of existing design pattern regarding LD. But they don't give concrete architectures or architecture relating informations, so this paper will not use its content. But it is suggested, that this design pattern catalogue is used additionally when creating an application.

2.3.2 LOD2 Stack

The LOD2 stack, introduced by Auer et. al., is *is an integrated distribution of aligned tools which support the whole life cycle of Linked Data from extraction, authoring/creation via enrichment, interlinking, fusing to maintenance.* [0] For this paper the proposed stack of technology was too generic to compare it with other frameworks and the website of the project [13] was at point of writing this paper offline, therefore it was excluded of this paper.

2.3.3 LODUM

Another interesting project is the LODUM project (Linked Open Data University of Münster), the Open Data initiative of the university, hosted at the Institute for Geoinformatics' Semantic Interoperability Lab (MUSIL). The project team has co-initiated both LinkedUniversities.org and LinkedScience.org.

[13] `http://stack.linkeddata.org/lod2//`

It was excluded for this paper because the project don't provide public documentation of their architecture or any other part of their technical details `http://lodum.de/`

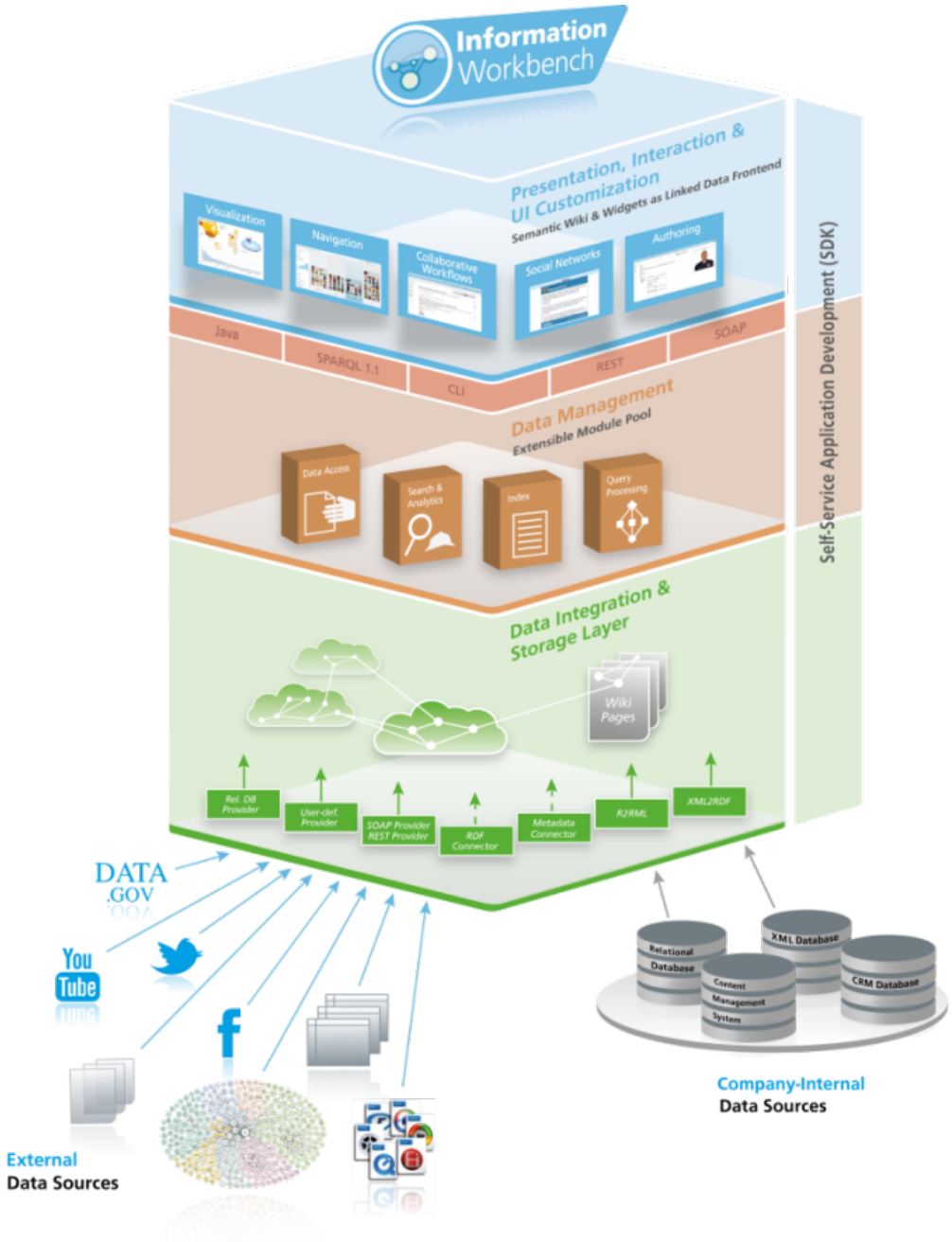

Figure 2.5: Architecture of the Information Workbench

CHAPTER 3

Methodology (RQ2 & RQ3)

Enter your text here.

3.1 Definitions for this paper

3.1.1 Framework

3.2 About the difficulty of comparing frameworks

3.3 Criteria

- Maintainability How much effort needs the maintenance?
- Data quality
  - Data freshness (ability to handle new data)
  - Flexibility (of ontology) (deal with heterogenous and/or legacy data)
- Usability: (adopted from Abran et.al. [4] to fit)
  - Effectiveness (How well do the users achieve their goals using the system?)
  - Efficiency (Time to achieve one task, complexity to handle)
  - Satisfaction
  - Security
  - Learnability (Documentation)

Rating	Explanation
–	high negative impact
-	negative impact
0	no impact
+	positiv impact
++	high positiv impact

Table 3.1: My caption

  - Performance

- Available data formats (HTML, Relational Databases, Wrapping Existing Application or Web APIs, XML, Tables/Spreadsheets)
- Linked Data Publishing Checklist (from Heath et.al. [1])
  - Does your data set links to other data sets?
  - Do you provide provenance metadata?
  - Do you provide licensing metadata?
  - Do you use terms from widely deployed vocabularies?
  - Are the URIs of proprietary vocabulary terms dereferenceable?
  - Do you map proprietary vocabulary terms to other vocabularies?
  - Do you provide data set-level metadata?
  - Do you refer to additional access methods?

3.3.1 Rating

Since a rating scale for each criteria or in each category would go beyond the scope of this paper, a much simpler scale will be used, seen in table **??**.

3.4 Test setting

define scale

To ensure comparable results with different frameworks, the same test setting will be used for each of them. A SQL database is assumed with x datasets, representing a publication database. Each framework then will be used to build a SPARQL endpoint under Ubuntu 16.10 (Yakkety Yak) and be rated against the proposed criteria.

CHAPTER 4

Comparison (RQ2 & RQ3)

Enter your text here.

4.1 Comparison of the Frameworks

4.1.1 Maintainability

4.1.2 Data quality

4.1.3 Usability

4.1.4 Data formats

4.1.5 Linked Data Publishing Checklist

4.2 Summary

CHAPTER 5

Critical reflection

Enter your text here.

5.1 Existing Best Practice

5.2 Analysis of the Implementation

5.3 Applicability and Adaptability

CHAPTER 6

Summary and future work

Enter your text here.

List of Figures

List of Tables

Index

References to refereed scientific work

[1] T. Heath and C. Bizer, "Linked data: Evolving the web into a global data space", *Synthesis lectures on the semantic web: theory and technology*, vol. 1, no. 1, pp. 1–136, 2011.

[2] J. Volz, C. Bizer, M. Gaedke, and G. Kobilarov, "Silk-A Link Discovery Framework for the Web of Data.", *LDOW*, vol. 538, 2009.

[3] L. Dodds and I. Davis, "Linked data patterns", *Online: http://patterns. dataincubator. org/book*, 2011.

[4] A. Abran, A. Khelifi, W. Suryn, and A. Seffah, "Usability meanings and interpretations in ISO standards", *Software Quality Journal*, vol. 11, no. 4, pp. 325–338, 2003.

References to non-refereed work

[5] C. Bizer and R. Cyganiak, "D2rq-lessons learned", in *W3C Workshop on RDF Access to Relational Databases*, 2007, p. 35. [Online]. Available: `https://www.w3.org/2007/03/RdfRDB/papers/d2rq-positionpaper/`.

[6] ——, "D2r server-publishing relational databases on the semantic web", in *Poster at the 5th international semantic web conference*, vol. 175, 2006.

[7] 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*, CEUR-WS. org, 2011, pp. 119–124.

[8] A. Gossena, P. Haasea, C. Hüttera, M. Meiera, A. Nikolova, C. Pinkela, M. Schmidta, A. Schwartea, and J. Tramea, "The Information Workbench–A Platform for Linked Data Applications",

[9] A. Jentzsch, R. Isele, and C. Bizer, "Silk-generating rdf links while publishing or consuming linked data", in *Proceedings of the 2010 International Conference on Posters & Demonstrations Track-Volume 658*, CEUR-WS. org, 2010, pp. 53–56.

[10] R. Isele, A. Jentzsch, and C. Bizer, "Silk server-adding missing links while consuming linked data", in *Proceedings of the First International Conference on Consuming Linked Data-Volume 665*, CEUR-WS. org, 2010, pp. 85–96.

[11] A. Schultz, A. Matteini, R. Isele, C. Bizer, and C. Becker, "Ldif-linked data integration framework", in *Proceedings of the Second International Conference on Consuming Linked Data-Volume 782*, CEUR-WS. org, 2011, pp. 125–130.

[12] A. Schultz, A. Matteini, R. Isele, P. N. Mendes, C. Bizer, and C. Becker, "LDIF-a framework for large-scale Linked Data integration", in *21st International World Wide Web Conference (WWW 2012), Developers Track, Lyon, France*, 2012.

References to websites

[13] EUCLID, *EUCLID — EdUcational Curriculum for the usage of LInked Data*, [Online; accessed 5-September-2016], 2012-2014. [Online]. Available: http://euclid-project.eu/index.html.

[14] E. Union, *Framework Programmes for Research and Technological Development*, [Online; accessed 8-September-2016], 2012-2014. [Online]. Available: https://ec.europa.eu/research/fp7/index_en.cfm.

[15] EUCLID, *About Euclid*, [Online; accessed 5-September-2016], 2012-2014. [Online]. Available: http://euclid-project.eu/about/project-description.html.

[16] ——, *EUCLID — Chapter 5: Building Linked Data Applications*, [Online; accessed 5-September-2016], 2012-2014. [Online]. Available: http://euclid-project.eu/about/project-description.html.

[17] M. d'Aquin, F. Zablith, E. Motta, O. Stephens, S. Brown, S. Elahi, and R. Nurse, *The LUCERO project – About*, [Online; accessed 15-September-2016], 11 Jun 2010. [Online]. Available: http://lucero-project.info/lb/about/index.html.

[18] ——, *The LUCERO project – Tabloid*, [Online; accessed 15-September-2016], 1Jun 2011. [Online]. Available: http://lucero-project.info/lb/tabloid/index.html.