Report on
Landscape Analysis of Ontology Engineering Tools

Grant Agreement: 958371

OntoCommons - Ontology-driven data documentation for Industry Commons, has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement no. 958371.

Project Title	Ontology-driven data documentation for Industry Commons
Project Acronym	OntoCommons
Project Number	958371
Type of project	CSA - Coordination and support action
Topics	DT-NMBP-39-2020 - Towards Standardised Documentation of Data through taxonomies and ontologies (CSA)
Starting date of Project	01 November 2020
Duration of the project	36 months
Website	www.ontocommons.eu

Report on Landscape Analysis of Ontology Engineering Tools

Work Package	Ontology Commons Ecosystem Toolkit
Task	Landscape Analysis of Ontological Platform Support
Lead authors	Martin G. Skjæveland (UiO), Laura Slaughter (UiO), Christian Kindermann (UiO)
Contributors	Jinzhi Lu (UiO), Emna Amdouni (ENIT), Francesca L. Bleken (SINTEF), Jesper Friis (SINTEF), Kevin Haller (TU Wien), Casper W. Andersen (SINTEF), Daniel Garijo (UPM), María Poveda-Villalón (UPM), Serge Chavez (UPM), Lan Yang (NUIG), Laura Waslmayr (TU WIEN), Stefani Tsaneva (TU WIEN)
Peer reviewers	Hedi Karray (ENIT), John Breslin (NUIG)
Version	Final
Date	20/04/2022

Keywords

Ontology Engineering, Toolkit, Ontology Tools, Software Systems

Disclaimer

OntoCommons.eu has received funding from the European Union's Horizon 2020 research and innovation programme under the Grant Agreement no. 958371. The content of this document does not represent the opinion of the European Union, and the European Union is not responsible for any use that might be made of such content. The European Commission is not liable for any use that may be made of the information contained herein.

Executive Summary

This report provides a landscape analysis of software systems for ontology engineering. We collect software systems that are said to be used in practice and compile them in an index providing information about their homepage, documentation, and other publicly available materials. Furthermore, we classify collected software systems according to high-level categories that have been proposed in the report on "Ontology Ecosystem Specification". This categorisation can be used as a first guide by practitioners to search and select software systems for ontology engineering tasks that are most suitable for their needs. Lastly, we conduct a desk review analysis of the collected software systems considering novel challenges arising from application domains as put forward by the report on "Requirements on ontology tools and ontologies and criteria for selection of further cases". We find that some areas in ontology engineering seem to be lacking in terms of tool support, e.g., modularisation, conceptual modelling, and ontology reuse. However, our results also suggests that for many challenges arising in practice there is often some kind of tool support that may prove to be useful for addressing said challenges.

Table of Contents

List of Figures

List of Tables

1. Introduction

Advances in Ontology Engineering give rise to a constantly evolving landscape of methodological guidelines and software systems to facilitate ontology development and maintenance. There exists many software systems for semantic technologies and new ones are being created to take into account novel challenges arising in various application domains. Yet, there is no established software repository that indexes such software systems to facilitate their dissemination.

As part of the OntoCommons project, the objective of this report is to provide a snapshot of the landscape of software systems for semantic technologies by (i) collecting software systems that are reported to be used in practice, (ii) categorising said software systems in terms of general activities during the ontology engineering life-cycle, (iii) indexing collected software systems by compiling summary information about resources such as available documentation, publications, recent releases, source code contributions, etc., and (iv) analysing collected software systems w.r.t. requirements arising in application domains.

While there are already efforts[1] to collect software systems and other resources for semantic technologies, we will limit the scope of our study to software systems that are most relevant in the context of the OntoCommons project. This means that instead of conducting a large-scale systematic review on software systems for ontology engineering, we take a more pragmatic approach and design our survey based on findings of other reports. In particular, this concerns the identification and selection of software systems as well as their categorisation.

2. Terminology

Many terms and notions used in this report are not well-defined and are to be interpreted in an informal and non-technical manner. For example, the terms "tool" and "software system" are not meant to be understood in any concrete or otherwise more specific way. So, a "tool" or "software system" is best understood as anything that is referred to as such by an ontology engineer or practitioner in the context of an empirical survey, e.g., report "Ontology Ecosystem Specification". Please note that both terms are used interchangeably in this report.

The same informal interpretation applies to notions that are usually associated with a more technical meaning in the field of ontology engineering, e.g., "modularity" or "ontology alignment". Otherwise, we provide references to other reports where notions with a more specific meaning are first introduced or described in more detail.

3. Method

We conduct a desk review survey on the landscape of software systems used in ontology engineering. We collect both quantitative as well as qualitative data for software systems via a questionnaire. The questionnaire was distributed as an online form to project members and to

[1] See for example https://github.com/semantalytics/awesome-semantic-web.

external experts associated with the OntoCommons project. Participants of our survey reviewed software systems that they are familiar with in a self-appointed manner. In the following, we present the design of this survey in more detail and provide references to other reports on which this design is based.

3.1 Survey Design

3.1.1 Research Questions and Objectives

The main motivation for our landscape analysis of software systems used for ontology engineering is to provide an overview of existing technologies in terms of their intended purpose, their usage in practice, and their relation to novel challenges arising in application domains. In particular, we are interested in the following research questions:

- What software systems are used in practice for ontology engineering?
- What services do such systems commonly support?
- Do existing systems provide suitable features as required by application domains in practice?
- To what extent are software systems for ontology engineering actively maintained?

Since a systematic review to answer these questions is out of scope for this work, we opted for a more pragmatic approach in which we consulted with project members and external experts associated with the OntoCommons project that work in the field of ontology engineering. We elicited their opinions on the matters mentioned above by using a questionnaire for reviewing one software system at a time (see Section 7 on page 75).

Our primary objectives in doing so are to

- provide an overview of software systems relevant to ontology engineering activities in practice,
- provide a high-level categorisation of available software systems,
- provide an index of existing software systems by listing their respective homepages, code repositories, documentation, and associated publications, which could be used by an ontology engineer (or practitioner) as a guide to the search for a suitable tool for a given use case, and
- conduct an analysis of features offered by currently available tools w.r.t. requirements arising in application domains.

3.1.2 Survey Scope

Due to the large number of existing software systems used in ontology engineering, we limit our survey to software systems that are reported to be used in practice. We build on work reported in report on "Ontology Ecosystem Specification" in which a workshop on ontology engineering tools was held. Most participants described themselves to be either expert practitioners, ontology users, or developers of software systems for semantic technologies. Participants of the workshop were asked to fill out a questionnaire about the state of existing ontology ecosystem toolkits, their usage, and possible directions for future work. The following tools were reported to be useful in the ontology engineering workflow by the workshop participants:

Protégé, TopBraid Composer, Text editor, GraphDB, Jena, SHACL, WIDOCO, github, OnTop, obo-ROBOT, LOV, OTTR, OWLAPI, OOPS!, Cameo, Virtuozo, OntoUML, Hermit, RDF4J, WebVOwl, Pellet Reasoner, OWBO, Fuseki, rdflib, metaphactory, Hets, PoolParty.

We excluded the responses "text editor" and "SHACL" from our survey since they cannot be associated with concrete software systems. Note that "text editor" is a category of tools and that SHACL is a specification and not a software system. Besides the 25 software systems collected, we also asked project members to suggest additional software systems that they either use themselves or consider useful for their work. This resulted in the inclusion of the following 21 software systems in our survey:

AllegroGraph, Chowlk, DBPedia Archivo, EMMOntoPy, F-uji, FaCT++ reasoner, FOOPS!, GRUFF, Menthor Editor, MetaGraph 2.0, O'FAIRE, Ontofox, ontology-toolkit/onto-tool, OnToology, OntoSpy, Owlready2, RDF4J, Stardog, SANSA, RDFS++, OntoViews.

Lastly, we asked external experts associated with the OntoCommons project for feedback regarding software systems that they use in practice. This following list of 31 software systems were reported and thus included in our survey:

Alignment Cubes, Atomic Data, CASPAR, CENtree, crowd-tool, Dynaccurate, Hozo Ontology Editor, IntelligentGraph, JSON-LD Playground, Konclude, LinkML, LUPOSDATE, LUPOSDATE3000, OData2SPARQL, Ontology Lookup Service (OLS), Ontopanel, OntoTrek, OPCloud, ORMiE, owl-db-tools, PathQL, pronto, RDFox, Semantic MediaWiki, SimPhoNy, Terminology Harmonizer, Visual Studio Code, VocBench, WIMU, yEd Graph editor.

The total number of software systems considered in this survey is 77. Note, however, that only 68 out of these 77 software systems are reviewed as will be explained in Section 3.3

3.1.3 Categorisation of Software Systems

To provide an overview of existing software system used in ontology engineering, we propose to categorise software systems according to their main purpose. The report on "Ontology Ecosystem Specification" specifies five high-level categories for ontology engineering tools. These categories can be used to gain a first understanding of the current landscape of ontology engineering tools. The categories are

1) Requirement Specification,
2) Implementation,
3) Publication,
4) Maintenance, and
5) Use.

This report also specifies more fine-grained subcategories for the above categories that we repeat here for convenience, as shown in Figure 1.

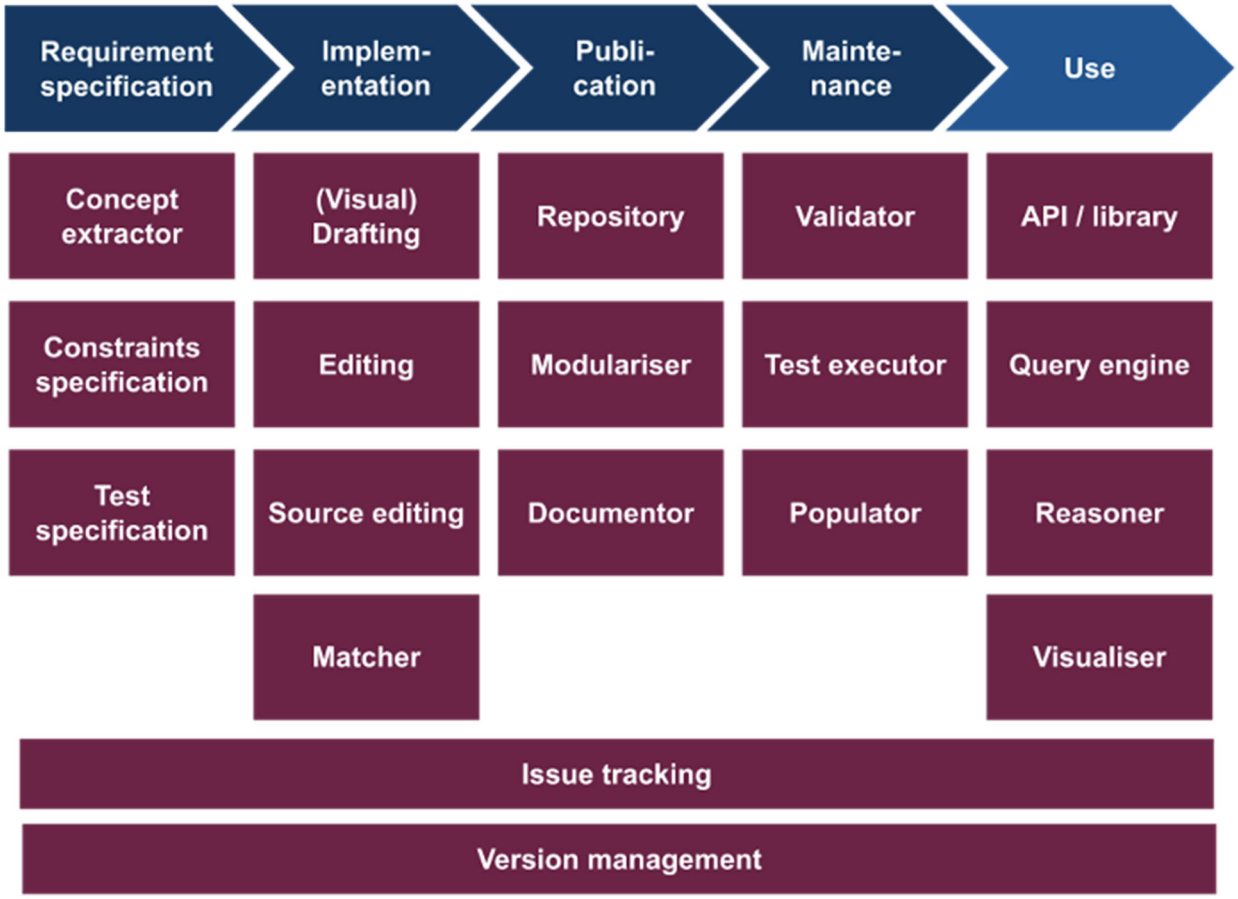

Figure 1: Components of the ontology ecosystem toolkit.

Project members were asked to discuss this categorisation for the purpose of conducting a landscape analysis of ontology engineering tools. It was suggested to

- introduce a new category for Evaluation and Validation,
- rename a few subcategories, and
- add an option that would allow reviewers to specify their own category in case no other category fits for a given tool.

This gave rise to the following categorisation:

1) **Requirement Specification tools**: a) concept-extractor, b) Constraints specification, c) Test specification
2) **Implementation tools**: a) Drafting, b) Editing, c) Source editing, d) Matcher
3) **Publication tools**: a) Repository, b) Modulariser c) Documentor
4) **Maintenance**: a) Validator, b) Text executor c) Populator
5) **Evaluation and Validation**
6) **Use**: a) API/library, b) Query engine, c) Reasoner, d) Visualiser
7) **Other**.

For convenience, we repeat the descriptions for the categories 1) to 4)

> ***Requirement Specification**: The ontology requirements specification activity aims to state why the ontology is being built and to identify and define the requirements the ontology should fulfil. In this step, involvement, and commitment by experts in the specific domain at hand is required to generate the appropriate industry perspective and knowledge.*
>
> ***Implementation:** The ontology implementation activity goal is to build the ontology using a formal language, based on the ontological requirements identified by the domain experts.*
>
> ***Publication:** During the ontology publication activity, the ontology development team should provide an online ontology which is accessible both as a human-readable document, i.e., as an HTML document with the description of the ontology and its terms, and a machine- readable file from its URI therefore enforcing FAIR Principles.*
>
> ***Maintenance**: [...] [T]he goal of the ontology maintenance activity is to update the ontology.*

The category "Evaluation and Validation" was proposed by a project members with the intent to distinguish between tasks that are related to data quality, which fall under the category of "Evaluation and Validation", and tasks that are related to managing changes of an ontology, which fall under the category of "Maintenance".

The category "Use" is also described in the report on "Ontology Ecosystem Specification" and can be summarised to encompass any tools that an end-user of an ontology might use to interact with the ontology in a given use-case scenario.

3.2 Material

The used questionnaire consists of two parts. The first part is about general information about a tool whereas the second part is about requirements as specified by OntoCommons report on "Requirements on ontology tools and ontologies and criteria for selection of further cases".

The questionnaire was distributed as an online form divided into two sections that correspond to the two different parts mentioned above. In both sections, participants were asked to provide information about a tool via text fields and multiple-choice questions. In the case of multiple-choice questions, participants were given the option to clarify their answer within a text field. The full questionnaire is attached in the appendix.

3.3 Procedure

Project members and external experts associated with the OntoCommons project were approached by email and asked to review ontology engineering tools of their choosing. They were provided with

a link to an online form to conduct a review using the questionnaire described in Section 3.2 Both groups were given two weeks to do so.

Project members were also encouraged to recruit qualified colleagues as participants, i.e., tool reviewers, for this survey by forwarding the provided materials. Please note that we were not able to recruit enough participants to review all tools initially collected for this survey as described in Section 3.1.2 To be more precise, only 68 out of the total of 77 collected tools were reviewed by some project members or an external experts.

4. Results

We report both the qualitative as well as the quantitative data collected with the method described in Section 8. There were 12 project members reviewing 38 software systems and 30 external experts reviewing a software system of their choosing. Please note that each tool was reviewed by only one person and that one project member often reviewed multiple tools.

We present the results in three separate sections that correspond to the three objectives formulated in Section 3.1.1 Namely an overview of tools used in ontology engineering in terms of high-level categories (cf. Section 4.1, an index of existing ontology engineering tools listing their homepage, code repository, etc. (cf. Section 4.2), and an analysis of features offered by currently available tools w.r.t. requirements arising in application domains (cf. Section 4.3)

4.1 Categorisation of Software Systems

4.1.1 Quantitative Results

Figure 2 shows the number of tools per category. Note that the categories are not mutually exclusive. A single tool can provide a variety of features for different purposes so that it can be considered to belong to multiple categories. It transpires that there is tool support for almost all categories with the only exception of "Test Specification". Otherwise, participants of the study suggested three additional categories, namely "Reuse", "FAIRness assessment", and "Ontology index" for the tools DBPedia Archivo, F-uji, Linked-Open Vocabularies, and O'Faire.

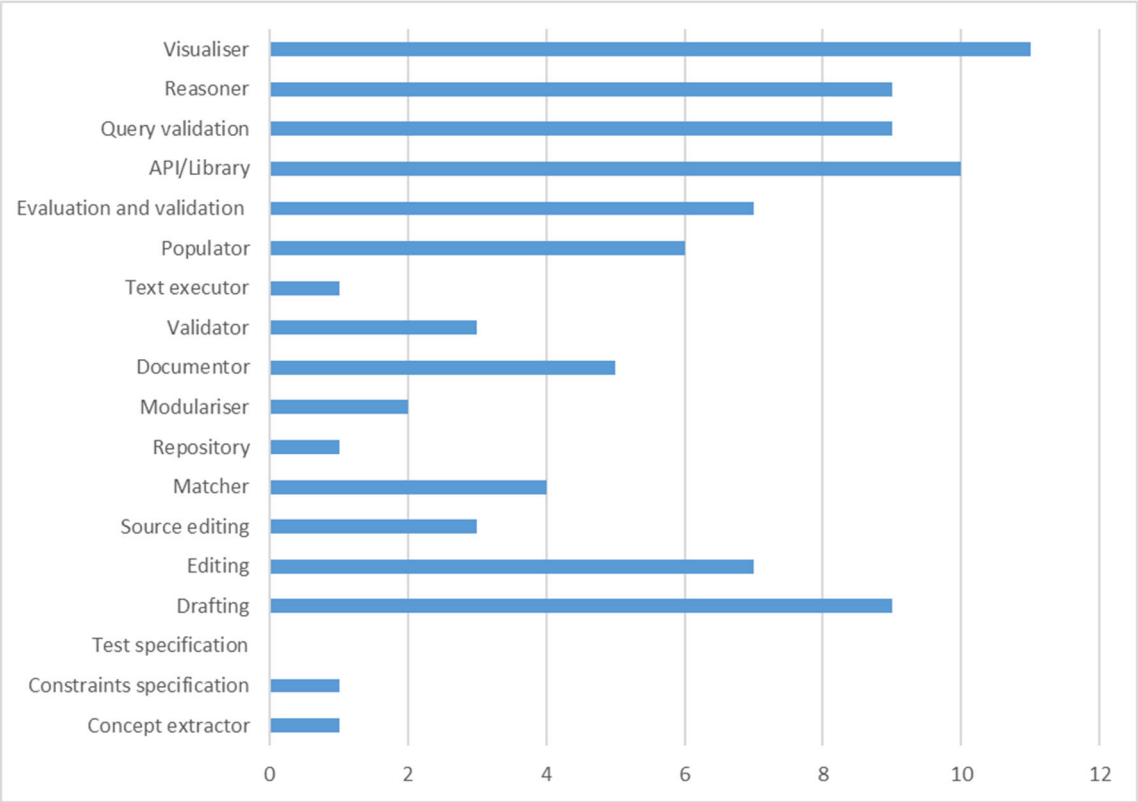

Figure 2: Tool categorisation

4.1.2 Qualitative Results

Reviewers did not provide clarifying comments on the way tools were assigned to categories. Table 1 shows each tool and the categories it was assigned to.

Table 1: Categorisation of Ontology Engineering Tools according to the following seven categories: Requirement Specification (1), Implementation (2), Publication (3), Maintenance (4), Use (5), Evaluation and Validation (6), Other (7).

Tool	1			2				3			4			5	6				7
	a	b	c	a	b	c	d	a	b	c	a	b	c		a	b	c	d	
AllegroGraph																X		X	
Apache Jena				X	X		X							X					
Chowlk				X															
DBPedia Archivo														X					X
EMMOntoPy						X				X		X	X		X			X	
F-uji														X					X

FaCT++ Reasoner														X	X			
FOOPS!											X							
GraphDB													X			X	X	X	
GRUFF																X		X	
HermiT OWL Reasoner																	X		
Hets		X																	
Menthor Editor				X	X									X					
Linked Open Vocabularies								X											X
Magic draw cameo				X	X		X							X					
MetaGraph 2.0																		X	
O'FAIRE														X				X	X
Ontofox					X	X													
OOPS!											X								
Ontology-toolkit									X	X					X				
OnToology									X	X								X	
Ontop																X	X		
OntoSpy										X					X			X	
OWBO	X			X											X				
OWL API															X				
Owlready2															X		X		
Pellet Reasoner																	X		
PoolParty				X			X						X			X	X		
Protégé				X									X			X	X	X	
RDF4J															X				
RDFLib															X				

Reasonable Ontology Templates					X								X		X			
ROBOT					X						X					X	X	
Stardog																X	X	X
TopBraid Composer				X	X	X								X				
Virtuoso					X		X						X			X		X
WebVOWL																		X
WIDOCO										X								

4.2 Index of existing Software Systems

4.2.1 Quantitative Results

Every reviewed tool was reported with a description and a publicly available homepage. Source code and documentation are available online for almost all reviewed tools. There are only five tools that have been reported without a link to a publicly available repository (some of which were reported to be offered as commercialised software), and nine tools that were reported without a link to a documentation page. Note that the tools without an online repository for their source code and tools without an online documentation page do not necessarily coincide — meaning a tool without an online repository might provide a documentation tool or vice-versa. Furthermore, more than half of the reviewed tools, namely 46, were reported to be the subject of a publication.

The majority of tools, i.e., 45, were reported to be deployable in production. Only 16 tools were described to be in development and the remaining seven tools were left uncategorised in this regard. Similarly, 59 tools were reported to be actively maintained, six were judged to be currently not maintained, and three were left uncategorised (these two are not the same two tools as the uncategorised tools w.r.t. their development stage). The tools' active maintenance is reflected in both their latest releases as well as their latest code contributions as shown in Table 2. New versions were released for most tools within the past year at the time of writing.

Table 2: Number of tools with their latest release (second column) and latest code contribution (third column) in a year.

Year	Latest Release	Latest Commit
2022	12	13
2021	29	27
2020	2	1

2019	3	2
2018	1	-
2017	-	4
2016	2	1
2013	2	-
2010	1	-
Not reported	16	20

4.2.2 Qualitative Result

We compile an index of ontology engineering tools based on a selection of items used in the questionnaire for the conducted tool review. As described in Section 9.

1. **Alignment Cubes**
    a. *Description:* AlignmentCubes is a tool which allows for interactive visual exploration of several ontology alignments and thus supports alignments' evaluation at different levels of detail.
    b. *Homepage:* https://www.ida.liu.se/~patla00/research/AlignmentCubes/
    c. *Code Repository:* https://www.ida.liu.se/~patla00/research/AlignmentCubes/
    d. *Documentation:* https://www.ida.liu.se/~patla00/research/AlignmentCubes/
    e. *Publication:* Ivanova V, Bach B, Pietriga E, Lambrix P, Alignment Cubes: Towards Interactive Visual Exploration and Evaluation of Multiple Ontology Alignments, 16th International Semantic Web Conference, LNCS 10587, 400-417, Vienna, Austria, 2017
2. **AllegroGraph**
    a. *Description:* AllegroGraph is a Horizontally Distributed, Multi-model (Document and Graph), Entity-Event Knowledge Graph technology that enables businesses to extract sophisticated decision insights and predictive analytics from their highly complex, distributed data that can't be answered with conventional databases.
    b. *Homepage*: https://allegrograph.com/products/allegrograph/
    c. *Code Repository*: https://github.com/franzinc
    d. *Documentation*: https://franz.com/agraph/support/documentation/current/agraph-introduction.html
    e. *Publication*: -
3. **Apache Jena**
    a. *Description:* Apache Jena (or Jena in short) is a free and open source Java framework for building semantic web and Linked Data applications. The framework is composed of different APIs interacting together to process RDF data, including APIs for direct RDF data manipulation, SPARQL querying with support for federated queries and free text search, persisting data, OWL, inference and rules.
    b. *Homepage*: https://jena.apache.org/

c. *Code Repository*: https://github.com/apache/jena
d. *Documentation*: https://jena.apache.org/documentation/
e. *Publication*: B. McBride, "Jena: a semantic Web toolkit," in IEEE Internet Computing, vol. 6, no. 6, pp. 55-59, Nov.-Dec. 2002, doi: 10.1109/MIC.2002.1067737. Note: this describes an early version of the toolkit.

4. Atomic Data

a. *Description:* Atomic Data is a modular specification for sharing, modifying and modeling graph data. It combines the ease of use of JSON, the connectivity of RDF (linked data) and the reliability of type-safety.
b. *Homepage:* https://docs.atomicdata.dev/
c. *Code Repository:* https://docs.atomicdata.dev/
d. *Documentation:* -
e. *Publication:* -

5. CASPAR

a. *Description:* CASPAR is a modern ETL framework aspiring to harvest the fruits of Semantic Technologies and knowledge graphs. 90% of the world's current digital data was produced in the last two years, and the U.S. alone produces upward of 2.6 million gigabytes of internet data every minute. With so much information, it is no wonder many organizations are paralyzed with indecision wondering: What do we do with all this data? How do we manage it, use it and extract value from it? In the big data era, an organization's approach to mining and managing information could lead to starvation in abundance. Within this context, CASPAR provides a domain-agnostic tool for the automated retrieval and fusion of in-house structured data from disparate sources into domain-specific semantic models (ontologies), in order to enable the discovery of new knowledge and facilitate the extraction of actionable insights in a way that simulates the human reason.
b. *Homepage:* *https://caspar.catalink.eu/*
c. *Code Repository:*Tool is proprietary
d. *Documentation:* Tool is proprietary
e. *Publication:* Kontopoulos, Efstratios, Mitzias, Panagiotis, Avgerinakis, Konstantinos, Kosmides, Pavlos, Piperigkos, Nikos, Anagnostopoulos, Christos, Lalos, Aris S., Stagakis, Nikolaos, Arvanitis, Gerasimos, Zacharaki, Evangelia I., & Moustakas, Konstantinos. (2021). An Extensible Semantic Data Fusion Framework for Autonomous Vehicles. 5–11. https://doi.org/10.5281/zenodo.5560772

6. CENtree

a. *Description:* SciBite's ontology management platform CENtree provides a centralised, enterprise-ready resource for ontology management and transforms the experience of maintaining and releasing ontologies for research-led businesses.
b. *Homepage:* https://www.scibite.com/platform/centree/
c. *Code Repository:* Proprietary software
d. *Documentation:* https://www.scibite.com/platform/centree/
e. *Publication:* -

7. Chowlk
    a. *Description:* Chowlk is a web application that takes as input an ontology conceptualization created with diagrams.net and generates the OWL implementation. The conceptualization should follow the Chowlk visual notation which is also provided as a diagrams.net library to allow users to easily reuse the correct shapes to avoid problems during the transformation and save to time during the conceptualization.
    b. *Homepage*: https://chowlk.linkeddata.es/index.html
    c. *Code Repository*: https://github.com/oeg-upm/Chowlk
    d. *Documentation: https://chowlk.linkeddata.es/index.html/*
    e. *Publication*: https://openreview.net/pdf?id=u1Vp2y_QE1
8. crowd-tool
    a. *Description:* The intention of the tool is to involve domain experts and users in modelling tasks by adopting standard CDMLs, providing visual support for them, and logic-based reconstructions (in addition to OWL 2 serialisations) for knowledge engineers. The tool is fully integrated with a powerful logic-based reasoning server acting as a background inference engine. That reasoning is relative to the diagram's graphical syntax so that users will see the original model graphically completed with all the deductions that are expressed in the graphical language itself. crowd focuses on graphical modelling of CDMs (and ontologies) at the type level, and does not consider individuals. It is compliant with W3C standards by allowing the definition of global naming schemes as well as the export of specifications to OWL 2 to interoperate with other tools.
    b. *Homepage:* https://crowd-app.fi.uncoma.edu.ar/
    c. *Code Repository:* https://github.com/iamcrowd
    d. *Documentation:* https://github.com/iamcrowd
    e. *Publication:* German Braun, Giuliano Marinelli, Emiliano Rios Gavagnin, Laura Cecchi, Pablo Fillottrani.
    Web Interoperability for Ontology Development and Support with crowd 2.0
    Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence Demo Track. Pages 4980-4983. https://doi.org/10.24963/ijcai.2021/707
9. DBPedia Archivo
    a. *Description:* DBPedia Archivo is an online interface and augmented archive for all kinds of vocabularies. It automatically crawls for new ontologies, updates the already enlisted ontologies regularly and performs some useful tests to check the fitness of the vocabulary for the semantic web.
    b. *Homepage*: https://archivo.dbpedia.org/
    c. *Code Repository*: https://github.com/dbpedia/archivo
    d. *Documentation:* https://github.com/dbpedia/archivo
    e. *Publication*: https://svn.aksw.org/papers/2020/semantics_archivo/public.pdf

10. Dynaccurate

a. *Description:* Dynaccurate is a new Artificial Intelligence Web-based application that solves the issues caused by the constant evolution of terminologies, helping data-dependent organisations to optimise data exploitability. it provides:
1. Remapping Maintenance support for mapping synchronization between terminologies
Identifies terminology changes between database versions, such as medical and scientific databases; and
Detects and automatically adapts only the invalid mapping between terminologies.
2. Semantic Annotation
Maintenance support for semantic annotations
Identifies terminology changes between database versions;
Detects impacted annotations at document level; and
Automatically adapts only the outdated annotations.
3.Terminology Lookup
Terminology Lookup service
Provides a single point of access to all terminology versions; and
Provides information about the evolution of terminologies.
b. *Homepage:* *www.dynaccurate.com*
c. *Code Repository: The code is proprietary and not made public*
d. *Documentation:* *https://www.dynaccurate.com/api*
e. Publication: -

11. EMMOntoPy

a. *Description:* EMMOntoPy is a Python API for working with ontologies. It is developed on top of Owlready2 and contains a general module 'ontopy' and a specific module 'emmopy' for the Elemental Multiperspective Material Ontology (EMMO). It provides an intuitive representation of EMMO in Python. The general module 'ontopy' is ontology-agnostic, but has been developed in accordance with the needs of EMMO. It is available on GitHub and on PyPI under the open source BSD 3-Clause license. EMMOntoPy provides a natural representation of EMMO in Python. On top of that EMMOntoPy provides: i) Access by label (as well as by names, important since class and property names in EMMO are based on UUIDs). ii) Test suite for EMMO-based ontologies. iii) Generation of graphs. iv) Generation of documentation. v) Command-line tools: - emmocheck: Checks an ontology against EMMO conventions. - ontoversion: Prints ontology version number. - ontograph: Versatile tool for visualising (parts of) an ontology. - ontodoc: Documents an ontology. - ontoconvert: Converts between ontology formats.
b. *Homepage*: https://emmo-repo.github.io/EMMO-python
c. *Code Repository*: https://github.com/emmo-repo/EMMO-python
d. *Documentation:* https://emmo-repo.github.io/EMMO-python
e. *Publication*: -

12. F-uji
    a. *Description:* The F-UJI assessment is based on 16 out of 17 core FAIR object assessment metrics developed within FAIRsFAIR and each corresponding to a part or the whole of a FAIR principle. F-UJI adheres to existing web standards and PID resolution services best practices and utilises external registries and resources such as re3dataand Datacite APIs, SPDX License List, RDA Metadata Standards Catalog, and Linked Open Vocabularies (LOV).
    b. *Homepage*: https://www.f-uji.net/
    c. *Code Repository*: https://github.com/pangaea-data-publisher/fuji
    d. *Documentation:* https://www.f-uji.net/index.php?action=docs
    e. *Publication*: https://datascience.codata.org/articles/10.5334/dsj-2021-004/

13. FaCT++ reasoner
    a. *Description:* FaCT++ is a tableaux-based reasoner for expressive Description Logics (DL). It covers OWL and OWL 2 (lacks support for key constraints and some datatypes) DL-based ontology languages.
    b. *Homepage*: http://owl.cs.manchester.ac.uk/tools/fact/
    c. *Code Repository*: https://bitbucket.org/dtsarkov/factplusplus
    d. *Documentation:* https://code.google.com/archive/p/factplusplus/
    e. *Publication*: -

14. FOOPS!
    a. *Description:* The Findable, Accessible, Interoperable and Reusable principles provide a set of guidelines and best practices for describing and accessing research data to be reused by others. We believe that ontologies, on their own, should also adapt these principles to be reused by others. With this validator we aim to provide the means for researchers to assess whether a vocabulary (OWL or SKOS) conforms or not to the best practices for publishing ontologies on the Web.
    b. *Homepage*: https://w3id.org/foops/
    c. *Code Repository*: https://github.com/oeg-upm/fair_ontologies
    d. *Documentation:* https://foops.linkeddata.es/about.html
    e. *Publication*: https://foops.linkeddata.es/assets/iswc_2021_demo.pdf

15. GraphDB
    a. *Description:* OntoText GraphDB is a highly efficient and robust graph database with RDF and SPARQL support. It provides a number of reasoning profiles, namely, RDFS, RDFS+, OWL-Horst, OWL-Max, OWL2-QL and OWL2-RL. Besides the core functionality of a triplestore, GraphDB offers a workbench GUI for managing repositories, visualizing data for exploration, querying data with SPARQL and integrating tabular data through OntoRefine. Moreover, GraphDB allows the creation of SHACL repositories for which integrated data is validated against a given number of SHACL shapes.
    b. *Homepage*: https://graphdb.ontotext.com/

c. *Code Repository*: http://maven.ontotext.com/service/rest/repository/browse/owlim-releases/com/ontotext/graphdb/graphdb-se/
d. *Documentation:* https://graphdb.ontotext.com/documentation/standard/index.html
e. *Publication*: -

16. GRUFF

a. *Description:* Gruff is an interactive tool for browsing, querying, and editing triple-stores (which are also known as graph databases or repositories). It works with AllegroGraph from Franz Inc. and to a somewhat lesser extent with any SPARQL endpoint. Information can be browsed as visual graphs of nodes and link lines that are laid out automatically, and also as tables of properties for particular nodes. Queries can be written textually as SPARQL or Prolog code, or designed graphically as diagrams of nodes and link lines. The various views and features are tightly integrated to facilitate a smooth and rapid workflow.
b. *Homepage*: https://allegrograph.com/products/gruff/
c. *Code Repository*: -
d. *Documentation:* https://franz.com/agraph/support/documentation/current/gruff.html
e. *Publication*: https://allegrograph.com/products/gruff/

17. HermiT OWL Reasoner

a. *Description:* HermiT is reasoner for ontologies written using the Web Ontology Language (OWL). Given an OWL file, HermiT can determine whether or not the ontology is consistent, identify subsumption relationships between classes, and much more.
b. *Homepage*: http://www.hermit-reasoner.com/
c. *Code Repository*: https://code.google.com/archive/p/hermit-reasoner/
d. *Documentation:* -
e. *Publication*: http://www.hermit-reasoner.com/publications.html

18. Hets

a. *Description:* The heterogeneous tool set (Hets) is a parsing, static analysis and proof management tool combining various such tools for individual specification languages, thus providing a tool for heterogeneous multi-logic specification. Hets is based on a graph of logics and languages (formalized as so-called institutions), their tools, and their translations. This provides a clean semantics of heterogeneous specification, as well as a corresponding proof calculus. For proof management, the calculus of development graphs (known from other large-scale proof management systems) has been adapted to heterogeneous specification. Development graphs provide an overview of the (heterogeneous) specification module hierarchy and the current proof state, and thus may be used for monitoring the overall correctness of a heterogeneous development.
b. *Homepage*: http://rest.hets.eu/
c. *Code Repository*: https://github.com/spechub/Hets

d. *Documentation*:http://www.informatik.uni-bremen.de/agbkb/forschung/formal_methods/CoFI/hets/UserGuideCommonLogic.pdf
e. *Publication*: http://iks.cs.ovgu.de/~till/papers/hets-paper.pdf

19. Hozo Ontology Editor

a. *Description:* Hozo is a tool for building ontologies in a distributed environment. It consists of three major components: Ontology editor, Ontology server and Ontology manager. Onto Studio is specialized to support ontology-building. Users build and us e ontologies through the Ontology editor which has a friendly GUI. Ontology mana ger manages projects in which several ontologies are built collaboratively and in a distributed environment through the internet. It also manages versions of ontologies. Ontology server stores ontologies and instances and provides APIs for clients. Unlike OWL, its conceptual level is closer to that of human beings. OWL is a low level language which is good for an interlingua for ontology exchange. If one uses it as an ontology representation language directly, it would degrade user's understanding of ontology by restricting their idea to semantic-network/description-logics levels which are inappropriate and less expressive for understanding the content of the ontology. Its representation scheme is based on a frame structure. It helps users build ontologies with Roles in a natural way supported by the advanced theory of Roles. It represents nested structures of slots. That is, any slot can have its own slots. Inheritance information is explicit and is always accessible. Two ways of inheritance: one from super classes through is-a link and the other from class constraint. A user-friendly GUI is available. Version management is available with a useful function for displaying changes. Ontology building in a distributed environment over internet is supported. APIs are available for accessing ontologies and instances.
b. *Homepage:* https://www.hozo.jp/
c. *Code Repository:* -
d. *Documentation:* https://www.hozo.jp/HozoManual_en_20140317.pdf (English) https://www.hozo.jp/oe5_manual_jp.pdf (Japanese) https://www.hozo.jp/oe5_reference_jp.pdf (Quick reference document in Japanese)
e. *Publication: "Hozo: An Environment for Building/Using Ontologies Based on a Fundamental Consideration of "Role" and "Relationship"", Kouji Kozaki, Yoshinobu Kitamura, Mitsuru Ikeda, and Riichiro Mizoguchi in Proc. of the 13th International Conference Knowledge Engineering and Knowledge Management(EKAW2002), pp.213-218, Siguenza, Spain, October 1-4, 2002.*

20. IntelligentGraph

a. *Description:* The IntelligentGraph SAIL offers an extended capability for embedded calculation support within any RDF graph. When enabled as an RDF4J SAIL, it offers calculation functionality as part of the RDF4J engine, on top of any RDF4J repository, using a variety of script engines including JavaScript, Jython, and Groovy. It preserves the SPARQL capability of RDF4J, but with additional capabilities for calculation debugging and tracing. IntelligentGraph includes the PathQL query language. Just as

a spreadsheet cell calculation needs to access other cells, an IntelligentGraph calculation needs to access other nodes within the graph. Although full access to the underlying graph is available to any of the scripts, PathQL provides a succinct, and efficient method to access directly or indirectly related nodes. PathQL can either return just the contents of the referenced nodes, or the contents and the path to the referenced nodes.

b. *Homepage:* https://github.com/peterjohnlawrence/com.inova8.intelligentgraph/tree/master/olgap
c. *Code Repository:* https://github.com/peterjohnlawrence/com.inova8.intelligentgraph/tree/master/olgap
d. *Documentation:* https://inova8.com/bg_inova8.com/intelligentgraph-getting-started/
e. *Publication:* https://inova8.com/bg_inova8.com/blog/

21. JSON-LD Playground

a. *Description:*A tool to play around with JSON LD specs and how to play around with it. From the website: "Play around with JSON-LD markup by typing out some JSON below and seeing what gets generated from it at the bottom of the page"
b. *Homepage:* https://json-ld.org/playground/
c. *Code Repository:* -
d. *Documentation:* -
e. *Publication:* -

22. Konclude

a. *Description:* Konclude is a tableau-based reasoner for the Description Logic SROIQV(D), i.e.,SROIQ(D) + Nominal Schemas, and covers almost all features of the Web Ontology Language (OWL) 2 DL. Konclude is released as free software, i.e., you can redistribute it and/or modify it under the terms of version 3 of the GNU Lesser General Public License (LGPLv3) as published by the Free Software Foundation.
b. *Homepage:* https://www.derivo.de/en/produkte/konclude.html
c. *Code Repository:* https://github.com/konclude/Konclude
d. *Documentation:* https://github.com/konclude/Konclude
e. *Publication:* Steigmiller, Andreas, Thorsten Liebig, and Birte Glimm. "Konclude: system description." Journal of Web Semantics 27 (2014): 78-85.

23. LinkML

a. *Description:* LinkML, the Linked Data Modeling Language, is a flexible modeling language that allows you to author schemas in YAML that describe the structure of your data. LinkML provides a framework for working with and validating data in a variety of formats (JSON, RDF, TSV) provides generators for compiling LinkML schemas to other frameworks, including SHACL, ShEx, and JSON-LD contexts.
b. Homepage: https://linkml.io/
c. *Code Repository:* https://github.com/linkml/linkml/

d. *Documentation:* https://linkml.io/linkml/
e. *Publication:* -

24. Linked Open Vocabularies (LOV)

a. *Description:* Linked Open Vocabularies (LOV) is a high quality catalogue of reusable vocabularies for the description of data on the Web. The LOV initiative gathers and makes visible indicators that have not been previously harvested such as the interconnections between vocabularies, version history along with past and current referent (individual or organization). It allows serching within the vocabularies stored, ranking the ontology elements by match and popularity. It also offers a dump and a SPARQL endpoint.
b. *Homepage*: https://lov.linkeddata.es/
c. *Code Repository*: https://github.com/pyvandenbussche/lov
d. *Documentation*: https://lov.linkeddata.es/dataset/lov/about
e. *Publication*: https://doi.org/10.3233/SW-160213

25. LUPOSDATE

a. *Description:* LUPOSDATE is a Semantic Web database supporting
- the RDF query language SPARQL 1.1
- the rule language RIF BLD
- RDFS and OWL2RL
- parts of geosparql and stparql
- visual editing of SPARQL queries, RIF rules and RDF data
- visual representation of query execution plans (operator graphs), optimization and evaluation steps, and abstract syntax trees of parsed queries and rules

We support indexing for large-scale datasets (disk based) and for medium-scale datasets (memory based) as well as processing of (possibly infinite) data streams. There are also extensions for the cloud (P-LUPOSDATE: https://github.com/luposdate/P-LUPOSDATE) and for utilizing FPGAs for SPARQL query processing.
b. *Homepage:* https://github.com/luposdate/luposdate
c. *Code Repository:* https://github.com/luposdate/luposdate
d. *Documentation:* -
e. *Publication:* Sven Groppe, Data Management and Query Processing in Semantic Web Databases, Springer, https://doi.org/10.1007/978-3-642-19357-6

26. LUPOSDATE3000

a. *Description:* LUPOSDATE3000 is a multi-platform triple store answering SPARQL queries. It is designed for and runs in different kinds of environments like large-scale main memory servers, distributed environments like IoT and browser. For benchmarking IoT scenarios, LUPOSDATE is integrated into the SIMORA simulator: https://github.com/luposdate3000/SIMORA There is a web demo, which sonifies each processing step during SPARQL query evaluation (for local processing): https://www.ifis.uni-luebeck.de/~groppe/soundofdatabases/

b. *Homepage:* https://github.com/luposdate3000/luposdate3000
c. *Code Repository:* https://github.com/luposdate3000/luposdate3000
d. *Documentation:* -
e. *Publication:* Benjamin Warnke, Muhammad Waqas Rehan, Stefan Fischer, Sven Groppe: Flexible data partitioning schemes for parallel merge joins in semantic web queries in: Datenbanksysteme für Business, Technologie und Web (BTW), 19. Fachtagung des GI-Fachbereichs Datenbanken und Informationssysteme, Dresden, Germany, 2021, Gesellschaft für Informatik, Bonn, LNI, Vol.P-311, this publication received the label Results Reproduced, p.237-56,https://doi.org/10.18420/btw2021-12

27. Magic Draw Cameo

a. *Description:* Cameo Systems Modeler ™ is an industry leading cross-platform collaborative Model-Based Systems Engineering (MBSE) environment, which provides smart, robust, and intuitive tools to define, track, and visualize all aspects of systems in the most standard-compliant SysML models and diagrams.
b. *Homepage*: https://www.3ds.com/products-services/catia/products/no-magic/cameo-systems-modeler/
c. *Code Repository*: https://www.3ds.com/products-services/catia/products/no-magic/cameo-systems-modeler/
d. *Documentation:* https://docs.nomagic.com/display/NMDOC/Documentation
e. *Publication*: -

28. Mentor Editor

a. *Description:* Menthor Editor is a platform that enables developing ontologies faster. It focuses on conceptual modelling with OntoUML. It is an automatic validation and codification toolkit that encompasses a set of features, such as OWL generation, visual simulation, semantic anti-patterns detection and SBVR documentation.
b. *Homepage*: https://ontouml.org/ontouml/tooling/
c. *Code Repository*: https://github.com/MenthorTools/menthor-editor
d. *Documentation:* -
e. *Publication*: https://ris.utwente.nl/ws/portalfiles/portal/28605012/8e5afb734533d459657a5801a6212fdd72e6.pdf

29. MetaGraph 2.0

a. *Description:* An model-based tool for architecture design based on KARMA language and GOPPRRE ontology (IoF SE ontology). In this software, meta-models and models can be developed. Requirement can be defined using ReqIF specification. Moreover, design structure matrix can be used to define traceability between requimrents and architecture models. After architecture modeling, the IoF ontology can be generated from such models.
b. *Homepage*: http://www.zkhoneycomb.com/
c. *Code Repository*: https://gitee.com/zkhoneycomb/open-share
d. *Documentation:* https://gitee.com/zkhoneycomb/open-share

e. *Publication*: Yuanfu Li, Jinwei Chen, Zhenchao Hu, Huisheng Zhang, Jinzhi Lu & Dimitris Kiritsis (2021) Co-simulation of complex engineered systems enabled by a cognitive twin architecture, International Journal of Production Research, DOI: 10.1080/00207543.2021.1971318

30. OData2SPARQL

a. *Description:* OData2SPARQL: an OData service provider for any SPARQL endpoint. OData2SPARQL is a proxy server that provides OData V4 access to any triplestore that published a SPARQL interface. OData2SPARQL provides the following capabilities:
1. A RESTful API conforming to the OData standard via which applications can access the underlying RDF datasets, allowing full CRUD operations
2. A mapping of the OData metadata model to the underlying vocabulary and vice versa
3. A conversion of the incoming OData query into the corresponding SPARQL query, and the conversion of the SPARQL results back into the OData results, which can be in Atom/XML or JSON format.
4. A vocabulary (OData4sparql) that allows the mapping between OData and RDF/RDFS/OWL to be described.
b. *Homepage:* https://inova8.com/bg_inova8.com/offerings/odata2sparql/
c. *Code Repository:* https://github.com/peterjohnlawrence/com.inova8.odata2sparql.v4
d. *Documentation:* https://github.com/peterjohnlawrence/com.inova8.odata2sparql.v4
e. *Publication:* https://inova8.com/bg_inova8.com/offerings/odata2sparql/

31. O'FAIRE

a. *Description:* O'FAIRE (Ontology FAIRness Evaluator) is an independent automatic FAIRness assessment Web service developed in the ontology repository AgroPortal (based on NCBO technology). It consumes an ontology acronym (for example, BFO for the Basic Formal Ontology). It returns a JSON output that contains the FAIRness score obtained for each question aggregated by indicator and scores aggregated by principle and global score. Every obtained score is justified by a short justification sentence so that the user may be aware of why this score was obtained.
b. *Homepage*: https://github.com/agroportal/fairness
c. *Code Repository*: https://github.com/agroportal/fairness
d. *Documentation:* https://github.com/agroportal/fairness
e. *Publication*: https://hal.archives-ouvertes.fr/lirmm-03208544/

32. Ontofox

a. *Description:* Ontofox is a web-based Ontology tool that fetches ontology terms and axioms. Ontofox supports ontology reuse. It allows users to input terms, fetch selected properties, annotations, and certain classes of related terms from source ontologies and save the results using the RDF/XML serialization of the OWL. Ontofox follows and expands the MIREOT principle. Inspired by existing ontology modularization techniques, Ontofox also develops a new SPARQL-based ontology term extraction

algorithym that extracts terms related to a given set of signature terms. In addition, Ontofox provides an option to extract the hierarchy rooted at a specified ontology.

b. *Homepage*: http://ontofox.hegroup.org/
c. *Code Repository*: https://github.com/OntoZoo/ontofox
d. *Documentation:* http://ontofox.hegroup.org/faqs.php
e. *Publication*: https://bmcresnotes.biomedcentral.com/articles/10.1186/1756-0500-3-175

33. Ontology Lookup Service (OLS)

a. *Description:* The Ontology Lookup Service (OLS) is a repository for biomedical ontologies that aims to provide a single point of access to the latest ontology versions. You can browse the ontologies through the website as well as programmatically via the OLS API. OLS is developed and maintained by the Samples, Phenotypes and Ontologies Team (SPOT) at EMBL-EBI.
b. *Homepage:* https://www.ebi.ac.uk/ols
c. *Code Repository:* https://github.com/EBISPOT/OLS
d. *Documentation:* https://www.ebi.ac.uk/ols/docs/index
e. *Publication:* http://ceur-ws.org/Vol-1546/paper_29.pdf

34. OntOlogy Pitfall Scanner! (OOPS!)

a. *Description:* OOPS! (OntOology Pitfall Scanner! http://oops.linkeddata.es/) is an online application for ontology evaluation that operates independently of any ontology development platform and is available through a web application and a web service. OOPS! currently detects 33 pitfalls from a catalogue of 41, triplicating the number of error detection in comparison with other systems. It has been broadly accepted by a high number of users worldwide and has been used more tan 9000 times from around 60 different countries. OOPS! has been integrated with third-party software (namely, the mainstream ontology registry "Linked Open Vocabularies", Widoco, OnToology, OntoHub, etc.) and used in private enterprises for ontology development activities and training courses (for example SemanticArts and Raytheon in USA, inova8 in UK). OOPS! is also being used for educational purposes in universities like TU of Wien, University of Toronto, "Universidad Politécnica de Madrid" in bachelor and master studies and "Universitat Oberta de Catalunya".
b. *Homepage*: http://oops.linkeddata.es/
c. *Code Repository*: -
d. *Documentation:* http://oops.linkeddata.es/OOPSUserGuidev1.pdf
e. *Publication*: https://doi.org/10.4018/ijswis.2014040102

35. ontology-toolkit/onto-tool

a. *Description:* ontology-toolkit is a command line tool created to maintain version and dependency info in RDF ontologies.
b. *Homepage*: https://github.com/semanticarts/ontology-toolkit
c. *Code Repository*: https://github.com/semanticarts/ontology-toolkit
d. *Documentation:* https://pypi.org/project/onto-tool/
e. *Publication*: -

36. OnToology

a. *Description:* A system to automate part of the collaborative ontology development process. Given a repository with an owl file, OnToology will survey it and produce diagrams, a complete documentation and validation based on common pitfalls.
b. *Homepage*: http://ontoology.linkeddata.es/
c. *Code Repository*: https://github.com/OnToology/OnToology/
d. *Documentation:* http://ontoology.linkeddata.es/tutorial
e. *Publication*: https://www.sciencedirect.com/science/article/pii/S1570826818300465

37. Ontop

a. *Description:* Ontop is a Virtual Knowledge Graph system. It exposes the content of arbitrary relational databases as knowledge graphs. These graphs are virtual, which means that data remains in the data sources instead of being moved to another database. Ontop translates SPARQL queries expressed over the knowledge graphs into SQL queries executed by the relational data sources. It relies on R2RML mappings and can take advantage of lightweight ontologies.
b. *Homepage*: https://ontop-vkg.org/
c. *Code Repository*: https://github.com/ontop/ontop
d. *Documentation:* https://ontop-vkg.org/guide/getting-started.html
e. *Publication*: Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao. Ontop: Answering SPARQL Queries over Relational Databases. In: Semantic Web Journal 8.3 (2017), pp. 471–487.

38. Ontopanel

a. *Description:* The toolchain is a drawio-plugin which contains a set of tools for ontology development. It is inspired by Chowlk, but it is highly extensible and users can do their ontology modeling in Drawio. Users can import ontologies, display and search for entities as in Protege, convert to OWL files, and realize data mapping, up to now. More functionalities are under development.
b. *Homepage:* https://github.com/yuechenbam/yuechenbam.github.io
c. *Code Repository:* https://github.com/yuechenbam/yuechenbam.github.io
d. *Documentation:* -
e. *Publication:* -

39. OntoSpy

a. *Description:* Ontospy is a lightweight Python library and command line tool for working with vocabularies encoded in the RDF family of languages.
b. *Homepage*: http://lambdamusic.github.io/Ontospy/
c. *Code Repository*: https://github.com/lambdamusic/Ontospy
d. *Documentation:* http://lambdamusic.github.io/Ontospy/
e. *Publication*: -

40. OntoTrek

a. *Description:* OntoTrek is an OBOFoundry.org ontology terminology viewer that takes advantage of WebGL 3d graph rendering software. There are a number of ontologies programmed into the menu system, but one can enter a URL pointing to an ontology file (import files are ignored). Some ontologies are easy to load like AGRO and ECOCORE, and some big ones like CLO can only be rendered by first applying filters like "wireframe", no "labels", and limiting node depth. Some ontologies demonstrate upper level conformance to the Basic Formal Ontology (BFO, whose 34 nodes are big, sunny, and always in the same position).
b. *Homepage:* http://genepio.org/ontotrek/
c. *Code Repository:* https://github.com/cidgoh/Ontotrek
d. *Documentation:* -
e. *Publication:* http://ceur-ws.org/Vol-2518/paper-SHAPES1.pdf

41. OPCloud

a. *Description:* OPCloud is a Web-based collaborative software environment for model-based systems engineering (MBSE) used for creating conceptual models in Object-Process Methodology, OPM, ISO 19450:2005. OPCloud is a high-end, Cloud-based Software-as-a-System (SaaS) tool. OPM is an upper-level ontology and therefore it is fit for modeling domain ontologies. The tool is used as a basis for a Model-Based Systems Engineering edX Professional Certificate https://www.edx.org/professional-certificate/israelx-model-based-systems-engineering and is used by many academic institutes and industrial organizations.
b. *Homepage:* https://www.opcloud.tech/
c. *Code Repository:* https://sandbox.opm.technion.ac.il/
d. *Documentation:* https://technionmail-my.sharepoint.com/:b:/g/personal/hanank_technion_ac_il/EXElq0pB87FAt0RZsIVKM0UBOZVXVFP3PQeH0Opu_hggsA?e=WR2CLW
e. *Publication: 1. Hanan Kohen and Dov Dori, Designing and Developing OPCloud, an OPM-Based Collaborative Software Environment, in a Mixed Academic and Industrial Setting: An Experience Report. Academia Letters, 2021, doi:10.20935/al1918. https://doi.org/10.20935/AL1918*
2. Dov Dori, Ahmad Jbara, Natali Levi, and Niva Wengrowicz, Object-Process Methodology, OPM ISO 19450 – OPCloud and the Evolution of OPM Modeling Tools. Systems Engineering Letters, Project Performance International (PPI) SyEN 61, January 30, 2018. https://www.ppi-int.com/wp-content/uploads/2018/01/SyEN_61.pdf

42. ORMiE

a. *Description:* ORMiE (ORM inference Engine) is an extension of NORMA and NORMA Pro, which enable ORM fact-based conceptual modelling in Microsoft Visual Studio 2019. ORMiE activates automated reasoning over ORM models (including derivation rules) providing an interface where mistakes, redundancies or more in general new inferred constraints are shown. A dynamic translation of ORM models in NORMA or NORMA Pro into OWL2 is provided.

b. *Homepage:* https://gitlab.inf.unibz.it/franconi/ormie-release/
c. *Code Repository:* https://gitlab.inf.unibz.it/franconi/ormie-release/
d. *Documentation:* https://gitlab.inf.unibz.it/franconi/ormie-release/
e. *Publication: Alessandro Artale, Enrico Franconi, Rafael Peñaloza, Francesco Sportelli: A Decidable Very Expressive Description Logic for Databases. ISWC (1) 2017: 37-52 Francesco Sportelli, Enrico Franconi: A Formalisation and a Computational Characterisation of ORM Derivation Rules. OTM Conferences 2019: 678-694*

43. OWBO - Ontology White Board

a. *Description:* OWBO is a very basic tool to create skeleton ontologies as simple diagrams. It allows to create, modify and organise classes and relations that can then be further refined in a more powerful editor. OWBO is designed to allow easy sharing of ontology drafts and to be easily sharable itself, since it fits in a single html file.
b. *Homepage*: https://github.com/mdaquin/OWBO
c. *Code Repository*: https://github.com/mdaquin/OWBO
d. *Documentation:* -
e. *Publication*: -

44. OWL API

a. *Description:* The OWL API is a Java API for creating, manipulating and serialising OWL Ontologies.
b. *Homepage*: http://owlcs.github.io/owlapi/
c. *Code Repository*: https://github.com/owlcs/owlapi
d. *Documentation:* http://owlcs.github.io/owlapi/apidocs_5/index.html
e. *Publication*: The OWL API: A Java API for OWL ontologies. Semantic Web 2(1): 11-21 (2011) by Matthew Horrdige and Sean Bechhofer

45. owl-db-tools

a. *Description:* A tool for reading RDF and OWL into a graph database
b. *Homepage*: https://github.com/pdenno/owl-db-tools
c. *Code Repository:* https://github.com/pdenno/owl-db-tools
d. *Documentation:* https://github.com/pdenno/owl-db-tools
e. *Publication:* -

46. Owlready2

a. *Description:* Owlready2 is a package for ontology-oriented programming in Python. It can load OWL 2.0 ontologies as Python objects, modify them, save them, and perform reasoning via HermiT (included). Owlready2 allows a transparent access to OWL ontologies (contrary to usual Java-based API).
b. *Homepage*: https://owlready2.readthedocs.io/en/v0.35/
c. *Code Repository*: https://bitbucket.org/jibalamy/owlready2/src/master/
d. *Documentation:* https://owlready2.readthedocs.io/en/v0.35/
e. *Publication*: https://www.springer.com/de/book/9781484265512

47. PathQL

    a. *Description:* PathQL query language is included in IntelligentGraph. Just as a spreadsheet cell calculation needs to access other cells, an IntelligentGraph calculation needs to access other nodes within the graph. Although full access to the underlying graph is available to any of the scripts, PathQL provides a succinct, and efficient method to access directly or indirectly related nodes. PathQL can either return just the contents of the referenced nodes, or the contents and the path to the referenced nodes. PathQL can also be used standalone to query the IntelligentGraph-enabled RDF database. This supplements, rather than replaces, SPARQL and GraphQL, as it provides graph-path querying rather than graph-pattern querying capabilities to any IntelligentGraph-enabled RDF database.
    b. *Homepage:* -
    c. *Code Repository:* https://github.com/peterjohnlawrence/com.inova8.intelligentgraph/tree/master/olgap/src/main/java/com/inova8/pathql
    d. *Documentation:* https://inova8.com/bg_inova8.com/pathpatternql-intelligently-finding-knowledge-as-a-path-through-a-maze-of-facts/
    e. *Publication:* https://inova8.com/bg_inova8.com/pathpatternql-intelligently-finding-knowledge-as-a-path-through-a-maze-of-facts/

48. Pellet Reasoner

    a. *Description:* Pellet is a Java based OWL 2 reasoner. Pellet can be used with Jena or OWL-API libraries. Pellet provides functionality to check consistency of ontologies, compute the classification hierarchy, explain inferences, and answer SPARQL queries.
    b. *Homepage*: http://pellet.owldl.com/
    c. *Code Repository*: https://github.com/stardog-union/pellet
    d. *Documentation:* -
    e. *Publication*: Pellet: A Practical OWL-DL Reasoner by Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz

49. Pool Party

    a. *Description:* PoolParty Semantic Suite is middleware. PoolParty serves as the "glue" between customer databases and applications so that customer knowledge models can continuously evolve in a stable, interconnected environment. The PoolParty Basic Server is used for building and fine-tuning taxonomies with a low to medium level of scope and complexity.
    b. *Homepage*: https://www.poolparty.biz/
    c. *Code Repository*: -
    d. *Documentation:* https://help.poolparty.biz/?lang=en
    e. *Publication*: -

50. pronto

    a. *Description:* Pronto is a Python library to parse, browse, create, and export ontologies, supporting several ontology languages and formats. It implement the specifications of the Open Biomedical Ontologies 1.4 in the form of an safe high-level interface. At the moment,

it can parse OBO, OBO Graphs or OWL in RDF/XML format, ontologies on the local host or from an network location, and export ontologies to OBO or OBO Graphs (in JSON format).

b. *Homepage:* https://github.com/althonos/pronto
c. *Code Repository:* https://github.com/althonos/pronto
d. *Documentation:* https://pronto.readthedocs.io/en/stable/
e. *Publication:* Archived on Zenodo as a library: https://doi.org/10.5281/zenodo.595572

51. Protégé

a. *Description:* A free, open-source ontology editor and framework for building intelligent systems.
b. *Homepage*: https://protege.stanford.edu/
c. *Code Repository*: https://github.com/protegeproject/
d. *Documentation:* https://protege.stanford.edu/support.php#documentationSupport
e. *Publication*: Musen, M.A. The Protégé project: A look back and a look forward. AI Matters. Association of Computing Machinery Specific Interest Group in Artificial Intelligence, 1(4), June 2015. DOI: 10.1145/2557001.25757003.

52. RDF4J

a. *Description:* Eclipse RDF4J, which is formerly known as OpenRDF Sesame, is an open-source Java framework for storing, querying, and analyzing RDF data.
b. *Homepage*: https://rdf4j.org/
c. *Code Repository*: https://github.com/eclipse/rdf4j
d. *Documentation:* https://rdf4j.org/documentation/
e. *Publication*: -

53. RDFLib

a. *Description:* RDFLib is a pure Python package for working with RDF. RDFLib contains most things you need to work with RDF, including: parsers and serializers for RDF/XML, N3, NTriples, N-Quads, Turtle, TriX, Trig and JSON-LD; a Graph interface which can be backed by any one of a number of Store implementations; store implementations for in-memory, persistent on disk (Berkeley DB) and remote SPARQL endpoints; a SPARQL 1.1 implementation - supporting SPARQL 1.1 Queries and Update statements; and SPARQL function extension mechanisms.
b. *Homepage*: https://rdflib.readthedocs.io/en/stable/
c. *Code Repository*: https://github.com/RDFLib/rdflib
d. *Documentation:* https://rdflib.readthedocs.io/en/stable/
e. *Publication*: -

54. RDFox

a. *Description:* RDFox is the high-performance in-memory knowledge graph and semantic reasoner. Designed by leading academics at the University of Oxford, RDFox is the child of groundbreaking research that has culminated in the fasted market-ready knowledge graph, owing to its optimised in-memory approach. Coupled with its unmatched reasoning capabilities, it is the high-end competitor to the industry

standards. Supporting Datalog, OWL 2 RL, SPARQL, and validation with SHACL, RDFox complies with W3C standards, allowing easier adoption.

b. *Homepage:* https://www.oxfordsemantic.tech/
c. *Code Repository:* -
d. *Documentation:* https://docs.oxfordsemantic.tech/
e. *Publication:* https://www.oxfordsemantic.tech/product

55. Reasonable Ontology Templates

a. *Description:* Reasonable Ontology Templates (OTTR) is a language for representing ontology modelling patterns, and is designed to support interaction with OWL or RDF knowledge bases at a higher level of abstraction, using modelling patterns rather than OWL axioms or RDF triples. This includes: building knowledge bases by instantiating templates; communicating (presenting, transferring and visualising) the knowledge base as a set of template instances at different levels of abstraction; and securing and improving the quality and sustainability of the knowledge base via structural and semantic analysis of the templates used to construct the knowledge base.
b. *Homepage*: https://ottr.xyz/
c. *Code Repository*: https://gitlab.com/ottr
d. *Documentation:* https://ottr.xyz/
e. *Publication*: Martin G. Skjæveland, Daniel P. Lupp, Leif Harald Karlsen, and Henrik Forssell. Practical Ontology Pattern Instantiation, Discovery, and Maintenance with Reasonable Ontology Templates In: Vrandečić D. et al. (eds) The Semantic Web—ISWC 2018. ISWC 2018. LNCS vol 11136. Springer. 2018.

56. ROBOT

a. *Description:* ROBOT is a tool for working with Open Biomedical Ontologies. It can be used as a command-line tool or as a library for any language on the Java Virtual Machine.
b. *Homepage*: http://robot.obolibrary.org/
c. *Code Repository*: https://github.com/ontodev/robot
d. *Documentation:* http://robot.obolibrary.org/
e. *Publication*: R.C. Jackson, J.P. Balhoff, E. Douglass, N.L. Harris, C.J. Mungall, and J.A. Overton. ROBOT: A tool for automating ontology workflows. BMC Bioinformatics, vol. 20, July 2019.

57. Semantic MediaWiki

a. *Description:* Semantic MediaWiki (SMW) is a free, open-source ex-ten-sion to MediaWiki – the wiki soft-ware that pow-ers Wikipedia – that lets you store and query data with-in the wiki's pages. Semantic MediaWiki is also a full-fledged frame-work, in con-junc-tion with many spin-off extensions, that can turn a wiki into a pow-er-ful and flexible knowledge management system. All data created within Semantic MediaWiki can easily be exported or published via the Semantic Web, allowing other systems to use this data seamlessly.
b. *Homepage:* https://www.semantic-mediawiki.org/
c. *Code Repository:* https://github.com/SemanticMediaWiki/SemanticMediaWiki

d. *Documentation:* https://www.semantic-mediawiki.org/
e. *Publication:* Krötzsch, M., Vrandečić, D., Völkel, M., Haller, H., Studer, R., 2007. Semantic Wikipedia. Journal of Web Semantics 5, 251–261.

58. SimPhoNy

a. *Description:* SimPhoNy is an ontology-based framework aimed at enabling interoperability between different simulation and data management tools, with a focus on materials science.
b. *Homepage:* https://www.simphony-project.eu/
c. *Code Repository:* https://github.com/simphony/osp-core
d. *Documentation:* https://simphony.readthedocs.io/en/latest/index.html
e. *Publication:* "Development of an integrated multi-scale modelling environment for nanomaterials and systems by design"

59. Stardog

a. *Description:* Stardog is a commercial RDF database with support for SPARQL querying and OWL reasoning. It supports multiple reasoning profiles, namely RDFS and OWL2 QL, EL, RL, DL. Besides the core functionality of a triplestore, Stardog offers two graphical user interface solutions with Stardog studio and Stardog explorer. Studio makes it possible to easily manage different repositories in a Stardog database, and it provides basic tools to explore data in those repositories. Stardog explorer is a dedicated search engine on-top of a Stardog database.
b. *Homepage*: https://www.stardog.com/
c. *Code Repository*: -
d. *Documentation:* https://docs.stardog.com/
e. *Publication*: -

60. Terminology Harmonizer

a. *Description:* Greendecision's Terminology Harmonizer is a web based tool which allows communities to agree upon definitions for terms and their classification. The simplified user interface was built to allow experts in their own fields, which are not familiar with ontologies, to create the basis for a complete ontology. Agreed terms inside the tool can be migrated to a basic ontology file which can be further elaborated by experts in the filed.
b. *Homepage:* https://terminology-harmonizer.greendecision.eu
c. *Code Repository:* -
d. *Documentation:* https://terminology-harmonizer.greendecision.eu/help/introduction
e. *Publication:* -

61. TopBraid Composer Maestro Edition

a. *Description:* TopBraid Composer Maestro Edition (TBC-ME) combines semantic web modelling capabilities with data conversion options and an Integrated Development Environment (IDE) for implementing Knowledge Graph and Linked Data services. TBC-ME is used to develop ontology models, configure data source integration, and create semantic services and user interfaces.

b. *Homepage*: https://www.topquadrant.com/project/introducing_topbraid_composer/
c. *Code Repository*: -
d. *Documentation:* https://www.topquadrant.com/resources/products/docs/TBC-Getting-Started-Guide52.pdf
e. *Publication*: -

62. Virtuoso

a. *Description:* Enables the construction and deployment of Knowledge Graphs atop existing data exposed by APIs such as HTTP, ODBC, JDBC, ADO.NET, OLE DB, XMLA, and many service-specific APIs (e.g., LinkedIn, Crunchbase, Twitter, Facebook, etc.)
b. *Homepage*: https://virtuoso.openlinksw.com/
c. *Code Repository*: http://vos.openlinksw.com/owiki/wiki/VOS
d. *Documentation:* http://docs.openlinksw.com
e. *Publication*: http://infolab.kaist.ac.kr/publications/public/docs/DE2012Q1.pdf

63. Visual Studio Code

a. *Description:* For a lot of ontologies, writing them in a text editor such as Visual Studio Code is sufficient. Visual Studio Code has a lot of useful extension for RDF, OWL and SPARQL language support: Syntax highlighting and sometimes also validation.
b. *Homepage:* https://code.visualstudio.com/
c. *Code Repository:* https://github.com/microsoft/vscode
d. *Documentation:* https://code.visualstudio.com/docs
e. *Publication:* -

64. VocBench

a. *Description:* Originally released by the Food and Agriculture Organization of the United Nations and the Artificial Intelligence Research Group of the University of Rome Tor Vergata, VocBench is :a free web-based platform facilitating collaborative editing and management; designed to meet the needs of the semantic web. Furthermore, this tool: (1) manages multilingual controlled vocabularies such as ontologies, thesauri, authority lists, glossaries and lexicons; (2) allows users to maintain, validate and publish content through a flexible group management environment.
The new version, VocBench3 (v6.0), includes: (1) an extension point for accessing dataset repositories and registries, for searching datasets of interest, and for importing them; (2) specific connectors to repositories and registries for the provisioning of datasets; (3) advanced concept management; (4) support for the assignment of properties to multiple resources; (5) support for the assignment of multiple values when enriching a property; (6) support for bulk editing and bulk deleting; (7) improved visualisation of graphs and resources; (8) new implemented features, exploration modes, filters and operations on nodes in the graph visualisation; (9) improved Sheet2RDF editing with a more powerful wizard; (10) updated alphabetic index of the lexical entry list when a new 'LexicalEntry' is created; (11) new 'blacklisting' feature for projects with enabled 'validation', which adds all terms proposed in a rejected action to a blacklist pool; (12) improved user interface

of metadata management; (13) improved alignment validation with the integration of Genoma; (14) updated registration form and description in the welcome page; (15) option to add a preference for a list of languages to be shown on the resource description; (16) improved options in the 'Resource view' by enabling or disabling the 'add' and 'delete' operations and asserting all inferred statements of a described resource.

VocBench3 can help public administrations to maintain and publish their controlled vocabularies in an open and interoperable way. The development is managed by the Publications Office of the European Union.

b. *Homepage:* https://ec.europa.eu/isa2/solutions/vocbench3_en
c. *Code Repository:* https://bitbucket.org/art-uniroma2/vocbench3/src/master/
d. *Documentation:* Please have a look on the project page
e. *Publication:* Please have a look on the project page

65. WebVOWL

a. *Description:* WebVOWL is a web application for the interactive visualisation of ontologies. It implements the Visual Notation for OWL Ontologies (VOWL) by providing graphical depictions for elements of the Web Ontology Language (OWL) that are combined to a force-directed graph layout representing the ontology.
b. *Homepage*: http://vowl.visualdataweb.org/webvowl.html
c. *Code Repository*: https://github.com/VisualDataWeb/WebVOWL
d. *Documentation:* https://github.com/VisualDataWeb/WebVOWL#readme
e. *Publication*: http://www.semantic-web-journal.net/content/visualizing-ontologies-vowl-0

66. WIDOCO

a. *Description:* Wizard for documenting ontologies
b. *Homepage*: http://dgarijo.github.io/Widoco/doc/tutorial/
c. *Code Repository*: https://github.com/dgarijo/Widoco/
d. *Documentation:* http://dgarijo.github.io/Widoco/doc/tutorial/
e. *Publication*: https://dx.doi.org/10.1007/978-3-319-68204-4_9

67. WIMU

a. *Description:* One of the Semantic Web foundations is the possibility to dereference URIs to let applications negotiate their semantic content. However, this exploitation is often infeasible as the availability of such information depends on the reliability of networks, services, and human factors. Moreover, it has been shown that around 21% of the information published as Linked Open Data is available as data dumps only and 58% of endpoints are offline. To this end, we propose a Semantic Web service called Where is my URI?. Our service aims at indexing URIs and their use in order to let Linked Data consumers find the respective RDF data source, in case such information cannot be retrieved from the URI alone. We rank the corresponding datasets by following the rationale upon which a dataset contributes to the definition of a URI proportionally to the number of datatype triples. We finally show use-cases of applications that can immediately benefit from our simple yet useful service.

b. *Homepage:* http://wimu.aksw.org/
c. *Code Repository:* https://github.com/firmao/wimu
d. *Documentation:* https://dice-group.github.io/wimu/
e. *Publication:* https://link.springer.com/chapter/10.1007/978-3-319-93417-4_43

68. yEd Graph editor

a. *Description:* yEd is the graph editor tool with variety of templates. It is flexible and it has an ontology template for classes and relations. it is very easy to use. Ontology template can also be customized. url: https://www.yworks.com/products/yed/download. It also has SDK for interface with other applications. It has Graphity applications from Confluence.
b. *Homepage:* https://www.yworks.com/products/yed/download
c. *Code Repository:* There are several GitHub repositories
d. *Documentation*: https://docs.yworks.com/
e. *Publication:* Falco R., Gangemi A., Peroni S., Shotton D., Vitali F. (2014) Modelling OWL Ontologies with Graffoo. In: Presutti V., Blomqvist E., Troncy R., Sack H., Papadakis I., Tordai A. (eds) The Semantic Web: ESWC 2014 Satellite Events. ESWC 2014. Lecture Notes in Computer Science, vol 8798. Springer, Cham. https://doi-org.proxy.library.ohio.edu/10.1007/978-3-319-11955-7_42

4.3 Requirement Analysis

In this section, we report on the results of the second part of the questionnaire as described in Section 8 landscape of ontology engineering tools w.r.t. requirements arising in application domains.

We present the results in 19 subsections that correspond to the 19 requirements on tools that were originally compiled in the report on "Requirements on ontology tools and ontologies and criteria for selection of further cases". There, each requirement is given a unique identifier that follows the format **CRQ_[CATEGORY]_[SERIAL NUMBER]**. CRQ is used as an abbreviation for "Common requirement". Since we are only interested in requirements for tools, we are only concerned with CRQ of the category **T**, i.e., "tools". We repeat these 19 requirements and their descriptions in Table 3 for convenience.

Table 3: Requirements on tools as specified by the report on "Requirements on ontology tools and ontologies and criteria for selection of further cases"

UID	Title	Description
CRQ_T_01	Collaboration of multiple stakeholders	The ontology development tool should allow different stakeholders to work simultaneously.
CRQ_T_02	Visualisation	The tools shall support visualisation of ontologies.

CRQ_T_03	Debugging	The tools shall support debugging.
CRQ_T_04	Validation	The tool shall support Validation.
CRQ_T_05	Quality assurance and analytics	Support quality assurance in domain operations and ontology development
CRQ_T_06	Develop REST APIs to access data	The tools should support easy interaction with ontologies via REST APIs instead of SPARQL queries for retrieving data.
CRQ_T_07	Support for OWL	The tool for edition and maintenance of the ontologies shall be able to edit OWL files.
CRQ_T_08	Ontologies import	The tool for edition and maintenance of the ontologies shall be able to import and reuse existing ontologies.
CRQ_T_09	User friendly	The tool for edition and maintenance of the ontologies should be easy to use.
CRQ_T_10	Connected ontologies	The tools for ontology should enable/ support compatibility of different domains (e.g. processes and materials), since different ontologies might be needed.
CRQ_T_11	Tools integration	Integrated tool shall be provided to support initial brainstorming on models of concepts relevant for the domain and applications, to enhance transition from initial ideas to standard tools (e.g. Protégé).

CRQ_T_12	Modularisation	The tools should facilitate ontologies modularisation.
CRQ_T_13	Tools for searching ontologies	The tools should allow for ontology search (find ontologies, entities, definitions, etc., e.g. finding existing ontologies to fit an application, based on the initial model of needed concepts).
CRQ_T_14	Tools for reusability of ontologies	Quality ontologies to be re-used should be guaranteed.
CRQ_T_15	Tools for correlation among ontologies	Tools shall be provided to support establishment of relation of concepts from diverse ontologies.
CRQ_T_16	Tools for deployment and generation of documentation	Tools should be provided to support effective deployment of ontologies, including effective generation of documentation etc
CRQ_T_17	Tool for conceptualisation	The tool should support the conceptualisation phase of ontology development.
CRQ_T_18	Tool for guide ontology reusability	The tool should guide the user and suggest whether existed ontologies can be reused or new are required
CRQ_T_19	Tool GUI labels consistency and understandability	The GUI labels of the tool should comply with terminology used in ontologies to be understandable by ontologists, but also clear and understandable by non-ontologists.

As part of the questionnaire, we asked respondents to rate each tool. Respondents were asked to indicate how well the tool met each requirement, selecting:

- 1: no support,

- 2: low support,
- 3: some support,
- 4: good support,
- 5: full support, or
- 6: not applicable.

Within the following subsections, we will count the scores of the tools and analyse those with a score of 4 or 5, that is, those with a better level of requirement support. These tools and the reasons for selected scores are described separately in each subsection, and will be further discussed in Section 5 Discussion.

4.3.1 CRQ_T_01 - Collaboration of Multiple Stakeholders

4.3.1.1 Quantitative Results

Figure 3 shows the responses for the 5-point rating scale w.r.t. CRQ_T_01. We observe that approximately one third of the tools do not seem to provide support for different stakeholders to work simultaneously. However, there are a few tools that provide support in this direction. The following 32 tools received a score of 4 or 5:

- AllegroGraph
- Atomic Data
- CASPAR
- CENtree
- Chowlk
- crowd-tool
- DBPedia Archivo
- Dynaccurate
- FaCT++ reasoner
- GraphDB
- IntelligentGraph
- JSON-LD Playground
- Konclude
- LinkML
- LUPOSDATE
- LUPOSDATE3000
- OData2SPARQL
- OnToology
- Ontop
- Ontopanel
- OPCloud
- ORMiE
- OWLAPI
- PathQL
- Pool Party
- RDFox

- Semantic MediaWiki
- SimPhoNy
- Stardog
- Terminology Harmonizer
- VocBench
- WIMU

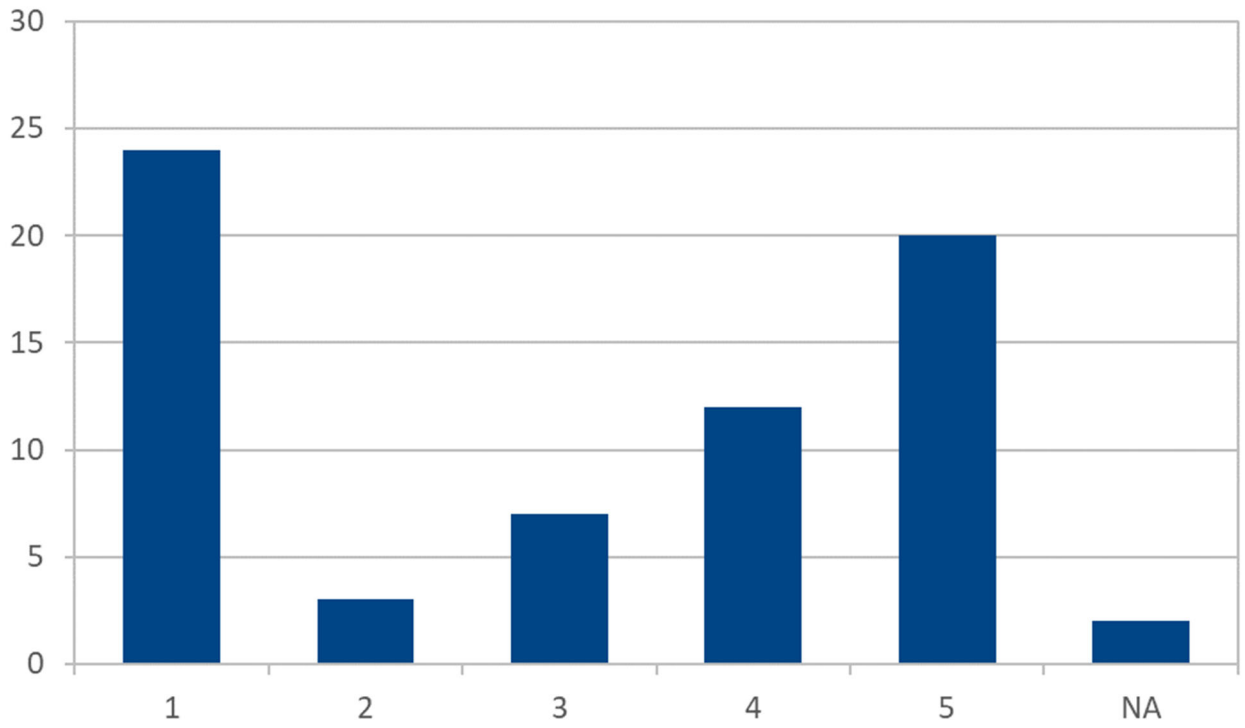

Figure 3: Responses for CRQ_T_01 - Collaboration of Multiple Stakeholders.

4.3.1.2 Qualitative Results

The provided justifications for given scores reveal that a high score does not necessarily mean that a tool provides direct support for collaboration between multiple stakeholders. Reviewers often justified a high score, i.e., 4 or 5, if a tool can be used in the context of workflows for collaboration.

For example, FACT++ reasoner received a score of 4. However, the reviewer justified this score as follows:

> *"This is not a collaboration tool. However, implementing this into continuous integration / continuous deployment (CI / CD) workflows facilitates collaboration through verification and unification."*

Similarly, GraphDB received a score of 5 but the reviewer qualified this score as follows:

> *"GraphDB offers the possibility to define users and their access to certain repositories, allowing multiple stakeholders to work simultaneously on their repositories. GraphDB is however no tool to simultaneously work on editing and drafting ontologies."*

Furthermore, some tools have been given a high score not because they facilitate such collaboration per se, but because they are either hosted or integrated in a system that allows for collaboration. Examples of this case are:

- LUPOSDATE3000 (score 4) and LUPOSDATE (score 4) with the justification

  "*The tool is open source and freely available via Github allowing the usual collaboration possibilities of Github.*"

- IntelligentGraph (score 4) with the justification

  "Deployed within GitHub, collaborators can either create a fork , or contact inova8 to become full collaborators"

- Ontopanel (score 4) with the justification

  "*It is based on drawio, and drawio has the collaboration functionality.*"

- LinkML (score 5) with the justification

  "Source files can be managed in github which allows multiple editors"

- Konclude (score 4) with the justification

  "Konclude supports multiple requests over the OWLLink interface."

4.3.2 CRQ_T_02 - Visualisation

4.3.2.1 Quantitative Results

Figure 4 shows the responses for the 5-point rating scale w.r.t. CRQ_T_02. While most tools do not support the visualisation of ontologies, there seem to be a fair number of tools to provide functionality in this direction. The following 24 tools received a score of 4 or 5:

- AllegroGraph
- CENtree
- Chowlk
- EMMOntoPy
- GRUFF
- Hozo Ontology Editor
- JSON-LD Playground
- Menthor Editor
- OFAIR
- Ontology Lookup Service (OLS)
- ontology-toolkit/onto-tool
- OnToology
- OntoTrek
- OPCloud

- ORMiE
- Pool Party
- RDFox
- Stardog
- TopBraid Composer Maestro Edition
- Virtuoso
- VocBench
- WebVOWL
- WIDOCO
- yEd Graph editor

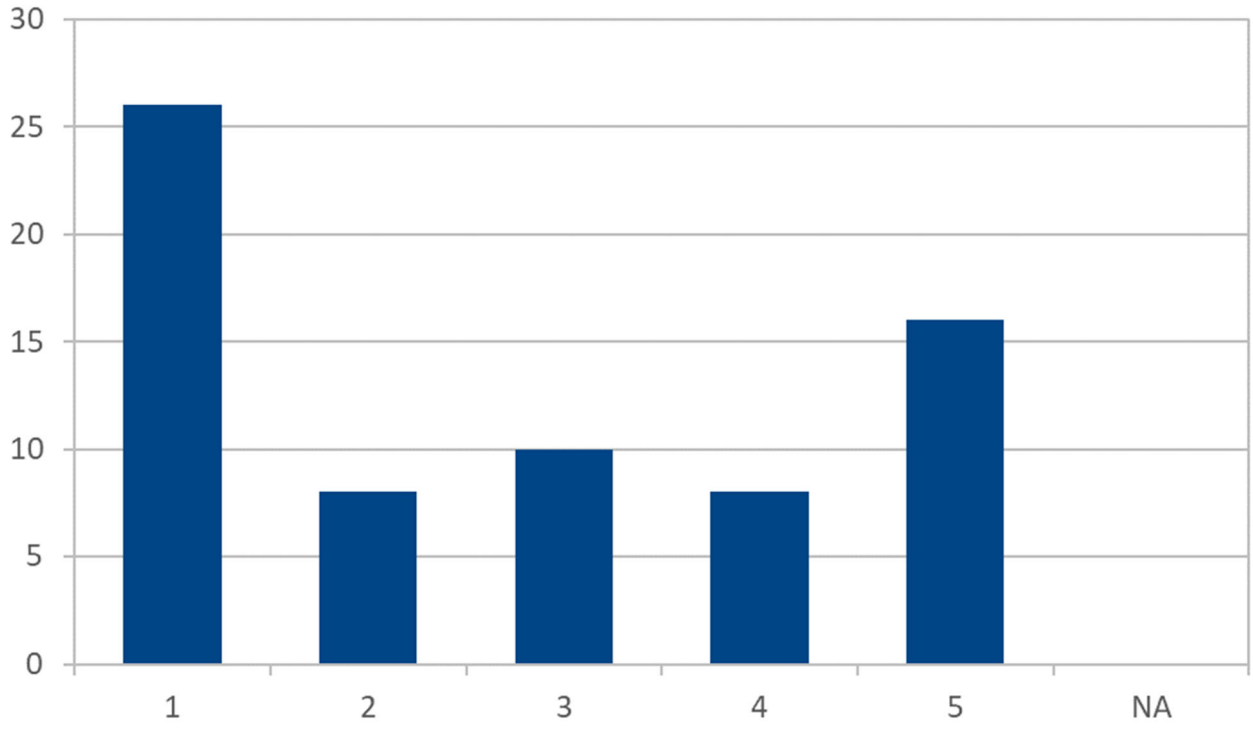

Figure 4: Responses for CRQ_T_02 - Visualisation.

4.3.2.2 Qualitative Results

The provided justifications for given scores suggest that many tools allow to visualise some aspects of an ontology but not necessarily the ontology as a whole. The tool Chowlk, for example, received a score of 5 but only seems to provide visualisation functionality for a conceptual view of an ontology:

> "The tool provides a library to develop conceptualizations which allows the visualization of the model."

Another example is the tool Stardog (with a score of 4) that seems to not support the visualisation of OWL properties:

> *"Ontologies can be visualized with Stardog studio. However, visualization of OWL properties as known from VOWL are not fully supported."*

Otherwise, we observe that many reviewers have given a score of 1 instead of selecting the last option "not applicable" in case visualisaiton is out of a tool's scope. Examples include Konclude, WIMU, PathQL, LinkML, and CASPAR.

4.3.3 CRQ_T_03 - Debugging

4.3.3.1 Quantitative Results

Figure 5 shows the responses for the 5-point rating scale w.r.t. CRQ_T_03. Even though score 1, i.e., no support, was given most often, we observe that more than two-thirds of tools seem to provide at least some support for debugging. The following 26 tools received a score of 4 or 5:

- Chowlk
- DBPedia Archivo
- Dynaccurate
- EMMOntoPy
- FaCT++ reasoner
- FOOPS!
- F-uji
- IntelligentGraph
- JSON-LD Playground
- Konclude
- LUPOSDATE3000
- OData2SPARQL
- OFAIR
- OntOlogy Pitfall Scanner! (OOPS!)
- OPCloud
- ORMiE
- PathQL
- Pool Party
- RDF4J
- RDFox
- ROBOT
- SimPhoNy
- Virtuoso
- Visual Studio Code
- VocBench
- WIMU

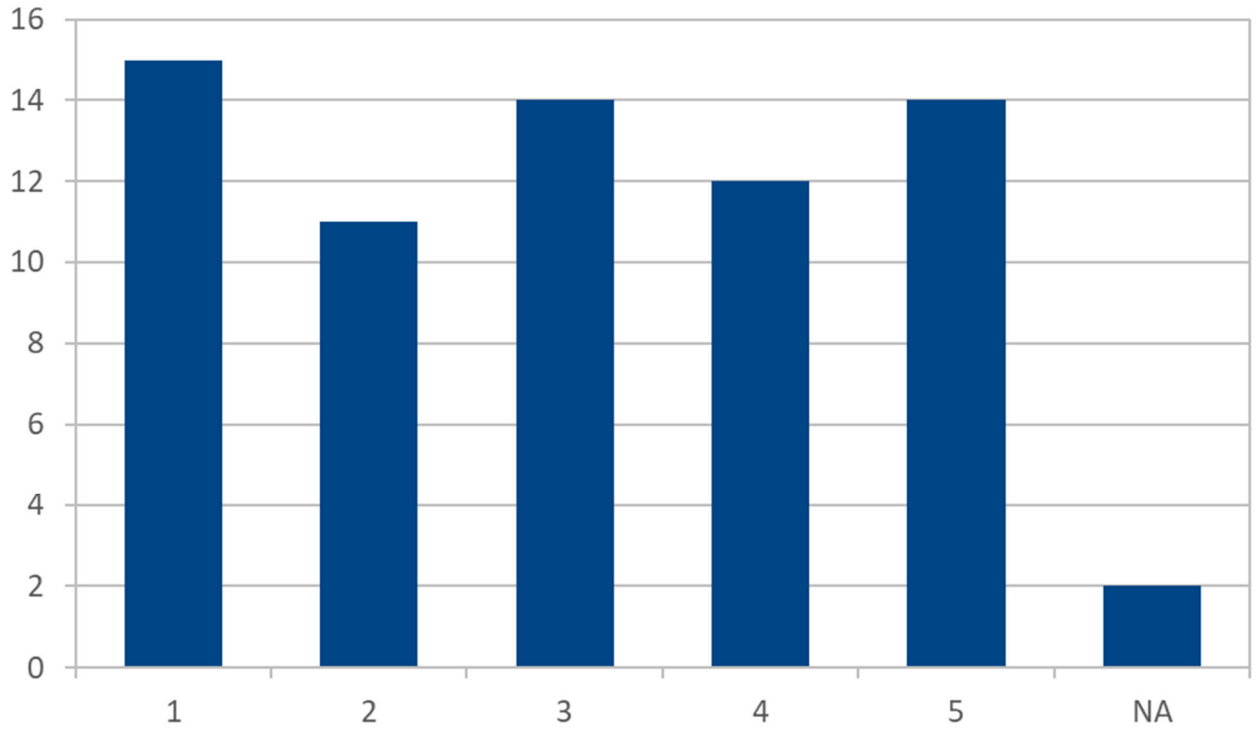

Figure 5: Responses for CRQ_T_03 - Debugging.

4.3.3.2 Qualitative Results

The almost uniform distribution of scores for CRQ_T_03 seems to suggest, that many tools provide at least some functionality for debugging. The provided justifications for given scores reveal that different tools provide different ways of detecting different kinds of errors. For example, different tools provide error messages for syntax errors, violations of FAIR principles, violations of (tool-specific) conventions, (potential) design errors, unsatisfiable classes, and inconsistent ontologies. Otherwise, some tools provide metrics and analytics to help a user to pinpoint potential faults in an ontology.

4.3.4 CRQ_T_04 - Validation

4.3.4.1 Quantitative Results

Figure 6 shows the responses for the 5-point rating scale w.r.t. CRQ_T_04. It appears that about one third of the reviewed tools provide some functionality, i.e., they received a score of 3 or higher, for validating the contents of an ontology. The following 29 tools received a score of 4 or 5:

- Apache Jena
- Atomic Data
- CASPAR
- DBPedia Archivo
- Dynaccurate

- FaCT++ reasoner
- FOOPS!
- F-uji
- GraphDB
- Hozo Ontology Editor
- Linked Open Vocabularies (LOV)
- LinkML
- Menthor Editor
- MetaGraph 2.0
- OData2SPARQL
- OFAIR
- OntOlogy Pitfall Scanner! (OOPS!)
- OnToology
- OPCloud
- ORMiE
- Pool Party
- RDF4J
- RDFox
- ROBOT
- SimPhoNy
- Stardog
- TopBraid Composer Maestro Edition
- Virtuoso
- VocBench

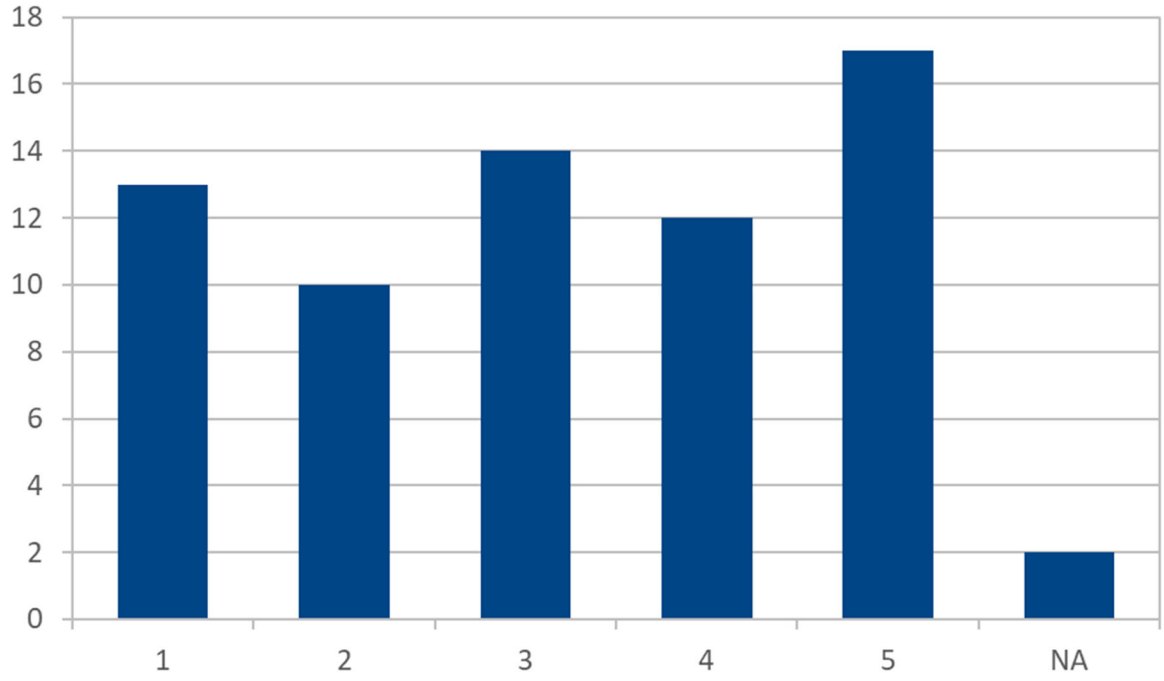

Figure 6: Responses for CRQ_T_04 - Validation.

4.3.4.2 Qualitative Results

The justifications for the given scores reveal that different reviewers interpret the term "validation" differently in the context of ontology engineering. Several reviewers justified a high score, if a tool provides support for SHACL. Some reviewers argued that reasoning support can be seen as a simple form of validation to ensure that an ontology is not inconsistent and does not contain unsatisfiable classes. Other reviewers justified a high score if a tool provides features to identify anti-patterns or incomplete instances of a data model. Yet, other reviewers justified a high score if a tool provides features to facilitate the adherence to FAIR principles.

4.3.5 CRQ_T_05 - Quality Assurance and Analytics

4.3.5.1 Quantitative Results

Figure 7 shows the responses for the 5-point rating scale w.r.t. CRQ_T_05. The data suggest that nearly half of the surveyed tools do not provide any features for quality assurance or analytics. However, there are a few tools that provide some functionality in this direction. The following 11 tools have received a score of 4 or 5:

- DBPedia Archivo
- Dynaccurate
- FaCT++ reasoner
- FOOPS!
- OFAIR
- OntOlogy Pitfall Scanner! (OOPS!)
- OnToology
- Pool Party
- RDF4J
- TopBraid Composer Maestro Edition
- VocBench

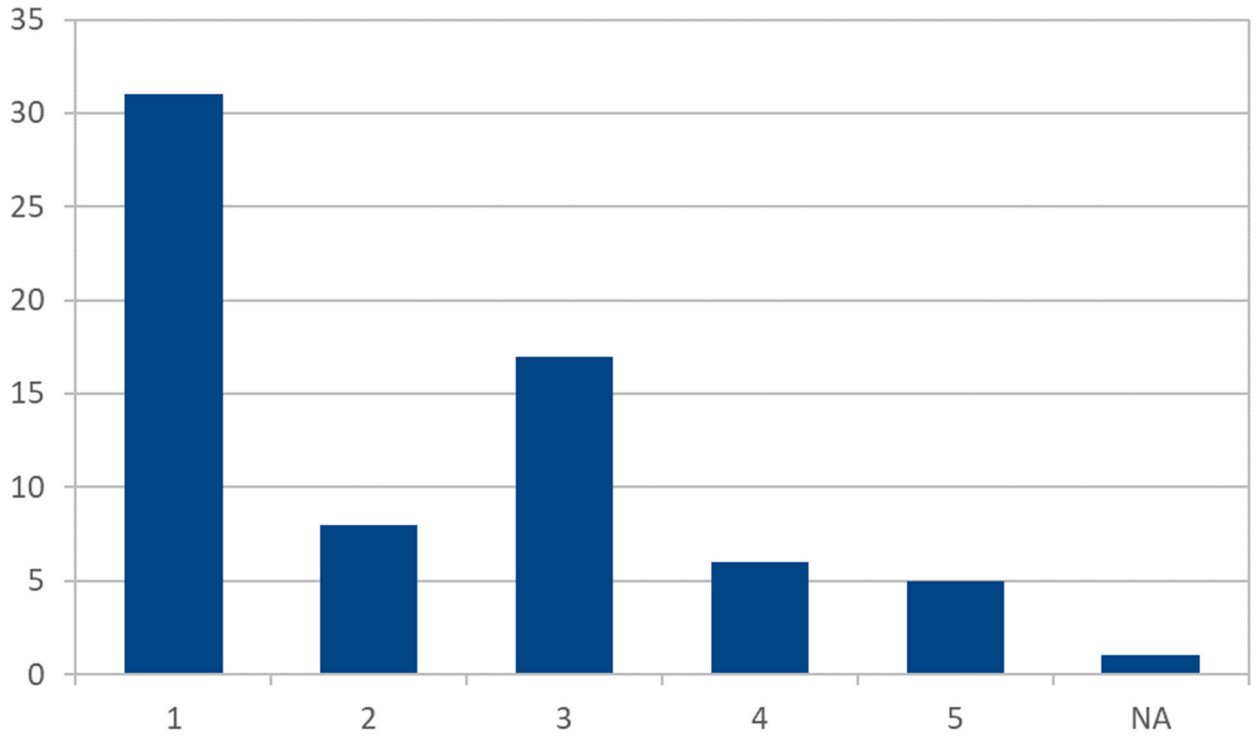

Figure 7: Responses for CRQ_T_05 - Quality assurance and analytics.

4.3.5.2 Qualitative Results

The justifications for the given scores reveal that different reviewers interpret the term "quality assurance" differently in the context of ontology engineering. Most scores of 4 or 5 were justified on the basis of support for data validation, e.g., via support for SHACL, or FAIR principles. Yet, other reviewers gave a lower score for tools that provide support for data validation. Similarly, reasoning support was considered to be a way of quality control "in the sense that [an ontology] is logically sound" that justified a score of 5 in the case of Fact++ reasoner whereas other reasoners, e.g., Pellet and HermiT, have received a score of 1. Otherwise, there are a few cases in which a tool has been assigned a score of 3 even though the corresponding justification points out that the requirement is either not applicable or not supported by the tool as for the case of LinkML, Visual Studio Code, and Hozo Ontology Editor,

4.3.6 CRQ_T_06 - Support for Mechanisms to Access Data

4.3.6.1 Quantitative Results

Figure 8 shows the responses for the 5-point rating scale w.r.t. CRQ_T_06. The data suggests that about three-quarters of the tools reviewed in this survey support some mechanism to access data in an ontology. The following 34 tools received a score of 4 or 5:

- AllegroGraph
- Atomic Data

- CASPAR
- CENtree
- DBPedia Archivo
- Dynaccurate
- GraphDB
- Hozo Ontology Editor
- IntelligentGraph
- JSON-LD Playground
- Konclude
- Linked Open Vocabularies (LOV)
- LinkML
- OData2SPARQL
- OFAIR
- Ontofox
- OntOlogy Pitfall Scanner! (OOPS!)
- Ontop
- Ontopanel
- OntoSpy
- OPCloud
- ORMiE
- OWLAPI
- owl-db-tools
- Owlready2
- PathQL
- Pellet Reasoner
- Pool Party
- RDFox
- ROBOT
- SimPhoNy
- Stardog
- TopBraid Composer Maestro Edition
- Virtuoso

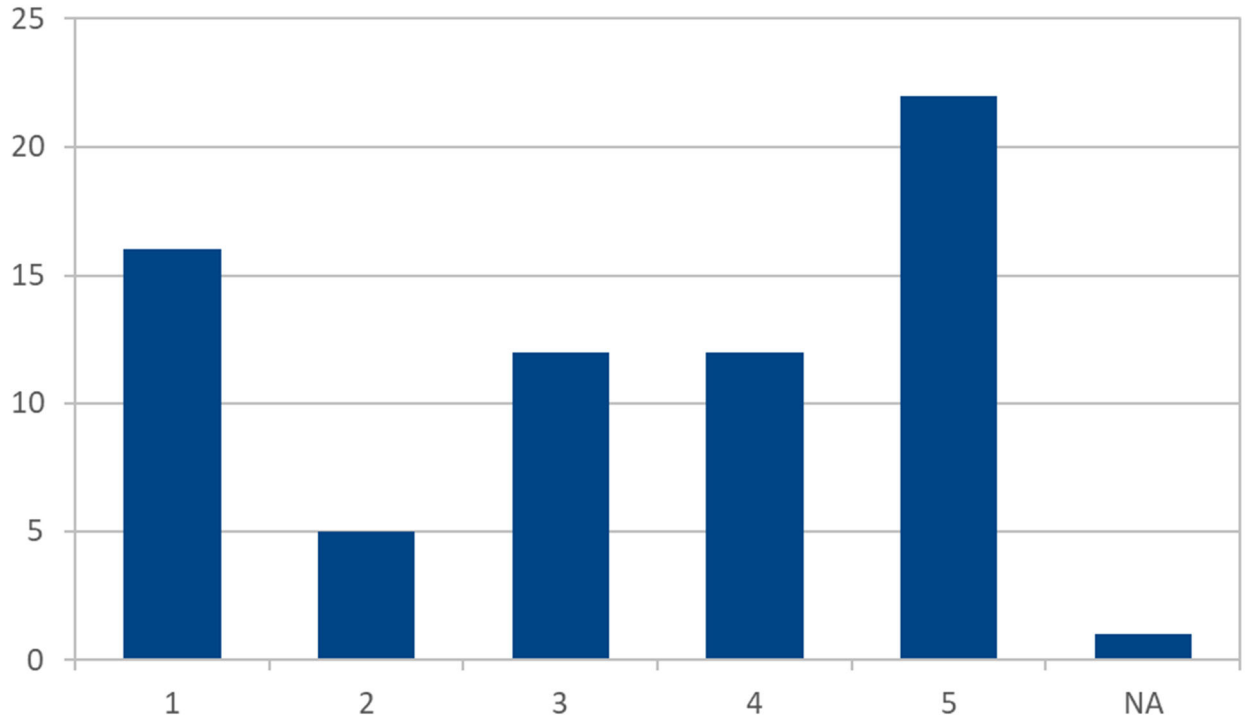

Figure 8: Responses for CRQ_T_06 - Support different mechanisms to access data

4.3.6.2 Qualitative Results

Justifications for scores of 4 or 5 have been given primarily on the grounds of support for SPARQL queries. Otherwise, integrated access to APIs has been used as another justification for scores of 4 or 5 for a few tools.

4.3.7 CRQ_T_07 - Support for OWL

4.3.7.1 Quantitative Results

Figure 9 shows the responses for the 5-point rating scale w.r.t. CRQ_T_07. The data suggests that more than a quarter of the reviewed tools do not support OWL ontologies. The following 26 tools have received a score of 4 or 5:

- Atomic Data
- CENtree
- Chowlk
- Dynaccurate
- EMMOntoPy
- Hozo Ontology Editor
- JSON-LD Playground
- LinkML
- Menthor Editor
- OData2SPARQL

- OFAIR
- ontology-toolkit/onto-tool
- Ontopanel
- ORMiE
- OWLAPI
- Owlready2
- PathQL
- Pool Party
- Protégé
- RDF4J
- RDFox
- SimPhoNy
- TopBraid Composer Maestro Edition
- Visual Studio Code
- VocBench
- WebVOWL

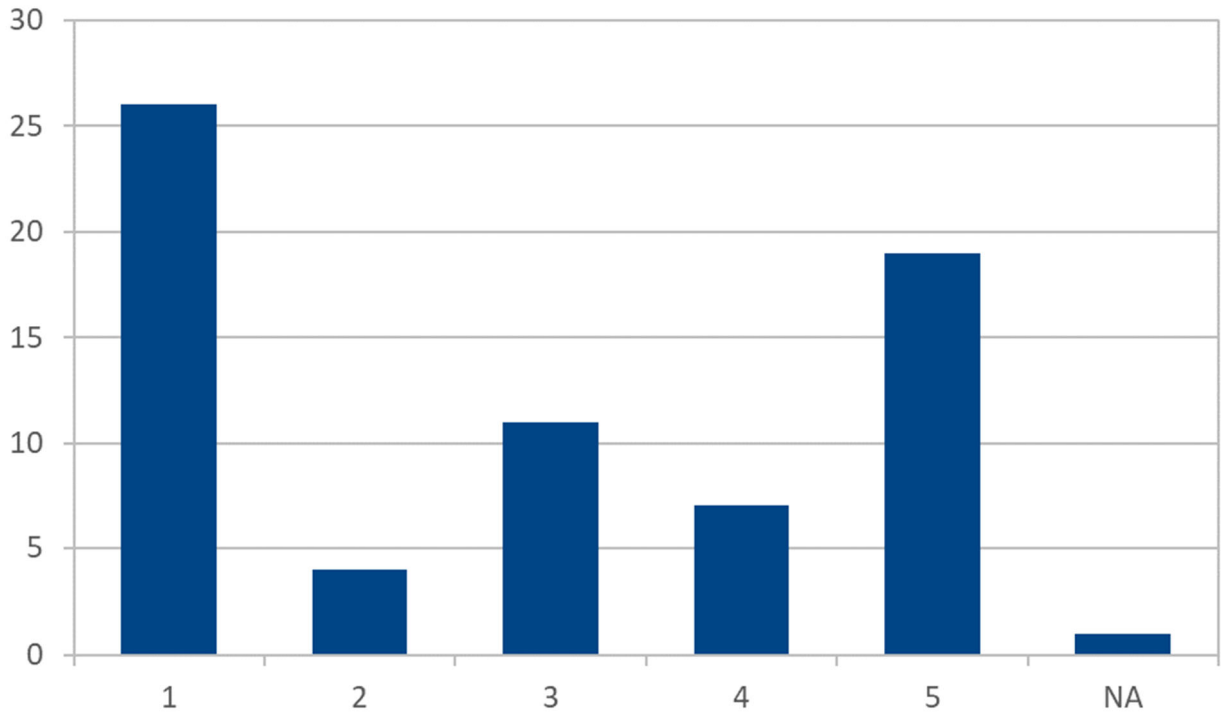

Figure 9: Responses for CRQ_T_07 - Support for OWL.

4.3.7.2 Qualitative Results

Many tools have received a score of 3 or lower with the justification that they either only support a subset of OWL or because they do not provide features for editing an OWL ontology as required by

CRQ_T_07. Yet, in other cases reviewers have justified a high score of 5 for systems that only provide support for RDF but not OWL.

4.3.8 CRQ_T_08 - Ontologies Import

4.3.8.1 Quantitative Results

Figure 10 shows the responses for the 5-point rating scale w.r.t. CRQ_T_08. The data suggests that two thirds of the reviewed tools support ontology imports. The following tools 32 received a score of 4 or 5:

- AllegroGraph
- Atomic Data
- CASPAR
- CENtree
- Dynaccurate
- EMMOntoPy
- GraphDB
- Hozo Ontology Editor
- Linked Open Vocabularies (LOV)
- LinkML
- Ontofox
- ontology-toolkit/onto-tool
- OnToology
- Ontop
- Ontopanel
- OntoSpy
- OWLAPI
- PathQL
- Pool Party
- pronto
- Protégé
- RDFox
- ROBOT
- Semantic MediaWiki
- SimPhoNy
- Stardog
- Terminology Harmonizer
- TopBraid Composer Maestro Edition
- Virtuoso
- VocBench
- WebVOWL
- WIDOCO

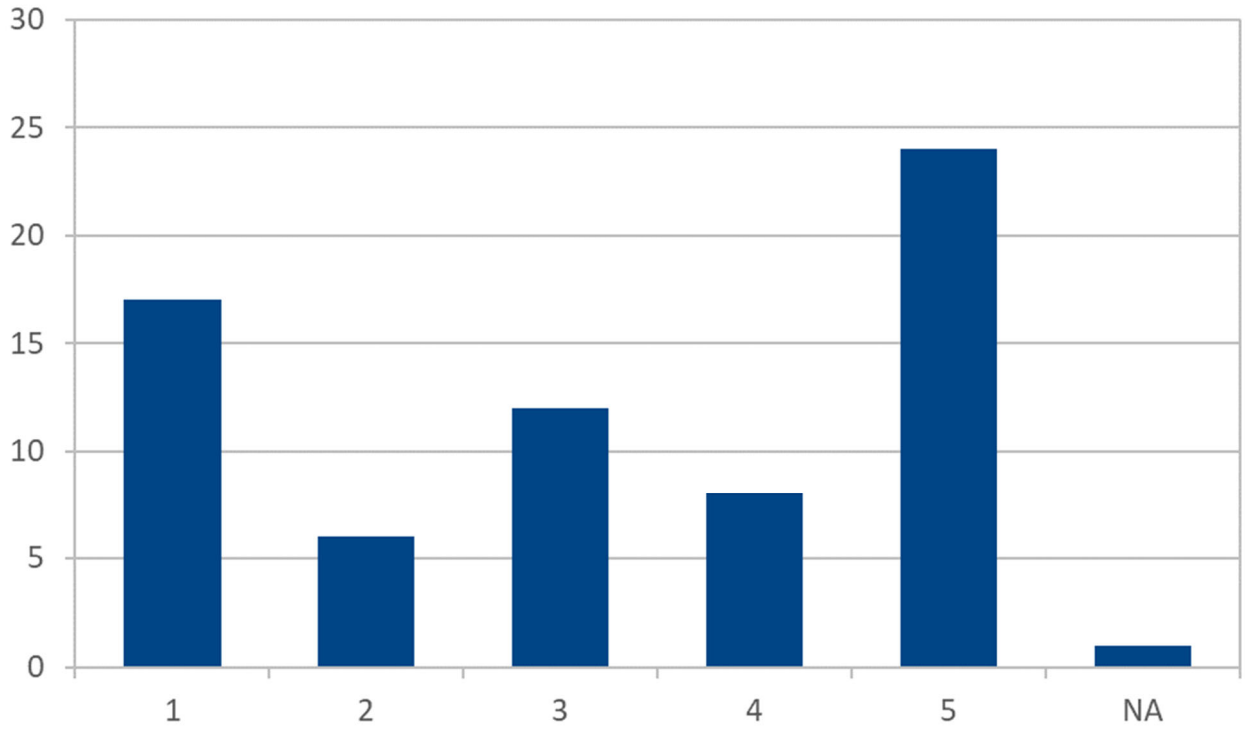

Figure 10: Responses for CRQ_T_08 - Ontologies Import

4.3.8.2 Qualitative Results

The provided justifications for the given scores reveal that many tools with a score of 3 or higher support ontology import but do not provide any features for ontology reuse otherwise.

4.3.9 CRQ_T_09 - User Friendly

4.3.9.1 Quantitative Results

Figure 11 shows the responses for the 5-point rating scale w.r.t. CRQ_T_09. The data suggests that most tools are perceived as user friendly. Only the following 27 tools received a score of 5:

- CENtree
- Dynaccurate
- IntelligentGraph
- JSON-LD Playground
- LinkML
- Linked Open Vocabularies (LOV)
- Magic Draw Cameo
- OData2SPARQL
- O'FAIRE
- Ontofox
- OOPS!

- OnToology
- OntoSpy
- OPCloud
- ORMiE
- PathQL
- pronto
- RDFox
- Semantic MediaWiki
- SimPhoNy
- Terminology Harmonizer
- Virtuoso
- Visual Studio Code
- WebVOWL
- WIDOCO
- WIMU
- yEd Graph editor

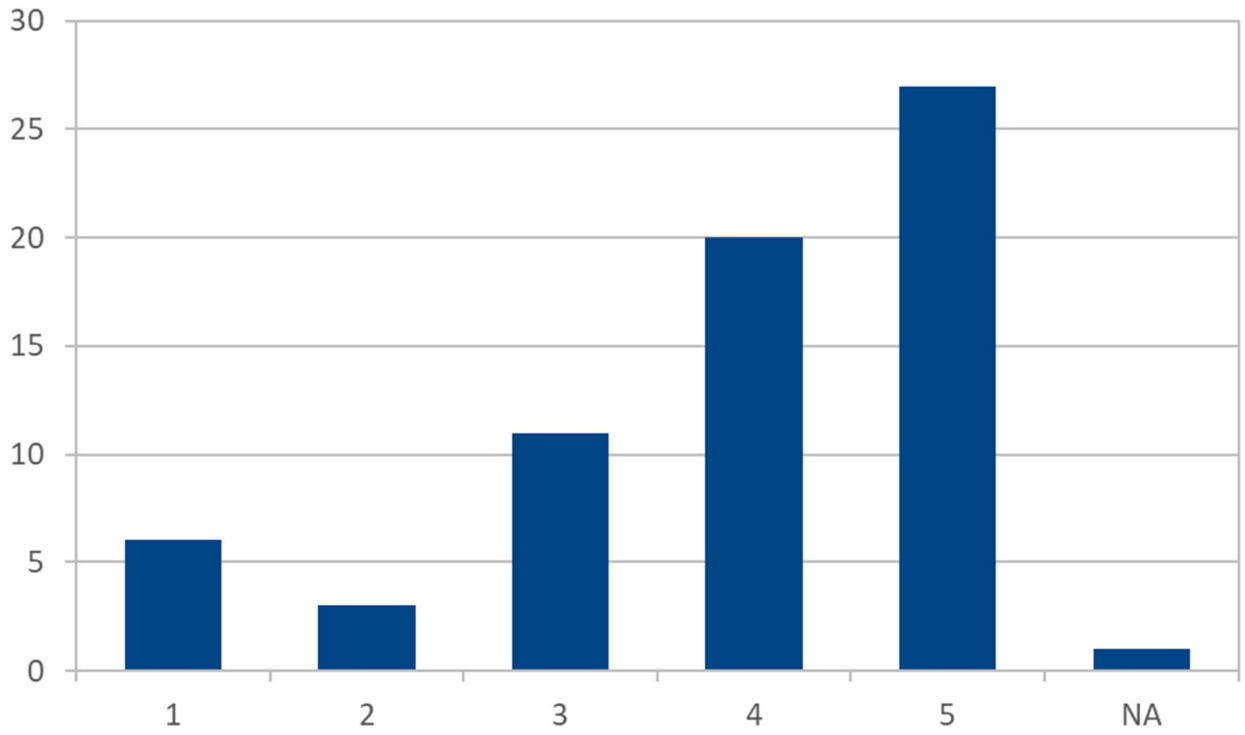

Figure 11: Responses for CRQ_T_9 - User Friendly.

4.3.9.2 Qualitative Results

The nine tools that received a score of only 1 or 2 have mainly been described as specialised tools for developers and are thus not considered user friendly to a broader audience. Otherwise, many

tools have received a score of 4 or 5 with the justification that they provide a simple user interface that is explained well with examples and tutorials.

4.3.10 CRQ_T_10 - Connected Ontologies

4.3.10.1 Quantitative Results

Figure 12 shows the responses for the 5-point rating scale w.r.t. CRQ_T_10. The results indicate that most tools do not enable or support the compatibility of ontologies from different domains. Yet, a few tools received a high score. The following 21 tools received a score of 4 or 5:

- Atomic Data
- CASPAR
- CENtree
- Chowlk
- Dynaccurate
- GRUFF
- LinkML
- OData2SPARQL
- Ontology Lookup Service (OLS)
- OntoSpy
- OPCloud
- OWBO - Ontology White Board
- OWLAPI
- Owlready2
- PathQL
- Pool Party
- RDFox
- SimPhoNy
- TopBraid Composer Maestro Edition
- Virtuoso
- WIMU

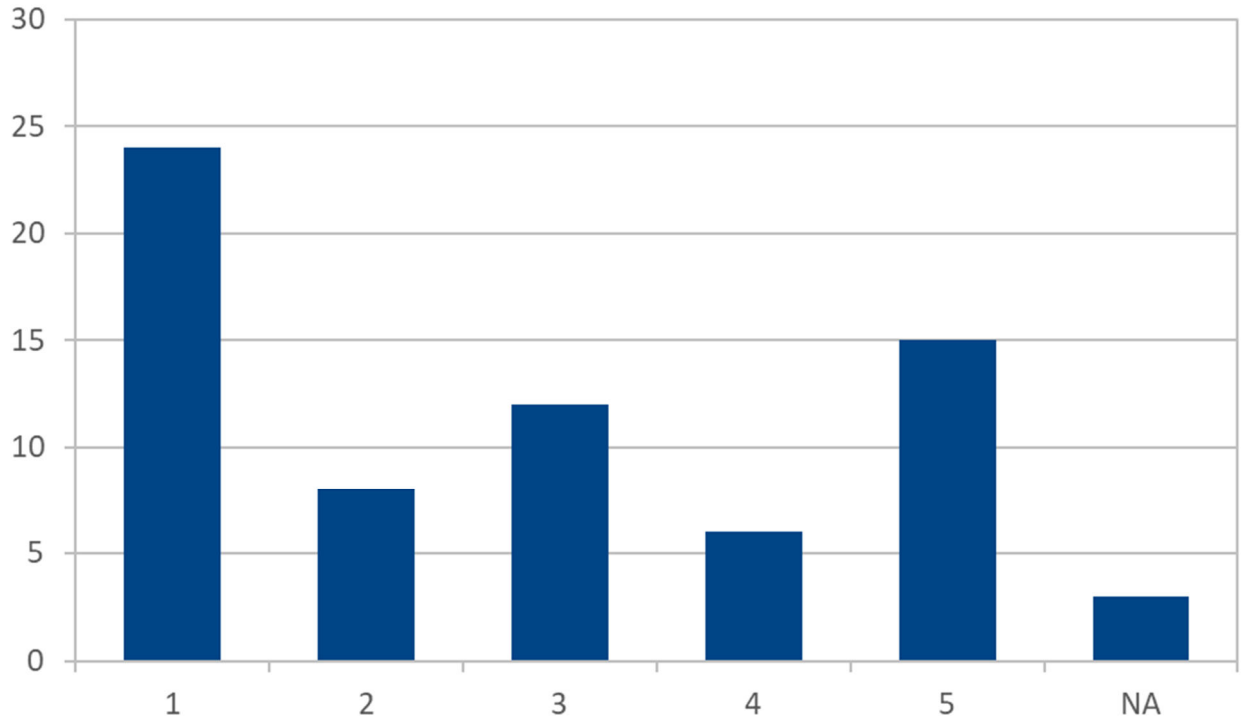

Figure 12: Responses for CRQ_T_10 - Connected Ontologies.

4.3.10.2 Qualitative Results

Many tools receiving a score of 4 or 5 have been justified on the basis that the tool is a general-purpose tool for ontology engineering that can be used independently from an ontology's domain. So, it is argued that, in principle, ontologies from different domains can be connected by the use of these tools. Only TopBraid Composer and CENtree seem to provide dedicated features for creating mappings between ontologies.

Otherwise, Chowlk is said to

> "[...] provide mechanisms to modularize and connect different ontologies."

4.3.11 CRQ_T_11 - Tool Integration

4.3.11.1 Quantitative Results

Figure 13 shows the responses for the 5-point rating scale w.r.t. CRQ_T_11. The data indicates that most tools do provide features to create initial conceptual models which can be easily integrated with other tools for the purpose of their actual implementation. However, the following 11 tools received a score of 4 or 5:

- CENtree
- Chowlk
- crowd-tool
- Dynaccurate

- Ontop
- Ontopanel
- OPCloud
- OWBO - Ontology White Board
- Terminology Harmonizer
- WebVOWL
- yEd Graph editor

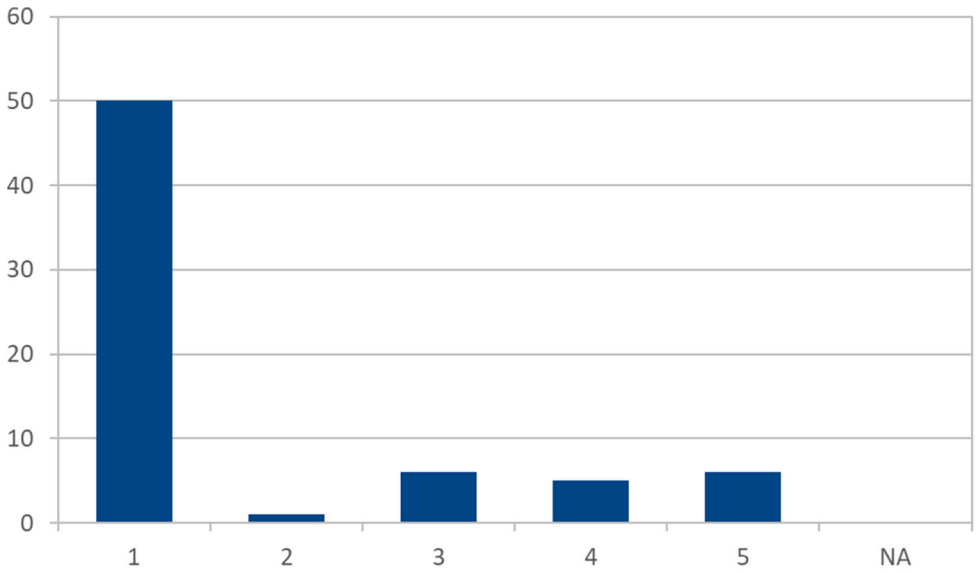

Figure 13: Responses for CRQ_T_11 - Tool Integration.

4.3.11.2 Qualitative Results

The justifications for the given scores suggest that both Chowlk and OWBO were designed for the purpose of transforming conceptual models into concrete OWL ontologies. OPCloud is said to support "rapid modelling for brainstorming and transition from high conceptual level to lower level computational processes in the same environment", while Terminology Harmonizer enables the "initial stage of agreeing upon terms definitions".

4.3.12 CRQ_T_12 - Modularisation

4.3.12.1 Quantitative Results

Figure 14 shows the responses for the 5-point rating scale w.r.t. CRQ_T_12. While most tools do not seem to provide any features for ontology modularisation, the following 13 tools received a score of 4 or 5:

- Atomic Data
- Chowlk
- Hozo Ontology Editor
- IntelligentGraph
- OData2SPARQL
- Ontofox
- OPCloud
- ORMiE
- OWLAPI
- PathQL
- Pool Party
- ROBOT
- SimPhoNy

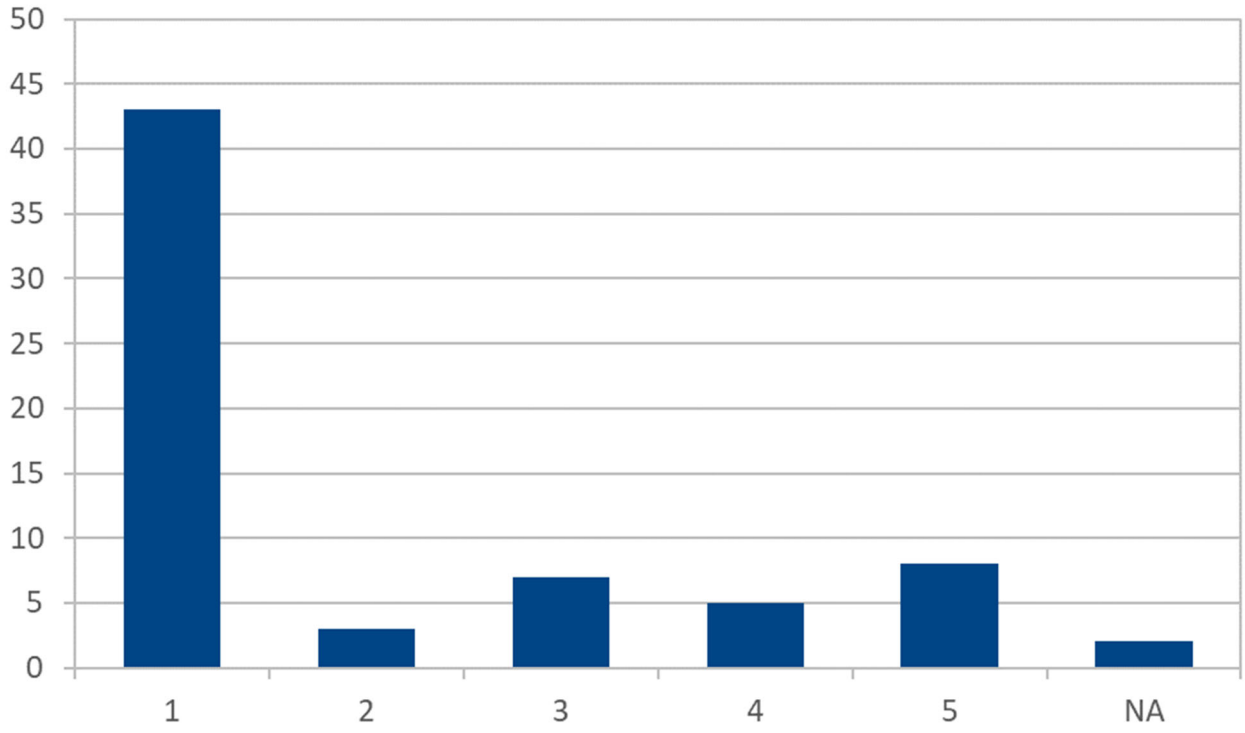

Figure 14: Responses for CRQ_T_12 - Modularisation

4.3.12.2 Qualitative Results

The justifications for the given scores do not specify what kind of modularisation is supported.

4.3.13 CRQ_T_13 - Searching Ontologies

4.3.13.1 Quantitative Results

Figure 15 shows the responses for the 5-point rating scale w.r.t. CRQ_T_13. Most surveyed tools to not seem do provide support for searching ontologies. However, the following 19 tools received a score of 4 or 5:

- AllegroGraph
- CENtree
- IntelligentGraph
- Linked Open Vocabularies (LOV)
- OData2SPARQL
- OFAIR
- Ontofox
- Ontology Lookup Service (OLS)
- OnToology
- Ontopanel
- OntoSpy
- OPCloud
- PathQL
- Pool Party
- Protégé
- ROBOT
- TopBraid Composer Maestro Edition
- WIDOCO
- WIMU

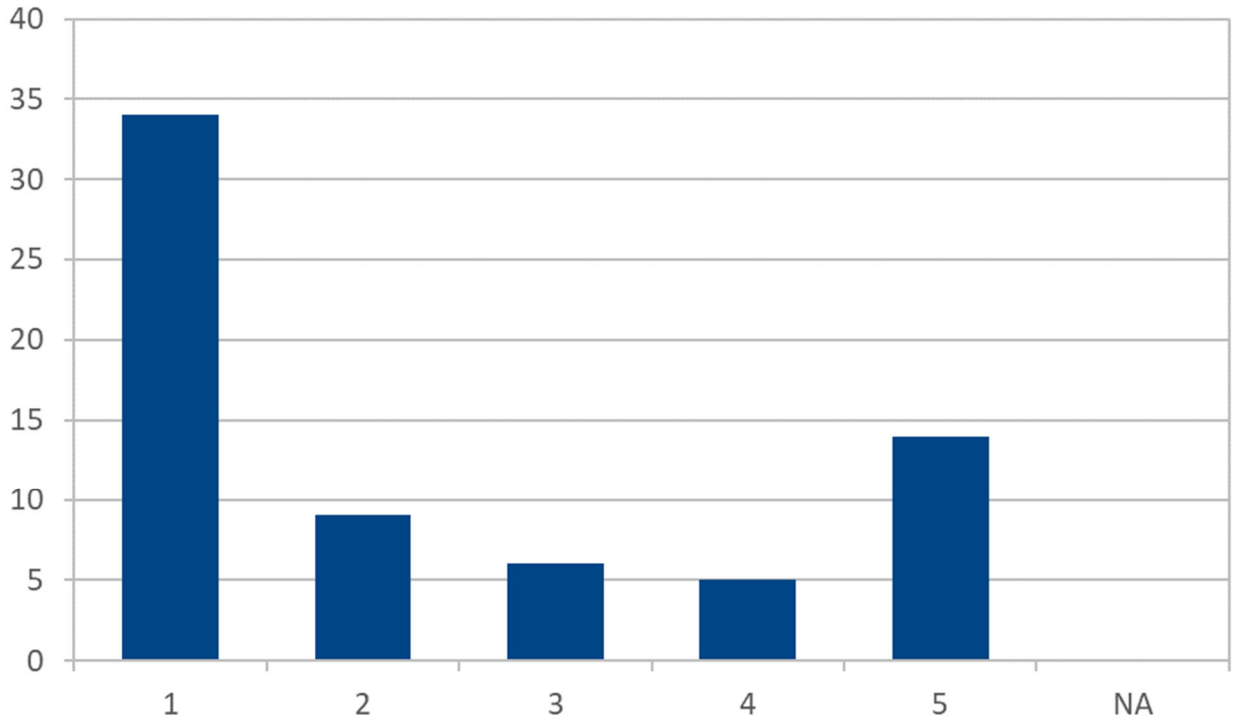

Figure 15: Responses for CRQ_T_13 - Tools for Searching Ontologies

4.3.13.2 Qualitative Results

The justification for given scores reveal that high scores have often been given for tools that support searching entities *within* a given ontology. Otherwise, only Protégé, Ontofox, O'FAIRE, CENtree, Ontopanel and Ontology Lookup Service (OLS) were said to provide features for searching ontologies proper.

4.3.14 CRQ_T_14 - Reusability of Ontologies

4.3.14.1 Quantitative Results

Figure 16 shows the responses for the 5-point rating scale w.r.t. CRQ_T_14. While most tools do not provide support for ontology reuse, the following 13 tools received a score of 4 or 5:

- DBPedia Archivo
- Dynaccurate
- FOOPS!
- IntelligentGraph
- Linked Open Vocabularies (LOV)
- OData2SPARQL
- OFAIR
- OntOlogy Pitfall Scanner! (OOPS!)

- OnToology
- OntoSpy
- OPCloud
- TopBraid Composer Maestro Edition
- VocBench

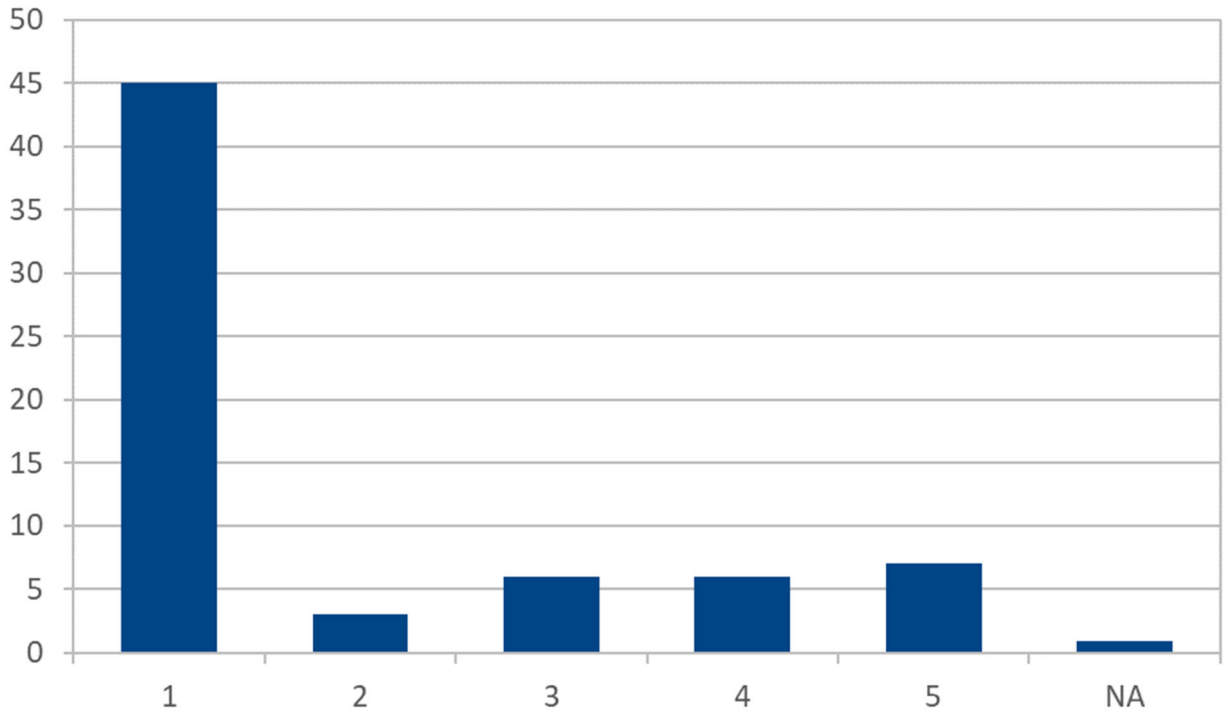

Figure 16: Responses for CRQ_T_14 - Tools for Reusability of Ontologies

4.3.14.2 Qualitative Results

The justifications for high scores often refer to support for data validation that can help with quality assurance of reusing ontologies. Another common justification for high scores refers to the ability of a system to load multiple ontologies simultaneously without necessarily providing dedicated features supporting ontology reuse.

4.3.15 CRQ_T_15 - Correlation among Ontologies

4.3.15.1 Quantitative Results

Figure 17 shows the responses for the 5-point rating scale w.r.t. CRQ_T_15. The results suggest that more than two thirds of tools do not provide features to establish relations between concepts of different ontologies. Only the following 12 tools have received a score of 4 or 5:

- Alignment Cubes
- Atomic Data

- Dynaccurate
- IntelligentGraph
- Linked Open Vocabularies (LOV)
- OData2SPARQL
- Ontop
- OPCloud
- PathQL
- Pool Party
- TopBraid Composer Maestro Edition
- Virtuoso

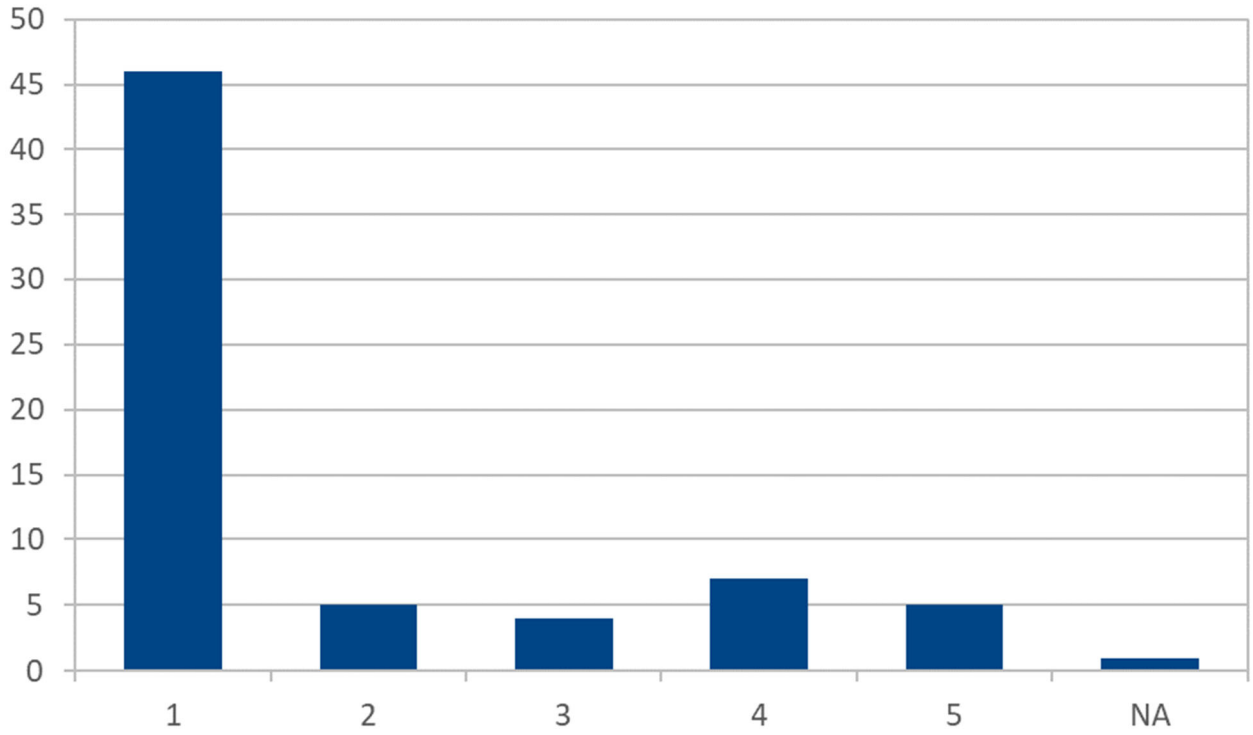

Figure 17: Responses for CRQ_T_15 - Tools for Correlation among Ontologies.

4.3.15.2 Qualitative Results

The justifications for tools that received a score of 4 or 5 hinted at available features for establishing mappings between ontologies. Such features include the computation of correlations (IntelligentGraph), model comparisons based on SPARQL queries (OData2SPARQL), structural relation associations (OPCloud), and matching (PoolParty).

4.3.16 CRQ_T_16 - Deployment and Generation of Documentation

4.3.16.1 Quantitative Results

Figure 18 shows the responses for the 5-point rating scale w.r.t. CRQ_T_16. While most tools do not seem to provide support for the deployment or generation of documentation, the following 12 tools have received a score of 4 or 5:

- Dynaccurate
- EMMOntoPy
- FOOPS!
- LinkML
- OData2SPARQL
- ontology-toolkit/onto-tool
- OnToology
- OntoSpy
- OPCloud
- Protégé
- WIDOCO
- WIMU

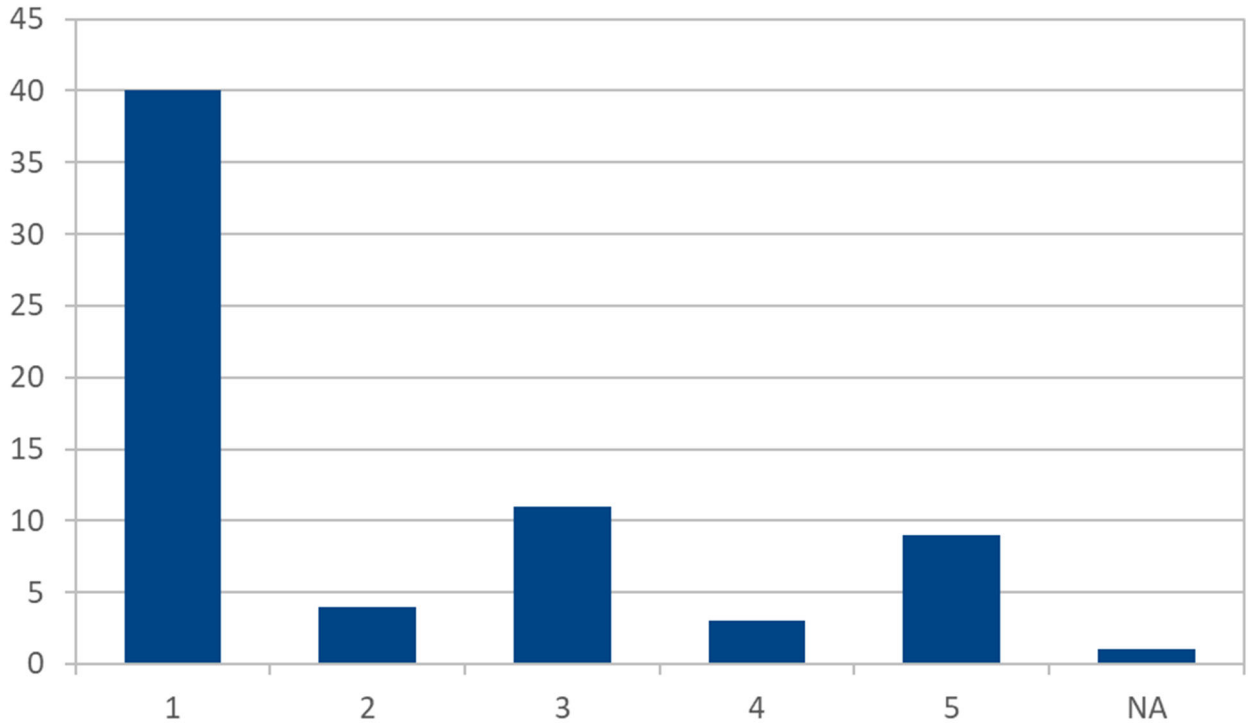

Figure 18: Responses for CRQ_T_16 - Tools for Deployment and Generation of Documentation.

4.3.16.2 Qualitative Results

The justification for scores of 4 or 5 did not specify in detail how documentation can be generated or deployed. However, OntoSpy, LinkML and OPCloud seems to provide features to generate HTML documentation. Otherwise, Ontology-toolkit is said to be designed for the purpose of generating documentation for ontologies.

4.3.17 CRQ_T_17 - Conceptualisation

4.3.17.1 Quantitative Results

Figure 19 shows the responses for the 5-point rating scale w.r.t. CRQ_T_17. The data suggests that most tools do not provide any support for the conceptualisation phase of ontology development. Only the following eight tools have received a score of 4 or 5:

- CENtree
- Chowlk
- crowd-tool
- Hozo Ontology Editor
- IntelligentGraph
- OPCloud
- OWBO - Ontology White Board
- yEd Graph editor

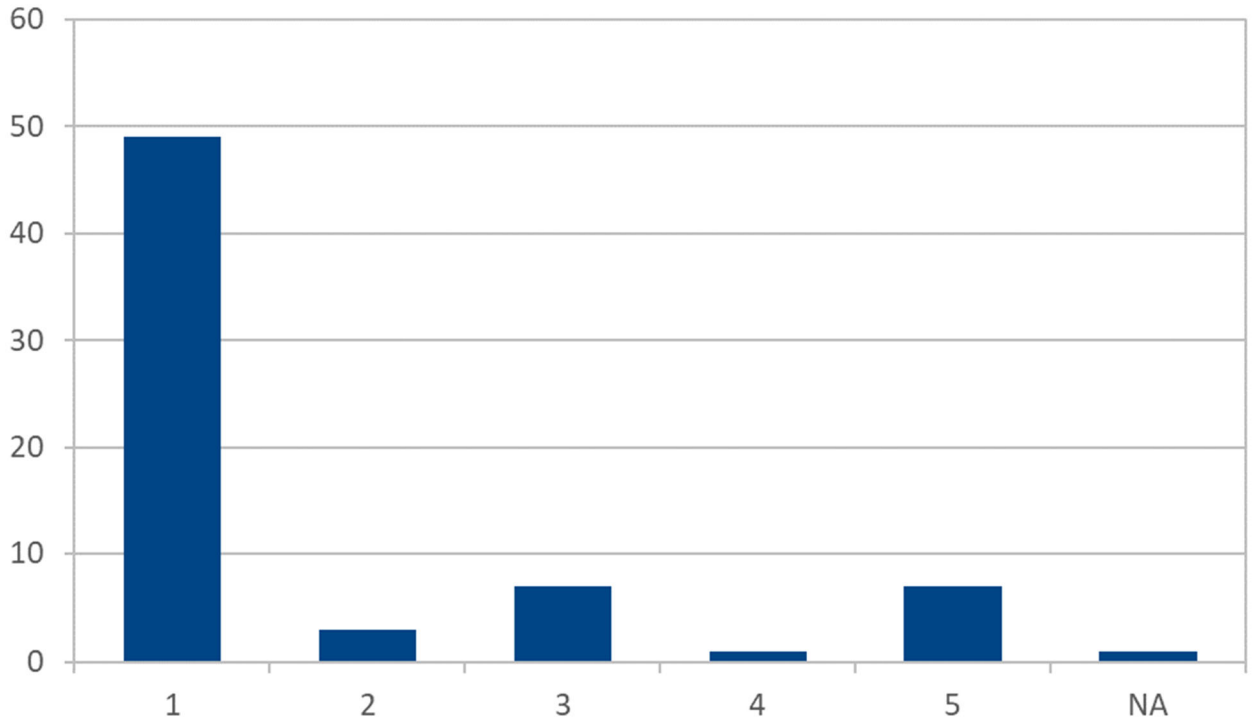

Figure 19: Responses for CRQ_T_17 - Conceptualisation.

4.3.17.2 Qualitative Results

The score for Chowlk was justified as follows:

> *"Tool mainly focused on the development of conceptualization providing a set of visual shapes to represent ontological elements and the providing a service to convert them into OWL."*

Otherwise, OWBO was said to be designed for conceptualisation and CENtree to create ontologies from scratch. Also, Hozo Ontology Editor is an ontology editor which support initial phases of ontology development. The high score of yEd Graph editor was justified as "Being graphical tool it is easy to conceptualize ontologies." Otherwise, crowd-tool was given a high score with the following justification "This is one of the aims of our tool based on the use of conceptual modelling languages.".

4.3.18 CRQ_T_18 - Guide Ontology Reusability

4.3.18.1 Quantitative Results

Figure 20 shows the responses for the 5-point rating scale w.r.t. CRQ_T_18. The data suggests that most tools do not provide any features help a user to determine whether an ontology can be reused or not. Only the following six tools have received a score of 4 or 5:

- CENtree
- Dynaccurate
- IntelligentGraph
- Linked Open Vocabularies (LOV)
- OData2SPARQL
- TopBraid Composer Maestro Edition

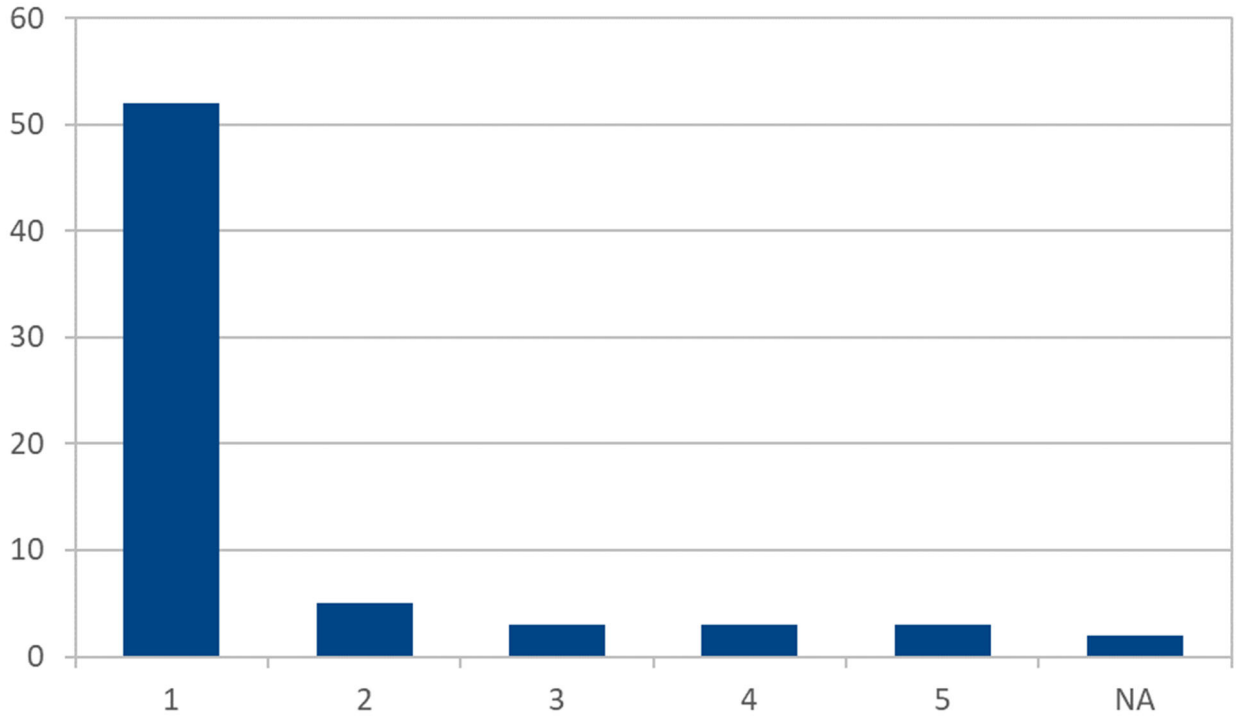

Figure 20: Responses for CRQ_T_18 - Tool for Guide Ontology Reusability.

4.3.18.2 Qualitative Results

The score for Linked Data Vocabulary (LOV) is justified as follows:

"The tools suggest existing ontologies if included in the system."

The score for TopBraid Composer is justified similarly with:

"Users can build ontologies based on existing standard ones."

The score for CENtree is justified in the same vein as:

"More than 80 public ontologies pre-configured that can be searched across to check whether classes already exist for reuse."

The score for OData2SPARQL (5) is justified as:

"Odata2SPARQL will publish the OData meta-model deduced from the underlying ontology, which can prove useful in debugging the completeness of that model."

4.3.19 CRQ_T_19 - GUI Labels Consistency and Understandability

4.3.19.1 Quantitative Results

Figure 21 shows the responses for the 5-point rating scale w.r.t. CRQ_T_19. The data suggests that more than half of the reviewed tools aim to provide helpful GUI labels and use terminology that is easy to understand. The following 27 tools received a score of 4 or 5:

- Alignment Cubes
- AllegroGraph
- CENtree
- Chowlk
- DBPedia Archivo
- Dynaccurate
- FOOPS!
- GraphDB
- Hozo Ontology Editor
- IntelligentGraph
- Linked Open Vocabularies (LOV)
- OData2SPARQL
- OFAIR
- Ontofox
- Ontology Lookup Service (OLS)
- OnToology
- Ontop
- OPCloud
- Pool Party
- Protégé
- RDFox
- SimPhoNy
- Stardog
- TopBraid Composer Maestro Edition
- WebVOWL
- WIDOCO
- yEd Graph editor

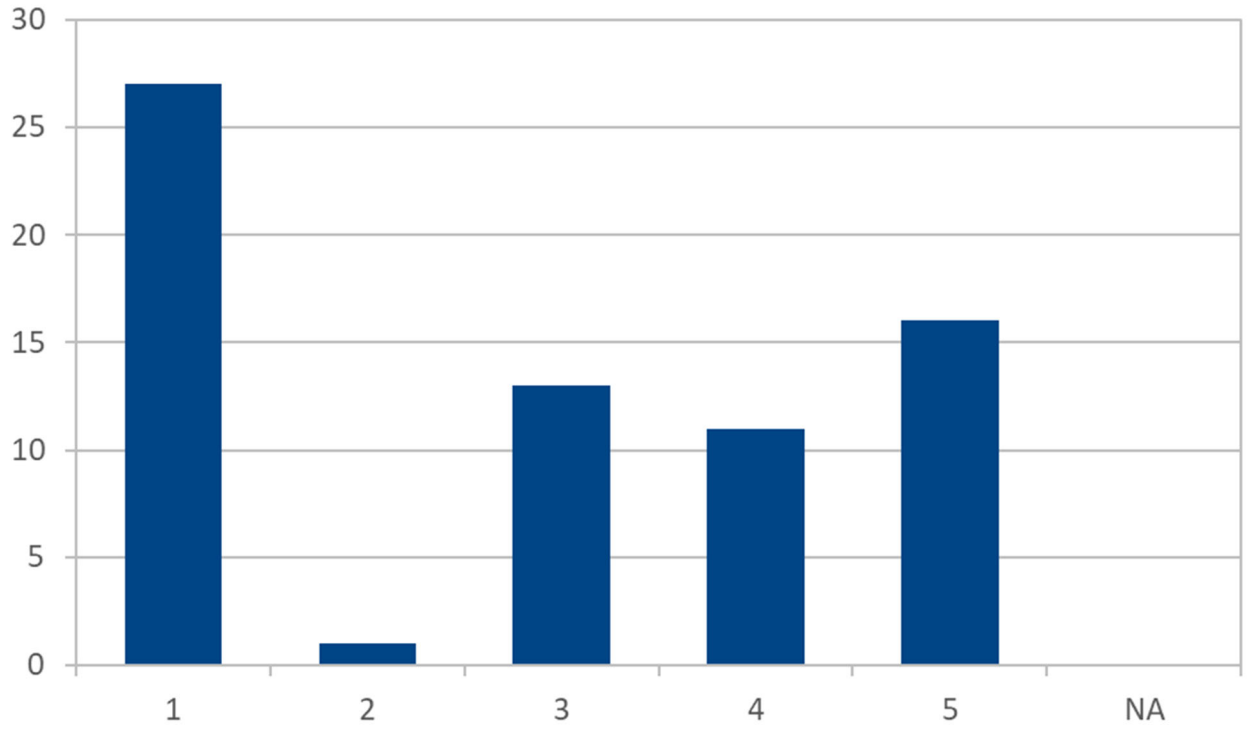

Figure 21: Responses for CRQ_T_19 - Tool GUI Labels Consistency and Understandability

4.3.19.2 Qualitative Results

The justifications for given scores do not explain how labels are used in a helpful manner by tools nor do they explain how they facilitate understandability.

5. Discussion

5.1 Tool Categorisation

There exist many software systems that are used in the context of ontology engineering. While such software systems are often tailored towards specific tasks in ontology engineering, this is not necessarily the case. For example, the ontology-development-kit[2] showcases how various software engineering tools, e.g., GitHub, Make, Docker, can be used to automate ontology release workflows.

Navigating the landscape of software systems that can be used in the context ontology is often challenging due to the vast number of available tools and their relation to one another. Therefore, we propose to categorise existing tools in terms of high-level categories that can be used as a guide to inform the search of suitable software systems for a given use case. While our survey does not aim to provide an exhaustive or otherwise complete overview of the current landscape of software

[2] https://github.com/INCATools/ontology-development-kit

systems in the context of ontology engineering, it does provide a first overview for a selection of tools that have been reported to be used in practice.

It transpires that many tools provide features related to at least one of the categories "Implementation" (2), "Publication" (3), "Evaluation and Validation" (5), and "Use" (6). However, only comparatively few provide features related to the categories "Requirement Specification" (1), "Maintenance" (4), and "Other" (7). Please note that these results do not warrant the conclusion that there is only scarce tool support for the categories "Requirement Specification" (1) and "Maintenance" (4) due to the focused scope of this study. However, the low number of responses for "Other" can arguably be interpreted to indicate that the categories (1) to (6) and their respective subcategories are appropriate to describe the main functionality of most tools.

While this categorisation is not meant to be conclusive, it suggests that the landscape of software systems for ontology engineering can, in principle, be organised in terms of a small number of high-level categories. Such a categorisation may be refined in terms of additional subcategories which can then be used to inform and facilitate the search of suitable tools for a given use case.

However, it needs to be mentioned that many participants were unsure about how to categorise a given tool in terms of the provided options. A fair number of participants pointed out that it was not clear to them when a tool qualifies to be included in a given category or not. It seems therefore that the proposed categorisation would need to be made much more concrete in terms of a well-defined terminology and a clear specification for each category in particular.

5.2 Index of Software Systems for Ontology Engineering

In addition to guiding the search for suitable software systems with high-level categories, we also provide an index of software systems that lists various resources for a given software system, e.g., its homepage and code repository, links to documentation materials and references to publications. Interestingly, this kind of material is publicly available on the web for most software systems that we surveyed. To some extent, this is to be expected from software systems that are used by ontology engineers and practitioners in practice.

While our survey did not attempt to assess the quality of said software systems, it is still noteworthy that most reviewers reported the existence of official releases for most software systems, many of which can be considered production-ready (44 out of 68) and most of which are actively maintained (59 out of 68).

However, it needs to be acknowledged that we did not guard against various forms of participation biases. For example, some participants in our study are not only users of reviewed tools but are in fact their main developers or primary contributors. On the one hand, active developers of a tool are most qualified to report on what kind of features their respective tools provide. On the other hand, it is possible that developers present their tools in a more positive light compared to actual users. In the same vein, it is possible that some tools collected in this survey are not necessarily well-established and proven software systems that are used or driven by an active user base.

5.3 Requirements on Tools

Despite the limited number of tools that are included in this survey, i.e., 68, we find that almost all common requirements for tools (CRQ_T_1 to CRQ_T_19) are supported by at least a few tools. While this could be taken as an indication that the landscape of tools for ontology engineering is rich and provides suitable tool support for almost all aspects of ontology engineering, one has to be careful not to equate high scores of a tool with a comprehensive and sophisticated set of provided features.

For example, many tools have received a score of 4 or 5 w.r.t. CRQ_T_04 "*Validation*". However, the justifications for these scores often reveal that a tool only provides very limited support for validating ontologies, e.g., by only providing support for SHACL shapes or by just performing simple syntax checks for a given file format. So, even though our analysis may provide a first overview of the current landscape of tools for ontology engineering, the work is still preliminary and does not yet warrant any strong conclusions as to whether the landscape of ontology engineering tools provides sufficient support for practical demands in application domains.

Nevertheless, the collected data for different requirements may still provide useful information to an ontology engineer or practitioner in search of a tool with certain features for a given use case. Instead of sifting through the web without any clear guidance, our analysis of tools with respect to high-level categories and requirements may be used as a catalogue of tools that can guide a search in terms of features one might be interested in.

5.4 Guiding Assumptions and Limitations

It is important to recognise that the work presented in this report is subject to a number of limitations. These limitations concern the scope of the survey, the selection of study participants, the allocation of study participants to tools, the used terminology, as well as the design of the used questionnaire. We will discuss each of these points in more detail in the following.

Survey Scope. In Section 3.1.2, we have described how we have restricted the scope of our landscape analysis of tools for ontology engineering. Even though the selected tools are based on the responses of three different groups of people, namely (a) participants of the workshop as described in report on "Ontology Ecosystem Specification", (b) project members, and (c) external expert associated with the OntoCommons project, we cannot argue that these three groups of people are representative of practitioners and ontology engineers that use tools in practice.

So, while some of the tools included in this study are likely to be widely used, there is a possibility sample and participation biases because neither the participants of the workshop held in the context on the report on "Ontology Ecosystem Specification" nor the project members or external experts have been randomly selected. In particular, it can be assumed that a fair number of participants of both the workshop as well as our study are interested in this kind of work because they are tool developers themselves. Furthermore, it is possible that tool developers have given their own tools more favourable scores with respect to requirements CRQ_T_1 to CRQ_T_19 than would be warranted.

Study participants. We need to highlight that we were able to recruit only 39 study participants for reviewing tools. As a consequence of this, we did not manage to review all the tools that were originally included in the scope of our survey and could only collect a single review for each tool. Furthermore, we allowed participants to review multiple tools. This is problematic in the sense that biases of individual participants can affect the review of multiple tools in the same manner. However, it is not straightforward to control for such biases in the context of our study as we will outline in the following.

One could argue that a comparison between multiple reviews of different study participants for a single tool could guard against potential biases. However, one should also be cautious to draw conclusions based on aggregates of responses in this context. Assume that we have 5 reviewers for a given tool and consider the case in which 4 provide a score of 1 w.r.t. visualisation while one reviewer gives a score of 5. It could simply be the case that the 4 reviewers giving a score of 1 do not know about the visualisation feature of a tool while the reviewer giving a score of 5 does know about this feature. This scenario may not be rare if we attempt to recruit a random group of people for each tool because the more people we recruit, the higher the chance of recruiting someone who is not sufficiently familiar with a tool. This is of particular importance since we did not manage to recruit a single reviewer for all the tools that we intended to include in our survey in the first place.

Terminology. As already mentioned in Section 2, the used terminology in this report as well as the reports it builds on is not well-specified and can give rise to subjective interpretations. This freedom in interpretation can make it difficult to compare responses of different reviewers for similar tools directly. However, the provided justifications for given answers can often shed light on how different reviewers have interpreted certain terms and how their interpretation has impacted their responses.

Consider for example, the responses of the two reviewers for the reasoners FaCT++ and HermiT. While the former was categorised under "Requirement Specification" (1), "Maintenance" (4), and "Use" (6), the latter was only categorised under "Use" (6). Furthermore, the reviewer of FaCT++ generally gave higher scores for questions regarding items CRQ_T_1 to CRQ_T_19 compared to the reviewer of HermiT. To be more precise, FaCT++ often received a score of 4 or 5 for many items while HermiT only received a score of 1.

So, given these responses for what one would expect to be similar tools with similar functionality, one might hypothesise that the reasoners FaCT++ and HermiT either differ considerably in terms of their provided features — or, alternatively, that the reviewers for both tools have interpreted the questions CRQ_T_1 to CRQ_T_19 differently which led them to give different scores. However, the provided justifications by both reviewers reveal that they have interpreted the items CRQ_T_1 to CRQ_T_19 in a similar way. Yet, the reviewer of FaCT++ still justified high scores for many questions if there was a way in which FaCT++ could be used *as a component* within an integrated system to address a requirement formulated by CRQ_T_1 to CRQ_T_19.

As a more concrete example of this, consider the item "*CRQ_T_01 - Collaboration of multiple stakeholders: The tool allows different stakeholders to work simultaneously.*" in our questionnaire. While the reviewers for both FaCT++ and HermiT state that the respective reasoner is not a tool for collaboration that would allow multiple stakeholders to work simultaneously, the reviewer for FaCT++ still gives a score of 4 by providing the following justification:

> "However, implementing this into continuous integration / continuous deployment (CI / CD) workflows facilitates collaboration through verification and unification."

This example shows that different participants may provide different scores to a given item in our questionnaire even if they interpret the item in essentially the same way and agree on the correct answer.

So, while a more precise terminology might help to reduce the probability of diverging answers due to subjective interpretations, it appears that the issue cannot be avoided. Instead, one has to take this into account for the data analysis and aim to distil the underlying reasons for given answers. This is what we tried to achieve by requiring reviewers to justify their responses to items of our questionnaire in prose.

Nevertheless, the description of requirements CRQ_T_01 to CRQ_T19 as proposed in the report "Requirements on ontology tools and ontologies and criteria for selection of further cases" have been commented on by a fair number of study participants as unclear if not incomprehensible. Many participants were not able to provide useful feedback with regards to a requirement because they could not make sense of a requirement's description. In other cases, it was mentioned that a requirement would need to be specified in much more detail to judge whether a tool provides support for a requirement or not.

Design of Questionnaire. The questionnaire used in this survey was designed to be aligned with work completed by other reports. In particular, the categorisation of tools was taken from "D4.1 - Ontology Ecosystem Specification" and the questions on requirements CRQ_T_01 to CRQ_T_19 were taken from the report on "Requirements on ontology tools and ontologies and criteria for selection of further cases". However, it is important to note that neither the categorisation nor the formulation of requirements were originally proposed to be used as items of a questionnaire.

So, it is not surprising that they often do not adhere to widely accepted best practices for the construction of questionnaires items. For example, one basic principle is to limit the number of questions per item to exactly one. However, the requirement CRQ_T_5 "*Quality assurance and analytics: The tool supports quality assurance in domain operations and ontology development.*" contains two questions, namely "The tool supports quality assurance in domain operations" and "The tool supports quality assurance in ontology development". Nevertheless, participants of the survey were required to justify their responses for every item in prose where they had the opportunity to disclose their confusion about a given item and discuss how this confusion may have affected their response.

6. Conclusion

In this report, we have surveyed the landscape of existing tools for ontology engineering. We have limited the scope of this survey to tools that have been reported to be reused in practice. We have categorised said tools and compiled them an index, so that practitioners searching for a tool with particular features can browse through them using high-level categories and can consult the index for more information.

The categorisation of tools proposed in the report on "Ontology Ecosystem Specification" seems to provide a sufficiently broad scope for describing tools in the context of the OntoCommons project as participants in this survey only suggested a few additional categories. However, participants were often unsure when a tool qualifies to be included under a given category. So, it seems advisable to revise the description of proposed categories and to develop a more rigorous specification for each category.

Furthermore, we have conducted an analysis w.r.t. requirements that have been reported to be of practical relevance. While there is no Swiss army knife tool for ontology engineering that provides features for all needs and purposes in practice, we found that for all of the 19 requirements listed in the report on "Requirements on ontology tools and ontologies and criteria for selection of further cases", there seem to exist a few tools that provide at least some helpful functionalities.

However, the quantitative data indicates that there are some areas in ontology engineering that are not yet well supported by tools. These are tool integration (CRQ_T_11), modularisation (CRQ_T_12), correlation (CRQ_T_15) conceptual modelling (CRQ_T_17), and ontology reusabliity (CRQ_T_14 as well as CRQ_T_18). Note that this conclusion needs to be interpreted in the context of the limitations of this study as discussed. In particular, one needs to keep in mind that the description of requirements as put forward by the report on "Requirements on ontology tools and ontologies and criteria for selection of further cases" were found to be confusing by study participants which is why they were often unable to provide useful information with regards to many requirements. Overall, it seems that these requirements for tools should be revised.

7. Appendix

7.1 Questionnaire

(see next page)

OntoCommons D4.3 Landscape Analysis Tool Review

This section of the form asks you to provide information about the tool.

* Required

1. Tool title *

2. Tool description *

Provide an introductory description to the tool, typically taken from the tool's website. The text provided here should be self contained and easy to understand. The text should be written so that it is possible to include directly into the report without any changes, i.e, write full sentences and not keywords and lists.

3. Tool categorisation *

Indicate the typical categorisation of the tool. Multiple choices are possible.

Check all that apply.

- [ ] Requirement Specification tools: Conceptualisations, Constraints, Tests
- [ ] Implementation tools: Drafting, Editing, Matching
- [ ] Publication tools: Modulariser, Documentor, Hosting
- [ ] Maintenance: Test executor, Populator
- [ ] Evaluation and Validation
- [ ] Use: APIs, query engines, reasoners, visualisers

Other: [ ]

4. Tool homepage

5. Tool code repository

Link to the tool's code repository, e.g., GitHub

6. Tool documentation

Link to the tool's documentation.

7. Main publication

If available, give the main publication where the tool is described.

8. Current release

What is the version number of the current/reviewed version of the tool? (e.g., 2.3.1)

9. Current status

Indicate the maturity or state of the tool: production, in development, prototype, testing.

10. Date of latest release of tool

When was the latest release of the tool?

Example: January 7, 2019

11. Date of latest commit of tool

When was the latest commit of the development branch of the tool?

Example: January 7, 2019

12. Is the tool actively maintained? *

We defined a actively maintained tool as a tool that is not stated to be no longer maintained and that has received attention (commits, issues, comments) by the maintainers of the tool in the last 6 months.

Mark only one oval.

- ◯ Yes
- ◯ No
- ◯ I don't know
- ◯ Other: _____

13. System requirements (OS, programming language, infrastructure)

Indicate the system requirements for the tools, e.g., Windows only, Java, Server application.

_____

14. Clarifying comments

Provide any additional comments that you think are relevant for the information provided in this form, i.e.,. useful clarifications.

_____

_____

_____

_____

_____

Tool review

This part of the form ask you to evaluate the tool against a list of requirements. The requirements are taken from OntoCommons deliverable D5.1. Each requirement has a code, title and short description. For each requirement, you are asked to evaluate the fulfillment of the requirement on a linear scale (Likert scale) ranging from no support to full support.

* 1: "No support" means that the tool does not support the requirement at all.
* 2: "Low support" means that the tools has little support for the requirement.
* 3: "Some support" means that the requirement is met to some degree; examples could be that this is not the core functionality of the tool, or that the requirement can be met to some degree by extending the tool with a plugin.
* 4: "Good support" means that the tool has good support for the requirement, but e.g., that this is not the main functionality of the tool and that there are some features of the requirements that is not supported.
* 5: "Full support" means that the requirement is fully met by the tool.
* In the case that the requirement does not make any sense for the tool, use "No support" and explain why in the justification of your score.

In addition to identifying the tool support of the each of the requirements, you are asked to justify and explain the choice. This description should be written so that it can be included directly into the tool report.

15. CRQ_T_01 - Collaboration of multiple stakeholders: The tool allows different stakeholders to work simultaneously. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

16. Justification for CRQ_T_01 - Collaboration of multiple stakeholders *

Justify the score you gave above.

17. CRQ_T_02 - Visualisation: The tool supports visualisation of ontologies. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

18. Justification for CRQ_T_02. *

Justify the score you gave above.

19. CRQ_T_03 - Debugging: The tool supports debugging. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

20. Justification for CRQ_T_03. *

Justify the score you gave above.

21. CRQ_T_04 - Validation: The tool supports validation. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

22. Justification for CRQ_T_04. *

Justify the score you gave above.

23. CRQ_T_05 - Quality assurance and analytics: The tool supports quality assurance in domain operations and ontology development. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

24. Justification for CRQ_T_05. *

Justify the score you gave above.

25. CRQ_T_06 - Allow/support different mechanisms to access data: The tool supports easy interaction with ontologies, for example, via REST APIs alongside SPARQL queries for retrieving data. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

26. Justification for CRQ_T_06. *

Justify the score you gave above.

27. CRQ_T_07 - Support for OWL: The tool supports editing OWL files. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

28. Justification for CRQ_T_07. *

Justify the score you gave above.

29. CRQ_T_08 - Ontologies import: The tool supports importing and reusing existing ontologies. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

30. Justification for CRQ_T_08. *

Justify the score you gave above.

31. CRQ_T_09 - User friendly: The tool should be easy to use. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

32. Justification for CRQ_T_09. *

Justify the score you gave above.

33. CRQ_T_10 - Connected ontologies: The tool supports compatibility of different domains (e.g. processes and materials), since different ontologies might be needed. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

34. Justification for CRQ_T_10. *

Justify the score you gave above.

35. CRQ_T_11 - Tool integration: The tool supports initial brainstorming on models of concepts relevant for the domain and applications, to enhance transition from initial ideas to standard tools (e.g. Protégé). *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

36. Justification for CRQ_T_11. *

Justify the score you gave above.

37. CRQ_T_12 - Modularisation: The tool supports facilitation of modularization of ontology models. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

38. Justification for CRQ_T_12. *

Justify the score you gave above.

39. CRQ_T_13 - Tools for searching ontologies: The tool supports ontology search (find ontologies, entities, definitions, etc., e.g. finding existing ontologies to fit an application, based on the initial model of needed concepts). *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

40. Justification for CRQ_T_13. *

Justify the score you gave above.

41. CRQ_T_14 - Tools for re-usability of ontologies: The tool supports quality assessment of the ontology models to be reused. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

42. Justification for CRQ_T_14. *

Justify the score you gave above.

43. CRQ_T_15 - Tools for correlation among ontologies: The tool supports establishment of relation of concepts from diverse ontologies. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

44. Justification for CRQ_T_15. *

Justify the score you gave above.

45. CRQ_T_16 - Tools for deployment and generation of documentation: The tool supports effective deployment of ontologies, including effective generation of documentation etc. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

46. Justification for CRQ_T_16. *

Justify the score you gave above.

47. CRQ_T_17 - Tool for conceptualisation: The tool supports the conceptualisation phase of ontology development. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

48. Justification for CRQ_T_17. *

Justify the score you gave above.

49. CRQ_T_18 - Tool for guide ontology reusability: The tool supports guiding the user and suggesting whether existing ontologies can be reused or if new are required. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

50. Justification for CRQ_T_18. *

Justify the score you gave above.

51. CRQ_T_19 - Tool GUI labels consistency and understandability: The GUI labels of the tool should comply with terminology used in ontologies to be used by ontologists, but also clear and understandable by non-ontologists. *

Mark only one oval.

	1	2	3	4	5	
No support	◯	◯	◯	◯	◯	Full support

52. Justification for CRQ_T_19. *

Justify the score you gave above.