FACULTY OF ENGINEERING AND ARCHITECTURE

Assessing, Creating and Using Knowledge Graph Restrictions

Sven Lieber

Doctoral dissertation submitted to obtain the academic degree of
Doctor of Information Engineering Technology

Supervisors
Prof. Ruben Verborgh, PhD* - Prof. Anastasia Dimou, PhD**

* Department of Electronics and Information Systems
Faculty of Engineering and Architecture, Ghent University

** Department of Computer Science
Faculty of Engineering Technology, KU Leuven

March 2022

GHENT UNIVERSITY

ISBN 978-94-6355-578-4
NUR 983, 988
Wettelijk depot: D/2022/10.500/19

Members of the Examination Board

Chair

Prof. Filip De Turck, PhD, Ghent University

Other members entitled to vote

Prof. Julie Birkholz, PhD, Ghent University
Prof. Pieter Colpaert, PhD, Ghent University
Tom De Nies, PhD, First Stage
Prof. Catia Pesquita, PhD, Universidade de Lisboa, Portugal
Prof. Harald Sack, PhD, Karlsruhe Institute of Technology, Germany

Supervisors

Prof. Ruben Verborgh, PhD, Ghent University
Prof. Anastasia Dimou, PhD, KU Leuven

Acknowledgements

When you read this, it means that I actually made it, I finished a PhD! Besides all the research and the book you see in front of you, this also included a lot of travelling and collaborating. In this section I would like to take the opportunity to express my gratitude to people I have met on my path. I would like to thank the jury of this PhD, colleagues, friends and family.

First of all I would like to thank the jury who took the time to read this thesis and question me during my defense(s). After participating in the scientific reviewing process as a reviewer myself, I know that it is challenging to read and judge a comprehensive scientific work on top of all the other duties in the academic and professional environment [1]. I consider every feedback and (requested) change an added value to this dissertation.

To put it bluntly, no I did not sit for years in an ivory tower or lonely in a library to write this dissertation. I had the pleasure to work in a modern office with talented and motivated colleagues. Together we put research in practice through various projects with industry, academia or the government. Our Knowledge on Web scale (KNoWS) research group at IDLab grew quite a lot since the time I started, thus I already would like to apologize to the people I maybe forgot to list here. It is also a pity that during the last 2 years I barely could meet (new) colleagues due to Covid.

I was mainly working in the Knowledge Graph generation "island" of our research group, thus I mostly worked a lot with a few colleagues and less with others. To "the Ben": you were always on top of things, you provided solutions for any problem I could imagine and you know well how to sell our research output to project partners. Thank you. To Pieter H.: thank you for all chats, fun moments and technical support. Thank you for taking initiative for several things such as arranging student jobs, managing websites, arranging gaming afternoons or starting the diversity book club.

Thanks also to Gerald, Dylan, Thomas and Geertjan. Even though we could not work together on an actual project, I really enjoyed collaborating with you for different papers and in general have you around for all kind of questions and chats. Thank you Dörthe for always providing insights based on your mathematical and logic-based background. Many thanks also to (former) colleagues working on other topics than Knowledge Graph generation with whom I also worked from time to time: Brecht, Julian, Harm, Joachim, Martin, Miel,

[1] I assume the jury members were happy that this is a cumulative dissertation and the main chapters of the thesis were already peer-reviewed.

Laurence, Raf, Sitt Min Oo, all Dieters, Pieters, Rubens and Femkes. I will really miss working with so many experts sharing the same technical background as me. There was often limited time and lots of research to perform, thus also a big thanks to all job and master students. You helped us creating demonstrators and hence putting our research into something tangible.

My PhD journey basically began because during my master thesis at the University of Freiburg my supervisors Io and Peter suggested to work together with "someone from Ghent". This someone turned out to be Tom. Thank you Tom for providing me the chance of pursuing this PhD, thank you for giving me all the infos, helping in initial paper work regarding the recognition of my master courses[2] and guiding me in my first project. I thank Anastasia for adopting me and start supervising me after Tom left academia. I might be biased, but I think you did a good job with this PhD student. It was not always smooth and we had some discussions sometimes, but there is always friction when something is moving forward! I learned a lot from you in the past 5 years, your recent appointment as professor is well deserved. I am pretty sure your new research group will flourish and grow, becoming as renowned as our KNoWS group in Ghent. To Ruben: thank you for creating such a great research environment, thank you for every comment and feedback for papers and presentations. I will always think about your feedback regarding how to create presentations for the rest of my life. Thanks to Erik and all postdocs to work in the background and always making sure that we have projects where we can apply our research on. Being a PhD research also comes with a lot of administration, thus also a big thank you to all the other people in the background: Kristof, Laura, Joke, Bernadette, Martine, Davinia and Karen.

Even though the research of this dissertation was performed while I was a full time researcher at IDLab, the finalizing of this dissertation mainly happened while I was already part time working at the Royal Library of Belgium (KBR). I am very happy that I can bring research on Knowledge Graphs (even more) into practice within my new role at KBR. From what I can see so far there are many motivated colleagues and I am looking forward to the years to come.

I also would like to reflect on my path in general, on the friends and family who supported me and of course also shaped me.

Looking back in my life, I guess one could describe the path I took as stumbling through opportunities. Opportunities which I always tried to take and which eventually brought me here. Thus if anyone wants a life advice: always keep an open mind![3] I would never have imagined an academic life, I was mediocre at school and also had no academic role model in my life[4]. Yet I was always easily motivated which helped me a lot in tackling challenges and most importantly to keep at it[5]. After finishing the "Hauptschule" the subsequent "Kaufmännische Schule" and obtaining the "Fachhochschulreife" I eventually was already 19 years old and the government knocked on my door. Since I didn't see myself doing a mandatory army service I did a mandatory civilian service or "Zivildienst" instead which

[2] Never underestimate paperwork!

[3] I admit, it is a quite ordinary and general advice.

[4] My impression of academic life came at best from pop cultural references.

[5] Especially helpful in a multi-year PhD!

gave me some very interesting experiences. Only several temporary jobs later at the age of 21 I realized that I would like to achieve more. I was always a huge fan of science fiction and computers fascinated me. I started my Bachelor in Communication and Software Engineering at a local applied science university that I commuted to every day. And what can say, I really liked the subjects and also got good grades! At this point I would like to express my gratitude to teachers, professors, but also the free educational system as a whole. I am pretty sure that without this kind of support I would not write these lines now.

Now, I have been living abroad for more than 5 years and it has been almost 9 years since I left my hometown. Throughout my path I met many interesting people and personalities, everyone shaping me and adding a small piece to the Sven I am today. Looking back to my hometown I would like to thank Alex M., Alex L., Basti, Chris, Fatima, Gregor, Georg, Hannes, Martin, Mani, Micha[6], Michi and Oli. Thank you for all the parties, summers and the time together in general. It is a pity we do not see each other as often as in the past because some of you already started your own families and I started to live abroad. But I am glad that I know you and happy that I am still in contact with most of you and go on travels with you from time to time[7].

Thinking back to my 4 years in Freiburg where I lived during my Master's, I would like to thank Kai, Lukas, Matze, Manu, Sarah, Sebastian, Simon, Sven2[8], Tom and the student restaurant Mensa Rempartstraße. I shared a great deal of my 20s with all of you and I will always look back with a smile to my time in Freiburg. And a special thanks to Matze with whom I also had the pleasure of studying also for a Bachelor's degree. You encouraged me to apply for a Master degree in Freiburg and I surely would not be here right now if it was not for that. It is a pity that there is no new decent (!) Star Trek series we could watch together.

This brings me to my time in Belgium, the past 5 years in Ghent. I already talked about stumbling through opportunities and I could not have stumbled better as to the house I ended up in. Doing co-housing in a new country was really one of the best decisions I took, besides actually moving to a new country. Especially during Corona times where I did not have to miss social contacts because there were always 5 other people around. Sometimes change was the only constant in our house and I have seen many people come and go. But I am happy that there was still stability and that I could extend my circle of friends with people from many different corners of the world also via that house. Thank you Alejandra[9], Angelos, Björn, Francesco, Helena, Kaushik, Maarten, Marta, Nuno, Nicole and Sarita. Thank you for shaping me and reminding me that there is also still a life besides work. Thanks also to the different people with whom we spent many fun weekends in the Ardennes or on other trips throughout Europe[10].

Als er iets is waar ik spijt van heb, is het dat ik niet eerder Nederlands heb geleerd. Het maakt echt een enorm verschil om je als tijdelijke gast te voelen of ergens deel van uit te maken. Maar eindelijk begon ik een tijdje geleden de taal te studeren en te gebruiken. Bovendien

[6] Kanns sein?

[7] At least when there is no global pandemic.

[8] It's not really his name, but we had to find a practical solution to avoid confusions.

[9] Muchas Gracias!

[10] I only say Ananas.

maken me recente ontwikkelingen zeker dat ik het nodig zal hebben, Ik kijk er naar uit[11]. Bedankt Liesbeth voor al je steun in de afgelopen maanden.

Since we are already in a multilingual mambo-jambo (normal in Belgium) I also would like to thank my family in the closing words of this section. Vielen Dank Mama und Papa, Ihr habt mir Selbstständigkeit beigebracht und mich unterstützt so gut Ihr es konntet. Auch wenn es nicht immer einfach war, habt ihr mich gut auf das Leben vorbereitet. Ich danke euch von tiefstem Herzen!

[11] Mijn kleuterfase in Nederlandse mopjes is vast snel voorbij en mijn grappen zullen verbeteren naar een puber en dan volwassen niveau. Maak je klaar!

Contents

Summary

Humanity has been collecting data and representing information for centuries, but the advent of digital technology and especially the World Wide Web lead to new challenges: the steadily growing amount of diverse data needs to be integrated in a systematic and meaningful way to manage it. Otherwise only large amounts of unconnected data with unknown quality remains.

To achieve smart management of information, we need to represent data in a uniform fashion. Additionally, we need to express restrictions to define which data connections are meaningful or valid in a certain use case to represent the information at hand. One simple but powerful method to represent information is by referring to two things: concepts and relationships between concepts. This forms a graph structure with concepts as nodes and relationships as edges connecting the nodes, a so-called Knowledge Graph. Like this, one can for example represent the three concepts "author", "person" and "book" as well as relationships such as "wrote" or "bought". Whereas the information that the author "Andy Weir" wrote the book "The Martian" is meaningful information, the book "The Martian" cannot write the author "Andy Weir". However, for a computer both examples are valid if no restrictions are in place to limit possible ways of connecting concepts with relationships.

Restrictions to represent what is meaningful in a given context or what is of good quality is subjective and has to be defined by humans. In the given example restrictions could be that an author writes books and that an author is also a person. In this case the restrictions are so-called axioms: stating what is true according to the model. These restrictions can be used by a computer to infer new knowledge: based on the knowledge that Andy Weir wrote the book "The Martian", it can be inferred that he is an author and a person. Another restriction could be that only persons can write books and that all books in a database need an author. In this case the restrictions are so-called constraints, used to identify invalid data. This could be used for a quality assessment to identify missing author information or wrong data.

This dissertation focuses on the creation and use of Knowledge Graph restrictions by humans. When defining abstract concepts, such as "author" or "book", one usually refers to it as vocabularies. Its terms may be restricted by axioms to define meaning, then the vocabulary may be called ontology. When connecting concrete data in Knowledge Graphs, such as the author "Andy Weir" and the book "The Martian", one refers to it as data using terms of such

a vocabulary, for example "Andy Weir is an author" and "The Martian is a book". What is valid for this data in a certain context may be restricted by constraints. To represent all this in a machine-friendly way one can use the following languages recommended by the World Wide Web Consortium (W3C): (i) the Resource Description Framework (RDF) to represent terms, (ii) the RDF Schema (RDFS) and the Web Ontology Language (OWL) to represent axioms (iii) and the Shapes Constraint Language (SHACL) to represent constraints.

The first challenge is supporting users to assess a Knowledge Graph with respect to used restrictions. When building a Knowledge Graph, existing vocabularies are often reused which makes it possible that information in one system is also understood in other systems. These vocabularies often contain axioms which influence potential reuse: some axioms are computationally more complex and one may want to avoid reusing vocabularies with such axioms in a certain use case. Similarly, one may have to assess the use of existing constraints for common vocabularies. But in both cases there is currently limited support for users to compare and select Knowledge Graphs with respect to used restrictions.

The second challenge is how to support users in the creation of constraints. Usually, domain experts know best which constraints they have to impose, but they are no Knowledge Graph experts and need a user-friendly way to create Knowledge Graph constraints. Other studies have shown that visual notations which denote how to represent certain concepts visually support users. Currently there is no such visual notation to visualize Knowledge Graph constraints.

The use of restrictions is use case specific, therefore in this dissertation we focus on a certain use case of data stewardship: supporting national libraries in the preservation of social media. One the one hand, different heterogeneous data sources need to be considered when preserving dynamic social media content. However, currently no complete workflow for social media archiving exists which meaningfully combines the different pieces of data. On the other hand, preserved content needs to be accessed and consulted which poses challenges regarding subjective data quality constraints.

To address the first challenge, we present an approach to measure the use of restrictions in Knowledge Graphs and present collected statistics for axioms and constraints. We first introduce Montolo, an approach to define abstract restriction types such as "subclass" and concrete expressions thereof in RDF such as `rdfs:subClassOf`. Then we present an implementation which creates interoperable restriction use statistics in RDF. We demonstrated the feasibility of this approach by measuring the (i) RDFS and OWL axiom use in more than a thousand ontologies from the generic LOV and domain specific BioPortal repositories, and (ii) constraint use in SHACL shapes from identified GitHub repositories.

To address the second challenge, we focus on how to support humans in the creation of constraints with visual notations that can visualize all constraints specified in SHACL. We built on existing commonly used visual notations in the computer science and Knowledge Graph domain and present the two visual notations ShapeUML and ShapeVOWL. We compare them based on cognitive effective design principles as they are meant to be cognitively processed by human users and evaluate both notations in a comparative user study.

To address the third challenge, we introduce a Knowledge Graph-based solution for social media archiving and a corresponding quality assessment with constraints. Our BESOCIAL solution is based on a declarative Knowledge Graph generation: using common vocabularies and their axioms to meaningfully integrate heterogeneous social media archiving-related data. Furthermore, we present social media archiving-related data quality categories, dimensions and metrics and a low-level validation with Knowledge Graph constraints to measure corresponding higher-level data quality metrics. We followed an established methodology, but compared to existing works, our quality assessment relies on specifications related to the World Wide Web Consortium (W3C) instead of custom software.

The contributions of this dissertation provide interoperable means to assess and work with Knowledge Graph restrictions.

Montolo enables users to assess existing Knowledge Graphs with respect to the use of axioms and constraints. Regarding axioms, we found that vocabularies from the generic LOV and domain specific BioPortal repositories show similar patterns: more than 95% use RDFS-based but only half OWL-based restrictions. The created statistics can support ontology reuse: ontology engineers can now rely on axiom use statistics for the assessment of existing ontologies. Regarding constraints, we found similar patterns to axiom use: relationships between concepts are often restricted to certain classes or data types, whereas constraints regarding literal values are used less. Our statistics reveal a possible issue: a self-fulfilling prophecy where tools to create constraints focus only on commonly used constraint types which eventually produces more of such constraints. Therefore less-used constraint types should get more attention.

The ShapeUML and ShapeVOWL visual notations are independent from a specific constraint language and are built with cognitive effectiveness in mind. Therefore, humans can utilize their fast cognitive system and do not have to rely on a specific textual syntax. The quantitative part of our comparative analysis revealed that users do not make fewer errors with one visual notation or the other, and that with both notations more than 80% of questions are answered correctly. Therefore both visual notations have potential to be adopted for different use cases, our qualitative analysis also points to possible improvements.

Our BESOCIAL workflow for social media archiving enables cultural heritage experts to preserve social media using declarative means, thus without having them to write code. Furthermore, we defined social media collection-related quality categories, dimensions and metrics which can be reused by the community. This use case exemplifies the use of both axioms and constraints to enable data stewardship and provide added value in terms of data integration and data quality. The developed data quality assessment can also be applied for other use cases because our solution relies only on openly available W3C-related specifications.

Interesting future directions include increasing the adoption of visual notations for constraints, as well as a methodology for the creation of Knowledge Graph restrictions.

With respect to the creation of constraints, results obtained from our comparative evaluation of both visual notations with Knowledge Graph experts is a first step towards user-friendly

support for working with Knowledge Graph constraints. Similar studies can be conducted with experts from various domains to improve both the visual notations and the tools implementing the notations. The latter can be improved by investigating different editing workflows. Furthermore, it can be investigated how other constraint languages than SHACL can be represented with our visual notations. One promising candidate is the Shape Expression Language (ShEx) which caught attention in communities working with Wikidata.

Several ontology engineering methodologies exist, but especially with the upcoming of constraint languages such as SHACL new modeling paradigms arose. In this dissertation we applied both axioms and constraints for a cultural heritage use case, future work could investigate a general methodology to support knowledge engineers in the creation of Knowledge Graphs. A methodology for the creation of Knowledge Graphs supporting in ithe decision when to use which axioms and when to use which constraints. This makes design decisions related to restrictions transparent, thus minimizing subjective discussions about the use of axioms vs the use of constraints.

Samenvatting

Al eeuwenlang verzamelt de mensheid data en representeert ze informatie, maar de komst van digitale technologieën en vooral het wereldwijde web leidt tot nieuwe uitdagingen: de gestaag groeiende hoeveelheid diverse gegevens moeten op een systematische en zinvolle manier worden geïntegreerd om deze te beheren. Zoniet blijven grote hoeveelheden niet-verbonden gegevens met onbekende kwaliteit over.

Om slim informatiebeheer te realiseren, moeten we data op een uniforme manier representeren. Bovendien moeten we begrenzingen uitdrukken om te definiëren welke gegevensverbindingen zinvol of geldig zijn in een bepaalde gebruikssituatie om de beschikbare informatie weer te geven. Een eenvoudige maar krachtige methode om informatie weer te geven is door naar twee dingen te verwijzen: concepten en relaties tussen concepten. Dit vormt een graafstructuur met concepten als knooppunten en relaties als randen die de knooppunten verbinden, een zogenaamde kennisgraaf. Op deze manier kan men bijvoorbeeld de drie begrippen äuteur", "persoonën "boek"vertegenwoordigen, evenals relaties zoals "schreeföf "gekocht". Terwijl de informatie dat de auteur "Andy Weir"het boek "The Martian"schreef zinvolle informatie is, kan het boek "The Martian"de auteur "Andy Weir"niet schrijven. Voor een computer zijn beide voorbeelden echter geldig als er geen manieren zijn om mogelijke verbindingen tussen concepten via relaties te begrenzen.

Begrenzingen om weer te geven wat zinvol is in een bepaalde context of wat van goede kwaliteit is, zijn subjectief en moeten door mensen worden gedefinieerd. In het gegeven voorbeeld kunnen de begrenzingen zijn dat een auteur boeken schrijft en dat een auteur ook een persoon is (in dit geval zijn de begrenzingen zogenaamde axioma's, die aangeven wat waar is volgens het model). Deze begrenzingen kunnen door een computer worden gebruikt om te concluderen dat Andy Weir een auteur is en dus ook een persoon, zelfs als we niet expliciet hebben vermeld dat Andy Weir een auteur is. Een andere begrenzing zou kunnen zijn dat alleen personen boeken kunnen schrijven en dat alle boeken in een database een auteur nodig hebben (in dit geval zijn de begrenzingen zogenaamde beperkingen, gebruikt om ongeldige gegevens te identificeren). Dit kan worden gebruikt voor een kwaliteitsbeoordeling om ontbrekende auteursinformatie of verkeerde gegevens te identificeren.

Dit proefschrift richt zich op de creatie en het gebruik van kennisgraafbegrenzingen door mensen. Bij het definiëren van abstracte concepten zoals äuteuröf "boek", verwijst men hier

gewoonlijk naar als vocabularia. De termen kunnen worden begrensd door axioma's om betekenis te definiëren, en dan kan het vocabularium een ontologie worden genoemd. Bij het verbinden van concrete gegevens in kennisgrafen, zoals de auteur "Andy Weirën het boek "The Martian", verwijst men ernaar als gegevens met termen van een dergelijk vocabularium, bijvoorbeeld "Andy Weir is een auteurën "The Martian is een boek". Wat in een bepaalde context voor deze gegevens geldt, kan door beperkingen worden begrensd. Om dit alles op een machinevriendelijke manier weer te geven, kan men de volgende talen gebruiken die worden aanbevolen door het World Wide Web Consortium (W3C): (i) het Resource Description Framework (RDF) om termen weer te geven, (ii) het RDF Schema (RDFS) en de Web Ontology Language (OWL) om axioma's weer te geven en (iii) de Shapes Constraint Language (SHACL) om beperkingen weer te geven.

De eerste uitdaging is om gebruikers te ondersteunen bij het beoordelen van een kennisgraaf met betrekking tot gebruikte begrenzingen. Bij het bouwen van een kennisgraaf worden vaak bestaande vocabularia hergebruikt waardoor het mogelijk wordt dat informatie in het ene systeem ook in andere systemen begrepen wordt. Deze vocabularia bevatten vaak axioma's die potentieel hergebruik beïnvloeden: sommige axioma's zorgen voor extra complexiteit en daarom kan men ervoor kiezen om het hergebruik van vocabularia met dergelijke axioma's in een bepaalde gebruikssituatie vermijden. Evenzo kan het nodig zijn om het gebruik van bestaande begrenzingen voor gemeenschappelijke vocabularia te beoordelen. Maar in beide gevallen is er momenteel beperkte ondersteuning voor gebruikers om kennisgrafen te vergelijken en te selecteren met betrekking tot gebruikte begrenzingen.

De tweede uitdaging is hoe gebruikers te ondersteunen bij het creëren van begrenzingen. Gewoonlijk weten domeinexperts het beste welke begrenzingen ze moeten opleggen, maar ze zijn geen kennisgraafexperts en hebben een gebruiksvriendelijke manier nodig om kennisgraafbegrenzingen te creëren. Andere onderzoeken hebben aangetoond dat visuele notaties die aangeven hoe bepaalde concepten moeten worden weergegeven, gebruikers visueel ondersteunen. Momenteel is er geen dergelijke visuele notatie om de begrenzingen van de kennisgraaf te visualiseren.

Het gebruik van restricties is gebruikssituatiespecifiek, daarom richten we ons in dit proefschrift op een bepaalde gebruikssituatie voor data stewardship: het ondersteunen van nationale bibliotheken bij het behoud van sociale media. Enerzijds moet bij het bewaren van dynamische sociale media-inhoud rekening worden gehouden met verschillende heterogene gegevensbronnen. Op dit moment bestaat er echter geen volledige workflow voor het archiveren van sociale media die de verschillende stukjes gegevens op een zinvolle manier combineert. Aan de andere kant moet bewaarde inhoud worden geopend en geraadpleegd, wat uitdagingen met zich meebrengt met betrekking tot subjectieve gegevenskwaliteitsbeperkingen.

Om de eerste uitdaging aan te gaan, presenteren we een aanpak om het gebruik van begrenzingen in kennisgrafen te meten en presenteren we verzamelde statistieken voor axioma's en beperkingen. We introduceren eerst Montolo, een manier om abstracte begrensingstypes zoals "subclassën concrete uitdrukkingen daarvan in RDF zoals `rdfs:subClassOf` te definiëren. Vervolgens presenteren we een implementatie die interoperabele gebruiksstatistieken

maakt in RDF. We hebben de haalbaarheid van deze aanpak aangetoond door het meten van (i) het RDFS- en OWL-axiomagebruik in meer dan duizend ontologieën van de generieke LOV en domeinspecifieke BioPortal-repositories, en (ii) het gebruik van beperkingen in SHACL-vormen van geïdentificeerde GitHub-repositories.

Om de tweede uitdaging aan te gaan, richten we ons op hoe we mensen kunnen ondersteunen bij het creëren van beperkingen met visuele notaties die alle beperkingen kunnen visualiseren die zijn gespecificeerd in SHACL. We bouwden voort op bestaande veelgebruikte visuele notaties in het domein van de informatica en de kennisgraaf en presenteren de twee visuele notaties ShapeUML en ShapeVOWL. We vergelijken ze op basis van cognitief effectieve ontwerpprincipes, aangezien ze bedoeld zijn om cognitief verwerkt te worden door menselijke gebruikers, en evalueren beide notaties in een vergelijkend gebruikersonderzoek.

Om de derde uitdaging aan te gaan, introduceren we een op kennisgraaf-gebaseerde oplossing voor archivering van sociale media en een bijbehorende kwaliteitsbeoordeling met beperkingen. Onze BESOCIAL-oplossing is gebaseerd op een declaratieve kennisgraafgeneratie: gemeenschappelijke vocabularia en hun axioma's gebruiken om heterogene sociale media-archiveringsgerelateerde gegevens op een zinvolle wijze te integreren. Verder presenteren we sociale media-gerelateerde gegevenskwaliteitscategorieën, dimensies en statistieken, en een validatie op laag niveau met kennisgraafbeperkingen om overeenkomstige gegevenskwaliteitsstatistieken op een hoger niveau te meten. We volgden een gevestigde methodologie, maar in vergelijking met bestaande werken, is onze kwaliteitsbeoordeling gebaseerd op specificaties met betrekking tot het World Wide Web Consortium (W3C) in plaats van op maat gemaakte software.

De bijdragen van dit proefschrift bieden interoperabele middelen om kennisgraafbegrenzingen te beoordelen en ermee te werken.

Montolo stelt gebruikers in staat om bestaande kennisgrafen te beoordelen met betrekking tot het gebruik van axioma's en beperkingen. Wat betreft axioma's, vonden we dat vocabularia van de generieke LOV en domeinspecifieke BioPortal-repositories vergelijkbare patronen vertonen: meer dan 95% gebruikt op RDFS gebaseerde maar slechts de helft van op OWL gebaseerde begrenzingen. De gecreëerde statistieken kunnen het hergebruik van ontologie ondersteunen: ontologie-ingenieurs kunnen nu vertrouwen op axioma-gebruiksstatistieken voor de beoordeling van bestaande ontologieën. Met betrekking tot beperkingen vonden we patronen die vergelijkbaar zijn met het gebruik van axioma's: relaties tussen concepten zijn vaak begrensd tot bepaalde klassen of gegevenstypen, terwijl beperkingen met betrekking tot letterlijke waarden minder worden gebruikt. Onze statistieken onthullen een mogelijk probleem: een zelfvervullende voorspelling waarbij tools om begrenzingen te creëren zich alleen richten op veelgebruikte types begrenzingen, die uiteindelijk meer van dergelijke begrenzingen opleveren. Daarom zouden minder gebruikte types begrenzingen meer aandacht moeten krijgen.

De visuele notaties van ShapeUML en ShapeVOWL zijn onafhankelijk van een specifieke beperkingstaal en zijn gebouwd met cognitieve effectiviteit in het achterhoofd. Daarom kunnen mensen hun snelle cognitieve systeem gebruiken en zijn ze niet afhankelijk van

een specifieke tekstuele syntaxis. Uit het kwantitatieve deel van onze vergelijkende analyse bleek dat gebruikers niet minder fouten maken met de ene of de andere visuele notatie, en dat met beide notaties meer dan 80% van de vragen correct worden beantwoord. Daarom hebben beide visuele notaties het potentieel om te worden gebruikt voor verschillende gebruiksscenario's, waarbij onze kwalitatieve analyse wijst op mogelijke verbeteringen.

Onze BESOCIAL-workflow voor archivering van sociale media stelt cultureel erfgoedexperts in staat om sociale media te bewaren met behulp van declaratieve middelen, dus zonder dat ze code hoeven te schrijven. Verder definiëerden we sociale mediacollectie-gerelateerde kwaliteitscategorieën, dimensies en statistieken die door de gemeenschap kunnen worden hergebruikt. Deze use case is een voorbeeld van het gebruik van zowel axioma's als beperkingen om data stewardship mogelijk te maken en toegevoegde waarde te bieden op het gebied van data-integratie en datakwaliteit. De ontwikkelde datakwaliteitsbeoordeling kan ook worden toegepast voor andere gebruikssituaties, omdat onze oplossing alleen vertrouwt op vrij beschikbare W3C-gerelateerde specificaties.

Interessante toekomstige richtingen zijn onder meer het vergroten van de acceptatie van visuele notaties voor beperkingen, evenals een methodologie voor het maken van kennisgraafbegrenzingen.

Met betrekking tot het creëren van beperkingen, zijn de resultaten die zijn verkregen uit onze vergelijkende evaluatie van beide visuele notaties met kennisgraafexperten de eerste stap naar gebruiksvriendelijke ondersteuning voor het werken met kennisgraafbeperkingen. Vergelijkbare studies kunnen worden uitgevoerd met experts uit verschillende domeinen om zowel de visuele notaties als de tools voor het implementeren van de notaties te verbeteren. Dit laatste kan worden verbeterd door verschillende bewerkingsworkflows te onderzoeken. Verder kan worden onderzocht hoe niet-SHACL beperkingstalen kunnen worden weergegeven met onze visuele notaties. Een veelbelovende kandidaat is de Shape Expression Language (ShEx) die de aandacht trok in gemeenschappen die met Wikidata werken.

Er bestaan verschillende ontologiebouw methodologieën, maar vooral met de opkomst van beperkingstalen zoals SHACL ontstonden nieuwe modelleringsparadigma's. In deze dissertatie hebben we zowel axioma's als beperkingen toegepast voor een cultureel erfgoed gebruikssituatie. Toekomstig werk zou een algemene methodologie kunnen onderzoeken om kennisingenieurs te ondersteunen bij het maken van kennisgrafen, d.w.z. ondersteuning bij het beslissen wanneer welke axioma's moeten worden gebruikt en wanneer welke beperkingen. Daarom moet het nemen van ontwerpbeslissingen met betrekking tot begrenzingen transparant zijn om subjectieve discussies over het gebruik van axioma's versus het gebruik van beperkingen te limiteren.

Zusammenfassung

Bereits seit Jahrhunderten verarbeitet die Menschheit Informationen, aber das Aufkommen der Digitaltechnik und insbesondere das World Wide Web führen zu neuen Herausforderungen, um die stetig wachsende Menge an Informationen und Daten systematisch und intelligent zu verwalten und zu gebrauchen. Andernfalls verbleiben nur große Mengen nicht verbundener Daten von unbekannter Qualität.

Informationen können einheitlich repräsentiert werden, um eine intelligente Datenverwaltung zu erreichen. Zusätzlich können Beschränkungen definiert werden, die sinnvolle und in Anwendungen gültige Datenkombinationen ausdrücken. Man benötigt lediglich zwei Dinge um Informationen auf einfache aber leistungsfähige Weise zu repräsentieren: Konzepte und Verbindungen zwischen Konzepten. Dadurch erhalten wir eine Graphstruktur mit Konzepten als Knoten und Verbindungen welche Konzepte verbinden als Kanten, ein sogenannter Wissensgraph. Damit kann man beispielsweise die drei Konzepte Äutor", "Personünd "Buchßowie die Verbindungen ßchreibenünd "kaufen"repräsentieren. Die Information, dass der Autor Ändy Weir"das Buch "The Martian"geschrieben hat ist sinnvoll, wohingegen das Buch "The Martian"nicht den Autor Ändy Weirßchreiben kann. Für einen Computer sind jedoch beide Beispiele gültig wenn keine Beschränkungen definiert sind, die mögliche Verbindungen zwischen Konzepten limitieren.

Beschränkungen um Kontextbezogenen Sinn oder Qualität auszudrücken sind subjektiv und müssen von Menschen definiert werden. Im gegebenen Beispiel könnte eine Beschränkung sein, dass Autoren Bücher schreiben, und dass ein Autor ebenfalls eine Person ist. In so einem Fall wird eine Beschränkung auch Axiom genannt: eine gültige Wahrheit bezüglich eines Datenmodells. Diese Beschränkungen können von einem Computer verwendet werden um neues Wissen abzuleiten: basierend auf dem Wissen, dass Ändy Weir"das Buch "The Martian"geschrieben hat, kann abgeleitet werden, dass Andy Weir ein Autor und eine Person ist. Eine andere Beschränkung könnte sein, dass nur Autoren Bücher schreiben, und dass alle Bücher in einer Datenbank Autoreninformationen nötig haben. In so einem Fall wird eine Beschränkung auch Constraint genannt: eine Beschränkung um ungültige Daten zu identifizieren. Constraints können daher für eine Qualitätsprüfung verwendet werden, um beispielsweise fehlende Autoreninformationen oder inkorrekte Daten zu identifizieren.

Der Fokus dieser Dissertation liegt auf der Erstellung und Verwendung von Wissensgraphbe-

schränkungen. Das Definieren von abstrakten Konzepten wie Äutor", "Buch", oder ßchreiben"führt zu einem sogenannten Vokabular. Die Bedeutung der Begriffe im Vokabular kann durch Beschränkungen definiert werden. Man spricht dann auch von einer Ontologie. Beim Verbinden konkreter Daten in einem Wissensgraphen, wie zum Beispiel von Ändy Weiründ dem Buch "The Martian", spricht man von Daten welche Begriffe eines Vokabulars verwenden. Beispielsweise Ändy Weir ist ein Autoründ "The Martian ist ein Buch". Kontextbezogene Gültigkeit in diesem Beispiel kann durch Constraints beschränkt werden. Um all das Maschinengerecht zu repräsentieren, kann man die folgenden Sprachen verwenden die allesamt vom World Wide Web Consortium (W3C) empfohlen werden: (i) das Resource Description Framework (RDF) um Begriffe zu repräsentieren, (ii) das RDF Schema und die Web Ontology Language (OWL) um Axiome zu repräsentieren, und (iii) die Shapes Constraint Language (SHACL) um Constraints zu repräsentieren.

Die erste Problemstellung dieser Dissertation behandelt Benutzerunterstützung Bei der Bewertung von Wissensgraphen bezüglich Beschränkungen. Beim Erstellen eines Wissensgraphen werden existierende Vokabulare oft wiederverwendet. Das stellt sicher, dass ein System die Informationen eines anderen Systems verwenden kann. Diese Vokabulare verwenden häufig Axiome was wiederum die Wiederverwendbarkeit beeinflusst: Einige Axiome sind rechnerisch komplexer als andere und je nach Anwendungsfall möchte man die Verwendung von Vokabularen mit solchen Axiomen vermeiden. Auf ähnliche Weise kann es nötig sein die Verwendung von Constraints für Wissensgraphen festzustellen. Jedoch gibt es momentan in beiden Fällen nur limitierte Benutzerunterstützung beim Vergleichen und Auswählen von Wissensgraphen bezüglich verwendeter Beschränkungen.

Die zweite Problemstellung befasst sich mit Benutzerunterstützung bei der Erstellung von Constraints. In der Regel wissen Domänenexperten am besten welche Constraints sie definieren müssen. Leider sind sie jedoch nicht unbedingt Wissensgraphexperten. Aus diesem Grund wäre eine benutzerfreundliche Methode für die Erstellung von Constraints für Wissensgraphen hilfreich. Studien haben gezeigt, dass visuelle Notationen, die definieren wie verschiedene Konzepte visuell repräsentiert werden, Benutzer unterstützen. Gegenwärtig gibt es keine visuelle Notation um Constraints in Wissensgraphen zu visualisieren.

Da die Verwendung von Beschränkungen anwendungsspezifisch ist, konzentrieren wir uns in dieser Dissertation auf einen bestimmten Anwendungsfall für Data Stewardship: Die Unterstützung von Nationalbibliotheken bei der Präservierung von sozialen Medien. Einerseits müssen verschiedene heterogene Datenquellen bei der Präservierung von sozialen Medien berücksichtigt werden. Derzeit existiert jedoch kein vollständiger Workflow für die Archivierung von sozialen Medien, der die verschiedenen Daten sinnvoll zusammenführt. Andererseits muss auf präservierte Sammlungen sozialer Medien zugegriffen werden können, was Herausforderungen bezüglich subjektiver Constraints für Datenqualität darstellt. Beispielsweise benötigt jede Sammlung einen Titel und eine Beschreibung.

Um die erste Problemstellung bezüglich der Bewertung von Wissensgraphen anzugehen, präsentieren wir einen Ansatz um die Verwendung von Beschränkungen in Wissensgraphen zu messen. Außerdem präsentieren wir erfasste Statistiken welche die Häufigkeit von

Axiomen und Constraints beschreiben. Zuerst stellen wir Montolo vor, einen Ansatz um abstrakte Beschränkungsarten wie ßubclassünd deren konkrete Kodierung in RDF zu messen, i.e. `rdfs:subClassOf`. Danach präsentieren wir eine Implementierung dieses Ansatzes die interoperable Statistiken in RDF zur Verwendung von Beschränkungen erstellt. Wir demonstrieren die Durchführbarkeit dieses Ansatzes indem wir die Verwendung von Axiomen und Constraints messen. Dafür haben wir einerseits die Häufigkeit von RDFS und OWL Axiomen in mehr als Tausend Ontologien, die wir vom generischen Repository LOV und vom Domänenspezifischen Repository BioPortal extrahiert haben untersucht. Andererseits haben wir die Häufigkeit von Constraints in SHACL shapes untersucht, welche wir von ausgewählten GitHub Repositories extrahiert haben.

Wir adressieren die zweite Problemstellung mit Benutzerunterstützung bei der Erstellung von Constraints mittels visueller Notationen, welche alle Constraint Arten von SHACL unterstützt. Wir bauen auf existierenden, häufig verwendeten visuellen Notationen aus der Informatik und Wissensgraphdomäne auf und präsentieren die beiden visuellen Notationen ShapeUML und ShapeVOWL. Diese Notationen vergleichen wir anhand von Design-Prinzipen aus der Kognitionswissenschaft, da sie schlussendlich von menschlichen Benutzern verarbeitet werden. Wir evaluieren beide Notationen in einer vergleichenden Nutzerstudie.

Um die dritte Problemstellung bezüglich Datenverwaltung für die Präservierung von sozialen Medien anzugehen, stellen wir eine Wissensgraph-basierte Lösung mit dazugehöriger Qualitätsprüfung durch Constraints vor. Die im Rahmen dieser Dissertation entwickelte BESOCIAL Lösung basiert auf einer deklarativen Wissensgraphgenerierung: Gängige Vokabulare samt Axiomen werden verwendet um heterogene Daten von sozialen Medien sinnvoll zu zusammenzuführen. Außerdem stellen wir soziale Medien-bezogene Datenqualitätskategorien, -dimensionen und -metriken vor. Diese high-level Metriken messen wir mittels low-level Validierung mit Wissensgraph Constraints. Für das Definieren dieser Metriken haben wir eine etablierte Methodik verwendet, aber im Vergleich zu bestehenden Studien, beruht unsere Qualitätsprüfung auf Spezifikationen des World Wide Web Consortiums (W3C), anstatt auf maßgeschneiderter Software.

Die Beiträge dieser Dissertation bieten interoperable Lösungen zur Bewertung und Verwendung von Wissensgraphbeschränkungen.

Montolo ermöglicht es Benutzern existierende Wissensgraphen hinsichtlich der Verwendung von Axiomen und Constraints zu beurteilen. Die durchgeführte Studie lieferte das Ergebnis, dass Vokabulare vom generischen LOV Repository und domänenspezifischen BioPortal Repository ähnliche Verwendungsmuster für Axiome aufzeigen: Mehr als 95% der Vokabulare verwenden RDFS-basierte und lediglich die Hälfte OWL-basierte Beschränkungen. Die erstellten Statistiken können die Wiederverwendung von Ontologien unterstützen: Macher von Ontologien können bei der Bewertung von existierenden Ontologien im Rahmen derer Wiederverwendung nun Statistiken zur Verwendung von Axiomen zu Rate ziehen. Für Constraints haben wir Verwendungsmuster ähnlich zu Axiomen gefunden: Beziehungen zwischen Konzepten werden oft hinsichtlich bestimmter Klassen oder Datentypen

beschränkt, wohingegen Constraints für Textwerte (literal values) weitaus weniger Verwendung finden. Unsere Constraint Statistiken haben ein mögliches Problem aufgedeckt: eine sich selbst erfüllende Prophezeiung bei der Werkzeuge zum Erstellen von Constraints nur häufig verwendete Constraint Arten zur Verfügung stellen was dann zur Folge hat, dass wiederum mehr von diesen Constraint Arten existieren. Weniger gebräuchliche Constraint Aten haben mehr Aufmerksamkeit verdient

Die visullen Notationen ShapeUML und ShapeVOWL sind unabhängig von spezifischen Constraintsprachen und wurden basierend auf Design-Prinzipien aus der Kognitionswissenschaft entwickelt. Daher können Menschen ihr schnelles kognitives System nutzen und müssen sich nicht auf eine bestimmte Textsyntax verlassen. Der quantitative Teil unserer vergleichenden Analyse hat ergeben, dass Benutzer mit der einen oder anderen visuellen Notation nicht weniger Fehler machen, und dass mit beiden Notationen mehr als 80% der Fragen richtig beantwortet werden. Daher sehen wir für beide visuellen Notationen Potenzial für unterschiedliche Anwendungsfälle, unsere qualitative Analyse weist auch auf mögliche Verbesserungen hin.

Unser BESOCIAL-Workflow für die Archivierung von sozialen Medien ermöglicht es Experten des Kulturerbes, soziale Medien mit deklarativen Mitteln zu bewahren, also ohne dass sie Programmieren müssen. Darüber hinaus haben wir Qualitätskategorien, -dimensionen und -metriken für Sammlungen von Daten sozialer Medien definiert, die von der Community wiederverwendet werden können. Dieser Anwendungsfall veranschaulicht die Verwendung von sowohl Axiomen als auch Constraints, um Datenverwaltung zu ermöglichen und einen Mehrwert in Bezug auf Datenintegration und Datenqualität zu bieten. Die entwickelte Datenqualitätsbewertung kann auch für andere Anwendungsfälle verwendet werden, da sich unsere Lösung nur auf offen verfügbare W3C-bezogene Spezifikationen stützt.

Interessante Zukunftsperspektiven umfassen zum einen Bemühungen die Übernahme der vorgestellten visuellen Notationen voranzutreiben und zum anderen eine allgemeine Methodik zur Erstellung und Verwendung von Wissensgraphbeschränkungen.

In Bezug auf die Erstellung von Constraints sind die Ergebnisse unserer vergleichenden Auswertung der beiden visuellen Notationen mit Knowledge Graph-Experten ein erster Schritt in Richtung einer benutzerfreundlichen Unterstützung bei der Arbeit mit Knowledge Graph Constraints. Ähnliche Studien können mit Experten aus verschiedenen Anwendungsfeldern durchgeführt werden, um sowohl die visuellen Notationen als auch die Tools zur Implementierung der Notationen zu verbessern. Verschiedene Bearbeitungsworkflows können untersucht werden um die Tools zu verbessern. Darüber hinaus können zukünftige Studien Untersuchen wie andere Constraint-Sprachen als SHACL mit unseren visuellen Notationen verwendet werden können. Ein vielversprechender Kandidat ist die Shape Expression Language (ShEx), da sie in Communities, die mit Wikidata arbeiten, Gebrauch findet.

Es gibt mehrere Ontologie-Engineering-Methoden, aber insbesondere mit dem Aufkommen von Constraint-Sprachen wie SHACL entstanden neue Modellierungsparadigmen. In dieser Dissertation haben wir sowohl Axiome als auch Constraints für einen Anwendungsfall des digitalen Kulturerbes angewendet. Zukünftige Arbeiten könnten eine allgemeine Methodik

untersuchen, um Wissensingenieure bei der Erstellung von Wissensgraphen zu unterstützen. Eine Methodik zur Erstellung von Wissensgraphen die bei der Entscheidung unterstützt wann welche Axiome verwendet werden sollen und wann welche Constraints verwendet werden. Damit werden Entwurfsentscheidungen in Bezug auf Beschränkungen transparent gemacht, um subjektive Diskussionen über die Verwendung von Axiomen gegenüber der Verwendung von Beschränkungen zu minimieren.

List of acronyms

API: Application Programming Interface

CWA: Closed World Assumption

FAIR: Findable, Accessible, Interoperable, Reusable

HTTP: HyperText Transfer Protocol

HTML: Hypertext Markup Language

IRI: Internationalized Resource Identifier

JSON: JavaScript Object Notation

KG: Knowledge Graph

LOD: Linked Open Data

N3: Notation3

OWA: Open World Assumption

OWL: Web Ontology Language

RDF: Resource Description Framework

RDFS: RDF Schema

RML: RDF Mapping Language

SHACL: Shapes Constraint Language

ShEx: Shape Expressions

SPARQL: SPARQL Query Language

UI: User Interface

URI: Uniform Resource Identifier

VOWL: Visual Notation for OWL Ontologies

W3C: World Wide Web Consortium

XML: Extensible Markup Language

Chapter 1

Introduction

Humans process data already for centuries in an analog fashion, maintaining bibliographic catalogs for manuscripts is even predating printing [1]. The invention of printing as well as the opening of intercontinental sea routes in the 15th century created the concept of a global information system [2]. An example for analog information processing is the Universal Bibliographic Repertory from the Belgian information scientist Paul Otlet in 1895: knowledge from books was written and organized on index cards, and one could query this knowledge by sending questions via mail [3].

Representing data in a digital fashion enabled the automatic and fast processing of large amounts of data. A task such as the US census which took 8 years for the 1880 census could be done in only 2 years for the 1890 census using punching cards[1]: data was represented physically as holes in cards but read electronically by detecting holes. With the advent of computers, data was also stored electronically. One milestone was the invention of dedicated database management systems, most popularly relational databases in the 1970's [4], where data is stored in normalized form as a collection of tables. Other forms of data storage were introduced later, but relational databases are still commonly used nowadays.

Nowadays in the 21st century, humans almost constantly create digital data by using online applications, but integrating all this diverse data in a meaningful way is a challenge [5]. In 2000, not even 7% of the world population used the internet, in 2019 it was already more than 55%[2]. Additionally, nowadays people are online while on the move: in 2014, 48% of adult Europeans used the internet from mobile devices, in 2019, only 5 years later, it was already 73%[3]. All this use results in data, stored for example within (databases of) different

[1] IBM, "The Punched Card Tabulator", https://web.archive.org/web/20210814091059/https://www.ibm.com/ibm/history/ibm100/us/en/icons/tabulator/ (archived website accessed February 12, 2022)

[2] The World Bank, "Individuals using the Internet", https://web.archive.org/web/20211030232034/https://data.worldbank.org/indicator/IT.NET.USER.ZS (archived website accessed February 12, 2022)

[3] EuroStat, "Digital economy and society statistics - households and individuals", https://web.archive.org/web/20211027205416/https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Digital_economy_and_society_statistics_-_households_and_individuals#Internet_usage (archived website accessed February 12, 2022)

applications. From an economical perspective, the global system integration market reached 303.2 billion dollars in 2020[4] which may quantify the challenges in getting a uniform view of data across systems.

Knowledge Graphs and the Semantic Web have the potential to integrate heterogeneous data in a uniform way. Knowledge Graphs represent things as nodes and relationships between the things as edges which provides flexibility to represent heterogeneous data. The Semantic Web builds upon the Web, and in particular uses addresses in the Web to identify nodes and edges globally unique. Therefore, data and metadata can be represented uniformly in a graph structure within a global information system using unique identifiers.

However, arbitrarily linking things does not make sense and would not help to integrate heterogeneous data. Formalized restrictions need to be in place to define how data in a Knowledge Graph can be linked such that also a machine understands it. One important aspect of data integration with Knowledge Graphs is that existing concepts and relationships are reused, this expresses a shared understanding. Reusing Knowledge Graphs may require an assessment of used restrictions by users due to different characteristics of different restriction types. But currently there is not much support for users when using restrictions, limited both for the assessment of restrictions but even for the creation of restrictions.

This dissertation focuses on the use of Knowledge Graph restrictions. The dissertation first investigates how Knowledge Graphs can be assessed with respect to restrictions using interoperable statistics. Then, research findings related to how users can be supported in the creation of constraints for Knowledge Graphs using visual notations is presented. Finally, the dissertation focuses on a cultural heritage use case in which Knowledge Graph restrictions are used to integrate heterogeneous social media data and to assess its data quality. Therefore enabling so-called data stewardship on the heterogeneous data.

In the remaining of this chapter, terminology and background is introduced in Section 1.1. Research challenges which are tackled are presented in Section 1.2. Corresponding to these challenges, research questions and hypotheses are formulated as well as contributions outlined in Section 1.3. Section 1.4 summarizes the related work to identify existing gaps, in Section 1.5 publications are listed on which this dissertation is based as well as other publications during this PhD. Finally Section 1.6 outlines the chapters of this dissertation.

1.1 Background and Definitions

This section introduces fundamental technologies and methods this dissertation builds upon. First the (Semantic) Web as a technological foundation for this dissertation is introduced to the reader: it provides a global information exchange for humans and machines. Then this section presents how knowledge can be formally expressed for the vision of the Semantic Web using so-called Knowledge Graphs based upon the Resource Description Framework (RDF). More specifically, this section elaborates on the Web Ontology Language (OWL) and the Shapes Constraint Language (SHACL) which are both used to define restrictions

[4] imarcgroup, "System Integration Market", `https://web.archive.org/web/20210922211542/https://www.imarcgroup.com/system-integration-market` (archived website accessed February 12, 2022)

for Knowledge Graphs following different assumptions. Lastly, domains relevant to the use case of this dissertation are explained: the domains of social media archiving and quality assessment.

1.1.1 The Web

The methods and technologies used in this dissertation build upon a global information space: the World Wide Web (WWW) which runs on top of the internet. The internet provides a technical infrastructure of computer networks and standardized communication protocols such as TCP/IP, and the Web is a service on top of these networks[5]. The Web was invented in 1989 by Sir Tim Berners-Lee at CERN. For this dissertation the following definition of the Web is used:

Definition 1 (World Wide Web): *The World Wide Web is an information space in which the items of interest, referred to as resources, are identified by global identifiers called Uniform Resource Identifiers (URI) [6].*

1.1.2 Protocols of the Web

The Web follows a client-server model, where servers offer content and clients request that content. Most commonly, a resource is a website (document) offered by a Web server which can be requested by humans using a Web browser. A Web browser uses the HTTP protocol [7] to request the resource from the server, usually in HTML [8] for human consumption, and then the browser interprets and displays the received HTML content. Via hyperlinks a Web resource may link to other Web resources (Figure 1.1 left), but even though the popularity of a Web resource could be measured by the number of ingoing hyperlinks, no meaning is defined with respect to what the link actually means or which of the Web resource's content is referred to.

1.1.3 The Semantic Web

Instead of only linking documents on the Web (Figure 1.1 on the left), the Semantic Web is built on top of the Web and uses the same technical specifications to link data.

Definition 2 (Semantic Web): *The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning. [9]*

Data gets unique identifiers by using URIs[6] and thus the data can be dereferenced (Figure 1.1 on the right). Clients can use the Web's HTTP protocol to request the information, but instead of only using HTML, a client can request the same resource also in other formats which are more machine-friendly, e.g., the turtle format [10] (Figure 1.1 bottom right).

[5] Britannica, "Internet", `https://web.archive.org/web/20211016083506/https://www.britannica.com/technology/Internet` (archived website accessed February 12, 2022)

[6] URIs are limited to the US-ASCII character set, thus Internationalized Resource Identifiers (IRIs) were introduced, but in this thesis the term URIs is used as it is more common.

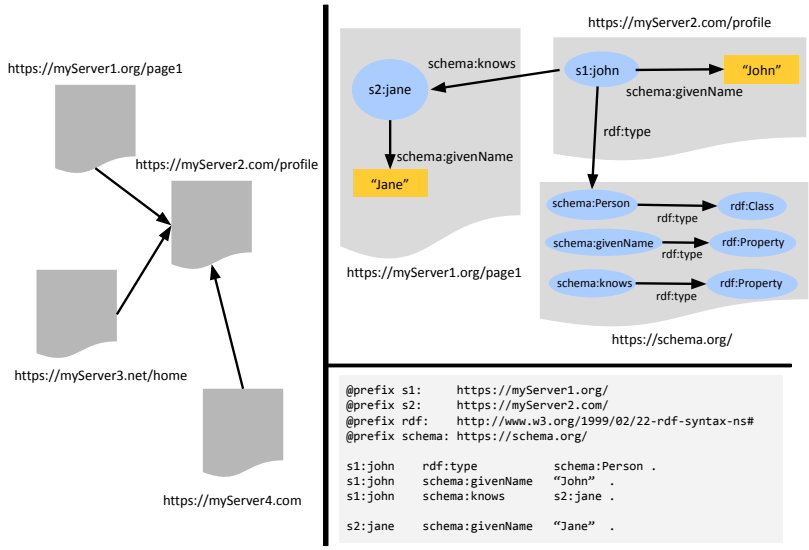


Figure 1.1: The Web of documents (left) and the Web of data (right): the Semantic Web builds upon the the Web and uses it to represent dereferencable subjects, predicates and objects within a graph model (top right) which can be serialized textually as triples (bottom right). URIs in the graph are shortened with a prefix, e.g. s1 instead of https://myServer1.org/, see prefix declaration on the bottom right.

1.1.4 Knowledge Graphs

Storing data in a graph structure allows integrating heterogeneous data. There is no formal definition of a "Knowledge Graph" yet, but Fensel et al. [11] provide an overview of definition efforts as well as examples of open and proprietary Knowledge Graphs. This dissertation uses the informal definition of Paulheim [12]:

Definition 3 (Knowledge Graph): *A Knowledge Graph (i) mainly describes real world entities and their interrelations in a graph structure, (ii) defines possible classes and relations of entities in a schema, (iii) allows for potentially interrelating arbitrary entities with each other, and (iv) covers various topical domains. [12]*

1.1.5 RDF

A way to represent knowledge in a graph is by using the Resource Description Framework (RDF) recommended by the World Wide Web Consortium (W3C).

Definition 4 (Resource Description Framework (RDF)): *RDF is a framework for representing information on the Web [13].*

RDF uses URIs which offer a solution to the data integration problem [5] and thus serves the vision of the Semantic Web. RDF is based on subjects which can link via predicates to objects (which may be other subjects, see Figure 1.2). These so-called triples form a graph in which every component of the triple may be uniquely identified using an URI [14]. Ideally these URIs are dereferencable and return a definition of that component also in RDF. The object of a triple may be a literal and not a URI, thus also literal data values can be represented such as the triple `s1:john schema:givenName "John"` as shown in Figure 1.2 (where parts of the URI are shortened with a prefix for readability).

1.1.6 Represent classes and relations within vocabularies

Following the presented Knowledge Graph definition, possible classes and relations of entities in a schema [12] may be also part of it. Such a schema which may define a domain can be described using RDF terms from RDFS [15].which then informally is usually referred to as a vocabulary[7].

Definition 5 (Vocabulary): *A vocabulary defines the concepts and relationships describing an area of concern*[8].

Definition 6 (RDF Schema (RDFS)): *RDF Schema provides a data-modelling vocabulary for RDF data [15].*

1.1.7 Express meaning with ontologies

Concepts and relationships of a Vocabulary may be linked based on some implicit meaning with different interpretations, this motivates a more formal data structure to address the vision of the Semantic Web, i.e. well-defined meaning (see definition 2). Ontologies can semantically represent a domain of interest with well-defined meaning by using axioms. Please note that within literature the terms "vocabulary" and "ontology" are often used interchangeably [8]. This dissertation builds upon definitions for ontologies from Gruber [16], Guarino et al. [17] as well as for RDF-related specifications from the W3C.

Definition 7 (Ontology): *An Ontology is a formal, explicit specification of a shared conceptualization [16].*

Definition 8 (Conceptualization): *A Conceptualization is an intensional semantic structure which encodes the implicit rules constraining the structure of a piece of reality [17].*

Definition 9 (Web Ontology Language (OWL)): *OWL 2 is a knowledge representation language, designed to formulate, exchange and reason with knowledge about a domain of interest. [18]*[9]

[7] Similarly, RDFS is informally also called a vocabulary

[8] W3C, "Vocabularies", `https://web.archive.org/web/20211223205759/https://www.w3.org/standards/semanticweb/ontology` (archived website accessed February 12, 2022)

[9] OWL2 is the successor of OWL, in this dissertation the term OWL is used to refer to OWL2.

Definition 10 (Axiom): *Axioms are statements that are asserted to be true in the domain being described [19].*

which specifically for OWL are defined as:

Definition 11 (Axioms in OWL): *Axioms are the basic statements that an OWL ontology expresses [18].*

The formal meaning of ontologies implemented in RDF graphs is defined by respective specifications about semantics, i.e. RDF Semantics and OWL RDF-based semantics compatible with the RDF Semantics:

Definition 12 (RDF Semantics): *The RDF Semantics are model-theoretic semantics for RDF graphs and the RDF and RDFS vocabularies, providing an exact formal specification of when truth is preserved by transformations of RDF or operations which derive RDF content from other RDF [20].*

Definition 13 (OWL2 RDFS-based semantics): *The OWL2 RDF-Based Semantics give a formal meaning to every RDF graph and is fully compatible with the RDF Semantics specification [21].*

By using the mentioned specifications based on these definitions, information on the Web can be represented using RDF-based Knowledge Graphs in a meaningful way.

1.1.8 Structural constraints

Even though formal meaning can be defined with the previously defined specifications, applications which have to exchange data may rely on local constraints to interact smoothly. For example, two applications processing information about persons may rely on the same formally defined specifications of what a person is. Yet, one application may require that persons have a birth date where for the other application this is not necessary. To cover such use cases so-called data shapes were introduced.

Definition 14 (Data shape): *Data shapes express "structural constraints to validate instance data" [22].*

Such data shapes can be expressed using the W3C-recommended Shapes Constraint Language (SHACL) specification [23] or Shape Expressions (ShEX) [24], both which have a significant intersection [5]. Both languages share a similar goal, but they follow different approaches [5]. The research in this dissertation focuses on SHACL for constraints because it is the W3C-recommended specification.

1.1.9 Restrictions

Depending on the use case a user may want to impose either axioms to restrict meaning by stating what is true (see definition 10) or constraints to restrict instance data by stating what is valid in a use case (see definition 14). Both axioms and constraints may be used in combination as well. Research performed for this dissertation involves both axioms and constraints to restrict RDF Knowledge Graphs. Therefore, in this dissertation the term restriction is used as overarching term for both axioms and constraints. Whenever something only applies to either axioms or constraints these terms are used in this dissertation.

1.1.10 The open and closed world assumptions

Axioms and constraints both follow different assumptions and hence support different use cases. Axioms in OWL2 follow the Open World Assumption (OWA) in which "what is not known to be true or false might be true" [25]. This is in contrast to constraints which follow the Closed World Assumption (CWA), in which "what is not known to be true, is false" [26]. Therefore, constraints can be used to detect invalid or missing data in a given dataset because that dataset is assumed to contain all information (Figure 1.2 bottom right).

For example, a restriction expressing that the `schema:knows` relationship connects two instances of `schema:Person` is interpreted differently with axioms and constraints (see Figure 1.2). If such a relationship connects an instance of `schema:Person` and another resource which is not a `schema:Person` an axiom will lead to new knowledge stating that the other resource is also a `schema:Person`. Yet the same restriction expressed as constraint will lead to an error message (probably what one would expect).

1.1.11 Linked Data

The above mentioned technologies and approaches are used to describe and link data, i.e. Linked Data. In 2006 the inventor of the Web, Sir Tim Berners-Lee, introduced four Linked Data principles[10]

- Use URIs for names of things
- Use HTTP URIs so that people can look up those names
- When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL)
- Include links to other URIs, so that they can discover more things

The last principle is especially important when considering Linked Data within the vision of the Semantic Web: including links to other URIs. This ensures that if for example

[10] Tim Berners-Lee, "Linked Data", `https://web.archive.org/web/20211124225515/https://www.w3.org/DesignIssues/LinkedData` (archived website accessed February 12, 2022)

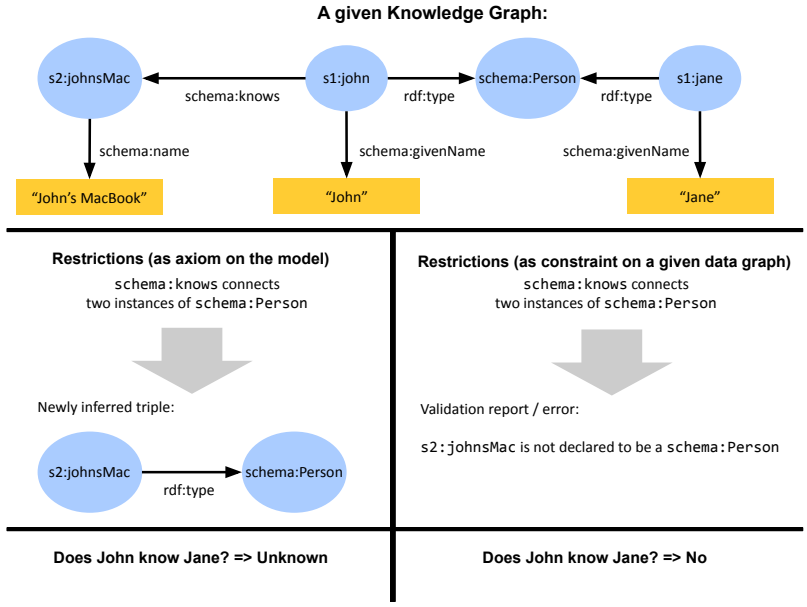


Figure 1.2: Restrictions expressed as axioms define meaning and may result in newly inferred triples (left) whereas restrictions expressed as constraints define what is valid/invalid in given data and may result in errors (right).

I declare the contacts of my address book to be instances of `schema:Person`, and someone else declares their contacts also using `schema:Person`, that we have a shared understanding of what we consider a person. And a tool programmed to display the `schema:familyName` of a `schema:Person`, could automatically display the name of my contacts. However, it is not just about the schema information, in this example concrete contacts have a URI too.

1.1.12 Linked Open Data

Linked Data could be used within one organization and never be shared. Another approach, added by Tim Berners-Lee in 2010 to the initial Linked Data article, is to open up this data. He proposed a five star rating system to encourage the publishing of Linked Data called Linked Open Data.

- **One star** Available on the Web (whatever format) but with an open licence, to be Open Data
- **Two stars** Available as machine-readable structured data (e.g. excel instead of image scan of a table)

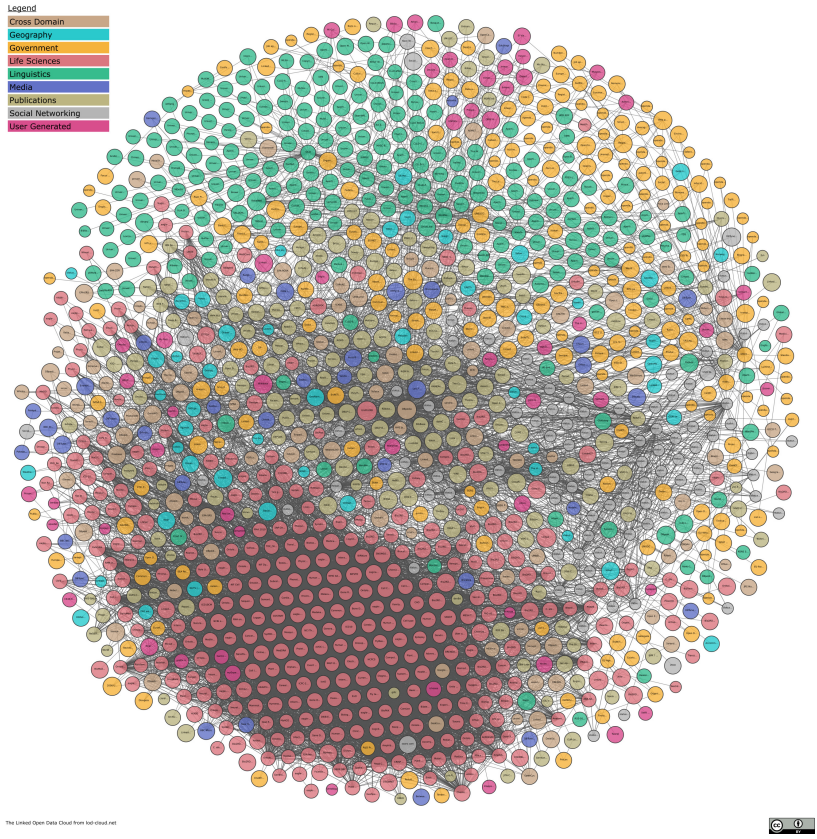


Figure 1.3: The LOD cloud as of December 2021: each node represents a dataset with potentially billion of RDF triples. Image obtained from lod-cloud.net

- **Three stars** as (2) plus non-proprietary format (e.g. CSV instead of excel)
- **Four stars** All the above plus, Use open standards from W3C (RDF and SPARQL) to identify things, so that people can point at your stuff
- **Five stars** All the above, plus: Link your data to other people's data to provide context

One famous initiative is the Linked Open Data cloud[11], established in 2007 with 12 datasets. As of May 2021 it contains 1,301 datasets with 16,283 links (see Figure 1.3). Different organizations from different domains provide data to this cloud. A large and domain-independent source of Linked Open Data which is commonly used within digital humanities,

[11] John P. McCrae, "The Linked Open Data Cloud", `https://web.archive.org/web/20211204035303/https://lod-cloud.net/` (archived website accessed February 12, 2022)

is Wikidata [27]. It is a collaborative platform in which both humans and machines can read and edit data.

1.1.13 FAIR data and data stewardship

The previously presented specifications and technologies are not a means to an end, but can serve data management, because information is represented in a meaningful way and available using open standards. Data management is not a goal in itself but the key conduit leading to knowledge discovery and innovation [28]. One term often used in this context is data stewardship which in this dissertation is informally defined based on Wilkinson et al.:

Definition 15 (data stewardship): *Beyond proper collection, annotation, and archival, data stewardship includes the notion of 'long-term care' of valuable digital assets, with the goal that they should be discovered and re-used for downstream investigations, either alone, or in combination with newly generated data [28].*

To support such data stewardship the following four guiding principles for scientific data management and stewardship were presented by Wilkinson et al. [28]: Findable, Accessible, Interoperable and Reusable (FAIR).

- To be Findable:
  - F1. (meta)data are assigned a globally unique and persistent identifier
  - F2. data are described with rich metadata (defined by R1 below)
  - F3. metadata clearly and explicitly include the identifier of the data it describes
  - F4. (meta)data are registered or indexed in a searchable resource
- To be Accessible:
  - A1. (meta)data are retrievable by their identifier using a standardized communications protocol
  - A1.1 the protocol is open, free, and universally implementable
  - A1.2 the protocol allows for an authentication and authorization procedure, where necessary
  - A2. metadata are accessible, even when the data are no longer available
- To be Interoperable:
  - I1. (meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.
  - I2. (meta)data use vocabularies that follow FAIR principles
  - I3. (meta)data include qualified references to other (meta)data

- To be Reusable:
  - R1. meta(data) are richly described with a plurality of accurate and relevant attributes
  - R1.1. (meta)data are released with a clear and accessible data usage license
  - R1.2. (meta)data are associated with detailed provenance
  - R1.3. (meta)data meet domain-relevant community standards

1.1.14 Social media archiving

This dissertation covers the archiving of social media data from social media providers as a form of digital preservation. Within this dissertation, social media data is considered to be the content users share online, more specifically text. For human consumption, this content is made available via the websites of the respective social media provider such as the company Twitter[12]. For machine consumption, this content is sometimes made available via Application Programming Interfaces (APIs). Social media archiving consists of methods to obtain data from social media providers to preserve it for the future. This is necessary, because the long term future of such companies is not certain, which in turns means uncertainty for the availability of their valuable social media data.

1.1.15 Data quality assessment

Parts of this dissertation concern the assessment of data quality. Data quality is subjective [29, 30], one has to define what good quality *is* in a given context. This dissertation considers data quality terminology from the Data on the Web Best Practices Working Group [30]. A *quality dimension* represents criteria for assessing quality. One or more *quality metrics* can be used to measure a dimension. Furthermore, dimensions can be grouped into *quality categories*. These generic principles can be used to define quality for a given use case. For example, one may have a UI dashboard to browse collections, each collection should have a human readable description of sufficient length. This could be described by stating the category *UI dashboard* with the dimension *collection description*, measured by the metrics *description available* and *minimum length description*.

1.2 Research Challenges

This dissertation investigates three challenges regarding the practical use of Knowledge Graph restrictions. Firstly, the assessment of existing Knowledge Graphs for reuse based on restrictions. Secondly, the support of humans in the creation of Knowledge Graph restrictions by using visual notations. And thirdly, the use of restrictions to support data stewardship tasks.

[12] Twitter, "Twitter", `https://web.archive.org/web/20220129003015/https://twitter.com/` (archived website accessed February 12, 2022)

Challenge 1: Assessing the use of restrictions in Knowledge Graphs.

The reusability of Knowledge Graphs is influenced by how, which types of restrictions are used. On the one hand, ontologies are used in the Semantic Web to represent real-world domains and concepts [31] as a Knowledge Graph. When reusing ontologies one has to consider different types of restrictions because they have a different computational complexity [32] and reuse costs in terms of effort one has to spend [33]. On the other hand, data in a Knowledge Graph may be restricted with constraints of different type, e.g. of the recently recommended W3C Shape Constraint Language (SHACL) [23]. However, for both cases there are currently no restriction-related information available which could support an assessment.

Challenge 2: Supporting users in the creation of KG constraints.

Support for the creation of constraints from more recent constraint languages is limited. Recent standardization efforts resulting in the W3C recommendation SHACL to express Knowledge Graph constraints paved the way for a broad use of interoperable constraints. The creation of constraints is often the task of domain experts which are not necessarily Knowledge Graph experts, currently they have to be familiar with the particular textual syntax of constraint languages such as SHACL. User evaluations of different Knowledge Graph concepts suggest that visualizations support users to perform respective tasks more intuitively [34, 35]. However, to date there is no standardized way to visualize all SHACL core constraints which impedes interoperability regarding user support.

Challenge 3: Enabling data stewardship using restrictions.

Use case specific needs for data stewardship can be generalized to the challenge of providing a meaningful representation of the data and performing a quality assessment of those data with respect to use case requirements. If the application has to deal with heterogeneous data, these data need to be integrated in order to provide such a meaningful representation. Knowledge Graphs offer a solution to this general challenge, but on the one hand they need to be generated first and on the other hand, the use case specific quality needs to be described and assessed.

1.3 Research Questions and Hypotheses

The aforementioned challenges motive the main research question of this dissertation:

How can we support users in the assessment and in the creation of Knowledge Graph restrictions?

To answer this question, this dissertation presents related work with respect to Knowledge Graph assessments where users in the role of ontology engineers need to reuse existing ontologies. Identified gaps are filled with an approach to measure the use of restrictions, and with FAIR quality metrics describing restriction use in a large Knowledge Graph corpus. Then, the dissertation discusses related work regarding user support in the creation of Knowledge Graph constraints by means of visual notation specifications. Existing gaps are filled by providing two visual notations which cover all SHACL core constraints. Lastly, the dissertation investigates the use of Knowledge Graph restrictions to enable data stewardship

in a particular cultural heritage use case, because the use of restrictions to define meaning or assess quality is subjective.

Research Question 1: *How can we support the assessment of restrictions in existing Knowledge Graphs?*

Hypothesis 1: *FAIR statistics of RDF encoded axioms and constraints enable restriction use assessments of several existing Knowledge Graphs not possible with state of the art tools.*

Contribution 1: *I developed Montolo, an approach to specify restriction types and measure their use to address the first challenge. The feasibility of this approach was demonstrated by applying it for (i) restrictions expressed using OWL restriction types for 660 ontologies from the repository Linked Open Vocabularies (LOV) [36] and 656 ontologies from the repository BioPortal [37], as well as (ii) restrictions expressed using SHACL core constraint types [23] for data shapes identified in 19 GitHub repositories. Montolo covers a common subset of axiom types (see Section 1.4) and different syntactical expressions of it to identify used modeling patterns. Montolo does not aim to be complete, but rather be extensible.*

Research Question 2: *How can we support users familiar with Linked Data in viewing RDF constraints?*

Since studies with respect to Visual Notation for OWL Ontologies (VOWL)-based visualizations for different RDF specifications report to perform respective tasks more intuitively [34, 35]. the following hypothesis is investigated:

Hypothesis 2: *Users familiar with Linked Data can answer questions about visually represented RDF constraints more accurately with a VOWL-based visual notation than with an UML-based visual notation*

Contribution 2: *Different visual notations to work with Knowledge Graphs already exist[34, 38]. To address the third challenge, two visual notations were extended to represent Knowledge Graph constraints. Commonly used visual notations were adapted for Knowledge Graph-related concepts to represent all SHACL core constraints. This allows a fair comparison between both notations, something which could not be done with state of the art tools because they do not visualize all SHACL core constraints. For the adaptation of the visual notations, cognitive effective design principles [39] were followed where possible. A conducted comparative user study revealed that there is no statistically significant difference in mean error rates between both visual notations.*

Research Question 3: *How can axioms and constraints support archiving institutions in the data stewardship of heterogeneous social media data?*

Hypothesis 3: *The W3C-recommended constraint language SHACL can be used to declaratively assess data quality metrics for use case specific data quality of heterogeneous social media data, integrated into an RDF graph with formal meaning*

Contribution 3: *Different publications of this PhD contribute Knowledge Graph generation for different use cases. Heterogeneous data was lifted to RDF where the data is represented using formal meaning by choosing appropriate ontologies. The assessment and assurance of use case specific data quality was out of the scope for the research projects related to learning analytics [40] and advertisement targeting [41]. However, research activities in the BESOCIAL use case about the preservation of social media content involved both heterogeneous data and a need for data quality assurance. To this end, this dissertation contributed a workflow to archive social media collections and their provenance by applying Knowledge Graph generation. Domain specific ontologies such as the PREMIS Data Dictionary for Preservation Metadata*[13]*. and domain independent ontologies such as the W3C recommended Provenance Ontology (PROV) [42] were used. This enables data stewardship because use case specific tasks such as querying specific heterogeneous data based on their provenance is possible. Additionally, the methodology of Rula and Zaveri [43] for data quality assessment was applied in a declarative way by defining quality dimensions and metrics using the Data Quality Vocabulary (DQV) [30] and measuring the metrics using related SHACL constraints which were developed.*

1.4 Related Work

The main research question of this dissertation concerns the assessment of Knowledge Graph restrictions as well as the creation of Knowledge Graph constraints, in both cases by human users. This section emphasizes briefly on related research efforts. Particularly, this section discusses related work regarding (i) restriction types, to elaborate on what to assess; (ii) existing Knowledge Graph assessments, to show which approaches already exist; (iii) current user support for restriction assessment, and (iv) current user support for the creation of restrictions, to show the current gap in user support. Please note that more detailed related work is discussed in the respective chapters of this dissertation.

1.4.1 Restrictions

This section discusses related work regarding different identified types of restrictions. It aims to show what can be measured with respect to restrictions.

Hartmann [44] investigated different constraint languages for RDF and published a set of 81 constraint types, independently of a specific restriction language. He emphasized that even though OWL is not a constraint language, it is often used as such in practice under the Closed World Assumption. How OWL can be used as a constraint language is for example shown by Motik et al. [45, 46] or Sirin and Tao [47]. Hartmann compared which constraint type can be expressed with which constraint language [48] and he also considered OWL. Therefore, for the rest of this dissertation his identified constraint types are referred to as restriction types according to the terms used in this dissertation (see Section 1.1.9). Not all

[13] Library of Congress, "PREMIS", `https://web.archive.org/web/20211009123549/http://www.loc.gov/standards/premis/ontology/owl-version3.html` (archived website accessed February 12, 2022)

restriction types can be modeled with each investigated language, e.g. some literal-value related restrictions cannot be expressed with OWL [48].

Kontokostas et al. [49] developed the test-driven evaluation framework RDFUnit using test patterns queried via SPARQL. Several test patterns cover aspects such as cardinality, disjointness or literal value restrictions. Arndt et al. [50] provided an alignment between the previously mentioned test patterns and identified restriction types of Hartmann.

Previous works identified different restriction types, this dissertation focuses on the assessment of both axioms and constraints. With respect to axioms, the previously mentioned alignment by Arndt et al. is considered a relevant subset of restriction types for this dissertation. With respect to constraints, the core constraints of SHACL are considered in this dissertation because it became a W3C recommendation in 2017.

1.4.2 Knowledge Graph assessment

This section discusses existing approaches to assess Knowledge Graphs, therefore listing what exists and where gaps exist this dissertation aims to fill. Knowledge Graphs are usually assessed with respect to their reuse potential or for data quality, but not specifically based on restriction use.

According to reuse, this dissertation considers the reuse of ontologies by users which are often referred to as ontology engineers. This reuse process usually follows a four step workflow to discover, select, customize and integrate potential reuse candidates [51]. Assessment is particularly important for the first two steps, the discovery and selection of reuse candidates. The BioPortal recommender service [52] provides functionality to discover ontologies relevant for a use case. However, that is achieved by matching content needs with available ontology concepts and does not consider restrictions which might be relevant for a use case too. The OOPS! [53] framework supports users in detecting anomalies and bad practices, thus supporting a qualitatively assessment.

Quality assessments are often performed on single Knowledge Graphs containing mostly instance data, recently also in the context of decentralized systems [54]. Zaveri et al. [55] presented a detailed systematic review on quality assessment for Linked Data. They identified a list of 18 quality dimensions and 69 metrics. Based on those findings, Rula and Zaveri [43] proposed a general data quality assessment methodology. Identified metrics can be measured using tools such as Luzzu [56], Loupe [57] or RDFUnit [49]. Other metrics were introduced in the RDFStats framework [58] and reused in the streaming-based solution LODStats [**Auer2010LODStats**]. Instance data can be large which motivated the scalable framework DistLODStats [59] which was later integrated into the SANSA stack [60, 61]. In 2020, OWLStats [62] was integrated into SANSA which computes statistics over OWL ontologies[14] However, the focus and evaluation of the SANSA stack is on scalability and they did not report restriction use statistics.

Previous works contributed tools which can be used to perform assessments. However, by the time of writing and publishing the assessment chapter of this dissertation, no tool

[14] their work was published after the work of Chapter 2.

supported the assessment of restrictions. Created statistics are usually from a dataset point-of-view, resulting in mixed statistics of all ontologies used in a dataset and, thus, are not considered useful in this dissertation for an ontology reuse scenario.

1.4.3 User support for the assessment of Knowledge Graph restrictions

Whereas the previous section discussed tools for the general assessment of Knowledge Graphs, this section elaborates on the user support particularly for Knowledge Graph restriction assessment.

The tool Widoco [63] produces enriched HTML documentation of ontologies by reusing among others the Live OWL Documentation Environment (LODE) [64], the quality framework OOPS! [53] and the WebVOWL [65] interactive visualization of ontologies. WebVOWL visualizes OWL axioms, therefore Widoco produces documentation which can be used for restriction assessment and is tailored for human users. However, this only allows an assessment of the single ontology this documentation was created for.

The Protégé [66] editor provides summaries about used axioms in an ontology, but these summaries only cover a fixed set of axioms. Plugins may provide more complete summaries or visualizations of used axioms, but similarly to documentation generated by Widoco, an assessment is only possible for the ontology currently loaded.

Online available ontology catalogs such as LOV [36] or BioPortal [37] provide basic statistics about ontologies in their catalog. These statistics are the basis of provided search and filter capabilities for users. However, these statistics do not cover different restriction types. In case of LOV basic statistics such as the number of used classes and properties is shown. Similar statistics are shown within BioPortal together with other metrics such as the maximum depth of the hierarchy tree or the average number of sibling concepts.

Existing tools either provide restriction information only for a single currently loaded ontology, or only provide limited statistics about several ontologies. No restriction use information is readily available to support users in the assessment of Knowledge Graph restrictions.

1.4.4 User support for the creation of Knowledge Graph restrictions

This dissertation considers visualizations and editing tools as support for restriction creation by users. In the following, related work is discussed showing that here are plenty of options for the creation of axioms, but not yet for constraints.

The probably most famous tool to create ontologies is Protégé [66] and since axioms are the basic statements of an OWL ontology (see definition 11) this tool can be used to create axioms. A recent study [67] used WebProtégé [68] as collaborative tool to create a taxonomy for the company Pinterest. Non ontology experts used the tool to create, update and maintain a large taxonomy realized using OWL.

There are plenty of visualization tools for ontologies, several surveys and other works [69, 70, 71, 72, 73, 74, 75, 76] identified an overlap of 84 ontology visualization tools in total.

With respect to constraint languages, which were introduced more recently, less works focused on user support for the creation of constraints. A limited number of constraints can be generated from UML diagrams [77] or from existing ontological axioms [78]. Tools to create RDF constraints exist, but do not provide a visual notation that covers all SHACL core constraints [79, 80, 81].

Tools to support users in the creation of axioms exist, but are sparse for recently proposed constraint languages: no tool supports users in the generation of all SHACL core constraints and provides a clearly specified visual notation.

1.5 Publications

The work presented in this dissertation is based on four peer-reviewed publications in international scientific journals and conference proceedings which are listed below in chronological order. My contributions to these four articles is as follows: As a first author, I devised the solution under guidance of my supervisors, was responsible to carry out the research, occasionally delegated development tasks to co-authors and wrote the articles which I eventually revised after rounds of both internal and external peer reviews.

- Sven Lieber et al. "MontoloStats – Ontology Modeling Statistics". In: *Proceedings of the 10th International Conference on Knowledge Capture - K-CAP '19*. Ed. by Raphaël Troncy. ACM, Nov. 2019, pp. 69–76. DOI: 10.1145/3360901.3364433
- Sven Lieber et al. "Statistics about Data Shape Use in RDF Data". In: *Proceedings of the 19th International Semantic Web Conference: Posters, Demos, and Industry Tracks*. Ed. by Kerry Taylor et al. Vol. 2721. CEUR Workshop Proceedings. Nov. 2020, pp. 330–335. URL: http://ceur-ws.org/Vol-2721/paper584.pdf
- Sven Lieber et al. "BESOCIAL: A Sustainable Knowledge Graph-Based Workflow for Social Media Archiving". In: *Further with Knowledge Graphs*. IOS Press, 2021, pp. 198–212
- Sven Lieber et al. "Visual Notations for Viewing RDF Constraints with UnSHACLed [to be published]". In: *Semantic Web Journal* Pre-press (Nov. 2021), pp. 1–36. DOI: 10.3233/SW-210450

Besides these publications, I (co-)authored several other publications throughout my PhD which are listed below. This list contains two other journals which I co-authored and eight conference publications, for which I was first author in five.

1.5.1 Publications in International Journals

I co-authored the following two journals. The first journal relates to another research topic which I investigated during my Master's thesis and is not related to this dissertation [86]. Work performed for the second journal relates to the BESOCIAL use case which is also

presented in this dissertation. However, the listed journal covers the topic of social media archiving as a whole and investigates based on a survey how international institutions tackle social media archiving currently [87]. Whereas this dissertation focuses on technical aspects to enable data stewardship.

- Io Taxidou et al. "Web-scale provenance reconstruction of implicit information diffusion on social media". In: *Distributed and Parallel Databases* 36.1 (Oct. 2017), pp. 47–79. DOI: 10.1007/s10619-017-7211-3

- Eveline Vlassenroot et al. "Web-archiving and social media: an exploratory analysis". In: *International Journal of Digital Humanities* (2021), pp. 1–22

1.5.2 Publications in International Conference Proceedings

Research performed for the following publications mostly relates to Knowledge Graph generation, with the exception of the first publication which relates to the research line investigated during my Master's thesis and early PhD months, not covered in this dissertation [88].

- Sven Lieber et al. "ProvDIVE: PROV Derivation Inspection and Visual Exploration". In: *Proceedings of the 9th USENIX Conference on Theory and Practice of Provenance (TaPP'17)*. Ed. by Adam Bates and Bill Howe. Seattle, WA, USA: USENIX Association, June 2017, p. 6

- Ben De Meester et al. "Interoperable User Tracking Logs using Linked Data for improved Learning Analytics". In: *Proceedings of the 19th International CALL Research Conference*. Antwerp: Universiteit Antwerpen, 2018. ISBN: 9789057285943. URL: https://biblio.ugent.be/publication/8575501

- Sven Lieber et al. "Linked Data Generation for Adaptive Learning Analytics Systems". In: *7th International Workshop on Learning and Education with Web Data (LILE2018)*. Ed. by Stefan Dietze et al. Amsterdam, The Netherlands, May 2018, pp. 23–26. URL: https://websci18.webscience.org/wp-content/uploads/2018/01/WebSci18_Events_PreProceedings-4-Linked_Learning_2018-lres.pdf

- Sven Lieber, Anastasia Dimou, and Ruben Verborgh. "SeGoFlow: A Semantic Governance Workflow Tool". In: *Lecture Notes in Computer Science*. Springer International Publishing, 2018, pp. 95–99. DOI: 10.1007/978-3-319-98192-5_18

- Sven Lieber. "Policy-compliant Data Processing: RDF-based Restrictions for Data-protection". In: *Proceedings of the Doctoral Consortium*. 2019, pp. 69–80. URL: http://ceur-ws.org/Vol-2548/paper-07.pdf

- Sven Lieber et al. "EcoDaLo: Federating Advertisement Targeting with Linked Data". In: 2020, pp. 87–103. DOI: 10.1007/978-3-030-59833-4_6

- Dörthe Arndt et al. "Dynamic Workflow Composition with OSLO-steps". In: *Proceedings of the 11th on Knowledge Capture Conference*. ACM, Dec. 2021, pp. 257–260. DOI: 10.1145/3460210.3493559
- Thomas Delva et al. "RML2SHACL: RDF Generation Taking Shape". In: *Proceedings of the 11th on Knowledge Capture Conference*. ACM, Dec. 2021, pp. 153–160. DOI: 10.1145/3460210.3493562

Knowledge Graph generation based on JSON-LD [94] was investigated in a learning analytics use case [89, 40], where I performed implementation work as well as research on how interoperable provenance information becomes relevant to assess data processing with respect to privacy. I performed more research on provenance relevant for privacy in form of a visual editor for data processing workflows [90] and in an early presentation about my PhD topic at a Doctoral Consortium [91]. Knowledge Graph generation in the context of federated querying was investigated within a research project on advertisement targeting [41]. Finally, I contributed my expertise with the constraint language SHACL in a project about improving customer journeys when interacting with public services [92] and in the generation of SHACL constraints from RDF mapping rules [93].

1.6 Outline

The remainder of the dissertation consists of three chapters addressing the different research questions and is based on the four peer-reviewed publications which contribute to my PhD (see Figure 1.4) as well as a conclusion chapter. Chapter 2 focuses on the assessment of restrictions and addresses Research Question 1. Specifically, the Montolo approach is introduced to create FAIR statistics of restriction use for RDFS/OWL axioms and SHACL constraints. Chapter 3 focuses on supporting human users in the creation of Knowledge Graph constraints and addresses Research Question 2. Specifically, two visual notations for Knowledge Graph constraints and results of a comparative user study are presented. Chapter 4 focuses on the use of Knowledge Graph restrictions to enable data stewardship and addresses Research Question 3. Specifically, this dissertation presents the Knowledge Graph-based BESOCIAL workflow for social media archiving and how related data quality can be defined and measured in a declarative way using Knowledge Graph constraints. In Chapter 5, this work is concluded and future research opportunities are discussed.

References

[1] Christine L Borgman. *Scholarship in the digital age: Information, infrastructure, and the Internet*. MIT press, 2010.

[2] A. Neelameghan and J. Tocatlian. "International cooperation in information systems and services". In: *Journal of the American Society for Information Science* 36.3 (May 1985), pp. 153–163. DOI: doi:10.1002/asi.4630360305.

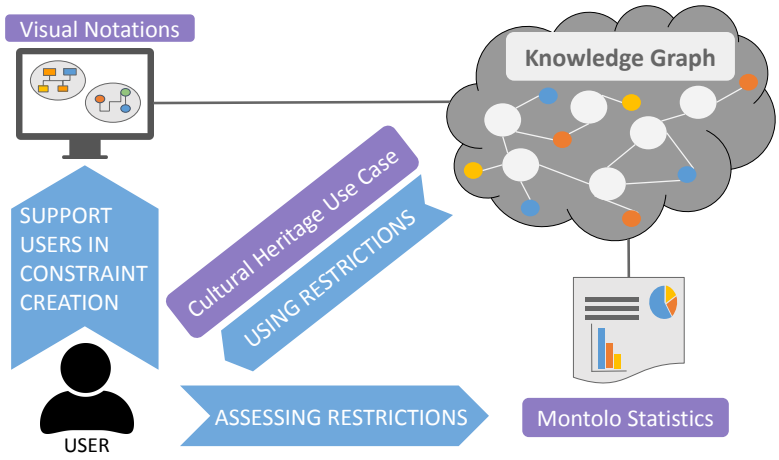


Figure 1.4: Overview of this dissertation with chapters as blue arrows and contributions in purple: interoperable solutions to (i) assess the use of restrictions (Chapter 2), (ii) support users in the creation of constraints with visual notations (Chapter 3), and (iii) enable data stewardship with restrictions in a cultural heritage use case (Chapter 4).

[3] Alex Wright. *Cataloging the World: Paul Otlet and the Birth of the Information Age*. Oxford University Press, 2014. ISBN: 978-0-19-993141-5.

[4] E. F. Codd. "A relational model of data for large shared data banks". In: *Communications of the ACM* 13.6 (June 1970), pp. 377–387. DOI: 10.1145/362384.362685.

[5] Jose Emilio Labra Gayo, Eric Prud'hommeaux, Iovka Boneva, and Dimitris Kontokostas. *Validating RDF Data*. Vol. 7. Synthesis Lectures on the Semantic Web: Theory and Technology 1. Morgan & Claypool Publishers LLC, Sept. 2017, pp. 1–328. DOI: 10.2200/s00786ed1v01y201707wbe016. URL: http://book.validatingrdf.com/.

[6] Ian Jacobs and Norman Walsh. *Architecture of the World Wide Web, Volume One*. Recommendation. World Wide Web Consortium (W3C), Dec. 2004. URL: https://www.w3.org/TR/webarch/.

[7] R T Fielding, J Gettys, J Mogul, H Frystyk, L Masinter, P Leach, and T Berners-Lee. *Hypertext Transfer Protocol – HTTP/1.1*. RFC. RFC Editor, June 1999. URL: http://www.rfc-editor.org/rfc/rfc2616.txt.

[8] *HyperText Markup Language (HTML): Living standard*. Living Standard. Web Hypertext Application Technology Working Group (WHATWG), Oct. 31, 2019. URL: https://html.spec.whatwg.org/.

[9] Tim Berners-Lee, James Hendler, Ora Lassila, et al. "The Semantic Web". In: *Scientific American* 284.5 (May 2001), pp. 28–37. ISSN: 00368733. DOI: 10.1038/scientificamerican0501-34.

[10] David Beckett, Tim Berners-Lee, Eric Prud'hommeaux, and Gavin Carothers. *RDF 1.1 Turtle – Terse RDF Triple Language*. Recommendation. World Wide Web Consortium (W3C), Feb. 2014. URL: http://www.w3.org/TR/turtle/.

[11] Dieter Fensel, Umutcan Şimşek, Kevin Angele, Elwin Huaman, Elias Kärle, Oleksandra Panasiuk, Ioan Toma, Jürgen Umbrich, and Alexander Wahler. "Introduction: What Is a Knowledge Graph?" In: *Knowledge Graphs*. Springer International Publishing, 2020. Chap. 1, pp. 1–10. DOI: 10.1007/978-3-030-37439-6_1.

[12] Heiko Paulheim. "Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods". In: *Semantic Web Journal* 8.3 (Dec. 2016), pp. 489–508. ISSN: 2210-4968. DOI: 10.3233/SW-160218. URL: http://www.semantic-web-journal.net/content/knowledge-graph-refinement-survey-approaches-and-evaluation-methods.

[13] Richard Cyganiak, David Wood, and Markus Lanthaler. *RDF 1.1 Concepts and Abstract Syntax*. Recommendation. World Wide Web Consortium (W3C), Feb. 2014. URL: http://www.w3.org/TR/rdf11-concepts/.

[14] Tim Berners-Lee, R Fielding, and L Masinter. *Uniform Resource Identifier (URI): Generic Syntax*. RFC 3986. RFC Editor, Jan. 2005. URL: https://www.rfc-editor.org/rfc/rfc3986.txt.

[15] Dan Brickley and R. V. Guha. *RDF Schema 1.1*. Recommendation. World Wide Web Consortium (W3C), Feb. 2014. URL: http://www.w3.org/TR/rdf-schema/.

[16] Thomas R Gruber. "A translation approach to portable ontology specifications". In: *Knowledge acquisition* 5.2 (June 1993), pp. 199–220. DOI: 10.1006/knac.1993.1008. URL: http://www.sciencedirect.com/science/article/pii/S1042814383710083.

[17] Nicola Guarino, Stati Uniti, and Pierdaniele Giaretta. "Ontologies and knowl-edge bases: towards a terminological clarification". In: *Towards Very Large Knowledge Bases*. IOS Press, 1995, pp. 25–32.

[18] Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter F. Patel-Schneider, and Sebastian Rudolph. *OWL 2 Web Ontology Language – Primer (Second Edition)*. Recommendation. World Wide Web Consortium (W3C), Dec. 2012. URL: http://www.w3.org/TR/owl2-primer/.

[19] Conrad Bock, Achille Fokoue, Peter Haase, Rinke Hoekstra, Ian Horrocks, Alan Ruttenberg, Uli Sattler, and Michael Smith. *OWL 2 Web Ontology Language – Structural Specification and Functional-Style Syntax (Second Edition)*. Recommendation. World Wide Web Consortium (W3C), Dec. 2012. URL: http://www.w3.org/TR/owl2-syntax/.

[20] Patrik J. Hayes and Peter F. Patel-Schneider. *RDF 1.1 Semantics*. Recommendation. World Wide Web Consortium (W3C), Feb. 2014. URL: http://www.w3.org/TR/rdf11-mt/.

[21] Michael Schneider. *OWL 2 Web Ontology Language RDF-Based Semantics (Second Edition)*. Tech. rep. https://www.w3.org/TR/2012/REC-owl2-rdf-based-semantics-20121211/. World Wide Web Consortium (W3C), Dec. 2012. URL: https://www.w3.org/TR/owl2-rdf-based-semantics/.

[22] Simon Steyskal and Karen Coyle. *SHACL Use Cases and Requirements*. Tech. rep. https://www.w3.org/TR/2017/NOTE-shacl-ucr-20170720/. W3C, July 2017.

[23] Holger Knublauch and Dimitris Kontokostas. *Shapes Constraint Language (SHACL)*. Recommendation. World Wide Web Consortium (W3C), July 2017. URL: https://www.w3.org/TR/shacl/.

[24] Eric Prud'hommeaux, Jose Emilio Labra Gayo, and Harold Solbrig. "Shape expressions: an RDF validation and transformation language". In: *Proceedings of the 10th International Conference on Semantic Systems*. Ed. by Harald Sack, Agata Filipowska, Jens Lehmann, and Sebastian Hellmann. ACM. New York, NY, United States: Association for Computing Machinery, 2014, pp. 32–40. DOI: 10.1145/2660517.2660523. URL: http://dl.acm.org/citation.cfm?id=2660523.

[25] C. Maria Keet. "Open World Assumption (OWA)". In: (2013), pp. 1567–1567. DOI: 10.1007/978-1-4419-9863-7_734.

[26] C. Maria Keet. "Closed World Assumption (CWA)". In: (2013), pp. 415–415. DOI: 10.1007/978-1-4419-9863-7_731.

[27] Denny Vrandečić and Markus Krötzsch. "Wikidata: A Free Collaborative Knowledgebase". In: *Commun. ACM* 57.10 (Sept. 2014), pp. 78–85. ISSN: 0001-0782. DOI: 10.1145/2629489. URL: https://doi.org/10.1145/2629489.

[28] Mark D. Wilkinson et al. "The FAIR Guiding Principles for scientific data management and stewardship". In: *Scientific Data* 3 (Mar. 2016), p. 160018. DOI: 10.1038/sdata.2016.18.

[29] J. M. Juran. *Juran's Quality Control Handbook*. Ed. by Frank M. Mryna. 4th. Texas, USA: Mcgraw-Hill, Aug. 1988. URL: http://www.pqm-online.com/assets/files/lib/books/juran.pdf.

[30] Jeremy Debattista, Makx Dekkers, Christophe Guéret, Deirdre Lee, Nandana Mihindukulasooriya, and Amrapali Zaveri. *Data on the Web Best Practices: Data Quality Vocabulary*. Working Group Note. World Wide Web Consortium, Dec. 2016. URL: https://www.w3.org/TR/vocab-dqv/.

[31] Elena Simperl and Christoph Tempich. "Ontology engineering: a reality check". In: *International Conference "On the Move to Meaningful Internet Systems"*. Springer. 2006, pp. 836–854. DOI: 10.1007/11914853_51.

[32] Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, and Carsten Lutz. *OWL 2 Web Ontology Language Profiles (Second Edition)*. Recommendation. World Wide Web Consortium (W3C), Dec. 2012. URL: https://www.w3.org/TR/owl2-profiles/.

[33] Elena Simperl, Christoph Tempich, and York Sure. "ONTOCOM: a cost estimation model for ontology engineering". In: *The Semantic Web - ISWC 2006*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 625–639. DOI: 10.1007/11926078_45.

[34] Steffen Lohmann, Stefan Negru, Florian Haag, and Thomas Ertl. "Visualizing ontologies with VOWL". In: *Semantic Web* 7 (May 2016), pp. 399–419. ISSN: 2210-4968. DOI: 10.3233/sw-150200.

[35] Pieter Heyvaert, Anastasia Dimou, Ben De Meester, Tom Seymoens, Aron-Levi Herregodts, Ruben Verborgh, Dimitrie Schuurman, and Erik Mannens. "Specification and implementation of mapping rule visualization and editing: MapVOWL and the RMLEditor". In: *Web Semantics: Science, Services and Agents on the World Wide Web* 49 (Mar. 2018), pp. 31–50. DOI: 10.1016/j.websem.2017.12.003. URL: https://biblio.ugent.be/publication/8559065/file/8559068.pdf.

[36] Pierre-Yves Vandenbussche, Ghislain A. Atemezing, María Poveda-Villalón, and Bernard Vatant. "Linked Open Vocabularies (LOV): a gateway to reusable semantic vocabularies on the Web". In: *Semantic Web Journal* 8.3 (Dec. 2016), pp. 437–452. DOI: 10.3233/SW-160213. URL: http://www.semantic-web-journal.net/content/linked-open-vocabularies-lov-gateway-reusable-semantic-vocabularies-web-1.

[37] M Musen, N Shah, N Noy, Benjamin Dai, Michael Dorf, N Griffith, JD Buntrock, Clement Jonquet, MJ Montegut, and Daniel L Rubin. "BioPortal: ontologies and data resources with the click of a mouse". In: *AMIA Annu Symp Proc*. Vol. 6. 2008, pp. 1223–1224.

[38] Stephen Cranefield and M. Purvis. "UML as an Ontology Modelling Language". In: *Intelligent Information Integration*. 1999.

[39] Daniel Moody. "The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering". In: *IEEE Transactions on Software Engineering* 35.6 (Nov. 2009), pp. 756–779. DOI: 10.1109/tse.2009.67.

[42] Timothy Lebo, Satya Sahoo, Deborah McGuinness, Khalid Belhajjame, James Cheney, David Corsar, Daniel Garijo, Stian Soiland-Reyes, Stephan Zednik, and Jun Zhao. *PROV-O: The PROV Ontology*. Recommendation. World Wide Web Consortium (W3C), Apr. 2013. URL: https://www.w3.org/TR/prov-o/.

[43] Anisa Rula and Amrapali Zaveri. "Methodology for Assessment of Linked Data Quality." In: *LDQ@ SEMANTICS*. 2014, p. 34.

[44] Thomas Hartmann. "Validation Framework for RDF-based Constraint Languages". PhD thesis. Karlsruher Institut für Technologie (KIT), 2016. DOI: 10.5445/ir/1000056458. URL: http://digbib.ubka.uni-karlsruhe.de/volltexte/1000056458.

[45] Boris Motik, Ian Horrocks, and Ulrike Sattler. "Adding Integrity Constraints to OWL". In: *OWL: Experiences and Directions* (Innsbruck, Austria). Ed. by Christine Golbreich, Aditya Kalyanpur, and Bijan Parsia. Vol. 258. CEUR Workshop Proceedings. CEUR-WS.org, 2007. URL: http://ceur-ws.org/Vol-258/paper11.pdf.

[46] Boris Motik, Ian Horrocks, and Ulrike Sattler. "Bridging the gap between OWL and relational databases". In: *Journal of Web Semantics* 7.2 (Apr. 2009), pp. 74–89. DOI: 10.1016/j.websem.2009.02.001.

[47] Evren Sirin and Jiao Tao. "Towards Integrity Constraints in OWL". In: *Proceedings of the 6th International Conference on OWL: Experiences and Directions – OWLED'09* (Chantilly, VA, United States). Ed. by Rinke Hoekstra and Peter F. Patel-Schneider. Vol. 529. CEUR Workshop Proceedings. Chantilly, VA: CEUR-WS.org, 2009, pp. 79–88. DOI: 10.5555/2890046.2890055. URL: http://dl.acm.org/citation.cfm?id=2890046.2890055.

[48] Thomas Hartmann. *Validation Framework for RDF-based Constraint Languages - PhD Thesis Appendix*. Tech. rep. Karlsruher Institut für Technologie (KIT), 2016. DOI: 10.5445/ir/1000054062. URL: http://digbib.ubka.uni-karlsruhe.de/volltexte/1000054062.

[49] Dimitris Kontokostas, Patrick Westphal, Sören Auer, Sebastian Hellmann, Jens Lehmann, Roland Cornelissen, and Amrapali Zaveri. "Test-driven evaluation of linked data quality". In: *Proceedings of the 23rd international conference on World Wide Web*. Ed. by Chin-Wan Chung. New York, NY, United States: Association for Computing Machinery, Apr. 2014, pp. 747–757. ISBN: 9781450327442. DOI: 10.1145/2566486.2568002. URL: http://dl.acm.org/citation.cfm?id=2568002.

[50] Dörthe Arndt, Ben De Meester, Anastasia Dimou, Ruben Verborgh, and Erik Mannens. "Using Rule-Based Reasoning for RDF Validation". In: *Rules and Reasoning: International Joint Conference, RuleML+RR 2017, London, UK, July 12–15, 2017*. Ed. by Stefania Constantini, Enrico Franconi, William Van Woensel, Roman Kontchakov, Fariba Sadri, and Dumitru Roman. Vol. 10364. Lecture Notes in Computer Science. Cham: Springer, July 2017, pp. 22–36. DOI: 10.1007/978-3-319-61252-2_3.

[51] Elena Simperl. "Reusing ontologies on the Semantic Web: A feasibility study". In: *Data and Knowledge Engineering* 68.10 (2009), pp. 905–925. ISSN: 0169-023X. DOI: 10.1016/j.datak.2009.02.002.

[52] Marcos Martínez-Romero, Clement Jonquet, Martin J. O'Connor, John Graybeal, Alejandro Pazos, and Mark A. Musen. "NCBO Ontology Recommender 2.0: an enhanced approach for biomedical ontology recommendation". In: *Journal of Biomedical Semantics* 8.1 (June 2017). DOI: 10.1186/s13326-017-0128-y.

[53] María Poveda-Villalón, Asunción Gómez-Pérez, and Mari Carmen Suárez-Figueroa. "OOPS! (OntOlogy Pitfall Scanner!): An on-line tool for ontology evaluation". In: *International Journal on Semantic Web and Information Systems (IJSWIS)* 10.2 (2014), pp. 7–34. ISSN: 1552-6283. DOI: 10.4018/ijswis.2014040102.

[54] Li Huang, Zhenzhen Liu, Fangfang Xu, and Jinguang Gu. "An RDF Data Set Quality Assessment Mechanism for Decentralized Systems". In: *Data Intelligence* 2.4 (Oct. 2020), pp. 529–553. DOI: 10.1162/dint_a_00059.

[55] Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Sören Auer. "Quality assessment for linked data: A survey". In: *Semantic Web Journal* 7.1 (Mar. 2015), pp. 63–93. DOI: 10.3233/SW-150175. URL: http://www.semantic-web-journal.net/system/files/swj556.pdf.

[56] Jeremy Debattista, Sören Auer, and Christoph Lange. "Luzzu – A Methodology and Framework for Linked Data Quality Assessment". In: *J. Data and Information Quality* 8.1 (Oct. 2016), 4:1–4:32. ISSN: 1936-1955. DOI: 10.1145/2992786. URL: http://doi.acm.org/10.1145/2992786.

[57] Nandana Mihindukulasooriya, María Poveda-Villalón, Raúl García-Castro, and Asunción Gómez-Pérez. "Loupe-An Online Tool for Inspecting Datasets in the Linked Data Cloud." In: *International Semantic Web Conference (Posters & Demos)*. Vol. 1. 1. 2015, p. 2.

[58] Andreas Langegger and Wolfram Woss. "RDFStats - An Extensible RDF Statistics Generator and Library". In: *Proceedings of the 20th International Workshop on Database and Expert Systems Applications*. Los Alamitos, Calif.: IEEE Computer Society, 2009, pp. 79–83. ISBN: 978-0-7695-3763-4. DOI: 10.1109/DEXA.2009.25.

[59] Gezim Sejdiu, Ivan Ermilov, Jens Lehmann, and Mohamed Nadjib Mami. "DistLODStats: Distributed Computation of RDF Dataset Statistics". In: *Lecture Notes in Computer Science*. Springer International Publishing, 2018, pp. 206–222. DOI: 10.1007/978-3-030-00668-6_13.

[60] Jens Lehmann et al. "Distributed Semantic Analytics Using the SANSA Stack". In: *Lecture Notes in Computer Science*. Springer International Publishing, 2017, pp. 147–155. DOI: 10.1007/978-3-319-68204-4_15.

[61] Gezim Sejdiu, Anisa Rula, Jens Lehmann, and Hajira Jabeen. "A Scalable Framework for Quality Assessment of RDF Datasets". In: *Lecture Notes in Computer Science*. Springer International Publishing, 2019, pp. 261–276. DOI: 10.1007/978-3-030-30796-7_17.

[62] Heba Mohamed, Said Fathalla, Jens Lehmann, and Hajira Jabeen. "OWLStats: Distributed Computation of OWL Dataset Statistics". In: *2020 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*. IEEE, Dec. 2020. DOI: 10.1109/wiiat50758.2020.00055.

[63] Daniel Garijo. "WIDOCO: a wizard for documenting ontologies". In: *The Semantic Web - International Semantic Web Conference (ISWC 2017)*. Ed. by Claudia d'Amato, Miriam Fernandez, Valentina Tamma, Freddy Lecue, Philippe Cudré-Mauroux, Juan Sequeda, Christoph Lange, and Jeff Heflin. Springer. Springer International Publishing, 2017, pp. 94–102. DOI: 10.1007/978-3-319-68204-4_9.

[64] Silvio Peroni, David Shotton, and Fabio Vitali. "Latest Developments to LODE". In: *EKAW 2012 : The 18th International Conference on Knowledge Engineering and Knowledge Management*. Springer Berlin Heidelberg, Oct. 2012, pp. 417–420. DOI: 10.1007/978-3-642-33876-2_37.

[65] Steffen Lohmann, Vincent Link, Eduard Marbach, and Stefan Negru. "WebVOWL: Web-based visualization of ontologies". In: *International Conference on Knowledge Engineering and Knowledge Management*. Springer. 2014, pp. 154–158. DOI: 10.1007/978-3-319-17966-7_21.

[66] Mark A. Musen. "The Protégé Project: A Look Back and a Look Forward". In: *AI Matters* 1.4 (June 2015), pp. 4–12. DOI: 10.1145/2757001.2757003.

[67] Rafael S. Gonçalves, Matthew Horridge, Rui Li, Yu Liu, Mark A. Musen, Csongor I. Nyulas, Evelyn Obamos, Dhananjay Shrouty, and David Temple. "Use of OWL and Semantic Web Technologies at Pinterest". In: *Lecture Notes in Computer Science*. Springer International Publishing, 2019, pp. 418–435. DOI: 10.1007/978-3-030-30796-7_26.

[68] Matthew Horridge, Tania Tudorache, Csongor Nuylas, Jennifer Vendetti, Natalya F. Noy, and Mark A. Musen. "WebProtégé: a collaborative Web-based platform for editing biomedical ontologies". In: *Bioinformatics* 30.16 (Apr. 2014), pp. 2384–2385. DOI: 10.1093/bioinformatics/btu256.

[69] A. Katifori, C. Halatsis, G. Lepouras, C. Vassilakis, and E. Giannopoulou. "Ontology visualization methods – a survey". In: *ACM Comput. Surv.* 39 (2007), p. 10. DOI: 10.1145/1287620.1287621.

[70] S. Mikhailov, Mikhail Petrov, and Birger Lantow. "Ontology Visualization: A Systematic Literature Analysis". In: *Joint Proceedings of the BIR 2016 Workshops and Doctoral Consortium co-located with 15th International Conference on Perspectives in Business Informatics Research (BIR 2016)*. Vol. 1684. CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[71] Nassira Achich, B. Bouaziz, Alsayed Algergawy, and F. Gargouri. "Ontology Visualization: An Overview". In: *17th International Conference on Intelligent Systems Design and Applications (ISDA)*. 2017. DOI: 10.1007/978-3-319-76348-4_84.

[72] Anton Anikin, Dmitry Litovkin, Marina Kultsova, Elena Sarkisova, and Tatyana Petrova. "Ontology Visualization: Approaches and Software Tools for Visual Representation of Large Ontologies in Learning". In: *Creativity in Intelligent Technologies and Data Science*. Springer International Publishing, 2017, pp. 133–149. ISBN: 978-3-319-65551-2. DOI: 10.1007/978-3-319-65551-2_10.

[73] Marek Dudáš, Steffen Lohmann, Vojtěch Svátek, and Dmitry Pavlov. "Ontology visualization methods and tools: a survey of the state of the art". In: *The Knowledge Engineering Review* 33 (2018). DOI: 10.1017/s0269888918000073.

[74] Merlin Florrence Joseph and Ravi Lourdusamy. "Feature analysis of ontology visualization methods and tools". In: *Computer Science and Information Technologies* 1.2 (2020), pp. 61–77. DOI: 10.11591/csit.v1i2.p61-77.

[75] Fatma Ghorbel, Nebrasse Ellouze, Fay Métais, Faiez Gargouri, Noura Herradi, et al. "MEMO GRAPH: an ontology visualization tool for everyone". In: *Procedia Computer Science* 96 (2016), pp. 265–274. DOI: 10.1016/j.procs.2016.08.139.

[76] G. Braun, C. Gimenez, L. Cecchi, and P. Fillottrani. "crowd: A Visual Tool for Involving Stakeholders into Ontology Engineering Tasks". In: *KI - Künstliche Intelligenz* 34.3 (2020), pp. 365–371. DOI: 10.1007/s13218-020-00657-8.

[77] Dieter De Paepe, Geert Thijs, Raf Buyle, Ruben Verborgh, and Erik Mannens. "Automated UML-Based Ontology Generation in OSLO²". In: *The Semantic Web: ESWC 2017 Satellite Events – ESWC 2017*. Ed. by Eva Blomqvist, Katja Hose, Heiko Paulheim, Agnieszka Ławrynowicz, Fabio Ciravegna, and Olaf Hartig. Vol. 10577. Lecture Notes in Computer Science. Springer, Cham, 2017, pp. 93–97. DOI: 10.1007/978-3-319-70407-4_18.

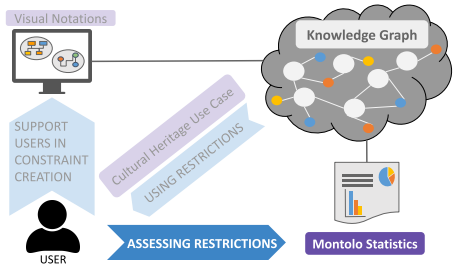
[78] Andrea Cimmino, Alba Fernández-Izquierdo, and Raúl García-Castro. "Astrea: Automatic Generation of SHACL Shapes from Ontologies". In: *European Semantic Web Conference (ESWC)*. Ed. by Andreas Harth, Sabrina Kirrane, Axel-Cyrille Ngonga Ngomo, Heiko Paulheim, Anisa Rula, Peter Haase, and Michael Cochez. Springer. Springer International Publishing, 2020, pp. 497–513. DOI: 10.1007/978-3-030-49461-2_29.

[79] Ben De Meester, Pieter Heyvaert, Anastasia Dimou, and Ruben Verborgh. "Towards a Uniform User Interface for Editing Data Shapes". In: *Proceedings of the 4th International Workshop on Visualization and Interaction for Ontologies and Linked Data*. Ed. by Valentina Ivanova, Patrick Lambrix, Steffen Lohmann, and Catia Pesquita. Vol. 2187. CEUR Workshop Proceedings. CEUR-WS.org, Oct. 2018, pp. 13–24. URL: http://ceur-ws.org/Vol-2187/paper2.pdf.

[80] Jose Emilio Labra Gayo, Daniel Fernández-Álvarez, and Herminio García-González. "RDFShape: An RDF Playground Based on Shapes". In: *Proceedings of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks co-located with 17th International Semantic Web Conference (ISWC 2018)*. Ed. by Marieke Van Erp, Medha Atre, Vanessa Lopez, Kavitha Srinivas, and Carolina Fortuna. Vol. 2180. CEUR Workshop Proceedings. CEUR-WS.org, Oct. 2018.

[81] Iovka Boneva, Jérémie Dusart, Daniel Fernández Alvarez, and Jose Emilio Labra Gayo. "Shape Designer for ShEx and SHACL Constraints". In: *Proceedings of the ISWC 2019 Satellite Tracks (Poster & Demonstrations, Industry, and Outrageous Ideas)*. Vol. 2456. CEUR-WS.org, Oct. 2019, pp. 269–272. URL: https://hal.archives-ouvertes.fr/hal-02268667.

[94] Manu Sporny, Gregg Kellogg, and Markus Lanthaler. *JSON-LD 1.0 – A JSON-based Serialization for Linked Data*. Recommendation. World Wide Web Consortium (W3C), Jan. 2014. URL: http://www.w3.org/TR/json-ld/.



Chapter 2

Assessment of Knowledge Graph Restrictions

Solving data integration problems with RDF relies on reusing existing Knowledge Graphs. Knowledge Graphs describe real world entities and their interrelations, as well as define possible classes and relations of entities in a schema [1]. A use case may demand to express structural constraints on real world entities or to already restrict classes and relations using axioms. Any such restriction may influence the reuse, and, thus, motivates an assessment of Knowledge Graphs with respect to used restrictions. This chapter presents the following contributions to the assessment of restrictions in Knowledge Graphs:

- The Montolo approach to define and assess different restriction types encoded using RDF terms
- An implementation of the approach to automatically compute W3C data cube compliant restriction use statistics for RDF Knowledge Graphs
- The MontoloStats dataset which contains statistics of used RDFS/OWL axioms in 660 ontologies obtained from LOV and 656 ontologies obtained from BioPortal
- The MontoloSHACLStats dataset which contains statistics of used W3C SHACL constraints in data shapes identified on GitHub

We address Research Question 1 "How can we support the assessment of restrictions in existing Knowledge Graphs?" and validate Hypothesis 1 "FAIR statistics of RDF encoded axioms and constraints enable restriction use assessments of several existing Knowledge Graphs not possible with state of the art tools.".

As this is a cumulative dissertation, both Section 2.1 and Section 2.2 correspond with a publication. Section 2.1 corresponds with "MontoloStats - Ontology Modeling Statistics" which introduces Montolo and presents an analysis of axiom use. Section 2.2 corresponds

with “Statistics about Data Shape Use in RDF Data” which applies Montolo on data shapes and presents an analysis of constraint use. The statistics and figures from the second publication are updated in this dissertation. Thus, they correspond to an update of the dataset created for the presentation at the ISWC conference[1].

[1] https://zenodo.org/record/4154456

2.1 Assessing and Analyzing the Use of Axioms for Ontologies using Montolo

Sven Lieber, Ben De Meester, Anastasia Dimou, and Ruben Verborgh

Published as "MontoloStats – Ontology Modeling Statistics", in *K-CAP '19: Proceedings of the 10th International Conference on Knowledge Capture*, Marina Del Rey CA, USA, November 19-21, 2019, Pages 69-76.

Abstract

Within ontology engineering concepts are modeled as classes and relationships, and restrictions as axioms. Reusing ontologies requires assessing if existing ontologies are suited for an application scenario. Different scenarios not only influence concept modeling, but also the use of different restriction types, such as subclass relationships or disjointness between concepts. However, metadata about the use of such restriction types is currently unavailable, preventing accurate assessments for reuse. We created the RDF Data Cube-based dataset *MontoloStats*, which contains restriction use statistics for 660 LOV and 565 BioPortal ontologies. We analyze the dataset and discuss the findings and their implications for ontology reuse. The *MontoloStats* dataset reveals that 94% of *LOV* and 95% of *BioPortal* ontologies use RDFS-based restriction types, 49% of *LOV* and 52% of *BioPortal* ontologies use at least one OWL-based restriction type, and different literal value-related restriction types are not or barely used. Our dataset provides modeling insights, beneficial for ontology reuse to discover and compare reuse candidates, but can also be the basis of new research that investigates novel ontology engineering methodologies with respect to restrictions definition.

2.1.1 Introduction

The Semantic Web uses *ontologies* to formally represent real-world domains and concepts [2]. An ontology is a conceptualization, an intensional semantic structure which encodes the implicit rules restricting the structure of a piece of reality [3]. In addition to containing *concepts* and *relationships*, an ontology is characterized by a set of *axioms* [4]. According to the terminology used in this dissertation, axioms are considered to be restrictions as they restrict meaning by stating what is true. Different types of restrictions exist, such as subclass relationships or disjointness between concepts. Each restriction type serves different purposes: subclass relationships can for instance describe taxonomic structures, and disjoint classes express mutual exclusiveness in a machine-understandable way.

Ontologies play different roles in different application scenarios [5], influencing how restrictions are used. In a semantic search scenario, ontologies are built to be used by machines, which demands *machine-understandable semantics* that are explicitly stated restrictions, such as cardinalities or disjoint properties. In more human-targeted scenarios, such heavily axiomatized ontologies would pose challenges regarding comprehensibility. For instance, a taxonomic structure defined with restriction type *subsumption*, when encoded

using a `rdfs:subClassOf` expression, imposes lower ontology reusability costs than other restriction types [6].

Whereas several insights on class and relationship usage exist, restriction types so far have remained insufficiently documented, making it difficult to inform ontology reuse. From a process point-of-view ontology reuse consists of multiple activities, such as *discovery* and *assessment* of reuse candidates [5]. Metadata about prevalent restriction types would support the selection of reuse candidates that are appropriate for a given application scenario. Restriction types can be expressed with different vocabularies and terms, and, thus, multiple expressions need to be considered to obtain comprehensive metadata. Consider for instance disjoint class restrictions which can be expressed either using the property `owl:disjointWith`, or the class `owl:AllDisjointClasses`.

To the best of our knowledge, currently no available dataset exists which provides statistics about restriction type use independent from their expressions.

We introduce the *MontoloStats* dataset describing the use of different restriction type expressions in *LOV* and *BioPortal* ontologies. We analyze the dataset and discuss the results with respect to ontology reuse. Our contributions are:

1. an approach to model restriction types' expressions and statistical measures using the W3C-recommended RDF Data Cube and PROV vocabularies;

2. an implementation of the approach as extension of *LODStats* [7] to automatically generate statistical measures;

3. the *MontoloStats* statistical dataset to describe restrictions use in *LOV* and *BioPortal* ontologies;

4. analysis and discussion of the *MontoloStats* dataset and its implications for ontology reuse and further research.

MontoloStats can foster further research in a plethora of research challenges related to, for instance, ontology reuse to assess different aspects of an ontology, or knowledge modeling. Statistics regarding the restriction type use and distribution may be used for further in-depth analysis of how restriction types were modeled as axioms and what impact this has on their further use. The resources accompanying this paper are published at `https://w3id.org/montolo`, specifically:

- The **MontoloStats statistical dataset** is published at `https://w3id.org/montolo/data/montolo-stats/`[2] under CCo license[3], with accompanying public SPARQL endpoint (DOI: 10.5281/zenodo.3407139);

[2] Sven Lieber, "MontoloStats", `http://web.archive.org/web/20220212123518/https://lov.ilabt.imec.be/montolo/data/montolo-stats/20190911/` (archived website accessed February 12, 2022)

[3] Creative Commons, "CC 1.0", `http://web.archive.org/web/20220212123024/https://creativecommons.org/publicdomain/zero/1.0/` (archived website accessed February 12, 2022)

- Definitions of identified restriction types, expressions and measures are published as **Montolo dataset** at `https://w3id.org/montolo/ns/montolo` under CCo license (DOI: 10.5281/zenodo.3343313);

- The **MontoloVoc vocabulary**, created to describe concepts of *Montolo* and *MontoloStats* is published at `https://w3id.org/montolo/ns/montolo-voc` and made available at `https://github.com/IDLabResearch/montolo-voc` under CCo license (DOI: 10.5281/zenodo.3343335);

- The **LODStats extension** used to create *MontoloStats* is available at `https://github.com/IDLabResearch/lovstats` under MIT license[4] (DOI: 10.5281/zenodo.2165747).

The remainder of the paper is organized as follows: Section 2.1.2 summarizes the related work, in Section 2.1.3 we present our proposed approach, that generates *MontoloStats* (Section 2.1.4). Last, we analyze *MontoloStats* in Section 2.1.5 and summarize our conclusions and future work in Section 2.1.6.

2.1.2 Related Work

Our work concerns statistics regarding the use of restrictions to support ontology reuse. Therefore, we investigate existing work regarding (i) restrictions in ontologies, (ii) ontology reuse, and (iii) statistics in the Semantic Web.

Restrictions More complex and possibly formal vocabularies containing restrictions, are usually referred to as ontologies[5]. represent knowledge machine-understandably. OWL2 is a knowledge representation language which uses different restriction types in the form of axioms, e.g. *disjoint classes* or *reflexive properties*. Whereas restrictions in the form of axioms are used to represent knowledge, restrictions in the form of constraints are used to e.g., validate data which should adhere to such a knowledge representation [8].

The latter was investigated mostly in the context of data quality. RDFUnit [9] is a test-driven evaluation framework for Linked Data, which uses a set of SPARQL templates, expressing data quality issues. Several Data Quality Test Patterns cover aspects such as cardinality, disjointness or literal value restrictions.

Arndt et al. [10] provided an alignment between RDFUnit's Data Quality Test Patterns and corresponding restriction types identified by Hartmann [11], to cover restriction types which minimally cover common validation requirements. An investigation in the use of such restriction types in ontologies could reveal beneficial information for ontology engineering.

[4] Open Source Initiative, "MIT License", `http://web.archive.org/web/20220212095116/https://opensource.org/licenses/MIT` (archived website accessed February 12, 2022)

[5] W3C, "Vocabularies", `https://web.archive.org/web/20211223205759/https://www.w3.org/standards/semanticweb/ontology` (archived website accessed February 12, 2022)

Ontology Reuse Ontology reuse implicitly follows a four step workflow involving the discovery, selection, customization and integration of potential reuse candidates [12]. Different methods exist to support each step's tasks, and especially ontology metadata can be of use for the first two steps.

The first step, discovery of existing ontologies and their concepts, is facilitated by vocabulary catalogs such as LOV [13] or Bioportal [14]. These catalogues provide search capabilities already considering a limited amount of metadata.

However, given an application scenario in which more or less axiomatized ontologies are required, the current search capabilities are insufficient, i.e. no filter on ontologies using specific restriction types or restriction type expressions. These search capabilities, and hence the ontology discovery step, would benefit from restriction use statistics.

The second step, selection of appropriate reuse candidates, entails the evaluation of the different reuse candidates with respect to the given application scenario.

OOPS! [15] validates ontologies by detecting anomalies and bad practices leading to modeling errors, thus, it supports users to qualitatively evaluate and compare reuse candidates.

Our statistics provide quantitative measurements of restriction type use which can complement a qualitative assessment and support users in selecting ontologies appropriate for given application scenarios with respect to modeled restrictions.

From an economical point-of-view the activities performed in a reuse process adhere to different costs. The ONTOCOM [6] cost estimation model, created based on expert interviews [2], tries to quantify these costs by calculating necessary person-months effort.

Several identified cost drivers could benefit from restriction use statistics, as users' effort may be reduced due to available restriction use statistics for ontology reuse related tasks.

Statistics in the Semantic Web Two main approaches to compute statistics were suggested: from a dataset and from an ontology point-of-view. Datasets are statistically analyzed in RDFStats [16], LODStats [7] and Loupe [17]. RDFStats [16] supports users to browse RDF graphs and applications dealing with large, possibly distributed RDF graphs. Statistical metrics of RDFStats were reused in LODStats [7], a statement-stream-based approach to analyze RDF data. LODStats, due to its streaming approach, is suitable for large datasets. It comes with a set of 32 statistical measures, which can be extended. Loupe [17], among others, analyzes implicit data patterns, regarding vocabulary use, and explicit vocabulary definitions regarding ontological axioms used in data. Focused on dataset structure, Loupe does not cover restriction-related information.

Dataset-related approaches focus on dataset structure, schema-level statistics are only considered to a small extent. Additionally, restrictions are covered from a dataset point-of-view, creating mixed statistics of all ontologies used in a dataset. Ontology reuse concerns the discovery and selection of possible reuse candidates, and if compared based on statistical metadata, restriction use statistics from an ontology point-of-view are needed.

From an ontology point-of-view, tools like Protégé [18] provide summaries about used axioms in an ontology, but these summaries only cover a fixed set of axioms, and are only shown for the currently loaded ontologies. In contrast, our approach describes generic

restriction types and concrete expressions which are extendible and provides a statistical dataset covering multiple ontologies.

ComplexOnto [19] is a score, expressing the complexity of ontologies, to better understand, maintain, reuse and integrate ontologies. The score consists of four metrics describing different interlinking characteristics, based on properties and subclass axioms. However, the score, as aggregated value, does not provide detailed information, and its constituents only focus on how connected used concepts are, leaving out information regarding used axioms.

The discovery and selection of ontologies for reuse based on statistical metadata regarding restriction use demands available restriction use statistics per ontology. Additionally, vocabularies such as RDF and OWL contain different expressions for identified restriction types which need to be considered to provide comprehensive statistics. To the best of our knowledge, existing approaches do not provide statistics on restriction use per ontology on the level of restriction types taking different expressions into account. Existing approaches do, however, provide a framework to create statistics which we extend for restriction use in ontologies.

2.1.3 Approach

We propose an approach to compute statistics of restriction type use in ontologies to support ontology engineering activities. We differentiate between (abstract) *restriction types*, e.g. disjointness, and (concrete) *restriction type expressions* per restriction type, e.g. disjoint classes expressed via the property `owl:disjointWith` or the class `owl:AllDisjointClasses`, to comprehensively describe restriction use information. More, we define measures to calculate statistics of restriction types and their expressions, e.g. number of occurrences of classes annotated with `owl:disjointWith`. Our approach consists of three steps: (i) the unambiguously description of restrictions, (ii) extraction of restriction type expressions from ontologies, and (iii) computation of statistics, described with our RDF DataCube-based *MontoloVoc* vocabulary.

1. describe restriction types, expressions and measures We followed the UPON-light methodology [20] to create our *MontoloVoc* vocabulary describing restriction types, their expressions and measures in a machine-understandable way. Restriction types and expressions can be defined and linked using the associated *MontoloVoc* classes[6], thus measured values can be linked to a single definition. An instance of the class `mov:RestrictionType` is created for each restriction type, as shown in listing 2.2, line 1-4 for the restriction type *disjoint classes*. Different expressions of this restriction type, such as `owl:disjointWith` (6-10) or `owl:AllDisjointClasses` (line 14-16) can be created using the introduced *MontoloVoc* class `mov:RestrictionTypeExpression`, which is linked via the property `frbr:realizationOf`[7] to their respective `mov:RestrictionType`, to make their relationship explicit. Different measures can be defined to analyze restriction use in ontologies. A measure, e.g. number of

[6] Abbreviated in this paper using the `mov` prefix.

[7] Ian Davis and Richard Newman, "Expression of Core FRBR Concepts in RDF", `https://web.archive.org/web/20201106232643/http://vocab.org/frbr/core` (archived website accessed February 12, 2022)

occurrences, can be described with the *MontoloVoc* class `mov:RestrictionTypeMeasure` (line 20-21).

2. extract restriction type expressions from ontologies This step concerns the extraction of identified restriction type expressions from ontologies. Different extraction mechanisms can be used for this step, e.g. queries on ontologies or stream-based solutions reading RDF.

```
1  # instances of collection cannot be instances of
2  # concepts or concept schemes and vice versa
3  skos:Collection
4    owl:disjointWith skos:Concept ;
5    owl:disjointWith skos:ConceptScheme .
6
7  skos:ConceptScheme owl:disjointWith skos:Concept .
8
9  # same restriction expressed using a class (pairwise exclusive)
10 [] a owl:AllDisjointClasses ;
11 owl:members ( skos:Collection skos:ConceptScheme skos:Concept).
```

Listing 2.1: Disjoint classes restriction, expressed with OWL in 2 different semantically equivalent ways.

3. compute restriction type measures Different measures can be defined to analyze restriction type use in ontologies, but need to be computed differently for each restriction type expression. Measures relate to restriction types, but to achieve a fair comparison between different restriction type expressions, the measure needs to be computed differently. Consider again the restriction type *disjoint classes*. The three RDF statements in listing 2.1 line 3-7 express the disjointness between `skos:Collection`, `skos:Concept` and `skos:ConceptScheme`, and therefore correspond to three restriction statements. Yet the two RDF statements in listing 2.1 line 10-11 also define three restrictions. An OWL restriction class instance with a list of pairwise disjoint classes is used, which corresponds to $\frac{n^2-n}{2}$ disjoint class statements. Both expressions lead to three disjoint classes, although the number of RDF statements differs. Hence the number of *disjoint classes* restrictions need to be computed differently for each expression, to achieve comparable restriction type measures between restriction type expressions. The computed measures can then be described with the class `lst:RestrictionTypeStatistic` (listing 2.2 line 23-33), subclass of an RDF data cube observation.

2.1.4 Montolo

We applied the approach to create both *Montolo*, descriptions of restriction types, and *MontoloStats*, a dataset describing restriction type use of LOV and Bioportal ontologies. In the following we describe (i) restriction types we cover in *Montolo*, (ii) the implementation of our approach as LODStats extension, and (iii) the *MontoloStats* dataset.

```
# Restriction Type
mon:disjointClasses
  a mov:RestrictionType ;
  rdfs:label "Disjoint classes restriction type"@en .

# Restriction Type Expression 1
mon:disjointClassesOwlDisjointWith
  a mov:RestrictionTypeExpression ;
  frbr:realizationOf mon:disjointClasses ;
  rdfs:label "owl:disjointWith restriction"@en .

# Restriction Type Expression 2
mon:disjointClassesOwlAllDisjointClasses
  a mov:RestrictioinTypeExpression ;
  frbr:realizationOf mon:disjointClasses ;
    rdfs:label "owl:AllDisjointClasses restriction"@en .

# Restriction Type Measure
mon:restrictionTypeOccurrence
  a mov:RestrictionTypeMeasure ;
  rdfs:label "Restriction type occurrence"@en .

# Restriction Type Statistic (example of a generated result)
[] a mov:RestrictionTypeStatistic ;
  mon:executionTimeDimension
    "2019-04-06T08:30:54.280117"^^xsd:dateTime ;
  mon:detectorVersionDimension
    mon:disjointClassesLODStatsDetectorOwlDisjointWith-v1 ;
  mon:ontologyRepository mon:lov ;
  mon:ontologyVersionDimension
    <http://www.w3.org/2004/02/skos/core#> ;
  mon:restrictionTypeDimension mon:disjointClasses ;
  mon:restrictionTypeOccurrence 3 .
```

Listing 2.2: Restriction type *disjoint classes* and its expressions in Montolo namespace (prefix `mon`), represented with *MontoloVoc* vocabulary (prefix `mov`).

2.1.4.1 Covered restriction types and measures

We described 18 restriction types based on related work [11, 10], using the proposed *MontoloVoc* vocabulary. We also define the *occurrence* measure expressing the number of axiom statements, following step 1 of our approach[8]. Table 2.1 lists the restriction types and restriction type expressions used to detect them. We consider restriction types expressed using RDFS and OWL vocabularies, because dataset-related statistics indicate that RDF(S) and OWL are the most prevalent vocabularies to define ontologies using RDF [21, 7].

From RDFS, we cover the three restriction types *subsumption*, *domain* and *range* to identify taxonomic structures. For the expression `rdfs:subClassOf` we also use the `isIRI` filter provided by LODStats to count actual taxonomic relationships between concepts and avoid counting common patterns in which e.g. `rdfs:subClassOf` is used to express that a concept is a subclass of a specific `owl:Restriction`. Furthermore we consider all six *cardinality-related restriction types* that OWL describes. For the restriction type *exact*

[8] Sven Lieber, "Montolo", `https://w3id.org/montolo/ns/montolo` (accessed February 12, 2022)

unqualified cardinality, we cover two expressions: the property `owl:cardinality` and a combination of `owl:minCardinality` and `owl:maxCardinality` with the same value. Also two expressions are defined for each of the two restriction types *disjoint classes* and *disjoint properties*, as machine-understandable disjointness is an important information for the Semantic Web. We also consider different *property* and *literal value-related* restriction types. This list is not exhaustive, it reflects on restriction types identified in related work and provides a few common expressions thereof, i.e. syntactical patterns used to encode the restriction type. We are particularly interested in measuring these different syntactical patterns which we also define using our Montolo vocabulary for provenance purposes.

2.1.4.2 LODStats extension

We build upon and contribute to existing work to provide statistics about restriction types. We take advantage of *LODStats*' extensibility to define statistical modules for restriction types. For each restriction type, we create one statistical module. Restriction types can be expressed in different ways, yet restriction type measures should be comparable between restriction type expressions. Thus, we introduce one *detector class* per restriction type expression which shares the same interface among its corresponding restriction type and provides same measures. Other restriction types can be added as statistical modules and other restriction type expressions can be added using a new detector. Thus our implementation adheres to the extendibility of our approach.

2.1.4.3 Dataset

We applied the approach on two repositories: (i) *LOV*, a general-purpose ontology repository, and (ii) *BioPortal*, a domain-specific ontology repository. The *MontoloStats* dataset consists of 395,675 triples and 31,850 RDF data cube observations. The *MontoloStats* dataset is small in size (22 MB) and interoperable as it adheres to the W3C recommendations RDF DataCube and PROV. We published *MontoloStats* on Zenodo under CC0 license to ensure its availability. All Montolo-related artifacts, such as the *MontoloVoc* vocabulary and LODStats extension, are publicly hosted on GitHub, to enable the community's engagement.

We provide badges for each ontology indicating the number of prevalent restriction types. Such badges allows for easy visual inspection and comparison of vocabularies, and eases integration in existing platforms and systems. Badges are available for every ontology in the *MontoloStats* dataset[9], redirecting to the detailed *MontoloStats* page per ontology[10].

LOV We analyzed ontologies listed in *LOV*, which contained by the time of writing 672 ontologies. We downloaded the latest version of each ontology from *LOV* in N-triples and stored them. Due to some errors during parsing, we could compute our statistics for 660 ontologies and, thus, the statistics cover 98% of *LOV*.

[9] URI template for a badge: `https://w3id.org/montolo/data/montolo-stats/latest/voc/[prefix]?type=svg`.
[10] URI template: `https://w3id.org/montolo/data/montolo-stats/latest/voc/[prefix]`.

BioPortal We analyzed OWL and OBO ontologies, which are OWL-compatible, listed in *BioPortal*. According to a JSON file obtained via BioPortal[11], 716 OWL and 123 OBO ontologies are listed. However, while downloading the ontologies we encountered several *Access Denied* responses due to a missing ontology file or license-restrictions. We used the *robot* tool [22] to convert the downloaded OWL/XML and OBO ontologies to RDF/XML, as it adheres to the W3C recommended OWL-TO-RDF mapping[12] and supports the OBO format. The conversion failed for 87 ontologies due to different parsing errors. Finally, the successful converted ontologies were transformed to N-triples and, thus, we could compute our statistics for 565 ontologies of BioPortal. Besides the conversion from OWL/XML and OBO to RDF/XML (performed by the same tool), we did not perform semantic normalizations because we are particularly interested in measuring different syntactical patterns.

2.1.5 Analysis

We analyze the restriction type distribution to provide an overview of their use in *LOV* and *BioPortal* and multiple expressions for restriction types to reveal modeling practices.

2.1.5.1 Restriction Type Distribution

We analyze *MontoloStats* with respect to (i) the distribution of restriction types across *LOV* and *BioPortal*, (ii) vocabularies used to encode restriction type expressions, (iii) cardinality-related and (iv) property-related restriction types, and (v) ontologies using no restriction types.

Restriction Types In total, 17 out of 18 restriction types occur in both LOV and BioPortal ontologies, from which 15 barely appear and 3 clearly dominate in LOV (Figure 2.1), and only 1 in BioPortal. 3 restriction types, namely *subsumption*, *domain*, and *range* in its RDFS-based expressions `rdfs:subClassOf`, `rdfs:domain` and `rdfs:range` stand out in LOV, as each of them occurs more than 27,000 times in total and in more than 94% of LOV ontologies. This indicates a taxonomic structure of the ontological concepts for the majority of LOV ontologies. Similarly, *subsumption* is also the most used restriction type in BioPortal, occurring more than 3 million times in total and in more than 93% of BioPortal ontologies. The restriction types *domain* and *range* are not as common in BioPortal as they are in LOV, both total numbers and amount of ontologies using it is considerably lower. But therefore *disjoint classes* restrictions are the second most used restrictions in BioPortal, used more than 760,000 times and in around 38% of the analyzed BioPortal ontologies. By total number, *subsumption* is the most used restriction type in both LOV and BioPortal ontologies. The restriction type *range* is the most used in 88% of LOV ontologies, and *subsumption* restrictions are the most used restrictions in BioPortal ontologies with 93%.

[11] API to download an ontology list: `http://data.bioontology.org/ontologies_full`

[12] Peter F. Patel-Schneider et al., "OWL2 Web Ontology Language Mapping to RDF Graphs (Second Edition)", `https://web.archive.org/web/20220120180334/https://www.w3.org/TR/owl2-mapping-to-rdf/` (archived website accessed February 12, 2022

Table 2.1: Restriction types and some corresponding expressions to detect them, the list is not exhaustive and new expressions can be identified and added. Restriction type expressions are listed as triple patterns and additional filter functions. For each found triple pattern we increase the corresponding counter by 1, except for the 2nd expression of *disjoint classes* and *properties*, where we compute $\frac{n^2-n}{2}$ (n is the ?*list*'s length).

Restriction Type	Restriction Type Expression
Subsumption	`{?s rdfs:subClassOf ?o .}` && *isIRI(?s)* && *isIRI(?o)*
Domain	`{?s rdfs:domain ?o .}`
Range	`{?s rdfs:range ?o .}`
Literal pattern matching	`{?s owl:withRestrictions ?list .` `?s2 xsd:pattern ?o2 . }` *isListMember(?list, ?s2)*
Literal ranges	`{?s owl:withRestrictions ?list .` `?s2 xsd:minInclusive|xsd:minExclusive` `|xsd:maxInclusive|xsd:maxExclusive ?o2 .}` `&& isListMember(?list, ?s2)`
Min unqualified cardinality	`{?s owl:minCardinality ?o .}`
Min qualified cardinality	`{?s owl:minQualifiedCardinality ?o .}`
Max unqualified cardinality	`{?s owl:maxCardinality ?o .}`
Max qualified cardinality	`{?s owl:maxUnqualifiedCardinality ?o .}`
Exact qualified cardinality	`{?s owl:qualifiedCardinality ?o .}`
Exact unqualified cardinality	`{?s owl:cardinality ?o .}` `{?s1 owl:minCardinality ?o1 .` `?s2 owl:maxCardinality ?o2 .}` `&& isEqual(?o1, ?o2)`
Functional properties	`{?s rdf:type owl:FunctionalProperty .}`
Inverse functional properties	`{?s rdf:type owl:InverseFunctionalProperty.}`
Universal quantification	`{?s owl:allValuesFrom ?o .}`
Asymmetric properties	`{?s rdf:type owl:AsymmetricProperty .}`
Irreflexive properties	`{?s rdf:type owl:IrreflexiveProperty .}`
Disjoint properties	`{?s owl:propertyDisjointWith ?o .}` `{?s rdf:type owl:AllDisjointProperties .` `?s owl:members ?list .} && isEqual(?o1, ?o2)`
Disjoint classes	`{?s owl:disjointWith ?o .}` `{?s rdf:type owl:AllDisjointClasses .` `?s owl:members ?list .} && isEqual(?o1, ?o2)`

On the other end of the spectrum, the restriction type *literal ranges* occurs only 64 times in 4 LOV ontologies, and 421 times in 13 BioPortal ontologies. This corresponds to less than 1% of the LOV and around 2% of BioPortal ontologies. Neither LOV nor BioPortal ontologies have the restriction type *literal pattern matching*. We assume that restrictions regarding literal values are either not popular, or the ontologies are modeled in such a way, that literal values-related restrictions are not necessary (a concept expressed as class rather than literal value). For example, one could use the OWLTime ontology [23] to semantically represent dates or a literal with a datatype, both variants have different implications in terms of reuse in a use case, e.g. partially known date values in the field of cultural heritage. Whereas the restriction type *literal ranges* is the least used in LOV ontologies, for BioPortal it is the restriction type *asymmetric properties*.

For BioPortal, trends in the total number of *subsumption* and *disjoint classes* are different compared to the number of ontologies using these restriction types. A few ontologies make heavy use of these restriction types and thus distort the result. This is different in LOV ontologies where for the 5 most-common restriction types the trends are similar between the total occurrence of a restriction type and ontologies using it, i.e. *subsumption*, *domain* and *range* dominate followed by *disjoint classes* and *universal quantification*.

Vocabularies used to express restriction types *MontoloStats* contains information about restriction types expressed with RDFS and OWL, for which LOV and BioPortal show similar use. More than 94% of both LOV and BioPortal ontologies include at least one of the RDFS-based restrictions *subsumption*, *domain* or *range*. OWL-based restrictions are used less than RDF-based restrictions, but again to a similar extent among LOV and BioPortal with 49% respectively 52% of ontologies using it. Considered individually, the OWL-based restriction types are used in less than 26% of ontologies in both LOV and BioPortal ontologies.

Cardinality-related restriction types Six restriction types regarding cardinality exist in *Montolo*: minimum and maximum qualified and unqualified cardinality, and exact qualified and unqualified cardinality. *MontoloStats* reveals a similar amount of use, but different use patterns between LOV and BioPortal ontologies.

In total, 24% of LOV and 21% of BioPortal ontologies use at least 1 of the 6 cardinality-related restriction types, demonstrating similar cardinality-related restriction type use in LOV and BioPortal ontologies. The *exact unqualified cardinality* restriction type is used 1,378 times in 110 ontologies, which corresponds to 16% of LOV ontologies, and, thus, the most used cardinality-related restriction type. In BioPortal ontologies, however, *minimum qualified cardinality* is the most used cardinality-related restriction type, used 1,166 times in 82 ontologies (14% of BioPortal ontologies). Comparing qualified and unqualified variants, *MontoloStats* reveals that unqualified variants are used more often than qualified in LOV ontologies, but qualified variants for *maximum* and *minimum* are more often used for BioPortal ontologies.

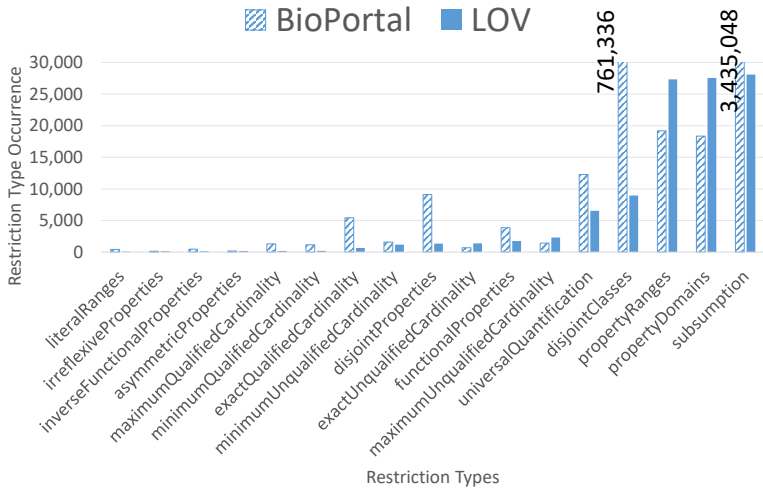


Figure 2.1: In 660 LOV ontologies, 3 restriction types were very common; the others were barely used. And across all 565 BioPortal ontologies, *subsumption* restrictions clearly dominate, followed by *disjoint classes* restrictions; their total occurrence is indicated as it is out of the chart bounds.

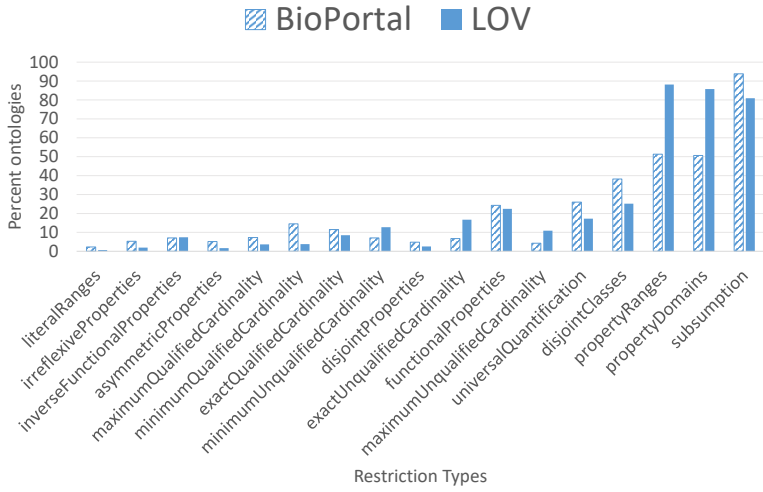


Figure 2.2: Restriction type use pattern is similar for LOV and BioPortal; less common OWL-based restrictions are used slightly more often in BioPortal.

In LOV ontologies, the unqualified variant of *maximum cardinality* restrictions is used 12 times more than the qualified, for *minimum cardinality* the unqualified variant is used 6 times more, and for *exact cardinality* the unqualified variant is used 2 times more. Besides these total numbers, in all 3 cases the unqualified variant is used between 2 (*exact cardinality*) and 3.4 (*minimum cardinality*) times more ontologies. While the number of ontologies for which unqualified variants are used more often is in the same range (2, 3 and 3.4 times respectively), we clearly see a trend in total numbers (12, 6 and 2 times more often), perhaps because *qualified cardinalities* were only introduced in OWL2[13], or because *qualified cardinalities* are more specific than unqualified cardinalities, which may explain that they are used less.

Compared to the above analysis of qualified and unqualified cardinalities for LOV ontologies, BioPortal ontologies show a different use. Whereas the qualified variants for *minimum* and *maximum cardinality* restrictions are used slightly less in total numbers, they are used in 2 times more ontologies. *Exact qualified* cardinalities are almost used in 2 times more ontologies and additionally 8 times more in total numbers. Thus, qualified variants of all cardinality-based restriction types seem to be more popular for BioPortal ontologies.

Property-related restriction types Different property-related restriction types are used in 226 LOV and 219 BioPortal ontologies, corresponding to around 34% and 38% of LOV and BioPortal ontologies respectively. However, from those restriction types only *functional properties* and *universal quantification* are used to a larger extent in 22% and 17% of LOV ontologies respectively. These 2 restriction types show similar statistics for BioPortal ontologies, with the only difference that *universal quantification* restrictions are slightly more used than *functional properties* restrictions, in 26% and 24% of BioPortal ontologies respectively. The remaining property-related restriction types are barely used by the ontologies, ranging from 2% to 7% of ontologies for both LOV and BioPortal.

Ontologies using no restriction types We found 22 LOV and 25 BioPortal ontologies which do not contain any of our identified restriction types at all. Interestingly, the Dataset Usage Vocabulary (duv) from W3C[14], part of LOV ontologies, does contain a *subsumption* restriction type. However, their used `rdfs:subClassOf` expression is differently capitalized (`rdfs:subclassOf`), which does not comply to IRI-equality[15], and was thus not considered.

[13] Christine Golbreich et al., "OWL2 Web Ontology Language New Features and Rationale (Second Edition)", https://web.archive.org/web/20220121030134/https://www.w3.org/TR/owl2-new-features/#F5:_Property_Qualified_Cardinality_Restrictions (archived website accessed February 12, 2022)

[14] Bernadette Farias Loscio et al., "Data on the Web Best Practices: Dataset Usage Vocacbulary", https://www.w3.org/TR/vocab-duv/ (archived website accessed February 12, 2022)

[15] Richard Cyganiak et al., "RDF1.1 Concepts and Abstract Syntax", https://web.archive.org/web/20220208094051/https://www.w3.org/TR/rdf11-concepts/#section-IRIs (archived website accessed February 12, 2022)

2.1.5.2 Restriction Type Expressions

Besides occurrence of restriction types, *Montolo* provides information regarding occurrence of different restriction types expressions, allowing to compare different modeling practices. We provide different restriction type expressions for the following restriction types:*disjoint classes*, *disjoint properties* and *exact unqualified cardinality*.

Disjoint Classes The *disjoint classes* restriction type can be expressed using the single property expression `owl:disjointWith`, and the list-based expression `owl:AllDisjointWith`, for which we found that the single property expression is more popular in both LOV and BioPortal ontologies.

For the `owl:disjointWith` expression of the *disjoint classes* restriction type, we count 5,303 axiom statements in 155 LOV ontologies, and 133,738 axiom statements in 203 BioPortal ontologies. Although this expression is used in a similar number of ontologies among LOV and BioPortal, the BioPortal ontologies make significantly more use of it.

The `owl:AllDisjointWith` expression of the *disjoint classes* restriction type counts 3,642 axiom statements in 34 of LOV and 627,598 axiom statements in 85 of BioPortal ontologies.

The `owl:AllDisjointWith` expression is also used to a much larger extent by total numbers in BioPortal ontologies compared to LOV ontologies, indicating more machine-understandable disjointness which may facilitate reasoning tasks. However, only 5% of LOV and 15% of BioPortal ontologies use this expression.

Comparing the 2 different expressions for *disjoint classes* restriction type, we see differences between LOV and BioPortal. In LOV, the single property `owl:disjointWith` expression compared to the list-based `owl:AllDisjointWith` is used slightly more in total numbers, but in 4.5 times more ontologies. Similarly, in BioPortal the property-based expression compared to the list-based expression is used in 2 times more ontologies. However, in total numbers BioPortal ontologies encode 4 times more concepts using the list-based expression compared to the single property expression. This indicates that BioPortal ontologies using the list-based expression encode lots of mutual exclusive disjointness.

Disjoint Properties The *disjoint properties* restriction type can be expressed with the property expression `owl:propertyDisjointWith`, and the list-based expression `owl:AllDisjointProperties`, for which we found that the single property expression is more popular in both LOV and BioPortal ontologies.

The `owl:propertyDisjointWith` expression is used 920 times in 17 LOV and 45 times in 21 BioPortal ontologies.

The `owl:AllDisjointProperties` expression is used 424 times in 4 LOV and 9,070 times in 6 BioPortal ontologies. The property expression `owl:propertyDisjointWith` is used in 4 times more ontologies for both LOV and BioPortal ontologies. Even if a few of LOV and BioPortal ontologies heavily use the list-based expression `owl:AllDisjointProperties`, the overall trend suggests that the single property-based expression `owl:propertyDisjointWith` is more popular.

Cardinality Restrictions The *exact unqualified cardinality* restriction type can be expressed with the property `owl:cardinality`, and a combination of `owl:minCardinality` and `owl:maxCardinality` with the same value. The latter expression is barely or not used at all which indicates that the `owl:cardinality` expression is common practice to express *exact unqualified cardinality* in both LOV and BioPortal ontologies.

The `owl:cardinality` expression is used 1,375 times in 108 LOV and 692 times in 38 BioPortal ontologies. Compared to that, the combination of `owl:minCardinality` and `owl:maxCardinality` is used only 3 times in 2 LOV ontologies, and not used at all in BioPortal ontologies. This states the use of `owl:cardinality` is not just more popular, but common practice to express *exact unqualified cardinality* restrictions in LOV and BioPortal ontologies.

2.1.6 Conclusions

We discuss findings, *MontoloStats*' potential for ontology reuse, lessons learned, and future evaluation plans.

Findings Even though the selected repositories cover different domains (LOV is generic while BioPortal is domain-specific), both show same patterns with respect to restriction types use but not to the extent they use them. *MontoloStats* reveals that both LOV and BioPortal use RDFS-based and OWL-based restriction types to a similar extent, i.e. more than 95% of ontologies use RDFS-based restrictions but only half of them use OWL-based. However, the extent of their use differs. LOV ontologies contain much more *domain* and *range* restrictions compared to BioPortal, whereas BioPortal ontologies make considerably more use of *disjointness* restrictions. Furthermore, cardinality-based restrictions seem to be preferred by LOV in their *unqualified* variant whereas BioPortal uses more *qualified* cardinalities.

We also found that different literal-value related restriction types are not used at all or to a negligible extent. This raises questions: *why is there no need to express literal-value related restrictions?*, and if there is a need *where are literal-value related restrictions currently encoded?*

Ontology reuse *MontoloStats*' restriction type statistics can support ontology reuse activities concerning the assessment of relevant reuse candidates with respect to an application scenario.

MontoloStats indicates if an ontology contains e.g. a taxonomic structure (restriction type *subsumption*), or defines concepts in a machine-understandable way (using i.a. the restriction type *disjoint classes*). Such information is needed to assess the relevance of an ontology for different application scenarios, e.g. ontologies used for classification tasks ideally contain taxonomic information, but other application scenarios might rely on reasoning which likely benefits from a higher degree of axiomatization [5].

For each ontology in the *MontoloStats* dataset a dedicated website exists, listing the statistics and additional information about restriction types, i.e. definitions from their

descriptions in the *Montolo* dataset. Thus, restriction type statistics can be retrieved on-demand by an ontology engineer without any additional effort with respect to the setup of a tool chain.

Ontology Engineers may perform a comparative analysis of ontology reuse candidates considering external information. *MontoloStats* and restriction type definitions in *Montolo* are available as Linked Data, and, thus, SPARQL queries can be used to retrieve and combine different data sources to semi-automatically create reusable evaluation reports.

Lessons learned and Impact *MontoloStats* shows that almost half of LOV and BioPortal ontologies could be considered "*lightweight*" as they are less axiomatized. Currently, domain experts provide their knowledge and ontology engineers have to encode this knowledge in an optimal way, i.e. fulfilling all requirements while satisfying raising needs towards lightweight ontologies.

MontoloStats reveals that not all restriction types are used and those that are used are not equally used by different ontologies. We need to investigate both the roots of the observation, as well as its impact and consequences.

By comparing restriction modeling in LOV and BioPortal we found implicit modeling patterns with respect to restrictions. However, research focused on the definition of explicit methodological guidelines supporting ontology engineers in their tedious task of encoding restrictions still requires improvement. We need to better understand the restrictions and their implications compared to practical needs in an environment with changing requirements. *Are the restrictions properly modeled?*

MontoloStats reveals that not all restrictions are broadly used. However, it has not been thoroughly investigated so far how appealing ontology modeling tools are for defining restrictions. *Can the available tools support the creation of all restriction types? Are they appealing for the task at hand?*

Similarly, *MontoloStats* reveals that certain ontologies contain several restrictions and others not. However, the correlation between the number of restrictions per ontology and the ontology's reuse is not investigated so far. *Are the ontologies with restrictions and without equally (re)used? How does this influence if restrictions should be defined?* In the same context, it has not been investigated for ontologies how frequently each type of restriction is involved in knowledge graph quality issues and how this affects the evolution of the ontology. *Should we force certain restriction types found to be violated in datasets?*

Evaluation plan Given an ontology engineering-related ontology reuse scenario, a user study could investigate to which extent *MontoloStats* improves the discovery and selection of ontologies.

Regarding *ontology-discovery*, a modified version of LOV's search interface could provide users the function to filter search results based on the existence/non-existence of restriction types or restriction type expressions. Given scenarios where more or less restrictions are desired, users can report how useful the filter-functionality based on *MontoloStats* was

perceived, which restriction types they found the most useful to filter, and what information they might miss.

An *ontology-selection*-related task could similarly assess how users perceive the usefulness of *MontoloStats* when comparing ontologies. Additionally, the effectiveness of *MontoloStats* can be evaluated by comparing the amount and duration of steps to evaluate and compare ontology reuse candidates using *MontoloStats* versus manual inspection.

We plan to update the *MontoloStats* dataset regularly, but also to incorporate new restriction types and restriction type expressions into *Montolo*, identified e.g. by the community. New measures besides *occurrence* can be defined to gain a deeper understanding of restrictions use in ontologies. Last, we plan experiments to investigate the incorporation of *MontoloStats* into the LOV and BioPortal platform, to e.g. use restriction type statistics, as search filter or for results ranking.

2.2 Assessing and Analyzing the Use of Constraints for Data Shapes using Montolo

Sven Lieber, Ben De Meester, Anastasia Dimou, and Ruben Verborgh

Published as "Statistics about Data Shape Use in RDF Data", in *Proceedings of the 19th International Semantic Web Conference: Posters, Demos, and Industry Tracks*, Globally online, November 1-6, 2020, Pages 330-335.

Abstract

Statistics about constraint use in RDF data bring insights in common practices to address data quality. However, we only have such statistics for OWL axioms, not for constraint languages, such as SHACL or ShEx, that have recently become more popular. We extended previous work on axiom statistics to provide evidence of constraint type use. In this poster[16] we present preliminary statistics about the use of SHACL core constraints in data shapes found on GitHub. We found that class, datatype and cardinality constraints are predominantly used, similar to the dominant use of domain and range in ontologies. Less-used constraint types need further attention in visualization or modeling tools to address data quality issues. More constraints of SHACL but also ShEx need to be included to deepen the understanding. Data quality researchers and tool designers can make informed decisions based on the provided statistics.

2.2.1 Introduction

Recently, RDF constraint languages, such as SHACL [24] or ShEx [25], have been developed to model restrictions in the form of constraints on data. Statistics for OWL ontologies

showed that only a subset of possible axioms are commonly used [26], but such evidence does not yet exist for constraints which poses a gap and leaves users to anticipate possible use cases or cover whole specifications.

Insights about used constraint types can be taken from generated constraints or curated repositories. Astrea [27] and OSLO [28] which generate shapes from existing sources cover specific subsets of SHACL, but this is due to limited mapping and not because of evidence of broad use. To the best of our knowledge, only small repositories of SHACL constraints with less than 5 entries exist[17] [18].

In this poster paper, we present preliminary statistics generated by a constraint type extension of our Montolo framework [26] to collect RDF Data Cube compliant statistics about axiom use. Following the same approach, we used the vocabulary of Montolo[19] to create definitions for all SHACL core constraints and created statistics for identified data shapes from GitHub.

Our work provides insights in constraint type use and is extendible with respect to constraint types of other RDF constraint languages. Preliminary results, the created corpus of SHACL shapes as well as the tool to download the shapes are available with a persistent identifier (DOI: 10.5281/zenodo.3988930[20]) and under an open license[21] to attract more research. An updated version of the statistics for this dissertation is available as new version of the original statistics[22].

2.2.2 Constraint Type Statistics

We explain the framework to collect constraint type statistics, which sources we consider and present preliminary results before we discuss the results.

Framework We briefly describe the framework to collect constraint type statistics and the selection of SHACL data shapes. Montolo uses an extension of LODStats [7] to define statistical modules to detect (patterns of) RDF terms[23]. We created a statistical module for each core constraint of SHACL to detect SHACL serializations of constraint types, e.g., `sh:class` or `sh:minCount`. Additionally, we created definitions for SHACL core constraints with the Montolo vocabulary.

We searched for the term "SHACL" in GitHub and manually selected repositories which contain valid SHACL shapes that do not appear as simple examples. We also considered

[17] Konrad Abicht, "Schreckl SHACL Discovery Service", `https://web.archive.org/web/20210515223512/https://schreckl.inspirito.de/` (archived website accessed February 12, 2022)

[18] Thomas Francart, "SHACL Play! Shapes Catalog", `https://web.archive.org/web/20200920001136/http://shacl-play.sparna.fr/catalog` (archived website accessed February 12, 2022)

[19] `http://w3id.org/montolo/ns/montolo-voc`

[20] `https://zenodo.org/record/3988930`

[21] Creative Commons, "CC 1.0", `http://web.archive.org/web/20220212123024/https://creativecommons.org/publicdomain/zero/1.0/` (archived website accessed February 12, 2022)

[22] `https://zenodo.org/record/4154456`

[23] `https://github.com/IDLabResearch/lovstats`

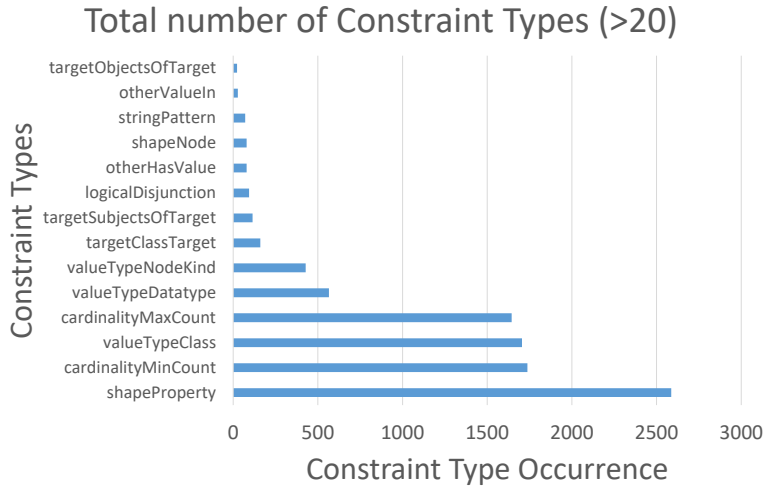


Figure 2.3: Constraints on properties, their cardinality and datatype or class are most frequently used in manually curated data shapes (excluding OSLO & schema.org' SHACL). Constraint types used less than 20 times are not shown.

common SHACL shapes, such as Schema.org's SHACL[24] SHACL constraints of SHACL itself[25]. We implemented a tool to download data shapes and merge the ones that conceptually belong together, e.g. because they are in the same repository; the tool is part of the accompanying resource of this paper.

Results In total, we analyzed the SHACL RDF files of 19 projects containing 2,037 NodeShapes. Two of the projects, the aforementioned OSLO and the SHACL version of schema.org are similar to the Astrea examples, i.e. data shapes generated based on a subset of SHACL. We describe statistics about constraint types of potentially manually curated SHACL shapes while comparing it with generated SHACL shapes of OSLO, schema.org and Astrea.

All constraint types are used (Figure 2.4) but constraint types regarding cardinality, class and datatype of properties are most frequently used by total number (Figure 2.3). Class and

[24] Holger Knublauch, "Schema.org (converted to SHACL by TopQuadrant)", `http://web.archive.org/web/20220209085046/https://datashapes.org/schema` (archived website accessed February 12, 2022)

[25] W3C, "A SHACL shapes graph to validate SHACL shape graphs", `http://web.archive.org/web/20220209093702/https://www.w3.org/ns/shacl-shacl` (archived website accessed February 12, 2022)

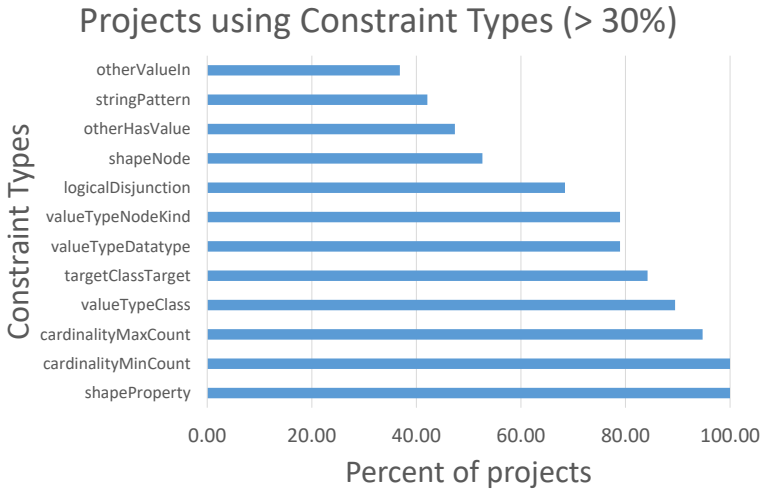


Figure 2.4: All constraint types are used, however, datatype, class and cardinality constraints of properties are most often used. Constraint types which are used in less than 30% of the projects are not shown.

datatype constraints are primarily found in our corpus which likewise is generated by Astrea, OSLO and SHACL of schema.org. This suggests that class and datatype constraints are main use cases for constraint types which find common use; it appears similar to the dominance of domain and range axioms for ontologies [26]. Disjunction constraints (`sh:or`) are used by more than 68% of the analyzed repositories and to a large extent by the automatically generated SHACL for schema.org. This can be explained by the flexibility of schema.org: properties are specified to expect one of several possible types. However, disjunction is almost non-existent in Astrea, showing that the selected ontologies barely contain `owl:unionOf` statements. Value range constraints (`sh:minExclusive`, `sh:maxInclusive`, etc) are barely found in our corpus and are neither generated for the Astrea examples nor OSLO, suggesting less future use, similar for other constraint types.

Discussion Constraint types complement ontology restrictions yet both show a similar use pattern. Our previous study on restrictions in ontologies found that taxonomic relationships (`rdfs:domain`, `rdfs:range`, `rdfs:subClassOf`) are extensively used whereas restrictions on literals were barely found. We see a similar pattern of constraint use compared to axiom use: relationships between concepts restricted to certain classes or datatypes. However, the

current analysis suggests that with respect to literals at least string patterns (`sh:pattern`) find some use in shapes which complements missing literal restrictions use of ontologies.

However, we see more potential in the use of constraints with respect to literals. One out of seven RDF statements in large knowledge graphs contains a literal as object [29]. Several string or literal value range constraint types are defined by SHACL and ShEx which can be used to impose precise restrictions on literals, e.g. quality-related patterns for book ISBN numbers or social security numbers, where uniformity is needed to properly query links between RDF resources. We have no insights with which tools the shapes were created yet this might be important. Current tools might focus too much on classes and datatypes while neglecting other constraint types. Appropriate tools with user-friendly interfaces are crucial and should be available such that users are made aware of possible constraint types and are assisted in using them.

Conclusion and Future Work Our preliminary results identified cardinality, class, datatype and disjunction constraints as commonly used. Developers of tools related to RDF constraints become able to iteratively implement their tools as they can cover first these commonly used constraint types. However, to exploit the existing data quality potential, developers should not neglect other constraint types completely especially regarding literal values. Future work can extend the statistics by including ShEx and extending the sample size. We currently work on visual notations for RDF constraints[26] which will benefit from this and future insights in constraint type use.

References

[1] Heiko Paulheim. "Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods". In: *Semantic Web Journal* 8.3 (Dec. 2016), pp. 489–508. ISSN: 2210-4968. DOI: 10.3233/SW-160218. URL: http://www.semantic-web-journal.net/content/knowledge-graph-refinement-survey-approaches-and-evaluation-methods.

[2] Elena Simperl and Christoph Tempich. "Ontology engineering: a reality check". In: *International Conference "On the Move to Meaningful Internet Systems"*. Springer. 2006, pp. 836–854. DOI: 10.1007/11914853_51.

[3] Nicola Guarino, Stati Uniti, and Pierdaniele Giaretta. "Ontologies and knowl-edge bases: towards a terminological clarification". In: *Towards Very Large Knowledge Bases*. IOS Press, 1995, pp. 25–32.

[4] Antonio De Nicola, Michele Missikoff, and Roberto Navigli. "A software engineering approach to ontology building". In: *Information systems* 34.2 (2009), pp. 258–275.

[26] `https://w3id.org/imec/unshacled/spec/shape-vowl` and `https://w3id.org/imec/unshacled/spec/shape-uml`

[5] Elena Simperl. "Guidelines for reusing ontologies on the semantic web". In: *International Journal of Semantic Computing* 4.02 (2010), pp. 239–283. DOI: 10.1142/S1793351X10001012.

[6] Elena Simperl, Christoph Tempich, and York Sure. "ONTOCOM: a cost estimation model for ontology engineering". In: *The Semantic Web - ISWC 2006*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 625–639. DOI: 10.1007/11926078_45.

[7] Sören Auer, Jan Demter, Michael Martin, and Jens Lehmann. "LODStats - An Extensible Framework for High-Performance Dataset Analytics". In: *EKAW 2012 : The 18th International Conference on Knowledge Engineering and Knowledge Management*. 2012.

[8] Jose Emilio Labra Gayo, Eric Prud'hommeaux, Iovka Boneva, and Dimitris Kontokostas. *Validating RDF Data*. Vol. 7. Synthesis Lectures on the Semantic Web: Theory and Technology 1. Morgan & Claypool Publishers LLC, Sept. 2017, pp. 1–328. DOI: 10.2200/s00786ed1v01y201707wbe016. URL: http://book.validatingrdf.com/.

[9] Dimitris Kontokostas, Patrick Westphal, Sören Auer, Sebastian Hellmann, Jens Lehmann, Roland Cornelissen, and Amrapali Zaveri. "Test-driven evaluation of linked data quality". In: *Proceedings of the 23rd international conference on World Wide Web*. Ed. by Chin-Wan Chung. New York, NY, United States: Association for Computing Machinery, Apr. 2014, pp. 747–757. ISBN: 9781450327442. DOI: 10.1145/2566486.2568002. URL: http://dl.acm.org/citation.cfm?id=2568002.

[10] Dörthe Arndt, Ben De Meester, Anastasia Dimou, Ruben Verborgh, and Erik Mannens. "Using Rule-Based Reasoning for RDF Validation". In: *Rules and Reasoning: International Joint Conference, RuleML+RR 2017, London, UK, July 12–15, 2017*. Ed. by Stefania Constantini, Enrico Franconi, William Van Woensel, Roman Kontchakov, Fariba Sadri, and Dumitru Roman. Vol. 10364. Lecture Notes in Computer Science. Cham: Springer, July 2017, pp. 22–36. DOI: 10.1007/978-3-319-61252-2_3.

[11] Thomas Hartmann. "Validation Framework for RDF-based Constraint Languages". PhD thesis. Karlsruher Institut für Technologie (KIT), 2016. DOI: 10.5445/ir/1000056458. URL: http://digbib.ubka.uni-karlsruhe.de/volltexte/1000056458.

[12] Elena Simperl. "Reusing ontologies on the Semantic Web: A feasibility study". In: *Data and Knowledge Engineering* 68.10 (2009), pp. 905–925. ISSN: 0169-023X. DOI: 10.1016/j.datak.2009.02.002.

[13] Pierre-Yves Vandenbussche, Ghislain A. Atemezing, María Poveda-Villalón, and Bernard Vatant. "Linked Open Vocabularies (LOV): a gateway to reusable semantic vocabularies on the Web". In: *Semantic Web Journal* 8.3 (Dec. 2016), pp. 437–452. DOI: 10.3233/SW-160213. URL: http://www.semantic-web-journal.net/content/linked-open-vocabularies-lov-gateway-reusable-semantic-vocabularies-web-1.

[14] M Musen, N Shah, N Noy, Benjamin Dai, Michael Dorf, N Griffith, JD Buntrock, Clement Jonquet, MJ Montegut, and Daniel L Rubin. "BioPortal: ontologies and data resources with the click of a mouse". In: *AMIA Annu Symp Proc*. Vol. 6. 2008, pp. 1223–1224.

[15] María Poveda-Villalón, Asunción Gómez-Pérez, and Mari Carmen Suárez-Figueroa. "OOPS! (OntOlogy Pitfall Scanner!): An on-line tool for ontology evaluation". In: *International Journal on Semantic Web and Information Systems (IJSWIS)* 10.2 (2014), pp. 7–34. ISSN: 1552-6283. DOI: 10.4018/ijswis.2014040102.

[16] Andreas Langegger and Wolfram Woss. "RDFStats - An Extensible RDF Statistics Generator and Library". In: *Proceedings of the 20th International Workshop on Database and Expert Systems Applications*. Los Alamitos, Calif.: IEEE Computer Society, 2009, pp. 79–83. ISBN: 978-0-7695-3763-4. DOI: 10.1109/DEXA.2009.25.

[17] Nandana Mihindukulasooriya, María Poveda-Villalón, Raúl García-Castro, and Asunción Gómez-Pérez. "Loupe-An Online Tool for Inspecting Datasets in the Linked Data Cloud." In: *International Semantic Web Conference (Posters & Demos)*. Vol. 1. 1. 2015, p. 2.

[18] Mark A. Musen. "The Protégé Project: A Look Back and a Look Forward". In: *AI Matters* 1.4 (June 2015), pp. 4–12. DOI: 10.1145/2757001.2757003.

[19] Niyati Baliyan and Sandeep Kumar. "Towards measurement of structural complexity for ontologies". In: *International Journal of Web Engineering and Technology* 11.2 (2016), p. 153. ISSN: 1476-1289. DOI: 10.1504/IJWET.2016.077343.

[20] Antonio De Nicola and Michele Missikoff. "A lightweight methodology for rapid ontology engineering". In: *Communications of the ACM* 59.3 (Feb. 2016). UPON lite method paper, pp. 79–86. DOI: 10.1145/2818359.

[21] Dominik Tomaszuk. "Inference rules for OWL-P in N3Logic". In: *Communication Papers of the 2018 Federated Conference on Computer Science and Information Systems*. Ed. by M. Ganza, L. Maciaszek, and M. Paprzycki. Vol. 17. ACSIS. Polskie Towarzystwo Informatyczne, Sept. 2018, pp. 27–33. DOI: 10.15439/2018f102.

[22] James A Overton, Heiko Dietze, Shahim Essaid, David Osumi-Sutherland, and Christopher J Mungall. "ROBOT: A command-line tool for ontology development". In: *International Conference on Biomedical Ontology (ICBO)*. Vol. 1515. CEUR Workshop Proceedings. CEUR-WS.org, July 2015.

[23] Chris Little and Simon Cox. *Time Ontology in OWL*. Candidate Recommendation. https://www.w3.org/TR/2020/CR-owl-time-20200326/. World Wide Web Consortium (W3C), Mar. 2020.

[24] Holger Knublauch and Dimitris Kontokostas. *Shapes Constraint Language (SHACL)*. Recommendation. World Wide Web Consortium (W3C), July 2017. URL: https://www.w3.org/TR/shacl/.

[25] Eric Prud'hommeaux, Jose Emilio Labra Gayo, and Harold Solbrig. "Shape expressions: an RDF validation and transformation language". In: *Proceedings of the 10th International Conference on Semantic Systems*. Ed. by Harald Sack, Agata Filipowska, Jens Lehmann, and Sebastian Hellmann. ACM. New York, NY, United States: Association for Computing Machinery, 2014, pp. 32–40. DOI: 10.1145/2660517.2660523. URL: http://dl.acm.org/citation.cfm?id=2660523.

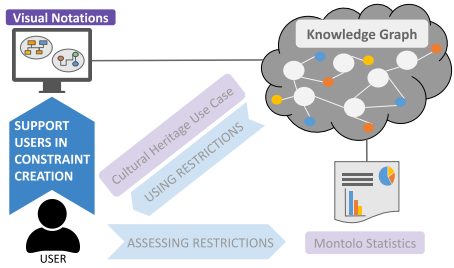
[26] Sven Lieber, Ben De Meester, Anastasia Dimou, and Ruben Verborgh. "MontoloStats – Ontology Modeling Statistics". In: *Proceedings of the 10th International Conference on Knowledge Capture - K-CAP '19*. Ed. by Raphaël Troncy. ACM, Nov. 2019, pp. 69–76. DOI: 10.1145/3360901.3364433.

[27] Andrea Cimmino, Alba Fernández-Izquierdo, and Raúl García-Castro. "Astrea: Automatic Generation of SHACL Shapes from Ontologies". In: *European Semantic Web Conference (ESWC)*. Ed. by Andreas Harth, Sabrina Kirrane, Axel-Cyrille Ngonga Ngomo, Heiko Paulheim, Anisa Rula, Peter Haase, and Michael Cochez. Springer. Springer International Publishing, 2020, pp. 497–513. DOI: 10.1007/978-3-030-49461-2_29.

[28] Dieter De Paepe, Geert Thijs, Raf Buyle, Ruben Verborgh, and Erik Mannens. "Automated UML-Based Ontology Generation in OSLO²". In: *The Semantic Web: ESWC 2017 Satellite Events – ESWC 2017*. Ed. by Eva Blomqvist, Katja Hose, Heiko Paulheim, Agnieszka Ławrynowicz, Fabio Ciravegna, and Olaf Hartig. Vol. 10577. Lecture Notes in Computer Science. Springer, Cham, 2017, pp. 93–97. DOI: 10.1007/978-3-319-70407-4_18.

[29] Wouter Beek, Filip Ilievski, Jeremy Debattista, Stefan Schlobach, and Jan Wielemaker. "Literally better: Analyzing and improving the quality of literals". In: *Semantic Web* 9 (2018), pp. 131–150.

Chapter 3



Creation of Constraints Using Visual Notations

Whereas the last chapter focused on the assessment of already encoded restrictions, this chapter focuses on supporting users in the creation of restrictions. User evaluations of visualizations for Knowledge Graph-related concepts suggest that such visualizations support users to perform respective tasks more intuitively [1, 2]. Ways to visualize axioms in ontologies were already presented in the past [3, 4]. However, constraint languages for RDF such as the W3C recommended SHACL are relatively new and currently no visualization was proposed which can cover all SHACL core constraints.

This chapter presents the following contributions to constraint creation: the visual notations ShapeUML and ShapeVOWL are proposed that cover all SHACL core constraints. We compared both notations using cognitive effective design principles and performed a comparative user study.

We address Research Question 2 "How can we support users familiar with Linked Data in viewing RDF constraints?" and validate Hypothesis 2 "Users familiar with Linked Data can answer questions about visually represented RDF constraints more accurately with a VOWL-based visual notation than with an UML-based visual notation".

This chapter corresponds with the publication "Visual Notations for Viewing RDF Constraints with UnSHACLed".

*

Sven Lieber, Ben De Meester, Pieter Heyvaert, Femke Brückmann, Ruben Wambacq, Erik Mannens, Ruben Verborgh, and Anastasia Dimou

Published as "Visual Notations for Viewing RDF Constraints with UnSHACLed", in *Semantic Web Journal*, 2021, vol. pre-press, Pages 1-36.

Abstract

The quality of knowledge graphs can be assessed by a validation against specified constraints, typically use-case specific and modeled by human users in a manual fashion. Visualizations can improve the modeling process as they are specifically designed for human information processing, possibly leading to more accurate constraints, and in turn higher quality knowledge graphs. However, it is currently unknown how such visualizations support users when *viewing* RDF constraints as no scientific evidence for the visualizations' effectiveness is provided. Furthermore, some of the existing tools are likely suboptimal, as they lack support for *edit operations* or common constraints types. To establish a baseline, we have defined visual notations to represent RDF constraints and implemented them in *UnSHACLed*, a tool that is independent of a concrete RDF constraint language. In this paper, we (i) present two visual notations that support all SHACL core constraints, built upon the commonly used visualizations VOWL and UML, (ii) analyze both notations based on cognitive effective design principles, (iii) perform a comparative user study between both visual notations, and (iv) present our open source tool UnSHACLed incorporating our efforts. Users were presented RDF constraints in both visual notations and had to answer questions based on visualization task taxonomies. Although no statistical significant difference in mean error rates was observed, all study participants preferred *ShapeVOWL* in a self assessment to answer RDF constraint-related questions. Furthermore, *ShapeVOWL* adheres to more cognitive effective design principles according to our performed comparison. Study participants argued that the increased visual features of *ShapeVOWL* made it easier to spot constraints, but a list of constraints – as in *ShapeUML* – is easier to read. However, also that both more deviations from the strict UML specification and introduction of more visual features in *ShapeUML* can improve *ShapeUML*. From these findings we conclude that *ShapeVOWL* has a higher potential to represent RDF constraints more effective compared to *ShapeUML*. But also that the clear and efficient text encoding of *ShapeUML* can be improved with visual features. A one-size-fits-all approach to RDF constraint visualization and editing will be insufficient. Therefore, to support different audiences and use cases, user interfaces of RDF constraint editors need to support different visual notations. In the future, we plan to incorporate different editing approaches, informed by visualization task taxonomies, and non-linear workflows into *UnSHACLed* to improve its *editing* capabilities. Further research can be built upon our findings and evaluate a *ShapeUML* variant with more visual features or investigate a mapping from both visual notations to ShEx constraints.

3.1 Introduction

Data interoperability is one of the biggest challenges of the current era and the Resource Description Framework (RDF) offers a solution as it is compositional: RDF graphs from different sources can be merged automatically which facilitates the integration of heterogeneous data [5]. However, advantages such as RDF's flexibility also result in challenges

such as the production/consumption dilemma [5] in which the structure of data needs to be described such that producers and consumers can validate transmitted data for reasons such as security or performance [5]. In 2017, the W3C *RDF Data Shapes Working Group* published a recommendation to define structural constraints of RDF data [6] which can address such needs.

Quality is defined as "fitness for use" [7] implying that constraints for validation are use-case specific; human users usually define these constraints in a manual fashion and need support. Users can use any text editor to create such constraints, but need to be familiar with the textual syntax of the underlying data shape language. User evaluations of visualizations for different Linked Data concepts, such as ontology modeling [1] or Linked Data generation [2], suggest that such visualizations support users to perform respective tasks more intuitively. However, the degree of actual support offered by existing visualizations for RDF constraints is currently unknown, given the lack of scientific evidence for their effectiveness. Furthermore, some of the existing tools are likely suboptimal, as they lack support for edit operations or common constraints types.

Clearly specified visualizations – already used for some Semantic Web concepts [1, 8, 9, 2] – provide a design rationale and can be designed with the human information processing system in mind [10], but are not yet taken into account for RDF constraints which makes the effectiveness of existing tools questionable. A visual notation [10] is defined as a set of graphical symbols, a set of compositional rules, as well as the definitions and meaning of each symbol, and provides an explicit design rationale. UnSHACLed [11], a tool built on top of SHACL [6], lists features for a visual data shape editor. However, important details regarding the used visual notation are not provided, for instance, the meaning of arrows or the selection of colors are not clearly specified. Similarly, RDFShape which uses "UML-like class diagrams" [12] to visualize ShEx [13] constraints does not provide a clear specification of its visual notation and neither do other recently developed tools[1] [2].

Existing tools only provide limited or no *editing capabilities*, if editing capabilities are provided they are not always in line with real-life constraint use. The first version of UnSHACLed supports constraints editing. However, it does not support all constraint types, for instance, logical constraints are not yet visualized. RDFShape does not support constraints *editing* at all as it only visualizes constraints, thus users need to use and understand the underlying textual syntax. Similar to the initial version of UnSHACLed, the implementation of RDFShape does not yet support logical relationships such as (exclusive) disjunction; recent statistics show that *disjunction constraints* are broadly used [14] and thus users probably have the need to *create and edit* such constraints.

[1] Natanael Arndt, "OntoPad", `https://web.archive.org/web/20201104091304/https://github.com/AKSW/OntoPad/` (archived website accessed February 12, 2022)

[2] Andre Valdestilhas, "shaclEditor", `https://web.archive.org/web/20201104091927/https://github.com/firmao/shaclEditor`(archived website accessed February 12, 2022)

3.1.1 Research question and approach

The aforementioned motivate our high-level research question: *How can we support users familiar with Linked Data in viewing RDF constraints?* To address this research question, we investigated *visual notations* supporting users when *viewing* RDF constraints. Furthermore, we present a new version of our tool *UnSHACLed* that implements visual notations and allows users to *create and edit* RDF constraints.

A few visual notations already exist, but are not formally defined or do not cover all SHACL core constraints which also prevents a fair comparison. Thus, we defined two visual notations to represent all SHACL core constraints and related concepts by reusing existing notations. Different candidates to reuse exist, i.e. commonly used visual notations already familiar to users. Both the Unified Modeling Language (UML) [15] and the Visual Notation for OWL Ontologies (VOWL) [1] can be considered for a visual notation for RDF constraints as they are commonly used for RDF constraints or related Semantic Web concepts [11, 12, 3, 4, 1, 8, 2].

3.1.2 Hypothesis

We defined the two visual notations *ShapeUML* and *ShapeVOWL* both representing all SHACL core constraints and related concepts. Since *VOWL*, the underlying notation of *ShapeVOWL* aims to be intuitive and comprehensible [1] and visualizes the tangible graph structure of RDF, we investigate in this paper the following hypothesis: "Users familiar with Linked Data can answer questions about visually represented RDF constraints more accurately with a VOWL-based visual notation than with an UML-based visual notation"

3.1.3 Contributions

We compare the notations with respect to design principles for visual notations [10] derived from several seminal works in human cognition [16, 17, 18, 19, 20] and evaluate them in a comparative user study[3]. We implemented both visual notations in *UnSHACLed* to allow *creating and editing* constraints in a constraint language independent way. Users can switch between visual notations and use the created RDF constraints to validate input data from within the same editor.

Our contributions in this paper are:

1. introduction of two alternative visual notations: *ShapeUML* and *ShapeVOWL*;
2. analysis of both visual notations with respect to cognitive effective design principles;
3. comparative evaluation between ShapeVOWL and ShapeUML with a user study; and
4. presentation of our open source UnSHACLed editor implementing both visual notations.

[3] Material: `https://doi.org/10.6084/m9.figshare.13614440.v2`

The comparative analysis based on cognitive effective design principles [10] reveals that *ShapeVOWL* adheres to more principles, thus in theory is more cognitively effective. An additional comparative user study shows that there is no significant mean error difference when answering questions about RDF constraints with both notations, however, also that in a self-assessment users prefer *ShapeVOWL*. We implemented both visual notations in our tool *UnSHACLed* to also allow editing of RDF constraints in a visual fashion.

The remainder of the paper is structured as follows. We provide background knowledge on data shape languages and visual notations in Section 3.2 and present two visual notations in Section 3.3. In Section 3.4 we compare both presented visual notations based on design principles for cognitive effective visualizations. In Section 3.5 we present our visual editor UnSHACLed. In Section 3.6 we present the comparative user evaluation and its results. We discuss and conclude in Section 3.7.

3.2 State of the Art

In this section, we discuss (i) existing RDF constraint languages (ii) the use of different constraint types suggesting visualizations for manual creation, (iii) existing RDF constraint visualization tools, (iv) closely related Semantic Web visualizations providing possible visualizations to extend, (v) visual notations for human cognition, and (vi) visualization tasks describing the interaction between humans and visualizations.

3.2.1 RDF constraint languages

Several RDF constraint languages were proposed in the past, we describe how they are related. In this work we consider the Shapes Constraint Language (SHACL) because it (i) is a W3C recommendation, (ii) clearly defines constraint types in its core specification, and (iii) has a significant intersection with the Shape Expressions Language (ShEx) [5], a widely used RDF constraint language.

SPARQL Inference Notation (SPIN) [21] was the earliest W3C member submission (2011). A syntax and a vocabulary were defined to describe constraints and inference rules based on SPARQL.

A few years later in 2014 another two W3C member submission were submitted: the **Resource Shape (ReSh)** [22, 23] which defines a high-level RDF vocabulary to specify the shape of RDF resources and the grammar-based **Shape Expressions Language (ShEx)** [24]. ShEx was inspired by ReSh yet provides more expressivity [13].

The **Shapes Constraint Language (SHACL)** [6] became a W3C recommendation in 2017 and is seen as the legitimate successor of SPIN [25]. SHACL is a constraint language for describing and validating RDF graphs. It defines a RDF vocabulary to define constraints and a specified validation process to validate RDF data based on described constraints: data graph nodes are validated with data shape graph constraints and a validation report in RDF following the SHACL vocabulary is generated. Furthermore, SHACL provides 31 core constraint types and other concepts related to validation both defined using the aforementioned vocabulary. These other concepts comprise (i) *a targeting mechanism* to assign data

graph nodes to data shape graph constraints, (ii) *property paths* to further specify on which reachable node properties constraints apply, (iii) *severity* of data shapes as annotation to indicate the severity of a constraint violation in the validation report, (iv) *deactivation* of data shapes to exclude them from the validation process, and (v) *non-validating characteristics* to annotate data shapes.

3.2.2 Creating Constraints

More than eighty constraint types were identified [26] from which a subset is used as axioms in ontologies [27] and a subset motivated the creation of SHACL [28]. Existing approaches to generate RDF constraints use UML diagrams or ontologies as source but usually cover only a limited subset of SHACL core constraint types due to an incomplete mapping. We count SHACL core constraint types based on the "Core Constraint Components" of SHACL specification [6].

The **Open Standards for Linking Organizations (OSLO)** initiative of Flanders, Belgium generates SHACL constraints annotated UML models [29] representing RDF classes and properties. The generated SHACL constraints are limited to a subset of constraint types, i.e. cardinality, class, and datatype, therefore only supporting 3 out of 31 SHACL core constraint types.

Automatic Generation of SHACL Shapes from Ontologies (Astrea) [30] is based on a mapping of conceptual restrictions between patterns of OWL axioms and SHACL constraints. These patterns only contain 20 out of the 31 SHACL core constraint types when counting the core constraint types of the SHACL specification and not their parameterizations. For instance, we count the constraint type `sh:nodeKind` once and we do not count its parameterizations, such as "`sh:nodeKind sh:Literal`" or "`sh:nodeKind sh:BlankNode`". Besides these core constraints, Astrea also covers other concepts of the SHACL specification, namely *property paths* and terms related to *targeting* which applies elements of the shapes graph to elements of the data graph; we also support these concepts and additionally the concepts of *deactivation* and *severity* of data shapes.

TopQuadrant generated SHACL constraints from the **RDFa of the schema.org vocabulary**[4] These constraints consist of the constraint types class, datatype and disjunction, i.e. only 3 out of 31 constraint types.

Manually created RDF constraints are theoretically not limited by any mapping as a user potentially can use all constraint types of a specification. However, similar to ontology axioms [27] only a subset seems to find common use. In our previous work [14] and later updated and extended statistics[5], we investigated the use of constraint types in SHACL shapes. We found that 30 out of 31 constraint types were used, but only a few are used in more than 60 percent of surveyed GitHub repositories: value type (class, datatype, nodekind), cardinality and disjunction constraints. Thus, RDF constraint visualizations and editors should *at least* cover these commonly used constraint types; however, to avoid a self-fulfilling

[4] Holger Knublauch, "Schema.org (converted to SHACL by TopQuadrant)", `http://web.archive.org/web/20220209085046/https://datashapes.org/schema` (archived website accessed February 12, 2022)

[5] `https://zenodo.org/record/4154456`

prophecy where such a limitation reinforces the use of already commonly used constraint types, editors should not be limited to *only* these constraint types either.

3.2.3 RDF Constraint Editors

Tools to edit RDF constraints already exist but are either based on a specific textual syntax or have no formally defined visual notation.

Fajar et al. [31] implemented a **SHACL editor as plugin for Protégé**. However, their plugin is text-based and does not use a visualization for RDF constraints, therefore users are required to learn a specific RDF constraint language. Similarly, the tool **ShapeDesigner** from Boneva et al. [32] provides a text-based interface in which users are confronted with ShEx and SHACL representations of RDF constraints.

De Meester et al. [11] list features for a visual data shape editor implemented in an early version of the visual editor **UnSHACLed**. Although a few comments regarding the visualization were made, important details are not specified. For instance, the meaning of arrows or the selection of colors is not clearly specified, preventing developers of other tools from effectively implementing the visual notation. As a result, the original visualization of UnSHACLed is coupled to the tool hampering the accessibility for users across tools.

RDFShape [12] considers UML-like class diagrams. However, it does not cover all commonly used constraint types and, similarly to UnSHACLed, does not specify all details of how RDF constraints are visualized. The tool visualizes RDF constraints without the possibility of editing the constraints via the visualization and, currently, does not support logical relationships, e.g. (exclusive) disjunction[6], – commonly used according to preliminary statistics [14]. Even though support for additional constraint types can be implemented, it is not specified how it should be visualized, leaving room for different interpretations.

OntoPad[7] and **shaclEditor**[8]. are visual editors for RDF providing a way to visually interact with SHACL shapes. Similar to the early version of UnSHACLed, neither of these two editors provide a formally specified visual notation.

The desktop application **SHAPEness**[9] visualizes RDF constraints and also allows visual editing, recently a user study was performed to evaluate the application (currently under review [33]). However, even though the user study with this tool reported that expectations of expert users were met, the visualization of the constraints is – similar to the other tools mentioned above – coupled to the tool and not formally defined.

---

[6] Jose Emilio Labra Gayo, "umlShaclex", https://github.com/weso/umlShaclex/blob/06230fc568d0d91d443bb9ae819b9a1e65c6cc4e/src/main/scala/es/weso/uml/ShEx2UML.scala#L112 (website accessed February 12, 2022)

[7] Natanael Arndt, "OntoPad", https://web.archive.org/web/20201104091304/https://github.com/AKSW/OntoPad/ (archived website accessed February 12, 2022)

[8] Andre Valdestilhas, "shaclEditor", https://web.archive.org/web/20201104091927/https://github.com/firmao/shaclEditor(archived website accessed February 12, 2022)

[9] Rossana Paciello et al., "SHAPEness METADATA EDITOR", https://web.archive.org/web/20220218225301/https://epos-eu.github.io/SHAPEness-Metadata-Editor/ (archived website accessed February 18, 2022)

Commercial tools with support for RDF constraints include TopBraid Composer, AllegroGraph, Stardog, GraphDB and Metaphactory.

The tool **TopBraid EDG** from TopQuadrant, accessible as demo version from the free TopBraid Composer Maestro edition, visualizes SHACL using UML diagrams. Node shapes are visualized as rectangles, properties of related property shapes are listed within this rectangle, and constraints are visualized in colored font and/or as relationships[10]. However, similarly to other tools listed above, this visualization is coupled to the tool and no dedicated specification of the visualization exists. Furthermore, these diagrams are not yet interactive, i.e. the constraints can not be edited using the visualization[11].

The triple stores/knowledge graph systems **AllegroGraph**[12], **Stardog**[13] and **GraphDB**[14] support SHACL for validation, but do not provide visualizations of constraints. Similarly, the knowledge graph management system **metaphactory** [34] supports SHACL, but only visualizes validation reports and not constraints.

3.2.4 Semantic Web Visualizations

We look into the visualization of other Semantic Web concepts because they might be relevant for the visualization of RDF constraints.

UML is often used for modeling ontologies. The creation of constraints on RDF data from a conceptual point of view shows similarities to the creation of axioms in an ontology. Thus, visualizations for ontologies would be expected to be applicable to RDF constraints as well. A simple version of UML is used within the structural specification of OWL [35] to visualize the definition of conceptual restrictions in the form of axioms. Cranefield and Purvis [3] demonstrate how a subset of UML and the associated Object Constraint Language (OCL) [36] is used to model ontologies. Even the Object Management Group (OMG) – which maintains the UML specification – defined a specific UML profile for OWL and RDF, the Ontology Definition Metamodel (ODM) [4].

A plethora of ontology visualizations exists, but **VOWL** appears to be the most prominent visualization with respect to practical use and user familiarity for several concepts related to RDF constraints. Combining findings of several surveys [37, 38, 39, 40, 41, 42] and two works [43, 44] presenting visualization tools, 84 ontology visualization tools were identified. *Widoco* [45], a widely used tool to create ontology documentations, uses *WebVOWL* [46] to visualize ontologies. WebVOWL implements the Visual Notation for OWL Ontologies

[10] Irene Polikoff, "Overview of TopBraid EDG Ontologies", `https://web.archive.org/web/20201201152524/https://www.topquadrant.com/edg-ontologies-overview/` (archived website accessed February 12, 2022)

[11] We received information from TopQuadrant that this visualization will get a significant re-work in the future, including interactive diagram and new styling.

[12] AllegroGraph, "AllegroGraph", `https://web.archive.org/web/20220130075556/https://allegrograph.com/products/allegrograph/` (archived website accessed February 12, 2022)

[13] Stardog, "The Enterprise Knowledge Graph Platform", `https://web.archive.org/web/20220202104647/https://www.stardog.com/` (archived website accessed February 12, 2022)

[14] Ontotext, "GraphDB", `https://web.archive.org/web/20220121021823/https://graphdb.ontotext.com/` (archived website accessed February 12, 2022)

(VOWL) [1]. VOWL is also implemented as a plugin for the commonly used modeling tool Protégé in ProtégéVOWL [47]. This suggests that users who use ontologies and read their documentations have at least encountered a VOWL-based visualization. Besides ontologies, VOWL-based visualizations also exist for queries [8], Linked Data visualization [9] and generation [2], all closely related to RDF constraints.

3.2.5 Visual Notations for Human Cognition

Visual notations are created for human users, thus works related to perception and cognition are relevant to our work. We outline how the most relevant frameworks are combined in the **design theory of Moody** [10] and its design principles, which we therefore consider for an analysis of our proposed visual notations. For a detailed list of works Moody's design theory is based on, we refer the reader to the "Physics Of Notation" [10].

Moody's design theory is based on **communication theory** [16] in which a diagram-creator encodes an intended message using a visual notation and a diagram-user decodes this message to retrieve the intended meaning [10]. Moody defines design principles that comprise existing theories and empirical findings to create visual notations for the human perceptual processing (seeing) and cognitive processing (understanding) [10], and thus aimed for optimized decoding by humans.

From a perceptual perspective, this decoding is described based on works related to **Gestalt principles** [18] and **feature integration theory** [17], the organization of visual stimuli into structures, respectively their pre-attentive and parallel detection by humans. Such decoding features influence the encoding for which Moody relies on **Bertin's work on Semiology of Graphics** [19], i.e. defined visual variables such as shape, size or color which can be used to formally define a visual notation. Furthermore, the design principles include measures of anomalies in the correspondence between symbols of the visual notation and the semantic constructs they represent. Therefore it can be measured whether a visual notation fulfills the requirements of a notational system according to **Goodman's theory of symbols** [20].

Considering cognition, slower conscious processes such as retrieving prior knowledge from long-term memory are involved. Prior knowledge in this context may refer to already familiar visual notations. **Dasgupta** [48] identified a familiarity bias, i.e. experts prefer using familiar but suboptimal notations with which the experts perform worse compared to non-familiar optimal visual notations. We explicitly address such prior knowledge as we base our designed visual notations on the already familiar and broadly used visual notations UML and VOWL, thus aiming for the sweet spot between familiar notations and optimal design.

3.2.6 Visualization Tasks

The interaction between humans and visualizations can be systematically described using visualization tasks, this allows us to consider a common set of tasks for our user study.

Brehmer and Munzner [49] reviewed more than 20 works to define a typology of user tasks, on the one hand powerful enough to describe the why (intention), how (interaction)

and what (input/output) aspect of visualization tasks, and on the other hand aligning visual tasks of all previous works; therefore their work also covers the seminal works of **Wehrend and Lewis** [50], **Zhou and Feiner** [51], and **Amar et al.** [52], which provide visual task taxonomies and exemplified tasks.

Such taxonomies were evaluated in user studies for example by **Morse and Lewis** [53] in the form of visual prototypes, or by **Valiati et al.** [54] for multidimensional visualizations. **Saket et al.** [55] performed a user study to compare tabular data to other visualization types by instantiating questions from visualization tasks of the taxonomy from Amar et al. [52]. Similarly, for our work – in which we compare two different visual notations representing the same data – we instantiate questions from this taxonomy which, based on a previous alignment [49], could also be annotated with intents and interactions to further investigate editing approaches in the future.

3.3 Visual Notations

We introduce two visual notations for RDF constraints to establish a baseline for a fair comparison, we provide general design considerations for both notations, *ShapeUML* (based on UML) and *ShapeVOWL* (based on VOWL). Both visualize fundamental constructs of RDF constraint languages: *constraints* and the context in which they are applied, i.e. *data shapes*. We describe which visual variables are used as graphical primitives for both notations, following Moody [10] and thus make design decisions transparent. Cognitive effective design principles [10] where taken into account where applicable, a detailed comparison between both notations based on these principles can be found in the next section (Section 3.4).

Both notations have different visual features and represent all SHACL core constraints and additionally concepts related to *targeting*, *property paths*, *severity* and *deactivation*; although both notations are built based on SHACL, they are constraint language independent and semantic constructs of other constraint languages can be mapped to it. Thus, both notations represent the same semantic constructs and their only difference are their visual features, enabling a fair comparison. Currently the visual notations visualize all SHACL core constraints, where necessary with (additional) constraint-language-independent text labels; Figures 3.1, 3.2, 3.5 and 3.6 list all SHACL core constraints and the other supported concepts together with a corresponding terminology mapping used by our notations *ShapeUML* and *ShapeVOWL*.

3.3.1 ShapeUML

The notation ShapeUML is based on the Ontology Definition Metamodel (ODM) [4] in which both nodes and properties are first-class UML constructs and, thus, graphically represented as class diagram boxes (rectangle). Therefore, constraints on both nodes and properties can be expressed and logical relationships between different types of data shapes can be visualized.

The graphical primitives of *ShapeUML* are the following visual variables [10]: shape, edge, text, border and position. The full specification is available at `https://w3id.org/`

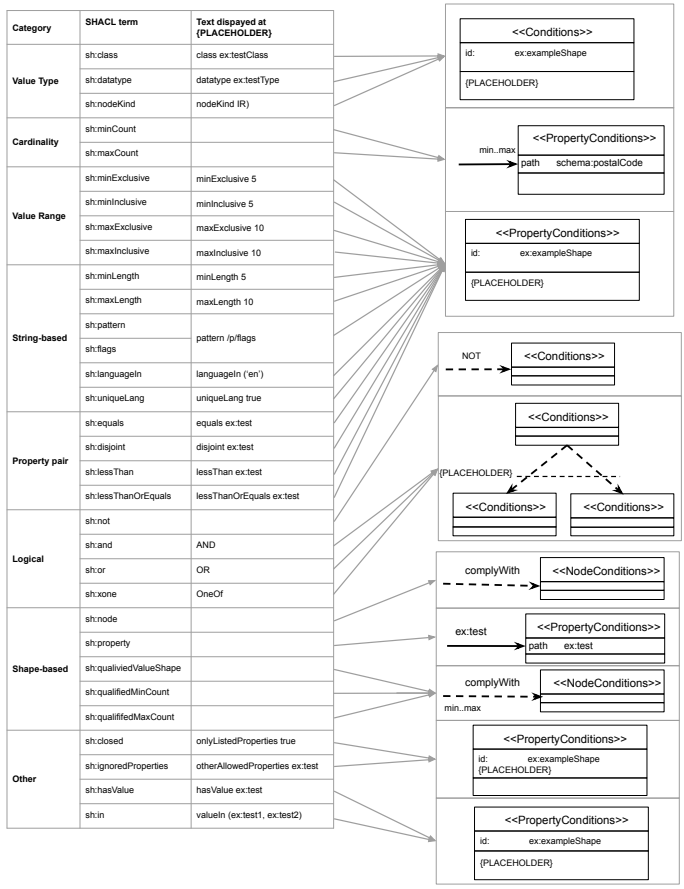


Figure 3.1: Correspondence between semantic constructs and *ShapeUML*: SHACL core constraints (left) and graphical notations (right).

`imec/unshacled/spec/shape-uml/20210118/`. In the remaining, we describe the graphical primitives and elaborate with an example.

3.3.1.1 Shape

We **reuse classes (rectangles)** from UML [15] to represent both node and property shapes, **redefine the meaning of rectangle's compartments** for RDF constraint specifics, **introduce data shape stereotypes** to indicate a data shape's type and distinguish it from other UML rectangles representing other concepts.

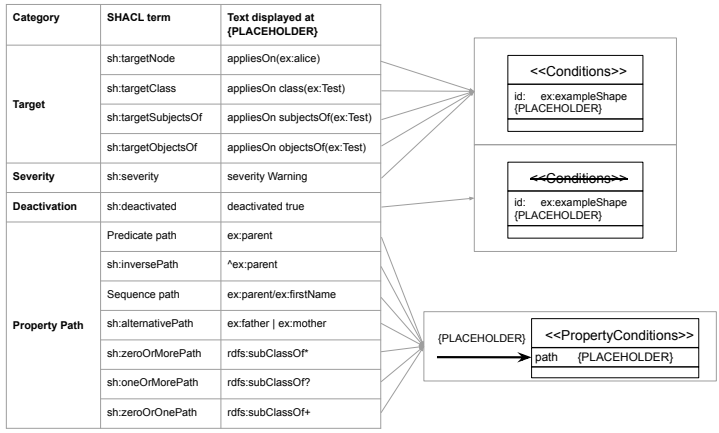


Figure 3.2: Correspondence between semantic constructs and *ShapeUML*: other relevant SHACL concepts besides core constraints (left) and graphical notations (right).

We use the graphical primitive *shape* to represent the fundamental construct *data shapes* and its subclasses *node* and *property shape* thus adhering to ODM [4]. *Data shapes* are represented using a **rectangle** (Figure 3.3 (1)), and describe constraints applying on subjects and objects from the data graph. *Node shapes* describe constraints on individual focus nodes, while *property shapes* describe constraints for reachable nodes via a property path.

In UML "a class is drawn as a solid-outline rectangle with three compartments separated by horizontal lines" [15] which we redefine for *data shapes*. The **upper compartment** contains the *data shape*'s type and name (Figure 3.3 1). We determine the *data shape*'s type by reusing UML concepts similar to the UML profile for OWL and RDF [4], i.e. we define UML "stereotypes" to signify what the rectangles represent: *node shapes* declared as «NodeConditions», property shapes declared as «PropertyConditions» and (if the data shape type is not specified) data shapes as «Conditions». The name of the data shape is displayed as bold text to support the user in the identification and differentiation of data shapes. This name may be populated from `rdfs:label` values of the data shape, thus following best practices in labeling RDF concepts for humans. Both the middle and lower compartment list text-based key-value pairs, therefore we stay compliant to *UML*. Additionally, constraint language independent labels (Figures 3.1 and 3.2) are used to convey meaning and support users. The **middle compartment** lists information about the *data shape*'s identification and validation (Figure 3.3 7). Thus, *data shapes* are similar to UML where the middle compartment usually contains the *attributes* of *classes*, i.e. what characterizes them. The **lower compartment** contains actual *constraints* as a key-value list (Figure 3.3 3).

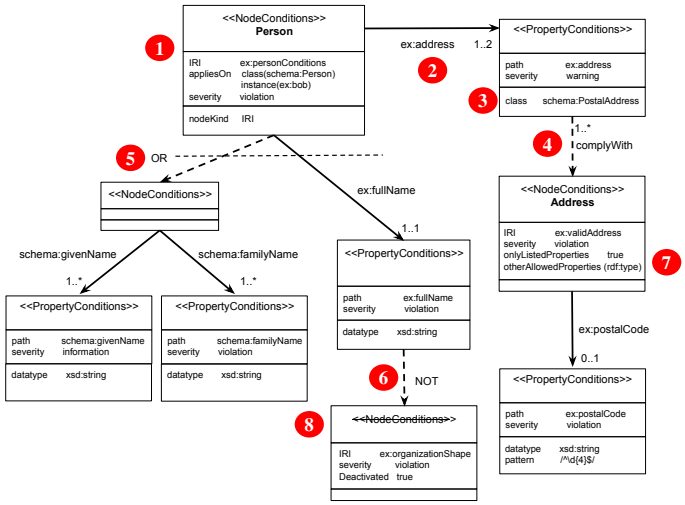


Figure 3.3: Constraints visualized using ShapeUML: A subject valid to the Person data shape should have an IRI (1), at least one but maximum two `ex:address` properties (2) of class `schema:PostalAddress` (3) and the object of at least one `ex:address` property should comply with the existing data shape `ex:validAddress` (4). Additionally, the subject valid to person should either have exactly one `ex:fullName` or at least one `schema:givenName` (5) and at least one `schema:familyName` all of datatype `xsd:string`. The value of `ex:fullName` must not comply with the data shape `ex:organizationShape` (6). Addresses must only have values for the property `postalAddress` with an exception for `rdf:type` (7). Constraints of the `ex:organizationShape` are not considered for validation (8).

3.3.1.2 Edge

We **reuse directed solid edges** from ODM/UML [4] to represent relationships, **reuse dashed edges overlaying individual edges** from UML [15] to represent one-to-many relationships, and **redefine directed dashed edges** for RDF constraint specifics.

Directed edges represent different relationships between data shapes and, thus, *ShapeUML* is able to represent relationships between different types of *data shapes*. Directed edges have a *label* at the **center of the edge** and possibly *cardinalities* next to the ends of the association (Figure 3.3 2). These edges associate a *data shape* with another *data shape* or set of *data shapes*.

We introduce **solid** and **dashed** directed edges to visually distinguish between different types of relationships. We indicate the edges from *node shapes* to *property shapes* as a **directed solid edge** (Figure 3.3 2) as it represents relationships between subjects and objects of the data graph. The *label* of such a connection is the property path of the connected *property shape* which supports readability as humans can read the label while processing the edge and

do not have to look for this label elsewhere in a rectangle; annotating an edge with a label also follows UML. A **dashed directed edge** with the label *complyWith* indicates that the source *data shape* needs to comply with the constraints of the destination *data shape* (Figure 3.3 5). Therefore such connections can be distinguished from property shape connections both via a visual difference and a different label. Similarly, a dashed directed edge with the label *NOT* indicates that the source *data shape* must not comply with the destination *data shape* (Figure 3.3 6). A **dashed line vertically** over individual edges with label next to the dashed line indicates one-to-many relationships between a *data shape* to a set of *data shapes*, following the UML specification [15] (Figure 3.3 5).

3.3.1.3 Text

We **reuse text** from UML to represent different concepts and **introduce striked through text** for data shape stereotypes to indicate a deactivated data shape.

Text represents *constraints* stated by a *data shape* and provides additional information where necessary. Text is added to the upper, middle and lower compartment of a *data shape* and as label on edges. The type of a data shape in the upper compartment can be struck through, showing that the *constraints* of this *data shape* are not used for validation, i.e. the data shape is deactivated (Figure 3.3 8). This visual aid aims in the quick identification of deactivated *data shapes* which does not introduce any visual symbol and thus does not deviate too much from the *UML* specification. Values referring to RDF terms can be shortened with a prefix, therefore the tool implementing the visual notation has to provide a prefix list.

3.3.1.4 Border

We **reuse solid borders** from UML, they are used for *data shapes*. According to the UML standard, stylistic details, such as line thicknesses, are not material to the specification. So, all *data shapes* are rendered using solid borders.

3.3.1.5 Position

We **reuse positions at the beginning and end of directed edges** from UML to represent cardinality-related constraint types. Within UML, *association ends* are among others specified by their cardinality.

In ShapeUML, cardinality constraints referring to properties are visualized next to the arrow head of a directed edge, i.e. *minCount* and *maxCount* (Figure 3.3 2); cardinality constraints referring to data shapes are visualized next to the source of a directed edge, i.e. *qualifiedMinCount* and *qualifiedMaxCount* (Figure 3.3 4). Thus, the visualization reflects the reading direction, for example: the person data shape requires the property `ex:address` at least 1 but maximum 2 times (Figure 3.3 2) vs a valid address property requires that at least 1 property value need to comply with the address node shape (Figure 3.3 4).

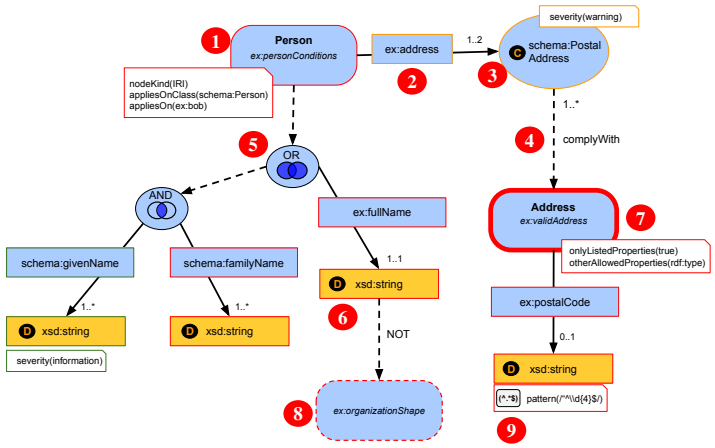


Figure 3.4: Constraints expressed using ShapeVOWL: A subject valid to the Person data shape should have an IRI (1), at least one but maximum two `ex:address` properties (2) of class `schema:PostalAddress` (3) and the object of at least one `ex:address` property should comply with the existing condition set `ex:validAddress` (4). Additionally, the subject valid to person should also either have exactly one `ex:fullName` or at least one `schema:givenName` (5) and at least one `schema:familyName` all of datatype `xsd:string`. The value of `ex:fullName` must not comply with the data shape `ex:organizationShape` (6). Addresses must only have values for the property `postalAddress` with an exception for `rdf:type` (7). Constraints of the `ex:organizationShape` are not considered for validation (8). ShapeVOWL also visualizes optional accompanying logos for constraint types (9).

3.3.1.6 Visual Example

The visual vocabulary of *ShapeUML* defined in the last section, can be used to represent SHACL shape graphs. We present and discuss an example (Figure 3.3).

ShapeUML defines visual elements for data shapes (Figure 3.3). *Data shapes* of different types («Conditions», «NodeConditions» and «PropertyConditions») can be uniquely identified with an IRI but can also have a human readable label. For example, a *node shape* uniquely identified (`ex:personConditions`, middle compartment) can have the human readable name *Person* (bold label in upper compartment) (Figure 3.3 1). Such a *node shape* can by default be applied on resources, e.g. `ex:bob`, or all instances of a class, e.g. `schema:Person`, both indicated by the key *appliesOn* in the middle compartment of a *ShapeUML data shape*.

Constraints are listed in the lower compartment of a *data shape* rectangle. A node could be constrained to be of a specific type using the *nodeKind* constraint. Similarly, constraints on property values are placed in the lower compartment of the corresponding «PropertyConditions» *property shape*. A fictive person node shape can represent the con-

straint that data valid to this *data shape* must have a unique identifier. And in the same fashion, the value of an `ex:address` property can be constrained to be of a specific class (Figure 3.3 ❸).

Cardinality constraints are represented using *text* and *position*. Therefore a constraint to express that a person must have at least *one* but maximum *two* addresses will be denoted with the (inclusive) cardinality specification `1..2` next to the arrow head of the directed edge which connects the *person node shape* with the *address property shape* (Figure 3.3 ❷).

Dashed directed edges can be used to indicate reuse of data shapes. To denote that the value of *at least one* of the aforementioned `ex:address` properties must comply with the `ex:validAddress` data shape, a dashed relationship with corresponding cardinalities `1..*` is drawn at the source *property shape* (Figure 3.3 ❹). In case every *address* should comply with the provided data shape, the qualified cardinalities at the source of the dashed arrow need to be removed. Such a removal would mean for a SHACL implementation that the two constraints `sh:qualifiedValueShape` and related `sh:qualifiedMinCount` are replaced by a single `sh:node` constraint. However, this is transparent in the visualization and users are not bothered with this specific terminology.

Data shapes can be connected with logical operators to build more complex constraints (Figure 3.3 ❺): subjects valid to the *Person node shape* should have either *exactly* one `ex:fullName` property, or at least one `schema:givenName` and at least one `schema:familyName`: dashed vertical OR edge overlaying individual edges.

3.3.2 ShapeVOWL

This visual notation is based on *VOWL* [1] and designed to be as close as possible to it. The graphical primitives of *ShapeVOWL* are shape, edge, text, border, position and color. The full specification is available online at `https://w3id.org/imec/unshacled/spec/shape-vowl/20211008` We describe the graphical primitives and elaborate with an example.

3.3.2.1 Shape

We **reuse blue ellipses and blue and yellow rectangles** from VOWL to represent subjects, predicates and objects of the data shape graph, **introduce white note-elements** to represent constraints and **introduce blue rectangles with rounded corners** to represent node shapes.

The graphical primitive *shape* distinguishes the fundamental constructs *node shapes*, *property shapes* and *constraints*, and represents one-to-many relationships. This follows *VOWL* where nodes in the graphs as well as specific restrictions such as *disjointness* are represented with dedicated nodes. *Node shapes*, subjects of triples, are represented as **blue rectangles with rounded corners** (Figure 3.4 ❶), *property shapes*, the predicate and object of a triple, as **rectangular label** on a directed edge (Figure 3.4 ❷) and either a **ellipse** or **rectangle** at the end of the edge representing the object (Figure 3.4 ❸, ❻), and *constraints* as **rectangle with the upper right corner bent** (note element) (Figure 3.4 ❶, ❾). Thus,

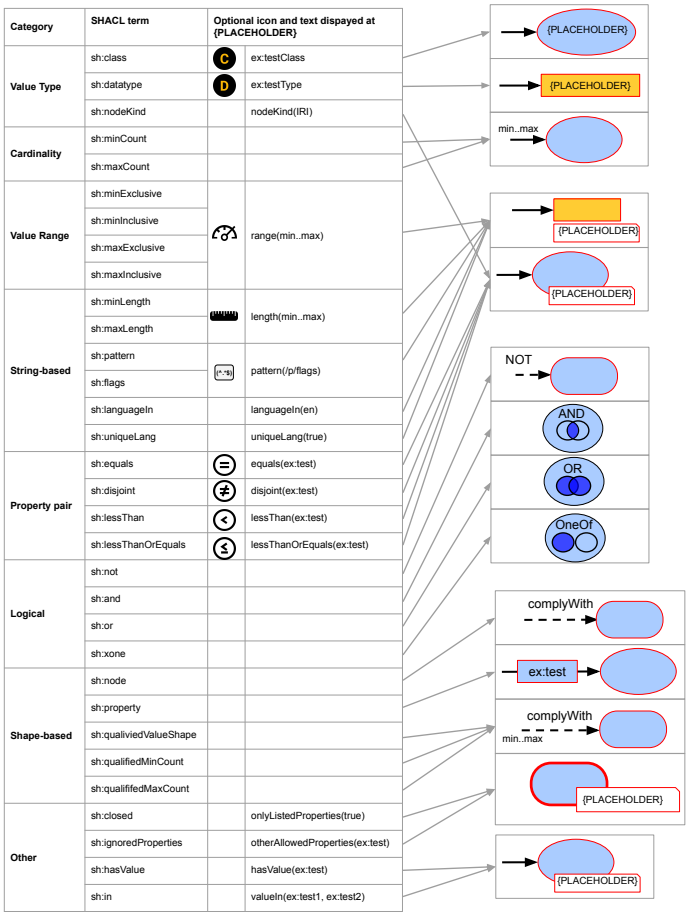


Figure 3.5: Correspondence between semantic constructs and *ShapeVOWL*: SHACL core constraints (left) and graphical notations (right).

node and property shapes align with VOWL as the *data shapes* appear like the RDF graph on which they define constraints on.

The note-element, containing constraints as text, is visually attached at the *node shape* or *property shape* indicating the constraints applying on the represented subjects, predicates or objects of a triple; constraints are visualized where they apply to facilitate the processing of the visualization by users. We also introduce ellipses as intermediate element to denote one-to-many relationships (see edges).

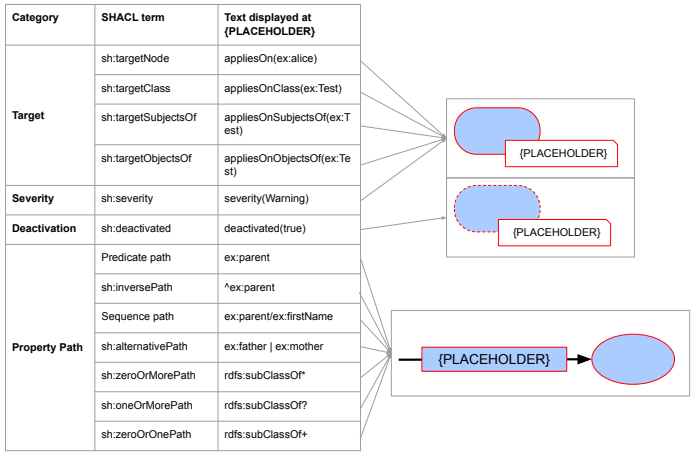


Figure 3.6: Correspondence between semantic constructs and *ShapeVOWL*: other relevant SHACL concepts besides core constraints (left) and graphical notations (right).

3.3.2.2 Edge

We **reuse directed solid edges with rectangular labels** from VOWL to represent properties and **redefine directed dashed edges** for RDF constraint specifics.

Edges represent relationships between *data shapes* which makes *ShapeVOWL* able to represent different kind of constraints in a visual fashion. **Directed dashed edges** (Figure 3.4 4) refer to relationships between *data shapes* and denote their label directly as text on top of the relationship. They indicate that the source *data shape* needs to comply with the constraints of the destination *data shape*.

Directed solid edges are part of a *property shape* and indicate their *label* in a **rectangle** above the edge (Figure 3.4 2), following *VOWL*. The *label* of directed solid edges is the property path of the represented property shape; relationships between *data shapes* are visually distinguished from *property shapes* due to the use of different edges.

Similar to VOWL, *cardinalities* are denoted next to the arrow head (Figure 3.4 3), but additionally *data shape* related qualified cardinalities are denoted at the start of a directed dashed edge (Figure 3.4 4). *Node and property shapes* may refer to multiple other *node and property shapes* in a **one-to-many relationship** to represent logical relationships. We represent such relationships using additional ellipses, representing the meaning of individual one-to-many relationship, i.e. conjunction and disjunction (Figure 3.4 5), similar to certain restrictions in *VOWL*, e.g., *disjointness*.

3.3.2.3 Text

We **reuse text** from VOWL to represent labels, **redefine datatype** to represent datatype constraints, **introduce text** to represent constraints and **italic text** to represent the unique identifier of data shapes.

We use **text** to represent constraints stated by *data shapes*, unique identifiers, and labels. Text is added in constraint *note elements*, *node shapes* and *property shapes*. Constraint note elements contain *constraints* in the form of text where the constraint's name is listed followed by its value in parentheses. This allows a consistent representation of different constraints without introducing a new visual variable for each of possibly more than 80 constraint types [26]. Values referring to RDF terms can be shortened with a prefix, therefore the tool implementing the visual notation has to provide a prefix list. *Data shapes* may have an optional human readable name which is denoted as bold text in the upper part of the *data shape* to facilitate the distinction of data shapes. This name may be populated from `rdfs:label` values, and, thus following best practices for labeling RDF concepts. Additionally, the unique identifier of *node shapes* is visualized as text in italics in the center of the rectangle with rounded corners representing the *node shape* (Figure 3.4 1). Users can also identify *node shapes* without a human readable label present. The *italic* type distinguishes the unique identifier from other text.

3.3.2.4 Border

All visual shapes have a border, we **reuse solid borders** from VOWL, **redefine dashed borders** to accommodate for validation-specific characteristics regarding deactivation and **introduce thick solid borders** to represent the constraint type *closed*.

VOWL uses dashed borders for specific OWL classes and literals without datatype. However, we use **dashed borders** to indicate which *data shapes* are not considered for validation (*deactivated*), because in contrast to an ontology visualization, we do not consider specific OWL classes but RDF constraints for validation, and our visualization of literals has a different meaning as we visualize constraints (Figure 3.4 9). For deactivated *node shapes* both the rectangle with rounded corner representing the *node shape* as well as a possibly attached note element with constraints will get a dashed border (Figure 3.4 8). Similarly, for deactivated *property shapes* the rectangle of the relationship label, the object and potentially attached note elements get a dashed border.

We introduce **thick solid borders** for *node shapes*, indicating that for validation only the explicitly linked properties are allowed (*closed data shape*, Figure 3.4 7).

The thick borders aim to represent the closeness whereas dashed borders aim to represent inactiveness. As the thickness and style of the edges are two different visual features, possible combinations of deactivated and closed data shapes can still be represented.

3.3.2.5 Position

We **reuse cardinality positions at directed edge endings** for property-based cardinality constraints, **introduce cardinalities at the beginning of a directed edge** to represent

data shape related cardinality constraints, **introduce positions for logical constraints** within dedicated nodes and **introduce positions for datatype and class constraints** within the objects of visualized triples.

We use specific **positions** for cardinality, datatype, class and logical constraints utilizing the graph visualization to support users in the parsing of information. In *ShapeVOWL*, cardinality constraints referring to properties are visualized next to the arrow head of a directed edge, i.e. *minCount* and *maxCount*; cardinality constraints referring to data shapes are visualized next to the source of a directed edge, i.e. *qualifiedMinCount* and *qualifiedMaxCount* (Figure 3.4 4). The visualization reflects the reading direction, for example: the person data shape requires the property `ex:address` at least 1 but maximum 2 times (Figure 3.4 2) vs a valid address property requires at least 1 property value to comply with the address node shape (Figure 3.4 4).

Datatype and class constraints are not visualized in a note element, but directly as text in the graphical element representing the object, i.e. a yellow rectangle for datatype constraints (Figure 3.4 6) or a blue ellipse for class constraints (Figure 3.4 3). VOWL visualizes datatypes as text within the yellow rectangle representing a literal. We reuse this visualization to denote a datatype constraint of a property value and add an additional datatype icon in front of the name of the datatype to indicate that a constraint exists (Figure 3.4 6). This icon is an orange D in a black circle (Figure 3.4). Consistently with datatypes, class constraints are denoted as text within the ellipse representing the property value. Class constraints have an additional class icon in front of the name of the class. This icon is an orange C in a black circle (Figure 3.4 3).

Logical constraints are not represented in a note element, but as dedicated nodes or as labels on dashed edges which enables *ShapeVOWL* to represent relationships between different types of *data shapes*. Conjunction and (exclusive) disjunction constraints are visualized as ellipse with respective labels on the upper part of the ellipse (Figure 3.4 5). Additionally, icons representing Venn diagrams are used to distinguish the different logical constraint types. These icons represent Venn diagrams, similar to certain VOWL constructs. Negation constraints are represented as text label "NOT" on top of dashed edges connecting data shapes (Figure 3.4 6).

3.3.2.6 Color scheme

We **reuse the VOWL base color** to represent subjects, predicates and objects of the data shape graph, **reuse the VOWL literal color** to represent literals and **introduce border colors** for data shapes' severity.

A color scheme is applied on the border color of *data shapes* and *note elements* to express different severities (Figure 3.4 1). VOWL uses a color scheme for a better distinction of the different elements [1]. We reuse the base color and literal color of VOWL.

Additionally, for *ShapeVOWL* colors on borders are used to express the severity of *data shapes*. For the severities *violation, warning and information* from the SHACL specification we recommend the respective colors **red, yellow and green**. Green is chosen instead of blue so the severity colors for *data shapes* are not confused with the VOWL *general color*.

3.3.2.7 Visual Example

The visual vocabulary of *ShapeVOWL* defined in the last section, can be used to represent SHACL shape graphs. We present and discuss an example (Figure 3.4).

ShapeVOWL defines visual elements for data shapes (Figure 3.4). Our color scheme is applied; *data shapes* are colored with respect to their severity.

Node shapes can be uniquely identified with an IRI but can also have a human readable label. For example, a *node shape* uniquely identified with the IRI `ex:personConditions` (center of rectangle with rounded corners representing a subject node) can have the human readable name *Person* (bold label in upper part of the rectangle with rounded corners) (Figure 3.4 ❶). Such a *node shape* can by default be applied on resources, e.g. `ex:bob` or all instances of a class such as `schema:Person`, both is is indicated by the *appliesOn* annotation in the attached white note-element of a *ShapeVOWL data shape*.

Constraints have a special position or are listed in white note-elements attached to a *data shape*; depending on the rendering either overlapping an ellipse/rectangle with rounded corners (Figure 3.4 ❶, ❸) or next to a rectangle (Figure 3.4 ❾). A fictive person node shape can represent the constraint that persons must have a unique identifier (Figure 3.4 ❶, nodeKind constraint). The value of an `ex:address` property can be constrained to be of a specific class whereas value type constraints are listed within the shape representing the object together with an icon (Figure 3.4 ❸). Cardinality constraints are represented using *text* and *position*. Thus, a constraint to express that a person must have at least *one* but maximum *two* addresses will be denoted with the (inclusive) cardinality specification `1..2` next to the arrow head of the directed edge which connects the *person node shape* with the *address property shape* (Figure 3.4 ❷).

Dashed directed edges with the label *complyWith* indicate **reuse of data shapes**. To denote the constraint that the value of *at least one* of the aforementioned `ex:address` properties must comply with the `ex:validAddress` data shape, a dashed relationship with corresponding cardinalities `1..*` is drawn at the source *property shape* (Figure 3.4 ❹). In case every *address* should comply with the provided data shape, the qualified cardinalities at the source of the dashed arrow have to be removed.

Data shapes can be connected with logical operators to build more complex constraints (Figure 3.4 ❺): subjects valid to the *Person node shape* should have either *exactly* one `ex:fullName` property, or at least one `schema:givenName` and at least one `schema:familyName`: disjunction node with label "OR" and Venn diagram icon. The logical operator negation only takes one data shape as argument and not a whole data shape list, therefore it is visualized with the label NOT on a dashed connection (Figure 3.4 ❻).

With respect to validation **data shapes may be closed or deactivated**. The `ex:validAddress` data shape is closed, visually indicated by a thick border: valid addresses are only allowed to have the property `postalCode` and an exception is made for `rdf:type` denoting the class (Figure 3.4 ❼). The data shape `ex:organizationShape` is deactivated, visually indicated by dashed border: its constraints are not considered during validation (Figure 3.4 ❽). Constraint types can be accompanied with a logo displayed before the constraint in the note element (Figure 3.4 ❾).

3.4 Comparative Analysis

Both *ShapeUML* and *ShapeVOWL* were designed by following basic principles of cognitive effectiveness [10], however, as we reused the existing notations *UML* and *VOWL* these principles could only be applied to a certain extent. Therefore, we analyze *ShapeUML* and *ShapeVOWL* with respect to these design principles with the aim of scientifically argue about the impact of design decisions on human information processing and thus the effectiveness of *ShapeUML* and *ShapeVOWL* from a theoretical perspective.

We refer to each principle's definition according to Moody [10] (which includes other frameworks as specified in Section 3.2.5) and discuss to which extent each visual notation complies. We omit the design principle *cognitive integration* as it only applies when multiple diagrams of different types are integrated. Table 3.1 summarizes the comparison which is discussed in Section 3.4.9.

3.4.1 Semiotic Clarity

Semiotic clarity relates to the correspondence between symbols and their referent concepts [10], there must be a one-to-one correspondence for a visual notation to satisfy the requirements of a notational system [10, 20]. If there is no one-to-one correspondence between semantic constructs and visual symbols, one of the following four anomalies can occur: symbol redundancy, symbol overload, symbol excess or symbol deficit [10]. In case of *symbol redundancy*, a semantic construct is represented by multiple graphical symbols; the opposite is *symbol overload*. *Symbol excess* occurs if graphical symbols do not correspond to any semantic construct; and the opposite is *symbol deficit*, a semantic construct with no graphical symbol.

ShapeUML All semantic constructs are represented in the visual notation (Figures 3.1 and 3.2), i.e. terms from the SHACL specification; some constructs use the same graphical symbol but text is used to differentiate, and, thus, to maintain visual expressiveness. Following the ODM-profile of UML, *ShapeUML* uses rectangles with solid borders to represent *data shapes*, thus *node* and *property shapes* share the same graphical symbol (**symbol overload**). However, *node* and *property shapes* are distinguished by additional text indicating the type. **Symbol deficit** was deliberately introduced to reduce graphic complexity: more than 30 constraint types are supported, but they are all represented as text, only *logical constraint types* and *cardinality constraints* use additional visual variables (*edges* and *position*). *ShapeUML* does not visualize any semantic construct with multiple graphical symbols (*symbol redundancy*) nor does it contain any graphical symbol which does not correspond to a semantic construct (*symbol excess*), thus semiotic clarity is achieved.

ShapeVOWL All semantic constructs are represented in the visual notation (Figures 3.5 and 3.6) and similar to ShapeUML, **symbol deficit** is deliberately introduced to increase visual expressiveness. Multiple graphical symbols are used in ShapeVOWL. Blue rectangles with rounded corners represent *node shapes* (subject of triples), blue rectangles over solid

arrows represent the property part of *property shapes* (predicate of triples), and blue ellipses or yellow rectangles represent the object part of *property shapes* (object of triples). Certain constraint types are represented using the visual variables *border*, *edge* and *position* but to reduce graphic complexity most of the 31 constraint types are represented textually within *note-elements*. However, constraint types may also be accompanied by an icon which we provide for commonly used constraint types [27] (see Figure 3.1) not visualized using other visual variables such as *position* (see next section). Similar to *ShapeUML*, *ShapeVOWL* achieves *semiotic clarity* as no *symbol redundancy* nor *symbol excess* are present.

3.4.2 Perceptual Discriminability

Perceptual discriminability describes the ease and accuracy with which graphical symbols can be differentiated from each other [10]. A factor is the *visual distance*, i.e. the number of visual variables on which the symbols differ and the size of differences in perceptible steps (capacity). Shapes are the *primary basis* for humans to identify objects in the real world, while *textual differentiation* is a cognitively ineffective way to handle graphic complexity [10]. This principle includes *perceptual popout*, i.e. preattentively detection of visual elements [10, 17]

ShapeUML *ShapeUML* uses the visual variables shape, edge, text, border, and position. But because *shape* is always a rectangle and *border* is always solid, both are not variable anymore and the perceptual discriminability of *ShapeUML* is low. However, therefore we stay close to the UML specification, where users potentially are familiar with. Given the limited number of graphical symbols, i.e. rectangles with solid borders for *data shapes*, text for *constraints* as well as solid and dashed edges to relate *data shapes*, *ShapeUML* only provides **limited visual distance**.

ShapeVOWL *ShapeVOWL* uses the visual variables shape, edge, text, border, position, and color. On the one hand, *ShapeVOWL* uses **one visual variable more than *ShapeUML***; and on the other hand, *ShapeVOWL* uses different shapes and borders, i.e. in constrast to *ShapeUML* these concepts are variable in *ShapeVOWL*. Nodes and properties are clearly distinguished by the visual variable *shape* and *color*, i.e. the VOWL base-color *blue* is used for nodes and property labels and the VOWL color *yellow* is used for literals. Additionally, the **visual distance between symbols is increased** because *ShapeVOWL* defines optional icons for different constraint types.

3.4.3 Semantic Transparency

Semantic transparency is the extent to which a notation's meaning can be inferred from its appearance, informally its "*intuitiveness*" or the degree of how much the appearance provides a cue to its meaning [10]. This principle is not measured binary, semantic transparency can appear in a continuum from *semantically immediate* where a novice can infer the meaning

(e.g. a stick figure to represent a person), to *semantic perversity* where even a wrong meaning is inferred [10].

ShapeUML *ShapeUML* is based on UML which uses abstract shapes, and, thus it does **not provide much semantic transparency**. The boxes representing *data shapes* do not provide a cue to their meaning. However, presenting the property path as a label on edges connecting *node* with *property shapes* may resemble the underlying graph structure of RDF and could minimally provide *semantic transparency*.

ShapeVOWL *ShapeVOWL* uses a graph visualization based on nodes and edges of the actual RDF graph for which it defines the *constraints*. Several indicators suggest that *ShapeVOWL* has a **higher semantic transparency** compared to *ShapeUML*. Previously defined VOWL-based visual notations already demonstrated that users find the graph visualization **intuitive** [1]. *ShapeVOWL* also reuses visual metaphors such as Venn diagrams for logical constraints, which, according to Moody, **increases semantic transparency**. *ShapeVOWL* attaches constraints visually to where they apply to which further increases semantic transparency; certain property shape constraints apply on the property, such as cardinalities, and others on the value of the property, such as *minimum inclusive value constraints*. If not visually separated, *min/max cardinality constraints* on the property and *min/max constraints* on the value might be confused.

To further increase *semantic transparency*, *ShapeVOWL* defines optional icons for constraint types which can speed up recognition and recall as well as improve understanding for novice users [10]. However, according to a recent meta study [56], semantic transparency is not increased with the use of icons per se, empirical tests need to be performed to diminish cultural associations. To this end ShapeVOWL mostly relies on icons representing arithmetic operators such as an equal-sign or less-than. Additionally, icons are only optional and future studies may provide more insights in appropriate icons for RDF constraints.

3.4.4 Complexity Management

Complexity management aims not to overload the human mind. For instance, visual representations often do not scale well [10]. *Modularization* and *hierarchy* offer solutions to manage complexity.

Both proposed visual notations **do not yet account for modularization or hierarchy**. However, tools implementing visual notations can account for this and e.g. offer zoom functionality [2]. Currently our tool *UnSHACLed* provides geometric zooming (Section 3.5).

3.4.5 Visual Expressiveness

Visual expressiveness refers to the number of visual variables in the whole notation. Each variable has a power denoting the information which can be used [10].

The visual expressiveness of both visual notations **is not very high** considering that most *constraints types*, one of the fundamental constructs are represented as text only (with

the exception of logical relationships in both notations). However, on the one hand this is because both notations were built with the **objective to reuse existing notations** already familiar to users, thus inheritance of *visual expressiveness*, and on the other hand we tried to **keep the graph complexity low** by deliberately not representing each constraint type with different visual variables.

If required by specific use cases, both notations can be improved specifically towards *visual expressiveness*. For example, **ShapeVOWL has higher expressiveness** due to the use of more visual variables compared to *ShapeUML*, in a similar fashion more visual variables can be used for both notations.

3.4.6 Dual Coding

Dual coding is the use of text to complement graphics. Text on its own is cognitively ineffective to encode information, but, in a supplementary fashion, it can reinforce and clarify meaning [10]. However, although *textual annotations* improve understanding, having a dedicated graphical symbol only for annotations not representing any semantic construct of the language it harms semiotic clarity, i.e. a case of *symbol excess* as the graphical symbol of annotation does not represent a semantic construct [10].

ShapeUML is based on UML, heavily text-based and thus **has limited dual coding**. Text is mostly used to denote constraints, but also for labels and unique identifiers. The *deactivation* of *data shapes* may be considered *dual coded* because, in addition to the textual declaration, the type of the *data shape* in the upper compartment is struck through, i.e. an additional visual change of font. *Node shapes* may refer to *property shapes* which in *ShapeUML* is encoded using a directed solid edge.

Following UML, *logical constraints* are represented with specific edges additionally labeled with the logical constraint's name. However, this is not considered *dual coded* as without label, edges of different *logical constraints* are not distinguishable. Both visual variable and text are needed to denote logical constraints.

ShapeVOWL visualizes graphs, and text is added to graph elements. **Several elements are dual coded** in *ShapeVOWL*. Similar to *ShapeUML*, text is mostly used to denote constraints, but also for labels and unique identifiers. All *constraints* are represented textually in a *note-element*, but some constraint types are also represented using additional icons or the visual variables *border*, *edge* and *color*. *ShapeVOWL* defines optional *icons* for constraint types, e.g. for class, datatype or literal pattern constraints. Together with the visual variable *color* and *border*, text also denotes the severity of data shapes. Dashed and thick solid borders, in addition to text, are used to indicate characteristics relevant to validation of the RDF constraints, the constraint type *closed* and deactivation of *data shapes*.

3.4.7 Graphic Economy

Graphic economy states that the size of the visual vocabulary should be cognitively manageable to achieve a low graphical complexity [10]. The *number of semantic constructs* can be limited, symbol deficit can be introduced or the visual expressiveness can be increased.

Both visual notations should be **cognitively manageable**. SHACL supports a subset of possibly more than eighty constraint types, thus the number of semantic constructs is already limited (all concepts listed in Figures 3.1, 3.2, 3.5 and 3.6). Additionally, symbol deficit is deliberately introduced by the design decision of not visualizing each constraint type of the SHACL core using separate visual variables. An unlimited number of symbols can be created by combining visual variables, however, this does not scale due to cognitive limits where humans must remember the meaning of the symbol [10]. Both *ShapeUML* and *ShapeVOWL* have a small visual vocabulary as both use less than five graphical primitives.

3.4.8 Cognitive Fit

Cognitive fit means different representations are suitable for different tasks and audiences [10]. Optimizing visual notations for novice users can reduce effectiveness for experts and vice versa. More, the medium on which a visual notation is presented influences the effectiveness, i.e. manual drawing with pen and paper vs computer display. Icons, color, and texture are more difficult to draw than simple geometric shapes [10].

ShapeUML *ShapeUML* is based on UML, and, thus **is suited for users already familiar with UML**. It also consists only of rectangles, edges and text which facilitates manual drawing. *ShapeUML* uses a small number of visual variables and encodes a lot as text. For novice users it may be difficult to understand *ShapeUML* but optimizing it for novice users might introduce large deviations from UML which would make it harder for experts to understand.

ShapeVOWL *ShapeVOWL* uses a graph visualization with nodes and edges. Experiments with other VOWL-based notations already suggest that **VOWL is intuitive** also for people with less knowledge about the underlying languages [1]. Additionally, semantic web experts are usually already familiar with different VOWL-based notations and the graph model in general; *ShapeVOWL* leverages this and **may provide a trade-off between understanding for experts and novices**. *ShapeVOWL* relies on simple geometric shapes and text, colors are optional, thus, with respect to perceptual discriminability, semantic transparency and visual expressiveness, *ShapeVOWL* can also be drawn by hand without effort (neglecting certain dual coding like more complicated icons).

3.4.9 Discussion

We analyzed both visual notations with respect to Moody's design principles, which itself is based on seminal works of human cognition such as communication theory [16], feature

integration theory [17], Bertin's work on Semiology of Graphics [19], or Goodman's theory of symbols [20]; and in the following discuss our findings which are summarized in Table 3.1.

One the one hand, *ShapeVOWL* uses more visual variables and symbols to express semantic constructs than *ShapeUML*. For example, it uses more *shapes*, meaning of *borders* but also *colors* and optionally *icons*. This – in addition to the depiction of the underlying RDF graph data, specific edges to connect elements, and Venn diagrams – results in high scores for *semiotic clarity* and *semantic transparency*. All other principles are at least partially addressed with the exception of *complexity management* which can be accomplished by a tool implementing *ShapeVOWL*, e.g. by providing different means of zooming. However, more research regarding appropriate icons is needed, following a recent meta-study on semantic transparency [56].

On the other hand, *ShapeUML* shows *semiotic clarity* and *graphic economy* with an advanced *cognitive fit*. This means that *ShapeUML* represents all RDF constraints' needed concepts in a cognitively manageable fashion and, additionally, may be suited for specific tasks and audiences. *Perceptual discriminability*, *semantic transparency* and *visual expressiveness* are affected by *cognitive fit* [10], thus, considering hand-drawn representations of *ShapeUML*, its simplicity may become an advantage as no special drawing abilities are needed.

Principle	**ShapeUML**	**ShapeVOWL**
Semiotic Clarity	+	++
Perceptual Discriminability	-	+
Semantic Transparency	-	++
Complexity Management	-	-
Visual expressiveness	-	+
Dual Coding	-	+
Graphic Economy	+	+
Cognitive Fit	++	+

Table 3.1: A comparative analysis with Moody's design principles [10] for cognitive effective visual notations reveals that ShapeVOWL scores better compared to ShapeUML. A double plus (++) indicates that each dimension of the principle is addressed, a single plus (+) that at least one dimension is addressed respectively not violated and a minus (-) indicates that a principle is not or very poorly addressed.

3.5 UnSHACLed editor

UnSHACLed is a graphical editor for RDF constraints. It allows users to validate RDF data against RDF constraints and view a validation report by loading existing RDF data into the tool and validate them with separately loaded or visually created RDF constraints. The main goal of *UnSHACLed* is to enable users familiar with RDF but not familiar with specific RDF constraint languages to create and edit RDF constraints. *UnSHACLed* offers a web interface and thus can be used with any browser. An early prototype was presented in

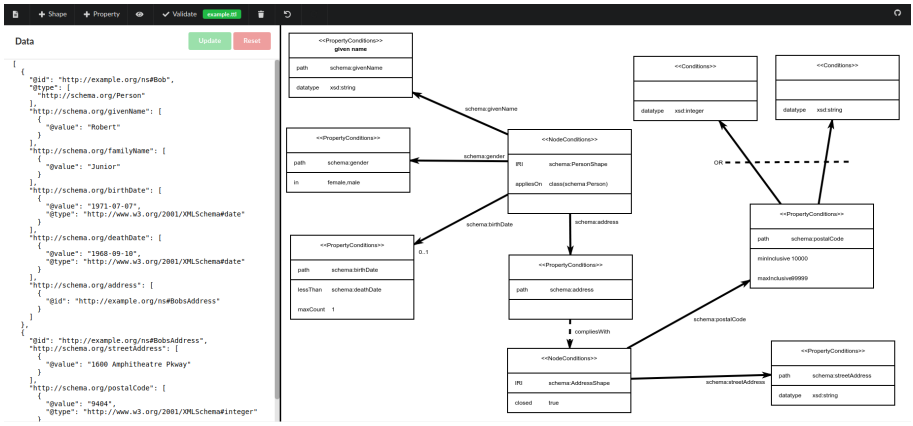


Figure 3.7: The user interface of our tool UnSHACLed consisting of several panels supporting different editing approaches.

previous work [11] and is available on GitHub[15]. In this paper we present a recently reworked version: `https://github.com/KNowledgeOnWebScale/unshacled`.

In this section we discuss features for an RDF constraint editor (Section 3.5.1) and how visual notations contribute to it, as well as introducing the implementation of our RDF editor *UnSHACLed* (Section 3.5.2).

3.5.1 Features for Data Shape Editing

In previous work [11] we introduced seven desired features for the editing of *data shapes*.

F1: Independence of constraint language Data shape editors should not confront domain experts with writing the textual syntax of a specific constraint language. Moreover, the visualization of the constraints should be independent of the underlying constraint language: generic (graphical) symbols can be used to (partially) hide language-specific textual syntax, as constraint languages have overlapping semantic constructs. Both *ShapeUML* and *ShapeVOWL* are constraint language independent and have a defined *visual vocabulary* covering semantic constructs of RDF constraints.

F2: Support multiple data sources Data shape editors should support domain experts in defining data shapes referring to multiple data sources at once. The proposed visual notations allow to define RDF constraints in a visual fashion for different data sources.

[15] Jonathan Van der Cruysse et al., "UnSHACLed", `https://web.archive.org/web/20200911095832/https://github.com/dubious-developments/UnSHACLed` (archived website accessed February 12, 2022)

F3: Support different serializations Data shape editors should not restrict domain experts to specific serializations of the data source nor the constraint language. A data graph can be serialized in different ways without changing the actual data or structure (e.g. RDF/XML vs Turtle). The visual vocabulary of both *ShapeUML* and *ShapeVOWL* covers semantic constructs of RDF constraints and is currently mapped to SHACL. Thus it is represented in RDF which can be serialized to different serializations.

F4: Support multiple ontologies Data shape editors should support domain experts in defining data shapes for data graphs annotated with multiple ontologies simultaneously. Both notations use URIs where necessary, e.g. for property paths or class constraints. Thus, multiple ontologies are supported by both notations.

F5: Multiple alternative modeling approaches Data shape editors should enable and support multiple alternative modeling approaches and allow domain experts to choose the most adequate one for their needs. Two modeling approaches, complementary to visual notations, were discussed in our previous work [11].

F6: Non-linear workflows Data shape editors should allow domain experts to keep an overview of the relationship between the data graph and data shapes, by providing non-linear editing. Although the data graph is not visualized together with the shapes graph, the data is visualized in the data panel. Terms related to data shapes' assignment to instance data is covered by the visual notations, i.e. the *appliesOn* concept indicating on which data shown constraints apply by default.

F7: Independence of execution Data shape editors should allow importing and exporting the data shapes specified by the domain experts, as a use case may require to execute the data shapes elsewhere. Both *ShapeUML* and *ShapeVOWL* are currently mapped to SHACL and, thus, to RDF which provides interoperability and allows the import and export of data shapes.

3.5.2 Implementation

We describe the modular architecture of our RDF constraint editor *UnSHACLed* (Section 3.5.2.1), and relevant GUI components providing user interactions in a visual fashion (Section 3.5.2.2).

3.5.2.1 Architecture

UnSHACLed is a web-based RDF constraint editor independent from specific data formats, visual notations or validation engines.

Framework *UnSHACLed* is implemented with the web framework *Vue.js* following the *model-view-viewmodel* (MVVM) design pattern introduced by John Gossman in 2005[16].

It therefore can run in modern Browsers and no additional server infrastructure such as databases are required.

Intermediate format *UnSHACLed* uses the *state management pattern and library Vuex* to store RDF constraints using an intermediate data format which allows all application components to access the RDF constraints in a controlled manner. Therefore other constraint languages can be supported by providing a mapping to the intermediate format without the need to change other parts of the implementation.

Visual notations *UnSHACLed* uses the *VueKonva* library to draw canvas graphics. Several components for both *ShapeUML* and *ShapeVOWL* were developed to render the two notations. New visual notations can be added in the form of new components which also read and write data to the intermediate format of Vuex store.

Validation For validation the intermediate format is transformed to a representation of a concrete RDF constraint language (currently supported is SHACL) and is passed together with the data to a separate *validation engine*. Another constraint language and validation engine can be used which only leads to adjustments in *UnSHACLed* with respect to transformations of the intermediate representation or invocation of another validation engine, no adjustments to the GUI are required.

3.5.2.2 Graphical User Interface

In this section we discuss the graphical user interface of *UnSHACLed*, namely the different existing panels and interactions elements with which users can interact using visual notations.

Panels The GUI consists of three panels representing different parts of a Linked Data validation workflow: a data panel, modeling panel and validation result panel.

The **Data panel** shows data which should be constrained or described (left panel in Figure 3.7). RDF is currently supported in different serializations, such as *turtle* and *JSON-LD*. This is raw data and can also be edited. *UnSHACLed* is modular and the functionality can be extended to also visualize data of other kind to support other *editing approaches*.

The **Modeling panel** shows RDF constraints in the visual notation chosen in the menu, both *ShapeUML* and *ShapeVOWL* are supported (right panel in Figure 3.7). Elements in the modeling panel are denoted visually and scalability is addressed with geometric zooming.

The **Validation result panel** shows the validation result of applying the *RDF constraints* of the *modeling panel* on the data of the *data panel* as reported by a *validation engine for*

[16] John Gossman, "Introduction to Model/View/ViewModel pattern for building WPF apps", `https://web.archive.org/web/20051029151624/http://blogs.msdn.com:80/johngossman/archive/2005/10/08/478683.aspx`(archived website accessed February 12, 2022)

RDF constraints. The validation result panel is implemented as a modal dialog, i.e. it appears after clicking the *validation button*. *UnSHACLed* is independent of concrete *RDF constraint languages*, it can be extended with different validation engines.

Interactions Visual notations specify how RDF constraints are visualized, but *UnSHACLed* also allows to interact with the visualizations. Most notably nodes in the graph can be dragged and dropped inside the *modeling panel*. When hovering over an element a red and a green button appear representing actions for delete and editing. In the latter case a modal dialog opens in which users can change or add constraints. Thus, users can also edit RDF constraints using the visual notations and do not have to learn a specific textual syntax.

3.6 User Evaluation

We conducted a comparative study to validate our main hypothesis that *users familiar with Linked data can answer questions about visually represented RDF constraints more effective with ShapeVOWL than with ShapeUML*. We compared how accurately users can answer questions about data shapes represented using either *ShapeUML* or *ShapeVOWL*. In Section 3.6.1, we describe the questionnaires to cover various aspects of the data shape domain based on the SHACL core specification. In Section 3.6.2, we elaborate on the experiment, in Section 3.6.3, we discuss potential threats to validity, in Section 3.6.4, we analyze the results of quantitative questions, and in Section 3.6.5, we analyze results of qualitative questions. Collected (anonymized) data, the questionnaire and user introductions as well as code for the quantitative and qualitative analysis are openly available at `https://doi.org/10.6084/m9.figshare.13614440.v2`.

3.6.1 Questionnaires

We created two RDF constraints-related questionnaires, the first containing questions related to RDF constraint concepts based on the SHACL specification which was used in an initial user study, and a second follow-up questionnaire covering more diverse visualization tasks and specific questions informed by the findings of the initial user study. These questionnaires are available at our online resource `https://doi.org/10.6084/m9.figshare.13614440.v2`

3.6.1.1 Constraint Concepts questionnaire

We derived questions from the SHACL specification relevant to RDF constraints and validation, which were used in a user study to validate our hypothesis.

We created questions to test (i) at least one constraint type per core constraint category of the SHACL specification, and (ii) other RDF constraint concepts relevant for validation, i.e. the targeting mechanism, property paths, severity and deactivation. The SHACL specification lists eight core constraint categories:

1. value type, 1 constraint
2. cardinality, 1 constraint
3. value range, 1 constraint
4. string-based, 1 constraint
5. property pair, 1 constraint
6. logical, 1 constraint
7. shape-based, 2 constraints
8. *other*, 2 constraints

We selected at least one constraint type for each category and created an associated question, for example "*How many datatype constraints can you see?*" for the constraint type *datatype* of *value type* category. The last two categories mix several relevant constraint types, so, we selected 2 constraints types for each.

Additionally we created one question for each of the aforementioned other relevant concepts, such as "*How many property conditions with the severity 'information' can you see?*" for the concept *severity* or "*How many zero-or-more property paths can you see?*" for the concept *property paths*.

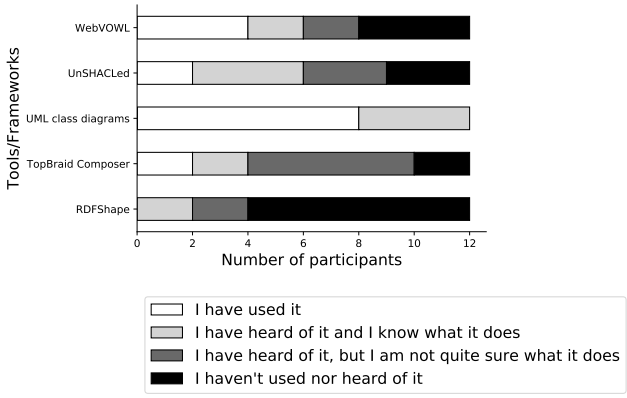


Figure 3.8: UML diagrams known by all participants and already used by the majority, other tools/frameworks less commonly known.

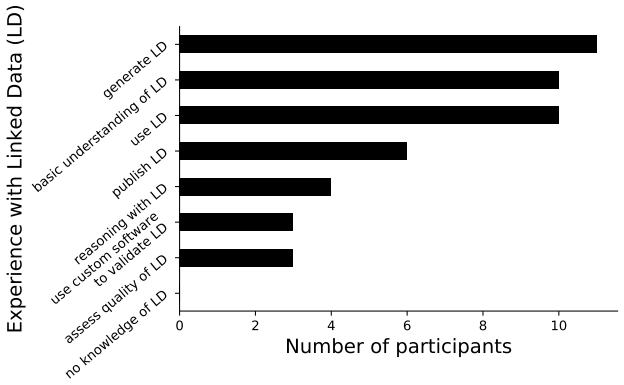


Figure 3.9: Answers based on self assessment: all participants are familiar with Linked Data, most participants generate or use Linked Data (all options were presented using multiple choice checkboxes).

3.6.1.2 Follow-up questionnaire

We created six questions to cover more diverse visualization tasks compared to the initial user study questionnaire and one question to test participants' understanding of *property paths* for which we observed the highest error rate in the initial user study.

The questions cover the following six visualization tasks from ten Amar et al. [52] tasks, which we have chosen based on a taxonomy alignment from Brehmer and Munzner [49], see also Figure 3.10 and the *Task* description paragraph in the next section.

- Find extremum
- Determine range
- Retrieve value
- Order
- Compute derived value
- Filter

To keep the follow-up questionnaire short, we have chosen to select maximum one task per taxonomy leaf node. Therefore, no question was asked for the tasks *Find anomalies*, *Find clusters*, *Find correlation* and *Characterize distribution* as the first three belong to the same taxonomy leaf node *explore* such as *Find extremum*, and the last belongs to the same taxonomy leaf node *identify* such as *Determine range*.

Additionally we asked the question "Do you see any 'property path' which is not just a single property?" because in the initial user study participants identified *property paths* in test cases when in fact no were shown. In case they answered yes, we also asked the question "Which is the property path and where do you see it, please elaborate" to obtain more information.

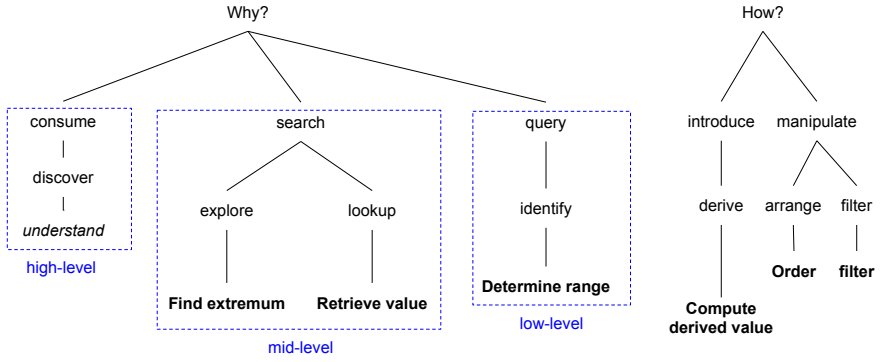


Figure 3.10: Selected tasks from Amar et al. [52] (bold) arranged in the typology of Brehmer and Munzner [49] according to their multi-level alignment. The user study had the high-level goal to discover, concretely to understand as defined by Pike et al. [57]. This involves mid-level search tasks of explore and lookup, low-level tasks of identifying, as well as cognitive interaction methods (introduce and manipulate). From 10 tasks introduced by Amar et al. only 6 were chosen to keep the follow-up study short, i.e. maximum 1 task per typology leaf node in case there were multiple.

3.6.2 Method

The user study follows a *within-subject design* (also referred to as *within-group* or *repeated measures* [58]) in which all participants are confronted with examples of both visual notations *ShapeUML* and *ShapeVOWL*. However, to mitigate learning effects, we decided not to show the same examples twice to a single participant (see threats to validity in Section 3.6.3). We discuss the method of the user study by explaining the procedure, elaborating on recruited participants, and introduce the example test cases.

Procedure Potential participants with Linked Data knowledge were directly contacted by the authors. Those who participated were assigned in a round-robin fashion to one of two groups (groups A and B) to mitigate order effects (see threats to validity Section 3.6.3), and had to (i) read introductions to both *ShapeUML* and *ShapeVOWL* (presented in this order), and (ii) complete an online questionnaire. The initial user study is divided into three steps:

a pre-assessment, a session in which the questionnaire is answered, and a post-assessment. Additionally, a smaller follow-up user study with only one questionnaire was performed in a later stage where the first author contacted previous participants again.

(i) The **pre-assessment** is focused on the participants' sociodemographic traits, such as year of birth, gender, and level of education, to provide indicators of the studied population. Additionally, through a self-assessment, the participants' expertise with Linked Data and with RDF constraints is assessed as well as their familiarity with the topic and tools.

(ii) The **main questionnaire** consists of 11 questions about data shapes presented using *ShapeUML* and *ShapeVOWL* to assess how effective visualized elements are recognized. After that, 15 questions on 4 test cases were asked to compare visualizations in *ShapeUML* and *ShapeVOWL*. These questions include 14 questions derived from the SHACL specification (Section 3.6.1.1) and one open question to provide feedback about the shown examples and asked questions. For group A the general example is first shown in *ShapeUML* afterwards in *ShapeVOWL* and then the test cases are presented started with the first test case in *ShapeUML*, the second in *ShapeVOWL* and so forth; it is the other way around for group B to mitigate order effects (see validity threats in Section 3.6.3).

(iii) The **post-assessment** consists of 4 questions and collects information about the participants' preference for either ShapeUML or ShapeVOWL to answer questions about data shapes, whether they want to use one of the notations also for the editing of data shapes, besides only to visualize them; and general feedback.

Tasks Questions of the main questionnaire and the follow-up study represent different visualization tasks which are well studied in visualization task taxonomies and typologies (see Section 3.2.6). Our questions cover tasks from Amar et al. [52] and have the high level goal to discover [49] and more concretely to understand as defined by Pike et al. [57, 49], depicted in Figure 3.10. Each question of the main questionnaire is a combination of *Retrieve Value* and *Compute derived value* tasks. Follow-up study tasks cover 6 out of 10 tasks from Amar et al. [52], maximum 1 task per typology leaf node in case there were multiple. Therefore we still cover all leaf nodes from the alignment between Amar et al. [52] and the multi-level topology from Brehmer and Munzner [49].

Participants The online questionnaire was sent to 14 potential participants in September 2020. 12 participants took part in the experiment, their age range was 23 to 40. All participants were highly educated: all have at least a master degree, one a PhD. According to a self assessment, all participants are familiar with Linked Data, most participants generate or use Linked Data (Figure 3.9). All participants are familiar with *UML class diagrams*, the underlying notation of *ShapeUML*, and the majority of the participants is familiar with the tool *WebVOWL*, a tool implementing *VOWL*, the underlying notation of *ShapeVOWL* (Figure 3.8). For the follow-up user study we could recruit 10 from the initially 12 participants in June 2021.

Test cases All test cases besides the initially shown general example are real world datasets from online resources such as GitHub, the visual benchmark *ShapeViBe*[17], and the SHACL performance benchmark by Schaffenrath et al. [59]. Figure 3.11 displays the distribution of RDF constraint concepts in the test cases from which a subset was relevant for asked questions.

We created the **general example** test case to expose various constraint concepts to participants in one example. This test case contains 40 constraint concepts in total, arranged around 3 node shapes and 6 property shapes (all predicate paths). This test case represents constraints on several attributes of a person as well as on email addresses. Some node shapes are linked with constraints but not all, additionally one node shape is deactivated.

The **Traffic Lights** test case represents constraints on RDF lists[18]. This test case contains 13 constraint concepts in total, arranged around 2 node shapes and 2 property shapes. Furthermore, this test case is characterized by containing a *zero-or-more* and *sequence* property path as well as several constraints on RDF list elements while also reusing an external data shape by referring to it with a constraint.

The **Address** test case is an excerpt from possible schema.org data shapes[19]. This test case contains 21 constraint concepts in total, arranged around 1 node shape and 3 property shapes (all predicate paths). It was manually curated to constrain schema.org addresses for Australia. This test case is characterized by containing logical constraints as well as a few other constraints on literal values.

The **DCAT** test case is an excerpt from the DCAT application profile for Swiss data portals[20]. This test case contains 30 constraint concepts in total, arranged around 1 node shape and 6 property shapes (all predicate paths). It has constraints on many properties of a single node, mostly constrained by their cardinality, datatype or class, but also by logical constraints, e.g. either class A *or* B.

The **Geo coordinates** test case is from the *ShabeViBe* benchmark[21]. This test case contains 36 constraint concepts in total, arranged around 2 node shapes and 5 property shapes (all predicate paths). It is characterized by containing combinations of different minimum and maximum constraints which can be easily confused. Namely, min/max cardinality constraints on properties, min/max value range constraints on property values as well as qualified cardinalities related to data shapes.

[17] Sven Lieber, "ShapeViBe", `http://web.archive.org/web/20220212145053/https://lov.ilabt.imec.be/unshacled/shape-vibe/` (archived website accessed February 12, 2022)

[18] Holger Knublauch, "How to define constraints on rdf:Lists using SHACL", `https://web.archive.org/web/20210102193406/https://www.topquadrant.com/constraints-on-rdflists-using-shacl/` (archived website accessed February 12, 2022)

[19] Holger Knublauch, "Schema.org (converted to SHACL by TopQuadrant) - Handwritten Example File", `http://web.archive.org/web/20220218231806/https://datashapes.org/schemashacl.shapes.ttl` (archived website accessed February 19, 2022)

[20] Reto Gmür, "SHACL Shapes for the DCAT Application Profile for Data Portals in Switzerland", `https://web.archive.org/web/20201228073627/https://github.com/factsmission/dcat-ap-ch-shacl` (archived website accessed February 12, 2022)

[21] Sven Lieber, "ShapeViBe - min-max-values module", `http://web.archive.org/web/20220218232114/https://lov.ilabt.imec.be/unshacled/shape-vibe/modules/min-max-values/` (archived website accessed February 19, 2022)

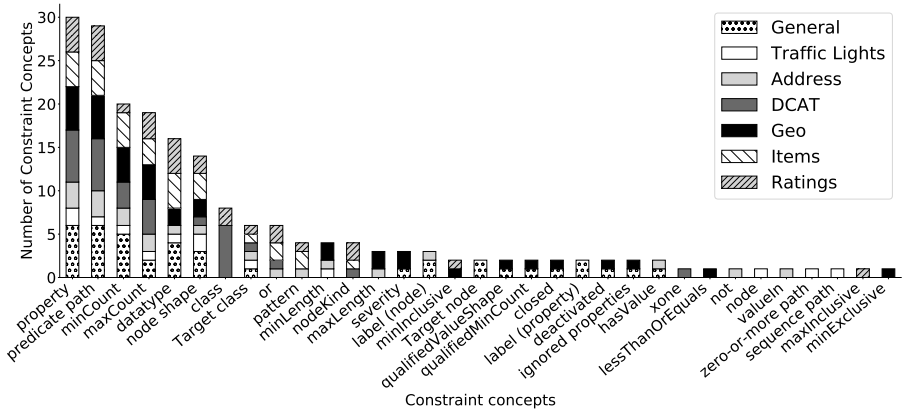


Figure 3.11: The occurrence of RDF constraint concepts in the test cases of our user study. Each test case contained several node and property shapes including cardinalities and a few selected other constraint concepts.

The **Items** test case is a subset of RDF constraints from a SHACL performance benchmark [59]. This test case contains 28 constraint concepts in total, arranged around 3 node shapes and 4 property shapes (all predicate paths). It mainly consists of datatype, class, disjunction, and literal pattern constraints. All property shapes are displayed with cardinalities, i.e. no default has to be assumed. Additionally 2 non-related node shapes are shown.

The **Ratings** test case is a subset of RDF constraints from a SHACL performance benchmark [59]. This test case contains 28 constraint concepts in total, arranged around 2 node shapes and 4 property shapes (all predicate paths) It mainly consists of datatype constraints, but also literal value constraints. One property shape has no cardinality constraints, thus default values need to be assumed.

3.6.3 Threats to Validity

External and internal threats to the experiment's validity exist, we identified the following threats and we discuss how we addressed them in our study design.

3.6.3.1 External Validity Threats

External validity threats occur when wrong inferences from sample data are made beyond the studied population or experimental setup [58]. We identified two external threats: **participants familiarity with Linked Data** and **experiment environment**.

Participants familiarity with Linked Data This threat concerns the generalization to individuals outside the study [58]. All our participants were recruited from Ghent University,

Belgium and RWTH Aachen, Germany and were familiar with Linked Data, thus the findings might not be generalizable to a more general population, e.g. Linked Data experts from industry or non Linked Data experts. However, at least with respect to Linked Data expertise this was intentional as we aimed to study users already familiar with RDF graphs. A prerequisite to understand RDF constraints which are the semantic constructs our visual notations represent.

Experiment environment This threat concerns the generalization to individuals outside the experiment's setting [58]. The experiment was an online questionnaire. Participants could use any browser and computer, thus, they participate from a well-known environment. No specific experimental setup prevents generalizations to individuals outside our study.

3.6.3.2 Internal Validity Threats

Internal validity threats concern the experimental setup or experience of participants which threaten the ability to draw correct conclusions about the population in the experiment [58]. We identified three internal threats: **selection bias**, **sample size** and **order effects**.

Selection bias This threat concerns the selection of biased participants, i.e. participants with certain characteristics that predispose them to have certain outcomes [58]. Our participants were all recruited from Ghent University and RWTH Aachen and have similar demographics. All participants have knowledge about Linked Data, but this is intentional as it is a prerequisite of the user study. To mitigate a selection bias all participants were assigned in a round-robin fashion to one of two groups, i.e. groups were not assigned based on specific characteristics. Some participants might be more familiar with one of the underlying visual notations of *ShapeUML* or *ShapeVOWL*. However, they self-assessed their familiarity with *UML class diagrams* and the *WebVOWL* tool in the pre-questionnaire, therefore any bias is visible. Please note that familiarity with one of the notations is considered positive as the design rationale of both visual notations is to build upon the underlying visual notation.

Sample size A small sample size may not have sufficient statistical power to detect an effect. Our sample size is relatively small. To mitigate this threat, we chose a *within-subject study design* [58]. It reduces errors associated with individual differences without requiring a large pool of participants[22].

Order effects When participants perform tasks several times certain effects like learning can occur. To counterbalance potential order effects when presenting *ShapeUML* and *ShapeVOWL*, we assigned participants in a round-robin fashion to two different groups. The

[22] David M. Lane, "Experimental Designs", `https://web.archive.org/web/20201216150003/http://onlinestatbook.com/2/research_design/designs.html` (archived website accessed February 12, 2022)

first group (group A) started with the first example in *ShapeUML*, the second in *ShapeVOWL*, the third in *ShapeUML* and so forth. Participants of the second group (group B) were presented the first example in *ShapeVOWL*, the second in *ShapeUML* and so forth.

3.6.4 Quantitative Results

We statistically validate the significance of the overall error rate differences between *ShapeUML* and *ShapeVOWL* (Section 3.6.4.1), analyze error rates per RDF constraint concept (Section 3.6.4.2), and analyze the participants' self assessment given by a Likert scale [60] (Section 3.6.4.3).

3.6.4.1 ShapeUML/ShapeVOWL Error Rate

Based on the correct answers, we calculated the error rates of all questions to compare *ShapeUML* and *ShapeVOWL*: initial questions for general examples and the 4 test cases (Section 3.6.1.1), as well as for questions in the follow-up study covering different tasks (Section 3.6.1.2).

There is no significant difference in the mean error rates of *ShapeUML* and *ShapeVOWL*. We first tested the normality of the error rates' distribution using a distribution plot and a *Shapiro-Wilk test* [61] with $\alpha = 5\%$ to determine which statistical test to choose. The data was not normally distributed, thus we performed a *Wilcoxon signed-rank test* [62] with $\alpha = 5\%$. The calculated p-value of 0.856 is bigger than α so we fail to reject the null hypothesis, which means there is no significant difference in the mean error rates.

3.6.4.2 Constraint Concepts

The questions of Section 3.6.1.1 represent tasks identifying fundamental concepts and core constraints of RDF constraint languages. **With both visual notations more than 81% of questions were answered correctly**. We elaborate for each constraint concept why mistakes possibly happened by qualitatively analyzing provided answers (Figure 3.12 and Figure 3.13) and optionally given free text feedback answers. We elaborate on (i) the tasks themselves, (ii) constraint concepts which have similar error rates between both notations, and (iii) constraint concepts which show more error variation between notations. Finally, we discuss overall findings. The analysis is further enriched with findings from a follow-up user study covering different questions, yet also involving certain constraint concepts.

Visualization tasks Questions of the main questionnaire combine the tasks to *retrieve a value* and *compute a derived value*, i.e. retrieving constraint types and compute the sum as aggregated value. If tasks resulted in wrong results it is usually because participants retrieved the value wrongly, for example because they did not understand a constraint concept or were unaware of default values (see following discussion); similarly for errors in the other tasks covered in the follow-up user study. However, for the main questionnaire a small possibility

that participants retrieved the value correctly and just computed the sum wrongly cannot be ruled out completely.

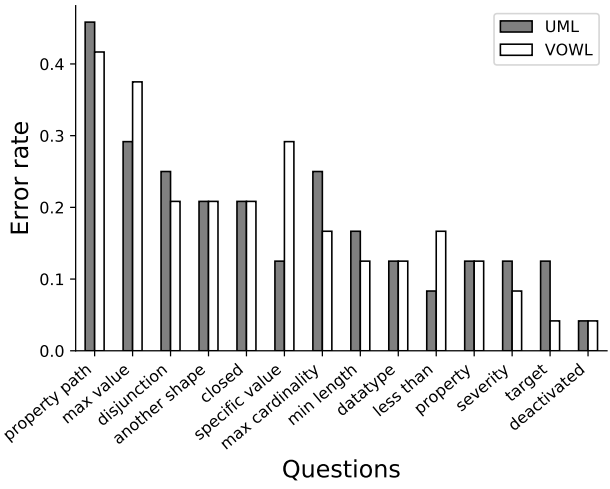


Figure 3.12: The error rates for both visual notations are relatively similar. Higher error rates for both visual notations were observed for *property paths* and *maximum value*.

Best and worst recognized constraint concepts There was (almost) no difference in error rates between ShapeUML and ShapeVOWL for 8 out of 13 constraint concepts. Most notably this covers the questions with the least and most errors, meaning that certain constraint concepts are equally good/bad recognized.

The least errors in the main questionnaire, only 4%, were observed for **deactivated** data shapes with both notations. This constraint type is indicated by struck-through text in ShapeUML and by dashed borders in ShapeVOWL. One participant who was not sure whether deactivation is a transitive constraint, pointed out that the visualization helped to make clear which data shapes are deactivated. The participant raised the same point for the **severity** constraint for which around 10% errors were made. Regarding visualization tasks, these constraint concepts had to be retrieved and then counted. In the follow-up study one question asked to only retrieve the correct value of a nodeKind constraint with multiple choice answers and there no errors were observed.

The most errors, more than 40%, were observed for **property paths**. In both notations this concept is encoded as an atomic label for property shapes. The provided answers suggest that participants have confused property paths with a combination of logical relationships and cardinalities on properties. To further investigate this issue, we performed a follow-up study where we explicitly asked the participants to indicate if they can identify property

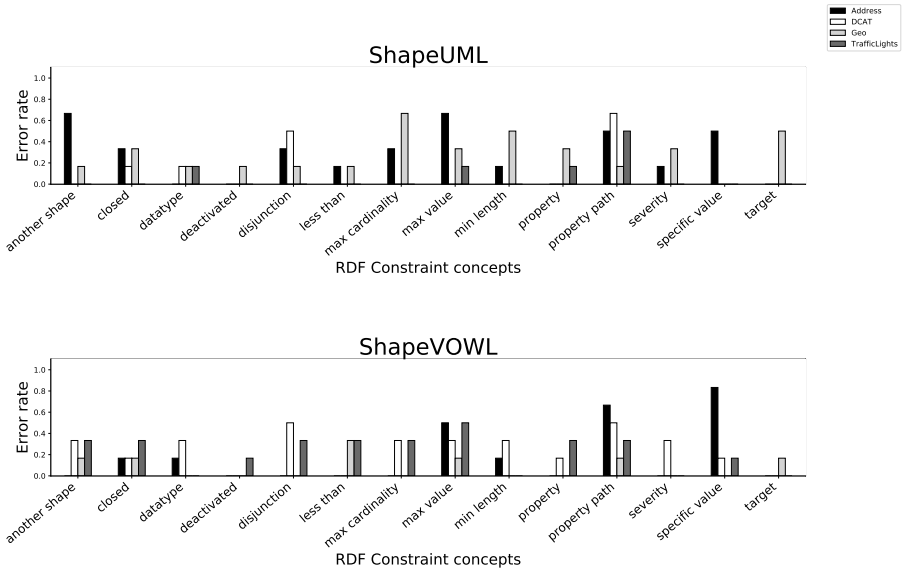


Figure 3.13: The error rates for the different questions across the 4 real world test cases of the main questionnaire. Most RDF constraint concepts related questions were answered correctly. Participants made the most errors for *property paths*, *maximum value*, *specific value* and *disjunction*.

paths and if so where. No property path was present in the examples, yet three participants identified property paths. One wrongly identified property path was a property constrained with a literal pattern containing a pipe symbol (logical disjunction) separated list of URIs, hence probably identified as an alternative path. In another example the value of one property shape contains a class constraint, the same class is mentioned as target by another shown node shape – without any explicit link. According to the provided feedback, this implicit link appeared to be a property path for at least one participant, another participant identified the same property but did not provide explicit reasoning. Finally, one participant mentioned to not understand the question "Do you see any 'property path' which is not just a single property?", which may indicate issues in understanding the underlying semantic construct (property paths were explained in the user introduction to both visual notations users had to read before the study).

Constraint concepts with similar error rates between visual notations Other constraint concepts with similar error rates between ShapeUML and ShapeVOWL were datatype, disjunction, property, closed and *comply with*. Whereas *comply with* constraints are encoded in the same way in both notations, the other constraint concepts are encoded differently, usually with more visual features in ShapeVOWL.

Datatype constraints were mostly identified correctly. Based on the provided answers it seems that one participant once wrongly counted a *nodeKind literal* constraint as a datatype constraint in a *retrieve value* task. Within an *ordering task* of the follow-up study participants had to alphabetically order properties with datatype constraints, but 40% of participants wrongly ordered a property with literal value without datatype constraint. However, they correctly excluded a property with class constraint in another example which at least for ShapeVOWL could indicate issues in understanding a datatype constraint if the value is visualized as a literal.

Two real world test cases contained **disjunction** constraints, one of the test cases additionally contained an exclusive disjunction constraint which was not supposed to be counted. However, one participant counted both and another participant indicated that in fact a second (exclusive) disjunction is present but was not counted by the participant. We acknowledge that our question leaves room for interpretation. One participant seemed to have counted properties instead of disjunctions and two participants identified this constraint when in fact it was not present, a zero-or-more property path constraint and an associated cardinality might have been counted as these were the only other constraint types shown in the example.

Property constraints were mostly correctly recognized. Mistakes were mainly made when the property shape or the corresponding node shape were deactivated, in this case some participants did not count the deactivated properties.

Two test cases contained **closed** constraints, participants also counted node shapes in other test cases when no closed constraint was present and did not count it when it was present.

Constraint concepts with error variation between visual notations Even though not significant, there is more variation between ShapeUML and ShapeVOWL error rates for 5 out of 13 constraint concepts. This includes the constraint concepts target, less than, specific value as well as constraints related to a minimum or maximum namely min length, max cardinality, and max value.

In 3 out of 4 real world test cases, there was 1 **target** concept encoded which was correctly recognized. However, in the last test case where no target concept was present, 4 participants probably counted the number of node shapes, which was 2, or the single node shape which had constraints attached. No additional feedback was provided, therefore we cannot interpret why these participants failed to recognize this constraint concept in the last test case.

A few participants miscounted the constraint type **less than or equals**, but the provided answers do not indicate they were mistaken for other constraint concepts.

Participants counted **specific value** constraints mostly correct. However, most errors occurred in the only test case which actually contained specific value constraints. One *valueIn* and one *hasValue* constraint were present which we both intended as "specific value" according to the question phrasing. Yet, 5 out of 12 participants only count 1 which suggests that they either counted *valueIn* or *hasValue*. This indicates the question was too ambiguous,

and indeed a participant also pointed out for another test case that even a *range* constraint might be considered "specific".

Error rates related to **minimum and maximum** values are explained by participants' misconception or misinterpreted questions. The provided answers suggest that participants counted cardinality constraints when in fact they were supposed to count min length or max value. One participant even pointed out to not understand the difference between max cardinality and maximum value. This indicates issues in understanding the underlying semantic constructs and not necessarily issues with the visual notations or questions. Furthermore, provided answers suggest that cardinalities were miscounted because firstly min/max pairs were counted instead of only minimum respectively maximum cardinalities in the general example, and secondly, wrong default cardinalities were assumed by the participants for the 4 real world test cases; "For me, an arrow without cardinalities means '1..1' " said by one participant is actually the wrong default, our introduction mentions a default cardinality of 0..* if no values are provided, i.e. no minimum or maximum constraints present.

Discussion of findings Based on the qualitative analysis of the results, we discuss findings related to default values, needed understanding of semantic constructs, and clarity of questions.

Default values should be encoded explicitly. On the one hand, according to provided feedback, encoded severities and constraint deactivation helped a participant to correctly interpret these concepts and discard the wrong assessment that these concepts are transitive. On the other hand, missing defaults lead participants to assume wrong cardinality defaults.

Clear documentation and/or tooltips are necessary to support users in understanding constraint concepts, because some constraint types are conceptually similar and need clarification. We noticed that users mistook e.g. different minimum and maximum constraint concepts with each other and property paths with a combination of logical relationships and cardinalities. A visual notation represents semantic constructs, the used visual notations do not suffer from symbol redundancy, symbol overload or symbol excess, thus they provide semiotic clarity (see Section 3.4). However, if underlying semantic constructs are not clear to a user, visual notations can only support to a small extent to alleviate misunderstandings, e.g. by providing semantic transparency.

We acknowledge that a limited number of questionnaire questions leave room for interpretation which negatively influences the analysis of results. For the follow-up user study we relied on adapted questions from related work, increasing consistency. Furthermore, for further research on RDF constraint visualization, we recommend a pilot study with a smaller number of participants – ideally from different backgrounds – to reveal possible ambiguities in questions.

3.6.4.3 Self Assessment

The post-questionnaire contained three questions in which the participants could self assess how *confident* they are with their answers, if they prefer *ShapeVOWL* over *ShapeUML* and

if they would like to use *ShapeVOWL* also for RDF constraint editing. These three questions were asked using a 7-point Likert scale from 1 (not agree at all) to 7 (fully agree).

All participants were asked if they are confident that their provided answers are correct. Their average value is 3.6 and median is 3, thus in a self assessment **participants are not very confident**. Participants could also provide feedback for each test case via a text field. Considering the provided feedback, some participants had trouble interpreting the asked questions which could relate with their low confidence.

All participants were asked if ShapeVOWL is preferred and the average value is 4.6 and median is 5, thus in a **self assessment participants prefer ShapeVOWL**. Similarly the average is 4.8 and median is 5 for the question if the participants would like to use ShapeVOWL to edit RDF constraints.

3.6.5 Qualitative analysis

Qualitative feedback is derived from each test case in the main questionnaire and generally for both notations in the post assessment. We qualitatively analyze provided answers for the post assessment following a common data analysis for qualitative data [58]: we explain the used analysis method, and present the results. In total 58% of participants answered this question.

3.6.5.1 Method

A general procedure for a qualitative analysis involves the process of "coding" [58], a commonly used technique for reducing qualitative data to meaningful information by assigning labels to chunks of data [63]. Following common guidelines [58] we read answers provided in the post questionnaire and thus were able to identify 5 high level codes: advantages, disadvantages, uncertainty, suggestion and preference. These codes are further detailed in a hierarchy, for example the high level code *advantages* is further specified as *easier comprehensible*, *display of sparse constraints* and *space efficiency*. In a similar fashion the other high level codes are further specified to be used as annotation for the qualitative data.

3.6.5.2 Interpretation and Meaning

Based on the created annotations we interpret the feedback provided by the participants by discussing the high level codes such as *advantages* and which information specifically was provided.

Participants preference and suggestions **Findings regarding preferences and provided suggestions correspond with our analysis of cognitive effective design principles in both notations**, i.e. *ShapeVOWL* adheres to more design principles. In total 4 participants explicitly indicated which visual notation they prefer, 3 of them prefer *ShapeVOWL* and 1 *ShapeUML*. To improve *ShapeUML* one participant suggested to remove

potential redundancies in *ShapeUML* (see disadvantages) and "a more user-friendly visualisation of UML (eg: colors, option to hide parts)".

Advantages Slightly more advantages were pointed out for *ShapeVOWL*, whereas both notations have their own advantages mostly related to how comprehensible they are for certain use cases. In total 5 participants provided feedback with respect to advantages, 3 of them for *ShapeUML* and 4 for *ShapeVOWL*. *ShapeUML* was recognized more space efficient by 1 participant whereas the same participant mentioned that for sparse constraints *ShapeVOWL* "looks cleaner". For both notations 3 participants indicated that the respective notation is easier comprehensible. For *ShapeUML* 2 participants pointed out that its list representation allows to condense more constraints of a single node and 1 participant expressed that *ShapeUML* is more intuitive. For *ShapeVOWL* 2 participants pointed out that it is easier to spot constraints due to visual features and 1 participant explicitly mentioned the familiarity to *VOWL* as reason.

Disadvantages Although *ShapeVOWL* was preferred, more disadvantages were explicitly pointed out for it compared to *ShapeUML*. In total 3 participants provided feedback with respect to disadvantages, 1 for *ShapeUML* and 3 for *ShapeVOWL*. *ShapeUML* was perceived redundant by 1 participant in a negative sense, i.e. the repetition of property paths both in the *data shape* rectangle and on the relationship between *node and property shape*. For *ShapeVOWL*, 2 participants reported possible complications when interacting with it, namely "many small comment boxes" for constraints which "would be less orderly" and the different geometrical shapes and colors as "things" which are "more of a hassle to work with (more clicking and less typing involved)". Additionally, 1 participant noted that *ShapeVOWL* "looks very simplistic, but needs more understanding to apply".

Uncertainty Corresponding with Likert-scale answers regarding confidence and our quantitative analysis, participants explicitly mentioned unclear terminology. In total 3 participants provided feedback with respect to unclarity, whereas 2 participants mentioned an unclear terminology and 1 participant ambiguous questions. This corresponds also with observations from the quantitative analysis, i.e. wrong answers for conceptually similar constraint types.

3.7 Discussion and Conclusion

Data integration as main challenge in our time can be addressed with the uniform graph data model of RDF. Use case specific data quality requires validation, but currently human users – often the creators of constraints – are not well supported when viewing and editing RDF constraints. Therefore, we investigated visual notations for RDF constraints tailored for the human information processing system to answer the research question *how we can support users in viewing RDF constraints?*. Furthermore, we presented a new version of our tool *UnSHACLed* that implements investigated visual notations. The human information

processing system requires effective visual notations that move the cognitive load from the slow cognitive processing to the fast perceptual processing.

The two visual notations *UML* and *VOWL* are broadly used within the Semantic Web community. We reused these already familiar to users notations and adapted them for RDF constraints: the two notations are dubbed *ShapeUML* and *ShapeVOWL*.

In particular, we investigated in this work the hypothesis that "Users familiar with Linked Data can answer questions about visually represented RDF constraints more accurately with a VOWL-based visual notation than with an UML-based visual notation" We could not validate this hypothesis: there was no significant difference in error mean values which would indicate that better results are achieved with *ShapeVOWL*. However, analyzing the design considerations of both visual notations and user study results in detail we conclude the following things.

Theory versus practice For both notations on average 81% of questions related to RDF constraints were answered correctly. Even though different constraint types were recognized more accurately with one or the other notation, we could not measure a statistically significant error difference between *ShapeUML* and *ShapeVOWL* in the performed user evaluation. However, according to a comparison between both visual notations based on design principles (Section 3.4 and Table 3.1) *ShapeVOWL* is more cognitive effective in theory. Participants also self assessed to prefer *ShapeVOWL*, however, there might have been a bias in the post questionnaire because it was directly asked if the participants prefer *ShapeVOWL*. Eventually, the number of participants was small which motivates further studies. Different use cases and types of users exist which also motivates further research covering specific domains.

ShapeVOWL disadvantages Disadvantages brought up in the qualitative analysis – such as complicated interaction or space efficiency – mainly concern more complex and dense RDF constraint graphs. Instead of aiming for a one-size-fits-all visual notation, such disadvantages can be mitigated by complementary functionality of RDF constraint editors implementing ShapeVOWL. Existing visualization task taxonomies [49, 50, 51] may guide this feature implementation as they allow to describe cognitive tasks with respect to goals and thus provide candidate tasks which can be implemented as interactive functionality, e.g. filtering or sorting displayed constraints based on selected criteria in the tool rather than the mind of the user. Similarly, such functionality can improve the use of *ShapeUML* as well.

Clear and efficient text encoding of ShapeUML with potential improvement Despite visual features for cognitive effective processing by humans, we noticed that *ShapeUML*s textual representation in certain cases was as effective as *ShapeVOWL* and sometimes even more effective. According to our qualitative analysis, *ShapeUML* has an advantage for more dense or complex RDF constraint graphs due to its space efficient representation. Although text is processed using the slower *cognitive processing system* [10], this system might be needed for RDF constraints in any case. **But instead of providing**

an enhanced alternative notation such as *ShapeVOWL*, the already space efficient *ShapeUML* can be improved by addressing specific design principles to support users even better. However, this would cause that *ShapeUML* may deviate from the *UML* specification, but as one participant put it: "I do believe a more UML-like format would be preferred by users IF [sic] users were allowed some slack from the rigid UML definitions".

Visual Notation Visual notations developed with effectiveness in mind may not be necessarily adopted [48], but we built on already familiar visual notations to increase a possible adoption. Despite optimized perceptual processing, users may fall for a familiarity bias: even though experts perform worse with familiar less-optimal notations, they still hesitate to switch to a non-familiar but more optimal notation [48]. Our solution tries to combine the best of both worlds, i.e. familiar notations adapted for RDF constraints by relying on cognitive effective design principles. However, more involvement from users of different domains is needed, for instance to resolve findings of our study related to misunderstood terminology or concepts. Improvements such as more specific labels or visual symbols for conceptually similar constraint types can be developed in a participatory fashion with targeted audiences, which increases the chance of adopting [48].

Limitations Our work covers the accurate processing of visually represented RDF constraint concepts and, thus, does not cover scalability of visual notations or the speed in which users processed presented information. To the best of our knowledge this is the first work investigating visual notations for RDF constraints in detail. Hence our results are initial results. We studied how different RDF constraint concepts can be visualized and how this affects the accuracy of user-provided answers based on related questions.

Future Work Findings of our analysis suggest **future work regarding the integration of visual notations in RDF editors, the visual notations itself and, additionally, a possible mapping from ShEx concepts**. In future work we plan to incorporate features in our tool *UnSHACLed* to complement both visual notations such as semantic zooming to improve working with large RDF constraint graphs or enhanced user interactions to accommodate for different use cases; generally, more research towards user interactions is needed to understand real needs, especially with respect to different editing approaches for RDF constraints.

Regarding the visual notations, a visually enhanced *ShapeUML* variant – as suggested by a participant – could represent a trade-off in space efficiency and effective processing and it would be an appropriate candidate for future developments and user evaluations.

Finally, a mapping from ShEx concepts to the presented visual notations could motivate efforts to extend the presented tool *UnSHACLed* with respect to ShEx validation of RDF data, thus more users would profit from the developed effective visual notations.

References

[1] Steffen Lohmann, Stefan Negru, Florian Haag, and Thomas Ertl. "Visualizing ontologies with VOWL". In: *Semantic Web* 7 (May 2016), pp. 399–419. ISSN: 2210-4968. DOI: 10.3233/sw-150200.

[2] Pieter Heyvaert, Anastasia Dimou, Ben De Meester, Tom Seymoens, Aron-Levi Herregodts, Ruben Verborgh, Dimitrie Schuurman, and Erik Mannens. "Specification and implementation of mapping rule visualization and editing: MapVOWL and the RMLEditor". In: *Web Semantics: Science, Services and Agents on the World Wide Web* 49 (Mar. 2018), pp. 31–50. DOI: 10.1016/j.websem.2017.12.003. URL: https://biblio.ugent.be/publication/8559065/file/8559068.pdf.

[3] Stephen Cranefield and M. Purvis. "UML as an Ontology Modelling Language". In: *Intelligent Information Integration*. 1999.

[4] OMG. *Ontology Definition Metamodel, Version 1.1*. Tech. rep. Object Management Group, Sept. 2014. URL: https://www.omg.org/spec/ODM/1.1.

[5] Jose Emilio Labra Gayo, Eric Prud'hommeaux, Iovka Boneva, and Dimitris Kontokostas. *Validating RDF Data*. Vol. 7. Synthesis Lectures on the Semantic Web: Theory and Technology 1. Morgan & Claypool Publishers LLC, Sept. 2017, pp. 1–328. DOI: 10.2200/s00786ed1v01y201707wbe016. URL: http://book.validatingrdf.com/.

[6] Holger Knublauch and Dimitris Kontokostas. *Shapes Constraint Language (SHACL)*. Recommendation. World Wide Web Consortium (W3C), July 2017. URL: https://www.w3.org/TR/shacl/.

[7] J. M. Juran. *Juran's Quality Control Handbook*. Ed. by Frank M. Mryna. 4th. Texas, USA: Mcgraw-Hill, Aug. 1988. URL: http://www.pqm-online.com/assets/files/lib/books/juran.pdf.

[8] Florian Haag, Steffen Lohmann, Stephan Siek, and Thomas Ertl. "QueryVOWL: A Visual Query Notation for Linked Data". In: *Proceedings of ESWC 2015 Satellite Events*. Vol. 9341. LNCS. Springer, 2015, pp. 387–402. DOI: 10.1007/978-3-319-25639-9_51.

[9] Marc Weise, Steffen Lohmann, and Florian Haag. "Extraction and visualization of tbox information from sparql endpoints". In: *European Knowledge Acquisition Workshop*. Springer. 2016, pp. 713–728. DOI: 10.1007/978-3-319-49004-5_46.

[10] Daniel Moody. "The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering". In: *IEEE Transactions on Software Engineering* 35.6 (Nov. 2009), pp. 756–779. DOI: 10.1109/tse.2009.67.

[11] Ben De Meester, Pieter Heyvaert, Anastasia Dimou, and Ruben Verborgh. "Towards a Uniform User Interface for Editing Data Shapes". In: *Proceedings of the 4th International Workshop on Visualization and Interaction for Ontologies and Linked Data*. Ed. by Valentina Ivanova, Patrick Lambrix, Steffen Lohmann, and Catia Pesquita. Vol. 2187. CEUR Workshop Proceedings. CEUR-WS.org, Oct. 2018, pp. 13–24. URL: http://ceur-ws.org/Vol-2187/paper2.pdf.

[12] Jose Emilio Labra Gayo, Daniel Fernández-Álvarez, and Herminio García-González. "RDFShape: An RDF Playground Based on Shapes". In: *Proceedings of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks co-located with 17th International Semantic Web Conference (ISWC 2018)*. Ed. by Marieke Van Erp, Medha Atre, Vanessa Lopez, Kavitha Srinivas, and Carolina Fortuna. Vol. 2180. CEUR Workshop Proceedings. CEUR-WS.org, Oct. 2018.

[13] Eric Prud'hommeaux, Jose Emilio Labra Gayo, and Harold Solbrig. "Shape expressions: an RDF validation and transformation language". In: *Proceedings of the 10th International Conference on Semantic Systems*. Ed. by Harald Sack, Agata Filipowska, Jens Lehmann, and Sebastian Hellmann. ACM. New York, NY, United States: Association for Computing Machinery, 2014, pp. 32–40. DOI: 10.1145/2660517.2660523. URL: http://dl.acm.org/citation.cfm?id=2660523.

[14] Sven Lieber, Ben De Meester, Anastasia Dimou, and Ruben Verborgh. "Statistics about Data Shape Use in RDF Data". In: *Proceedings of the 19th International Semantic Web Conference: Posters, Demos, and Industry Tracks*. Ed. by Kerry Taylor, Rafael Gonçalves, Freddy Lecue, and Jun Yan. Vol. 2721. CEUR Workshop Proceedings. Nov. 2020, pp. 330–335. URL: http://ceur-ws.org/Vol-2721/paper584.pdf.

[15] OMG. *Unified Modeling Language, Version 2.5.1*. Tech. rep. Object Management Group, Dec. 2017. URL: https://www.omg.org/spec/UML/2.5.1/.

[16] Claude Elwood Shannon and Warren Weaver. *The Mathematical Theory of Communication*. 1949. DOI: 10.1063/1.3067010.

[17] Anne M Treisman and Garry Gelade. "A Feature Integration Theory of Attention". In: *Cognitive psychology* 12.1 (1980), pp. 97–136. DOI: 10.1093/acprof:osobl/9780199734337.003.0011.

[18] Max Wertheimer. "Laws of Organization in Perceptual Forms". In: *A source book of Gestalt psychology* (1938), pp. 71–88.

[19] Jacques Bertin. *Semiology of Graphics: Diagrams Networks, Maps*. Tech. rep. University of Wisconsin Press, 1983.

[20] Nelson Goodman. *Languages of Art: An Approach to a Theory of Symbols*. Hackett publishing, 1976. ISBN: 9780915144341. DOI: 10.2307/2066203.

[21] Holger Knublauch, James A. Hendler, and Kingsley Idehen. *SPIN – Overview and Motivation*. Member Submission. World Wide Web Consortium (W3C), Feb. 2011. URL: https://www.w3.org/Submission/spin-overview/.

[22] Arthur Ryman. *Resource Shape 2.0*. Member Submission. World Wide Web Consortium (W3C), Feb. 2014. URL: https://www.w3.org/Submission/shapes/.

[23] Arthur G. Ryman, Arnaud J. Le Hors, and Steve Speicher. "OSLC Resource Shape: A language for defining constraints on Linked Data". In: *Proceedings of the WWW2013 Workshop on Linked Data on the Web* (Rio de Janeiro, Brazil). Ed. by Christian Bizer, Tom Heath, Tim Berners-Lee, Michael Hausenblas, and Sören Auer. Vol. 996. CEUR Workshop Proceedings. CEUR-WS.org, May 2013. URL: http://ceur-ws.org/Vol-996/papers/ldow2013-paper-02.pdf.

[24] Eric Prud'hommeaux. *Shape Expressions 1.0 Primer*. Member Submission. World Wide Web Consortium (W3C), June 2014. URL: https://www.w3.org/Submission/2014/SUBM-shex-primer-20140602/.

[25] Holger Knublauch. *From SPIN to SHACL*. Tech. rep. Aug. 2017. URL: https://spinrdf.org/spin-shacl.html.

[26] Thomas Bosch, Andreas Nolle, Erman Acar, and Kai Eckert. *RDF Validation Requirements – Evaluation and Logical Underpinning*. arXiv preprint. July 2015. URL: http://arxiv.org/abs/1501.03933.

[27] Sven Lieber, Ben De Meester, Anastasia Dimou, and Ruben Verborgh. "MontoloStats – Ontology Modeling Statistics". In: *Proceedings of the 10th International Conference on Knowledge Capture - K-CAP '19*. Ed. by Raphaël Troncy. ACM, Nov. 2019, pp. 69–76. DOI: 10.1145/3360901.3364433.

[28] Simon Steyskal and Karen Coyle. *SHACL Use Cases and Requirements*. Tech. rep. https://www.w3.org/TR/2017/NOTE-shacl-ucr-20170720/. W3C, July 2017.

[29] Dieter De Paepe, Geert Thijs, Raf Buyle, Ruben Verborgh, and Erik Mannens. "Automated UML-Based Ontology Generation in OSLO²". In: *The Semantic Web: ESWC 2017 Satellite Events – ESWC 2017*. Ed. by Eva Blomqvist, Katja Hose, Heiko Paulheim, Agnieszka Ławrynowicz, Fabio Ciravegna, and Olaf Hartig. Vol. 10577. Lecture Notes in Computer Science. Springer, Cham, 2017, pp. 93–97. DOI: 10.1007/978-3-319-70407-4_18.

[30] Andrea Cimmino, Alba Fernández-Izquierdo, and Raúl García-Castro. "Astrea: Automatic Generation of SHACL Shapes from Ontologies". In: *European Semantic Web Conference (ESWC)*. Ed. by Andreas Harth, Sabrina Kirrane, Axel-Cyrille Ngonga Ngomo, Heiko Paulheim, Anisa Rula, Peter Haase, and Michael Cochez. Springer. Springer International Publishing, 2020, pp. 497–513. DOI: 10.1007/978-3-030-49461-2_29.

[31] Fajar J. Ekaputra and Xiashuo Lin. "SHACL4P: SHACL constraints validation within Protégé ontology editor". In: *2016 International Conference on Data and Software Engineering (ICoDSE)*. IEEE, Oct. 2016. DOI: 10.1109/icodse.2016.7936162.

[32] Iovka Boneva, Jérémie Dusart, Daniel Fernández Alvarez, and Jose Emilio Labra Gayo. "Shape Designer for ShEx and SHACL Constraints". In: *Proceedings of the ISWC 2019 Satellite Tracks (Poster & Demonstrations, Industry, and Outrageous Ideas)*. Vol. 2456. CEUR-WS.org, Oct. 2019, pp. 269–272. URL: https://hal.archives-ouvertes.fr/hal-02268667.

[33] Rossana Paciello, Daniele Bailoa, Luca Tranib, Valerio Vinciarellia, Manuela Sbarrac, and Sara Capotostic. *SHAPEness: a SHACL-driven RDF Graph Editor*. 2021. URL: http://www.semantic-web-journal.net/content/shapeness-shacl-driven-rdf-graph-editor.

[34] Peter Haase, Daniel M Herzig, Artem Kozlov, Andriy Nikolov, and Johannes Trame. "metaphactory: A Platform for Knowledge Graph Management". In: *Semantic Web* 10.6 (2019), pp. 1109–1125. DOI: 10.3233/sw-190360.

[35] Conrad Bock, Achille Fokoue, Peter Haase, Rinke Hoekstra, Ian Horrocks, Alan Ruttenberg, Uli Sattler, and Michael Smith. *OWL 2 Web Ontology Language – Structural Specification and Functional-Style Syntax (Second Edition)*. Recommendation. World Wide Web Consortium (W3C), Dec. 2012. URL: http://www.w3.org/TR/owl2-syntax/.

[36] OMG. *Object Constraint Language, Version 2.4*. Tech. rep. Object Management Group, Feb. 2014. URL: https://www.omg.org/spec/OCL/2.4.

[37] A. Katifori, C. Halatsis, G. Lepouras, C. Vassilakis, and E. Giannopoulou. "Ontology visualization methods – a survey". In: *ACM Comput. Surv.* 39 (2007), p. 10. DOI: 10.1145/1287620.1287621.

[38] S. Mikhailov, Mikhail Petrov, and Birger Lantow. "Ontology Visualization: A Systematic Literature Analysis". In: *Joint Proceedings of the BIR 2016 Workshops and Doctoral Consortium co-located with 15th International Conference on Perspectives in Business Informatics Research (BIR 2016)*. Vol. 1684. CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[39] Nassira Achich, B. Bouaziz, Alsayed Algergawy, and F. Gargouri. "Ontology Visualization: An Overview". In: *17th International Conference on Intelligent Systems Design and Applications (ISDA)*. 2017. DOI: 10.1007/978-3-319-76348-4_84.

[40] Anton Anikin, Dmitry Litovkin, Marina Kultsova, Elena Sarkisova, and Tatyana Petrova. "Ontology Visualization: Approaches and Software Tools for Visual Representation of Large Ontologies in Learning". In: *Creativity in Intelligent Technologies and Data Science*. Springer International Publishing, 2017, pp. 133–149. ISBN: 978-3-319-65551-2. DOI: 10.1007/978-3-319-65551-2_10.

[41] Marek Dudáš, Steffen Lohmann, Vojtěch Svátek, and Dmitry Pavlov. "Ontology visualization methods and tools: a survey of the state of the art". In: *The Knowledge Engineering Review* 33 (2018). DOI: 10.1017/s0269888918000073.

[42] Merlin Florrence Joseph and Ravi Lourdusamy. "Feature analysis of ontology visualization methods and tools". In: *Computer Science and Information Technologies* 1.2 (2020), pp. 61–77. DOI: 10.11591/csit.v1i2.p61-77.

[43] Fatma Ghorbel, Nebrasse Ellouze, Fay Métais, Faiez Gargouri, Noura Herradi, et al. "MEMO GRAPH: an ontology visualization tool for everyone". In: *Procedia Computer Science* 96 (2016), pp. 265–274. DOI: 10.1016/j.procs.2016.08.139.

[44] G. Braun, C. Gimenez, L. Cecchi, and P. Fillottrani. "crowd: A Visual Tool for Involving Stakeholders into Ontology Engineering Tasks". In: *KI - Künstliche Intelligenz* 34.3 (2020), pp. 365–371. DOI: 10.1007/s13218-020-00657-8.

[45] Daniel Garijo. "WIDOCO: a wizard for documenting ontologies". In: *The Semantic Web - International Semantic Web Conference (ISWC 2017)*. Ed. by Claudia d'Amato, Miriam Fernandez, Valentina Tamma, Freddy Lecue, Philippe Cudré-Mauroux, Juan Sequeda, Christoph Lange, and Jeff Heflin. Springer. Springer International Publishing, 2017, pp. 94–102. DOI: 10.1007/978-3-319-68204-4_9.

[46] Steffen Lohmann, Vincent Link, Eduard Marbach, and Stefan Negru. "WebVOWL: Web-based visualization of ontologies". In: *International Conference on Knowledge Engineering and Knowledge Management*. Springer. 2014, pp. 154–158. DOI: 10.1007/978-3-319-17966-7_21.

[47] Steffen Lohmann, Stefan Negru, and David Bold. "The ProtégéVOWL plugin: ontology visualization for everyone". In: *European Semantic Web Conference*. Springer. 2014, pp. 395–400. DOI: 10.1007/978-3-319-11955-7_55.

[48] Aritra Dasgupta. "Experts' Familiarity versus Optimality of Visualization Design: How Familiarity Affects Perceived and Objective Task Performance". In: *Cognitive Biases in Visualizations*. Springer, 2018, pp. 75–86. DOI: 10.1007/978-3-319-95831-6_6.

[49] Matthew Brehmer and Tamara Munzner. "A Multi-level Typology of Abstract Visualization Tasks". In: *IEEE transactions on visualization and computer graphics* 19.12 (2013), pp. 2376–2385. DOI: 10.1109/tvcg.2013.124.

[50] Stephen Wehrend and Clayton Lewis. "A Problem-oriented Classification of Visualization Techniques". In: *Proceedings of the First IEEE Conference on Visualization: Visualization90*. IEEE. 1990, pp. 139–143.

[51] Michelle X Zhou and Steven K Feiner. "Visual Task Characterization for Automated Visual Discourse Synthesis". In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. 1998, pp. 392–399. DOI: 10.1145/274644.274698.

[52] Robert Amar, James Eagan, and John Stasko. "Low-level Components of Analytic Activity in Information Visualization". In: *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.* IEEE. 2005, pp. 111–117. DOI: 10.1109/INFVIS.2005.1532136.

[53] Emile Morse, Michael Lewis, and Kai A Olsen. "Evaluating Visualizations: Using a Taxonomic Guide". In: *International Journal of Human-Computer Studies* 53.5 (2000), pp. 637–662. DOI: 10.1006/ijhc.2000.0412.

[54] Eliane RA Valiati, Marcelo S Pimenta, and Carla MDS Freitas. "A Taxonomy of Tasks for Guiding the Evaluation of Multidimensional Visualizations". In: *Proceedings of the 2006 AVI workshop on Beyond time and errors: novel evaluation methods for information visualization*. 2006, pp. 1–6. DOI: 10.1145/1168149.1168169.

[55] Bahador Saket, Alex Endert, and Çağatay Demiralp. "Task-based Effectiveness of Basic Visualizations". In: *IEEE transactions on visualization and computer graphics* 25.7 (2018), pp. 2505–2512. DOI: 10.1109/tvcg.2018.2829750.

[56] Saša Kuhar and Gregor Polančič. "Conceptualization, measurement, and application of semantic transparency in visual notations". In: *Software and Systems Modeling* (May 2021). DOI: 10.1007/s10270-021-00888-9.

[57] William A Pike, John Stasko, Remco Chang, and Theresa A O'connell. "The Science of Interaction". In: *Information visualization* 8.4 (2009), pp. 263–274. DOI: 10.1057/ivs.2009.22.

[58] J. W. Creswell. *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*. 3rd ed. Sage Publications Ltd., 2008. ISBN: 9781506386706.

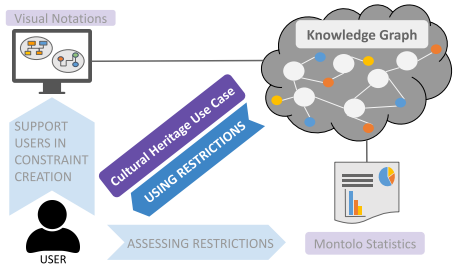
[59] Robert Schaffenrath, Daniel Proksch, Markus Kopp, Iacopo Albasini, Oleksandra Panasiuk, and Anna Fensel. "Benchmark for Performance Evaluation of SHACL Implementations in Graph Databases". In: *International Joint Conference on Rules and Reasoning*. Springer. 2020, pp. 82–96. DOI: 10.1007/978-3-030-57977-7_6.

[60] Rensis Likert. "A Technique for the Measurement of Attitudes". In: *Archives of psychology* 22.140 (1932), p. 55.

[61] Samuel Sanford Shapiro and Martin B Wilk. "An analysis of variance test for normality (complete samples)". In: *Biometrika* 52.3/4 (1965), pp. 591–611. DOI: 10.2307/2333709.

[62] Frank Wilcoxon. "Individual comparisons by ranking methods". In: *Breakthroughs in statistics*. Springer, 1992, pp. 196–202. DOI: 10.1007/978-1-4612-4380-9_16.

[63] Jan Recker. *Scientific Research in Information Systems: A Beginner's Guide*. Springer Science & Business Media, 2013. ISBN: 9783642-300479.



Chapter 4

Knowledge Graph Restrictions for Social Media Archiving

Whereas the previous chapters tackled challenges with respect to the assessment and creation of restrictions, this chapter focuses on their use. Because the use of restrictions is use case specific, this chapter focusses on a particular use case: data stewardship for the preservation of social media as cultural heritage. This chapter presents the following contributions to the use of restrictions:

- The BESOCIAL workflow for social media archiving which was validated in a social media archiving use case at the Royal Library of Belgium (KBR)
- An approach for declarative data quality assessment using W3C-related specifications

We address Research Question 3 "How can axioms and constraints support archiving institutions in the data stewardship of heterogeneous social media data?" and validate Hypothesis 3 "The W3C-recommended constraint language SHACL can be used to declaratively assess data quality metrics for use case specific data quality of heterogeneous social media data, integrated into an RDF graph with formal meaning"

Section 4.1 which corresponds with the publication "BESOCIAL: A Sustainable Knowledge Graph-Based Workflow for Social Media Archiving" presents the Knowledge Graph and the use case in which it was validated. Section 4.2 builds upon related work and proposes a declarative data quality assessment using Knowledge Graph constraints.

4.1 BESOCIAL: A Knowledge Graph-based Workflow for Social Media Archiving

Sven Lieber, Dylan Van Assche, Sally Chambers, Fien Messens, Friedel Geeraert, Julie M. Birkholz, Anastasia Dimou

Published as "BESOCIAL: A Sustainable Knowledge Graph-Based Workflow for Social Media Archiving", in *SEMANTICS2021, the 17th International Conference on Semantic Systems*, Amsterdam, The Netherlands, September 6-9, 2021, Pages 198-212.

Abstract

Social media as infrastructure for public discourse provide valuable information that needs to be preserved. Several tools for social media harvesting exist, but still only fragmented workflows may be formed with different combinations of such tools. On top of that, social media data but also preservation-related metadata standards are heterogeneous, resulting in a costly manual process. In the framework of BESOCIAL at the Royal Library of Belgium (KBR), we develop a sustainable social media archiving workflow that integrates heterogeneous data sources in a Europeana and PREMIS-based data model to describe data preserved by open source tools. This allows data stewardship on a uniform representation and we generate metadata records automatically via queries. In this paper, we present a comparison of social media harvesting tools and our Knowledge Graph-based solution which reuses off-the-shelf open source tools to harvest social media and automatically generate preservation-related metadata records. We validate our solution by generating Encoded Archival Description (EAD) and bibliographic MARC records for preservation of harvested social media collections from Twitter collected at KBR. Other archiving institutions can build upon our solution and customize it to their own social media archiving policies.

4.1.1 Introduction

The web, and in particular social platforms, have become social infrastructures for public discourse [1, 2] which serve as records of the past. However, these records are usually centrally maintained by profit-based social media providers and, thus, preservation by third parties is necessary.

Data preservation is a resource expensive task which requires long term commitment involving software, data and human resources [3]. Social media poses preservation challenges: non-technical experts of the GLAM domain[1] have to select harvesting tools, and social media consists of dynamic content[4] and heterogeneous data formats which have to be adequately processed and described.

Furthermore, preservation-related metadata for social media is also heterogeneous, aggravating interoperability and data stewardship. Usually metadata documents which describe collections allow efficiently identifying sources [3]. Yet, different preservation systems may

[1] Galleries, Libraries, Archives, and Museums.

require metadata in different syntax which also represent different perspectives. For example, MARCXML[2] records from the library domain may be used to describe a social media collection from a bibliographic point of view, whereas Encoded Archival Description (EAD) [3] XML records from the archive domain may be used to describe the collection's content hierarchically in more detail. This hampers data stewardship because there is no uniform and interoperable description of the preserved social media collections, let alone provenance of the collection process itself which is crucial[5, 4].

Semantic Web and Knowledge Graphs are promising solutions in the GLAM domain [6] as they enable applications across heterogeneous data and address the mentioned issues. However, existing approaches [7, 8] assume already curated metadata records as inputs for Knowledge Graphs. Thus, they do not solve the initial issue of a costly manual curation of metadata records. Instead, a Knowledge Graph-based solution can be applied earlier in the workflow to support data stewardship by a uniform description of both social media collections and provenance information about the collection process.

We reuse existing open-source tools – and metadata they produce – to generate a Knowledge Graph, addressing interoperability issues and enabling data stewardship. Therefore we support users in the GLAM domain with basic IT understanding but limited technical skills [9]. Because we provide a workflow based on open source software and data models, independent of particular archiving use cases, we consider our solution sustainable. We analyzed existing social media harvesting tools to identify promising reuse candidates. Then we complemented selected tools with open source components to design a sustainable workflow driven by a Knowledge Graph: heterogeneous data are mapped to RDF, from which domain-specific metadata records are generated via queries. We validate our workflow by applying it on a social media archiving use case at Royal Library of Belgium (KBR). in which we created a Knowledge Graph based on harvested Twitter content, and generate MARC and EAD records.

Our contributions are (i) a comparative analysis of existing social media archiving tools, and (ii) a sustainable social media archiving workflow based on declarative RML mapping rules to generate Europeana Data Model and PREMIS-based [10] RDF from heterogeneous data sources, and metadata record generation based on reusable templates and Knowledge Graph queries. These open source resources as well as a full version of the comparison are available at `https://github.com/RMLio/social-media-archiving`.

In Section 4.1.2 we present related work. In Section 4.1.3 we provide a comparative analysis of social media harvesting tools. In Section 4.1.4 we present our Knowledge Graph-based solution which we validate in an archiving use case in Section 4.1.5. Finally, in Section 4.1.6 we discuss and conclude.

[2] Library of Congress, "MARC21 Format for BIBLIOGRAPHIC DATA", `http://web.archive.org/web/20220203181828/https://www.loc.gov/marc/bibliographic/` (archived website accessed February 12, 2022)

[3] Library of Congress, "Encoded Archival Description", `https://www.loc.gov/ead/` (archived website accessed February 12, 2022)

4.1.2 Related Work

This work aims for an open source solution for social media archiving. To the best of our knowledge, there are no openly available workflows for social media archiving which cover both harvesting and cataloguing in an automated fashion. We discuss (i) tools and frameworks related to web archiving and social media harvesting in Section 4.1.2.1, to reflect on existing efforts to archive social media, (ii) metadata standards of the GLAM domain related to archiving in Section 4.1.2.2, to elaborate on domain-specific practices, and (iii) how our solutions compares to existing Knowledge Graph-based solutions in Section 4.1.2.3.

4.1.2.1 Social Media Archiving

We discuss web archiving, tools to harvest social media, as well as methodologies and tools used in the GLAM domain to analyze social media.

Commonly-used workflows for web archiving involve (i) describing collections, i.e. which website domains should be harvested and how often, (ii) fetching content using web harvesters, e.g., Heritrix [11] to preserve websites in Web ARChive (WARC) files [12], a format to preserve both content and HTTP requests, and (iii) accessing archived collections using replay software, e.g., WaybackMachine [13] or pyweb[4] as in the internet archive[5]. Software like Web Curator Tool [14] or Annotation and Curation Tool (w3act)[6] can be used as management interface to describe collections and schedule harvests. Websites for preservation are usually selected based on their top-level domain for which archival institutions may have a legal obligation to preserve its content. However, such workflows keep harvested information and metadata locked up in several data formats. Social media poses different challenges compared to web archiving due to its dynamic content [4] and different data formats used by different providers. Thus, web archiving workflows cannot be adjusted to sustainable social media harvesting workflows out of the box.

Similar tooling exists for social media archiving, but is limited to collection creation and harvesting. The modular frameworks Social Feed Manager (SFM) [15, 5] and STACKS [16] create collections and schedule harvests. SFM reuses existing social media harvesters and wraps collections in WARC files, preserving harvested metadata while providing a uniform file format across harvested social media data. However, the replay of WARC files harvested in this way is difficult, because the content of the WARC files varies in format, i.e. harvested from different social media providers using different harvesting methods.

Social media can be harvested either by fetching data from Application Programming Interfaces (APIs) or via simulating a web browser. API-based tools, e.g., Twarc[7] for Twitter or Instaloader[8] for Instagram, provide command line interfaces abstracting concrete API requests. They usually provide rich metadata represented as structured data. Tools like

[4] https://github.com/webrecorder/pywb
[5] https://archive.org/
[6] https://github.com/ukwa/w3act
[7] https://github.com/DocNow/twarc
[8] https://github.com/instaloader/instaloader

Brozzler[9] or Webrecorder/Conifer[10] harvest less metadata but preserve the look and feel. They simulate a browser or provide live recording functionality to harvest the HTML-based web version of social media content using the WARC format [12]. The aforementioned frameworks and tools create, describe and harvest social media collections. Technical details of API access are wrapped into user interfaces or command line tools, suitable for GLAM institutions with limited technical skills [9].

Several GLAM-related frameworks concern social media analysis related to social media harvesting, but not necessarily to social media archiving. In the case of ArchivesUnleashed[9], a project aiming to improve scholarly access to web archives, the collection development and harvests are explicitly excluded. Similarly, the GLAM workbench[11] aims for scholarly access by providing Jupyter notebooks[12], a combination of narrative text and live code. Candela et al. [17] investigated a methodology to create reproducible notebooks for the GLAM domain. Such frameworks are more concerned with analysis of already collected/described data and thus are complementary to our solution, i.e. they can be applied on archived data described with our Knowledge Graph.

4.1.2.2 Metadata Standards and Cataloguing

We discuss existing metadata standards and tools to create records adhering to those standards. The Online Computer Library Center (OCLC)[13], a global library cooperative, released recommendations for web archiving metadata fields [18]. They distilled 14 elements from the general vocabularies Dublin Core[14] and Schema.org[15], the XML-based standards Encoded Archival Description (EAD)[3], MARC21[2], and the Metadata Object Description Schema (MODS)[16].

However, the structure in which such elements are used is equally important, several subtly different standards exist. The General International Standard Archival Description (ISAD(G))[17] provides general guidance for the preparation of archival descriptions. EAD is a document-based hierarchical standard used to describe archival records. Although EAD is criticized to be document-centered rather than data-centered[19], hierarchical EAD

[9] https://github.com/internetarchive/brozzler

[10] https://github.com/Rhizome-Conifer/conifer

[11] Tim Sherratt, "GLAM Workbench", http://web.archive.org/web/20220121164421/https://glam-workbench.net/web-archives/ (archived website accessed February 12, 2022)

[12] Jupyter, "Jupyter", http://web.archive.org/web/20220218232731/https://jupyter.org/ (archived website accessed February 12, 2022)

[13] OCLC, "OCLC", http://web.archive.org/web/20220216202514/https://www.oclc.org/en/home.html (archived website accessed February 12, 2022)

[14] DCMI, "Dublin Core Metadata Initiative", https://dublincore.org/ (archived website accessed February 12, 2022)

[15] Schema.org, "Schema.org", http://web.archive.org/web/20220217233411/https://schema.org/ (archived website accessed February 19, 2022)

[16] Library of Congress, "Metadata Object Description Schema", http://www.loc.gov/standards/mods/ (archived website accessed February 12, 2022)

[17] ICA, "ISAD(G)", http://web.archive.org/web/20220120064033/https://www.ica.org/en/isadg-general-international-standard-archival-description-second-edition (archived website accessed February 12, 2022)

records can be used to describe social media collections[18]. Compared to archival standards, MARC21 and MODS are bibliographic standards more focused on the library domain. The Metadata Encoding & Transmission Standard (METS)[19] encodes descriptive, administrative, and structural metadata regarding objects within a digital library, popular to describe elements on an item level[7, 20]. Incorporating all standards in a single model is difficult, as they take different perspectives [21]. Thus, we designed a Knowledge Graph in RDF, generated from heterogeneous born-digital data sources and described using domain-specific vocabularies. This allows generating records of different metadata standards.

Existing tools to generate archival metadata records are usually manual or semi-automatic cataloguing tools, closed source or commercial. According to embedded technical metadata, available EAD records for social media collections[18] are generated from the tool KE EMu[22]. Similarly, the ArchivesHub[20], a portal to integrate collections of several UK archives, uses the commercial software CIIM[21]. Such cataloguing tools are commercial software relying on existing archival records, either created manually or integrated from existing collections, and do not solve the problem of a costly manual creation. In our case, collection information is integrated via open source software from heterogeneous data sources and metadata records are generated automatically. Thus, web archivists are supported by initially generated metadata records to refine if necessary.

4.1.2.3 Knowledge Graph-based solutions

The GLAM domain already recognized Knowledge Graphs as promising future direction [6]. Dedicated ontologies and RDF representations for data models were developed, such as the official RDF ontology for MODS[22] and XSL Stylesheets to transform EAD documents to some RDF representation[23]. However, those RDF representations and ontologies do not describe data and their provenance, but metadata records summarizing data from a specific perspective.

The Europeana Data Model (EDM) [10], developed with technical experts from the GLAM domain, was designed to accommodate different standards. It represents a cultural heritage object together with different representations of it and contextual metadata. ArDO [23] is an ontology for hierarchical multimedia archival records based on specific application requirements and thus not extending EDM, but reusing it as guidance. Hierarchical archival data are also possible metadata records in our case. We use EDM and enrich our

[18] Collection of social media posts from Facebook and Twitter: `https://tiaki.natlib.govt.nz/#details=ecatalogue.1016365` `https://tiaki.natlib.govt.nz/#details=ecatalogue.1016484`

[19] Library of Congress, "Metadata Encoding and Transmission Standard", `http://web.archive.org/web/20220203182758/http://www.loc.gov/standards/mets/` (archived website accessed February 12, 2022)

[20] Jisc, "Archives Hub", `http://web.archive.org/web/20220216165847/https://archiveshub.jisc.ac.uk/` (archived website accessed February 19, 2022)

[21] Knowledge Integration, "CIIM", `http://web.archive.org/web/20220121011751/http://www.k-int.com/products/ciim/` (archived website accessed February 12, 2022)

[22] Library of Congress, "MODS RDF Initiatives", `http://web.archive.org/web/20220120153847/https://www.loc.gov/standards/mods/modsrdf/` (archived website accessed February 12, 2022)

[23] Archives Hub, "EAD to RDF XSLT Stylesheet", `http://web.archive.org/web/20220120123051/http://data.archiveshub.ac.uk/ead2rdf/` (archived website accessed February 12, 2022)

data with other more domain-specific vocabularies, e.g., TweetsKB [24] for social media content, and Dublin Core Collection Description[24] to describe social media collections. The PREMIS Data Dictionary for Preservation Metadata is a standard for which an ontology was developed [25], in version 2.2, meanwhile succeeded by a new ontology version to reflect PREMIS changes of version 3[25]. PREMIS was built on the Open Archival Information System (OAIS) reference model, an ISO standard [26] which among others describes different information packages. We reuse the PREMIS ontology to describe harvested data and its provenance. Similarly to EDM, PREMIS distinguishes between an actual object and its different representations, easing the integration with EDM and the rest of our model.

Regarding archival records, Knowledge Graph solutions are mostly applied on top of existing archival descriptions. Dobreski et al. [7] generate Linked Data for non-textual item-level data, e.g., images, sound, and videos, from XML-based archival records. Hennicke et al. [8] described how existing Bibliopolis and EAD records can be converted to EDM. Although only few Linked Data principles are followed, Gartner [19] devised a solution to represent archival description in a more constrained version of EAD as XML Schema from which regular EAD records can be generated. In contrast to these solutions, we do not generate a Knowledge Graph from existing metadata records and taking their perspective, but integrate raw data into a Knowledge Graph and generate different domain and perspective-specific metadata records in a following step. This way, we avoid the costly manual creation of archival records in the first place, while still providing means to curate data and metadata records.

4.1.3 Comparative Analysis of Social Media Harvesting Tools

Several social media archiving tools exist, varying in supported social media providers, usability and functionality. We compare available open source tools based on features relevant to social media archiving (Table 4.1).

We adapt a framework of the Data Together Initiative[26] originally used to compare generic web harvester tools. We reuse existing columns and add specific columns related to social media archiving in the GLAM domain. We compare the tested tools based on their approach, output format, setup, supported social media providers, configuration, and provenance. All tools but *APIBlender* are still maintained, i.e. commits or pull requests which indicate maintenance.

[24] DCMI, "Dublin Core Collection Description Application Profile", http://web.archive.org/web/20220120141026/https://www.dublincore.org/specifications/dublin-core/collection-description/collection-application-profile/ (archived website accessed February 12, 2022)

[25] Library of Congress, "PREMIS OWL Ontology Version 3", http://web.archive.org/web/20211009123549/http://www.loc.gov/standards/premis/ontology/owl-version3.html (archived website accessed February 12, 2022)

[26] Data Together Initiative, "Comparison of Web Archiving Software", http://web.archive.org/web/20211022023745/https://github.com/datatogether/research/tree/master/web_archiving (archived website accessed February 12, 2022)

Tool	Approach	Output format	Social Media providers			Setup	Config	PROV
			T	F	I			
4CAT	Framework	JSON	+	-	+	advanced	UI	+
APIBlender	Framework	JSON	+	+	-	n/a	file	n/a
Brozzler	Browser	WARC/ HTML	+	+	+	advanced	file	+
Instaloader	API	JSON	-	-	+	beginner	file	+
DMI-TCAT	API	SQL	+	-	-	advanced	file	+
STACKS	Framework	JSON	+	-	-	advanced	file	+
SFM	Framework	WARC/ JSON	+	-	-	advanced	UI	++
Twarc	API	JSON	+	-	-	beginner	file	+
WebRecorder/ Conifer	Browser	WARC/ HTML	+	+	+	advanced	UI	+

Table 4.1: A comparison of features of different social media harvesting tools, T=Twitter, F=Facebook, I=Instagram. Full version available online `https://github.com/RMLio/social-media-archiving`

Approach and output format The approach followed by the tool to harvest social media data and influences the output format: querying data from a single API, simulating a browser, or providing a whole framework. Despite their different approaches, all tools provide interfaces to abstract from the technical aspects of harvesting, and therefore have the potential to suit users in the GLAM-domain.

Different use cases demand different approaches. API-based tools provide machine-readable JSON data and can be used to harvest large amounts of data facilitating further analyses. Even though most JSON harvesting tools store data as files, STACKS stores JSON in a MongoDB and DMI-TCAT in a relational MySQL database. This may increase performance when interacting with the data, but in the case of MySQL also involves yet another data format negatively influencing interoperability. On the other hand, tools simulating a browser store HTML content in WARC containers and thus preserve the look and feel and performed HTTP requests, but usually are slower and may pose more technical challenges compared to API-harvesters as social media content is dynamic [4].

Frameworks provide harvesting functionality for several social media providers and graphical user interfaces, and a promising code base for GLAM institutions. They are usually extensible with own modules or use existing harvesters, e.g. SFM uses Twarc for Twitter harvests. The output format for such frameworks usually depends on the harvesters used, but interestingly, SFM harvests data in JSON format, but preserves it in WARC files [5]. Thus, it provides a uniform interface of harvested social media data across providers while preserving technical metadata which positively influences downstream tasks requiring provenance.

Supported social media providers From which social media providers the tool can harvest data. For this analysis we consider Twitter, Instagram and Facebook as they are part of the long-term goals for our BeSocial use case. Most tools support Twitter, some Instagram and only a few Facebook. Tools harvesting Facebook are simulated browsers, technological challenges for Facebook might be a reason [27] why no tool uses other harvesting means for Facebook. API-based tools are focused on a single provider, frameworks usually support several providers, and tools simulating a browser are technically not limited to any provider as they aim to harvest web content in general. Therefore, either frameworks or simulated browser tools are promising candidates if several social media providers should be supported by the use case.

Setup of the tool We distinguish two levels of difficulty for setting up tools for harvesting: *beginner*, where only a script needs to be installed using a package manager; *advanced*, where several components need to be installed. Most tools can be set up with minimum programming experience, e.g., only by installing one command line tool. The majority of tools requires more steps as they consist of several components. However, such tools usually provide means to compensate, e.g. by providing docker images which, can be started and stopped as containers with minor configuration and a single command, or by providing the harvester as a service. Yet, debugging of such a docker setup, if needed, requires a deeper technical understanding, possibly challenging for users in the GLAM domain.

Tool's configuration How the tools can be used to create social media collections: the more technical abstractions, the better considering less-technical users. All tools are configured via config files or web interfaces, lowering the reuse barrier.

Provenance information Technical metadata captured via the harvesting process and/or descriptive metadata from the harvested content, considering archiving: usually the more the better. In terms of harvested content, tools harvesting data from APIs usually provide rich descriptive metadata facilitating analyses and data stewardship tasks, whereas tools harvesting HTML content in WARC files only provide technical metadata within the WARC HTTP headers. From a collection-level point of view, descriptive metadata in form of collection description needs to be added manually via the configuration of the tools. Regarding provenance information, SFM provides the best trade-off as it preserves technical metadata from harvests within WARC files, descriptive metadata of harvested content as part of the API responses, and descriptive metadata of collections – entered by users via a UI – within a relational database.

Discussion Since frameworks may reuse existing harvesters, they are promising reuse candidates for use cases where several social media providers are considered for archiving. Compared to other frameworks, SFM has the advantage of storing harvested data within WARC files which provides additional provenance information. Additionally, collections in

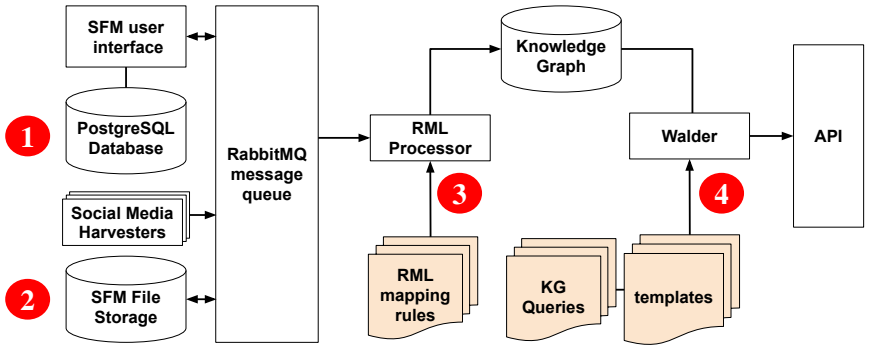


Figure 4.1: Our sustainable social media archiving workflow's architecture is based on open source components and is controlled with only three lightweight and declarative components (orange): RML mapping rules to create Knowledge Graphs, templates to specify metadata records, and queries to populate the templates.

SFM are configured via an user interface which addresses users of the GLAM domain and thus our use case.

4.1.4 Sustainable Workflow

Our workflow reused open-source components to (i) describe social media collections, (ii) harvest social media content, and generate (iii) Knowledge Graphs and (iv) domain-specific metadata records. We present our modular architecture (Figure 4.1) based on open source frameworks in Section 4.1.4.1 and discuss design decisions regarding regarding RDF representations in Section 4.1.4.2.

4.1.4.1 Architecture and Components

Our modular solution integrates into an existing framework and provides three declarative ways to control the social media archiving workflow. We describe the components of our architecture with the following contributions: (i) integration of automatic Knowledge Graph generation into the existing social media harvesting framework SFM, (ii) reusable declarative Knowledge Graph generation rules to describe social media archives, and (iii) reusable declarative queries and templates to generate domain-specific metadata records.

Social media harvesting We reuse the Social Feed Manager (SFM) where a central RabbitMQ message queue is used for communication among components. Archivists create social media collections via a UI where they specify the seeds to harvest, a harvesting schedule, and provenance information regarding the collection (Figure 4.1, 1), i.e. title and description.

At specified intervals a harvesting message is sent to the message queue which triggers existing social media harvesters, e.g., Twarc for Twitter, to fetch data.

SFM supports several API-based harvesters and uses a WARC proxy to preserve technical provenance information by recording performed HTTP requests and store them together with the received HTTP response in WARC files (Figure 4.1, 2). Thus, SFM offers a uniform file format with technical provenance information for differently described social media content from different social media providers. We utilize this uniform format to generate interoperable provenance across social media content. Harvesters indicate the status to the message queue, e.g., a successful harvest with listed information such as the location of newly created WARC files, which we use as input for our Knowledge Graph generation.

Knowledge Graph generation SFM provides a rich source of heterogeneous (meta) data which we lift to a Knowledge Graph to get a uniform and interoperable description of captured and preserved social media. We integrate descriptive collection metadata from SFM and the content harvested, as well as technical metadata produced by SFM and enclosed in preserved WARC files.

We use the RML.io framework[27] (Figure 4.1, 3) to generate the BESOCIAL Knowledge Graph. RML.io generalizes the W3C recommended R2RML specification [28] to integrate heterogeneous data based on declarative mapping rules which is needed for our use case. We use the RMLMapper[28] to generate the Knowledge Graph based on declarative mapping rules following the RML specification.

Metadata records generation Although a Knowledge Graph-based data model enables semantic interoperability of data, concrete preservation systems or other stakeholders in the GLAM domain demand metadata records summarizing certain data in a domain-specific syntax, e.g. MARC21 for libraries, EAD for archives. We provide a component to automatically generate such metadata records from our Knowledge Graph avoiding a costly manual curation. We use Walder[29] which allows setting up a website or API over decentralized knowledge graphs. Using existing template libraries from web development, e.g., Handlebars[30], templates for metadata records are created. The query language GraphQL-LD [29] is used to query the Knowledge Graph and populate declarative templates with content, generating metadata records published via an API using Walder (Figure 4.1, 4), while avoiding needing in-depth programming experience.

[27] rml.io, "RML", `http://web.archive.org/web/20220205092255/https://rml.io/` (archived website accessed February 12, 2022)

[28] `https://github.com/RMLio/rmlmapper-java`

[29] `https://github.com/KNowledgeOnWebScale/walder`

[30] Yehuda Katz, "handlebars", `http://web.archive.org/web/20220214195001/https://handlebarsjs.com/` (archived website accessed February 19, 2022)

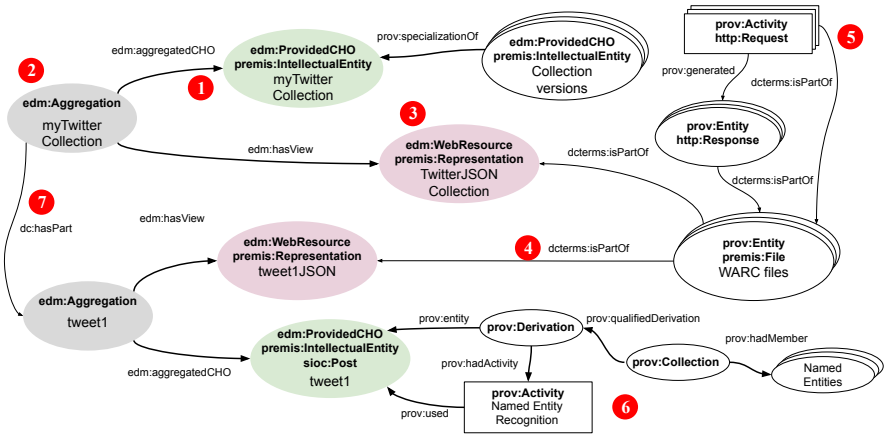


Figure 4.2: The Europeana Data Model (EDM) is used to represent social media collections and posts as cultural heritage objects (green) and their different representations (violet), aligned with PREMIS and PROV to represent provenance.

4.1.4.2 Data-driven Workflow

We describe how the Europeana Data Model (EDM), the de-facto standard for cultural heritage data, and other common W3C recommended vocabularies can be used to represent social media collections in an interoperable way.

We followed a Competency Question (CQ)-based approach, commonly used to express requirements in ontology engineering [30]. We defined more than 20 CQs for our archival use case based on user-stories to determine which data needs to be integrated. A full list is openly available at our online resource.

We reuse the EDM to describe harvested social media content because it enables us to represent not only the object itself, e.g. a Tweet via its ID, but also differently harvested representations, e.g. captured JSON or HTML representations of a Tweet stored in WARC files. A whole collection, created by users via SFM and stored in a relational database, and social media posts (items of the collection) are represented as cultural heritage objects using the class `edm:ProvidedCHO` and `premis:IntellectualEntity` (Figure 4.2, 1). Such a collection or item may have different representations linked by an instance of `edm:Aggregation` (Figure 4.2, 2), in our case the harvesters used by SFM fetch information in JSON from APIs, and thus we use `edm:WebResource` and `premis:Representation` to represent a JSON representation (Figure 4.2, 3); someone may harvest social media posts (additionally) in their HTML representation which would then be another `edm:WebResource`, linked to the associated aggregation (Figure 4.2, 2). To increase interoperability we represent social media posts also using `sioc:Post` from TweetsKB [24].

Harvested social media data is enclosed in WARC files by SFM (Figure 4.2, 4) preserving

harvest metadata of HTTP requests. We represent such harvest metadata using PROV activities, listing when and how WARC files were created (Figure 4.2, 5), WARC files are represented using `premis:File`. On item level, we perform Named Entity Recognition (NER) during mapping via the DBpedia spotlight API[31] to enrich our Knowledge Graph (Figure 4.2, 6). This information is useful later when generating archival records. PROV is used to preserve information of the NER process. Hierarchical information, such as which item belongs to which collection, is explicitly represented using Dublin Core and following EDM guidelines[32] (Figure 4.2, 7).

4.1.5 Social Media Archiving at KBR

BESOCIAL is a cross-institutional research project, aiming to develop a sustainable strategy for archiving and preserving social media in Belgium. The solution supports this goal by offering a sustainable social media archiving workflow. We outline the use case and describe how we applied our workflow within a pilot.

BESOCIAL use case KBR, as the federal scientific library of Belgium, is legally mandated to collect and preserve all Belgian publications. To tackle challenges of the digital-era, KBR invests in the digital preservation of online content. In the past KBR worked on a federal strategy for the preservation of the Belgian Web [4]. Due to the uniqueness and ephemeral nature of social media, BESOCIAL brings together interdisciplinary partners to consider conservation, preservation and accessibility of developing a social media archive.[33] Twitter was selected as promising social media platform, but Instagram and Facebook are considered in the long-term. Recent outcomes of BESOCIAL are the analysis of an online survey in which 15 international archiving institutions participated, and which showed that many institutions are engaged in social media archiving, but also that the stage and efforts vary in size and scope [27].

Content selection Web archivists define so-called seed lists with content that should be archived. For BESOCIAL, a seed list with 86 relevant Belgian entities of 14 categories, such as governmental institutions and online news, was curated by KBR for a test pilot. From these 86 entities, 79 had accounts on Twitter. We used the user interface of SFM to create a collection for these accounts.

Content collection Collections created with the user interface of SFM were scheduled to harvest social media data daily. This, so far, resulted in 50 compressed WARC files of

[31] DBpedia spotlight API: `https://www.dbpedia-spotlight.org/api`

[32] Europeana, "Europeana Data Model - Mapping Guidelines v2.4", `http://web.archive.org/web/20220122194051/https://pro.europeana.eu/files/Europeana_Professional/Share_your_data/Technical_requirements/EDM_Documentation/EDM_Mapping_Guidelines_v2.4_102017.pdf` (archived website accessed February 12, 2022)

[33] Royal Library of Belgium (KBR), "BESOCIAL", `http://web.archive.org/web/20220131085437/https://www.kbr.be/en/projects/besocial/` (archived website accessed February 12, 2022)

88 MB enclosing around 200,000 Tweets in JSON format. The first harvest resulted in roughly 150,000 tweets as the used Twarc harvester of SFM fetches the most recent 3,200 tweets per account. Subsequent daily harvests resulted in less content of up to 2,000 tweets. These are heterogeneous data which we need to lift to a Knowledge Graph to facilitate data stewardship tasks.

Knowledge Graph generation We used the data model and its requirements expressed as Competency Questions (CQs) described in Section 4.1.4.2 to systematically guide the integration process, i.e. one RML mapping contributes data to answer at least one CQ. Applying these mappings resulted in one RDF file per WARC file and one RDF file for collection-level metadata extracted from the SFM PostgreSQL database. We generated RDF triples consisting among others of 213,000 EDM cultural heritage object resources representing collections and social media posts, and 222,000 W3C PROV activities reflecting provenance.

Metadata records generation Different domain-specific data formats exist. Already available social media collections are described using EAD records[18], thus we consider this a baseline, and KBR as a library works with bibliographic MARC-based records to describe collections. Additionally, human users may want to browse collections. Thus, we created two XML-based and one HTML-based template and related GraphQL queries for Walder to populate these templates from our Knowledge Graph to accommodate these use cases; available at our online resource[34]. We can query heterogeneous data, to among others, get aggregated information about named entities, enabling users to assess the content i.e. which locations or events are mentioned within a whole social media collection. Hierarchical information is present in our Knowledge Graph as we reused terms like `dc:hasPart` (Figure 4.2, 7).

Discussion We discuss the added value of the Knowledge Graph in our use case and findings related to the Knowledge Graph's use with respect to collection-level and item-level (social media post) data.

Instead of many-to-many mappings from heterogeneous data sources to heterogeneous metadata records, our solution results in a semantically described RDF Knowledge Graph which facilitates data stewardship as it describes all preserved data including provenance information. The generation of metadata records and HTML views are thus not limited to harvested data, but also profit from contextual information of the Knowledge Graph, because item-level data (social media posts) are put in relationship to collections and provenance information. This information can be queried using SPARQL or GraphQL, therefore we are able to identify e.g. social media posts belonging to different collections or collections/posts mentioning similar named entities. Similarly, more fine-grained queries are possible with more integrated linked data in the future, i.e. archivists may rather spend manual curation efforts in enriching the Knowledge Graph instead of domain-specific metadata records.

[34] `https://github.com/RMLio/social-media-archiving`

Use cases related to the collection-level may not need the full graph. Whereas harvested data preserved and compressed in WARC files are relatively small, the Knowledge Graph is considerably larger. This may present a performance bottleneck for smaller setups without adequate RDF database or hardware. However, HTML views providing an overview of collections, or MARC records describing bibliographic information of collections do not need all item-level details such as detailed post provenance. We used decentralized Knowledge Graphs partitioned between collection and item level data to improve performance of collection-level tasks.

If certain use cases demand some item-level information we declaratively create aggregations. Based on the data model and extracted information, we used SPARQL-CONSTRUCT queries to enrich collection-level information with aggregated information from item-level, such as most often used named entities and their type; vocabularies such as the W3C recommended WebAnnotations[35] or DataCube[36] may be used to semantically describe aggregates, further research is required.

Libraries usually provide full access to collections only via reading rooms or after login, and from a legal perspective it is also problematic to provide public access to harvested social media data. However, collection-level related parts of the Knowledge Graph including aggregations present a smaller sub-graph which may be made publicly available, directly as API or via HTML views. Therefore, end users may assess more detailed information about collections using contextual-rich collection information before requesting access to the full collection on-premise or online which could positively influence the user experience. However, more research towards the needs of different types of users is needed.

4.1.6 Conclusion

Social media is already a paramount part of our society and, thus, its content needs to be preserved. However, archiving is an expensive long-term commitment and currently only fragmented workflows for social media archiving exists. We developed an open source Knowledge Graph-based solution using the Europeana Data Model and PREMIS to describe WARC-preserved social media as cultural heritage objects with different representations. Now we can support automatic generation of GLAM-related metadata records, e.g., MARC and EAD, or provide collection overviews via HTML for users to assess the collections' content.

Human-in-the-loop provenance Social media harvesting tools play a crucial role regarding provenance information, as they cover initial phases of selection and collection where human users define what to harvest and when. Currently SFM provides a detailed change history of collections, but descriptive information is limited to titles and descriptions. Similar to how some web archiving tools require the upload of legal deposit documents before

[35] Robert Sandersion et al., "Web Annotation Data Model", http://web.archive.org/web/20220208094058/https://www.w3.org/TR/annotation-model/ (archived website accessed February 12, 2022)

[36] Richard Cyganiak et al., "The RDF Data Cube Vocabulary", http://web.archive.org/web/20220210140303/https://www.w3.org/TR/vocab-data-cube/ (archived website accessed February 12, 2022)

harvests are initiated [14], SFM could be extended with UI fields to collect specific information from users in a uniform fashion. Our Knowledge Graph-based solution allows a data-centric perspective driven by downstream tasks which can inform improvements of SFM's UI and database, to include more, and more-specific metadata fields which would positively influence the quality of generated metadata records.

Data stewardship of digital collections Social media archives are not static and pose new challenges for which data stewardship is needed: some content may have to be removed from public access due to intellectual property or privacy-related take-down requests, and on top of that several terms of services from different social media providers need to be taken into account. Such stewardship tasks are supported by our solution. For example, our Knowledge Graph already encodes provenance information of harvesting, and as it is based on PREMIS and W3C PROV, existing data can be annotated or additional provenance information regarding take-down requests can be included in the same fashion. Therefore, consuming applications can perform policy-compliant operations with the harvested data.

Future Work Future work will investigate the quality of generated metadata records and extend the metadata record queries if necessary. The modular tool SFM can be extended with new functionality or other social media harvesters. Based on our Knowledge Graph, operational and legal challenges of social media archiving can be reconsidered and addressed.

4.2 Declarative Data Quality Assessment for Social Media Archiving

Sven Lieber, Pieter Heyvaert, Anastasia Dimou

Abstract

Social media as infrastructures of public discourse provide valuable information that needs to be preserved, hence we recently created a Knowledge Graph to describe social media archives. The data quality of such a Knowledge Graph needs to be assessed, but currently no social media-specific data quality metrics exist and furthermore, existing solutions rely on custom software. We assess application-specific data quality to satisfy user needs in the framework of the BESOCIAL project at the Royal Library of Belgium (KBR). In this paper, we apply an existing methodology to systematically define data quality for social media archives, and we declaratively assess the quality using the Data Quality Vocabulary (DQV), the W3C recommended SHACL, as well as SPARQL queries. Quality measurements provided detailed descriptions of quality issues, most of which we could fix during Knowledge Graph generation. Furthermore, our quality assessment informs changes in used software to enforce needed quality already at the source. Lessons learned comprise guidelines in the use of SHACL for data quality assessment, as well as practical issues we encountered. In the future we plan to add more quality requirements from different types of users.

4.2.1 Introduction

Social media platforms have become social infrastructures for public discourse[1, 2], yet social media data is centrally maintained by profit-based social media providers which motivates preservation of social media collections by third parties. The quality of such collections determine which services can be built upon them [31].

In the domain of Galleries, Libraries, Archives, and Museums (GLAM) several tools for social media archiving were developed, e.g., Twarc[37], Brozzler[38], or Instaloader[39] to harvest social media, and tools for analysis, e.g., ArchivesUnleashed [9] or the GLAM workbench[40] We recently presented a Knowledge Graph-based solution to offer an end-to-end preservation workflow which includes both harvested data and collection-related provenance [32].

However, the quality of collections' metadata is not yet considered, but is important with respect to provenance [4, 5]. User interfaces which enable exploration of social media collections may rely on particular shapes of data [33]. For example, a dashboard providing details of collections to users relies on use-case specific data qualities, e.g., temporal information about posts in the collection.

Existing data quality methodologies and metrics are generic and do not provide use-case specific assessment or guidance, needed for our social media archiving use case. Generic metrics were proposed [34] and measured by existing tools [35, 36, 37, 38]. However, a predefined selection of use case-specific metrics, e.g, for web or social media archiving, is more practical than standard quality metrics [39]. New metrics can be defined with a high-level methodology [40] and represented with existing vocabularies [41, 39], but this was not done so far for social media archiving.

More, existing frameworks for RDF data quality rely on custom solutions [37, 33], suboptimal for interoperability in case not all features of the tools are needed. The use of declarative constraint languages, e.g., SHACL [42] or ShEx [43], was only outlined or vaguely mentioned [41, 33] but concrete examples and lessons learned are missing. Yet using such languages creates opportunities to involve domain experts in defining constraints as text editor or graphical means [44, 45] can be used to create constraints.

We define declarative data quality-related constraints on our Knowledge Graph to increase quality and in turn support the findability and accessibility of social media archives, i.e. contextual rich views on our collections enabled by consistent data quality. We built on existing work regarding data quality methodologies and discuss lessons learned regarding application-specific constraints in social media archiving expressed with SHACL. Our findings can be considered for other use cases besides social media archiving.

Our contributions are: (i) social media quality categories, dimensions and metrics defined using a FAIR template in RDF [46] **available at: `https://doi.org/10.6084/m9.figshare.16655239.v1`**; (ii) an approach for quality assessment using the Data Quality Vocabulary

[37] `https://github.com/DocNow/twarc`

[38] `https://github.com/internetarchive/brozzler`

[39] `https://github.com/instaloader/instaloader`

[40] Tim Sherratt, "GLAM Workbench", `http://web.archive.org/web/20220121164421/https://glam-workbench.net/web-archives/` (archived website accessed February 12, 2022)

(DQV), SHACL and SPARQL queries[41]; (iii) discussion of challenges and lessons learned related to data quality assessments on RDF data with SHACL and DQV.

The paper is organized as follows: Section 4.2.2 covers related work, Section 4.2.3 explains our social media archiving workflow, Section 4.2.4 presents the methodology for assessing the data quality of social media archives, Section 4.2.5 discusses lessons learned and future work.

4.2.2 Background and Related Work

Our work covers data quality assessment in Knowledge Graph-based social media archives. Therefore, web and social media archiving, data quality in web archiving as well as data quality assessment for RDF Knowledge Graphs are relevant.

The archiving of web content is based on crawling content based on seeds which are harvested from the live web and are preserved as files. Research regarding quality for archiving of web content focuses on the content, i.e. defining seeds to be archived [47], or the outcome of the crawling process itself [48, 49] because websites may not be captured in a correct fashion. Social media is often harvested from Application Programming Interfaces (APIs) due to its dynamic content [4] with tools such as **Twarc**[37] or **Instaloader**[39]. Therefore the quality of the capture is usually higher compared to general web archiving as descriptively rich (meta) data are harvested.

Data quality is fitness-for-purpose [50] and can be achieved by defining and assessing quality dimensions using quality metrics. **Zaveri et al.** [34] identified general data quality dimensions and metrics. These can be measured using tools such as **Luzzu** [37] or **Loupe** [51] which use custom quality definition languages and produce non-interoperable data quality reports, or **RDFUnit** [36] which produces data quality reports described using the **Data Quality Vocabulary (DQV)** [41]. **Rula and Zaveri** [40] proposed a methodology to describe data quality dimensions and metrics. Such quality dimensions and metrics can be described with the above mentioned DQV or the extension from **Langer et al.** [39] which hints toward the use of SHACL but did not fully investigate its use.

Our Knowledge Graph reuses existing vocabularies such as the **Europeana Data Model (EDM)** [10] for which a textual description of integrity constraints is available in natural language within the EDM mapping guidelines[42] A few years ago those constraints were formalized using a preliminary version of SHACL[43], however, these shapes do not comply with the finalized version of W3C SHACL. Europeana constraints in the form of SHACL would be valuable to assess the data quality, thus we adapted those shapes to current W3C

[41] https://github.com/RMLio/social-media-archiving/tree/master/data-quality

[42] Europeana, "Europeana Data Model - Mapping Guidelines v2.4", http://web.archive.org/web/20220122194051/https://pro.europeana.eu/files/Europeana_Professional/Share_your_data/Technical_requirements/EDM_Documentation/EDM_Mapping_Guidelines_v2.4_102017.pdf (archived website accessed February 12, 2022)

[43] https://github.com/hugomanguinhas/europeana_shapes

SHACL[44]. Recently, **Čerāns et al.** [52] generated enriched SHACL shapes from Europeana data. However, their solution aims for visual querying and their shapes describe the existing data and not quality-related constraints for validation. **Király and Büchler** [31] presented a data quality framework for EDM which is able to measure generic data quality metrics. However, an interoperable representation of constraints using SHACL and data quality reports using DQV is future work and the presented metrics do not cover our application-specific needs entirely. A similar work, the scalable open data management platform **Piveau** [53], measures data quality metrics and among others uses SHACL and DQV, however, they focus on generic metrics for open data datasets and not on application-specific social media collections.

4.2.3 BESOCIAL Workflow

Social media collections are created by users using a UI, are harvested in specified intervals by harvesting tools and are mapped to a Knowledge Graph for data stewardship and access by a dashboard. We briefly introduce data quality-relevant details of our workflow [32]: (i) Collection creation via a UI, (ii) harvest of social media content via APIs, (iii) a declarative two-phase Knowledge Graph generation, (iv) access to preserved social media content.

Collection creation With the user interface of the Social Feed Manager (SFM) [5], archivists create collections by specifying collection metadata, e.g., title and description, and harvesting-related information, e.g., API credentials and schedule. This information is stored by SFM in a relational database. User-provided metadata of this step influences data quality: the more descriptive metadata, the better for humans, and the more structured, the better for machines.

Collection harvest Existing harvesters integrated in SFM, e.g., Twarc[37], are executed in certain intervals and store harvested content as files on disk. SFM-integrated harvesters usually query the APIs of social media providers and receive descriptively rich information in e.g., JSON, stored within WARC files [12] which compresses the content with technical metadata related to the performed HTTP request. Provenance information of this harvesting process is valuable for users [5], positively influencing data quality.

Knowledge Graph generation We generate a Knowledge Graph in two phases: (i) we map heterogeneous data with RML [54], i.e. collection-level data from SFMs relational database and item-level data from harvested WARC files, to a Knowledge Graph; (ii) we use SPARQL INSERT to generate aggregated RDF data from item-level to ease access and comply with operational and legal objectives. This mapping is automatically triggered by the SFM framework and uses vocabularies such as the Europeana Data Model (EDM) [10] and

[44] https://github.com/RMLio/social-media-archiving/tree/master/data-quality/shapes/europeana-shapes

Dublin Core[45]. Correct use of existing data models and mapping all required use case-specific data is important for data quality.

Access via a dashboard and exports We provide collection-level access via a web API generated by the tool Walder[46]. This web API is built by querying our Knowledge Graph to populate views, and thus provides HTML views of the collections as well as domain-specific MARC and EAD[47] XML records. We consider this dashboard one of the main sources for data quality requirements, because it specifies needed functionality the data has to support.

4.2.4 Social Media Archive Quality

Considering our use case of social media collections from heterogeneous social media data and our goal to manage these collections and ensure quality for access, we apply the data quality assessment methodology of Rula and Zaveri [40] and adapt where necessary.

4.2.4.1 Phase 1 - Requirements Analysis

This phase aims to gather requirements from the use case [40]. It consists of the single step *use case analysis* for which we consider user stories describing needs from users' perspective, a common technique also used in ontology engineering [55].

We created 40 user stories[48] (Figure 4.3, 1) from which we consider 20 in particular relevant for data quality in our use case, because they refer to the different views on the social media archive. We defined these user stories using CSV as this is a lightweight format, and then derived quality-related requirements.

4.2.4.2 Phase 2- Data Quality Assessment

This phase involves the quality assessment based on the previously identified requirements [40].

Identification of quality issues This step aims to identify a set of data quality issues based on the use case [40]. We suggest deriving data quality requirements, dimensions and metrics from the previous step's user stories. Therefore, the creation of data quality dimensions and metrics follows a similar strategy as the creation of data model requirements, i.e. building upon user stories and deriving quality dimensions for data quality and competency questions for the data model [55].

We derive metrics from quality requirements by using the FAIR metrics template from Wilkinson et al. [46] to systematically describe the metrics and the DQV vocabulary [41] to

[45] DCMI, "Dublin Core Metadata Initiative", `https://dublincore.org/` (archived website accessed February 12, 2022)

[46] `https://github.com/KNowledgeOnWebScale/walder`

[47] Library of Congress, "Encoded Archival Description", `https://www.loc.gov/ead/` (archived website accessed February 12, 2022)

[48] `https://github.com/RMLio/social-media-archiving/blob/master/data-model/user-stories.csv`

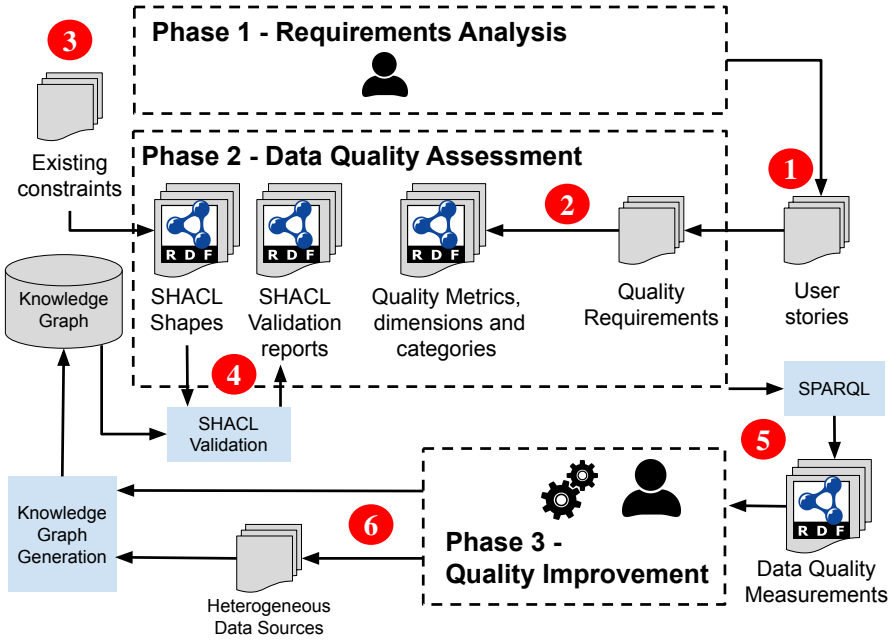


Figure 4.3: Apply the methodology of [40] for declarative quality assessment of social media archives with SHACL.

represent data quality categories, dimensions and metrics. Similar to user stories, we represent these quality elements in CSV format and declaratively create their DQV RDF representation via RML (Figure 4.3, 2). We created valid SHACL shapes from legacy Europeana SHACL shapes[43]. Additionally, we generated SHACL shapes from used ontologies with Astrea [56] and refined where necessary. We define boolean metrics to indicate quality for a single collection, e.g. "is a description available?" and related integer metrics to indicate the same across collections, e.g. "the number of collections without a description". We link each metric to its related requirement and dimension. Then we defined reusable SHACL shapes for metrics, i.e. a SHACL shape to validate if a description is available is used both for a boolean and integer metric. Depending on the metrics' expected datatype only the query to create a quality measurement based on the SHACL validation report is different, i.e. select either violations of a type per collection, or the sum of violations of a type across collections (Figure 4.3, 5).

Analysis Rula and Zaveri [40] define the step *statistical and low-level analysis* to automatically compute scores indicating the value of each assessed data quality, and the step *advanced analysis* to perform the data quality assessment. In our methodology, a SHACL

validation is used to identify issues (Figure 4.3, 4), and SPARQL queries on the SHACL validation report and the defined DQV quality metrics are used to create an assessment report (Figure 4.3, 5).

We use the SHACL processor integrated in the free version of the Stardog triple store[49]. because it supports all SHACL core constraints and can be executed directly on the data. We create DQV quality measurements by querying our DQV quality information and results of the SHACL validation.

4.2.4.3 Phase 3 - Quality Improvement

This phase builds upon the results from the previous analysis and aims to improve the quality of a dataset with respect to the quality dimensions from the first phase [40].

Root cause analysis This step finds explanations of detected quality issues and involves determining if the data quality issues occur in the assessed or original dataset [40]. Our workflow involves heterogeneous data sources and Knowledge Graph generation in two declarative phases. Based on information queried from the SHACL validation report and the defined DQV quality metrics, we found that 10 from 12 data quality metrics report issues for some or most of our social media collections (Figure 4.3, 5), see Table 4.2.

The root cause of the problems can be identified by analyzing the query results, i.e. look at which SHACL shapes, linked to which quality metric, report violations for which collections. We found that problems relate for 4 metrics in an insufficient Knowledge Graph generation, for 4 metrics in an insufficient quality of the heterogeneous data sources, and in 2 cases in too strict metrics. An overview of found problems and root causes is shown in Table 4.2

The Knowledge Graph generation did miss to generate a few important properties, for example linking collection seeds to collections. Source data was of insufficient quality too, for some collections either no harvests where yet executed (missing harvest start and end date) or harvests were executed but no social media posts were added during the harvests (missing content start and end date). Furthermore, 2 metrics seem to be too strict for our use case: each collection should report the top 10 used hashtags and top 10 used named entities, but some collections were so small that less than 10 unique hashtags or named entities were found.

Fixing quality problems This step addresses the identified root causes of data quality issues, where issues can be resolved both semi-automatic or manual [40]. We can resolve almost half of data quality issues by adapting the Knowledge Graph generation, informed by the extensive DQV and SHACL-based data quality reports. As shown by Dimou et al. [38], this is a cost efficient process because quality issues are resolved during generation and not on a considerably large set of instance data. Additionally, the 2 issues regarding too strict

[49] Stardog, "The Enterprise Knowledge Graph Platform", `https://web.archive.org/web/20220202104647/https://www.stardog.com/` (archived website accessed February 12, 2022)

Quality metric	Affected collections	Problem source
Number of missing collection version harvests	100%	KG generation
Number of missing collection harvests	100%	KG generation
Number of missing collection seeds	100%	KG generation
Number of missing collection version info	100%	KG generation
Number of missing collection hashtags	72%	Too strict metric
Number of missing start and end dates content	67%	Source quality
Number of missing top 10 named entities	67%	Too strict metric
Number of insufficient descriptions	44%	Source quality
Number of missing descriptions	33%	Source quality
Number of missing start and end dates harvest	33%	Source quality

Table 4.2: An overview of found quality issues and root cause.

metrics can be resolved by refining the quality metrics and adapt them to our use case, i.e. a smaller n for top-n hashtags and named entities.

4.2.5 Discussion and Future Work

We discuss lessons learned which motivate future work.

Social media collections With the performed data quality assessment, we could fix most quality issues during Knowledge Graph generation by adding missing properties, and thus avoiding cumbersome repairs in instance data. A few issues, e.g., collections with a description of less than 200 characters can be addressed by adapting the UI of our harvesting tool Social Feed Manager [5]. This shows how a quality assessment based on user needs during access via UI can inform improvements of initial harvesting tools.

SHACL guidelines for quality assessment Currently, there are no guidelines in how to use SHACL, but for the creation of data quality assessment results, it is necessary to create SHACL shapes in a certain structure. We assigned SHACL property shapes to one or more DQV quality metrics such that querying a SHACL validation report yields in a direct measurement of metrics. However, all data reachable via property paths need to be available in the same database as there is no federated validation for shapes. This exemplifies the need for guidelines in using SHACL in different scenarios, reusable SHACL patterns in the form of application profiles might be a solution, similarly to how Ontology Design Patterns [57] provide reusable ontology patterns, however this requires more research. Future work can also investigate the use of ShEx [43].

Technology readiness level Surprisingly, most open source SHACL processors do not support validation on triple stores, which makes validation in real life scenarios challenging. RDFUnit [36] does support it, but `sh:qualifiedValueShape` costraints are not supported which we use. Other implementations supporting triple stores focusing on recursive SHACL [58] or performant processing [59] but support even less SHACL core constraints and additionally need SHACL constraints in a custom format. Such challenges could indicate why quality assessments using SHACL are less common compared to custom tools which scale. However, the mentioned works can be extended to foster further adoption of RDF Knowledge Graphs outside academia by providing mature quality assessment.

References

[1] Amelia Acker and Adam Kreisberg. "Social Media Data Archives in an API-driven World". In: *Archival Science* 20.2 (2020), pp. 105–123.

[2] Elisabeth Fondren and Meghan Menard McCune. "Archiving and Preserving Social Media at the Library of Congress: Institutional and Cultural Challenges to Build a Twitter Archive". In: *Preservation, Digital Technology & Culture* 47.2 (2018), pp. 33–44.

[3] Christine L Borgman. *Scholarship in the digital age: Information, infrastructure, and the Internet*. MIT press, 2010.

[4] Eveline Vlassenroot, Sally Chambers, Emmanuel Di Pretoro, Friedel Geeraert, Gerald Haesendonck, Alejandra Michel, and Peter Mechant. "Web archives as a data resource for digital scholars". In: *International Journal of Digital Humanities* 1.1 (2019), pp. 85–111.

[5] Justin Littman, Daniel Chudnov, Daniel Kerchner, Christie Peterson, Yecheng Tan, Rachel Trent, Rajat Vij, and Laura Wrubel. "API-based social media collecting as a form of web archiving". In: *International Journal on Digital Libraries* 19.1 (2018), pp. 21–38.

[6] Greta Bahnemann, Michael Carroll, Paul Clough, Mario Einaudi, Chatham Ewing, Jeff Mixter, Jason Roy, Holly Tomren, Bruce Washburn, and Elliot Williams. "Transforming metadata into linked data to improve digital collection discoverability". In: (2021).

[7] Brian Dobreski, Jaihyun Park, Alicia Leathers, and Jian Qin. "Remodeling archival metadata descriptions for linked archives". In: *International Conference on Dublin Core and Metadata Applications*. 2020, pp. 1–11.

[8] Steffen Hennicke, Marlies Olensky, Viktor de Boer, Antoine Isaac, and Jan Wielemaker. "A data model for cross-domain data representation". In: *Proceedings of the 12th International Symposium on Information Science*. 2011, pp. 136–147.

[9] Nick Ruest, Jimmy Lin, Ian Milligan, and Samantha Fritz. "The Archives Unleashed Project: Technology, Process, and Community to Improve Scholarly Access to Web Archives". In: *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*. 2020, pp. 157–166.

[10] Martin Doerr, Stefan Gradmann, Steffen Hennicke, Antoine Isaac, Carlo Meghini, and Herbert Van de Sompel. "The Europeana Data Model (EDM)". In: *World Library and Information Congress: 76th IFLA general conference and assembly*. Vol. 10. 2010, p. 15.

[11] Gordon Mohr, Michael Stack, Igor Rnitovic, Dan Avery, and Michele Kimpton. "Introduction to Heritrix". In: *4th International Web Archiving Workshop*. 2004, pp. 109–115.

[12] ISO Central Secretary. *ISO 28500:2017 Information and documentation — WARC file format*. en. Standard ISO 28500:2017. Geneva, CH: International Organization for Standardization, 2017. URL: https://www.iso.org/standard/68004.html.

[13] Brad Tofel. "'Wayback' for Accessing Web Archives". In: *Proceedings of the 7th International Web Archiving Workshop*. 2007, pp. 27–37.

[14] Gordon Paynter, Susana Joe, Vanita Lala, and Gillian Lee. "A year of Selective Web Archiving with the Web Curator at the National Library of New Zealand". In: *D-Lib Magazine* 14.5/6 (2008), pp. 1082–9873. URL: http://www.dlib.org/dlib/may08/paynter/05paynter.html.

[15] George Washington University Libraries. *Social Feed Manager. Version 2.3.0*. Version 2.3.0. May 2020. DOI: 10.5281/zenodo.3784836. URL: https://doi.org/10.5281/zenodo.3784836.

[16] Jeff Hemsley, Sam Jackson, Sikana Tanupabrungsun, and Billy Ceskavich. *STACKS - Social Media Tracker, Analyzer, & Collector Toolkit at Syracuse*. Version 3.1. Apr. 2019. DOI: 10.5281/zenodo.2638848. URL: https://doi.org/10.5281/zenodo.2638848.

[17] Gustavo Candela, María Dolores Sáez, MPilar Escobar Esteban, and Manuel Marco-Such. "Reusing digital collections from GLAM institutions". In: *Journal of Information Science* (2020).

[18] Jackie M Dooley and Kate Bowers. *Descriptive Metadata for Web Archiving: Recommendations of the OCLC Research Library Partnership Web Archiving Metadata Working Group*. OCLC Research, 2018.

[19] Richard Gartner. "An XML schema for enhancing the semantic interoperability of archival description". In: *Archival Science* 15.3 (2015), pp. 295–313.

[20] Mary W Elings and Günter Waibel. "Metadata for all: Descriptive standards and metadata sharing across libraries, archives and museums". In: *First Monday* (2007).

[21] Richard Gartner and Raphaële Mouren. "Archives, museums and libraries: breaking the metadata silos". In: *Paper presented at IFLA WLIC 2019*. Athens, Greece, 2019.

[22] María Consuelo Sendino. "KE EMu and the future for natural history collections". In: *Collections* 5.2 (2009), pp. 149–158.

[23] Oleksandra Vsesviatska, Tabea Tietz, Fabian Hoppe, Mirjam Sprau, Nils Meyer, Danilo Dessì, and Harald Sack. "ArDO: An Ontology to Describe the Dynamics of Multimedia Archival Records". In: *Proceedings of the 36th Annual ACM Symposium on Applied Computing*. SAC '21. Virtual Event, Republic of Korea: Association for Computing Machinery, 2021, pp. 1855–1863. ISBN: 9781450381048. DOI: 10.1145/3412841.3442057. URL: https://doi.org/10.1145/3412841.3442057.

[24] Pavlos Fafalios, Vasileios Iosifidis, Eirini Ntoutsi, and Stefan Dietze. "TweetsKB: A Public and Large-Scale RDF Corpus of Annotated Tweets". In: *European Semantic Web Conference*. Springer. 2018, pp. 177–190.

[25] Sam Coppens, Ruben Verborgh, Sébastien Peyrard, Kevin Ford, Tom Creighton, Rebecca Guenther, Erik Mannens, and Rik Van de Walle. "PREMIS OWL". In: *International Journal on Digital Libraries* 15.2 (2015), pp. 87–101. DOI: 10.1007/s00799-014-0136-9. URL: http://link.springer.com/article/10.1007%5C%2Fs00799-014-0136-9.

[26] ISO Central Secretary. *ISO 14721:2012 Space data and information transfer systems*. en. Standard ISO 14721:2012. Geneva, CH: International Organization for Standardization, 2012. URL: https://www.iso.org/standard/57284.html.

[27] Eveline Vlassenroot, Sally Chambers, Sven Lieber, Alejandra Michel, Friedel Geeraert, Jessica Pranger, Julie Birkholz, and Peter Mechant. "Web-archiving and social media: an exploratory analysis". In: *International Journal of Digital Humanities* (2021), pp. 1–22.

[28] Souripriya Das, Seema Sundara, and Richard Cyganiak. *R2RML: RDB to RDF Mapping Language*. Working Group Recommendation. World Wide Web Consortium (W3C), Sept. 2012. URL: http://www.w3.org/TR/r2rml/.

[29] Ruben Taelman, Miel Vander Sande, and Ruben Verborgh. "GraphQL-LD: Linked Data Querying with GraphQL". In: 2018. URL: https://comunica.github.io/Article-ISWC2018-Demo-GraphQlLD/.

[30] Michael Grüninger and Mark S Fox. "The Role of Competency Questions in Enterprise Engineering". In: *Benchmarking—Theory and practice*. Springer, 1995, pp. 22–31.

[31] Péter Király and Marco Büchler. "Measuring Completeness as Metadata Quality Metric in Europeana". In: *2018 IEEE International Conference on Big Data (Big Data)*. IEEE. 2018, pp. 2711–2720.

[32] Sven Lieber, Dylan Van Assche, Sally Chambers, Fien Messens, Friedel Geeraert, Julie M Birkholz, and Anastasia Dimou. "BESOCIAL: A Sustainable Knowledge Graph-Based Workflow for Social Media Archiving". In: *Further with Knowledge Graphs*. IOS Press, 2021, pp. 198–212.

[33] Nandana Mihindukulasooriya, Giuseppe Rizzo, Raphaël Troncy, Oscar Corcho, and Raúl García-Castro. "A Two-Fold Quality Assurance Approach for Dynamic Knowledge Bases: The 3cixty Use Case." In: *(KNOW@ LOD/CoDeS)@ ESWC*. 2016.

[34] Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Sören Auer. "Quality assessment for linked data: A survey". In: *Semantic Web Journal* 7.1 (Mar. 2015), pp. 63–93. DOI: 10.3233/SW-150175. URL: http://www.semantic-web-journal.net/system/files/swj556.pdf.

[35] André Langer, Valentin Siegert, Christoph Göpfert, and Martin Gaedke. "SemQuire - Assessing the Data Quality of Linked Open Data Sources Based on DQV". In: *International Conference on Web Engineering*. Springer. 2018, pp. 163–175.

[36] Dimitris Kontokostas, Patrick Westphal, Sören Auer, Sebastian Hellmann, Jens Lehmann, Roland Cornelissen, and Amrapali Zaveri. "Test-driven evaluation of linked data quality". In: *Proceedings of the 23rd international conference on World Wide Web*. Ed. by Chin-Wan Chung. New York, NY, United States: Association for Computing Machinery, Apr. 2014, pp. 747–757. ISBN: 9781450327442. DOI: 10.1145/2566486.2568002. URL: http://dl.acm.org/citation.cfm?id=2568002.

[37] Jeremy Debattista, Sören Auer, and Christoph Lange. "Luzzu – A Methodology and Framework for Linked Data Quality Assessment". In: *J. Data and Information Quality* 8.1 (Oct. 2016), 4:1–4:32. ISSN: 1936-1955. DOI: 10.1145/2992786. URL: http://doi.acm.org/10.1145/2992786.

[38] Anastasia Dimou, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, Sebastian Hellmann, and Rik Van de Walle. "Assessing and Refining Mappings to RDF to Improve Dataset Quality". In: 2015, pp. 133–149. DOI: 10.1007/978-3-319-25010-6_8. URL: http://link.springer.com/chapter/10.1007/978-3-319-25010-6_8.

[39] André Langer and Martin Gaedke. "DaQAR – An Ontology for the Uniform Exchange of Comparable Linked Data Quality Assessment Requirements". In: *Web Engineering*. Ed. by Tommi Mikkonen, Ralf Klamma, and Juan Hernández. Vol. 10845. Lecture Notes in Computer Science. Cham: Springer, 2018, pp. 234–242. ISBN: 978-3-319-91662-0. DOI: 10.1007/978-3-319-91662-0_18.

[40] Anisa Rula and Amrapali Zaveri. "Methodology for Assessment of Linked Data Quality." In: *LDQ@ SEMANTICS*. 2014, p. 34.

[41] Jeremy Debattista, Makx Dekkers, Christophe Guéret, Deirdre Lee, Nandana Mihindukulasooriya, and Amrapali Zaveri. *Data on the Web Best Practices: Data Quality Vocabulary*. Working Group Note. World Wide Web Consortium, Dec. 2016. URL: https://www.w3.org/TR/vocab-dqv/.

[42] Holger Knublauch and Dimitris Kontokostas. *Shapes Constraint Language (SHACL)*. Recommendation. World Wide Web Consortium (W3C), July 2017. URL: https://www.w3.org/TR/shacl/.

[43] Eric Prud'hommeaux, Jose Emilio Labra Gayo, and Harold Solbrig. "Shape expressions: an RDF validation and transformation language". In: *Proceedings of the 10th International Conference on Semantic Systems*. Ed. by Harald Sack, Agata Filipowska, Jens Lehmann, and Sebastian Hellmann. ACM. New York, NY, United States: Association for Computing Machinery, 2014, pp. 32–40. DOI: 10.1145/2660517.2660523. URL: http://dl.acm.org/citation.cfm?id=2660523.

[44] Ben De Meester, Pieter Heyvaert, Anastasia Dimou, and Ruben Verborgh. "Towards a Uniform User Interface for Editing Data Shapes". In: *Proceedings of the 4th International Workshop on Visualization and Interaction for Ontologies and Linked Data*. Ed. by Valentina Ivanova, Patrick Lambrix, Steffen Lohmann, and Catia Pesquita. Vol. 2187. CEUR Workshop Proceedings. CEUR-WS.org, Oct. 2018, pp. 13–24. URL: http://ceur-ws.org/Vol-2187/paper2.pdf.

[45] Sven Lieber, Ben De Meester, Pieter Heyvaert, Femke Brückmann, Ruben Wambacq, Erik Mannens, Ruben Verborgh, and Anastasia Dimou. "Visual Notations for Viewing RDF Constraints with UnSHACLed [to be published]". In: *Semantic Web Journal* Pre-press (Nov. 2021), pp. 1–36. DOI: 10.3233/SW-210450.

[46] Mark D. Wilkinson et al. "The FAIR Guiding Principles for scientific data management and stewardship". In: *Scientific Data* 3 (Mar. 2016), p. 160018. DOI: 10.1038/sdata.2016.18.

[47] Alexander Nwala, Michele Weigle, and Michael Nelson. "Using Micro-collections in Social Media to Generate Seeds for Web Archive Collections". In: *2019 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*. IEEE. 2019, pp. 251–260.

[48] Marc Spaniol, Dimitar Denev, Arturas Mazeika, Gerhard Weikum, and Pierre Senellart. "Data Quality in Web Archiving". In: *Proceedings of the 3rd Workshop on Information Credibility on the Web*. 2009, pp. 19–26.

[49] Dimitar Denev, Arturas Mazeika, Marc Spaniol, and Gerhard Weikum. "The SHARC Framework For Data Quality in Web Archiving". In: *The VLDB Journal* 20.2 (2011), pp. 183–207.

[50] J. M. Juran. *Juran's Quality Control Handbook*. Ed. by Frank M. Mryna. 4th. Texas, USA: Mcgraw-Hill, Aug. 1988. URL: http://www.pqm-online.com/assets/files/lib/books/juran.pdf.

[51] Nandana Mihindukulasooriya, María Poveda-Villalón, Raúl García-Castro, and Asunción Gómez-Pérez. "Loupe-An Online Tool for Inspecting Datasets in the Linked Data Cloud." In: *International Semantic Web Conference (Posters & Demos)*. Vol. 1. 1. 2015, p. 2.

[52] Kārlis Čerāns, Jūlija Ovčiņņikova, Uldis Bojārs, Mikus Grasmanis, Lelde Lāce, and Aiga Romāne. "Schema-Backed Visual Queries over Europeana and Other Linked Data Resources". In: *The Semantic Web: ESWC 2021 Satellite Events*. Ed. by Ruben Verborgh, Anastasia Dimou, Aidan Hogan, Claudia d'Amato, Ilaria Tiddi, Arne

Bröring, Simon Mayer, Femke Ongenae, Riccardo Tommasini, and Mehwish Alam. Cham: Springer International Publishing, 2021, pp. 82–87. ISBN: 978-3-030-80418-3.

[53] Fabian Kirstein, Kyriakos Stefanidis, Benjamin Dittwald, Simon Dutkowski, Sebastian Urbanek, and Manfred Hauswirth. "Piveau: A Large-Scale Open Data Management Platform Based on Semantic Web Technologies". In: *European Semantic Web Conference*. Springer. 2020, pp. 648–664.

[54] Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. "RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data". In: *Proceedings of the 7th Workshop on Linked Data on the Web*. Ed. by Christian Bizer, Tom Heath, Sören Auer, and Tim Berners-Lee. Vol. 1184. CEUR Workshop Proceedings. CEUR-WS.org, 2014. URL: http://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf.

[55] Valentina Presutti, Enrico Daga, Aldo Gangemi, and Eva Blomqvist. "eXtreme Design with Content Ontology Design Patterns". In: *WOP 2009 - Workshop on Ontology Patterns*. Ed. by Eva Blomqvist, Kurt Sandkuhl, Francois Scharffe, and Vojtech Svatek. Vol. 516. 2009, pp. 83–97. URL: http://ceur-ws.org/Vol-516/pap21.pdf.

[56] Andrea Cimmino, Alba Fernández-Izquierdo, and Raúl García-Castro. "Astrea: Automatic Generation of SHACL Shapes from Ontologies". In: *European Semantic Web Conference (ESWC)*. Ed. by Andreas Harth, Sabrina Kirrane, Axel-Cyrille Ngonga Ngomo, Heiko Paulheim, Anisa Rula, Peter Haase, and Michael Cochez. Springer. Springer International Publishing, 2020, pp. 497–513. DOI: 10.1007/978-3-030-49461-2_29.

[57] Eva Blomqvist, Karl Hammar, and Valentina Presutti. "Engineering Ontologies with Patterns: The eXtreme Design Methodology". In: *Ontology Engineering with Ontology Design Patterns*. Ed. by Pascal Hitzler, Aldo Gangemi, Krzysztof Janowicz, Adila Krisnadhi, and Valentina Presutti. Vol. 25. Studies on the Semantic Web. IOS Press, 2016, pp. 23–50. DOI: 10.3233/978-1-61499-676-7-23.

[58] Julien Corman, Fernando Florenzano, Juan L Reutter, and Ognjen Savkovic. "SHACL2SPARQL: Validating a SPARQL Endpoint against Recursive SHACL Constraints." In: *ISWC Satellites*. 2019, pp. 165–168.

[59] Mónica Figuera, Philipp D Rohde, and Maria-Esther Vidal. "Trav-SHACL: Efficiently Validating Networks of SHACL Constraints". In: *Proceedings of the Web Conference 2021*. 2021, pp. 3337–3348.

Chapter 5

Conclusion

In this chapter, the research questions and hypothesis are reviewed based on how the contributions impact the challenges (Table 5.1), and finally the remaining challenges and future directions are discussed.

Table 5.1: Alignment between the contributions, challenges, research questions, hypotheses, and whether the contribution is part of restriction *assessment*, restriction *use* or constraint *creation*.

Contribution	Research challenge	Research question	Hypothesis	Impact
Montolo	I	1	1	Restriction assessment
Visual Notations	II	2	2	Constraint creation
BESOCIAL	II	3	3	Restriction use

5.1 Impact of Contributions

This dissertation tried to answer the main research question "How can we support users in the assessment and in the creation of Knowledge Graph restrictions?". The first sub research question is

Research Question 1: *How can we support the assessment of restrictions in existing Knowledge Graphs?*

with corresponding hypothesis

Hypothesis 1: *FAIR statistics of RDF encoded axioms and constraints enable restriction use assessments of several existing Knowledge Graphs not possible with state of the art tools.*

In this dissertation, the gap of missing user support for Knowledge Graph assessment was identified: assessment tools exist, but only consider a single ontology or in case statistics are created they miss restriction use. The presented Montolo solution provides FAIR statistics of restriction use with respect to different syntactical modeling patterns (restriction type expressions). Additionally, the RDF description of restriction types and expressions in Montolo is extensible and also FAIR. This dissertation contributes FAIR axiom use statistics for over 1,000 ontologies from the LOV and BioPortal repositories, as well as constraint use statistics for data shapes from 19 selected GitHub repositories. Additionally, these results were analyzed to provide insights and evidence of current restriction use. Montolo covers a common subset of restriction types from related work [1] and some possible expressions thereof. More restriction types and expressions, as well as more expressions of the existing restriction types can be implemented because the approach is extensible. Additionally, the Montolo dataset is published using a CC0 license and thus is dedicated to the public domain to allow integration in other tools. Even though within this PhD I could not further experiment with Montolo in ontology reuse use cases, because no project presented a use case where different restriction types mattered, Montolo's extension was demonstrated in section Section 2.2 for other restriction types and expressions. **Therefore I accept Hypothesis 1**.

Since Montolo was released, the scalable SANSA stack [2, 3] was extended with OWLStats [4], providing the capability of measuring OWL axiom use. With the current LODStats-based [5], and thus streaming-based implementation of Montolo, no performance bottlenecks critical for Montolo's use cases were observed: even a large scale restriction use analysis of LOV and BioPortal which does not even need to be executed for every ontology reuse, was performed in several minutes. Nevertheless, the development of OWLStats on the one hand likely improves use cases where large scale ontologies need to be assessed even quicker, and on the other hand provides a scalable and maintained code base interesting for future implementations of Montolo.

The Montolo solution, as an extensible approach to separate definition of restriction types from restriction type expressions provides added value to the SANSA stack and still seems to reflect the state of the art regarding the definition and detection of syntactical restriction modeling patterns. Similarly, to the best of my knowledge no other restriction use statistics are provided in a FAIR fashion. The evaluation of the OWLStats solution focused on performance and no measured statistics are provided.

The second research question is

Research Question 2: *How can we support users familiar with Linked Data in viewing RDF constraints?*

with corresponding hypothesis

Hypothesis 2: *Users familiar with Linked Data can answer questions about visually represented RDF constraints more accurately with a VOWL-based visual notation than with an UML-based visual notation*

This dissertation identified the gap of missing user support for the visual creation of Knowledge Graph constraints: existing tools either did not support all SHACL core constraints or did not provide a visual notation specification. To fill this gap, this dissertation contributed the two visual notation specifications ShapeUML and ShapeVOWL, which both support all SHACL core constraints. The conducted comparative user study revealed that more than 80% of the questions regarding visualized constraints were answered correctly with both notations. However, there is no statistically significant difference in error rates between ShapeUML and ShapeVOWL. **Therefore Hypothesis 2 is rejected**: users could not answer questions more accurately with a VOWL based visual notation compared to an UML-based visual notation.

However, the conducted study also provides other findings and contributions. A detailed comparison of both notations based on cognitive effective design principles provides an objective documentation about strengths and weaknesses of each notation. This is crucial as it can guide systematic improvements of the notations. None of the notations was preferred over the other, both notations are valuable in different use cases.

As part of this PhD, also the web-based editor UnSHACLed was provided which implements both visual notations. Therefore, an artefact is provided to answer questions relevant to human problems contributing new knowledge [6]. Such design science research [7] enables researchers to understand the creation and use of constraints as the editor may help in further studies to explore the use of the notations in new use cases.

The third research question investigating a particular Knowledge Graph restrictions use case is

Research Question 3: *How can axioms and constraints support archiving institutions in the data stewardship of heterogeneous social media data?*

with corresponding hypothesis

Hypothesis 3: *The W3C-recommended constraint language SHACL can be used to declaratively assess data quality metrics for use case specific data quality of heterogeneous social media data, integrated into an RDF graph with formal meaning*

The challenge of enabling data stewardship was addressed by using Knowledge Graph restrictions. On the one hand, RDFS-based ontologies with formal meaning are reused to integrate heterogeneous social media data. This enables data stewardship, according to the informal definition of this dissertation (definition 15) as information is accessible via a Knowledge Graph: all relevant information can be discovered and reused for downstream investigations. On the other hand, SHACL-based constraints in conjunction with semantic descriptions of data quality metrics are used to support data quality assessments. This provides the added value of ensuring correct access, because the intended structure of data is described, deviations from this structure can be identified and fixed. A common data quality methodology was followed to create use case specific data quality dimensions and metrics. These dimensions and metrics were described using interoperable RDF and corresponding

constraints to measure the metrics were implemented using SHACL. This enabled a declarative quality assessment by executing a SHACL validation process and querying the results as well as metrics and dimensions using SPARQL. **Therefore I accept Hypothesis 3.**

The presented use case explored the use of restrictions for data stewardship. In principle it is generalizable as (i) a Knowledge Graph needs to be generated, which can happen via declarative rules, (ii) quality dimensions and metrics need to be specified, which can be defined by following existing methodologies, and (iii) quality metrics need to be measured, which can be performed by a validation process of corresponding SHACL constraints on the previously generated Knowledge Graph. However, especially the declarative measurement of quality metrics by detecting violations of constraints is limited by the expressivity of the used constraint language. For example, a constraint to check if a URI on the Web resolves can currently not be expressed using SHACL nor ShEx. Therefore, the presented approach is only generalizable to a certain extent and custom software to measure quality metrics might still be needed.

The evaluation of these three hypotheses contribute to the main research question "How can we support users in the assessment and in the creation of Knowledge Graph restrictions?". Montolo can be used to assess Knowledge Graphs with respect to restrictions: FAIR statistics contributed by this dissertation can be reused by ontology engineers and new restriction types or expressions can be added if necessary. Therefore, the capabilities of ontology engineers have been improved compared to the state of the art because now FAIR statistics are at their disposal. Different tools exist to provide visual support for axioms, but not all SHACL core constraints could be visualized with state of the art tools. This dissertation contributed two visual notations. None of the presented visual notations was preferred over the other by users in a performed user study, but users are now visually supported in the creation of constraints. Finally, the presented use case of social media preservation demonstrates how restrictions can be used to enable data stewardship: a declarative Knowledge Graph generation provides a uniform view on heterogeneous social media data and a declarative quality assessment could be performed using SHACL constraints without the need of a custom quality assessment tool.

5.2 Remaining Challenges and Future Directions

As with any research, there are remaining challenges for the presented contributions. This section first elaborates on remaining challenges of restriction assessment and constraint creation in Section 5.2.1. Then it focuses on general challenges and future directions with respect to the use of restrictions within a broader knowledge engineering context in Section 5.2.2.

5.2.1 Challenges for the Creation and Assessment of Restrictions

This section discusses challenges for two of the contributions of this dissertation: the assessment of restrictions in existing Knowledge Graphs using Montolo and the support of users in constraint creation using the visual notations ShapeUML and ShapeVOWL.

User support via the visual notations ShapeUML and ShapeVOWL The creation of constraints can be improved by extending the visual notations ShapeUML and ShapeVOWL both with respect to received user feedback and with respect to covering other constraint languages. Qualitative findings underline that both notations have their strengths in different use cases, for example several visual variables in ShapeVOWL which make the detection of constraints easier, or a listing of constraints in ShapeUML perceived more orderly. Thus further research can investigate different use cases as well as more adaptations to the visual notations.

The contributed visual notations are independent of a specific constraint language, but currently all W3C SHACL core constraints are represented as it is the W3C recommendation and explicitly lists constraint types. However, other RDF constraint languages exist, such as the broadly used language ShEx [8]. Both SHACL and ShEx have different perspectives and formalisms but share many similarities [9], therefore future research investigating a mapping from ShEx to the presented visual notations is promising: new use cases and communities can be supported, for instance the Wikidata community which adopted ShEx.

Different creation approaches may provide more insights in the use of visual notations within tools such as UnSHACLed. The performed user study investigated visual notations, however, such notations are often used within tools and that poses more challenges and opportunities. RDF constraints might be defined for validation, user interface generation or documentation [9], therefore different workflows are possible which need to be supported.

Restriction assessment using Montolo This dissertation demonstrated how statistics about restrictions can be obtained using Montolo. However, future research is needed to evaluate such statistics in different use cases and to make them more accessible. With respect to axioms, the usefulness of Montolo can be evaluated within an ontology reuse scenario where users have to discover and select ontologies fitting their use case. This could provide insights with respect to statistics which are not yet covered. With respect to the accessibility of the statistics, it could be investigated how Montolo can be integrated into ontology repositories such as LOV, i.e. computing the statistics when cataloging ontologies and making the statistics accessible as filters in the existing search capabilities. This could provide data on how restriction statistics are actually used by users. From a broader assessment context, Montolo provides metrics of restriction use which can be used to represent characteristics of higher level quality dimensions. Thus, further research can investigate quality categories and dimensions relevant for Knowledge Graph assessments. Also other higher level metrics can be defined based on restriction types, for example the OWL profile used by an ontology which depends also on used OWL terms, detectable by Montolo.

Large scale repositories for ontologies (containing axioms) exist, but nothing comparable exists yet for data shapes (containing constraints) which poses a gap. One could argue that ontologies are meant to be reused and therefore should be findable via repositories, whereas constraints are application-specific. However, in the practical use of RDF-based Knowledge Graphs data consumers need to know the structure of the data to provide seemingless working workflows. Therefore data producers need to describe the structure of the data

i.e. the production/consumption dilemma [9]. Research regarding repositories or data aggregation portals for constraints would allow data producers and consumers to exchange information and would lower the bar for large scale analysis of constraint use.

SHACL and ShEX When assessing or creating constraints, this dissertation focused on SHACL as it is the W3C recommended constraint language for RDF. However, the presented contributions can also be adapted for ShEx because both languages have a significant intersection [9]. For the restriction assessment with Montolo, new restriction type expressions would need to be defined and implemented, i.e. syntactical patterns to measure ShEX classes and properties in RDF. Further research is required to identify which ShEX constraints can be defined as expressions of which existing restriction types. For visual creation of constraints, the presented visual notations would need to be mapped to semantic constructs of ShEX. This may involve changes in the visual notation and hence also requires further research.

5.2.2 Future Directions for Knowledge Engineering

This dissertation covered the BESOCIAL social media archiving use case whose data stewardship with a Knowledge Graph profits from restrictions in the form of axioms and constraints. Especially because RDF-based constraint languages were just recommended in recent years, general methodologies are needed to decide when and how axioms and constraints need to be used. This section first elaborates on future directions for restriction assessment which reveal issues. Based on these issues, future directions for knowledge engineering are outlined with the potential to provide a systematic methodology for the use of axioms and constraints.

Future directions for restriction assessment The Montolo approach considers restrictions already encoded with RDF terms, but a more holistic approach to assess restrictions is needed to validate knowledge representations. Montolo is a first step to assess and compare individual Knowledge Graphs or get an understanding about which terms are used by different communities, and which are not. However, it does not answer questions related to the motivation to use those restrictions in the first place, i.e. the why and how. Such questions recently sparked discussion in the Semantic Web community, manifested in arguments for a more pro OWL[1] (axiom-based restrictions) or pro SHACL[2] (constraint-based restrictions) modeling approach.

Investigating why and how different restrictions are used requires analyzing the use case and context of a Knowledge Graph. This is challenging because, compared to measuring already encoded RDF terms, (standardized) data regarding ontology documentation (for example ontologies' requirements) is sparse, especially on a large scale such as for all ontologies of LOV. Therefore future work could, on the one hand, qualitatively analyze a

[1] Triply, "Why We Use OWL Every Day at Triply", https://web.archive.org/web/20210901132625/https://triply.cc/blog/2021-08-why-we-use-owl (archived website accessed February 12, 2022)

[2] Irene Polikoff, "Why I Don't Use OWL Anymore", https://web.archive.org/web/20211103090854/https://www.topquadrant.com/owl-blog/ (archived website accessed Februar 12, 2022)

limited number of Knowledge Graphs with respect to their requirements – and in turn modeling choices regarding restrictions. And on the other hand, improve the availability of (standardized) documentation of requirements and design decisions during the systematic creation of ontologies using ontology engineering methodologies.

Future directions for knowledge engineering A view across the contributions of this dissertation motivates a general methodology for the development of Knowledge Graph restrictions. Existing ontology engineering methodologies provide generic frameworks one can follow to represent knowledge. They provide workflows to among others (i) collect requirements, (ii) support in the modeling of domain knowledge, (iii) compare and reuse existing ontologies, or (iv) verify and evaluate created ontologies. From the perspective of restrictions and the earlier mentioned axiom vs constraint-based thinking, it is interesting to consider the whole context: the domain the knowledge aims to describe and the actual use of this domain knowledge in applications. Requirements of the whole context – and not just of the domain knowledge – need to be described in an interoperable form. This would guide a more systematic approach to identify, and eventually encode, restrictions compared to the state of the art. Our contributions to assess, create and use restrictions become then relevant methods to support users in the complex and tedious task of knowledge engineering.

Such a more holistic knowledge engineering methodology will make the creation of knowledge representations more systematic. Design decisions will be based on explicit documentation of the needs and will solve the use case at hand by appropriate representation of restrictions. For example, a decision to encode certain restrictions using RDFS/OWL or SHACL/ShEx will be more systematic and traceable.

References

[1] Dörthe Arndt, Ben De Meester, Anastasia Dimou, Ruben Verborgh, and Erik Mannens. "Using Rule-Based Reasoning for RDF Validation". In: *Rules and Reasoning: International Joint Conference, RuleML+RR 2017, London, UK, July 12–15, 2017*. Ed. by Stefania Constantini, Enrico Franconi, William Van Woensel, Roman Kontchakov, Fariba Sadri, and Dumitru Roman. Vol. 10364. Lecture Notes in Computer Science. Cham: Springer, July 2017, pp. 22–36. DOI: 10.1007/978-3-319-61252-2_3.

[2] Jens Lehmann et al. "Distributed Semantic Analytics Using the SANSA Stack". In: *Lecture Notes in Computer Science*. Springer International Publishing, 2017, pp. 147–155. DOI: 10.1007/978-3-319-68204-4_15.

[3] Gezim Sejdiu, Anisa Rula, Jens Lehmann, and Hajira Jabeen. "A Scalable Framework for Quality Assessment of RDF Datasets". In: *Lecture Notes in Computer Science*. Springer International Publishing, 2019, pp. 261–276. DOI: 10.1007/978-3-030-30796-7_17.

[4] Heba Mohamed, Said Fathalla, Jens Lehmann, and Hajira Jabeen. "OWLStats: Distributed Computation of OWL Dataset Statistics". In: *2020 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*. IEEE, Dec. 2020. DOI: 10.1109/wiiat50758.2020.00055.

[5] Sören Auer, Jan Demter, Michael Martin, and Jens Lehmann. "LODStats - An Extensible Framework for High-Performance Dataset Analytics". In: *EKAW 2012 : The 18th International Conference on Knowledge Engineering and Knowledge Management*. 2012.

[6] Alan Hevner and Samir Chatterjee. *Design Research in Information Systems*. Springer US, 2010. ISBN: 978-1-4419-5653-8. DOI: 10.1007/978-1-4419-5653-8.

[7] Jan Recker. *Scientific Research in Information Systems: A Beginner's Guide*. Springer Science & Business Media, 2013. ISBN: 9783642-300479.

[8] Eric Prud'hommeaux, Jose Emilio Labra Gayo, and Harold Solbrig. "Shape expressions: an RDF validation and transformation language". In: *Proceedings of the 10th International Conference on Semantic Systems*. Ed. by Harald Sack, Agata Filipowska, Jens Lehmann, and Sebastian Hellmann. ACM. New York, NY, United States: Association for Computing Machinery, 2014, pp. 32–40. DOI: 10.1145/2660517.2660523. URL: http://dl.acm.org/citation.cfm?id=2660523.

[9] Jose Emilio Labra Gayo, Eric Prud'hommeaux, Iovka Boneva, and Dimitris Kontokostas. *Validating RDF Data*. Vol. 7. Synthesis Lectures on the Semantic Web: Theory and Technology 1. Morgan & Claypool Publishers LLC, Sept. 2017, pp. 1–328. DOI: 10.2200/s00786ed1v01y201707wbe016. URL: http://book.validatingrdf.com/.

Appendix A

Resources

Within this dissertation I tried to follow Open Science principles as much as possible. Whenever datasets or other resources were created the mantra "As Open as Possible, as Closed as Necessary" was followed. These resources were deposited on common repositories such as Zenodo or FigShare or on code platforms such as GitHub.

However, to make this dissertation self-contained, relevant subsets of these resources are added in this appendix: the questionnaires from group A of the user study regarding visual notations and Montolo definitions related to the disjoint classes restriction type. For convenience all publicly available resources of this dissertation are listed below.

- RDF description of Montolo Restriction Types and Expressions: `https://doi.org/10.5281/zenodo.3343313`
- MontoloStats dataset: `https://doi.org/10.5281/zenodo.334305a3`
- MontoloSHACLStats dataset: `https://doi.org/10.5281/zenodo.4154456`
- ShapeUML visual notation specification: `https://w3id.org/imec/unshacled/spec/shape-uml`[1]
- ShapeVOWL visual notation specification: `https://w3id.org/imec/unshacled/spec/shape-vowl`[2]
- ShapeViBe visualization benchmark: `https://w3id.org/imec/unshacled/shape-vibe`[3]
- Visual notation user study material: https://doi.org/10.6084/m9.figshare.13614440.v2
- BESOCIAL quality datasets: https://doi.org/10.6084/m9.figshare.16655239.v1

All public GitHub repositories are listed below:

- Montolo: https://github.com/IDLabResearch/Montolo
- Montolo vocabulary: https://github.com/IDLabResearch/montolo-voc
- Adapted LODStats extension LOVStats: https://github.com/IDLabResearch/lovstats
- UnSHACLed web editor: https://github.com/KNowledgeOnWebScale/unshacled
- BESOCIAL: https://github.com/RMLio/social-media-archiving

A.1 User study Questionnaire Group A

[1] Sven Lieber, "ShapeUML", `http://web.archive.org/web/20210313073403/https://lov.ilabt.imec.be/unshacled/spec/shape-uml/` (archived website accessed February 12, 2022)

[2] Sven Lieber "ShapeVOWL", `http://web.archive.org/web/20220212144950/https://lov.ilabt.imec.be/unshacled/spec/shape-vowl/` (archived website accessed February 12, 2022)

[3] Sven Lieber, "ShapeViBe", `http://web.archive.org/web/20220212145053/https://lov.ilabt.imec.be/unshacled/shape-vibe/` (archived website accessed February 12, 2022)

RDF constraints visualization (version A)

A comparison between the visual notations ShapeUML and ShapeVOWL
There are 92 questions in this survey.

Pre-questionnaire

This is a pre questionnaire to gather social demographics and skill level

What is your year of birth? *

❗ Only numbers may be entered in this field.
❗ Your answer must be between 1900 and 2005
Please write your answer here:

With what gender do you identify? *

❗ Choose one of the following answers
Please choose **only one** of the following:

- ◯ Male
- ◯ Female
- ◯ Other

What is the highest level of education that you completed? *

Choose one of the following answers
Please choose **only one** of the following:

- ◯ Did not complete high school
- ◯ High school
- ◯ Bachelor's degree
- ◯ Master's degree
- ◯ Advanced graduate work or PhD
- ◯ Not sure

What is your employment status? *

Choose one of the following answers
Please choose **only one** of the following:

- ◯ Employed for wages
- ◯ Self-employed
- ◯ Out of work
- ◯ A homemaker
- ◯ A student
- ◯ Retired

What is your experience with Linked Data? (Multiple options possible) *

Check all that apply
Please choose **all** that apply:

- [ ] I generate Linked Data
- [ ] I check the quality of Linked Data
- [ ] I use Linked Data
- [ ] I publish Linked Data
- [ ] I perform reasoning on Linked Data
- [ ] I have a basic understanding of Linked Data
- [ ] I have no knowledge about Linked Data

Please indicate where you assess yourself in the topic of Linked Data?: *

Choose one of the following answers
Please choose **only one** of the following:

- ◯ Novice
- ◯ Emerging
- ◯ Developing
- ◯ Proficient
- ◯ Expert

Did or do you already create custom *tools / software / scripts* to validate Linked Data? *

Please choose **only one** of the following:

- ◯ Yes
- ◯ No

Please specify which custom *tools / software / scripts* you have created (open question)

Only answer this question if the following conditions are met:
PreQ7 (/survey322/index.php/admin/questions/sa/view/surveyid/698467/gid/609/qid/6922) == "Y"

Please write your answer here:

Please indicate if you have heard or used any of these tools or frameworks: *

Please choose the appropriate response for each item:

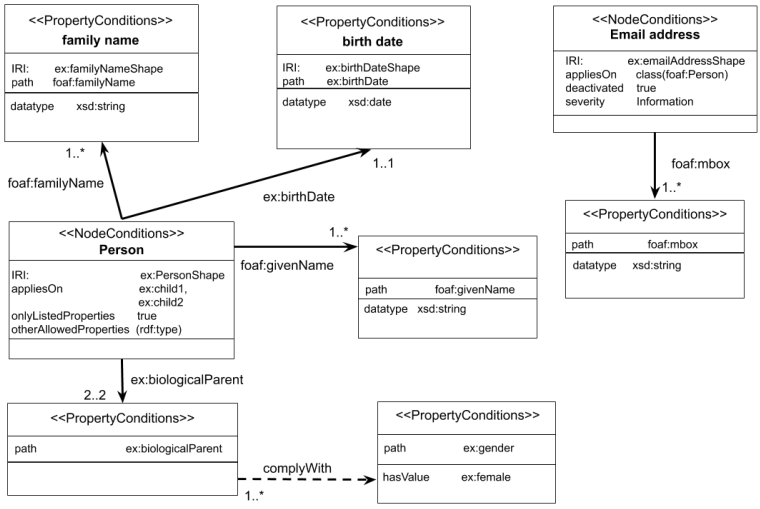
	I haven't used nor heard of it	I have heard of it, but I am not quite sure what it does	I have heard of it and I know what it does	I have used it
UnSHACLed (http://unshacled.com/)	◯	◯	◯	◯
RDFShape (http://rdfshape.weso.es/)	◯	◯	◯	◯
TopBraid Composer (https://www.topquadrant.com/products/topbraid-composer/)	◯	◯	◯	◯
WebVOWL (http://vowl.visualdataweb.org/webvowl.html)	◯	◯	◯	◯
UML class diagrams	◯	◯	◯	◯

If you have a research/professional position in the Semantic Web, what are your main topics? (open question)

Please write your answer here:

General (ShapeUML)

These data shapes define constraints on a person.



On how many subjects do the shown “Person” conditions apply by default? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many node conditions only allow to have the listed properties? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many properties do have conditions? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many min or max cardinality conditions can you see (infinity and zero not counted)? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many datatype constraints can you see? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many property values should comply with a specific data shape? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

The conditions of how many properties will not be validated? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many properties have the severity Violation? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many data shapes have a human-readable name? *

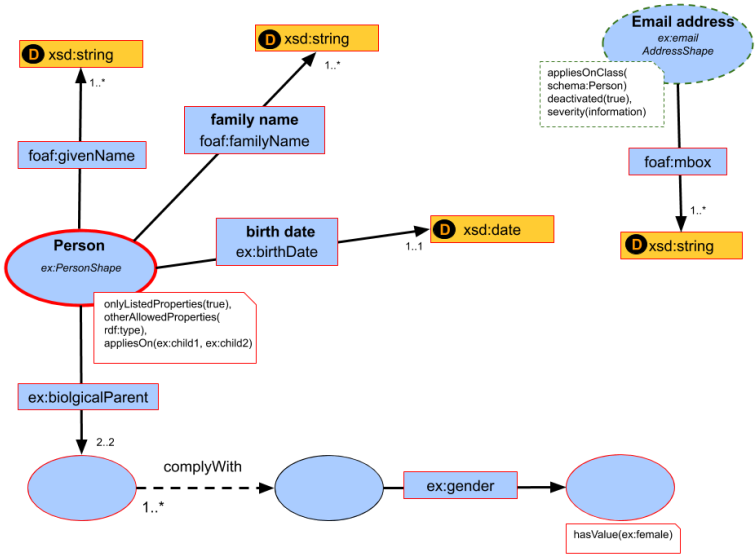
Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

Is there anything else you want to tell us about the shown example?

Please write your answer here:

General (ShapeVOWL)

These data shapes define constraints on a person.



On how many subjects do the shown “Person” conditions apply by default? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many node conditions only allow to have the listed properties? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many properties do have conditions? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many min or max cardinality conditions can you see (infinity and zero not counted)? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many datatype constraints can you see? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many property values should comply with a specific data shape? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

The conditions of how many properties will not be validated? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many properties have the severity Violation? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many data shapes have a human-readable name? *

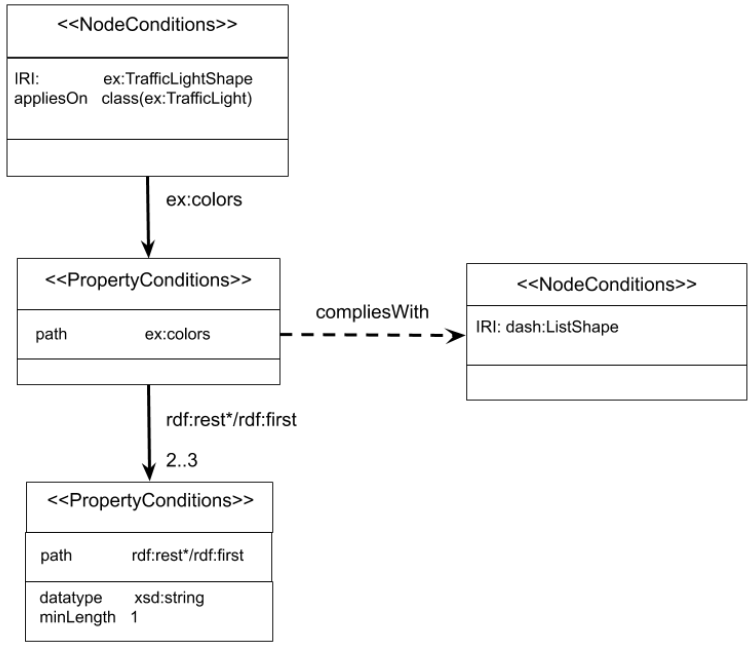
❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

Is there anything else you want to tell us about the shown example?

Please write your answer here:

Traffic Lights (ShapeUML)

This example shows constraints on a fictive traffic light.



On how many RDF classes are the shown constraints applied by default? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many properties do have conditions? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many zero-or-more property paths can you see? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many property conditions with the severity “information” can you see? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many node or property conditions are deactivated? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many node conditions are closed, i.e. can only have values for the listed properties to be valid? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many datatype constraints can you see? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many properties have a maximum cardinality (infinity not counted)? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many property values have an allowed maximum value (infinity not counted)? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values must have a minimum length (zero not counted)? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values must be less than other property values? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many disjunctions (logical or) relationships can you see? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values should comply with a specific data shape? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values must have a specific value? *

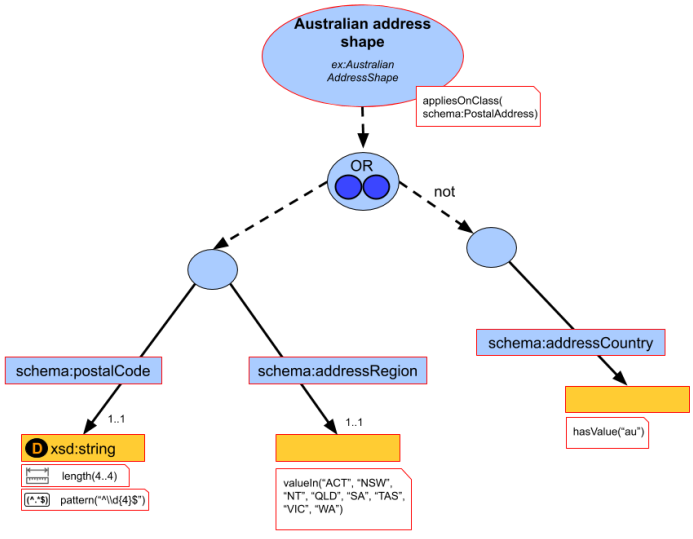
Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

Is there anything else you want to tell us about the shown example?

Please write your answer here:

Australian Address (ShapeVOWL)

This example shows constraints on an Australian address defined using schema.org (real world example taken from the web).



On how many RDF classes are the shown constraints applied by default? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many properties do have conditions? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many zero-or-more property paths can you see? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many property conditions with the severity “information” can you see? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many node or property conditions are deactivated? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many node conditions are closed, i.e. can only have values for the listed properties to be valid? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many datatype constraints can you see? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many properties have a maximum cardinality (infinity not counted)? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values have an allowed maximum value (infinity not counted)? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values must have a minimum length (zero not counted)? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values must be less than other property values? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many disjunctions (logical or) relationships can you see? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values should comply with a specific data shape? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values must have a specific value? *

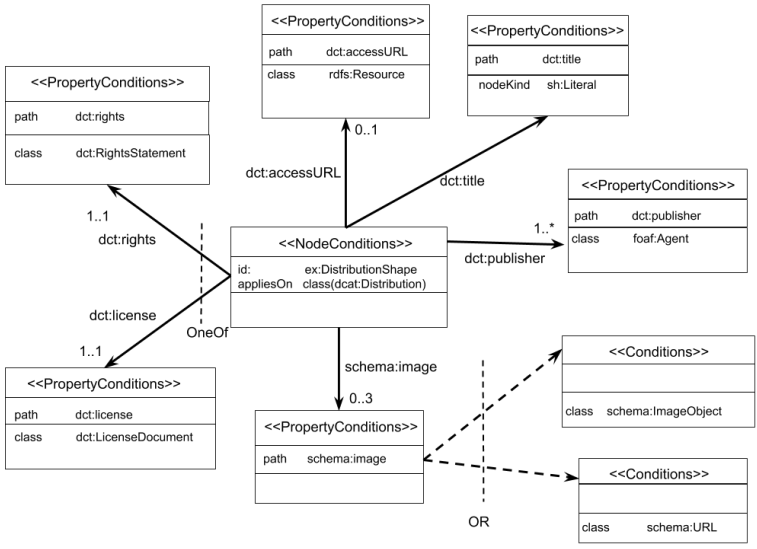
Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

Is there anything else you want to tell us about the shown example?

Please write your answer here:

DCAT-AP Switzerland (ShapeUML)

This example shows constraints on an the DCAT-AP vocabulary for data portals in Switzerland (excerpt of a real world example taken from GitHub).



On how many RDF classes are the shown constraints applied by default? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many properties do have conditions? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many zero-or-more property paths can you see? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many property conditions with the severity “information” can you see? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many node or property conditions are deactivated? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many node conditions are closed, i.e. can only have values for the listed properties to be valid? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many datatype constraints can you see? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many properties have a maximum cardinality (infinity not counted)? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values have an allowed maximum value (infinity not counted)? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values must have a minimum length (zero not counted)? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values must be less than other property values? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many disjunctions (logical or) relationships can you see? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values should comply with a specific data shape? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values must have a specific value? *

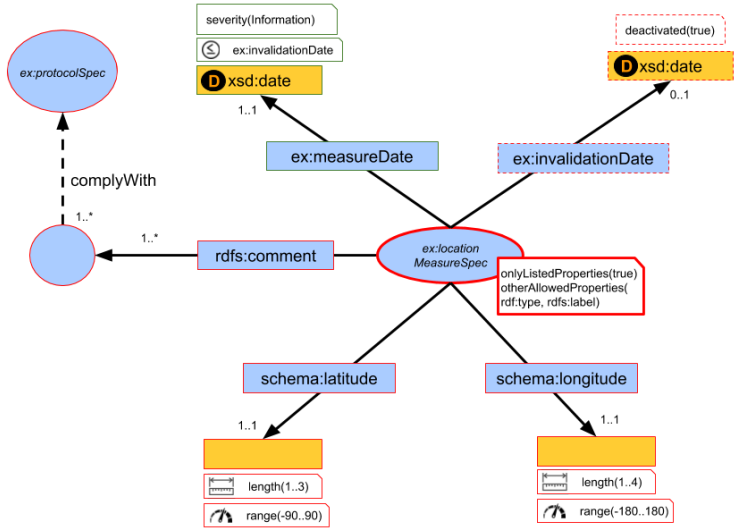
Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

Is there anything else you want to tell us about the shown example?

Please write your answer here:

Geo Coordinates (ShapeVOWL)

This example shows constraints on a fictive measurement of geo coordinates.



On how many RDF classes are the shown constraints applied by default? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many properties do have conditions? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many zero-or-more property paths can you see? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many property conditions with the severity “information” can you see? *

❗ Your answer must be between 0 and 99
❗ Only an integer value may be entered in this field.
Please write your answer here:

How many node or property conditions are deactivated? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many node conditions are closed, i.e. can only have values for the listed properties to be valid? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many datatype constraints can you see? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many properties have a maximum cardinality (infinity not counted)? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values have an allowed maximum value (infinity not counted)? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values must have a minimum length (zero not counted)? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values must be less than other property values? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many disjunctions (logical or) relationships can you see? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values should comply with a specific data shape? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

How many property values must have a specific value? *

Your answer must be between 0 and 99
Only an integer value may be entered in this field.
Please write your answer here:

Is there anything else you want to tell us about the shown example?

Please write your answer here:

Post-questionnaire

Please indicate in how far you agree with the following statements (1=not agree at all, 7=totally agree) *

Please choose the appropriate response for each item:

	1 - not agree at all	2	3	4	5	6	7 - totally agree
I am confident that the answers I provided are correct	◯	◯	◯	◯	◯	◯	◯
I prefer ShapeVOWL over ShapeUML to answer such questions.	◯	◯	◯	◯	◯	◯	◯
I would like to use ShapeVOWL to edit RDF constraints	◯	◯	◯	◯	◯	◯	◯

Is there anything else you want to tell us about the shown ShapeUML or ShapeVOWL visualizations, the questionnaire or the user study? Or any other opinion about both visual notations and your opinion on which you prefer in which situation and why?

Please write your answer here:

Thank you again for participating in our research. Stay safe and have a nice day!

Submit your survey.
Thank you for completing this survey.

A.2 User Study Follow-up Questionnaire Group A

RDF constraints - follow up (version A)

A comparison between the visual notations ShapeUML and ShapeVOWL
Thank you for participating in our research.

This survey investigates your experience with ShapeUML and ShapeVOWL, two user-oriented visual notations for Knowledge Graph constraints. To be able to sensefully complete the survey a basic understanding of the Resource Description Framework (RDF) is a prerequisite.

Filling in the survey will take approximately 15 minutes. For a greater scientific quality of the results, we would kindly ask you to fully complete it.

The results will be processed anonymously and will only be used for non-commercial, scientific purposes. The survey has been set up as part of a research project of IDLab (http://idlab.technology/). For more information concerning this research or for other questions, you can contact Sven Lieber (Sven.Lieber@UGent.be).

Before you start and if not already done, please have a look at the following two tutorials:

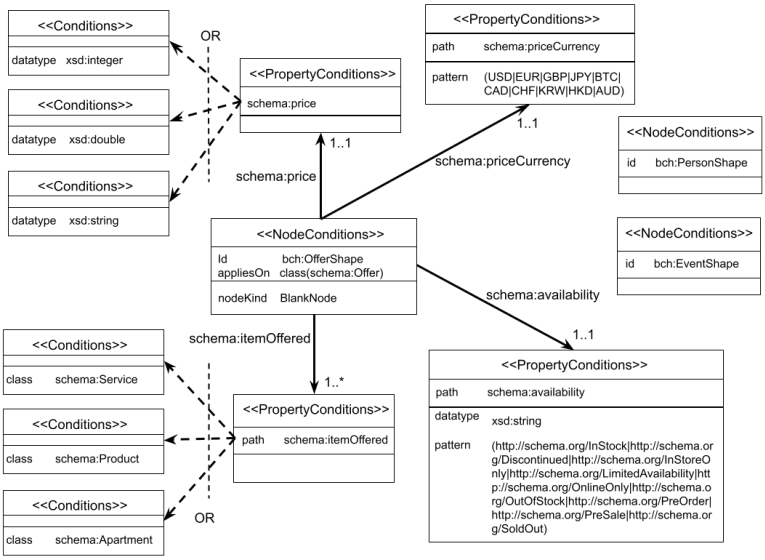
- ShapeUML introduction (document link (https://drive.google.com/file/d/1P3mZcpSTphAjYsAmdvjSVzvbFdsyoUG4/view?usp=sharing))
- ShapeVOWL introduction (document link (https://drive.google.com/file/d/1dIkL2lr_rRdnaEAVpDi4LggAFcL7fhmT/view?usp=sharing))

Please note that there will be a timeout if you take too long for a question. Please use the resume later button in case you need to interrupt the experiment for any reason.

There are 18 questions in this survey.

Offered items (ShapeUML)

These data shapes describe valid offers for products or apartments in a tourism context.



What is the property with the highest possible cardinality? *

❗ Check all that apply
❗ Please select at most one answer
Please choose **all** that apply:

- [ ] schema:price
- [ ] schema:itemOffered
- [ ] schema:availability
- [ ] schema:priceCurrency
- [ ] None of the above

What are valid datatypes for the property schema:price? *

❗ Check all that apply
❗ Please select at most one answer
Please choose **all** that apply:

- [ ] xsd:integer, xsd:double and xsd:string
- [ ] any datatype
- [ ] xsd:integer and xsd:double

What is the value of the nodeKind constraint for bch:OfferShape? *

❗ Check all that apply
❗ Please select at most one answer
Please choose **all** that apply:

- [ ] IRI
- [ ] BlankNode
- [ ] no value given

Which of the following options list the properties with a datatype constraint, if you were to put them in alphabetical order? *

❗ Check all that apply
❗ Please select at most one answer
Please choose **all** that apply:

- [ ] schema:availability, schema:itemOffered, schema:price
- [ ] schema:availability, schema:price, schema:priceCurrency
- [ ] None of the above

What is the number of constrained properties in the shown figure? *

Check all that apply
Please select at most one answer
Please choose **all** that apply:

- [ ] 4
- [ ] 3
- [ ] 8

What is the maximum cardinality of the property with the valid string value of "http://schema.org/PreSale'? *

Check all that apply
Please select at most one answer
Please choose **all** that apply:

- [ ] 1
- [ ] infinity
- [ ] no property has this as valid value

Do you see any "property path" which is not just a single property? *

Please choose **only one** of the following:

- ◯ Yes
- ◯ No

Which is the property path and where do you see it, please elaborate. *

Only answer this question if the following conditions are met:
AItemsUMLppath (/survey322/index.php/admin/questions/sa/view/surveyid/758248/gid/5607/qid/37729) == "Y"

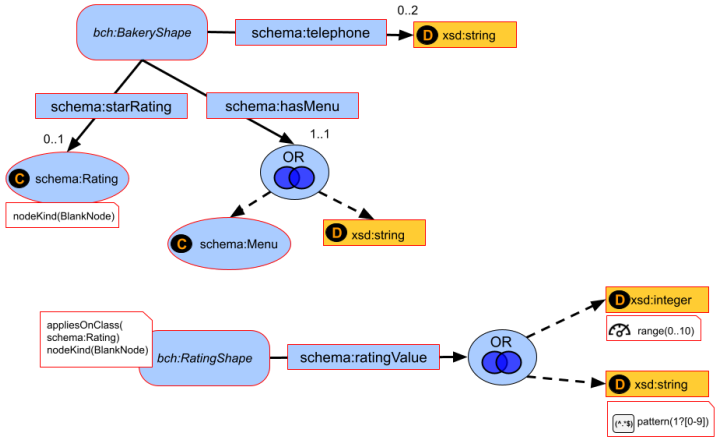
Please write your answer here:

Please let us know, was anything unclear in the shown example or with the questions? *

Please write your answer here:

Ratings (ShapeVOWL)

These data shapes describe ratings in a tourism context.



What is the property with the highest possible cardinality? *

❗ Check all that apply
❗ Please select at most one answer
Please choose **all** that apply:

- [ ] schema:telephone
- [ ] schema:ratingValue
- [ ] schema:starRating
- [ ] schema:hasMenu
- [ ] None of the above

What are valid datatypes for the property schema:ratingValue? *

❗ Check all that apply
❗ Please select at most one answer
Please choose **all** that apply:

- [ ] xsd:integer, xsd:double and xsd:string
- [ ] any datatype
- [ ] xsd:integer and xsd:double

On which subjects does the bch:RatingShape apply by default? *

❗ Check all that apply
❗ Please select at most one answer
Please choose **all** that apply:

- [ ] Instances of the class schema:Rating
- [ ] schema:ratingValue properties
- [ ] No default given

Which of the following options list the properties with a datatype constraint, if you were to put them in alphabetical order? *

❗ Check all that apply
❗ Please select at most one answer
Please choose **all** that apply:

- [ ] schema:hasMenu, schema:ratingValue, schema:telephone
- [ ] schema:hasMenu, schema:ratingValue, schema:starRating
- [ ] None of the above

What is the number of constrained properties in the shown figure? *

Check all that apply
Please select at most one answer
Please choose **all** that apply:

- [ ] 5
- [ ] 3
- [ ] 4

What is the maximum cardinality of the property with the valid integer value of "10"? *

Check all that apply
Please select at most one answer
Please choose **all** that apply:

- [ ] 1
- [ ] infinity
- [ ] There is no such property

Do you see any "property path" which is not just a single property? *

Please choose **only one** of the following:

- ◯ Yes
- ◯ No

Which is the property path and where do you see it, please elaborate. *

Only answer this question if the following conditions are met:
ARatingsVOWLppath (/survey322/index.php/admin/questions/sa/view/surveyid/758248/gid/5608/qid/37738) == "Y"

Please write your answer here:

Please let us know, was anything unclear in the shown example or with the questions? *

Please write your answer here:

Submit your survey.
Thank you for completing this survey.

A.3 Montolo Description

```
1
2   @prefix owl: <http://www.w3.org/2002/07/owl#> .
3   @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
4   @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
5   @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
6   @prefix mov: <https://w3id.org/montolo/ns/montolo-voc#> .
7   @prefix prov: <http://www.w3.org/ns/prov#> .
8   @prefix qb: <http://purl.org/linked-data/cube#> .
9   @prefix mon: <https://w3id.org/montolo/ns/montolo#>  .
10  @prefix dct: <http://purl.org/dc/terms/> .
11  @prefix frbr: <http://purl.org/vocab/frbr/core#> .
12
13  mon:disjointClasses a mov:RestrictionType ;
14      rdfs:isDefinedBy <https://w3id.org/montolo/ns/montolo#>  ;
15      rdfs:seeAlso <https://publikationen.bibliothek.kit.edu/1000054062> ;
16      dct:description """The constraint type disjoint classes states that all of the
17                         classes C_i, 1 <= i <= n, are pairwise disjoint."""@en ;
18      rdfs:label "Disjoint classes restriction type"@en .
19
20  mon:disjointClassesOwlDisjointWith a mov:RestrictionTypeExpression ;
21      rdfs:isDefinedBy <https://w3id.org/montolo/ns/montolo#>  ;
22      rdfs:comment """A disjoint classes restriction, expressed using the owl:disjointWith property."""@en ;
23      frbr:realizationOf mon:disjointClasses ;
24      rdfs:label "owl:disjointWith restriction"@en .
25
26  mon:disjointClassesOwlAllDisjointClasses a mov:RestrictionTypeExpression ;
27      rdfs:isDefinedBy <https://w3id.org/montolo/ns/montolo#>  ;
28      rdfs:comment """A disjoint classes restriction, expressed using the owl:AllDisjointClasses class."""@en ;
29      frbr:realizationOf mon:disjointClasses ;
30      rdfs:label "owl:AllDisjointClasses restriction"@en .
31
32  mon:disjointClassesDetectorOwlDisjointWith a mov:RestrictionTypeExpressionDetector ;
33      rdfs:isDefinedBy <https://w3id.org/montolo/ns/montolo#>  ;
34      rdfs:comment """A method or software component to detect owl:disjointWith restrictions."""@en ;
35      rdfs:label "owl:disjointWith detector"@en .
36
37  mon:disjointClassesLODStatsDetectorOwlDisjointWith-v1 a mov:RestrictionTypeExpressionDetectorVersion ;
38      rdfs:isDefinedBy <https://w3id.org/montolo/ns/montolo#>  ;
39      rdfs:label "owl:disjointWith detector (LODStats) v1"@en ;
40      frbr:realizationOf mon:disjointClassesDetectorOwlDisjointWith .
41
42
43  mon:disjointClassesDetectorOwlAllDisjointClasses a mov:RestrictionTypeExpressionDetector ;
44      rdfs:isDefinedBy <https://w3id.org/montolo/ns/montolo#>  ;
45      rdfs:comment """A method or software component to detect owl:AllDisjointClasses restrictions."""@en ;
46      rdfs:label "owl:AllDisjointClasses detector"@en .
47
48  mon:disjointClassesLODStatsDetectorOwlAllDisjointClasses-v1 a mov:RestrictionTypeExpressionDetectorVersion ;
49      rdfs:isDefinedBy <https://w3id.org/montolo/ns/montolo#>  ;
50      rdfs:label "owl:AllDisjointClasses detector (LODStats) v1"@en ;
51      frbr:realizationOf mon:disjoinClassesDetectorOwlAllDisjointClasses .
52
53
54  mon:restrictionTypeOccurrence a mov:RestrictionTypeMeasure ;
55      rdfs:comment """The occurrence of a restriction of a certain type."""@en ;
56      rdfs:label "Restriction type occurrence"@en .
```

Listing A.1: Montolo definitions for disjoint classes related entities.