Towards a collaborative framework for ontology engineering : Impact on ontology evolution and pitfalls in ontology networks and versioned ontologies

Omar Alqawasmeh

Thèse de doctorat

Université de Lyon

Opérée au sein de

Université Jean Monnet

École Doctorale 488 Sciences, Ingénierie et Santé (SIS)
Spécialité: Informatique
Nom d'ordre NNT: 2020LYSES017

Towards A Collaborative Framework for Ontology Engineering: Impact on Ontology Evolution and Pitfalls in Ontology Networks and Versioned Ontologies

---

Soutenue publiquement par:

Omar Alqawasmeh
Université de Lyon, Laboratoire Hubert Curien.

Superviseurs:

Prof. Pierre Maret
Université de Lyon, Laboratoire Hubert Curien.

Dr. Maxime Lefrançois
Université de Lyon, MINES Saint Étienne.

Dr. Antoine Zimmermann
Université de Lyon, MINES Saint Étienne.

Membres du jury:

Prof. Clément Jonquet	University of Montpellier	Rapporteur
Prof. Rajendra Akerkar	Western Norway Research Institute	Rapporteur
Dr. Ana Roxin	Laboratory LIB, Univ. Bourgogne Franche-Comté	Examinateur
Prof. Sylvie Després	University Sorbonne, Paris-Nord	Examinateur

Le 25 Septembre 2020

Doctoral Thesis

Université de Lyon

Prelared at

Université Jean Monnet

École Doctorale 488 Sciences, Ingénierie et Santé (SIS)
Specialty: Computer Science
Nom d'ordre NNT: 2020LYSES017

Towards A Collaborative Framework for Ontology Engineering: Impact on Ontology Evolution and Pitfalls in Ontology Networks and Versioned Ontologies

---

Presented and defended by:

Omar Alqawasmeh
Université de Lyon, Laboratoire Hubert Curien.

Supervisors:

Prof. Pierre Maret
Université de Lyon, Laboratoire Hubert Curien.

Dr. Maxime Lefrançois
Université de Lyon, MINES Saint Étienne.

Dr. Antoine Zimmermann
Université de Lyon, MINES Saint Étienne.

Jury members:

Prof. Clément Jonquet	University of Montpellier	Reviewer
Prof. Rajendra Akerkar	Western Norway Research Institute	Reviewer
Dr. Ana Roxin	Laboratory LIB, Univ. Bourgogne Franche-Comté	Examiner
Prof. Sylvie Després	University Sorbonne, Paris-Nord	Examiner

On the 25/09/2020

Declaration of Authorship

I, Omar ALQAWASMEH, declare that this thesis titled, "Towards A Collaborative Framework for Ontology Engineering: Impact on Ontology Evolution and Pitfalls in Ontology Networks and Versioned Ontologies" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a doctoral degree at the University of Lyon.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.

Signed: Omar ALQAWASMEH

Date: 25/09/2020

"*We are drowning in information but starved for knowledge...*"

John Naisbitt

Abstract

Towards A Collaborative Framework for Ontology Engineering: Impact on Ontology Evolution and Pitfalls in Ontology Networks and Versioned Ontologies

by Omar ALQAWASMEH

Ontologies are at the heart of the semantic web. Using ontologies leads to a better understanding, sharing and analyzing of knowledge in a specific domain. However, domains' description are subject to changes, thus arises the need to evolve ontologies in order to have an adequate representation of the targeted domain.

In this thesis, we assume that studying how the development and evolution of ontologies affect and is affected by the evolution of related artifacts, may help knowledge engineers in their tasks. Artifacts can be either external ontologies that are connected to a specific ontology or a service that take advantage of a specific ontology. Hence, we build up upon a comprehensive ontology evolution life-cycle. We introduce the following contributions: 1. a definition for a situation to detect the need of ontology evolution, 2. an original approach for ontology enrichment using external knowledge bases, 3. a new definition related to ontology evolution, named "ontology co-evolution" is used to assess the impact of ontology evolution, and 4. a new categorization of ontology pitfalls along with an evaluation of their importance and potential impact on versioned ontologies and ontology networks.

Ontology, Ontology engineering, Ontology networks, Versioned ontologies, Ontology evolution, Impact on ontology evolution, Pitfalls

Résumé

Vers un cadre de collaboration pour l'ingénierie de l'ontologie: impact sur l'évolution de l'ontologie et les pièges dans les réseaux d'ontologie et les ontologies versionnées

par Omar ALQAWASMEH

Les ontologies sont au cœur du web sémantique. L'utilisation d'ontologies permet de mieux comprendre, partager et analyser les connaissances dans un domaine spécifique. Cependant, la description des domaines est sujette à modifications, d'où la nécessité de faire évoluer des ontologies afin d'avoir une représentation adéquate du domaine visé.

Dans cette thèse, nous supposons que l'étude du développement et l'évolution des ontologies affectent et sont affectées par l'évolution des artefacts peut aider les ingénieurs du savoir dans leurs tâches. Par conséquent, nous nous appuyons sur un cycle de vie complet d'évolution d'ontologie. Nos contributions sont les suivantes: 1. une définition d'une situation pour détecter le besoin d'évolution des ontologies, 2. une approche originale pour l'enrichissement des ontologies utilisant des bases de connaissances externes, 3. une nouvelle définition liée à l'évolution des ontologies, nommée "co-évolution d'ontologie" est utilisé pour évaluer l'impact de l'évolution des ontologies, et 4. une nouvelle catégorisation des écueils dans le développement ou l'évolution des ontologies ainsi qu'une évaluation de leur importance et de leur impact potentiel sur les ontologies versionnées et les réseaux d'ontologies.

Extended Abstract

Ontologies play nowadays an important role in organizing and categorizing data in information systems and on the web. This leads to a better understanding, sharing and analyzing of knowledge in a specific domain. However, domains' description are subject to changes, thus arises the need to evolve ontologies in order to have an adequate representation of the targeted domain. Ontology evolution is the process of maintaining an ontology up to date with respect to the changes that arise in the targeted domain or in the requirements.

In this thesis, we assume that studying how the development and evolution of ontologies affect and is affected by the evolution of related artifacts, may help knowledge engineers in their tasks. An example of an artifact can either be: 1. an ontology O' that imports another ontology O, or 2. a search engine that query an ontology O.

To show that we build upon a comprehensive ontology evolution life-cycle that consists of five phases: (1) Detecting the need of evolution, (2) Suggesting changes to evolve ontologies, (3) validating the suggested changes, (4) Assessing the impact of the evolution, and (5) Managing the changes and keep track of them. With respect to this life-cycle, we present the following contributions:

Firstly, we introduce a definition for a situation to detect the need of ontology evolution (i.e., when an ontology O uses terms that have the namespace of another ontology O', then O' evolves). We list the set of cases that could occur during the evolution of the imported ontology. This definition could be used as fundamental for a methodological framework to maintain ontologies during the evolution process.

Secondly, we introduce an original approach for ontology enrichment using external knowledge bases: DBpedia, WikiData, and NELL. Our experiments showed that our system performs better than the current research work that target ontology enrichment using external knowledge bases.

Thirdly, we newly present a situation of ontology evolution, namely: *Ontology co-evolution*. We provide an exhaustive categorization of the different cases that could occur during this situation. We observe these cases over two ontology portals: the Linked Open Vocabulary and BioPortal. We conclude by showing that knowledge engineers could take advantage of a methodological framework based on our study for the maintenance of their ontologies.

Fourthly, we introduce a new categorization of ontology pitfalls: stand-alone ontology pitfalls, pitfalls in versioned ontologies and, pitfalls in ontology networks. We list a set of candidate pitfalls that are related to versioned ontologies and ontology networks. We evaluate the importance and potential impact of the candidate pitfalls by means of a web-based survey we conducted in the semantic web community. Moreover, we provide a set of recommendations to avoid or solve the different pitfalls we identified.

Finally, we conclude that knowledge engineers could take advantage of a methodological framework based on the thesis for the maintenance of their ontologies during their evolution process. This will reflect positively on the quality of the evolved ontologies.

Acknowledgements

I'm very excited to write this acknowledgment section. I'm truly grateful for the support and inspiration of many people during my Ph.D. studies.

I would like to thank my supervisors, Pierre Maret, Maxime Lefrançois, and Antoine Zimmermann, for their patient guidance, encouragement, and advice they have provided throughout my thesis. You have been always there guiding and answering all my queries in both scientific and personal manners. It was a great opportunity to work with you. I have learned a lot and gained great experience throughout my doctoral study.

I'm deeply grateful to my mother Amal Al-Qasem and my father Abdallah Alqawasmeh for their endless love, support and encouragement. There are not enough words to express my deep appreciation for all what have you done for me. I would not be where I'm today without you. My siblings: Khaled, Ahmad, and Anoud. Despite the distance between us, you have always been there when I needed you.

A special thank to Oudom Kem, Radha Krishna, Mehdi Benhani, Khadim Ndiaye, and Andrei Ushkov. Thanks for the good time we spent, the serious discussions and the "n'importe quoi" discussions, dinners, and for the many ski trips we had.

A special thank to Erika Koussi, Julie Van-eckhoutte, Daniëlle Hooijenga, Maísa Duarte, Dennis Diefenbach, José Gimenez Garcia, Vinicius De-Almeida, Ala'a Daoud, Hiba Qasir, Christian Moritz, Maria Mislene, Alexandra Herczku, Adrian Weinberg, Irati Malkorra, Valentine Delorme, Rada Deeb, Carlos Arango, Ali Haidar, and Mohammad Elawady. My stay in France would not be more pleasing without you being here.

I would like to thank my colleagues at Connected Intelligence Group: Kamal Singh, Fabrice Muhlenbach, Mohammad Noorani Bakerally, Flavien Balbo, Olivier Boissier, Francesco Antoniazzi, Edison Chung, and Gabriel Martins Lopes who have made valuable comments and suggestions during my thesis work.

I would like to thank the jury members: Prof. Rajendra Akerkar, Dr. Clément Jonquet, Prof. Sylvie Després, and Dr. Ana Roxin for their valuable reviews and comments.

I would like to express my great gratitude to France for the amazing time I'm spending here. Finally, I would like to acknowledge the French ministry of national education for funding my thesis work.

Contents

List of Figures

List of Tables

Listings

List of Abbreviations

IRI	**I**nternationalized **R**esource **I**dentifier
URL	**U**niform **R**esource **L**ocator
KB	**K**nowledge **B**ase
XML	The e**X**tensible **M**arkup **L**anguage
RDF	The **R**esource **D**escription **F**ramework
RDFs	The **R**esource **D**escription **F**ramework **S**chema
OWL	The **W**eb **O**ntology **L**anguage
Turtle	**T**erse **R**DF **T**riple **L**anguag**e**
LOV	The **L**inked **O**pen **v**ocabulary

List of Prefixes

The set of prefixes that are used over the thesis:

Namespace prefix	IRI	Ontology
childcare	http://childcare.fr/	Childcare (a specifically designed ontology for the thesis)
edu	http://education.fr/	Education (a specifically designed ontology for the thesis)
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#	RDF vocabulary
rdfs	http://www.w3.org/2000/01/rdf-schema#	RDF Schema vocabulary
owl	http://www.w3.org/2002/07/owl#	OWL vocabulary
dc	http://purl.org/dc/elements/1.1/	Dublin core
dcterms	http://purl.org/dc/terms/	DCMI metadata terms
vann	http://purl.org/vocab/vann/	A vocabulary for annotating vocabulary descriptions
xsd	http://www.w3.org/2001/XMLSchema#	XML schema datatypes in RDF and OWL
seas	https://w3id.org/seas/	Smart Energy Aware Systems
saref	https://saref.etsi.org/saref#	Smart Applications REFerence ontology
saref4city	https://saref.etsi.org/saref4city/	SAREF extension for Smart City
saref4ener	https://saref.etsi.org/saref4ener/	SAREF extension for the energy domain
saref4bldg	https://saref.etsi.org/saref4bldg/	SAREF extension for building devices
prov	https://www.w3.org/TR/prov-o/#	PROV-O: The PROV Ontology
dul	http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#	DOLCE+DnS Ultralite
cito	http://purl.org/spar/cito/	The Citation Typing Ontology
foaf	http://xmlns.com/foaf/0.1/	Friend of a friend

qb	http://purl.org/linked-data/cube	The data cube vocabulary
osspr	http://data.ordnancesurvey.co.uk/ontology/spatialrelations	The spatial relations ontology
scovo	http://purl.org/NET/scovo	The Statistical Core Vocabulary
prv	http://purl.org/net/provenance/ns	Provenance Vocabulary Core Ontology Specification
gn	http://www.geonames.org/ontology	The geonames ontology
dctype	http://purl.org/dc/dcmitype/	Dublin core metadata initiative
vcard	http://www.w3.org/2006/vcard/ns	vCard Ontology
schema	http://schema.org/#	Schema.org ontology
edm	http://www.europeana.eu/schemas/edm/	The Europeana Data Model vocabulary
mo	http://purl.org/ontology/mo/	The music ontology
org	http://www.w3.org/ns/org#	Core organization ontology
bio	http://purl.org/vocab/bio/0.1/	BIO: A vocabulary for biographical information
rdag1	http://rdvocab.info/Elements/	Open metadata registry
adms	http://www.w3.org/ns/adms#	Asset description metadata schema
mads	http://www.loc.gov/mads/rdf/v1#	MADS/RDF (Metadata Authority Description Schema in RDF)
txn	http://lod.taxonconcept.org/ontology/txn.owl#	TaxonConcept Ontology
voaf	http://purl.org/vocommons/voaf#	Vocabulary of a Friend (VOAF)
wo	http://purl.org/ontology/wo/	Wildlife ontology
rov	http://www.w3.org/ns/regorg#	Registered Organization Vocabulary
lingvo	http://www.lingvoj.org/ontology#	Lingvoj Ontology
locn	http://www.w3.org/ns/locn#	ISA Programme Location Core Vocabulary
frbrer	http://iflastandards.info/ns/fr/frbr/frbrer/	IFLA Element Sets (FRBRer model)
osadm	http://data.ordnancesurvey.co.uk/ontology/admingeo/	The administrative geography and civil voting area ontology
sio	http://semanticscience.org/resource/	semanticscience ontology

oborel	https://bioportal.bioontology.org/ontologies/OBOREL	Relations Ontology
clo	http://purl.obolibrary.org/obo/clo.owl	Cell Line Ontology
gfvo	https://bioportal.bioontology.org/ontologies/GFVO	Genomic Feature and Variation Ontology
plana	http://purl.obolibrary.org/obo/plana.owl	Planaria ontology
dso	http://purl.org/ontology/dso#	Document Service Ontology (DSO)
ons	https://bioportal.bioontology.org/ontologies/ONS	ONS: an ontology for a standardized description of interventions and observational studies in nutrition

Derived Publications

The following publications were derived out of the thesis, along with their abstracts:

1. Omar Qawasmeh: A Collaborative Framework for Ontology and Instance Data Co-evolution and Extraction. EKAW (Doctoral Consortium) 2018 [Qawasmeh, 2018].

   Abstract: Ontologies are at the heart of the semantic web, i.e., making data published on the web comprehensible to intelligent added value services. Ontologies consensual design ensures its usefulness and wide acceptance by service developers. The collaboration of ontology engineers, domain experts from multiple disciplines, and end-users is required to design, evolve, and populate ontologies with the instance data. In this thesis, we investigate in the different systems that are concerns in designing and evolving the ontologies in automatic or semi-automatic techniques. Moreover, we are working to provide a collaborative framework for ontology and instance data co-evolution and extraction.

   This publication presented the different research ideas behind this thesis work. It was published as a doctoral consortium paper at EKAW 2018. Doctoral consortium is an event where PhD students can present their research ideas, confront them with the scientific community, receive feedback from mentors and tie cooperation bounds. EKAW conference is ranked as **B** by CORE association.

2. Omar Qawasmeh, Maxime Lefrançois, Antoine Zimmermann, Pierre Maret: Computer-Assisted Ontology Construction System: Focus on Bootstrapping Capabilities. The Semantic Web - ESWC 2018 Satellite Events. **Acceptance rate: 26.8%.** [Qawasmeh et al., 2018].

   Abstract: In this research, we investigate the problem of ontology construction in both automatic and semi-automatic approaches. There are two key issues for the ontology construction process: the cold start problem (i.e., starting the development of an ontology from a blank page) and the lack of availability of domain experts. We describe a functionality for ontology construction based on the bootstrapping feature. For this feature, we take advantage of large public knowledge bases. We report on a comparative study between our system and the existing ones on the wine ontology.

   This publication presents the contributions in Chapter 2, Section 2.2. ESWC conference is ranked as **A** by CORE association.

3. Omar Qawasmeh, Maxime Lefrançois, Antoine Zimmermann, Pierre Maret: Observing the Impact and Adaptation to the Evolution of an Imported Ontology. Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (KEOD 2019). **Nominated for best paper award. Acceptance rate: 22%** [Qawasmeh et al., 2019].

   Abstract: Ontology evolution is the process of maintaining an ontology up to date with respect to the changes that arise in the targeted domain or in the requirements. Inspired by this definition, we introduce two concepts related to observe the impact and the adaptation to the evolution of an imported ontology. In the first one we target the evolution of an imported ontology (if ontology O uses ontology O', and then O' evolves). The second one targets the adaptation to the evolution of the imported ontology. Based on our definition we provide a systematic categorization of the different cases that can arise during the evolution of ontologies (e.g. a term t is deleted from O', but O continues to use it). We led an experiment to identify and count the occurrences of the different cases among the ontologies referenced on two ontology portals: (a) the Linked Open Vocabulary (LOV) ontology portal which references 648 different ontologies, 88 of them evolved. We identified 74 cases that satisfy our definition, involving 28 different ontologies. (b) the BioPortal which references 770 different ontologies, 485 of them evolved. We identified 14 cases that satisfy our definition, involving 10 different ontologies. We present the observation results from this study and we show the number of different cases that occurred during the evolution. We conclude by showing that knowledge engineers could take advantage of a methodological framework based on our study for the maintenance of their ontologies.

   This publication presents the contributions in Chapter 2, Section 2.1 and Chapter 3. This paper was nominated for best paper award. KEOD conference is ranked as **C** by CORE association.

4. Omar Qawasmeh, Maxime Lefrançois, Antoine Zimmermann, Pierre Maret: Pitfalls in Networked and Versioned Ontologies. Series: Communications in Computer and Information Science. Springer 2020.[1]

   Abstract: The listing and automatic detection of ontology pitfalls are crucial in ontology engineering. Existing work mainly focused on detecting pitfalls in stand-alone ontologies. Here, we introduce a new categorization of ontology pitfalls: stand-alone ontology pitfalls, pitfalls in versioned ontologies and, pitfalls in ontology networks. We investigate pitfalls in a situation of ontology co-evolution and we provide a systematic categorization of the different cases that could occur during the co-evolution process

[1] This research article is currently under review (June-2020)

over two ontology portals: the Linked Open Vocabulary and BioPortal. We also identify 9 candidate pitfalls that may affect versioned ontologies or ontology networks. We evaluate the importance and potential impact of the candidate pitfalls by means of a web-based survey we conducted in the semantic web community. Participants agreed that listing and investigating ontology pitfalls can effectively enhance the quality of ontologies and affect positively the use of ontologies. Moreover, the participants substantially agreed with the new categorization we proposed. We conclude by providing a set of recommendations to avoid or solve the different pitfalls we identified.

This publication presents the contributions in Chapter 4. It extends the previous publication with a pitfalls study over ontology networks and versioned ontologies. The paper is accepted to be published as a book chapter in Communications in Computer and Information Science (CCIS) series, Springer 2020. CCIS series has an **H-index of 45**.

To my family . . .

Introduction

Ontology engineering is a research field that targets the different methods and methodologies for building ontologies. Ontologies are at the heart of the semantic web, i.e., making data published on the Web understandable to intelligent added value services. Ontologies' consensual design ensures their usefulness and wide acceptance by service developers. Ontologies play nowadays an important role in organizing and categorizing data in information systems and on the web. This leads to a better understanding, sharing and analyzing of knowledge in a specific domain. However, domains' description are subject to changes, thus arises the need to evolve ontologies in order to have an adequate representation of the targeted domain. Based on [Zablith et al., 2015], we reformulate the definition of ontology evolution as *the process of maintaining an ontology up to date with respect to the changes that might arise in the described domain, and/or in the requirements.*

The usage of ontologies is increasing, so is the need of developing and maintaining them. Several methodologies were proposed to help controlling the evolution process of ontologies. The main aim of these methodologies is: 1. to help the knowledge engineers evolve the targeted ontologies, and 2. to prevent inconsistencies that may be caused as a consequence of the evolution process.

In this thesis we build upon the ontology evolution life-cycle proposed by [Zablith et al., 2015] (more details in Section 1.1.2), which consists of five phases:

1. Detect the need for evolution by either studying the users' behavior while using ontology-based systems or by analyzing the data sources that use the ontology.
2. Suggest changes to evolve the ontology by taking advantage of different types of resources, such as unstructured data (e.g. using some information extraction techniques to suggest changes by processing a raw text), or structured data (e.g. take advantage of external knowledge bases that can be publicly available, such as DBpedia, Wikidata).
3. Validate the suggested changes by running a set of syntax checking scripts before adopting them into the ontology.
4. Assess and study the impact of the evolution, by evaluating the impact of the changes on external artifacts that rely on the ontology (e.g. other ontologies, systems) and/or the cost of performing the changes.
5. Keep track of the implementation of the changes, in order to facilitate the management of the different versions that are created during the evolution process.

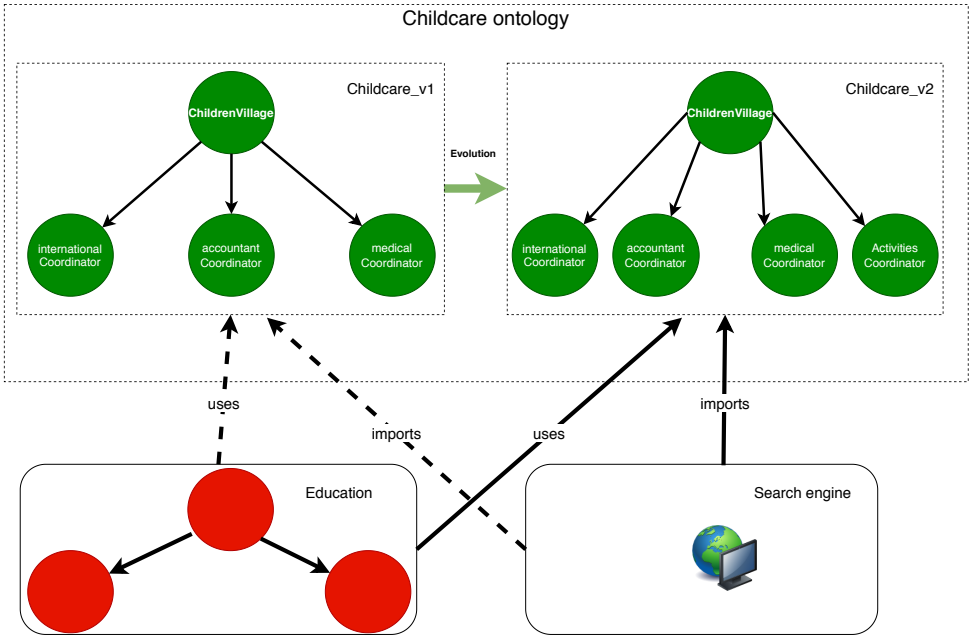


FIGURE 1: An illustrative example of a situation where the *Childcare* ontology evolves, while its first version is used by some external artifacts via *uses* and *imports* relationships. However, after noticing the evolution of *Childcare* v_1, the artifacts stopped using this version (dotted arrow) and start using *Childcare* v_2 (solid arrow)

Figure 1 illustrates an example that represents the five phases of the life-cycle. This example involves multiple ontologies that are connected with each other via two relationships (i.e., arrows inside the figure):

1. *uses*: happens when an ontology O uses a term t (that is, an IRI denoting an individual, a class or a property) that has the namespace of a different ontology O'.

2. *imports*: happens when an ontology O imports another ontology O', using the OWL importing mechanism.

The rest of the example can be interpreted as follows: The owners of the ontology *Childcare*, describing the child care domain, detect a change in the described domain for their ontology (Phase 1. Detecting changes). The owners collaboratively decide to add a new term `ex:activitiesCoordinator` and a set of axioms (Phase 2. Suggesting changes). They then check that the new additions do not introduce important semantic problems such as an ontology inconsistency, and decide to validate the change and create a new ontology version *Childcare* v_2 (Phase 3. Validating changes). The ontology *Childcare* is imported by some external ontologies, and implemented in a computer system. The owners investigate if problems may occur in these artifacts as an impact of the evolution (Phase 4. Assessing the evolution impact). Finally, the knowledge engineer keep track of changes and the different versions of the ontology

(Phase 5. Managing changes). Moreover, the described ontologies inside Figure 1 appeared in three different settings:

1. Stand-alone ontologies: ontologies that do not use any other ontologies, such as: the ontology *Childcare* v_1.
2. Versioned ontologies: happens whenever an ontology evolves to a newer version, such as: the evolution of the ontology *Childcare* v_1 to a newer version *Childcare* v_2.
3. Ontology network: happens whenever two or more ontologies are connected to each other via a relationship, such as: the ontologies *Childcare* v_1 and *Education* via the *uses* relationship.

Ontologies can be negatively affected by bad design decisions that could be made during the ontology development or the ontology evolution. Such bad design decisions are called *ontology pitfalls*. Ontology pitfalls may cause issues in the ontology itself, or in any external artifact that uses it (e.g. an ontology or a computer system). Previous work proposed a set of pitfalls for stand-alone ontologies. In this thesis we enrich the existing state of the art by introducing a set of pitfalls that could occur in versioned ontologies (i.e., when an ontology O evolves from O_{v1} to O_{v2}), and ontology networks (i.e., when an ontology O' uses another ontology O).

Motivating scenario

Let Amal be a knowledge engineer who develops an ontology for the child care domain, called *Childcare*. In the version $v_{1.1}$ of *Childcare*, created in May 2019, Amal used a specific term `edu:programmOfStudy` from another ontology called *Education* created in January 2019. *Childcare* contains at least a link to a term from *Education*. This creates a two ontologies network. In September 2019 the creators of the *Education* ontology released version $v_{1.2}$. Amal does not notice the evolution. Thus, she thinks that her ontology is still using $v_{1.1}$ version of the *Education* ontology. Inside this simple ontology network, several issues may arise:

- The term `edu:programmOfStudy` was removed from *Education*, however it is still used in *Childcare*. This has an impact over *Childcare* and Amal should adapt her ontology.
- New terms were introduced in *Education* $v_{1.2}$ (e.g. `edu:boarding school`). Amal should be made aware of these new terms as they may be useful to her ontology.

After noticing the evolution of the *Education* ontology, Amal decided to create $v_{1.2}$ of her *Childcare* ontology. Instead of creating $v_{1.2}$ in a manual way, Amal took advantage of existing knowledge bases to update her ontology with new classes, relationships and instances. *Childcare* $v_{1.2}$ was created in November 2017. During this versioning, several issues might arise, such as:

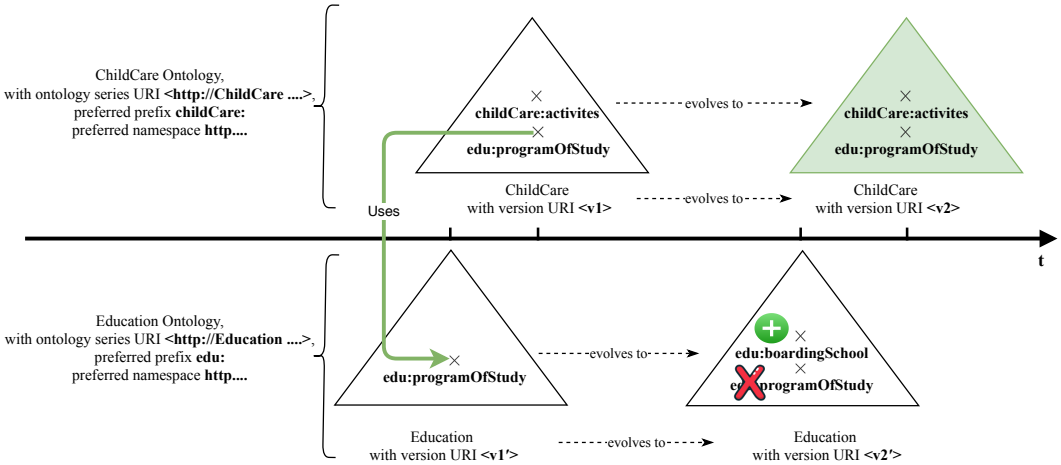


FIGURE 2: An illustrative figure for the motivating example. This figure presents the *Childcare* and *Education* ontologies. Where the first version of the *Childcare* ontology uses the term `edu:programOfStudy` that is defined inside the *Education* ontology. Then after a while, the term `edu:programOfStudy` was deleted in the second version of the *Education* ontology. As a consequence of this evolution, the ontology *Childcare* evolved to its second version. The different cases that might happen are described in the motivating scenario's section.

- The $v_{1.1}$ of *Childcare* ontology is not accessible any more by its IRI. This versioned ontology pitfall is caused by Amal, and she is the responsible of maintaining the *Childcare* ontology.
- Let us assume that the $v_{1.2}$ of the *Education* ontology is inconsistent, importing this ontology in the *Childcare* $v_{1.2}$ will make it become inconsistent too. This versioned ontology pitfall is caused by the owners of the *Education* ontology, and it is their responsibility to maintain their ontology.

If Amal publishes a bigger network of ontologies, the connections between these ontologies are expanding which makes it vulnerable to falling into some pitfalls. For example if the network contains an inconsistent ontology, then other ontologies that use this ontology will become inconsistent too.

What can be interesting for Amal, is to have a framework to manage these different cases in a (semi)-automatic technique.

Thesis hypothesis and research questions

Our main assumption in this thesis is the following:

Studying how the development and evolution of ontologies affect and is affected by the evolution of related artifacts, may help knowledge engineers in their tasks.

From this assumption, we derive two main research goals (RG), and associated research hypothesis (RH), and research questions (RQ):

RG.1 *To study the evolution need and evolution implementation of ontologies.*

RH.1 An ontology may need to evolve after some changes in some ontologies it uses.

RQ.1 How to detect the need of evolving an ontology through the observation of structural changes in the ontologies it uses?

RH.2 Using existing knowledge sources may help to develop and evolve ontologies.

RQ.2 How to take advantage of external knowledge bases to develop and evolve ontologies?

RG.2 *To study how the evolution and the quality of an ontology impacts the ontologies that use it.*

RH.3 Ontology portals may contain traces of incoherences in the evolution of ontologies that use one another.

RQ.3 How to detect and assess incoherences in the evolution of ontologies that use terms of one in another?

RH.4 Identifying pitfalls that affect ontology networks and versioned ontologies may help to design better ontologies.

RQ.4 What pitfalls affect ontology networks?

RQ.5 What pitfalls affect versioned ontologies?

Thesis structure

Chapter 1 Framework and Positioning: Ontology Evolution and Ontology Pitfall Analysis. This chapter contains an overview of the different topics that are related to the thesis:

- Section 1.1 presents an overview about ontologies, their different components and standards to describe them. It also defines ontology evolution and introduces the ontology evolution life-cycle we adopt in this thesis.
- Section 1.2 presents ontology pitfalls and state of the art on pitfall analysis.
- Section 1.3 presents existing methodologies that are used to develop and evolve ontologies.
- Section 1.4 presents a targeted literature review study for existing research works investigate similar research goals. We categorize these research items based on the life-cycle proposed by [Zablith et al., 2015].

Chapter 2 On Detecting the Need for Evolution and Enriching Ontologies using External Knowledge Bases. This chapter presents our contribution to the first research goal (i.e., RG 1. To study the evolution need and evolution implementation, of ontologies), where:

- Section 2.1 presents our definition to detect the need of ontology evolution by observing the evolution of an imported ontology.
- Section 2.2 presents our semi-automatic approach for ontology enrichment using external knowledge bases, along with our experimental evaluation.
- Section 2.3 presents our discussion points related to this chapter.

Chapter 3 Assessing the Impact of Ontology Evolution. This chapter presents our contributions to the second research goal (i.e., RG 2. To study how the evolution and quality of an ontology impacts those that use it), where:

- Section 3.1 presents different mechanisms and relationships to connect ontologies.
- Section 3.2 presents our definition to observe the adaptation to the evolution of an imported ontology.
- Section 3.3 presents our analyzing study to the adaptation of the evolution over two ontology portals.
- Section 3.4 presents our experimental evaluations results.
- Section 3.5 presents our discussion points related to this chapter.

Chapter 4 Pitfalls in Networked and Versioned Ontologies. This chapter presents our contributions to the second research goal (i.e., RG 2. To study how the evolution and quality of an ontology impacts those that use it), where:

- Section 4.1 presents our definition for the term “ontology networks”.
- Section 4.2 presents our definition for the term “versioned ontology”.
- Section 4.3 presents a list of candidate pitfalls that might occur inside versioned ontologies and ontology networks.
- Section 4.4 presents our experimental evaluation study to measure the importance and impact of the defined pitfalls
- Section 4.5 presents our discussion points related to this chapter.

Chapter 5 General Conclusion and Perspectives. This chapter concludes the thesis with a discussion about our findings along with a set of possible future work that can be further investigated.

Part I

Framework and Positioning

Chapter 1

Framework and Positioning: Ontology Evolution and Ontology Pitfall Analysis

Overview

This chapter contains an overview of the different topics that are related to the thesis:

- Section 1.1 presents an overview about ontologies, their different components and standards to describe them. It also defines ontology evolution and introduces the ontology evolution life-cycle we adopt in this thesis.
- Section 1.2 presents ontology pitfalls and the state of the art on pitfall analysis.
- Section 1.3 presents existing methodologies that are used to develop and evolve ontologies.
- Section 1.4 presents a targeted literature review study for existing research works investigate similar research goals. We categorize these research items based on the life-cycle proposed by [Zablith et al., 2015].

1.1 An overview on ontologies

"An *ontology* is an explicit specification of a conceptualization" [Gruber, 1993]. Ontologies play nowadays an important role in organizing and categorizing data in information systems and on the web, which leads to a better understanding, sharing and processing of knowledge in a specific domain. For example, going back to the motivating scenario, when creating the *Childcare* ontology, Amal made a clear description of her domain. This will effectively help her to clarify her understanding of the domain, and to better share the outcome ontology with her peers.

Ontologies share the following minimal set of components [Suárez-Figueroa et al., 2011]:

- Classes: represent the concepts of a domain. For example, the *Childcare* ontology can have the following classes: `Person`, `Student`, and `ProgramOfStudy`.
- Properties: represent the relations between the different concepts of the domain. For example, the *Childcare* ontology may use the property `participatesIn` between the concepts `Student` and `Activity`.
- Axioms: represent the facts of the domain, (i.e., sentences that are always true [Gruber, 1993]). For example the sentence "Each student is a person" can be an axiom of the *Childcare* ontology.
- Instances: represent the individuals that populate the classes and are linked by properties. For example: "Julie is a student" asserts that the instance "Julie" is in the class of `Student`, and "Julie participates to the semantic web class" asserts that the instance "Julie" is linked to the instance "the semantic web class" through the `participatesIn` property.

In the semantic web community, ontologies are implemented following different standards (e.g. RDF, RDFS, OWL 2) that have different expressiveness and inference mechanisms. Here we briefly mention three standards of the World Wide Web Consortium (W3C):

The Resource Description Framework (RDF)[Manola et al., 2004]

RDF is a standard model for describing web resources and interchanging their descriptions (i.e., resources can be anything). This helps to share the meaning of these resources between the different participants (e.g. computer systems, knowledge engineers). A key-point of RDF is to identify resources that are related to a specific domain, over the World Wide Web using Internationalized Resource Identifiers (IRIs), and later to describe these identified resources using properties and property values.

RDF allows linking the different concepts together by extending the links of the web into IRIs. Each relationship (that is identified by an IRI) links described concepts (i.e., subject and object). This gives what is known as an *RDF triple*. An RDF triple is a combination of three components:

1. Subject: can be an IRI or a blank node (i.e., presents a resource for which a URI or literal is not given).
2. Predicate: always an IRI.
3. Object, can be: IRI, literals, or blank nodes. Literals are used to present values that are not IRIs, such as strings, numbers, or dates. Literals are composed of a UTF-8 lexical form and a datatype. The datatype defines the meaning of every lexical form (e.g. `"26091991"^^<http://www.w3.org/2001/XMLSchema#integer>` has the meaning of the integer 26,091,991). In addition, literals with the datatype `<http://www.w3.org/2001/XMLSchema#langString>` can have a language tag (e.g. `"Paris is amazing"@en`).

Prefixes:
@prefix childcare: <http://childcare.fr/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

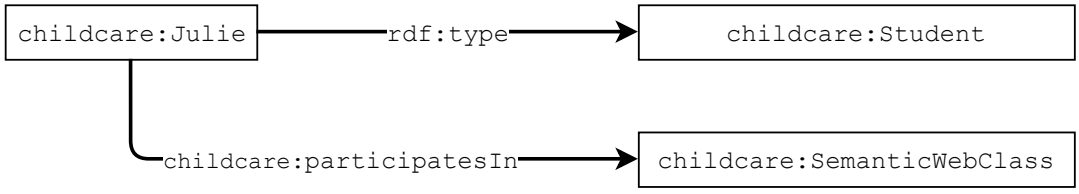


FIGURE 1.1: RDF graph that describes the sentence **Julie is a student and Julie participates in the semantic web class** along with the used prefixes

A set of RDF triples is called an RDF graph. Another important concept to be introduced is *RDF vocabulary*, which is a collection of IRIs intended to be used in an RDF graph. These IRIs often begin with a common substring known as namespace IRI. Some namespace IRIs are linked by convention with a short name known as a namespace prefix [Schreiber and Raimond, 2014].

The web service at `www.prefix.cc` is used by knowledge engineers to search for well known and widely used namespace prefixes. Table 1, on page xviii lists the set of prefixes that are used over the thesis.

Figure 1.1 shows a RDF graph for the information **Julie is a Student and Julie participates in the semantic web class**. This graph contains two triples:

1. `childcare:Julie rdf:type childcare:Student`, where all of subject, predicate, and object are IRIs.
2. `childcare:Julie childcare:participatesIn` "SemanticWebClass"^^xsd:string, where both of subject and predicate are IRIs, and the object is a literal.

RDF Schema (RDFS) [Brickley et al., 2014]

RDFS is a semantic extension of RDF. It offers mechanisms to describe groups of resources and the relationships between these resources. The groups of related resources are called classes (`rdfs:Class`), where a member of a class is called an instance. RDFS supports creating hierarchies between the classes by using `rdfs:subClassOf`, and between the properties by using `rdfs:subPropertyOf`.

In addition, the relations between the different resources can be described using a set of predefined properties, such as:

- `rdfs:range`: used to specify that the values of a property are instances of one or more classes. For example, let us assume that the property `childcare:participatesIn` has the class `childcare:Activity` as a range. This indicates that all values of the `childcare:participatesIn` property are members of `childcare:Activity` class.

- `rdfs:domain`: used to specify that a resource that has a given property is an instance of one or more classes. For example, let us assume that the property `childcare:participatesIn` has the class `childcare:Student` as a domain. This indicates that any resource that has the `childcare:participatesIn` property is an instance of the class `childcare:Student`.
- `rdf:type`: used to specify that a resource is an instance of a class. For example, to specify that the instance `childcare:Julie` is of type `childcare:Student` class, the property `rdf:type` is used: `childcare:Julie rdf:type rdfs:Student`.
- `rdfs:subClassOf`: used to specify that all instances of one class are also instances of another. For example, to indicate that every student is a person: `childcare:Student rdfs:subClassOf childcare:Person`.
- `rdfs:subPropertyOf`: used to specify that all resources related by one property are also related by another. For example the properties `childcare:participatesIn` and `childcare:successfullyPasses` are related in the sense that every student who `childcare:successfullyPasses` an activity should be `childcare:participatesIn` in the first place.
- `rdfs:label`: used to provide human-readable (multilingual) version of a resource name, such as: `childcare:Student rdfs:label` "Étudiant"@fr.
- `rdfs:comment`: used to provide human-readable description of a resource, such as: `childcare:Student rdfs:comment` "This class represents the students ..."@en.

Figure 1.2 illustrates an RDFS schema that models the knowledge that "Each student is a person. Each student participates in an activity" inside the *Childcare* ontology. The following can be noted, firstly:

1. There exist three classes defined inside the ontology *Childcare*: `childcare:Student`, `childcare:Person`, `childcare:Activity`. They are all linked to the class `rdfs:Class` using the `rdf:type` property.
2. One property (i.e., `childcare:participatesIn`) is defined inside the *Childcare* ontology, where the domain and range for this property are the classes `childcare:Student` and `childcare:Activity` respectively.

Secondly, considering the RDF graph from Figure 1.1 (lower part of Figure 1.2), i.e., "Julie is a student and Julie participates in the semantic web class", the following facts can be inferred (marked in red color):

- `childcare:Julie rdfs:subClassOf childcare:Student` (given fact). `childcare:Student rdf:type childcare:Person` (given fact). Then Julie is also a person; `childcare:Julie rdf:type childcare:Person` (inferred).
- `childcare:Julie childcare:participatesIn childcare:SemanticWebClass` (given fact). `childcare:participatesIn`

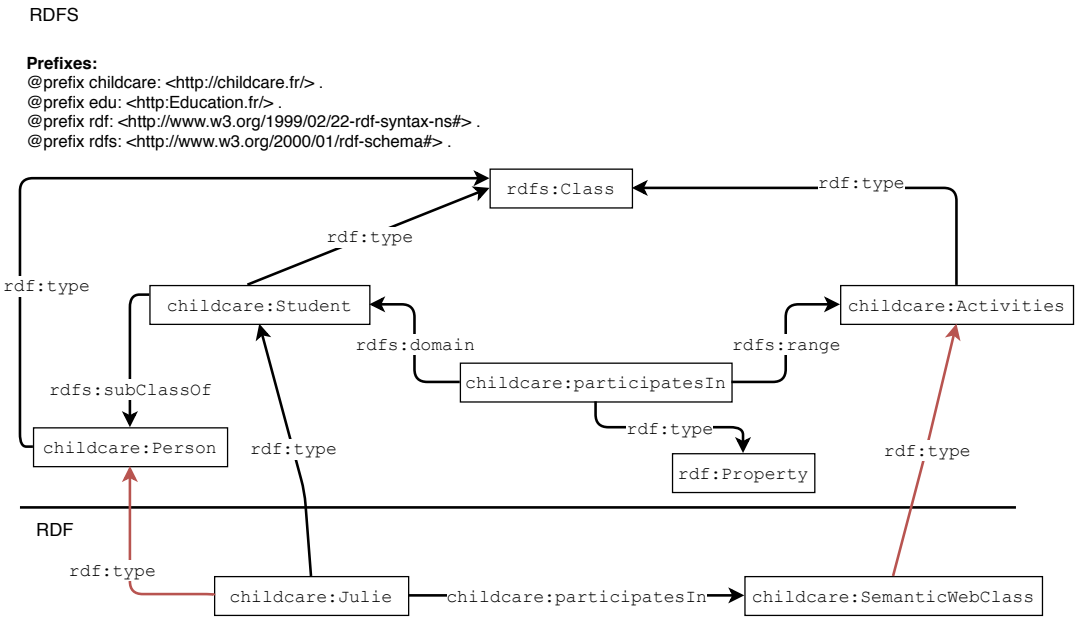


FIGURE 1.2: An RDFS graph that describes the piece of information **Each student is a person. Each student participates in an activity**. The red links are inferred using RDFS semantics.

`rdfs:range childcare:Activity` (given fact). Then `childcare:SemanticWebClass rdf:type childcare:Activity` (inferred).

OWL 2 [Motik et al., 2009]

OWL 2 is the standard ontology language for the semantic web with formally defined meaning. OWL 2 is an extension of the Web Ontology Language (OWL). Before investigating further in OWL 2, we briefly define four main notations that are related to OWL:

- Ontology declaration: the triple `X rdf:type owl:Ontology`, is used to define OWL ontologies, where `X` is the subject that represent the ontology IRI. For example: the triple `<http://childcare.fr/> rdf:type owl:Ontology` declares that `<http://childcare.fr/>` is an OWL ontology.
- Re-usability of ontologies: one of the main advantages of ontologies is the re-usability, instead of defining a new set of terms (i.e., resources) knowledge engineers could reuse existing ontologies to save time and effort. There exist two main ways to reuse ontologies:
  - Importing an ontology using the `owl:imports` property. For example: `<http://childcare.fr/> owl:imports <http://education.fr/>` will include all the axioms and terms from `<http://education.fr/>` ontology into the `<http://childcare.fr/>` ontology.

  - Re-use a specific term from a specific ontology using its IRI. This situation will not ensure to use the axioms of the external ontology, but only to re-use the imported term. For example, in Figure 2 the ontology *Childcare* uses a specific term `edu:ProgramOfStudy` that is defined in the *Education* ontology.

- OWL versioning: IRIs are used to identify ontologies (mainly named ontology IRI). In case of having an ontology with multiple versions, each version could use an IRI to identify it (mainly named version IRI). It is recommended that the ontology IRI is different from a version IRI.[1] In addition to these two types of IRIs, the following properties can be used to present more information:
  - `owl:versionInfo`, where it has the version information of the ontology in its object, such as: `<http://childcare.fr/> owl:versionInfo` "1.0" .
  - `owl:versionIRI`, where it used to identify the version IRI of an ontology, such as: `<http://childcare.fr/> owl:versionIRI <http://childcare.fr/v1.1>`
  - `owl:priorVersion`, where it contains a reference to another ontology in its object to specify that an ontology is a prior version to another, such as: `<http://childcare.fr/v1.2> owl:priorVersion <http://childcare.fr/v1.1>`. This information may be used by computer systems to help organizing the ontologies by their versions.
  - `rdfs:seeAlso`, used to specify more information about a subject, such as: `childcare:Julie rdfs:seeAlso <http://childcare.fr/Julie-info>`.
  - `rdfs:isDefinedBy`, used to specify that a subject is defined by an object, such as: `childcare:participatesIn rdfs:isDefinedBy <http://childcare.fr/>`.

OWL 2 is designed to formulate, exchange and reason on the knowledge of a specific domain. Three basic notions are used to represent knowledge in OWL 2:

1. Axioms: statements (i.e., pieces of knowledge) that are taken to be true.
2. Entities: the elements that are used to describe objects, i.e., classes, properties, and individuals.
3. Expressions: the combinations of two or more entities to produce complex representation from the basic ones. There are two types of expressions:
   (a) *Property expressions:* properties can be used to define property expressions, two situations can occur:
      i. Object properties are used to form *Object property expressions*, which helps to represent the relationship between pairs of resources. For example, the object property `childcare:participatesIn` can

[1] For further details, interested readers may refer to https://www.w3.org/TR/2012/REC-owl2-syntax-20121211/.

be considered as an object property as it connects two resources together `childcare:Student` and `childcare:Activity`. In addition, more semantics can be added to properties by using: `rdfs:subPropertyOf` (i.e., used to describe properties hierarchies), `owl:inverseOf` (i.e., to define a property as an inverse to another one), `owl:equivalentProperty` (i.e., to say that two properties are equivalent).

ii. Data properties are used to form *Data property expressions*, which help to represent the relationship between a resource and a literal (e.g. strings, numbers, date). For example, the data property `ex:hasBirthDate` relates the resource `childcare:Student` to a date (e.g. `xsd:date`).

(b) *Class expressions:* Classes and property expressions are used to formulate class expressions (a.k.a. descriptions or complex classes). Class expressions describe individuals by setting a set of conditions on the individuals' properties, where each individual that satisfy these conditions is considered as instance of the class [Motik et al., 2009, §8]. We provide below some examples of class expression constructors. Table 1.1 gives examples for these class expressions over the *Childcare* ontology:

i. Propositional connectives and enumeration of individuals: used to define complex classes by applying logical constructors, such as:

- `owl:intersectionOf` which can be used to define that a class C results from the intersection of all individuals of two classes or more.
- `owl:unionOf` which can be used to define that a class C results from the union of all individuals of two classes or more.

ii. OWL property restrictions: used to describe complex classes via defining restrictions on the range of a property (i.e., Object properties) or to define the expected data values of a property (i.e., Data property). OWL 2 has the following restrictions:

- Restrictions on values: (1) `owl:hasValue`: to describe the individuals that are connected by an object property expression to a particular individual, (2) `owl:allValuesFrom`: to fix the range of a property to a specific class C (strict binding, all values should be of type class C), and (3) `owl:someValuesFrom`: to specify that the range of a property should have at least one member of a class C.
- Restrictions on cardinality: it contains all those individuals that are connected by a property to fixed (`owl:cardinality`), lower bounds (`owl:minCardinality`) or upper bounds (`owl:maxCardinality`) number of individuals that are instances of a specific class.

TABLE 1.1: Examples of some class expressions for the *Childcare* ontology

Expression	**Feature**	**Example**	**Example written in Turtle 1.1**

Propositional connectives and enumeration of individuals	`owl:intersectionOf`	The intersection of all individuals of the classes `ex:Woman` and `ex:Parent` gives us the class `ex:Mother`.	`ex:Woman a owl:Class .`
			`ex:Parent a owl:Class .`
			`ex:Mother a owl:Class;`
			`owl:equivalentTo [`
			`owl:intersectionOf(ex:Woman ex:Parent)`
			`] .`

	`owl:unionOf`	The union of all individuals of the classes `childcare:Sport` and `childcare:Art` give us the class `childcare:Activity`.	`childcare:Activity a owl:Class ;`
			`owl:equivalentClass [`
			`owl:unionOf ( childcare:Sport childcare:Art )`
			`] .`

Property restrictions on values	`owl:hasValue`	The class "JulieCourses" is a subclass for every course that has been participated in by one individual "Julie".	`childcare:JulieCourses a owl:Class;`
			`rdfs:subClassOf [`
			`rdf:type owl:Restriction ;`
			`owl:onProperty childcare:participatesIn ;`
			`owl:hasValue childcare:Julie`
			`].`

	`owl:allValuesFrom`	For each Activity, all participants must be Students.	`childcare:Activity a owl:Class;`
			`rdfs:subClassOf [`
			`rdf:type owl:Restriction;`
			`owl:onProperty childcare:participatesIn ;`
			`owl:allValuesFrom childcare:Student`
			`] .`

	`owl:someValuesFrom`	Each Activity must have at least one participant that is a Student	`childcare:Activity a owl:Class;`
			`rdfs:subClassOf [`
			`rdf:type owl:Restriction;`
			`owl:onProperty childcare:participatesIn ;`
			`owl:someValuesFrom childcare:Student`
			`] .`

Property restrictions on cardinality	`owl:cardinality`	Each student must participate in exactly 4 activities	`childcare:Student a owl:Class;`
			`rdfs:subClassOf [`
			`rdf:type owl:Restriction ;`
			`owl:onProperty childcare:participatesIn;`
			`owl:cardinality "4"^^xsd:nonNegativeInteger`
			`] .`

	`owl:minCardinality`	Each student must participates in 4 activities at least	`childcare:Student a owl:Class;`
			`rdfs:subClassOf [`
			`rdf:type owl:Restriction ;`
			`owl:onProperty childcare:participatesIn;`
			`owl:minCardinality "4"^^xsd:nonNegativeInteger`
			`].`

	`owl:maxCardinality`	Each student must participates in 4 activities at most	`childcare:Student a owl:Class;`
			`rdfs:subClassOf [`
			`rdf:type owl:Restriction ;`
			`owl:onProperty childcare:participatesIn;`
			`owl:maxCardinality "4"^^xsd:nonNegativeInteger`
			`].`

OWL 2 presents an expressive language to represent complex knowledge about things using more expressive features, such as asymmetric, reflexive and disjoint properties. However, there exist some fragments (i.e., profiles) that are trimmed down from OWL 2 to trade some expressive power in favor of the reasoning efficiency, we shortly present the three profiles of OWL 2 in addition to OWL 1 DL profile that is associated with OWL 1:

- OWL 2 EL:[2] suitable for applications that use ontologies with very large numbers of classes and properties. OWL 2 EL manage the expressive power for such ontologies. Moreover, it manages and checks different tasks related to ontologies in polynomial time, such as: ontology consistency, class expression subsumption, and instance checking.
- OWL 2 QL:[3] suitable for situations where the query answering is the most important task to deal with, mainly with applications that use enormous volumes of instance data.
- OWL 2 RL:[4] suitable for applications that need scalable reasoning without losing too much expressive power.
- OWL 1 DL (from OWL 1):[5] a fragment from OWL 1, which provides a set of constraints on the use of OWL 1, such as: 1. object properties and datatype properties are always required to be disjoint, 2. a pairwise separation between the different resources (classes, object properties, annotation properties, etc.) is required, e.g. a class cannot be at the same time an individual.

Figure 1.3 extends the previous example (Figure 1.2) with some of the notations that are related to OWL 2, these are:

1. Each student participates in at least 4 activities.
2. Activity class is the union of two classes, Sport and Art.
3. For each activity, all participants must be students.

Serialization formats Ontologies expressed using these different semantic web standards (RDF, RDFS, and OWL 2) can be serialized using different syntaxes. This gives more liberty for developers to choose among these different standards based on their tasks and requirements. Here, we illustrate with some examples these serializations:

1. RDF/XML [Beckett and McBride, 2004] is the first serialization to write RDF graphs using the XML format. For example the class `childcare:Student` can be represented as: `<owl:Class rdf:about="http://childcare.fr/Student">`.

[2] https://www.w3.org/TR/owl2-profiles/#OWL_2_EL
[3] https://www.w3.org/TR/owl2-profiles/#OWL_2_QL
[4] https://www.w3.org/TR/owl2-profiles/#OWL_2_RL
[5] https://www.w3.org/TR/owl-ref/#OWLDL

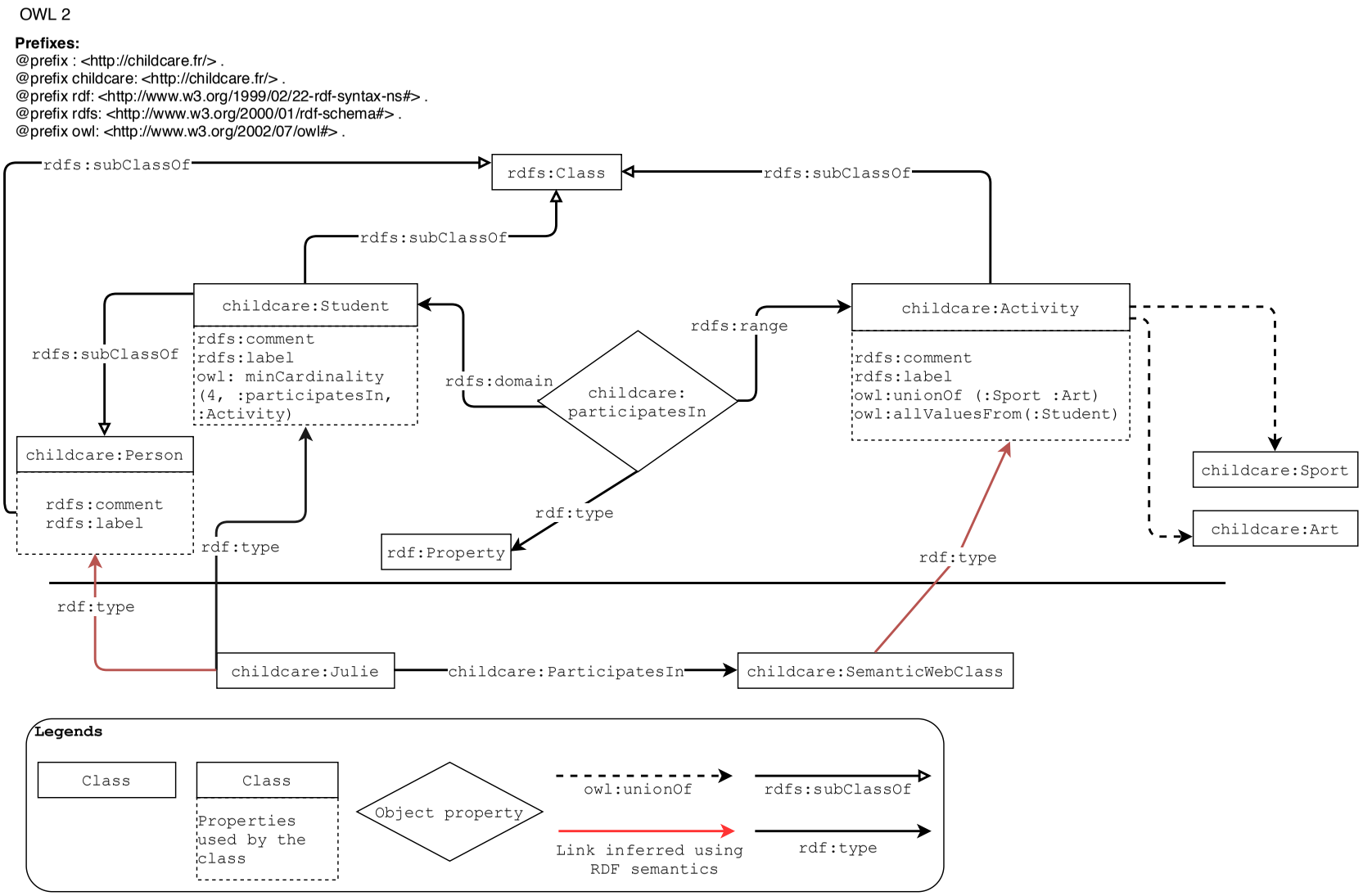


FIGURE 1.3: OWL 2 graph that describes the piece of information **Each student is a person. Each student participates in at least 4 activities. For each activity, all participants must be students The red links are inferred using RDFS semantics.** The legends representation is inspired by ETSI TS 103 673 V1.1.1 [ETSI, 2019b] technical report.

2. The Terse RDF Triple Language Turtle [Beckett et al., 2014] is used to write RDF graphs in a compact textual format (human-readable). The Turtle representation will be used over the thesis. Listing 1.1 presents the Turtle representation of the graph represented in Figure 1.3.

3. N-Triples [Beckett, 2014] is a simplified subset of Turtle. It is harder to read, however it is easier to parse by computer systems. For example, the class `childcare:Student` can be represented as: `<http://childcare.fr/Person> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#Class> . .`

SPARQL: an RDF query language [Harris et al., 2013]

SPARQL is an "RDF query language is a semantic query language for databases able to retrieve and manipulate data stored in Resource Description Framework (RDF) format" [Segaran et al., 2009, Rapoza, 2006]. SPARQL has a wide number of features and variations that can be used in many use cases. Here, we mention only some examples of these features:

1. In its query syntax, SPARQL uses terms that are defined in RDF, such as: IRI, language tags, and literals.

2. SPARQL defines different query forms for various purposes:

   (a) SELECT: used to extract values from a SPARQL endpoint (i.e., a service that takes a SPARQL query as an input and return results), then the results are returned in the form of: IRI, literals, and blank nodes. This type will be used over the thesis work.

   (b) CONSTRUCT: used to create an RDF graph based on the query criteria.

   (c) ASK: used to test whether the RDF graph contains some data of interest based on a True/False qyery.

   (d) DESCRIBE: used to generate an RDF description of a resource or a set of resources.

3. SPARQL provides filtering technique that is able to restrict the queries' results based on predefined aspects using the keyword FILTER.

4. SPARQL supports aggregation through using one of the following predefined aggregates:

   (a) COUNT: to count the number of occurrences of a given expression inside an aggregate group.

   (b) SUM: to return the sum of a set of numeric values within an aggregate group.

   (c) MIN: to return the minimum value inside an aggregate group.

   (d) MAX: to return the maximum value inside an aggregate group.

   (e) AVG: to calculate the average of values inside an aggregate group.

Listing 1.1: The RDF graph from Figure 1.3 presented in Turtle syntax

```
@prefix childcare: <http://childcare.fr/> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix vann: <http://purl.org/vocab/vann/> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://childcare.fr/> a owl:Ontology ;
                       dc:title "Child care ontology"@en ;
                       dc:description "This ontology describe the childcare domain."@en ;
                       dc:publisher <https://www.someonecool.org/> ;
                       dcterms:creator <http://www.qqn-cool.com/foaf.rdf#me> ;
                       owl:versionInfo "1.0" ;
                       vann:preferredNamespacePrefix "childcare" ;
                       vann:preferredNamespaceUri "https://childcare.fr" .
################################
# # Object Properties
################################
# http://childcare.fr/participatesIn
childcare:participatesIn a owl:ObjectProperty ;
  rdfs:domain childcare:Student ;
  rdfs:range childcare:Activity ;
  rdfs:comment "A relationship between the class Child and the class Activities"@en ;
  rdfs:label "participates in"@en .

################################
# # Classes
################################
# http://childcare.fr/Person
childcare:Person a owl:Class ;
  rdfs:comment "The class Person"@en;
  rdfs:label "Person"@en .

# http://childcare.fr/Student
childcare:Student a owl:Class ;
  rdfs:subClassOf childcare:Person , [
  rdf:type owl:Restriction;
  owl:onProperty childcare:participatesIn ;
  owl:minCardinality "4"^^xsd:nonNegativeInteger ; ] ;
  rdfs:comment "The class student"@en;
  rdfs:label "Student"@en ;
  rdfs:label "L'etudiant"@fr .

# http://childcare.fr/Art
childcare:Art a owl:Class ;
rdfs:comment "The mind exertion activities"@en;
rdfs:label "Art"@en .

# http://childcare.fr/Sport
childcare:Sport a owl:Class ;
rdfs:comment "The physical exertion activities"@en;
rdfs:label "Sport"@en ;
rdfs:label "Le sport"@fr .

# http://childcare.fr/Activities
childcare:Activity a owl:Class ;
  rdfs:comment "The activities that a child can participate in"@en;
  rdfs:label "Activities"@en ;
  owl:equivalentClass [
  owl:unionOf (childcare:Sport childcare:Art) ] .
```

(f) GROUP_CONCAT: to provide a string concatenation for the values inside an aggregate group.

(g) SAMPLE: to return a random value from the set of values inside an aggregate group.

Listing 1.2 presents a simple SPARQL query to retrieve all the triples that exist in the knowledge graph (Listing 1.1) and have their object's value written in French.

LISTING 1.2: A SPARQL query to retrieve all the triples that exist in the knowledge graph described in Listing 1.1 which have a label written in French language

```
PREFIX voaf:<http://purl.org/vocommons/voaf#>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX dcat: <http://www.w3.org/ns/dcat#>
PREFIX text: <http://jena.apache.org/text#>
SELECT * WHERE {
     ?subject ?property ?object .

    FILTER langMatches( lang(?object), 'fr')
}
```

1.1.1 Best practices for publishing ontologies

The developed ontologies can be published online so that they can be accessible. There exist some requirements (i.e., best practices) to be followed during the publication procedure of ontologies, described by: [Berrueta et al., 2008, Janowicz et al., 2014]. Bernard Vatant proposed five requirements to publish ontologies over the web.[6] These requirements are used as a foundation stone to create the Linked Open Vocabulary (LOV) ontology portal [Vandenbussche et al., 2017]. The main requirements can be listed as:

REQ.1 The ontology should have a stable IRI. Having persistent IRIs help to stabilize and maximize the reuse of ontologies. In contrast, using a non-persistent IRI causes problems, as the ontology becomes inaccessible by the external artifacts that are using it. For example, the *Childcare* ontology has an IRI at http://childcare.fr/. It is recommended that the ontology IRI remains the same, and it should not be changed.

REQ.2 Knowledge engineers should provide a human-readable documentation along with basic metadata information such as date of creation, publisher, last modification, version number, etc. The different metadata information helps in managing the different versions of ontologies which leads to enhance the usage of ontologies in different tasks. Some solutions are proposed to keep track of the metadata in automatic way. For instance, the Shapes Constraint

[6] https://bvatant.blogspot.com/2012/02/is-your-linked-data-vocabulary-5-star_9588.html

Language (SHACL)[7] can be used to check that a predefined set of metadata are included in an ontology.

REQ.3 Multi-lingual labels and descriptions should be added. If the ontology is created using one language (e.g. English language), it may restrict the ontology of being used by global users (i.e., users that do not know the language, hence they prefer not to use the ontology). It is recommended to have at least the labels and comments in several languages, language tags can be used for this purpose (e.g. @ar for Arabic, @fr for French and @en for English).

REQ.4 The ontology should have a persistent namespace IRI. When accessing this namespace IRI, it should be used to provide the ontology as a formal file and human-readable documentation.

REQ.5 Existing ontologies should be reused instead of creating new ones and reinventing the wheel. Re-usability is considered as a good practice as it creates connections between the different ontologies, and saves time and energy for the knowledge engineers during the development process of their ontologies. For example, in Figure 1 instead of creating a new terms to describe coordinator offices organizations, the owners of *Education* ontology use the terms that are defined in the *Childcare* ontology.

These main requirements can be generalized to publish ontologies online. However, they can be adapted in case of working on specific project, ontologies, etc. Authors in [ETSI, 2019a] published specific requirements that should be followed to publish ontologies. They identified a set of best practices to be followed:

1. The ontologies should be valid OWL DL ontologies.
2. Each ontology should be versioned based on a specific versioning mechanism described in [Motik et al., 2009]
3. Each ontology should be imported based on a specific importing mechanism described in [Motik et al., 2009].
4. Each ontology should be accessible at its IRI.
5. Each ontology should have a persistent IRI.
6. The terms defined by an ontology should be defined in the namespace of the ontology.
7. Each term's description should be accessible by its IRI.
8. Different representations of ontologies should be served depending on what is requested for. This can be implemented using content negotiation, which is an HTTP mechanism that helps to serve different versions or formats of the same web document. For example, if an agent (e.g. web browser) requests the following URL `https://saref.etsi.org/saref4ener/latest/` without specifying the preferred data format, it will be intuitively redirected to

[7] https://www.w3.org/TR/shacl/

https://saref.etsi.org/saref4ener/latest/saref4ener.html, however if the agent specifies in the HTTP request that it wants (Text/Turtle) representation, then the URL will be redirected to https://saref.etsi.org/saref4ener/latest/saref4ener.ttl.

As the number of ontologies is increasing over the web, ontology portals are developed to facilitate the retrieval process of the published ontologies by grouping them based on their described domain. As any data, ontologies need to follow FAIR principles [Wilkinson et al., 2016, Jonquet et al., 2018b], i.e., findable, accessible, interoperable, and re-usable [Jonquet, 2018]. Mainly, the grouping of ontologies is done using features, such as: creation date, authors, number of published versions. Moreover, ontology portals are used to group the different versions of each ontology along with the metadata that describe each version (e.g. creation data, publisher, contributors). These metadata is described using different existing vocabularies, such as: DCAT and Dublin Core that are considered as the most used vocabularies to describe ontologies [Toulet et al., 2018].

In this thesis, we will take advantage of two ontology portals: 1. the linked open vocabulary (LOV) portal [Vandenbussche et al., 2017], and 2. the NCBO BioPortal. [Whetzel et al., 2011] They are well known repositories, rich with metadata, and they reference a large number of ontologies that are available on the Web.

1.1.2 Ontology evolution

As mentioned earlier, ontologies describe a specific domain. As the domains' descriptions are subject to changes, thus arises the need to evolve ontologies in order to have an adequate representation. Based on [Zablith et al., 2015], we reformulate the definition of ontology evolution as *the process of maintaining an ontology up to date with respect to the changes that might arise in the described domain, and/or in the requirements*. Zablith et al. [Zablith et al., 2015] studied the different methodologies and approaches to evolve ontologies, and they defined a comprehensive life-cycle of ontology evolution (Figure 1.4).

In this thesis, we adopt this life-cycle, and it will be used later to categorize the set of related work in Section 1.4. The life-cycle is described as:

Phase 1. Detect the need for evolution: by either studying the users behavior while using systems that rely on an ontology or by analyzing the data sources that use the ontology. Some examples of approaches that are designed to solve this need are given in: [Stojanovic, 2004, Noy et al., 2006, Javed et al., 2011, Pruski et al., 2011, Hartung et al., 2013].

Phase 2. Suggest changes to evolve the ontology: Different text mining and information retrieval techniques are used to suggest changes from unstructured data sources [Cimiano and Völker, 2005, Maynard et al., 2009, Bloehdorn et al., 2006, Novacek and Handschuh, 2007, Zablith et al., 2009]. Other techniques rely on structured data sources [Kong et al., 2006, Moldovan

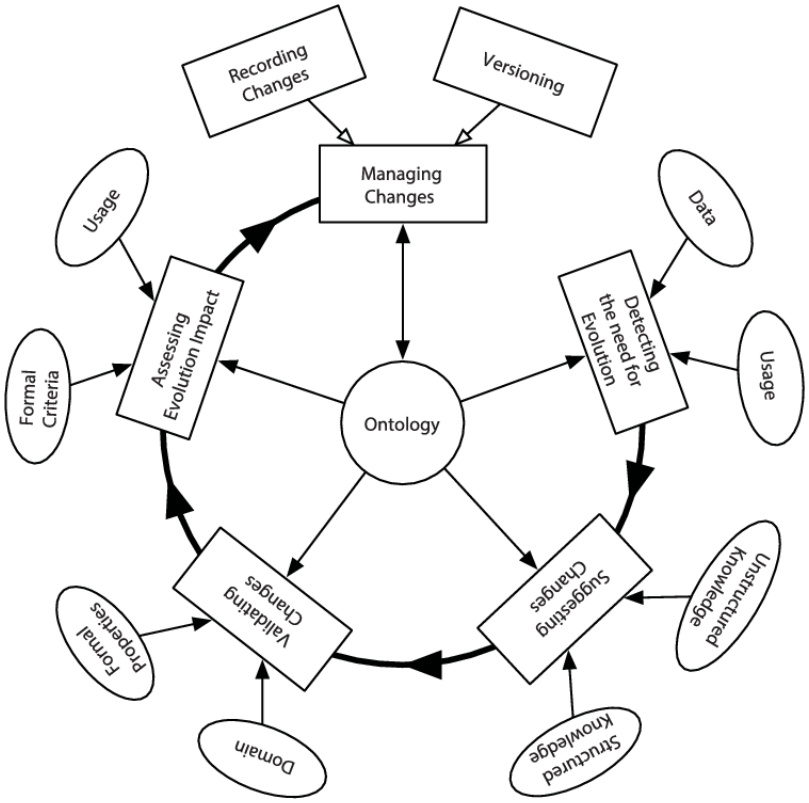


FIGURE 1.4: Ontology evolution life-cycle proposed by [Zablith et al., 2015]

and Girju, 2000, Agirre et al., 2000, Kietz et al., 2000, Cahyani and Wasito, 2017].

Phase 3. Validate the suggested changes before adopting them into the ontology. Two levels of validation are introduced in [Zablith et al., 2015]:

- Domain-based validation: to use existing domain data to evaluate the suggested changes before applying them to the ontology. Some systems that rely on domain data to validate the changes are given in: [Cimiano and Völker, 2005, Maynard et al., 2009, Novacek and Handschuh, 2007, Zablith et al., 2010]
- Formal properties-based validation: to use formal techniques to ensure that the proposed change does not break the required constraints, such as the consistency of the ontology. Some systems that use the formal properties-based validation are given in: [Konstantinidis et al., 2008, Papavasileiou et al., 2013, Rieß et al., 2010]

Phase 4. Assess and study the impact of the evolution: this step measures the impact of evolution on external artifacts that rely on the evolved ontology (e.g. other ontologies that uses or import the evolved ontology, systems that query the evolved ontology, data sets that are described using the evolved ontology), and/or the cost of performing a given change.

Phase 5. Manage the changes to keep track of the performed changes to facilitate to handle the different versions that are created during the evolution process. [Noy et al., 2003] proposed a system that implements the management of the performed changes.

1.2 Existing analyses of ontology pitfalls

Following good practices during the development process of ontologies helps to increase their quality, which reflects in their usage [Bernaras et al., 1996, Doran et al., 2007]. Reusing an ontology is considered as a good practice [Gyrard et al., 2015, Noy et al., 2001, Suárez-Figueroa et al., 2012a] that leads to the creation of connections between different ontologies, which results in having networks of ontologies.

In the field of semantic web, several researchers in the domain of ontology evaluation used the term "pitfall" to refer to the set of mistakes or errors that can be made during the development or usage of ontologies. These pitfalls may cause abnormal behaviors for the ontologies, such as: breaks in the connections between ontologies, or wrong results for search queries for these ontologies.

Several researchers worked on observing (e.g. [Gaudet and Dessimoz, 2017, Vigo et al., 2014]) or listing (e.g. [Poveda-Villalón et al., 2014]) the set of pitfalls that might affect stand-alone ontologies and networked ontologies ([Sabou and Fernández, 2012]). Hence, here we present the set of approaches that observe and list the set of pitfalls in different scenarios.

Sabou and Fernandez [Sabou and Fernández, 2012] provide a methodological guidelines for evaluating both stand-alone ontologies and ontology networks. Their methodology relies on selecting a targeted ontology component to evaluate based on a predefined goal. Authors defined a workflow to evaluate the targeted ontologies. It consists of the following tasks:

Task 1. Select an ontology individual component to be evaluated. The selection is recommended to be done based on two criteria: (a) the importance of the component for the overall ontology network, and (b) the possibility of evaluating the selected elements based on some existing frameworks, guidelines, etc.

Task 2. Select an evaluation goal and an evaluation approach. Authors defined four evaluation goals that are associated to several evaluating approaches. The goals are:

- Domain coverage, i.e., whether the ontology covers the topic domain or not. To evaluate this goal, a comparison technique between the ontology and existing frames of references can be applied, such as comparing the ontology with a gold standard ontology (e.g. [Maedche and Staab, 2002]) or with a reference dataset that is representative to the domain (e.g. [Alani et al., 2006]).
- Quality of modeling, i.e., does the ontology development process follow the ontology modeling best practices or not. This goal is associated with the quality of the ontology and its correctness in both semantic and syntax manners. Several evaluating approaches can be applied to verify this goal, such as [Tartir et al., 2005, Burton-Jones et al., 2005].
- Suitability for an application or a task, i.e., whether the ontology is suitable to be used for a certain application or for a certain task.
- Adoption and use, i.e., whether the ontology has been imported as a part of other ontologies or it has been rated by users.

Task 3. Identify a frame of reference and an evaluation metric, and concretely select the ingredients of the evaluation, mainly, for selection: (a) frame of reference, i.e., a baseline value that the ontology should be compared to, and (b) evaluation metric, i.e., what are the evaluation metrics that should be adopted, e.g. precision, recall, or similarity measures.

Task 4. Apply the selected evaluation approaches in an automatic way or in a semi-automatic way with the help of domain experts.

These four tasks are repeated until all the ontology components are evaluated. Once all the components are evaluated, the evaluation results from the different tasks are combined to have an evaluation report that contains errors, potential corrections, and improvements.

Poveda et al. [Poveda-Villalón et al., 2014] present a catalogue of stand-alone ontology pitfalls. They gathered the different pitfalls from different resources and they

TABLE 1.2: The set of critical pitfalls presented by [Poveda-Villalón et al., 2014]

Code	Pitfall	Affects
P06	Including cycles in the hierarchy	Classes
P19	Swapping intersection and union of classes	Object properties and datatype properties
P01	Creating polysemous elements	Classes, object properties, and datatype properties
P03	Creating the relationship "is" instead of using `rdfs:subClassOf`, `rdf:type` or `owl:sameAs`	objects properties, and datatype properties
P29	Defining wrong transitive relationships	Object properties
P28	Defining wrong symmetric relationships	Object properties
P31	Defining wrong equivalent classes	Classes
P05	Defining wrong inverse relationships	Object properties
P14	Misusing `owl:allValuesFrom`	Classes
P27	Defining wrong equivalent relationships	Object properties and datatype properties
P15	Misusing "not some" and "some not"	Classes
P16	Misusing primitive and defined classes	Classes

categorize them based on three dimensions, namely: 1. structural (i.e., syntax and formal semantics), 2. functional (i.e., the usage of a given ontology) and 3. usability (i.e., the communication context of an ontology). In addition, they tag each pitfall with its importance level (i.e., critical, important, or minor). Moreover, they developed OOPS,[8] which is a pitfall scanner tool. They introduced 41 pitfalls[9] that might occur in stand-alone ontologies. Table 1.2 summarizes the set of pitfalls that are considered critical by [Poveda-Villalón et al., 2014] (sorted from high to low).

Gaudt and Dessimoz [Gaudet and Dessimoz, 2017] introduced an analysis study for the annotations pitfalls that exist in the GO-basic ontology. [10] The author summarized the set of pitfalls (e.g. Annotator Bias and Authorship Bias) and provide good practices to help solving them. They showed how these pitfalls might introduce problems when the data is used in other tasks.

As a conclusion, we show that there is a lack of research papers that observe and list the set of pitfalls that might affect versioned ontologies and ontology networks. In this thesis, we will extend the current state of the art of pitfalls detection by listing a new set of pitfalls that are related to versioned ontologies and ontology networks.

[8] http://oops.linkeddata.es/

[9] Last check April 2020, can be found here: http://oops.linkeddata.es/catalogue.jsp

[10] http://geneontology.org/docs/download-ontology/

Chapter 4 investigates the fourth hypothesis (RH 4 Identifying pitfalls that affect ontology networks and versioned ontologies may help to design better ontologies) where we will answer the fourth and fifth research questions, i.e., RQ 4 What pitfalls affect ontology networks?, and RQ 5 What pitfalls affect versioned ontologies?

1.3 Methodologies for designing ontologies

Several methodologies have been proposed in order to facilitate the development or the evolution of ontologies, such as: [Fernández-López et al., 1997, Noy et al., 2001, Sure et al., 2004]. For interested readers, the following survey papers discuss the different methodologies of ontology developments: [Jones et al., 1998, Cristani and Cuel, 2005, Iqbal et al., 2013, Simperl and Luczak-Rösch, 2014]. In this section we present the most relevant methodologies to our scope of study:

The DILIGENT methodology [Pinto et al., 2004] supports domain experts[11] in distributed setting to develop and evolve ontologies. The methodology consists of:

- Preparing an initial ontology collaboratively between the different parties that are involved in the ontology, mainly: knowledge engineers, domain experts, and end-users. The team size should be relatively small in order to have the initial ontology in easier way.
- Local adaptation to the initial ontology. Other participants are free to create a local ontology from the initial one that is shared by all users.
- Apply tests to define the similarities between the different local copies, so that each participant updates his/her local version in order to have a global coherent version.

The NeOn methodology [Suárez-Figueroa et al., 2012a] supports both the collaborative aspects of ontology development/reuse, along with the evolution of ontology networks in distributed environments. The methodology defines nine scenarios for building ontologies that can be combined in different ways. The scenarios are summarized as follows:

Scenario 1. From specification to implementation: this scenario concerns the development of an ontology (or ontology network) from scratch without reusing any other available resources.

Scenario 2. Reusing and re-engineering non-ontological resources that have not been yet formalized by means of ontologies, such as classification schemes, thesauri [Villazón-Terrazas, 2012]) based on the requirements that the ontology should answer. Ontology engineers should follow three steps to decide whether to include a non-ontological resource or not:

[11] A domain expert is a person with special knowledge or skills in a particular area of endeavor (https://wiki.c2.com/?DomainExpert)

(a) Search existing non-ontological resources. The aim of this step is to search the resources in highly reliable websites, and different resources within the involved organizations. These resources should be decided based on the ontology requirements list.

(b) Evaluate the set of the candidate resources. The aim of this step is to check whether the different resources that are gathered from the first step are relevant or not.

(c) Choose the most suitable resources from the set of candidate resources that are generated from the second step.

Finally, the chosen resources should be transformed into an ontology. This transformation is done either by: (a) reverse engineering for the non-ontological resources, (b) generate a conceptual model from the non-ontological resources, and (c) forward engineering of the ontology on the basis of the conceptual model that is generated in the previous phase.

Scenario 3. Reusing ontological resources. Resources could be other ontologies, ontology modules, and/or ontology terms. This scenario is composed of the five activities:

Activity 1. Search for candidate ontological resources that satisfy the requirements list.

Activity 2. Examine the candidate ontology resources to check whether they satisfy the specific needs inside the ontology requirements specification document (ORCD).

Activity 3. Compare the set of ontologies (gathered in Activity 2) with the set of criteria proposed by the ontology developers, such as: quality, and clarity of the ontologies.

Activity 4. Select the set of ontologies that satisfy the most requirements based on the comparison in Activity 4.

Activity 5. Integrate the ontological resources (chosen from Activity 4) into the ontology network.

Scenario 4. Reusing and re-engineering ontological resources. As some of the ontological resources can be not useful in their current state, ontology engineers rather to re-engineer them in order to re-use them in their ontology networks. Changes can be (a) re-specification in the requirements list (b) re-conceptualization of the ontology structure, (c) re-formalization, such as changing the ontology paradigm from description logic to frames, and (d) re-implementation, such as changing the texual syntax from RDFS to OWL.

Scenario 5. Reusing and merging ontological resources. This scenario happens when a knowledge engineer merges a set of different resources to create a new

ontology. Ontology aligning and ontology merging can performed to satisfy this scenario.

Scenario 6. Reusing, merging, and re-engineering ontological resources, this scenario happens when a knowledge engineer merges and re-engineers a set of different resources to create a new ontology.

Scenario 7. Reusing ontology design patterns (ODPs) for building the ontology network.

Scenario 8. Restructuring ontological resources. This scenario happens when a knowledge engineer restructures the ontological resources in order to be integrated into the developed ontology network. This scenario is composed of the five activities that are related to ontology engineering:

- Activity 1. Modularization: to create different modules inside the ontology network, which helps to reuse these modules later.
- Activity 2. Pruning: to delete the non-necessary parts of the ontology in order to satisfy the ontology requirements.
- Activity 3. Enrichment: to enrich the ontology by adding new concepts and relations, etc.

Scenario 9. Localizing ontological resources. This scenario happens when a knowledge engineer alters all the ontology terms to a different language and to a different community, which will produce a multi-lingual ontology.

The UPON Lite methodology [Nicola and Missikoff, 2016] provides a lightweight rapid ontology engineering method. A domain glossary is prepared by listing the set of terms that are related to a specific domain. The terms are used then to prepare a taxonomy that includes the different terms represented with a hierarchy. Finally, a textual syntax is used (e.g. Turtle) to sequentially produce the ontology. Formally, the UPON Lite methodology consists of the following steps:

Step 1. Prepare a domain terminology, or list the set of terms that characterize a targeted domain.

Step 2. Prepare a domain glossary that is composed of the set of terms from the previous step along with a textual description.

Step 3. Prepare a taxonomy that includes the different terms represented with a hierarchy.

Step 4. Connect the different terms from the previous steps with properties.

Step 5. Use one textual syntax (e.g. Turtle) to sequentially produce the ontology from the previous steps.

SAMOD [Peroni, 2016] is a simplified agile methodology that targets both domain experts and knowledge engineers. It consists of three iterative phases:

Phase 1. The knowledge engineer gathers the information about a targeted domain with the help of domain experts, in order to build a modelet (i.e., a standalone model that describes a specific aspect of the targeted domain). Then a bag of test cases (BoT) (e.g. queries) is examined in order to release a milestone (i.e., a snapshot of the current state of each process).

Phase 2. The knowledge engineer combines the modelet of a new test case with the modelet from the first phase and consequently updates the BoT preparing to release a new milestone.

Phase 3. The knowledge engineer refactors the milestone from the previous phase taking into consideration the good practices for ontology development. If all the test queries inside the BoT are working fine, a new milestone is released. In case of having other requirements from the domain experts side, the knowledge engineer redoes the second phase, otherwise the phases end and a final ontology is released.

Authors in [Zablith et al., 2015] studied different methodologies and approaches to design and evolve ontologies, and they derive an overarching ontology evolution life-cycle (discussed earlier in Section 1.1.2). In this thesis, we adopt this life-cycle, hence in Section 1.4 we categorize the literature review study based on three phases of this life-cycle (i.e., Phase 1. Detect the need for evolution, Phase 2. Suggest changes to evolve the ontology, and Phase 4. Assess and study the impact of the evolution).

Table 1.3 compares the previous methodologies based on the used mechanism, the support of the collaborative development, and the support of ontology evolution.

1.4 A literature review study over the lifecycle of ontology evolution

This section presents the existing research with respect to three phases of the life-cycle of ontology evolution proposed by [Zablith et al., 2015], mainly, Section 1.4.1 presents the approaches that work on detecting the need of the evolution. Section 1.4.2 present existing research on suggesting the changes to bootstrap or enrich ontologies, and Section 1.4.3 presents the existing on studying and observing the impact of the ontology evolution.

1.4.1 Detecting the need for the evolution

Previous work studied the problem of detecting the need of evolution. These approaches are categorized into two categories based on the way of the detection:

1. Detect the need of evolution from data, either by observing the external data (i.e., external knowledge bases, or external raw documents describing the targeted domain) or by observing the internal data (i.e., within the ontology itself).

Table 1.3: A comparison between the different methodologies based on their detection method, whether they support ontology validation and whether they support ontology evolution

	Design mechanism	**Support ontology validation**	**Support ontology evolution**
Diligent [Pinto et al., 2004]	Collaborative development between the different parties (i.e., knowledge engineers, domain experts, and end-users to reach the final ontology	Ontology validation is not supported	Supports ontology evolution, where participants can evolve their local copies in order to evolve and adapt the shared version
NEON [Suárez-Figueroa et al., 2012a]	Collaborative development, along with supporting ontology reuse	Ontology validation is not supported	Support ontology evolution especially in networked settings (i.e., Scenario 8 and Scenario 9)
UPON Lite [Nicola and Missikoff, 2016]	Collaborative development with introduction of the end-users into the process	The different glossaries and terminologies are validated	Terms are updated by evolving the different glossaries and terminologies
SAMOD [Peroni, 2016]	Collaborative development between domain experts and knowledge engineers	Best practices of ontology development are taking into account when refactoring the ontology	Ontology evolution is not supported
Ontology evolution life-cycle [Zablith et al., 2015]	Automatic support for the different phases of the life-cycle	Support ontology validation	Support ontology evolution

TABLE 1.4: A comparison between the state of the art approaches to detect the need of the evolution

	Detection method	
Approach	From data	From usage
[Stojanovic, 2004]	✓	
[Zablith, 2009]	✓	
[Castano et al., 2006]		✓
[Tartir et al., 2010]	✓	
[Noy and Musen, 2002]	✓	
[Papavassiliou et al., 2009]	✓	

2. Detect the need of evolution from usage, by detecting the users behaviors in using the ontologies, which is formally called usage driven ontology evolution.

In the work [Stojanovic, 2004], the author proposed two techniques to detect the need for the evolution: 1. Detect the need of the evolution by studying the ontology instances using data mining techniques. 2. Detect the need of the evolution by observing the structural changes inside an ontology.

In [Zablith, 2009], the author proposed a comparison technique to detect the need for evolution, by comparing the concepts of the targeted ontology with external data sources (e.g. text documents, databases), and suggest new concepts based on the external data sources.

Castano et al. [Castano et al., 2006] rely on the external data sources to detect the need for ontology evolution. Their approach detects whether the ontology needs to be enriched if it does not have concepts that are able to describe a new resource.

Tartir et al. [Tartir et al., 2010] emphasize the proposal of [Noy and Musen, 2002] and they mention that ontology evolution is caused mainly by three reasons: 1. Changes in the described domain. 2. Changes in the conceptualization (e.g. deletion and addition). 3. Changes in the explicit specification.

In [Papavassiliou et al., 2009] a change detection algorithm is proposed. It relies on a specific language they also proposed. One feature of their algorithm is to detect the need of evolution out of the changes that happen, such as renaming a class (i.e., delete and add).

Table 1.4 presents a comparison between the different approaches we discussed earlier. The comparison is done based on the detection method (i.e., from data or from usage).

In the next section, we present the set of related work that tackle the problem of suggesting changes to develop and evolve ontologies.

1.4.2 Suggesting changes to develop and evolve ontologies

Different approaches and tools have been proposed to support the development and the evolution of ontologies, with the help of different resources. Bedini et al. [Bedini and Nguyen, 2007] classify such approaches into four categories:

1. **Conversion or translation:** approaches that use conversion or translation algorithms to construct ontologies from a well-defined representation such as XML or UML. This approach shows a high automation ratio, however, it does not really address the problem of ontology construction.
2. **Mining based:** approaches that use data mining or natural language processing algorithms to construct ontologies. These approaches process unstructured data or text. The approaches in this category require human assistance to help mine or organize the different concepts extracted from the data sources.
3. **External knowledge based:** approaches that use external knowledge bases to construct or to enrich the ontologies. Examples of such external knowledge bases include WordNet [Miller, 1995], Wikidata [Vrandecic and Krötzsch, 2014], DBpedia [Auer et al., 2007], etc.
4. **Frameworks:** approaches that integrate different modules to achieve the goal of constructing ontologies.

We use this categorization to present the current state of the art research work as follow.

Ontology development by conversion or translation

Zarembo [Zarembo, 2015] proposes a system that is able to translate a relational database schema to an OWL ontology. The translating process is done based on a set of rules. For example, their approach directly translates simple attributes from the relational database to the property datatype, but for the composite attributes, they additionally map the component attributes to the sub-property for the corresponding datatype. For evaluating their system they match the generated ontology with two other baseline ontologies. They achieve a similarity ratio of almost 50%.

Following the same idea, Hazber et al. [Hazber et al., 2016] propose a system that is able to translate relational database schema to RDF-OWL ontology, along with the instances. Each column from the relational database is automatically translated into the corresponding representation of the ontology. In order to express more semantics, their system is able to study the constraints between the elements in the relational database in order to extract additional relations. They compare the generated ontology with a relational database scheme, and their system shows an enhancement of almost 40% on precision, recall, and F-measure.

Mining-based ontology development

Dahab et al. [Dahab et al., 2008] propose a system called TextOntoEx, that is able to construct ontology from a raw text using a semantic pattern-based approach. First of all, the ontological engineer is required to annotate the piece of text that is related to the required ontology domain. The second phase is to assign the natural domain text to a specific domain, and finally to construct the ontology out of the natural text. Moreover, their system is able to construct relations based on a set of predefined semantic patterns. The output of the system is a list of ontology classes and all other semantic elements for patterns matched. They match their output to an annotated corpus that consists of 65 sentences describing the agricultural domain. They achieve a recall ratio of almost 54%.

Balakrishna and Srikanth [Balakrishna and Srikanth, 2008] propose a system to construct an ontology for the National Intelligence Priorities Framework (NIPF) topics.[12] They have collected 500 documents from the web, and manually classified them and verified their relevance for the NIPF's topics. Then, the system uses the Jaguar-Kat tool[Moldovan et al., 2007] to extract the textual content along with the hierarchy and the semantic relations. They match their results with manual annotations for four topics, best results are for the weapons topic with a 61% accuracy.

Balakrishna and Moldovan [Balakrishna and Moldovan, 2013] propose an automatic system to build ontologies from unstructured data (e.g. web articles, blogs, manuals). First, their system extracts relevant concepts with two main relations (i.e., IS-A and SYNONYMY) using a set of natural language processing techniques (word boundary detection, part-of-speech tagging, sentence boundary detection, etc.). Then the system uses a set of classification algorithms to define hierarchy for the concepts. They randomly collect and annotate a set of 1k sentences for the intelligence and financial domain. Their system is able to extract 68.5% of knowledge concepts in the text with an accuracy of 61.5%, and it is able to create 68.75% of the domain hierarchy with an accuracy of 84.25%.

Mukherjee et al. [Mukherjee et al., 2014] propose an unsupervised framework to create shallow domain ontologies from text corpus. The ESG parser [McCord et al., 2012] is used to extract important domain terms from a set of documents that are related to some domain. Moreover, they define four relations (Synonyms, Type-of, Action-on, Features-of). Then using a random indexing reduction technique [Sahlgren, 2005] along with a classifier for each relation, they predict the relations that might exist in the text. They manually collected a set of 5k articles related to the smartphone domain. Their system is able to identify 40.87% of the domain terms, compared to 22% extracted by WordNet [Miller, 1995], 43.77% extracted by Yago [Suchanek et al., 2007] and 53.74% extracted by BabelNet [Navigli and Ponzetto, 2010].

Confort et al. [Confort et al., 2015] propose a system that uses a set of natural language processing, clustering, and machine learning techniques, to learn and evolve ontologies from a storytelling corpus. Their system does the following: 1. Extract

[12] https://www.hsdl.org/?abstract&did=761901 Last visit on June 2020

the concepts, attributes, relationships, and axioms that are necessary for generating the ontology, 2. Matching and analyzing the extracted terms with other ontologies, 3. Merging different ontologies together, and 4. Validating the extracted terms. They compare their results with a baseline ontology based on the concept level. Their system is able to retrieve 16 out of 23 concepts that already existed in the baseline ontology.

Using statistical natural language processing techniques, Kumar et al. [Kumar et al., 2016] propose an automatic system to construct ontologies from raw text. After preprocessing the text to remove unwanted words (e.g. stop words), their system studies the morphological analysis. Then using a predefined dictionary of concepts, their system extracts the different concepts that might occur in the text. To extract the relations and the properties, they follow a set of rules and algorithms. They construct an ontology by using a small set of text documents. Their conclude by showing that the quality of the developed ontology can vary depending on the richness of rules and the size of the concepts' dictionary.

Huang et al. [Huang et al., 2016] use a set of probabilistic and semantic features to extract relations from Wikipedia texts. They compare the extracted relations to a manually annotated Wikipedia documents related to the IT domain. For the "IS-A" relationship their system has an F-measure of 77.71%, for the "used-for" relationship their system has an F-measure of 68.39%, for the "produces" relationship their system has an F-measure of 82.61% and for the "provides" relationship their system has a 62.96%.

Lossio-Ventura et al. [Lossio-Ventura et al., 2016] propose a framework to enrich ontologies out of text corpus for the biomedical domain. Their framework consists of four steps: 1. *Term extraction:* to extract a set of candidate terms that are related to the biomedical domain, 2. *Polysemy detection:* for the candidate terms using a machine learning algorithm trained with 23 features, 3. *Sense induction:* to generate the different senses (if exist) for the candidate keywords, and 4. *Semantic linkage:* to decide which terms can be added to the targeted ontologies. Authors used a text corpus that is related to the biomedical domain to extract the list of candidate term, then they computed cosine similarity between the extracted terms and a bag of terms (used as a baseline).

Ontology development based on external knowledge

Kong et al. [Kong et al., 2006] use WordNet [Miller, 1995] as a general ontology to extract a set of concepts to build a domain specific ontology. Their system queries WordNet based on a set of keywords to extend the ontology by adding the list of new concepts. They compare their results to the wine ontology[13] developed by W3C. Examples of other approaches that use WordNet as an external knowledge base include [Moldovan and Girju, 2000, Agirre et al., 2000].

[13] https://www.w3.org/TR/owl-guide/wine.rdf

Kietz et al. [Kietz et al., 2000] propose an approach that uses three knowledge bases to construct ontologies. Each one of the knowledge bases is used to achieve a specific task. The three knowledge bases are: 1. a generic ontology to generate the main structure, 2. a dictionary containing generic terms close to the required domain, and 3. a textual corpus specific to the required domain to enhance and clean the ontology from unrelated concepts. The result is an ontology composed of 381 terms (200 new terms) and 184 relations (42 new relations). The new terms and relation is added to a baseline ontology.

Cahyani and Wasito [Cahyani and Wasito, 2017] propose an automatic system to build an ontology for the Alzheimer's disease. Their system consists of the following steps: 1. term relation extraction, 2. matching the relations to Alzheimer glossary,[14] 3. matching with ontology design patterns, 4. similarity computation, and 5. ontology building and evaluation. To evaluate their system they use a list of 125 papers on Alzheimer disease. Their system is able to retrieve 1,995 correct terms with 42 relations.

Ontology development using frameworks

Zhang et al. [Zhang et al., 2015] propose a system that processes a set of competency questions to extract a list of concepts from them. Those concepts are shown to the user so they can add or re-arrange them in a hierarchy. Then they can extend the hierarchy with a set of other relevant concepts. Moreover, the system provides a short text from Wikipedia to help enriching the ontology. They evaluate their system based on a user study that focuses on the utility, learnability and users satisfaction. They involved 12 participants in the study, and the results were satisfactory.

In the next section we compare all these approaches.

Comparing approaches for ontology development and evolution

Bedini et al. [Bedini and Nguyen, 2007] define a life-cycle (Figure 1.5) to be followed for the automatic ontology development and evolution process. The steps are summarized as follow:

1. Extraction: defining the type of the input that the ontology construction tool should receive. The input can be structured data (e.g. databases), unstructured data (e.g. articles, raw text) or semi-structured data (e.g. XML, JSON).
2. Analysis: matching and analyzing the previously extracted entities (e.g. classes, relations) based on the alignment between a set of selected baseline ontologies. Some techniques are used to help in this step, such as semantic analysis to specify the different relations (e.g. synonyms, homonyms), an analysis of the structure of concepts to find the hierarchy for different concepts in the generated ontology.

[14] https://www.alz.org/care/alzheimers-dementia-glossary.asp

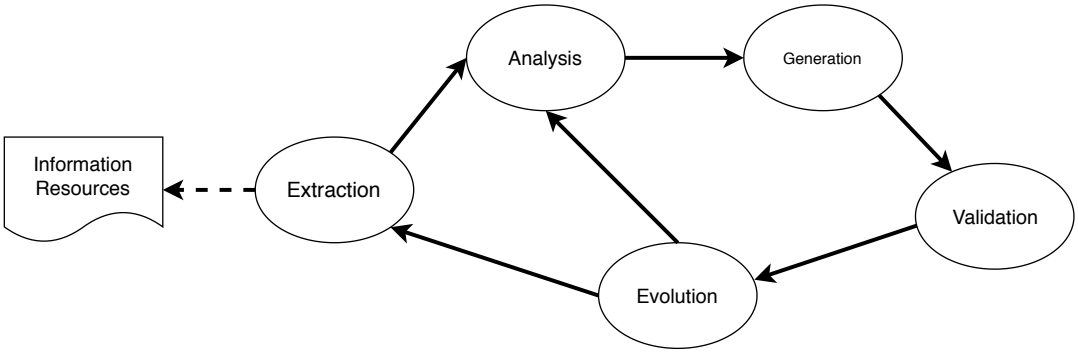


FIGURE 1.5: The life-cycle for automatic ontology construction tools

3. Generation: merging the generated ontology with other ontologies.
4. Validation: the validation of the extracted information (e.g. classes, relations, instances). The validation can be automatic, semi-automatic or manual.
5. Evolution: since the ontology may not be a fixed description of a domain, its evolution in time may be required.

We propose to use this life-cycle to compare the state of art approaches. Moreover, in addition to the five steps of the life-cycle, we introduce three new dimensions that help to better describe the state art approaches. These new dimensions are:

- Type of reusability: the possibility of reusing the proposed system, or the availability of its source code for further extensions.
- Types of extracted data from the system: these types can be classes, properties, instances, relations, and classification of objects.
- Types of bootstrapping capabilities: automatic or semi-automatic (supervised/unsupervised).

Table 1.5 presents synthetically the different approaches based on the life-cycle and on the three newly introduced dimensions. From this table, we draw the following conclusions and comparisons:

1. Regarding the extraction step, we notice that structured data leads to a more advanced output (i.e., ontology generation, instead of a list of terms). Also, we notice that today's NLP techniques show some limitation regarding the accuracy of generated results from crawled documents. Hence, in the extraction step the quality of documents plays an important role.
2. Most of the approaches considering external knowledge bases make use of predefined dictionaries (e.g. list of concepts) or lexicons (e.g. WordNet), or they use specialized glossaries (e.g. Alzheimer glossary[15]). Several limits can be listed regarding these resources: the existence and availability of such dictionary or

[15] https://www.alz.org/care/alzheimers-dementia-glossary.asp

glossary for a given domain, the limited richness of the vocabulary, and the supported languages (generally limited to English).

3. There are two ways to validate the generated output: by humans or using a baseline ontology. This second option offers better scientific comparison method.
4. Most of the validation techniques rely on comparisons with a baseline or with annotated lists of different kinds of output (classes, relations, instances, etc.).
5. Current approaches do not mention any evolution functions, or their solution for evolution relies on a list of learned rules inferring relation proposals. We notice that none of the approaches uses existing ontologies as seeds for the evolutions.
6. Approaches that rely on translating from structured data sources generate a complete ontology (e.g. RDF, OWL). Other approaches (relying mostly on raw text) only generate sets of classes, relations, instances, etc. Structured data sources should then be preferred in automatic ontology construction.

In the next section, we list the research works that tackle the problem of assessing the impact of ontology evolution in different scenarios.

1.4.3 Assess and study the impact of ontology evolution

Several approaches have studied the impact of ontology evolution in different scenarios. We propose to organize these approaches based on the scenarios they use to measure the impact of evolving ontologies: 1. Observing the structural changes such as: addition, deletion, and moving (i.e., deletion then addition) during the life time of an ontology. 2. Measuring the impact of the evolution over the different artifacts (e.g. search systems) that might rely on the evolved ontologies. 3. Listing the changes, the frequency of each change, and the time it took to adopt a specific set of changes. In this subsection we discuss the most relevant research work.

Dragoni and Ghidini [Dragoni and Ghidini, 2012] followed the second scenario, and they investigated how ontology evolution operations affect the effectiveness of search systems. They focused on three operations: 1. rename a concept, 2. delete a concept, and 3. move a concept. They analyzed the impact of the evolution of the ontology over a search system: they performed 75 queries over a search system at every version of the evolved ontology and they calculated the effectiveness of the system by comparing with a baseline.

Abgaz et al. [Abgaz et al., 2012] followed the third scenario, and analyzed structural impact and semantic impact over ontologies. They defined a set of rules to analyze the impact by detecting unsatisfiable statements and wrong instances. They defined 10 change operations that cover the different change scenarios.

Groß et al. [Groß et al., 2012] followed both of the first and second scenarios, and investigated how the changes in the *Gene* ontology[16] might affect the statistical applications for the experimental and simulated data (external artifacts). CODEX tool

[16] http://geneontology.org/

TABLE 1.5: A comparison between the state of the art approaches based on the life-cycle proposed in [Bedini and Nguyen, 2007], enriched by three newly added dimensions: reusability, type of structured data, and bootstrapping capabilities

	Extraction	**Analysis**	**Generation**	**Validation**	**Evolution**	**Reusability**	**Type of Extracted Data**	**Bootstrapping capabilities**
[Zarembo, 2015]	Relational database	Support ontology matching	No information	Matching with a baseline ontology	No information	Reusable in respect of rules	OWL ontology	Automatic bootstrapping
[Hazber et al., 2016]	Relational database	No information	No information	Matching with a baseline ontology	No information	Reusable With modifications to the rules to map the RDB	RDF Ontology	Automatic bootstrapping
[Dahab et al., 2008]	Raw text	Semantic patterns	No information	Matching with annotated list of classes, semantic elements	Enrich ontology with relations	Can be reused with respect to semantic patterns	List of classes and semantic elements	Automatic bootstrapping
[Balakrishna and Srikanth, 2008]	Raw text	No information	Support merging	Matching with annotated concepts	No information	Reusable	List of concepts with their hierarchy & semantic relation	Semi-automatic bootstrapping
[Balakrishna and Moldovan, 2013]	web articles, blogs & manuals	No information	Support merging	Validated by domain experts	No information	Reusable	List of relevant concepts and relations between them	Automatic bootstrapping
[Mukherjee et al., 2014]	Text corpus&set of relations	No information	Do not support merging	Matching with a baseline ontology	No information	Reusable	List of relevant concepts and relations between them	Unsupervised framework
[Confort et al., 2015]	Story tellings text corpus	Matching with existing ontologies	Merging different ontologies together	Matching with a baseline ontology	Ability to add more concepts, relations with time	Reusable with respect to the text source	List of concepts, attributes and relations	Automatic bootstrapping
[Kumar et al., 2016]	Raw text	No information	No information	Matching with annotated list of concepts and relations	No information	Reusable with respects the text source	List of concepts and relations	Automatic bootstrapping
[Huang et al., 2016]	Wikipedia's IT documents	No support for alignment	No support for merging	Comparing to annotated set of relations	Might be used to extend relations	Reusable in case of having annotated features set	List of relations	Semi automatic for bootstrapping
[Lossio-Ventura et al., 2016]	Text corpus	No information	Suggest new terms to be added	Cosine similarity between existing contexts and extracted terms	Might be used to extend terms	Reusable	List of candidate terms	Semi automatic
[Kong et al., 2006]	Set of Keywords	No information	Merging in semi-automatic way	Matching with a baseline ontology	No information	Reusable Approach- No tool	List of classes, relations and instances	Automatic bootstrapping
[Kietz et al., 2000]	Textual corpus	Semi automatic alignment	Merging in semi-automatic way	Matching with a baseline ontology	support evolution	Reusable	List of concepts	Semi-automatic bootstrapping
[Cahyani and Wasito, 2017]	Papers related to Alzheimer's disease	Matching with existing ontology, and lexicon	Automatic way to ontology merging	Matching with a baseline ontology	No information	Method is reusable with other domains	List of concepts and relations between them	Automatic bootstrapping
[Zhang et al., 2015]	Set of competency questions	No information	Merging by a semi automatic way	Validation using users interaction	Support enriching ontologies by providing some related text	Reusable with different domain	An ontology	Semi automatic bootstrapping

1. Extraction: input values. 2. Analysis: matching/alignment of two or more existing ontologies. 3. Generation: ontology merging
4. Validation: automated validation for the extracted entities. 5. Evolution: adding/deleting from the ontology.
6. Reusability: Is the tool reusable, valid for general scenarios. 7. Type of Extracted Data: output values. 8. Bootstrapping capabilities: way of generating

[Hartung et al., 2012] was used to detect the changes (e.g. addition, merging, moving). They introduced their own stability measure by choosing a fixed set of genes to compute the experimental result set at different points of time with freely chosen ontology and annotation versions.

Mihindukulasooriya et al. [Mihindukulasooriya et al., 2016] followed the first scenario, and introduced a study that shows how *DBpedia* [Lehmann et al., 2015], *Schema.org* [Guha et al., 2016], *PROV-O* [Lebo et al., 2013] and *FOAF* [Brickley and Miller, 2010] ontologies evolved through their life time. They counted the changes that occurred between the different versions such as, addition and deletion of classes, properties, sub-classes and sub-properties. They showed that ontology evolution is more challenging when the ontology size is large. Moreover, they showed the need of having tools that can help during the evolution process.

Abdel-Qader et al. [Abdel-Qader et al., 2018] followed the first scenario, and analyzed the impact of the evolution of terms in 18 different ontologies referenced in LOV. Their method consisted of two phases: 1. retrieve all the ontologies that have more than one version, and 2. investigate how terms are changed and adopted in the evolving ontologies. They applied their analysis on three large-scale knowledge graphs: DyLDO,[17] BTC[18] and Wikidata.[19] They found that some of the term changes in the 18 ontologies are not mapped into the three knowledge graphs. Also they concluded that there is a need for a service to keep an eye on the ontology changes. They claim that it would help the knowledge engineers and the data publishers maintaining their artifacts (other ontologies, systems or data sets).

Table 1.6 summarizes the related work that assess and study the impact of ontology evolution. This table shows 1. the different operation types that were observed, 2. followed scenario to study the impact, and 3. the dataset used to observe the impact.

[17] http://km.aifb.kit.edu/projects/dyldo/data
[18] https://km.aifb.kit.edu/projects/btc-2012
[19] https://www.wikidata.org

TABLE 1.6: A comparison between the state of the art approaches for assessing the impact of ontology evolution.

	Ontology operation	Experimental evaluation	Dataset
Dragoni and Ghidini [Dragoni and Ghidini, 2012]	Scenario 2: Rename, Delete, and Move concepts	Detect changes and measure the impact on a search system	Two document collections annotated by MeSH light-Ontology along with 75 queries
Abgaz et al. [Abgaz et al., 2012]	Scenario 3: Structural and Semantic changes	Set of predefined rules to observe the impact, and 4 experts to evaluate it	An ontology that contains 80 classes, 8 data proprieties, 10 object proprieties, and 500 axioms.
Grob et al. [Groß et al., 2012]	Scenarios 1 and 2: Addition (category, relation) Deletion (category, relation), Merge, move and split category	Stability measure to check how the changes might affect the statistical applications for the experimental and simulated data	GENE ontology versions
Mihindukulasooriya et al. [Mihindukulasooriya et al., 2016]	Scenario 1: Addition and Deletion for (Sub-)Classes and (Sub-)Properities	Observe the different changes during the life time of the targetted ontologies	DBpedia, Schema.org, PROV-O, and FOAF
Abdel-Qader et al. [Abdel-Qader et al., 2018]	Scenario 1: Addition and Deletion of terms	How terms are changed and adopted in the evolving ontologies	Selected ontologies from LOV

1.4.4 Discussion over the state of the art

After examining the current research work over the life-cycle of ontology evolution, we stress the following cornerstones of our research: 1. The life-cycle of ontology evolution proposed in [Zablith et al., 2015]. This life-cycle has been designed based on observing and studying a wide range of ontology evolution research. It is comprehensive and corresponds to our needs for describing our proposal. 2. The comprehensive analysis for stand-alone ontology pitfalls proposed in [Poveda-Villalón et al., 2014]. Our pitfalls analysis will focus on versioned and networked ontologies, however we will take full inspiration from there approach and from their catalogue of pitfalls.

The following points summarize the current limitations and drawbacks that we can identify in the current research with respect to the thesis hypothesis and research questions:

- Current pitfalls analysis studies and tools addressed stand-alone ontology pitfalls. However, pitfalls in versioned ontologies are not addressed and pitfalls in ontology networks are formally addressed in [Sabou and Fernández, 2012].
- Two techniques are used in current research work for the detection of the need of evolution: detect from the data; and detect from the usage. However, we notice that current approaches are not considering detecting the need of evolution by observing the evolution of imported ontologies.

- Current approaches rely on the quality of the resources (i.e., whether structured or unstructured) to generate or evolve ontologies. In addition, these approaches considering external knowledge bases make use of predefined dictionaries, lexicons, or specialized glossaries. Several limits can be listed regarding these resources: the existence and availability of such dictionary or glossary for a given domain, the limited richness of the vocabulary, and the supported languages (generally limited to English).
- There is no formal agreement of a definition that targets assessing the impact of ontology evolution.

Conclusion

This chapter presented the following:

Firstly, in Section 1.1 we presented an overview of the concept of ontology, the different standards that are used to describe ontologies (RDF, RDFS, and OWL), the main guidelines and recommendations that are used to publish them online, and an overview of ontology evolution along with the ontology evolution life-cycle proposed by [Zablith et al., 2015].

Secondly, in Section 1.2 we presented the current studies of analyzing pitfalls and we showed that the current solutions focus on the set of pitfalls that are targeted to stand-alone ontology pitfalls. In Chapter 4 we will present our contribution to this topic, where we will list and evaluate a set of pitfalls that are related to versioned ontologies and ontology networks.

Thirdly, in Section 1.3 we presented an overview of different methodologies that are used to develop and evolve ontologies. We introduced a comparison between these methodologies based on their design mechanism, whether they support ontology validation, and whether they support both ontology development and evolution.

Finally, in Section 1.4 we presented a targeted literature review on three phases of the ontology evolution life-cycle that are of importance for our work. Our contributions will be situated in these three phases of ontology evolution:

- Detecting the need of evolution: in Chapter 2, Section 2.1 we will introduce a new definition that is used to detect the need of evolution by observing the evolution of an imported ontology.
- Suggesting new changes to evolve ontologies: in Chapter 2, Section 2.2 we will introduce a new functionality that takes advantage of existing knowledge bases to evolve ontologies by suggesting sets of new classes, relations and instances to be added.
- Assessing the impact of the evolution of an ontology: in Chapter 3 we will introduce a new definition to assess the impact of the ontology evolution in ontology networks.

Part II

Contributions

Chapter 2

On Detecting the Need for Evolution and Enriching Ontologies using External Knowledge Bases

Overview

In this chapter we target the first research goal:
RG.1 *To study the evolution need and evolution implementation of ontologies.*
Where we investigate the following research hypotheses and research questions:
RH.1 An ontology may need to evolve after some changes in some ontologies it uses.
RQ.1 How to detect the need of evolving an ontology through the observation of structural changes in the ontologies it uses?
RH.2 Using existing knowledge sources may help to develop and evolve ontologies.
RQ.2 How to take advantage of external knowledge bases to develop and evolve ontologies?

Introduction

Ontology evolution is a crucial task in ontology engineering. Several methodologies and life-cycles were proposed in order to control and facilitate the process of ontology evolution. In this chapter we investigate both of *Phase 1. Detect the need for evolution* and *Phase 2. Suggest changes to evolve the ontology* from [Zablith et al., 2015] (Figure 1.4).

We propose to illustrate this chapter with the continuation of the motivating scenario described in the introduction of this thesis, two behaviors can be derived: first,

in her *Childcare* $v_{1.1}$ (created in May 2019) ontology, Amal used a specific term `edu:programOfStudy` from the *Education* $v_{1.1}$ (created in January 2019). In September 2017, *Education* $v_{1.2}$ was created and the term `edu:programOfStudy` was deleted. This deletion might have an impact over the *Childcare* ontology, hence Amal should update her ontology. This need of evolution can be derived from the evolution of the imported ontology (i.e., the *Education* ontology). In Section 2.1 we propose a definition that can be used to detect the need of evolving ontologies based on the evolution of the imported ontologies.

Second, after detecting the need of evolving her *Childcare* $v_{1.1}$ ontology, Amal decided to create a second version *Childcare* $v_{1.2}$, what can be interesting for Amal is to use some tools that help her to evolve her ontologies. These tools can help her to enrich her set of classes, relations, etc. In Section 2.2 we introduce a functionality that takes advantage of existing knowledge bases to evolve ontologies. Moreover, this functionality can also help to initiate the process of creating ontologies instead of starting the process from a blank page.

2.1 Detect the need of ontology evolution

A RDF term is defined as: $IRI \cup B \cup L$, where B: blank nodes, and L: literals. In this thesis, we take into consideration only the IRIs (I). Detecting the need for evolving the set of terms can be manifested through two behaviors:

1. There is already a problem: If an ontology O uses a term t that has the namespace of another ontology O', however it is not defined in O'.
2. A problem has occurred because of the evolution process: Let's assume that there is an ontology O that uses a term t that has the namespace of another ontology O'. O' evolved which cause the deletion of t. This evolution might cause problems for O. This raises the need to evolve O in order to reflect the new changes, which leads to solve the different problems.

Hereinafter, we defined an *imported ontology evolution* case as:

Definition 1 ***Imported ontology evolution***

Imported ontology evolution is a situation where: O is an ontology which has at least one version v_1. O' is a different ontology which has at least two versions v'_1 and v'_2. O uses terms that have the namespace of O'. $time(v)$ is the creation time for a version.

A case of imported ontology evolution is noted $\langle v_1, v'_1, v'_2 \rangle$ and holds when the following condition are satisfied:
$time(v'_1) < time(v'_2) \wedge time(v'_1) < time(v_1)$

Figure 2.1 presents an example of one case of imported ontology evolution: the *Neo-Geo Geometry* Ontology has one version (v_1: ngeo_2012-02-05) that uses the *Dublin Core Metadata Element Set* ontology (v'_1: dce_2010-10-11, v'_2: dce_2012-06-14). Table 2.1 reflects the different cases that may occur with respect to Definition 1. t is a term that has the namespace of O'. The circles at every line represent the set of

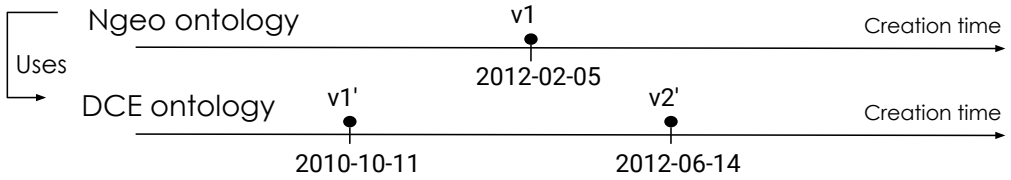


FIGURE 2.1: A time line showing the creation times of the music ontology (mo) and the bio ontology (bio), where mo uses terms that are defined by bio

TABLE 2.1: The set of cases that might happen during the evolution of O' considering a term t that has the namespace of O'

	uses O / O′	a: v_1	b: v_1 t
1	v_1' v_2'	Case 1.a: No changes occurred	Case 1.b: Term is used in v_1 without being defined in O′
2	v_1' t v_2'	Case 2.a: No impact occurred	Case 2.b: There is a need for evolution, because the term is no longer in O′
3	v_1' t v_2'	Case 3.a: No impact occurred. Suggest to add new terms	Case 3.b: No impact occurred
4	v_1' t v_2'	Case 4.a: Suggest to add new terms	Case 4.b: Term used before it is defined

terms (t) that exist in the the two versions of the ontology O' (i.e., v_1' and v_2'). The columns represents the set of terms t that exist in the ontology O (i.e., v_1).

2.1.1 The possible cases

Four possible cases might happen:

1. No changes over t

Case 1.a There is no change of t to detect, therefore there is no interest in studying this case. This case holds for all the terms t with the namespace of O', that are neither defined in O' nor used in O

Case 1.b This case holds when O uses a term t with the namespace of O', but that is not defined in O'. Some terms that have the namespace of O' are being used in v_1 without being defined before. This is a mistake, hence there is a need to evolve v_1 to reflect the latest changes.

2. t is deleted in v_2'
The owners of O' decided to stop using a term (e.g. `edu:programOfStudy`) in v_2':

Case 2.a The term is not used in v_1. No problems to be reported, and v_1 was not affected by the evolution of O'.

Case 2.b During the evolution, the term t was deleted. However, it is still being used in v_1. This might introduce a problem of using terms that does not exist any more. So v_1 should evolve to better reflect the changes of O'.

3. t exist in both v_1' and v_2'
There is no changes on t:

Case 3.a The term is not used in v_1. However, it can be recommended to use in the upcoming versions of v_1.

Case 3.b No changes over the terms during the evolution.This case is not problematic.

4. t is added to v_2'
The owners of O' introduced a new term (e.g. `edu:boardingSchool`) in v_2':

Case 4.a The term t is not used in v_1. It can be interesting to use, thus this addition can be notified.

Case 4.b The term t is used in v_1, however it was defined later in v_2'.

Hitherto, in Definition 1 we present how the evolution of an imported ontology O' can lead to detect the need of the evolution of the ontology O. Table 2.2 extends Table 1.4 and includes our proposed approach to compare it with the existing state of the art approaches.

2.2 A semi-automatic approach for ontology enrichment using external knowledge bases

"A knowledge base (KB) is a technology used to store complex structured and unstructured information used by computer systems.[1]" In this thesis, we investigate how existing knowledge bases can be used to help evolving ontologies. We aim at targeting three public knowledge bases:

1. DBpedia [Auer et al., 2007]: a knowledge base used to store extracted structured content from Wikipedia pages.

[1] https://en.wikipedia.org/wiki/Knowledge_base

TABLE 2.2: A comparison between the state of the art approaches described in Table 1.4 and our proposed approach for detecting the need of the evolution

	Detection method	
Approach	From data	From usage
[Stojanovic, 2004]	✓	
[Zablith, 2009]	✓	
[Castano et al., 2006]		✓
[Tartir et al., 2010]	✓	
[Noy and Musen, 2002]	✓	
[Papavassiliou et al., 2009]	✓	
Our proposed approach	✓	✓

2. Wikidata [Vrandecic and Krötzsch, 2014]: a collaboratively edited knowledge base.
3. NELL: a Never Ending Language Learner [Carlson et al., 2010]: a machine learning computer system that is used to extract facts over the World Wide Web.

We propose to use these knowledge bases to help enriching ontologies by suggesting a set of classes, relations and instances to be added to the targeted ontology (i.e., the ontology that needs to evolve). The pros of using these knowledge bases are that they are structured (RDF for DBpedia and Wikidata; specific data format for NELL), very large, include rich relations, are dynamic (i.e., evolving in time), machine understandable and multilingual.

In Section 2.2.1 we will present our research methodology for this chapter. Then in Section 2.2.2 we will present our extraction algorithm for the set of keywords using Apache Lucene [Białecki et al., 2012]. Section 2.2.3 will present our extraction algorithm for the general information using DBpedia knowledge base. Section 2.2.4 will present our extraction algorithm for the set of classes and relations using Wikidata knowledge base. Section 2.2.5 will present our extraction algorithm for the instances using NELL knowledge base.

2.2.1 Research methodology

In Section 1.4.2 we showed that the process of ontology development is facing two main problems: the initiation of the extraction phase (cold start, blank page problem) [Zhang et al., 2015], and the large number of micro-contributions that the domain experts must do, which requires availability and strong involvement. In [Qawasmeh et al., 2018] we presented a functionality that takes advantage of publicly available knowledge bases: DBpedia, Wikidata and NELL to initiate the process of ontology engineering instead of starting from scratch (i.e., cold start, blank page problem). Algorithm 1 presents our proposed functionality, it can be illustrated in Part 1 from

Figure 2.2. We follow a semi-automatic bootstrapping technique, where users are asked to enter a set of keywords related to a specific domain (e.g. wine, grapes, wine color, wine region, for the wine domain). Then a series of tasks are performed as follow:

1. We query DBpedia knowledge base to extract the related abstract for the entered keyword(s), this helps to solve any ambiguity issues for the users [step 1].
2. We query Wikidata knowledge base to extract several classes and relations that are related to the entered keyword [step 2]
3. The generated list is shown to the user for selection [step 3].
4. After the user's validation, the set of classes is used to extract the instances from the NELL knowledge base [step 4]

This process can be repeated and provides thus interaction between the knowledge engineers and the different knowledge bases. Table 2.3 extends Table 1.5 and includes our proposed approach to compare it with the existing state of the art approaches that have similar techniques to our proposed method.

Algorithm 1: The used algorithm implemented by our system for Part 1

ConstructInitialOntology($keywords$);
Input : $keywords$, a list of keywords given by the domain expert
Output : $\langle classes, relations, instances \rangle$ lists of terms to bootstrap the ontology.
$\langle classes, relations, instances \rangle \leftarrow \langle \varnothing, \varnothing, \varnothing \rangle$
foreach *keyword in keywords* **do**
    $\langle abstract, labels, uri \rangle \leftarrow$ queryDBPedia($keyword$)
    $\langle classes, relations \rangle \leftarrow$ queryWikiData($keyword$)
    $instances \leftarrow$ queryNELL($keyword$) $\langle classes', relations', instances' \rangle \leftarrow$ pick($abstract, labels, uri, classes, relations, instances$); `// let the users pick the terms they want`
    $classes \leftarrow classes \cup classes'$;
    $relations \leftarrow relations \cup relations'$;
    $instances \leftarrow instances \cup instances'$;
return $\langle classes, relations, instances \rangle$;

The selection process of the keywords is essential, as choosing a good set of related keywords can help effectively during the process of querying the different knowledge bases. Hence, we propose to add a new sub-functionality to help extracting a candidate set of keywords from the targeted ontology itself. Then, these keywords are proposed to the users to choose from in order to start the functionality. This can be illustrated in Part 2 from Figure 2.2.[2]

[2]Source code is available at: https://github.com/OmarAlqawasmeh/OntologyEnrichment

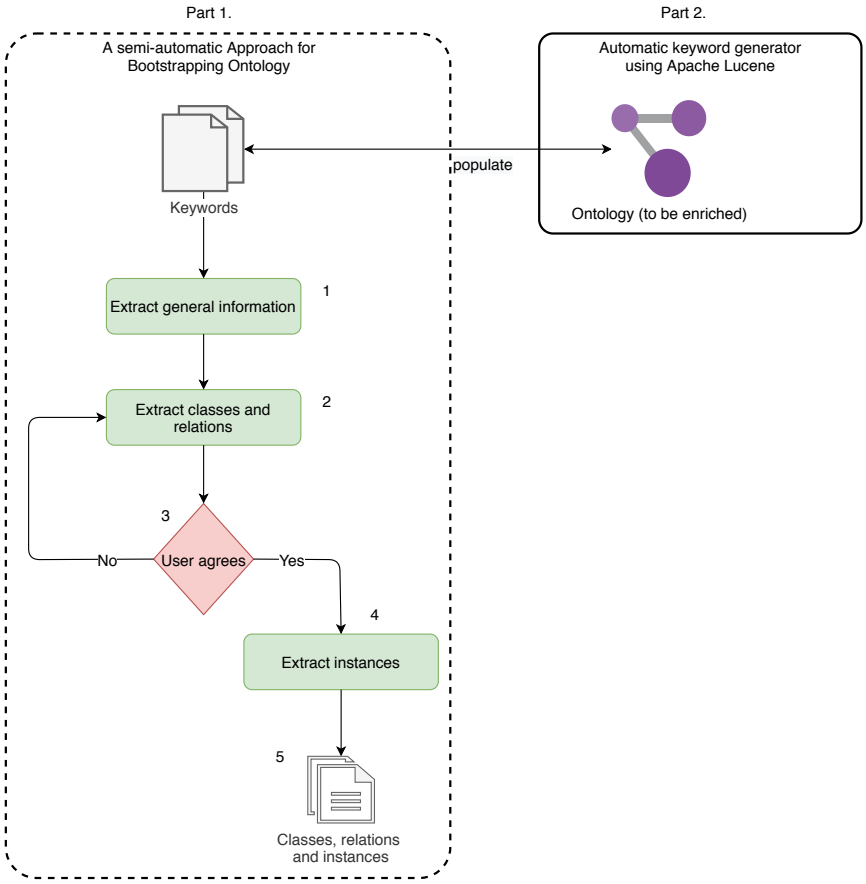


FIGURE 2.2: An overview of the proposed methodology to enrich ontologies

2.2.2 Extract the set of keywords using Apache Lucene

Choosing the set of keywords is essential in our functionality, it helps to better perform the search queries over the knowledge bases. Hence, we propose to extract the set of candidate keywords from the targeted ontology itself (i.e., the ontology to evolve).

In order to do that, we propose to add a new sub-functionality to our workflow. As shown in Figure 2.2, Part 2, the users can specify an ontology as an input along with a first hit-keyword. Then the ontology is queried using a text query search that is performed with the help of Apache Lucene. Apache Lucene [Białecki et al., 2012] is a high-performance, full-featured text search engine written in Java that provides indexing and search features, as well as spellchecking. Apache Lucene can be used to add search capacities to the extraction process of the keywords.

As shown in Algorithm 2, the algorithm takes an ontology (i.e., targeted to be enriched) and a hit-keyword as an input. Then a text search query is applied to the

TABLE 2.3: A comparison between the state of the art approaches and our proposed approach based on the life-cycle proposed in [Bedini and Nguyen, 2007], enriched by three newly added dimensions: reusability, type of structured data, and bootstrapping capabilities

	Extraction	**Analysis**	**Generation**	**Validation**	**Evolution**	**Reusability**	**Type of Extracted Data**	**Bootstrapping capabilities**
[Zarembo, 2015]	Relational database	Support ontology matching	No Information	Matching with a baseline ontology	No Information	Reusable in respect of rules	OWL ontology	Automatic bootstrapping
[Hazber et al., 2016]	Relational database	No Information	No Information	Matching with a baseline ontology	No Information	Reusable With modifications to the rules to map the RDB	RDF Ontology	Automatic bootstrapping
[Dahab et al., 2008]	Raw text	Semantic patterns	No Information	Matching with annotated list of classes, semantic elements	Enrich ontology with relations	Can be reused with respect to semantic patterns	List of classes and semantic elements	Automatic bootstrapping
[Balakrishna and Srikanth, 2008]	Raw text	No Information	Support merging	Matching with annotated concepts	No Information	Reusable	List of concepts with their hierarchy & semantic relation	Semi-automatic bootstrapping
[Balakrishna and Moldovan, 2013]	web articles, blogs & manuals	No Information	Support merging	Validated by domain experts	No Information	Reusable	List of relevant concepts and relations between them	Automatic bootstrapping
[Mukherjee et al., 2014]	Text corpus&set of relations	No Information	Do not support merging	Matching with a baseline ontology	No Information	Reusable	List of relevant concepts and relations between them	Unsupervised framework
[Confort et al., 2015]	Story tellings text corpus	Matching with existing ontologies	Merging different ontologies together	Matching with a baseline ontology	Ability to add more concepts, relations with time	Reusable with respect to the text source	List of concepts, attributes and relations	Automatic bootstrapping
[Kumar et al., 2016]	Raw text	No Information	No Information	Matching with annotated list of concepts and relations	No Information	Reusable with respects the text source	List of concepts and relations	Automatic bootstrapping
[Huang et al., 2016]	Wikipedia's IT documents	No support for alignment	No support for merging	Comparing to annotated set of relations	Might be used to extend relations	Reusable in case of having annotated features set	List of relations	Semi automatic for bootstrapping
[Lossio-Ventura et al., 2016]	Text corpus	No information	Suggest new terms to be added	Cosine similarity between existing contexts and extracted terms	Might be used to extend terms	Reusable	List of candidate terms	Semi automatic
[Kong et al., 2006]	Set of Keywords	No Information	Merging in semi-automatic way	Matching with a baseline ontology	No Information	Reusable Approach- No tool	List of classes, relations and instances	Automatic bootstrapping
[Kietz et al., 2000]	Textual corpus	Semi automatic alignment	Merging in semi-automatic way	Matching with a baseline ontology	support evolution	Reusable	List of concepts	Semi-automatic bootstrapping
[Cahyani and Wasito, 2017]	Papers related to Alzheimer's disease	Matching with existing ontology, and lexicon	Automatic way to ontology merging	Matching with a baseline ontology	No Information	Method is reusable with other domains	List of concepts and relations between them	Automatic bootstrapping
[Zhang et al., 2015]	Set of competency questions	No Information	Merging by a semi automatic way	Validation using users interaction	Support enriching ontologies by providing some related text	Reusable with different domain	An ontology	Semi automatic bootstrapping
Our proposed approach	Keywords: user inputted or aromatically extracted	With the help of domain experts	NA	With the help of domain experts	Evolving ontologies by suggesting new classes, relations and instances	Reusable with different domain	List of classes, relations, and instances	Semi-automatic evolution

1. Extraction: input values. 2. Analysis: matching/alignment of two or more existing ontologies. 3. Generation: ontology merging
4. Validation: automated validation for the extracted entities. 5. Evolution: adding/deleting from the ontology.
6. Reusability: Is the tool reusable, valid for general scenarios. 7. Type of Extracted Data: output values. 8. Bootstrapping capabilities: way of generating

Algorithm 2: Extract a set of candidate keywords that are used to enrich ontologies

Input : *ontology*, An ontology to be enriched
Input : *initKeyword*, A hit-keyword used to start the process
Output : $\langle candidateKeywords \rangle$ lists of candidate keywords to be used later to enrich the ontology.

1 $\langle candidateKeywords \rangle \leftarrow \langle \varnothing \rangle$
2 getCandidateKeywords(*ontology*, *initKeyword*)
read *ontology*;
execute text query search presented in Listing 2.1;
rank the extracted entities that are relevant to *initkeyword*;
candidateKeywords ← *bestrankedretrievedlabels*;
3 **return** $\langle candidateKeywords \rangle$;

input ontology using the Jena Full Text Search module of Apache Jena[3] that defines so-called magic properties[4] for full text search using Apache Lucene. This text search aims to extract the labels of the terms that are associated to the hit-keyword ranked in descending order (most relevant to less relevant). Finally, we retrieve the top 3 keywords with the highest rank (if found). Then the user can choose from these keywords to start the process of enriching the targeted ontology. If there are no retrieved results, the user is asked to enter an initial keyword to start the process.

Listing 2.1: The performed query to retrieve the set of candidate keywords

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX text: <http://jena.apache.org/text#>
SELECT DISTINCT ?s ?label ?comment ?score
WHERE {
       ( ?s ?score ?literal ) text:query $keyword$ .
         ?s rdfs:label ?label
       OPTIONAL { ?s rdfs:comment ?comment }
   }
       ORDER BY DESC ( ?score )
       LIMIT 3;
```

Listing 2.1 shows the performed query to apply the text search. Three arguments are used: 1. *?s* (subject URI): the subject of the indexed RDF triple. 2. *?score*: the score for the retrieved match. 3. *?literal*: the matched object literal. For example, Table 2.4 shows the output of applying this query to the *saref4watr* ontology,[5] with an initial keyword "Tariff".

[3] https://jena.apache.org/documentation/query/text-query.html
[4] https://jena.apache.org/documentation/query/extension.html#property-functions
[5] Extends the SAREF ontology for the water domain, available at: https://forge.etsi.org/rep/SAREF/saref4watr

TABLE 2.4: The output from performing Listing 2.1 over the *saref4watr* ontology

s	label	score
saref4watr:Tariff	"Tariff"@en	3.40
saref4watr:hasDuration	"has duration"@en	2.72
saref4watr:hasPeriod	"has period"@en	2.72

2.2.3 Extract general information (DBpedia)

DBpedia knowledge base [Auer et al., 2007] contains structured information from Wikipedia that is accessible via a SPARQL endpoint [Harris et al., 2013]. The English DBpedia describes 4.22M resources in a consistent ontology. Full DBpedia data set contains 38M labels and abstracts available in 125 different languages.[6]

LISTING 2.2: The DBpedia query to extract general information for a certain keyword

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>

SELECT ?uri ?label ?abstract ?type
WHERE {
?uri rdfs:label ?label .
?uri dbpedia-owl:abstract ?abstract .
?uri rdf:type ?type .
filter(?label="keyword"@en) .
FILTER langMatches( lang(?abstract), 'en')
}
```

In this phase, the set of keywords are used to perform a SPARQL query (see Listing 2.2) over the DBpedia knowledge base to retrieve some information that will help the user to choose clearly among the related terms that can be retrieved. This will help to:

1. Resolve the ambiguity that might occur during the extraction phase of the classes, relations and instances. This is achieved by having the abstract, label, the URI to the retrieved page on DBpedia and the type of the targeted keyword (e.g. beverage, food for the wine domain).
2. As DBpedia is language independent, the different keywords can be searched in different languages, such as English, Arabic, French, etc. This will support the possibility of multilingualism usage for the proposed functionality.

[6] All numbers are based on: http://wiki.dbpedia.org Last visit March-2020

Listing 2.3 shows the output for the keyword "wine", which is the abstract from wine's Wikipedia page,[7] the label in DBpedia (retrieved in English language), and the keyword type as defined in DBpedia (i.e., http://dbpedia.org/ontology/Food).

Listing 2.3: The results of the query written in Listing 2.2

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<entity name="Wine">
<label>Wine@en</label>
<abstract> Wine (from Latin vinum) is an alcoholic beverage made from fermented
    grapes, generally Vitis vinifera or its hybrids with Vitis labrusca or Vitis
    rupestris. Grapes ferment without the addition of sugars, acids, enzymes, water,
    or other nutrients, as yeast consumes the sugar in the grapes and converts it to
    ethanol and carbon dioxide. Different varieties of grapes and strains of yeasts
    produce different styles of wine. These variations result from the complex
    interactions between the biochemical development of the grape, the reactions
    involved in fermentation, the terroir (the special characteristics imparted by
    geography, geology, climate, viticultural methods and plant genetics), and the
    production process. Many countries define legal appellations intended to define
    styles and qualities of wine these typically restrict the geographical origin and
     permitted varieties of grapes, as well as other aspects of wine production.
    There are also wines made from fermenting other fruits or cereals, whose names
    often specify their base, with some having specific names. Wines made from plants
     other than grapes include rice wine and various fruit wines such as those made
    from plums or cherries. Some well known example are hard cider from apples, perry
     from pears, pomegranate wine, and elderberry wine. Wine has been produced for
    thousands of years. The earliest known evidence of wine comes from Georgia (
    Caucasus), where 8000-year-old wine jars were found. Traces of wine have also
    been found in Iran with 7000-year-old wine jars and in Armenia, in the 6100-year
    old Areni-1 winery, the earliest known winery. Wine had reached the Balkans by c.
     4500 BC and was consumed and celebrated in ancient Greece, Thrace and Rome.
    Throughout history, wine has been consumed for its intoxicating effects, which
    are evident at normal serving sizes. Wine has long played an important role in
    religion. Red wine was associated with blood by the ancient Egyptians and was
    used by both the Greek cult of Dionysus and the Romans in their Bacchanalia;
    Judaism also incorporates it in the Kiddush and Christianity in the Eucharist.@en
     </abstract>
<type>http://dbpedia.org/ontology/Food</type>
</entity>
```

2.2.4 Extract classes and relations (Wikidata)

Wikidata [Vrandecic and Krötzsch, 2014] is a collaborative, multilingual, structured knowledge base that can be read and modified by both humans and machines. The information on Wikidata is accessible by querying services. Wikidata has 80.5M of data items.[8] Each entity in Wikidata has a unique ID. Using the query listed in Listing 2.4 we retrieve the Wikidata IDs for the inputted keywords.

[7] https://en.wikipedia.org/wiki/Wine Last visit Jan-2018

[8] Based on: https://www.wikidata.org Last visit: March-2020

TABLE 2.5: Set of RDF-Relations Extracted for the keyword wine

Relation	Count	URI of the relation
instance of	2254	http://www.wikidata.org/entity/P31
subclass of	96	http://www.wikidata.org/entity/P279
depicts	35	http://www.wikidata.org/entity/P180
main subject	8	http://www.wikidata.org/entity/P921
has part	6	http://www.wikidata.org/entity/P527
material used	6	http://www.wikidata.org/entity/P186
product or material produced	5	http://www.wikidata.org/entity/P1056

LISTING 2.4: The performed query to retrieve Wikidata ID for the candidate keyword(s)

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?id
WHERE
{
    ?id rdfs:label $Keyword$ @en
}
```

Then, using these retrieved IDs, we perform different queries over the Wikidata knowledge base to retrieve a set of classes and the relations that will be shown to users to choose from. We use different queries to have the following output:

Firstly, the most connected relations for each class: In this query (Listing 2.5), a list of relations that are connected to a specific class is retrieved along with the number of instances that are using this relation. For instance, the query with "wine" retrieves 6 different relations and their number of use (for this class). The set of relations along with the number of use is shown in Table 2.5.

LISTING 2.5: The performed query to retrieve relationships from Wikidata

```
PREFIX wd: <http://www.wikidata.org/entity/>
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>
PREFIX wikibase: <http://wikiba.se/ontology#>

SELECT ?property (COUNT(?item) AS ?count) WHERE {

 ?item ?statement wd:$retrievedWikidataID$ .
 ?property wikibase:statementProperty ?statement .
 } GROUP BY ?property

 ORDER BY DESC(?count)
```

TABLE 2.6: Set of top classes between wine class and alcoholic class

Class	Count	URI of the Classes
red wine	14	https://www.wikidata.org/wiki/Q1827
white wine	5	https://www.wikidata.org/wiki/Q10210
Champagne	3	https://www.wikidata.org/wiki/Q134862
sparkling wine	3	https://www.wikidata.org/wiki/Q321263
fortified wine	3	https://www.wikidata.org/wiki/Q722338
rosé	1	https://www.wikidata.org/wiki/Q12979

Secondly, classes, along with their top-level high classes: In this query (Listing 2.6), a list of relations that are connected to two different classes are retrieved along with the number of instances that are using this relation. For example for the class wine and the class alcoholic beverage the query was able to retrieve 7 subclasses (Table 2.6).

LISTING 2.6: The performed query to retrieve classes from Wikidata

```
prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX wd: <http://www.wikidata.org/entity/>
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>

SELECT ?s ?desc WHERE
{
?s wdt:P279 wd:$WikidataID$.
OPTIONAL {
 ?s rdfs:label ?desc
 filter (lang(?desc) = "en") .
}
}
```

2.2.5 Extract instances (NELL)

Since January 2010, a computer system called NELL (Never-Ending Language Learner) [Carlson et al., 2010] has been running continuously, in order to learn over time from the World Wide Web.[9] NELL performs two tasks: 1. Read/Extract the facts from raw text than can be found in hundreds of millions of web pages. (e.g. "Barack Obama" is a person and politician); 2. Improve its reading competence, in order to extract more facts accurately.

NELL currently has more than 50 million beliefs.[10] These beliefs are attached to different levels of confidence. In order to access NELL knowledge base, we use three main files that are provided by NELL:

[9] NELL service has been stopped since 2018. Latest version can be found at http://rtw.ml.cmu.edu/rtw/

[10] Based on: http://rtw.ml.cmu.edu/rtw/ Last visit: October-2017

1. Relations: contains 460 relations that were extracted manually. Each relation is related to a set of features (e.g. domain, range, examples for some instances, a simple description about the relation).
2. Categories: contains 291 categories that were extracted manually. Each category is related to a set of features (e.g. mutex exceptions, generalization of the category, some examples of different instances that might be related to the category, edit date, description).
3. Instances: contains 2,971,069 instances. Each instance is related to a set of features (e.g. connection to a specific relation, a URI to the instance, confidence value for the information related to the instance).

In this phase, we use the NELL knowledge base in order to build a candidate list of instances that are related to the given set of keywords. NELL is queried based on a set of features such as domain, range, and confidence values. The number of instances can vary from one keyword to another. For example there are almost 1400 instances that are related to the "wine" keyword. Note that the confidence values attached to the instances can vary, which affects the retrieval process. These instances provide an additional information to the knowledge engineers (i.e., decision to include or not a specific entity, based on the instances received), they are used as candidates to populate the generated or existing ontologies.

Algorithm 3 is the algorithm we implemented to extract the set of instances. Mainly, we take advantage of the following features that are listed in NELL's knowledge base:

- *Categories for Entity:* presents the set of categories in which NELL believes that an entity is related to a concept (property). For example: the entity `celebrity:marion_cotillard` is of type `concept:actor` and `concept:celebrity`.
- *Probability:* a confidence value associated to every belief (triple) in NELL's knowledge base. We have chosen a threshold value of 94% to retrieve the instance and to promote it to the users. For example, the triple: `celebrity:marion_cotillard concept:actorstarredinmovie movie:la_vie_en_rose` has a confidence value of 96.8%.
- *Entity literalStrings:* presents the set of possible string forms that the entity is associated with. For example, the concept `celebrity:marion_cotillard` appeared in the following forms: "Marion Cotillard", "marion cotillard", "marion-cotillard" or "MARION COTILLARD".

2.2.6 Comparative evaluation

Most approaches cannot be evaluated on an arbitrary domain as it would require numerous specific data sources: a specific database on a domain, a corpus describing the domain, existing ontologies on the domain, etc. So, in order to validate our approach, we compare our results to those published in [Kong et al., 2006]. Recall authors in [Kong et al., 2006] proposed an ontology construction approach based on

Algorithm 3: Extract instances from the NELL knowledge base

```
1 GetInstances(keywords);
  Input  : keywords, a list of keywords extracted from the targeted ontology
  Output: ⟨instances⟩ lists of instances that are related to a keyword.
2 ⟨instances⟩ ← ⟨∅⟩
3 foreach belief in NELL_KnowledgeBase do
4     foreach keyword in keywords do
5         if getCategoriesForEntity(keyword) = haswikipediaurl then
6             if getProbability(keyword) > 0.94
                and getCategoriesForEntity(keyword) = "concept : keyword" then
7                 instance ←getEntityLiteralStrings(keyword);
                  instanceIRI ←getWikipediaIRI(keyword);
                  instances ← instance ∪ instanceIRI;
8 return ⟨instances⟩ ;
```

WordNet, and validated it comparing the numbers of extracted classes, properties and instances with the W3C's *wine* ontology.[11] We therefore lead a similar experiment to evaluate our system, and we compare our results to the baseline ontology (the W3C's *wine* ontology) and to the results in [Kong et al., 2006].

Authors in [Kong et al., 2006] use keyword "wine" to perform a query over WordNet. We used the same keyword "wine" as an input to our system. The raw results of our experiment, i.e., the full lists of classes, relations, and instances, our system suggests to the user, are made available in a Google sheet online.[12] Table 2.7 gives an overview of these results and compares them to the W3C's wine ontology and to the results of [Kong et al., 2006]. Out of the 80 classes our system extracted, 11 were already part of the W3C's wine ontology. We judge the remaining 69 relevant for a Wine ontology, so they could be used to extend this existing ontology. Our system also extracted 6 relations as listed in Table 2.5, apart from `instanceOf` and `subClassOf`, all of them are relevant for a wine ontology but not in the set of relations the W3C's wine ontology declares. As for the instances, we extracted 500 instances from NELL using a confidence threshold of 0.94 to filter NELL's beliefs. This experiment shows that our system performs better than [Kong et al., 2006] while proposing only relevant concepts, which allows us to assert it would be a good fit for the bootstrapping phase of ontology development.

2.3 Conclusion

Theretofore, in Chapter 1 we presented the domain description and the state of the art study based on the life-cycle of ontology evolution proposed by [Zablith et al.,

[11] https://www.w3.org/TR/owl-guide/wine.rdf
[12] "wine" experiment: full lists of terms for our system's output http://bit.ly/2EEKItn

TABLE 2.7: Comparison of the Number of Classes, Relations, and Instances between our proposed approach, [Kong et al., 2006]'s approach and the W3C's wine ontology

Approach	W3C's *wine* ontology	[Kong et al., 2006]'s wine ontology	Our Approach
Class Number	74	62	**80**
Property Number	**13**	7	6
Instance Number	161	98	**500**

2015]. We presented the current limitations for the state of the art research work.

In this section, we investigated two research hypotheses:

Firstly, in Section 2.1 we investigated *RH 1 An ontology may need to evolve after some changes in some ontologies it uses*, and we answered our first research question *RQ 1 How to detect the need of evolving an ontology through the observation of structural changes in the ontologies it uses?* Where we proposed a definition for a situation to detect the need of ontology evolution (i.e., when an ontology O uses some terms that has the namespace of another ontology O', then O' evolves). We listed the set of cases that could occur during the evolution of the imported ontology.

Secondly, in Section 2.2 we investigated *RH 2 Using existing knowledge sources may help to develop and evolve ontologies*, and we answered our second research question *RQ 2 How to take advantage of external knowledge bases to develop and evolve ontologies?* Where we proposed an original approach for ontology enrichment based on the usage of three external knowledge bases: DBpedia, WikiData, and NELL. Our experiments showed that our system performs better than [Kong et al., 2006] that is based on WordNet, and proposes only relevant concepts. This allows us to assert it would be a good fit for the evolution phase of ontology development, and could even be reused as a starting point to develop ontologies to avoid what is called the cold start problem (i.e., starting the development process of ontologies from a blank page).

Next, in Chapter 3 we will investigate our third research hypothesis: *RH 3 Ontology portals may contain traces of incoherences in the evolution of ontologies that use one another*, and we will answer our third research question: *RQ 3 How to detect and assess incoherences in the evolution of ontologies that use terms of one in another?*

Chapter 3

Assessing the Impact of Ontology Evolution

Overview

In this chapter we target our second research goal:
RG.2 *To study how the evolution and the quality of an ontology impacts the ontologies that use it.*
And starting from the research hypotheses:
RH.3 Ontology portals may contain traces of incoherences in the evolution of ontologies that use one another.
We investigate the following research question:
RQ.3 How to detect and assess incoherences in the evolution of ontologies that use terms of one in another?

Introduction

In the previous chapter Chapter 2, we introduced a definition to detect the need of evolution in Section 2.1 (targets *Phase 1 Detect the need for evolution* of the ontology evolution life-cycle [Zablith et al., 2015]) and we proposed a functionality that evolves ontologies with the help of external knowledge bases in Section 2.2 (targets *Phase 2 Suggest changes to evolve the ontology* of the ontology evolution life-cycle).

In this chapter, we contribute to the fourth phase of the life-cycle (*Phase 4 Assess and study the evolution impact*) where we will introduce a framework to identify and assess the impact of evolving connected ontologies.

We propose to illustrate this chapter with the continuation of the motivating scenario described in the introduction of this thesis. Amal, after noticing the evolution of the *Education* ontology, created the ontology *Childcare* $v_{1.2}$ in November 2017 (evolved from *Childcare* $v_{1.1}$) and that uses *Education* $v_{1.2}$ that is created in September 2017. Although it looks like a quite standard behavior, several issues might occur in this

evolution, if for example: 1. in *Childcare* $v_{1.2}$ Amal is still using old terms from *Education* $v_{1.1}$, or 2. Amal uses terms in *Childcare* $v_{1.2}$ that do not exist in any of the versions of the *Education* ontology.These kind of issues cause different problem in both of the ontologies and in the external artifacts that use these evolved ontologies.

In Section 3.1 we will introduce different ways of connecting ontologies. In Section 2.1 we have seen the impact of the evolution of an imported ontology. As a consequence of this evolution, the impacted ontology should be adapted to the changes and evolve accordingly. This creates a situation which we call ontology co-evolution (Section 3.2). In Section 3.3, we will present our method to analyze the LOV portal and the BioPortal based on our co-evolution definition (Section 3.2, Definition 2). In Section 3.4, we will describe the evaluation of our experiments. In Section 3.5, we will analyze and categorize our findings.

3.1 Re-usability of ontologies

As mentioned earlier, re-usability is considered as a good practice while designing an ontology [Simperl, 2009]. On the one hand, re-usability saves time for knowledge engineers while developing ontologies, but on the other hand it raises the problem of adapting one's ontology to the evolution of a re-used ontology and thus complicates the maintenance process.

Re-usability of ontologies can be manifested through different types of links. Authors in [Savic et al., 2019] define a set of basic links that connect ontologies together, described as:

1. *Sub-links* that are used to connect ontological entities of the same type, such as: (a) SubClassOf to connect classes, (b) SubObjectPropertyOf to connect object properties, (c) SubDataPropertyOf to connect data properties, and (d) SubAnnotationPropertyOf to connect annotation properties.
2. *Assertion* links that represent associations between ontological entities induced by assertion axioms. Assertion axioms are the facts that describe relations between objects, such as same objects or different objects, or to specify that an object is an instance of a particular class.
3. *Equivalent* links that are used to state that two ontological entities are equivalent, such as: (a) EquivalentClasses between two classes (b) SameIndividuals between two objects, and (c) EquivalentObjectProperties between two object properties.
4. *Disjoint* links that state that two ontological entities are not equivalent. Disjoint links can connect classes, objects or object properties.
5. *References* links that connect anonymous classes with named classes.
6. *Contains* links that associate ontologies or ontology modules to other ontological entities.

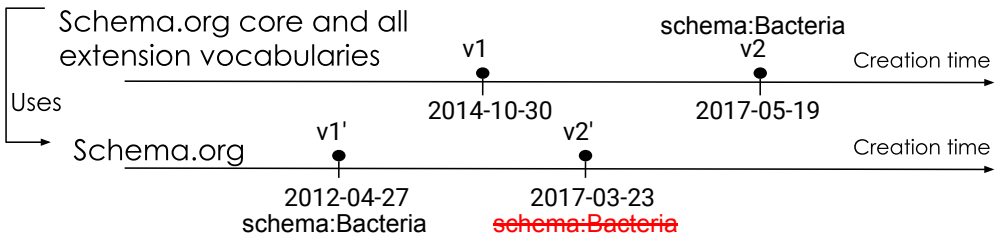


FIGURE 3.1: An introduction example to ontology co-evolution

7. *Imports* links that denote dependencies between ontology modules.
8. *Ad hoc* links that represent user-defined associations between classes determined by relevant pairs of ObjectPropertyDomain and ObjectPropertyRange axioms.

These different links are combined together to construct an *ontology graph*.

In the chapter, we are interested in observing the impact of the evolution of a special case of ontology evolution. For example, as shown in Figure 3.1 the ontology *Schema.org core and all extension vocabularies (O)* has a version that was published in 2014-10-30 (v_1), and that uses terms from the ontology *Schema.org (O′)* that was created in 2012-04-27 (v'_1). O' evolved to v'_2 in 2017-03-23, and similarly, O evolved to v_2 in 2017-05-19. The term `Bacteria` was deleted in v'_2, yet v_2 still uses it. This has an impact (O needs to be adapted), and illustrates an issue that might arise when an imported ontology evolves.

This situation of ontology evolution we called "Ontology co-evolution". In the upcoming sections, we propose a frame for observing the impact and adaptation to the evolution of an imported ontology (i.e., ontology co-evolution). Then we explain the occurrences of such cases in the history of ontologies in two ontology portals.

3.2 Observing the adaptation to the evolution of an imported ontology

The term "ontology co-evolution" has been already used in three research papers. Authors in [Kupfer et al., 2006, Kupfer and Eckstein, 2006] define co-evolution as the integration between the database schemes and ontologies to design and evolve the targeted ontologies. Also [Ottens et al., 2007] defines the co-evolution as the creation of ontologies by extracting terms and relations from text by means of natural language techniques. These definitions are irrelevant to the problem we investigate. We define then ontology co-evolution as:

Definition 2 ***Ontology Co-Evolution***

Ontology co-evolution is a situation where: O is an ontology which has at least two versions v_1 and v_2. O' is a different ontology which has at least two versions v'_1 and v'_2. O uses terms that have the namespace of O'. And let's note time(v) the creation time for a version.

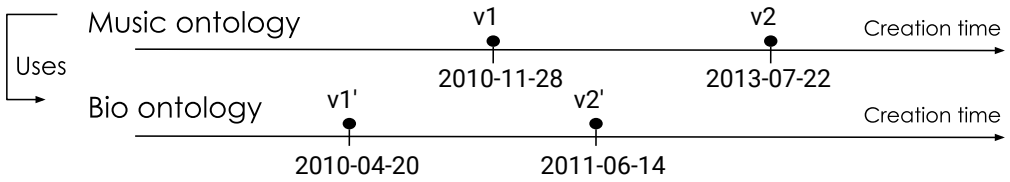


FIGURE 3.2: A time line showing the creation times of the music ontology (mo) and the bio ontology (bio), where mo uses terms that are defined by bio

In order to have a co-evolution case between O and O' with the ontology versions $\langle v_1, v'_1, v_2, v'_2 \rangle$, the following condition must be satisfied:
$time(v_1) < time(v_2) \wedge time(v'_1) < time(v'_2) \wedge time(v'_1) < time(v_1) \wedge time(v'_2) < time(v_2)$

We mentioned in Section 2.1 that a RDF term is defined as: $IRI \cup B \cup L$, where B: blank nodes, and L: literals. In this definition we take into consideration only the set of IRI.

Based on this definition, we propose an exhaustive categorization of the different cases that can arise during the adaptation to ontology evolution (i.e., co-evolution). In order to illustrate this categorization, Figure 3.2 presents an example of one case of ontology co-evolution: the *Music* ontology has two versions (v_1: mo_2010-11-28, v_2: mo_2013-07-22) that are respectively using the two versions of the *Bio* ontology (v'_1: bio_2010-04-20, v'_2: bio_2011-06-14).

During the evolution of O', terms may be added or deleted (notice that the changes we target in this thesis consist of a sequence of deletion, addition of terms). We identified the occurrences of adaptation to ontology evolution of O and O'. We observe the set of terms that has the namespace of O'. Table 3.1 shows the different cases that may occur. The left circles represent the set of terms that exist in the first version of an ontology (i.e., v_1 and v'_1), and the right circles represent the set of terms that exist in the second version of an ontology (i.e., v_2 and v'_2). t is a term that has the namespace of O'.

Back to our illustrating example, let us assume that Amal finally noticed the evolution of *Education* ontology and decided to evolve her ontology *Childcare* to $v_{1.2}$ on November 2017. Based on Definition 2, the ontology *Childcare* is considered as O which has two versions v_1: *Childcare* $v_{1.1}$, created in May 2017 and v_2: *Childcare* $v_{1.2}$, created in November 2017. The ontology *Education* is considered as O' and has two versions v'_1: *Education* $v_{1.1}$, created in January 2017 and v'_2: *Education* $v_{1.2}$, created in September 2017. Amal is using the term `edu:programOfStudy` from O'. Following each line of Table 3.1, the following set of cases might occur during the life journey of Amal's ontology:

TABLE 3.1: The set of cases that might happen during the ontology co-evolution considering a term t that has the namespace of O'

		a	b	c	d
	O / O′	v_1 v_2	v_1 t v_2	v_1 t v_2	v_1 t v_2
1	v_1' v_2'	Case 1.a: No changes occurred	Case 1.b: Term is used in v_1 but doesn't exist in O′	Case 1.c: Term is used in both v_1 and v_2 but doesn't exist O′	Case 1.d: Term is used in v_2 but doesn't exist O′
2	v_1' t v_2'	Case 2.a: Term is deleted from v_2' and not used in O	Case 2.b: Term is deleted in v_2' but still used in v_1	Case 2.c: Term is deleted in v_2' but still used in both v_1 and v_2	Case 2.d: Term is deleted in v_2' and still used in v_2
3	v_1' t v_2'	Case 3.a: Term exists in both v_1' and v_2' and not used in O	Case 3.b: Term exists in both v_1' and v_2' and used in v_1	Case 3.c: Term exists in both v_1' and v_2' and used in both v_1 and v_2	Case 3.d: Term exists in both v_1' and v_2' and used in v_2
4	v_1' t v_2'	Case 4.a: Term introduced in v_2' and not used in O	Case 4.b: Term exists in v_2' and used in v_1	Case 4.c: Term exists in v_2' and used in both v_1 and v_2	Case 4.d: Term exists in v_2' and used in v_2

1. No changes over the terms of v_1', or v_2'

Case 1.a There is no change of t to detect, therefore there is no interest in studying this case.

Case 1.b Amal made a typo by using the term `edu:programmOfStudy` (i.e., program is written with two "m"s instead of one) in v_1, but then she realizes that this term does not exist in O'. She fixes this mistake by not using it in v_2 anymore.

Case 1.c Amal uses t in both v_1 and v_2. This case might be explained by the fact that t is defined in a previous version (e.g. v_0) of the ontology O' (i.e., $t(v_0') < t(v_1')$).

Case 1.d Amal introduces a mistake by using t in v_2.

2. t is deleted in v_2'

The owners of O' decided to stop using the term `edu:programOfStudy` in v_2':

Case 2.a Amal does not use t that was recently deleted. Hence v_1 and v_2 were not affected.

Case 2.b Amal realizes that t was deleted, so she stops using it in v_2.

Case 2.c Amal does not realize the deletion of t, and she keeps using it in v_2.

Case 2.d Amal starts to use t in her second version (v_2), which introduces a mistake.

3. t exist in both v_1' and v_2'
None of the cases (3.a, 3.b, 3.c, and 3.d) is problematic.

4. t is added to v_2'
The owners of O' introduced a new term `edu:boardingSchool` in v_2':

Case 4.a Amal has not noticed the addition of t, even if it might be interesting for her to introduce it.

Case 4.b Amal was already using t in (v_1), but she decided to remove it from v_2.

Case 4.c Amal was already using t in v_1, and she continues using it in v_2.

Case 4.d Amal realizes the addition of t, and she start using it in v_2.

Cases 4.b and 4.c are corner cases that are discussed further in section Section 3.5.

We report on an experiment to systematically observe the occurrences of these cases in two ontology portals: LOV and BioPortal. The next sections present our analysis process.

3.3 Analyzing the adaptation of the evolution over ontology portals

The main aim of ontology portals is to group different ontologies in order to facilitate the process of finding and reusing them. Moreover, ontology portals are convenient to keep versions of ontologies. These versions are time stamped in order to keep track of their creation time. This facilitates our process to extract the time of each version.

This section presents the two ontology portals we used and the method we applied to detect the occurrences of ontology co-evolution. Section 3.3.1 presents LOV portal [Vandenbussche et al., 2017], and Section 3.3.2 presents BioPortal [Whetzel et al., 2011]. We have selected these two portals because they are the ones that reference the greatest number of vocabularies available on the Web.

3.3.1 Analyzing the adaptation of ontology evolution over the Linked Open Vocabulary (LOV)

Linked Open Vocabulary (LOV) [Vandenbussche et al., 2017] is considered a rich repository of ontologies. LOV's main goal is to help publishers and users of linked data and vocabularies to assess, reuse, and publish different vocabularies based on their needs. LOV currently references 648 different vocabularies,[1] each one being described with different properties, such as number of incoming links (i.e., how many ontologies are using ontology O), number of outgoing links (i.e., how many ontologies

[1] Last counted on June 2018

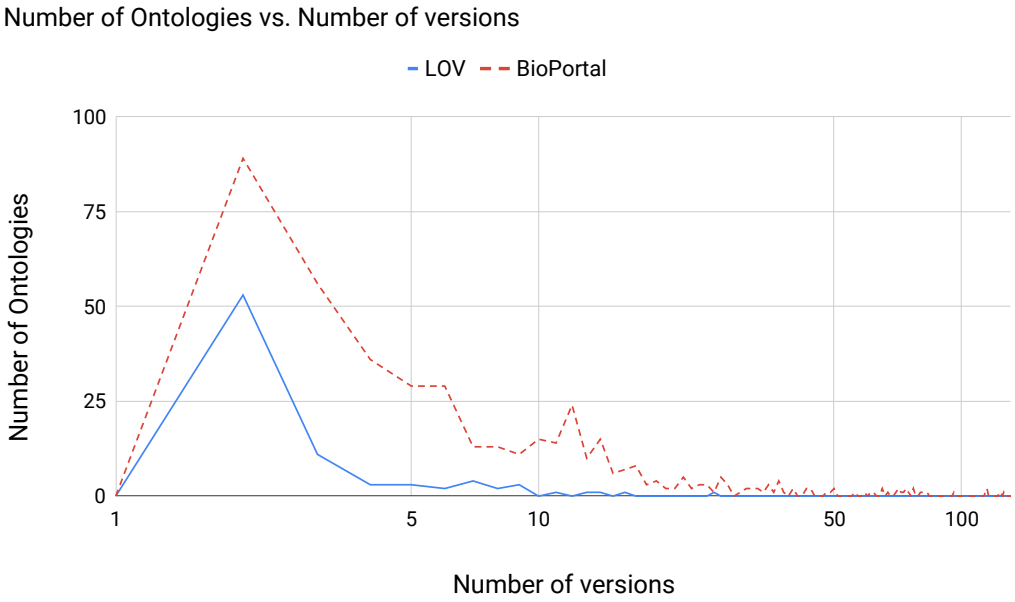


FIGURE 3.3: The relation between the total number of versions and the number of ontologies that have the specific number of versions for the ontologies that are referenced in LOV and BioPortal

are used by ontology O), number of different versions, and datasets that are using ontology O.

Out of the 648 ontologies of LOV, there are 88 ontologies that have evolved with a total number of 344 versions. The number of different versions that is associated with each ontology varies. For example the *FOAF*[2] ontology has 10 available versions to download from LOV ontology portal. Figure 3.3 shows the relation between the total number of versions and the number of ontologies that have the specific number of versions.

However, not all the ontologies that evolved are connected (i.e., using terms from one to another). In order to retrieve the set of ontologies that satisfy the conditions in Definition 2, we first issued a SPARQL query (Listing 3.1) over the LOV RDF dump in order to retrieve all ontologies that have at least two different versions and at least 1 incoming link. The result is 46 different ontologies and a total of 205 different versions.

Second, we used Apache-Jena in order to get all the different ontologies that have more than one outgoing links. This decreased the list of versions to 198. Third, we extracted all the creation times for the different ontologies versions, and we filtered them based on the selection criteria of Definition 2. As a result we identified 74 cases of ontology co-evolution, involving 28 different ontologies.

[2]Available at: https://lov.linkeddata.es/dataset/lov/vocabs/foaf

```
PREFIX voaf:<http://purl.org/vocommons/voaf#>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX dcat: <http://www.w3.org/ns/dcat#>

SELECT DISTINCT * WHERE {
 GRAPH <http://lov.okfn.org/dataset/lov>{

 ?vocab a voaf:Vocabulary ;
  dcterms:title ?title ;
  dcterms:modified ?modified ;
  voaf:reusedByDatasets ?dataset ;
  voaf:reusedByVocabularies ?import ;
  dcat:distribution ?distrib .

 ?distrib dcterms:issued ?issued .

 FILTER ( !isBlank(?distrib) )
 FILTER ( ?import >0 )
 BIND (STR(?issued) AS ?date)
 }
} ORDER BY DESC(?dataset)
```

Listing 3.1: This query returns all ontologies which have at least 2 versions and at least 1 incoming link

3.3.2 Analyzing the adaptation of ontology evolution over BioPortal

BioPortal [Whetzel et al., 2011] is an open repository for biomedical ontologies. BioPortal's main goal is to make biomedical knowledge and data available on the internet using ontologies. This is useful for boosting biomedical science and clinical care domains. BioPortal currently references 770 different ontologies,[3] each one being described with different properties, such as the number of different versions, along with general metrics (e.g. number of classes, properties and instances).

Out of the 770 ontologies of BioPortal, 485 ontologies have evolved with a total number of 15,025 versions. An example is the *HIV*[4] ontology. It has 12 available versions. Figure 3.3 shows the relation between the total number of versions and the number of ontologies that are associated with the specific number of versions. As explained earlier, only some of the ontologies that evolved satisfy our configuration of ontology co-evolution.

In order to retrieve the set of ontologies that satisfy the conditions in Definition 2, we firstly used the BioPortal API[5] to retrieve all ontologies that have at least two versions. Secondly, we used Apache-Jena in order to get all different ontologies that have more than one outgoing link. And thirdly, we extracted all the creation times for the different ontology versions, and we filter them based on the selection criteria

[3] Last counted on January 2019

[4] Available at: http://data.bioontology.org/ontologies/HIV/submissions

[5] http://data.bioontology.org/documentation

of Definition 2. We identified 14 cases of ontology co-evolution, involving 10 different ontologies.

3.4 Identification of the occurrences of adaptation to ontology evolution

In this section we present the results of an experiment[6] to detect ontology co-evolution using the cases that are defined in Section 3.2. In Section 3.5 we discuss these results in more details.

We retrieved from LOV the set of 28 ontologies with 74 co-evolution instances (Section 3.3.1). As for BioPortal we retrieved a set of 10 ontologies with 14 co-evolution instances(Section 3.3.2).[7] Appendix C lists the different co-evolution instances.

We extracted the set of terms for each version, and the name spaces for the used ontologies (O''s versions), and we used them to compute the number of occurrences of the different cases.

Table 3.2 shows the number of occurrences for each co-evolution case for LOV (first value in each cell) and BioPortal (second value).

Now we discuss further different interesting cases from our experiment. In the following subset of cases, a version of O uses a term t that has the namespace of O', however t is not defined in the two versions of O':

Case 1.b (i.e., a term t is used in v_1, however it does not exist in v'_1 and v'_2): This co-evolution case occurred 130 times in BioPortal but never in LOV. An example is the co-evolution process of the *Schema.org core and all extension vocabularies* (v_1: created in 2014-10-30 and v_2: created in 2017-05-19), with *Schema.org* ontology (v'_1: created in 2012-04-27 and v'_2: created in 2017-03-23). The terms `Bacteria`, `FDAcategoryC` and `Diagnostic` are used in v_1, however they do not exist in v'_1 and v'_2.

Case 1.c (i.e., a term t is used in both v_1 and v_2, however it does not exist in v'_1 and v'_2): This case occurred 3 times in LOV and 929 times in BioPortal. An example is the co-evolution process of the *Statistical Core Vocabulary* (v_1: created in 2011-08-05 and v_2: created in 2012-08-09), with *DCMI Metadata Terms* (v'_1: created in 2010-10-11 and v'_2: created in 2012-06-14). The terms `dc:status` and `dc:partOf` are used in v_1 and v_2, however they do not exist in v'_1 and v'_2.

Case 1.d (i.e., a term t is used in v_2, however it does not exist in v'_1 and v'_2): This case occurred 3 times in LOV and 115 times in BioPortal. An example is the co-evolution process of the *Europeana Data Model vocabulary* (v_1: created in 2012-01-23 and v_2:

[6] The experiment can be found at: https://github.com/OmarAlqawasmeh/coEvolutionTermsExtraction (Full results can be found inside *resources* folder)

[7] The co-evolution cases of LOV and BioPortal are inside the *resources* folder at https://github.com/OmarAlqawasmeh/coEvolutionTermsExtraction

TABLE 3.2: The number of occurrences for each co-evolution case for LOV (first value) and BioPortal (second value) with respect to namespace of (O'). The values inside the parentheses represents the number of cases where these different occurrences took place. For example, in case 1.c, the values 3 (2) indicates that there were 3 cases of 1.c that occurred in 2 instances of co-evolution.

	a	b	c	d
uses: O / O′	v_1 v_2	v_1 v_2 t	v_1 v_2 t	v_1 v_2 t
1	v_1' v_2'	0 130 (1)	3 (2) 929 (2)	3 (2) 115 (1)
2	v_1' v_2' t	23 (18) 27 (10)	0 0	0 3 (1)	0 0
3	v_1' v_2' t	16875 (73) 9135 (12)	10 (6) 0	270 (66) 2058 (8)	23 (13) 0
4	v_1' v_2' t	2420 (29) 1560 (12)	0 115 (1)	0 908 (4)	0 0

created in 2013-05-20), with *Dublin Core Metadata Element Set* (v_1': created in 2010-10-11 and v_2': created in 2012-06-14). The terms `dc:issued` and `dc:modified` are used in v_2 however they do not exist in v_1' and v_2'.

In the following subset of cases a term t is defined in v_1' and is deleted in v_2':

Case 2.a (i.e., a term t is deleted in v_2', and it is not used in any of O's versions): This case occurred 23 times in LOV and 27 times in BioPortal. This is a normal case, and no problem occurred during the co-evolution.

Case 2.b (i.e., a term t is deleted in v_2', and then deleted in v_2): This case has no occurrences in LOV nor in BioPortal. We are not discussing it further.

Case 2.c (i.e., a term t is deleted in v_2', however it is used in v_1 and still in v_2): This case occurred 3 times in BioPortal. It shows a problem of using terms that do not exist anymore in O'. For example in the co-evolution process of the *Schema.org core and all extension vocabularies* (v_1: created in 2014-10-30 and v_2: created in 2017-05-19), with *Schema.org* ontology (v_1': created in 2012-04-27 and v_2': created in 2017-03-23). The terms `MedicalClinic`, `Optician` and `VeterinaryCare` are used in both v_1 and v_2, however they do not exist in the latest version of O' (these different terms were deleted from v_2 of *Schema.org*).

Case 2.d (i.e., a term t is deleted in v_2', however it is added in v_2): This case has no occurrences in LOV nor in BioPortal. We are not discussing it further.

Cases 3.a, 3.b, 3.c, and 3.d are not problematic cases, so they are not investigated further.

In the following subset of cases a term t is introduced in v_1':

Case 4.a (i.e., a term t is added in v_2', and it was not used in v_1 or v_2): There were 2,420 terms added to v_2' in the ontologies that are referenced in LOV, and 1,560 terms were added to v_2' in the ontologies that are referenced in BioPortal. These different terms are not used in v_1 or v_2. An example is the co-evolution process of the *Semanticscience Integrated Ontology (SIO)* (v_1: created in 2015-06-24 and v_2: created in 2015-09-02), with *The Citation Typing Ontology (CITO)* (v_1': created in 2010-03-26 and v_2': created in 2015-07-03). The term `isDocumentedBy` from O' could be useful to use by O, thus the owners of O can be notified and recommended to use it.

Case 4.b (i.e., a term t is added in v_2', and it was already used in v_1): This case occurred 115 times in BioPortal, and it has no occurrence in LOV. An example is the co-evolution process of the *Schema.org core and all extension vocabularies* (v_1: published in 2014-10-30 and v_2: published in 2017-05-19), with *Schema.org* ontology (v_1': created in 2012-04-27 and v_2': created in 2017-03-23). The terms `SoundtrackAlbum`, `Hardcover` and `SingleRelease` are used in v_1, however they were introduced later in v_2'.

Case 4.c (i.e., a term t is added in v_2', and it was already used in both of O's versions): This case occurred 951 times in BioPortal, and it has no occurrence in LOV. An example is the co-evolution process of the *Semanticscience Integrated Ontology (SIO)* (v_1: created in 2015-06-24 and v_2: created in 2015-09-02), with *The Citation Typing Ontology (CITO)* (v_1': created in 2010-03-26 and v_2': created in 2015-07-03). The term `citesAsAuthority` is used in both v_1 and v_2, however it was introduced in v_2'.

Case 4.d (i.e., a term t is added in v_2', and v_2 starts to use it): This case has no occurrences in LOV or BioPortal, so we are not investigating them.

3.5 Discussion

Table 3.3 extends Table 1.6 and includes our proposed approach to compare with the existing state of the art approaches. The process we have followed for our experiment can be closely compared to the one followed by [Abdel-Qader et al., 2018], where:

1. Both analysis observe the additions and deletions of terms, however in [Abdel-Qader et al., 2018] they observe how the terms are changed and adopted in the evolving ontologies, where we observe the changes when two ontologies are connected to each other as defined in Definition 2.
2. 13 ontologies with 37 versions that are referenced in LOV were used in the experiments of [Abdel-Qader et al., 2018], where we retrieved a total of 38 ontologies referenced in LOV and BioPortal and we observed 88 evolution cases.

Table 3.3: A comparison between the state of the art approaches for assessing the impact of ontology evolution (Table 1.6 and our proposed approach

	Ontology operation	Experimental evaluation	Dataset
Dragoni and Ghidini [Dragoni and Ghidini, 2012]	Rename, Delete, and Move concepts	Detect changes and measure the impact on a search system	Two document collections annotated by MeSH light-Ontology along with 75 queries
Abgaz et al. [Abgaz et al., 2012]	Structural and Semantic changes	Set of predefined rules to observe the impact, and 4 experts to evaluate it	An ontology that contains 80 classes, 8 data proprieties, 10 object proprieties, and 500 axioms.
Grob et al. [Groß et al., 2012]	Addition (category, relation) Deletion (category, relation), Merge, move and split category	Stability measure to check how the changes might affect the statistical applications for the experimental and simulated data	GENE ontology versions
Mihindukulasooriya et al. [Mihindukulasooriya et al., 2016]	Addition and Deletion for (Sub-)Classes and (Sub-)Properities	Observe the different changes during the life time of the targetted ontologies	DBpedia, Schema.org, PROV-O, and FOAF
Abdel-Qader et al. [Abdel-Qader et al., 2018]	Addition and Deletion of terms	How terms are changed and adopted in the evolving ontologies	Selected ontologies from LOV
Our proposed approach	Addition and Deletion of terms	How terms are changed and adopted in the co-evolution situation	Ontologies from LOV and BioPortal that satisfy co-evolution definition

After analyzing our results, we confirm the observation of [Groß et al., 2012, Kirsten et al., 2009] showing that in general the addition of terms occurs more frequently than the deletion of terms during the evolution process. Table 3.4 shows the number of added terms comparing to the number of deleted terms for the set of ontologies that are referenced in LOV and BioPortal that satisfy our definition of co-evolution (Definition 2). In order to calculate these numbers, we kept track of the evolution of the different ontologies we collected. For example, in total there were 26 terms that were added as a cause of the evolution of v_1 to v_2.

Table 3.4: Number of added terms comparing to number of deleted terms in both LOV and BioPortal

Portal	LOV		BioPortal	
	Added	Deleted	Added	Deleted
v_2	26	10	115	245
v_2'	2420	23	2583	30

From our experiment and results, it appears that the different cases can be conveniently classified into three categories:

1. Assessment of Good Practices 2. Detection of Wrong Practices (Pitfalls) 3. Uncertain cases. In the next subsections we discuss these different categories.

3.5.1 Assessment of good practices

Case 2.a in LOV and BioPortal shows a good practice from the owners of O'. They noticed that the term t is not used in both v_1 and v_2 so they decided to delete it from v'_2.

Cases 2.b and 2.d have no occurrences in all of the ontologies that are referenced in both LOV and BioPortal. This is the preferred case of ontology evolution, and it is one indicator of the quality of the co-evolution. For instance, case 2.b indicates that the set of ontologies stops using the terms after they have been deleted in O', and case 2.d indicates that there were no mistake of using the set of deleted terms in the newest version of O.

3.5.2 Detection of wrong practices

Cases (1.c and 1.d) from LOV and cases (1.b, 1.c and 1.d) from BioPortal, demonstrate the problem of using terms that do not exist in v'_1 and v'_2. A possible explanation is that these terms were used from a previous version of O'. Let's assume that this previous version is v'_0, then these cases can happen only if the publishing time of $t(v'_0)$ is before the publishing time of $t(v'_1)$. In these cases, the owners of O, should be notified of the changes, and they should be suggested to delete the terms that do not exist any more.

Case 2.c from BioPortal shows that some terms are still used in both of O's versions after being deleted from O'. In order to prevent such kind of problems the owners should be notified about these cases.

Case 4.b from BioPortal shows that some terms have been already used in v_1, however they were added later in v'_2. The v_1 of *Schema.org core and all extension vocabularies* uses terms that were later defined by v'_2 of *Schema.org* ontology. The *Schema.org core and all extension vocabularies* is an extension of *Schema.org*, however it has its own namespace. Each reviewed extension for schema.org has its own chunk of schema.org namespace (e.g. if extension name is x, the namespace of this extension is x1.schema.org).[8] We retrieved all terms that has the namespace of *Schema.org*.[9] Other terms with different namespaces were discarded.[10] This reflects a bad practice

[8]More details about the extensions managing of schema.org can be found at: https://schema.org/docs/extension.html

[9]namespace of *Schema.org* is http://schema.org/

[10]Some examples of discarded namespaces: https://health-lifesci.schema.org/, https://pending.schema.org/, https://meta.schema.org/

in a way of using terms that have not been defined in the second version. These terms could be a harbinger to add in the next versions.

Case 4.c from BioPortal shows that some terms are used in v_1 and v_2, however they were firstly introduced in v_2'. Both of v_1 and v_2 of *Semanticscience Integrated Ontology (SIO)* use terms that were later defined by v_2' of *The Citation Typing Ontology (CITO)*. The term `citesAsAuthority` was firstly defined in v_2' *The Citation Typing Ontology (CITO)*, however there is an object property that has the same name in v_1'. One explanation for this kind of errors is that the knowledge engineers might introduce a typo during the development process of the ontology. In these cases, the owners of O, should be notified that the term they use is not a term. They should look at it carefully and possibly delete it.

3.5.3 Uncertain cases

In cases (3.a, 3.b, 3.c and 3.d) from LOV and BioPortal, there was no change of terms in the two versions of O'. This indicates that the co-evolution process has no problem to report. Some terms are shared between v_1' and v_2' so there was no addition or deletion over them.

Cases 4.a and 4.d in both LOV and BioPortal shows the number of terms that were added during the evolution of O'. These terms were not used in any of O versions. These cases can be explained in two ways:

1. The owners of O did not notice the addition of these terms, however they might be interested in using some of these new terms. This might introduce a problem, thus further content analysis should be introduced to possibly recommend changes to the owners.
2. The owners of O noticed the addition of these terms and they decided not to add them.

3.6 Conclusion

Theretofore, in Chapter 1 we presented the domain description and the state of the art study based on the life-cycle of ontology evolution proposed by [Zablith et al., 2015]. We presented the current limitations for the state of the art research work.

In Chapter 2 we investigated in two research parts:

1. We proposed a definition that helps to detect the need of evolution by observing the evolution of imported ontologies (Section 2.1). This part investigated our first research hypothesis: *RH 1 An ontology may need to evolve after some changes in some ontologies it uses*, and it answered our first research question: *RQ 1 How to detect the need of evolving an ontology through the observation of structural changes in the ontologies it uses?*

2. We proposed a functionality that takes advantage of existing knowledge bases to evolve ontologies (Section 2.2). This part investigated our second research hypothesis: *RH 2 Using existing knowledge sources may help to develop and evolve ontologies* and it answered our second research question: *RQ 2 How to take advantage of external knowledge bases to develop and evolve ontologies?*

In this chapter, we investigated our third research hypothesis: *RH 3 Ontology portals may contain traces of incoherences in the evolution of ontologies that use one another*, and we answered our third research question: *RQ 3 How to detect and assess incoherences in the evolution of ontologies that use terms of one in another?*

We showed that there is a need to formalize a conceptual frame for assessing the impact of ontology evolution and for tackling the different issues that arise during the evolution of ontologies. Hence, in Section 3.2 we have presented a situation of ontology evolution (i.e., ontology co-evolution) which considers the evolution of an ontology O that imports another one O' (i.e., O uses terms that have the namespace of O'). In both of Section 3.3 and Section 3.4 we provided an exhaustive categorization of the adaptation to ontology evolution for this situation. We observed these cases over two ontology portals:

1. The Linked Open Vocabulary (LOV) ontology portal which references 648 different ontologies, 88 of them evolved. We identified 74 cases of ontology co-evolution, involving 28 different ontologies.

2. The BioPortal which references 770 different ontologies, 485 of them evolved. We identified 14 cases of ontology co-evolution, involving 10 different ontologies.

Our proposed definition, i.e., ontology co-evolution can be used as a first step test to validate ontologies before referencing them in ontology portals. This will help to detect the problematic cases and enhance the quality of the referenced ontologies. As a direct research perspective to this research task, what could be interesting is to investigate other ontology portals, such as: AgroPortal [Jonquet et al., 2018a] that is an ontology portal for the agronomy domain.

The usage of ontologies is increasing, so there is the need of managing them, especially in the evolution process. We emphasize the need of having a service that can automatically observe and notify the owners of the ontologies during the evolution process. Having such tool can help to keep track of the different ontologies during the co-evolution and help to facilitate the process of ontology evolution.

Next, in Chapter 4 we will investigate our fourth research hypothesis *RH 4 Identifying pitfalls that affect ontology networks and versioned ontologies may help to design better ontologies*, and we will answer the last two research questions, *RQ 4 What pitfalls affect ontology network?* and *RQ 5 What pitfalls affect versioned ontologies?*

Chapter 4

Pitfalls in Networked and Versioned Ontologies

Overview

In this chapter we target our second research goal:
RG.2 *To study how the evolution and the quality of an ontology impacts the ontologies that use it.*
And starting from the research hypotheses:
RH.4 Identifying pitfalls that affect ontology networks and versioned ontologies may help to design better ontologies.
We investigate the following research question:
RQ.4 What pitfalls affect ontology network?
RQ.5 What pitfalls affect versioned ontologies?

Introduction

In Chapter 3, we have identified the different cases (good, pitfalls and uncertain) that could occur during ontology co-evolution. In this chapter we will extend our approach and investigate deeply ontology pitfalls in two situations: 1. versioned ontologies and, 2. ontology networks.

We propose to illustrate this chapter with the continuation of the motivating scenario described in the introduction of this thesis. We show that during the versioning of her ontology, Amal could face several issues, such as: 1. Her first version of *Childcare* is not accessible any more by its IRI. This presents a pitfall that is related to versioned ontology. 2. the ontology network that is created by Amal, could cause problems in case of importing an inconsistent ontology. This presents a pitfall that is related to ontology networks.

In Section 4.1 we will present a formal definition of ontology networks. Section 4.2 will present a formal definition of versioned ontologies. Section 4.3 will present our

new categorization for the ontology pitfalls along with 9 new candidate pitfalls that are related to versioned ontologies and ontology networks. And in Section 4.4 we will present our experimental evaluation to assess the importance and impact of the pitfalls over versioned and networked ontologies.

4.1 Ontology networks

The term "ontology network" is informally defined by [Savic et al., 2019, Suárez-Figueroa et al., 2012b, Haase et al., 2006] as the set of ontologies that are connected to each other via a variety of relationships (e.g. `owl:imports`, modularization, version). We are not aware of another definition of this term in the literature. However, authors in [Poveda Villalón et al., 2012] studied 18,589 terms appearing in 196 ontologies, and they concluded that *Uses* and *Imports* are the main relationships between ontologies. Based on their conclusion, in this chapter we propose a formal definition of an ontology network as:

Definition 3 ***An ontology network*** *An ontology network is a directed graph* $G = (\mathcal{N}, \mathcal{E})$*, consisting of a set* $\mathcal{N}$ *of ontologies and a set* $\mathcal{E}$ *of relationships, which are ordered pairs of elements of* $\mathcal{N}$*. Furthermore, every ontology* $O \in \mathcal{N}$ *has an owner* $\text{author}(O)$*, an IRI* $\text{iri}(O) \in \text{IRI}$*, an ontology series IRI* $\text{series\_iri}(O) \in \text{IRI}$*, a namespace* $\text{ns}(O) \in \text{IRI}$*, and a publication date* $\text{date}(O) \in \mathbb{N}$*; Every ontology relationship* $e \in \mathcal{E}$ *is labeled by a non-empty set of types* $\text{type}(e) \in \mathcal{T}$*.*

Based on [Poveda Villalón et al., 2012] conclusions, we limit our study to the two main types of relationships between the different ontologies. $\mathcal{T} = \{uses, imports\}$:

uses $uses \in \mathcal{T}$ happens when an ontology O uses a term t (that is, an IRI denoting an individual, a class or a property) that has the namespace of a different ontology O'.

imports $imports \in \mathcal{T}$ happens when an ontology O imports another ontology O', using the OWL importing mechanism.[1]

4.2 Versioned ontologies

Several research papers that concern ontology versioning tasks agree that ontology versioning is the process of creating/publishing a new version of an ontology O [Noy and Musen, 2004, Redmond et al., 2008, Rogozan and Paquette, 2005]. A more precise definition is proposed by [Klein and Fensel, 2001], where the authors define ontology versioning as: *the ability to handle changes in ontologies by creating and managing different variants of it.* We formally define and adopt the following definition:

[1] https://www.w3.org/TR/owl2-syntax/, sections 3.4

Definition 4 ***Versioned ontology*** *Let us assume that O is an ontology, that has a set of versions $\mathcal{V} = \{O_{v.1}, O_{v.2}, O_{v.3}, \dots\}$. O is considered a* versioned ontology *if* $|\mathcal{V}| \geq 2$.

4.3 The set of pitfalls over ontology networks or versioned ontologies

In Section 4.3.1 we present a new categorization of ontology pitfalls based on our analysis of the current research work (see Section 1.2). In Section 4.3.2 we describe 9 new candidate pitfalls that are related to versioned ontologies and ontology networks. This list of 9 pitfalls extends the list of stand-alone ontology pitfalls established by [Poveda Villalón, 2016].

4.3.1 Proposed categorization for ontology pitfalls

As described in Chapter 1, the term "pitfall" refers to the set of mistakes/errors that can be made during the development or usage of ontologies. In current related works, two types of pitfalls are yet identified: stand-alone ontologies pitfalls and networked ontologies pitfalls. In order to better analyse ontology engineering processes and pitfalls, we propose to introduce a third category, namely versioned ontology pitfalls. Thus, in order to put our research work in a right perspective, we propose to distinguish the three categories of pitfalls:

1. *Stand-alone ontology pitfalls:* can happen within a single ontology O that is created by $author(O)$ (e.g. *Childcare* $v_{1.1}$ from Figure 2).
2. *Versioned ontologies pitfalls:* can happen when an $author(O)$ creates/publishes a new version of the ontology O (e.g. the evolution of *Education* ontology from Figure 2).
3. *Ontology network pitfalls* can happen within the set of ontologies that are connected to each other, such as when an ontology O is connected to a different ontology O' (e.g. both of the ontologies *Childcare* and *Education* from Figure 2). The person responsible of resolving these pitfalls is either $author(O)$ or $author(O')$, depending if the pitfall occur in O or O'.

4.3.2 Candidate pitfalls

In this section, we list the candidate pitfalls we propose:

Pitfall 1. Ontology is not accessible at its IRI.
This pitfall is related to the ontology relationship $type(e) \in \{uses, imports\}$. It can occur in the following cases:

1. If an ontology was never published on-line, for instance, if an ontology is used internally by a company, and/or if an ontology file becomes private and it is not accessible anymore.
2. If the ontology is not available at its IRI anymore. For example: the IRI of the *pizza* ontology is http://www.co-ode.org/ontologies/pizza/pizza.owl#, but it is not accessible at this IRI anymore.
3. If the IRI of the ontology has been changed. For example: the IRI of the *DOLCE Ultralite upper* ontology was originally http://www.loa-cnr.it/ontologies/DUL.owl#. The website loa-cnr.it closed, and the ontology is now available at http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#.

This pitfall affects ontology networks. Any import of an ontology that has this pitfall will fail. To solve or avoid this pitfall, we suggest to verify the imported IRIs for any changes that could occur or to locally maintain a copy of the ontology and use it offline.

Pitfall 2. Importing an ontology using a non persistent IRI or the IRI of a representation (the file URL)
This pitfall is related to the ontology relationship $type(e)$ = "*imports*". Persistent identifier (PID) is resolvable unique name associated with a digital object. [Weigel et al., 2014] This means that if the object relocates to a different server or owner, the identifier name remains the same. Persistent IRIs follow the same idea of PIDs, where each IRI is permanently assigned to a particular resource. In the world of ontology engineering, using persistent IRIs is considered as a good practice while publishing an ontology (see Section 1.1.1).

1. If a knowledge engineer imports a non persistent IRI, for example: the *SEAS* ontology has persistent IRI https://w3id.org/seas/, which no longer redirects to the location https://ci.emse.fr/seas/. Assume an ontology imported the IRI https://ci.emse.fr/seas/. Due to the renaming of the EMSE institution, the IRI now redirects to the location https://ci.mines-stetienne.fr/seas/, thus the import would break.
2. If a knowledge engineer imports the file URL instead of the ontology IRI, for example: the W3C organization ontology has persistent IRI https://www.w3.org/ns/org, with two representations at https://www.w3.org/ns/org.rdf and https://www.w3.org/ns/org.ttl. Assume an ontology imports the ontology representation https://www.w3.org/ns/org.rdf instead of the ontology series https://www.w3.org/ns/org. In case of the deletion of the RDF/XML representation any import would break.

This pitfall affects ontology networks. Any import of an ontology that has this pitfall will fail. To solve or avoid this pitfall, we suggest: 1. to use only persistent IRIs when importing ontologies, and 2. to always use the IRI of the ontology, and not the URL of the file representation.

Pitfall 3. Importing an inconsistent ontology
This pitfall is related to the ontology relationship $type(e)$ = "*imports*". It occurs if a knowledge engineer imports an inconsistent ontology, for example: the *SAREF4ENER* ontology (EEbus/Energy@home) https://w3id.org/saref4ee is inconsistent. The importing ontology would become inconsistent too. This pitfall affects both ontology networks and versioned ontologies. To solve or avoid this pitfall, we suggest: 1. to check the consistency of an ontology before importing it, 2. to use only the specific terms that are needed from the ontology (i.e., by their IRIs) instead of importing the whole ontology, and/or 3. to try contacting the ontology owners so that they solve the inconsistency.

Pitfall 4. Only the latest version of the ontology is available online
This pitfall is related to the ontology relationship $type(e) \in \{$"*uses*", "*imports*"$\}$. It occurs when the only available version of the ontology is the latest version. For example, the *S4WATR* ontology is published at https://w3id.org/def/S4WATR, but only the latest version is available online. Let's assume an ontology imports the *S4WATR* ontology at a certain point in time. Later, some terms are deleted or added in S4WATR ontology. Then the importing ontology could break or become inconsistent.

This pitfall affects both ontology networks and versioned ontologies. To solve or avoid this pitfall, the following practices could be followed: 1. to import an ontology with its version URI, and/or 2. to monitor the evolution of the imported ontology to react appropriately.

Pitfall 5. Importing an ontology series IRI instead of an ontology version IRI
This pitfall is related to the ontology relationship $type(e) \in \{$"*uses*", "*imports*"$\}$. It occurs if a knowledge engineer imports an ontology series IRI instead of an ontology version IRI. For example, the *SAREF* ontology series has IRI https://saref.etsi.org/saref#, and version 2.1.1 has IRI https://saref.etsi.org/saref/v2.1.1/saref#. A new version 3.1.1 is under development and will delete terms from version 2.1.1. Let O' be an ontology that imports *SAREF* ontology 2.1.1 using https://saref.etsi.org/saref#. When the new version 3.1.1 is released, the importing ontology O' could break or become inconsistent.

This pitfall affects both ontology networks and versioned ontologies. To solve or avoid this pitfall we recommend to import the ontology version IRI instead of the ontology series IRI.

Pitfall 6. Ontology series IRI is the same as the ontology version IRI
This pitfall is related to the ontology relationship $type(e)$ = "*imports*". It occurs whenever a IRI refers to a specific version of the ontology. For example, the *Units of Measure (OM)* ontology version 1.8 has IRI http://www.wurvoc.org/vocabularies/om-1.8/, and version 2.0 has IRI http://www.ontology-of-units-of-measure.org/resource/om-2/. Each time a new version is published, the ontology IRI should be updated, this does not conform to the OWL2

specification.[2] This pitfall affects both ontology networks and versioned ontologies. To solve or avoid this pitfall we recommend the delete the version number from the IRI of the ontology, or to send a notification message with the new IRI when a new version of the ontology is released.

Pitfall 7. A term is moved from one ontology module to another
This pitfall is related to the ontology relationship $type(e) =$ "*uses*". It occurs when a term is moved from one ontology module to another, which causes the change in its IRI. For example, the *SAREF* ontologies [SmartM2M, 2019] consist of 1. *SAREF core*, 2. *SAREF4SYST*, and 3. several ontologies for the verticals, such as SAREF4ENER, SAREF4BLDG, and SAREF4ENVI. In *SAREF-core 1.1.1*, created in 2015, the authors defined the term `saref:BuildingObject`. Later in 2016, SAREF-core 2.1.1 was published without the term `saref:BuildingObject`. However, another ontology SAREF4BLDG was created with the term `sbldg:BuildingObject`, with the same definition as `saref:BuildingObject`.

In this case, the IRI of the term `saref:BuildingObject` has been changed. This might have a functional impact over the artifacts that are reusing the term (e.g. some queries might be affected by the change of the IRI).

This pitfall affects ontology networks. To solve or avoid this pitfall we recommend to take extra care while moving terms between the different modules, and to notify the users of the ontology in case of changes.

Pitfall 8. Namespace hijacking [from [Poveda Villalón, 2016]]
This pitfall is related to the ontology relationship $type(e) = \{$"*uses*"$\}$. It refers to reusing or referring to terms from another namespace that are not defined in such namespace [Poveda Villalón, 2016]. For example, the description of classes `qudt-1.1:QuantityValue` and `qudt-1.1:Quantity` are not available at their own IRIs. Instead, they are defined in the ontology `http://qudt.org/1.1/schema/quantity#`.

This pitfall affects ontology networks as it prevents the retrieval of valid information when looking for the hijacked terms, which violates the Linked Data publishing guidelines [Heath and Bizer, 2011]. To solve or avoid this pitfall we recommend to define new terms in the namespace that is owned and controlled by the knowledge engineer.

Pitfall 9. The IRI of a term contains a file extension
This pitfall is related to the ontology relationship $type(e) =$ "*uses*". It occurs if a term's IRI contains a file extension. For example, the terms in the *Dolce ultra lite* ontology have the namespace `http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#`. Let's assume that some day the publisher of *dolce-very-lite* wants to set up content negotiation to expose an HTML documentation of their ontology. As the IRI of the terms contains the file extension ".owl", no content negotiation should take place. If a human looks up the term IRI, he/she will access the OWL file, and not

[2] `https://www.w3.org/TR/owl2-syntax/`, sections 3.1 and 3.3

the HTML documentation. To solve or avoid this pitfall we recommend to stop using file extensions inside IRIs, and follow the rules of cool URIs for the semantic web.[3]

As a summary, Table 4.1 presents the set of pitfalls we have proposed. For each pitfall we describe the following criteria:

1. Affect: whether the pitfall affects ontology networks and/or versioned ontologies.
2. Problems that might occur as a consequence of having the pitfall.
3. Recommendations to avoid or solve the pitfall.

4.4 Evaluating the importance and impact of the candidate pitfalls

We evaluated the importance and potential impact of the candidate pitfalls using a survey we conducted in the semantic web community. Section 4.4.1 describes the survey, and Section 4.4.2 presents the quantitative evaluation of the answers. Finally Section 4.4.3 reports on the different opinions and suggestions we gathered from the participants.

4.4.1 Description of the survey

The survey[4] firstly requests some information about the level of expertise of the participant in: 1. ontology engineering in general, 2. versioned ontologies, and 3. networked ontologies.We used a Likert scale with values from 1 (beginner) to 10 (expert). Secondly, each pitfall is described with an illustrative example, and the participant is asked to answer to the following questions:

1. How often have you encountered this pitfall before?
2. How problematic is this pitfall?
3. How would you rate the impact on subsequent versions of the ontology?
4. How problematic is it to import ontologies that have this pitfall?

For the answers, we also use a Likert scale from 1 (Strongly Disagree) to 5 (Strongly Agree). For each pitfall, the participant may additionally share known occurrences of the pitfall, and ideas or recommendations to solve or avoid it. Finally, we ask the participant to what extent he/she agrees or not with our pitfalls categorization, and to rate about his/her overall confidence while filling the survey.

[3]Cool URIs can be found at: https://www.w3.org/TR/cooluris/
[4]The survey can be found at: http://bit.ly/36JQfgO

Table 4.1: A summary of the set of the 9 candidate pitfalls

Code	Description	Affects	Problem caused	Recommendations to solve or to avoid
P1	Ontology is not accessible at its IRI	Ontology networks and versioned ontologies	Import failure	Keep the ontology always available at its IRI by controlling and managing the possible changes during the evolution
P2	Importing an ontology using a non persistent IRI or the IRI of a representation (the file URL)	Ontology networks	Import failure	Try to always use a persistent IRI while importing an ontology
P3	Importing an inconsistent ontology	Ontology networks and versioned ontologies	Inheritance of the inconsistency	Make sure to import consistent ontologies or using specific terms instead of importing the whole ontology
P4	Only the latest version of the ontology is available online	Ontology networks and versioned ontologies	Import failure and/or the ontology become inconsistent	Import an ontology with its version URI, and/or follow the evolution of the imported ontology and change your ontology accordingly
P5	Importing an ontology series IRI instead of an ontology version IRI	Ontology networks and versioned ontologies	Import failure and/or the ontology become inconsistent	Import the ontology version IRI instead of the ontology series IRI
P6	Ontology series IRI is the same as the ontology version IRI	Ontology networks and versioned ontologies	Import failure	Delete the version number from the IRI, and/or to send a notification message with the new IRI when a new version is released.
P7	A term is moved from one ontology module to another with different IRI	Ontology network	Functional impact, research queries might break.	Take extra care when moving terms, and notify the different users in case of changes.
P8	Namespace hijacking	Ontology networks	Retrieve invalid information while looking up for the hijacked terms	Define terms in the namespace that is owned and controlled by the knowledge engineer
P9	The IRI of a term contains a file extension	Stand-alone ontology pitfall, however can affect ontology networks	No content negotiation should take place	Stop using file extensions inside IRIs

4.4.2 Quantitative evaluation of pitfalls

A total of 29 participants answered the survey between November 2019 to January 2020.[5] As shown in Figure 4.1, most of the participants declared expertise in ontology engineering, ontology networks and ontology versioning.

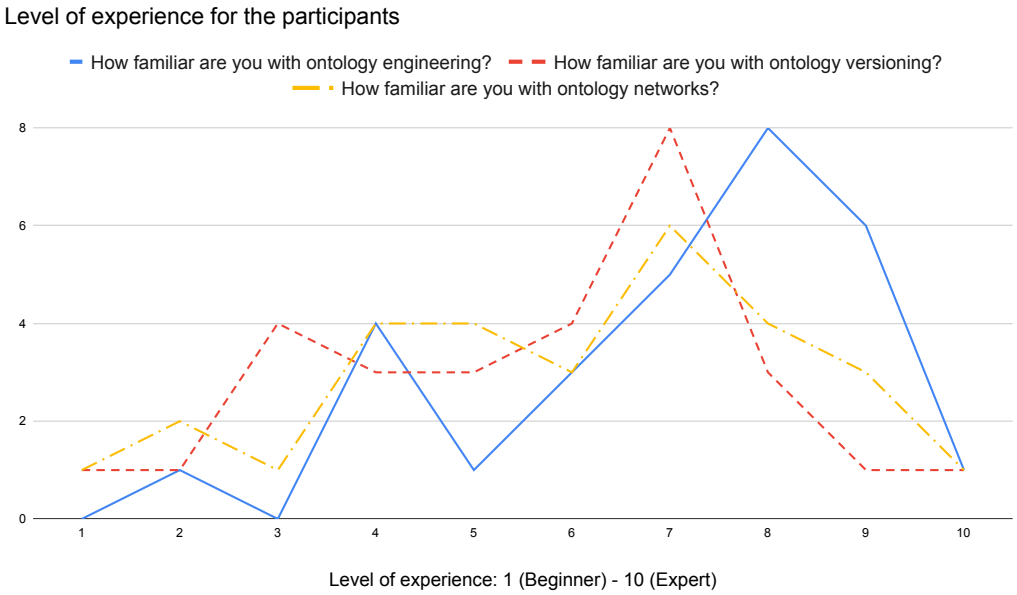


FIGURE 4.1: Level of experience for the participants (weighted average)

We consider that the value of the participants' opinion is increasing with their level of experience. Thus, we calculated the weighted average (WA) for the different answers for each pitfall:

$$\text{WA} = \frac{\sum_{i=1}^{N} w_i \cdot x_i}{\sum_{i=1}^{N} w_i}$$

where w_i is the value of expertise of participant i and x_i the response. Then, we assess the agreement level between the different participants using the consensus measure (Cns) proposed by [Tastle and Wierman, 2007]:

$$\text{Cns}(X) = 1 + \sum_{j=1}^{n} p_j \cdot \log_2 \left(1 - \frac{|X_j - \mu_X|}{d_X}\right)$$

where X is the values vector (i.e., values from 1-5), p_j is the relative frequency of answer j, μ_X is the mean of X, and $d_X = X_{max} - X_{min}$ is the width of X.

[5] Raw results can be found at: https://bit.ly/2SQ0o6m

The value of the consensus measure ranges between 0 (total disagreement) and 1 (total agreement). Authors in [Landis and Koch, 1977] proposed the following interpretation for intermediate values:

a) Less than 0: poor agreement.

b) 0.01–0.20: slight agreement.

c) 0.21–0.40: fair agreement.

d) 0.41–0.60: moderate agreement.

e) 0.61–0.80: substantial agreement.

f) 0.81–1.00: almost perfect agreement.

We adapted the definition of p_j in the consensus (Cns) formula to account for the level of expertise of each participant:

$$p_j = \frac{\sum_{i=1}^{N} w_i \cdot \delta_{vote(i),j}}{\sum_{i=1}^{N} w_i}$$

where $\delta_{vote(i),j} = 1$ if participant i voted j, and 0 otherwise. Table 4.2 presents the weighted average and the consensus value for the different questions of the survey, computed using R.[6]

Out of Table 4.2, we can derive the following outcomes (O):

Outcome 1. The vast majority of the participants have an experience with ontology engineering, ontology versioning and ontology networks (Figure 4.1). Moreover, the mean of the level of confidence of the participants of the survey is around 65%.

Outcome 2. The participants substantially agreed with the new categorization we proposed for ontology pitfalls (i.e., stand-alone ontology pitfalls, versioned ontologies pitfalls, and pitfalls inside ontology networks). The percentage of agreement is around 74.29%.

Outcome 3. For *P1. Ontology is not accessible at its IRI*, there is a substantial agreement that this pitfall is problematic in ontology engineering (i.e., the weighted average for the answers is 4.06, with a consensus ratio of 67.85%). In addition, the participants substantially agreed that it has a major impact on subsequent versions (i.e., the weighted average for the answers is 4.14, with a consensus ratio of 77.00%). Finally, the participants substantially agreed that it has also a major impact on ontology networks (i.e., the weighted average for the answers is 4.20, with a consensus ratio of 71.60%).

[6]The source code found in *resources/SurveyExperiments* at https://github.com/OmarAlqawasmeh/coEvolutionTermsExtraction

TABLE 4.2: Weighted average and consensus ratio for the survey's answers

	Weighted average (/5) VS Consensus value (/100)					
	How problematic is it?		Impact on versioned ontologies		Impact on ontology networks	
Pitfall	Weighted Avg.	Consensus	Weighted Avg.	Consensus	Weighted Avg.	Consensus
P1	4.06	67.85	4.14	77.00	4.20	71.60
P2	3.50	66.10	3.60	59.03	3.62	63.56
P3	3.85	62.30	3.62	58.04	4.10	66.65
P4	3.65	65.18	3.77	59.10	3.55	59.00
P5	3.59	65.19	3.57	58.80	3.53	63.16
P6	2.60	58.35	2.64	64.64	2.54	60.66
P7	3.51	59.68	3.48	62.49	3.55	59.29
P8	3.53	55.05	3.33	65.29	3.33	62.90
P9	3.06	63.04	2.98	55.60	2.53	58.14
Agreement on the classification (/100)				74.29		
Level of confidence (/100)				64.71		

Outcome 4. For *P2. Importing an ontology using a non persistent IRI or the IRI of a representation*, there is a substantial agreement that this pitfall is problematic in ontology engineering (i.e., the weighted average for the answers is 3.50, with a consensus ratio of 66.10%). In addition, the participants moderately agreed that it has a major impact on subsequent versions (i.e., the weighted average for the answers is 3.60, with a consensus ratio of 59.03%). Finally, the participants substantially agreed that it has also a major impact on ontology networks (i.e., the weighted average for the answers is 3.62, with a consensus ratio of 63.56%).

Outcome 5. For *P3. Importing an inconsistent ontology*, there is a substantial agreement that this pitfall is problematic in ontology engineering (i.e., the weighted average for the answers is 3.85, with a consensus ratio of 62.30%). In addition, the participants moderately agreed that it has a major impact on subsequent versions (i.e., the weighted average for the answers is 3.62, with a consensus ratio of 58.04%). Finally, the participants substantially agreed that it is problematic and has major impact on ontology networks (i.e., the weighted average for the answers is 4.10, with a consensus ratio of 66.65%).

Outcome 6. For *P4. Only the latest version of the ontology is available online*, there is a substantial agreement that this pitfall is problematic in ontology engineering (i.e., the weighted average for the answers is 3.65, with a consensus ratio of 65.18%). In addition, the participants moderately agreed that it has a major impact on subsequent versions (i.e., the weighted average for the answers is 3.77, with a consensus ratio of 59.10%). Finally, the participants moderately agreed that it has a major impact on ontology networks (i.e., the weighted average for the answers is 3.60, with a consensus ratio of 55.16%).

Outcome 7. For *P5. Importing an ontology series IRI instead of an ontology version IRI*, there is a substantial agreement that this pitfall is problematic in ontology engineering (i.e., the weighted average for the answers is 3.59, with a consensus ratio of 65.19%). In addition, the participants moderately agreed that it has a major impact on subsequent versions (i.e., the weighted average for the answers is 3.57, with a consensus ratio of 58.80%). Finally, the participants moderately agreed that it has also a major impact on ontology networks (i.e., the weighted average for the answers is 3.53, with a consensus ratio of 63.16%).

Outcome 8. For *P6. Ontology series IRI is the same as the ontology version IRI*, there is a moderate agreement that this pitfall is neutral and does not cause problems in ontology engineering (i.e., the weighted average for the answers is 2.60, with a consensus ratio of 58.35%). In addition, the participants substantially agreed that it has less impact on subsequent versions (i.e., the weighted average for the answers is 2.64, with a consensus ratio of 64.64%). Finally, the participants substantially agreed that

it has also less impact on ontology networks (i.e., the weighted average for the answers is 2.54, with a consensus ratio of 60.66%).

Outcome 9. For *P7. A term is moved from one ontology module to another with different IRI*, there is a moderate agreement that this pitfall causes problems in ontology engineering (i.e., the weighted average for the answers is 3.51, with a consensus ratio of 59.68%). In addition, the participants moderately agreed that it has a middle impact on subsequent versions (i.e., the weighted average for the answers is 3.48, with a consensus ratio of 62.49%). Finally, the participants moderately agreed that it has also a major impact on ontology networks (i.e., the weighted average for the answers is 3.55, with a consensus ratio of 59.29%).

Outcome 10. For *P8. Namespace hijacking*, there is a moderate agreement that this pitfall causes problems in ontology engineering (i.e., the weighted average for the answers is 3.53, with a consensus ratio of 55.05%). In addition, the participants substantially agreed that it has middle impact on subsequent versions (i.e., the weighted average for the answers is 3.33, with a consensus ratio of 65.29%). Finally, the participants substantially agreed that it has middle impact on ontology networks (i.e., the weighted average for the answers is 3.33, with a consensus ratio of 62.90%).

Outcome 11. For *P9. The IRI of a term contains a file extension*, there is a substantial agreement that this pitfall is neutral and does not cause problems in ontology engineering (i.e., the weighted average for the answers is 3.06, with a consensus ratio of 63.04%). In addition, the participants substantially agreed that it has less impact on subsequent versions (i.e., the weighted average for the answers is 2.98, with a consensus ratio of 55.60%). Finally, the participants moderately agreed that it has also almost no impact on ontology networks (i.e., the weighted average for the answers is 2.53, with a consensus ratio of 58.14%).

Out of these outcomes, Table 4.3 categorizes the pitfalls based on their estimated impact into:

1. Major impact (WA $>$ 3.5).
2. Middle impact (3 $<$ WA $<$ 3.5).
3. Less impact (WA $<$ 3).

We rank the pitfalls' impact in descending order (i.e., high to less). As shown in Table 4.3, there is a substantial agreement that *P1* and *P4* have a major impact on versioned ontologies, and *P1* and *P3* have a major impact on ontology networks. Pitfalls *P1, P2, P3, P4, and P5* have a major impact on both versioned ontologies and ontology networks. *P7* has a middle impact on versioned ontology but a major impact on ontology networks. *P8* has a middle impact on both versioned ontologies and ontology networks. As for *P6 and P9* the participants substantially agree that they have less impact.

Table 4.3: Pitfalls ranked by their impact over versioned and networked ontologies

Impact on	Major	Middle	Less
Versioned ontologies	P1. Ontology is not accessible at its IRI^{+} P4. Only the latest version of the ontology is available online^{+} P3. Importing an inconsistent ontology* P5. Importing an ontology series IRI instead of an ontology version IRI* P2. Importing an ontology using a non persistent IRI or the IRI of a representation*	P7. Term is moved from one ontology module to another with different IRI* P8. Namespace hijacking^{+}	P6. Ontology series IRI is the same as the ontology version IRI^{+} P9. The IRI of a term contains a file extension^{+}
Ontology networks	P1. Ontology is not accessible at its IRI^{+} P3. Importing an inconsistent ontology^{+} P7. Term is moved from one ontology module to another with different IRI* P4. Only the latest version of the ontology is available online* P2. Importing an ontology using a non persistent IRI or the IRI of a representation* P5. Importing an ontology series IRI instead of an ontology version IRI*	P8. Namespace hijacking*	P6. Ontology series IRI is the same as the ontology version IRI^{+} P9. The IRI of a term contains a file extension^{+}

+: substantial agreement
*: moderate agreement

Moreover, Figure 4.2 presents how often the participants encountered the different pitfalls. We can see that except for P6, P7, and P8, all participants encountered the different pitfalls before.

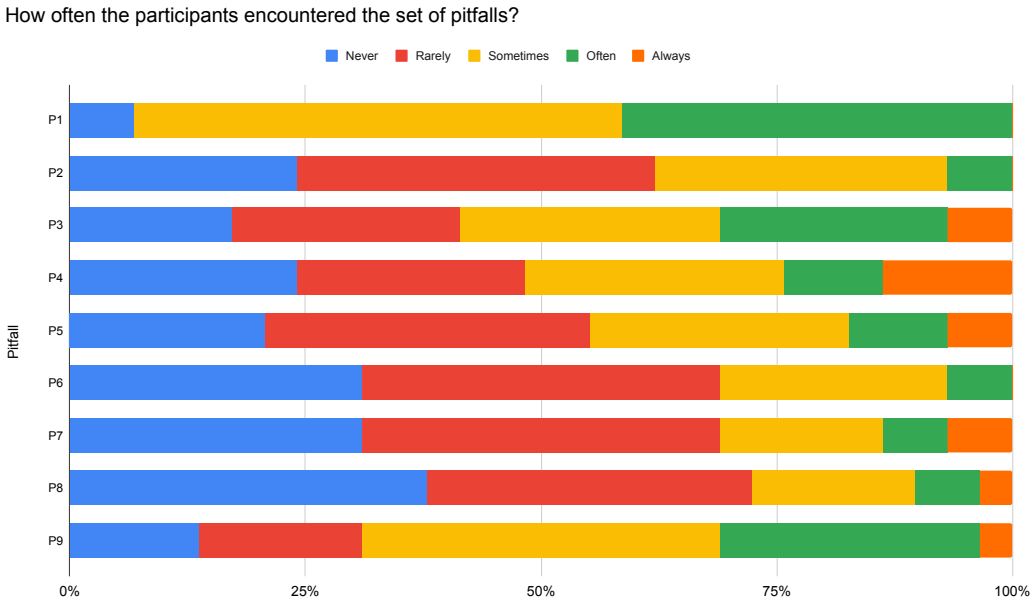


FIGURE 4.2: How often the participants encountered the candidate pitfalls

4.4.3 Analyzing the survey's participants opinions

For each pitfall, participants could share known occurrences of the pitfall, and ideas or recommendations to solve or avoid it. We summarize the gathered opinions (OPN) below.

OPN 1. *Persistent IRIs are important:* participants agreed about the importance of persistent IRIs when creating or reusing ontologies. Some suggested to use services or catalogues to ensure the usage of persistent IRIs. Using persistent IRIs can effectively help to avoid pitfalls P1 and P2.

OPN 2. *Consistency tests should be made on the imported ontologies:* ontology editors should applying consistency tests on the imported ontologies to avoid pitfall P3.

OPN 3. *When reusing terms, refer only to those that are needed:* Some of the participants suggested to avoid importing the whole ontology and only declare the required terms. Ontology editors should check that these terms are correctly declared (e.g., a term that is originally declared a datatype property should not be declared as an annotation property), and services should be developed to monitor the evolution of the ontologies to prevent pitfall P7.

OPN 4. *Import the ontology using its version IRI.* This point has both advantages and disadvantages. On the one hand, importing an ontology version IRI prevents any issue that may arise if the imported ontology evolves. But on the other hand, it may be interesting to update an ontology when a new version of an imported ontology is issued. Again, services could be developed to notify ontology editors about any new version release of the imported ontologies.

OPN 5. *A notification message should be send in case of moving terms from one module to another.* A subscription mechanism could be used to notify the different external artifacts (e.g. systems, ontologies) when an ontology evolves.

OPN 6. *Focusing only on the ontology level is not sufficient.* A participant argued that focusing on the quality of ontologies is less important than focusing on the quality of their usage. The following questions have been raised:

- How to improve the integration of heterogeneous data that was designed independently of the ontologies?
- What can go wrong when the data and the ontology become misaligned?
- How to deal with noisy knowledge situations where the logic embedded in the ontology becomes unusable?

These different points can be topics for future work.

4.5 Conclusion

Theretofore, in Chapter 1 we presented the domain description and the state of the art study based on the life-cycle of ontology evolution proposed by [Zablith et al., 2015]. We presented the current limitations in the state of the art research work.

In Chapter 2, we proposed two contributions:

1. We proposed a definition that helps to detect the need of evolution by observing the evolution of imported ontologies (Section 2.1). This part investigated our first research hypothesis: *RH 1 An ontology may need to evolve after some changes in some ontologies it uses*, and it answered our first research question: *RQ 1 How to detect the need of evolving an ontology through the observation of structural changes in the ontologies it uses?*
2. We proposed a functionality that takes advantage of existing knowledge bases to evolve ontologies (Section 2.2). This part investigated our second research hypothesis: *RH 2 Using existing knowledge sources may help to develop and evolve ontologies.* and it answered our second research question: *RQ 2 How to take advantage of external knowledge bases to develop and evolve ontologies?*

In Chapter 3, we investigated our third research hypothesis: *RH 3 Ontology portals may contain traces of incoherences in the evolution of ontologies that use one another*, and we answered our third research question: *RQ 3 How to detect and assess incoherences in the evolution of ontologies that use terms of one in another?*

We showed that there is a need to formalize a conceptual frame for assessing the impact of ontology evolution and for tackling the different issues that arise during the evolution of ontologies. Hence, we presented a situation of ontology evolution, which we called ontology co-evolution. Ontology co-evolution considers the evolution of an ontology O that imports another one O' (i.e., O uses terms that have the namespace of O'). We provided an exhaustive categorization of the adaptation to ontology evolution for this situation. We observed these cases over two ontology portals: 1. the Linked Open Vocabulary (LOV) ontology portal (Section 3.3.1), and 2. the BioPortal (Section 3.3.2).

In this chapter, we investigated our last research hypothesis: *RH 4 Identifying pitfalls that affect ontology networks and versioned ontologies may help to design better ontologies*, and we answered our two last research questions: *RQ 4 What pitfalls affect ontology network?*, and *RQ 5 What pitfalls affect versioned ontologies?*

In Section 1.2, we concluded that the state of the art studies on ontology pitfalls list only stand-alone ontologies pitfalls. In this chapter we identified ontology pitfalls that target ontology networks, i.e., when an ontology O uses or imports another ontology O' (Section 4.1), and/or versioned ontologies, i.e., when an ontology O_1 evolves to O_2 (Section 4.2). Therefore, in Section 4.3 we proposed 9 additional candidate pitfalls for ontology engineering.

In order to validate these pitfalls and to measure their importance and potential impact, we distributed a survey to the semantic web community (Section 4.4). Participants agreed that listing and investigating in ontology pitfalls can effectively enhance the quality of ontologies which reflects in a positive way in using these ontologies for the different tasks (e.g. question answering). In addition, we suggested a set of best practices to be followed in order to prevent or solve the candidate pitfalls.

Lastly, we would like to stress the following issues and potential future research tracks that are derived from this chapter. These points will be further discussed in the next chapter:

1. From our study, we show that there is a need to initiate the design of an ontology management framework. This framework will help to provide automatic analysis and notification services to avoid pitfalls in ontology networks and versioned ontologies.
2. Some existing ontology development tools lead to the creation of certain pitfalls.
3. The inheritance of a pitfall inside ontology networks or in versioned ontologies.
4. It would be interesting to investigate further on the set of pitfalls that might occur on the data level.

Part III

Conclusion and Future work

Chapter 5

General Conclusion and Perspectives

Summary of the contributions

In this thesis, we investigated the following two research goals:

RG.1 *To study the evolution need and evolution implementation of ontologies* (Chapter 2).

RG.2 *To study how the evolution and the quality of an ontology impacts the ontologies that use it* (Chapter 3 and Chapter 4).

In Chapter 2 we investigated the first research goal, i.e., *RG 1. To study the evolution need and evolution implementation of ontologies.* We introduced two research hypotheses that are associated with two research questions:

RH 1. An ontology may need to evolve after some changes in some ontologies it uses.

In Section 2.1 we answered the first research question (*RQ 1. How to detect the need of evolving an ontology through the observation of structural changes in the ontologies it uses?*), where we introduced a definition of a situation that could be used to detect the need for evolving an ontology O based on the evolution of a used ontology O' (i.e., when O uses terms that have the namespace of O').

RH 2. Using existing knowledge sources may help to develop and evolve ontologies.

In Section 2.2 we answered the second research question (*RQ 2. How to take advantage of external knowledge bases to develop and evolve ontologies?*), where we proposed an original approach for ontology enrichment based on the use of three external knowledge bases: DBpedia, WikiData, and NELL. Our results show that our system performs better than [Kong et al., 2006] that is based on WordNet. This allows us to assert that our proposal would be a good fit for the enrichment (evolution) phase of ontology development. Moreover, it could even be used as a first step to bootstrap ontologies from blank pages to avoid the cold-start problem.

In Chapter 3 and Chapter 4 we investigated the second research goal, i.e., *RG 2. To study how the evolution and the quality of an ontology impact the ontologies that use it.* We introduced two research hypotheses that are associated with three research questions:

RH 3. Ontology portals may contain traces of incoherences in the evolution of ontologies that use one another.

In Chapter 3 we answered the third research question (*RQ 3. How to detect and assess incoherences in the evolution of ontologies that use terms of one in another?*), where we present a situation of ontology evolution which considers the evolution of an ontology O that uses another one O' (i.e., O uses terms that have the namespace of O'). We provided an exhaustive categorization of the adaptation to ontology evolution for this situation. We observed these cases over two ontology portals: 1. The Linked Open Vocabulary (LOV) ontology portal which references 648 different ontologies, 88 of them evolved. We identified 74 cases of ontology co-evolution, involving 28 different ontologies (Section 3.3.1), and 2. The BioPortal which references 770 different ontologies, 485 of them evolved. We identified 14 cases of ontology co-evolution, involving 10 different ontologies (Section 3.3.2).

RH 4. Identifying pitfalls that affect ontology networks and versioned ontologies may help to design better ontologies.

In Chapter 4 we answered both the fourth and fifth research questions (*RQ 4. What pitfalls affect ontology networks?* and *RQ 5. What pitfalls affect versioned ontologies?*). We identified 9 candidate pitfalls that may affect versioned ontologies, i.e., when an ontology O_1 evolves to O_2, and/or ontology networks, i.e., when an ontology O uses or imports another ontology O'. In order to measure the importance and potential impact of the candidate pitfalls, we distributed a survey to the semantic web community. The 29 participants agreed that listing and investigating in ontology pitfalls can effectively enhance the quality of ontologies which reflects in a positive way in using these ontologies for the different tasks (e.g. question answering). Moreover, we suggested a set of best practices to be followed in order to prevent or solve the candidate pitfalls.

Table 5.1 presents a summary of the different contributions of the thesis.

TABLE 5.1: Our main contributions based on the current state of the art work

Chapter	Contribution area	Comparison with state of the art
Chapter 2	Detect the need of ontology evolution	**Limitations:** From the literature two techniques are used: detect the need of evolution from data, and/or detect the need of evolution from usage. However, current approaches do not keep track of the reused ontologies to evolve ontologies. **Our contributions:** Our proposed approach follows and combines both of the two techniques. The definition we introduced can be used to detect the evolution either by observing the internal data, i.e., evolution of terms, or by detecting the behavior that is caused by this evolution.
	Suggest changes to enrich ontologies	**Limitations:** Current approaches rely on the quality of the documents that are used to generate ontologies. These documents might be language strict, and unstructured. **Our contributions:** After detecting the need of evolution, ontologies should be changed accordingly. We introduced an algorithm that can be used in two directions: 1. Initiate ontologies by taking advantage of current large-scale knowledge bases. This may help to avoid cold start (i.e., blank page) problem. 2. Enrich ontologies by taking advantage of current large-scale knowledge bases. This is done based on keyword search engine. The keywords are extracted from the ontologies that need to evolve. Then a text query is done to suggest the most relevant classes, properties and instances to be added to the ontology.
Chapter 3	Assessing the impact of ontology evolution	**Limitations:** There is no formal definition that targets assessing the impact of ontology evolution. From literature three techniques are used: 1. observing the structural changes (e.g. addition and deletion), 2. measuring the impact of the evolution over external artifacts (e.g. search systems), and 3. provide some statistics, such as: listing the changes and the frequency of each change. **Our contributions:** 1. We introduced the term "Ontology co-evolution" which considers the evolution of an ontology O that uses another one O' (i.e., O uses terms that have the namespace of O'). 2. We provided and analyzed an exhaustive categorization of the adaptation to ontology evolution for this situation.
Chapter 4	Pitfalls in networked and versioned ontologies	**Limitations:** Current studies and tools target pitfalls only in stand-alone ontologies. **Our contributions:** 1. We introduced a new categorization of ontology pitfalls: stand-alone ontology pitfalls, pitfalls in versioned ontologies, and pitfalls in ontology networks. 2. We identified 9 candidate pitfalls that may affect versioned ontologies or ontology networks. 3. We evaluated the importance and potential impact of the candidate pitfalls by means of a web-based survey we conducted in the semantic web community. 4. We provided a set of recommendations to avoid or solve the different pitfalls we identified.

Perspectives

This section presents some topics that can be of interest for further investigation:

Firstly, to help enhancing the ability to bootstrap and evolve ontologies, several paths could be followed:

- Support the collaborative functionalities between the different parties, i.e., knowledge engineers, domain experts, and computer systems.
- Implement a web application service that supports the bootstrapping and enrichment processes for ontologies.

Secondly, as the usage of ontologies is increasing, there is the need of managing them, especially in the evolution process. The main aim of this research is to introduce fundamentals for a methodological framework for ontology management during ontology evolution. Having such kind of framework would effectively help to automate the process of managing ontologies during their evolution cycle which could lead to save time and effort. We emphasize the need of having a service that can automatically observe and notify the ontologies' owners during the evolution process. Having such tool could help to keep track of the different ontologies during the co-evolution and help to facilitate the process of ontology evolution.

Thirdly, regarding the pitfall analysis study, the following issues we identified can be of interest for further studies:

- *Some ontology development tools lead to the creation of pitfalls*: There exists some tools that generate pitfalls. For example OnToology tool[1] publishes only the latest version of an ontology, and the documentation of this latest version (even if the ontology includes provenance information and information about the previous versions). It is important to update this tool so that all the versions are published. In case the owners of O' uses OnToology to publish it. Any other ontology that uses O' will be forced to import O' using its ontology series IRI or the latest ontology version IRI. Then using another ontology version IRI will have the risk to break this import in the future.
- *The inheritance of a pitfall*: Some pitfalls inside ontologies can be inherited either when the ontologies evolve or when the ontology is used by other ontologies. For example, if an ontology O' has the pitfall "creating the relationship (is) instead of using `rdfs:subClassOf`, `rdf:type` or `owl:sameAs`"[2], it means that O' has a property called *is*. If another ontology O uses O', then this pitfall will propagate to O automatically.
- *Pitfalls on the data level*: to investigate further on the set of pitfalls that might occur on the data level (suggested by some of the survey's participants). Mainly, to focus on the set of pitfalls that occur between the data and the ontologies

[1] http://ontoology.linkeddata.es/

[2] Pifall number 3 from http://oops.linkeddata.es/catalogue.jsp

such as misalignment between the ontology and the data due to evolution of the ontology.

Fourthly, being part of the ETSI STF 578 group [3], our main aim is to develop a multi-tests pipeline that is used to check the development guidelines for the Smart Applications REFerence Ontology, and extensions (SAREF) as specified by ETSI Technical Specification TS 103 673 v1.1.1. "SAREF Development Framework and Workflow, Streamlining the Development of SAREF and its Extensions".

These tests are used to facilitate the development process of SAREF ontologies and to enhance the quality of the ontologies. Some examples of such tests are: 1. checking the metadata for the developed ontologies using SHACL shapes, 2. checking the existence of the external terms that are used inside the ontologies, and 3. checking the consistency of the developed ontologies. Some of our proposed contributions in this thesis are highly relevant and recommended to be used at similar tests pipelines.

Fifthly, as a direct industrial perspective, the contributions of this thesis can be used to create a tool suite that can be fundamental for a startup enterprise. A market-plan study has been already conducted, and a market need was detected. This study took part along with the guidance of business experts at the University of Lyon in a special program to help young researchers to mature their research work to help creating a startup. The general idea is to create and manipulate knowledge graphs from unstructured or structured data. The main phases for such enterprise are: 1. moving from unstructured (raw text documents) to knowledge graphs (using natural language processing and information extraction techniques), 2. moving from structured data to knowledge graphs (modulation and conversion), 3. re-usability of existing knowledge graphs, 4. managing the evolution of the resulted knowledge graphs (i.e., manage the co-evolution, avoiding the defined pitfalls in this thesis), and 5. exploiting the knowledge graphs and take the most advantage of them (e.g. prepare to use the resulted knowledge graphs in question answering systems).

Finally, we would like to stress the need for having a framework that can automatically observe and notify the ontologies' owners during the evolution process. This could positively help them maintaining their ontologies during the evolution's life-cycle. Also, it can help to keep track of the ontologies in the different varieties we proposed (i.e., ontology co-evolution, ontology versioning, and ontology networking) Which will reflect positively on the quality of the different ontologies and will help knowledge engineers in their tasks.

[3] https://portal.etsi.org/STF/STFs/STF-HomePages/STF578

Appendix A

Survey on pitfalls in versioned and networked ontologies

In order to assess the potential impact and importance of the candidate pitfalls we identified, we distributed the following survey to the semantic web community.

Pitfalls in versioned ontologies and ontology networks

Ontology pitfalls are situations that are the result of bad practices in the development, evolution, or/and publication, of ontologies.

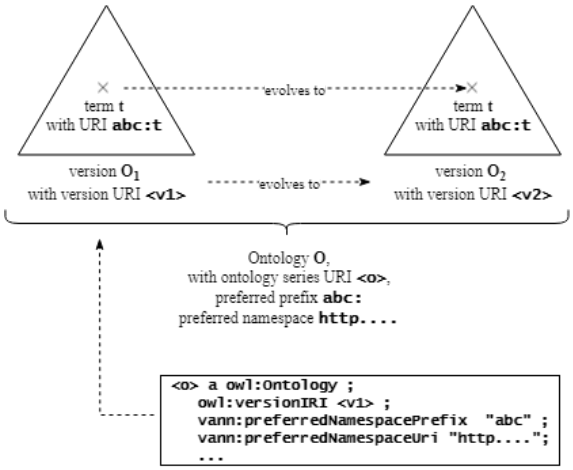
Research has been led to list and classify pitfalls for single ontologies. However, situations that may not be considered as problematic for one ontology may become pitfalls when this ontology is versioned, or used by other ontologies.

This survey aims at validating and classifying a list of candidate pitfalls for versioned ontologies and ontology networks.

The approximative time to fill this survey is 15 minutes

* Required

Illustration of an ontology series with two versions



1. **How familiar are you with ontology engineering?** *
Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Beginner	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Expert

2. **How familiar are you with ontology versioning?** *
Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Beginner	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Expert

3. **How familiar are you with ontology networks?** *
Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Beginner	◯	◯	◯	◯	◯	◯	◯	◯	◯	◯	Expert

Pitfall 1. Ontology is not accessible at its IRI

An ontology is described in a research paper and is not online.
This ontology cannot be reused.

The IRI of the Pizza ontology is http://www.co-ode.org/ontologies/pizza/pizza.owl# , but it is not accessible at this IRI.
Any import of the Pizza ontology using this IRI will fail.

The IRI of the DOLCE Ultralite upper ontology was originally http://www.loa-cnr.it/ontologies/DUL.owl# .
The website loa-cnr.it closed, and the ontology is now available at http://www.ontologydesignpatterns.org/ont/dul/DUL.owl# .
Any import of this ontology using the old IRI will fail.

4. **How often have you encountered this pitfall before?** *
Mark only one oval.

- ◯ Never
- ◯ Rarely
- ◯ Sometimes
- ◯ Often
- ◯ Always

5. **How problematic is this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

6. **How would you rate the impact on subsequent versions of the ontology?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

7. **How problematic is it to import ontologies that have this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

8. **How would you solve this pitfall? (optional)**

9. **If you know any other occurrences, please list some below (optional)**

Pitfall 2. Importing an ontology using a non-persistent IRI or the IRI of representation (the file URL)

The SEAS ontology has persistent IRI https://w3id.org/seas/ , which redirected to the location https://ci.emse.fr/seas/
Assume an ontology imported IRI https://ci.emse.fr/seas/ instead of https://www.w3.org/ns/org .
Due to the renaming of the EMSE institution, the IRI now redirects to the location https://ci.mines-stetienne.fr/seas/
Any such import broke.

The W3C organization ontology has persistent IRI https://www.w3.org/ns/org, and there are two representations of this ontology with IRI https://www.w3.org/ns/org.rdf and https://www.w3.org/ns/org.ttl .
Assume an ontology imports the ontology representation https://www.w3.org/ns/org.rdf instead of the ontology series https://www.w3.org/ns/org .
Assume someday the RDF/XML representation is deleted, and replaced by a new Turtle 1.1 representation.
Then the import will break.

10. **How often have you encountered this pitfall before?** *

Mark only one oval.

- ◯ Never
- ◯ Rarely
- ◯ Sometimes
- ◯ Often
- ◯ Always

11. **How problematic is this pitfall?** *

Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

12. **How would you rate the impact on subsequent versions of the ontology?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

13. **How problematic is it to import ontologies that have this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

14. **How would you solve this pitfall? (optional)**

15. **If you know any other occurrences, please list some below (optional)**

Pitfall 3. Importing an inconsistent ontology

The SAREF4ENER ontology (EEbus/Energy@home) https://w3id.org/saref4ee is inconsistent. Any ontology that imports this ontology will become inconsistent.

16. **How often have you encountered this pitfall before?** *
Mark only one oval.

- ◯ Never
- ◯ Rarely
- ◯ Sometimes
- ◯ Often
- ◯ Always

17. **How problematic is this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

18. **How would you rate the impact on subsequent versions of the ontology?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

19. **How problematic is it to import ontologies that have this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

20. **How would you solve this pitfall? (optional)**

21. **If you know any other occurrences, please list some below (optional)**

Pitfall 4. Only the latest version of the ontology is available online

The S4WATR ontology is published at https://w3id.org/def/S4WATR , but only the latest version is available online.
Assume an ontology imports the S4WATR ontology at a certain point in time.
Assume the S4WATR ontology evolves and deletes some terms or add some axioms.
Then the importing ontology may break or become inconsistent.

22. **How often have you encountered this pitfall before?** *
Mark only one oval.

- ◯ Never
- ◯ Rarely
- ◯ Sometimes
- ◯ Often
- ◯ Always

23. **How problematic is this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

24. **How would you rate the impact on subsequent versions of the ontology?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

25. **How problematic is it to import ontologies that have this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

26. **How would you solve this pitfall? (optional)**

27. **If you know any other occurrences, please list some below (optional)**

Pitfall 5. Importing an ontology series IRI instead of an ontology version IRI

The SAREF ontology series has IRI https://saref.etsi.org/saref# , and version 2.1.1 has IRI https://saref.etsi.org/saref/v2.1.1/saref# .
A new version 3.1.1 is under development and will delete terms from version 2.1.1 .
Assume an ontology imports the SAREF ontology 2.1.1 using https://saref.etsi.org/saref#
When the new version 3.1.1 will be released, the importing ontology may break or become inconsistent.

28. **How often have you encountered this pitfall before?** *
Mark only one oval.

- ◯ Never
- ◯ Rarely
- ◯ Sometimes
- ◯ Often
- ◯ Always

29. **How problematic is this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

30. **How would you rate the impact on subsequent versions of the ontology?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

31. **How problematic is it to import ontologies that have this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

32. **How would you solve this pitfall? (optional)**

33. **If you know any other occurrences, please list some below (optional)**

Pitfall 6. Ontology series IRI is the same as the ontology version IRI

The Units of Measure (OM) ontology version 1.8 has the IRI http://www.wurvoc.org/vocabularies/om-1.8/,
and version 2.0 has the IRI http://www.ontology-of-units-of-measure.org/resource/om-2/.

According to the OWL 2 specification (section 3.1 and 3.3), there are two different ontology series each having one single version.

34. **How often have you encountered this pitfall before?** *
Mark only one oval.

- ◯ Never
- ◯ Rarely
- ◯ Sometimes
- ◯ Often
- ◯ Always

35. **How problematic is this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

36. **How would you rate the impact on subsequent versions of the ontology?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

37. **How problematic is it to import ontologies that have this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

38. **How would you solve this pitfall? (optional)**

39. **If you know any other occurrences, please list some below (optional)**

Pitfall 7. A term is moved from one ontology module to another with different IRI

The SAREF ontologies consist of: 1) SAREF-core, 2) SAREF4SYST and several ontologies for verticals (e.g. SAREF4ENER, SAREF4BLDG, and SAREF4ENVI).

In SAREF-core 1.1.1 (created 2015) owners defined saref:BuildingObject. Later in 2016, SAREF-core 2.1.1 was published without saref:BuildingObject.

However, another ontology SAREF4BLDG was created with the term s4bldg:BuildingObject (with the same definition as saref:BuildingObject).

In this case, the IRI of the term BuildingObject has been changed. This might lead to functional impact over the artifacts that are reusing the term (e.g. some queries might be affected by the change of the IRI).

40. **How often have you encountered this pitfall before?** *
Mark only one oval.

- ◯ Never
- ◯ Rarely
- ◯ Sometimes
- ◯ Often
- ◯ Always

41. **How problematic is this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

42. **How would you rate the impact on subsequent versions of the ontology?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

43. **How problematic is it to import ontologies that have this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

44. **How would you solve this pitfall? (optional)**

45. **If you know any other occurrences, please list some below (optional)**

Pitfall 8. Namespace hijacking

The description of classes qudt-1.1:QuantityValue and qudt-1.1:Quantity is not available at their own IRIs. Instead, they are defined in the ontology http://qudt.org/1.1/schema/quantity# .

Namespace hijacking refers to reusing or referring to terms from another namespace that are not defined in such namespace. This pitfall can affect ontology networks as it might cause not retrieving valid information while looking for the hijacked terms (which violates LD publishing guidelines).

46. **How often have you encountered this pitfall before?** *

Mark only one oval.

- Never
- Rarely
- Sometimes
- Often
- Always

47. **How problematic is this pitfall?** *

Mark only one oval.

	1	2	3	4	5	
Low						High

48. **How would you rate the impact on subsequent versions of the ontology?** *

Mark only one oval.

	1	2	3	4	5	
Low						High

49. **How problematic is it to import ontologies that have this pitfall?** *

Mark only one oval.

	1	2	3	4	5	
Low						High

50. **How would you solve this pitfall? (optional)**

51. **If you know any other occurrences, please list some below (optional)**

Pitfall 9. The IRI of a term contains a file extension

The terms in the Dolce ultra lite ontology have namespace
http://www.ontologydesignpatterns.org/ont/dul/DUL.owl# .

Assume some day the publisher of dolce-very-lite wants to set up content negotiation to expose an html documentation of their ontology.
As the IRI of the terms contain the file extension ".owl", no content negotiation can take place.
If a human looks up the term IRI, he will access the OWL file, and not the html documentation.

52. **How often have you encountered this pitfall before?** *
Mark only one oval.
- Never
- Rarely
- Sometimes
- Often
- Always

53. **How problematic is this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low						High

54. **How would you rate the impact on subsequent versions of the ontology?** *
Mark only one oval.

	1	2	3	4	5	
Low						High

55. **How problematic is it to import ontologies that have this pitfall?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

56. **How would you solve this pitfall? (optional)**

57. **If you know any other occurrences, please list some below (optional)**

Pitfalls categorization

We propose to distinguish between three types of pitfalls:

1) Stand-alone ontology pitfalls: happen within a single ontology.

2) Versioned ontology pitfalls: happen when a new version of the ontology is created.

3) Ontology network pitfalls: happen when an ontology uses terms that have a namespace of a different ontology, or when an ontology imports a different ontology.

58. **How much do you agree with this classification?** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

59. **Your level of confidence while filling the survey:** *
Mark only one oval.

	1	2	3	4	5	
Low	◯	◯	◯	◯	◯	High

60. **Further comments and suggestions:**

Powered by
Google Forms

Appendix B

From the preface to Ph.D. thesis

PhD candidate at University of Lyon. Researcher at Connected Intelligence group at Laboratoire Hubert Curien. Field of the research: Semantic Web, Natural Language Processing, and Data Mining.

B.1 Grants and awards

Activities and awards during my Ph.D. studies:

1. Prix ASEC du concours posters de la journée de la recherche (Best poster award given by the ASEC association at the poster session of the day of research in University of Lyon) 14 Jun 2018.
2. Full grant to attend the International Semantic Web Research School (ISWS 2018) 1-7 Jul 2018 (Selection based on candidates' profiles, estimated amount 950 euros).

B.2 Scientific activities

During my PhD, I presented scientific articles or demos in the following events (some are directly related to my thesis, and the other are collaborative projects with external teams)

- Statlearn 2017 workshop in Lyon 5-7 APR 2017.
- The Web Intelligence summer school (WISS) in Saint-Etienne, France 3-7 JUL 2017.
- Invited speaker at the ontology group at IRSTEA-centre de Clermont-Ferrand. Presentation title: Collaborative ontology development: focus on bootstrapping capabilities.
- Extended Semantic Web Conference (ESWC 2018) in Heraklion, Greece 3-7 JUN 2018.

- Poster presentation at the International Semantic Web Research School (ISWS 2018).
- The Fifth International Conference on Social Networks Analysis Management and Security (SNAMS-2018) in Valencia, Spain 15-18 OCT 2018.
- 21st International Conference on Knowledge Engineering and Knowledge Management (EKAW-2018) in Nancy, France 12-16 NOV 2018.
- LDK 2019 – 2nd Conference on Language, Data and Knowledge in Leipzig, Germany 20-23 May 2019.
- KEOD 2019 – the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management 17-10 SEP 2019.

B.3 List of publications

Here are a complete list of my scientific publications from 2015- present (June-2020):

1. **Omar Qawasmeh**: A Collaborative Framework for Ontology and Instance Data Co-evolution and Extraction. EKAW (Doctoral Consortium) 2018 [Qawasmeh, 2018].

2. **Omar Qawasmeh**, Maxime Lefrançois, Antoine Zimmermann, Pierre Maret: Computer-Assisted Ontology Construction System: Focus on Bootstrapping Capabilities. The Semantic Web - ESWC 2018 Satellite Events - ESWC 2018 Satellite Event. **Acceptance rate: 26.8%.** [Qawasmeh et al., 2018].

3. **Omar Qawasmeh**, Maxime Lefrançois, Antoine Zimmermann, Pierre Maret: Observing the Impact and Adaptation to the Evolution of an Imported Ontology. Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (KEOD 2019). **Nominated for best paper award. Acceptance rate: 22%** [Qawasmeh et al., 2019].

4. **Omar Qawasmeh**, Maxime Lefrançois, Antoine Zimmermann, Pierre Maret: Pitfalls in Networked and Versioned Ontologies. Journal: Communications in Computer and Information Science. Springer 2020. [Under review].

The following publications were derived out of my MSc thesis:

5. Hybrid Approach for Event Extraction from Arabic Tweets. **Omar Qawasmeh**. **Master Thesis.**

6. Mohammad AL-Smadi, and **Omar Qawasmeh**. "A Supervised Machine Learning Approach for Events Extraction out of Arabic Tweets." In 2018 Fifth International Conference on Social Networks Analysis, Management and Security (SNAMS), pp. 114-119. IEEE, 2018.
7. Mohammad AL-Smadi, and **Omar Qawasmeh**. "Knowledge-based approach for event extraction from arabic tweets." International Journal of Advanced Computer Science and Applications 1, no. 7 (2016): 483-490.
8. **Omar Qawasmeh**, Mohammad Al-Smadi, and Nisreen Fraihat. "Arabic named entity disambiguation using linked open data." In 2016 7th International Conference on Information and Communication Systems (ICICS), pp. 333-338. IEEE, 2016.
9. Mohammad AL-Smadi, Bashar Talafha, **Omar Qawasmeh**, Mohammed N. Alandoli, Wegdan A. Hussien, and Christian Guetl. "A hybrid approach for Arabic named entity disambiguation." In Proceedings of the 15th International Conference on Knowledge Technologies and Data-driven Business, pp. 1-4. 2015.

The following publications were the results of external collaborations:

10. Ilkcan Keles, **Omar Qawasmeh**, Tabea Tietz, Ludovica Marinucci, Roberto Reda, and Marieke Van Erp. "A Proposal for a Two-Way Journey on Validating Locations in Unstructured and Structured Data." In 2nd Conference on Language, Data and Knowledge (LDK 2019). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.
11. Dennis Diefenbach, Pedro Henrique Migliatti, **Omar Qawasmeh**, Vincent Lully, Kamal Singh, and Pierre Maret. "QAnswer: A Question Answering prototype bridging the gap between a considerable part of the LOD cloud and end-users." In The World Wide Web Conference, pp. 3507-3510. 2019.
12. Linked Open Data Validity – A Technical Report from ISWS 2018.
13. Mohammad AL-Smadi, Mahmoud Al-Ayyoub, Yaser Jararweh, and **Omar Qawasmeh**. "Enhancing aspect-based sentiment analysis of Arabic hotels' reviews using morphological, syntactic and semantic features." Information Processing and Management 56, no. 2 (2019): 308-319.
14. Mohammad AL-Smadi, **Omar Qawasmeh**, Mahmoud Al-Ayyoub, Yaser Jararweh, and Brij Gupta. "Deep Recurrent neural network vs. support vector machine for aspect-based sentiment analysis of Arabic hotels' reviews." Journal of computational science 27 (2018): 386-393.
15. Mohammad AL-Smadi, **Omar Qawasmeh**, Bashar Talafha, Mahmoud Al-Ayyoub, Yaser Jararweh, and Elhadj Benkhelifa. "An enhanced framework for aspect-based sentiment analysis of Hotels' reviews: Arabic reviews case study." In 2016 11th International Conference for Internet Technology and Secured Transactions (ICITST), pp. 98-103. IEEE, 2016.

16. Mohammad AL-Smadi, **Omar Qawasmeh**, Bashar Talafha, and Muhannad Quwaider. "Human annotated arabic dataset of book reviews for aspect based sentiment analysis." In 2015 3rd International Conference on Future Internet of Things and Cloud, pp. 726-730. IEEE, 2015.

Appendix C

The co-evolution cases of the Linked Open Vocabulary and BioPortal

C.1 The set of co-evolution cases from LOV

v_1	v_2	v_1'	v_2'	namespace
dcterms_2008-01-14.n3	dcterms_2010-10-11.n3	dcam_2008-01-14.n3	dcam_2010-10-11.n3	http://purl.org/dc/dcam/
dcterms_2010-10-11.n3	dcterms_2012-06-14.n3	dcam_2010-10-11.n3	dcam_2012-06-14.n3	http://purl.org/dc/dcam/
foaf_2010-08-09.n3	foaf_2014-01-14.n3	dce_2008-01-14.n3	dce_2012-06-14.n3	http://purl.org/dc/elements/1.1/
dce_2008-01-14.n3	dce_2010-10-11.n3	dcam_2008-01-14.n3	dcam_2010-10-11.n3	http://purl.org/dc/dcam/
dce_2008-01-14.n3	dce_2010-10-11.n3	dcterms_2008-01-14.n3	dcterms_2010-10-11.n3	http://purl.org/dc/terms/
dce_2010-10-11.n3	dce_2012-06-14.n3	dcterms_2010-10-11.n3	dcterms_2012-06-14.n3	http://purl.org/dc/terms/
dce_2010-10-11.n3	dce_2012-06-14.n3	dcam_2010-10-11.n3	dcam_2012-06-14.n3	http://purl.org/dc/dcam/
qb_2010-11-27.n3	qb_2013-03-02.n3	dcterms_2010-10-11.n3	dcterms_2012-06-14.n3	http://purl.org/dc/terms/
qb_2013-07-26.n3	qb_2014-07-31.n3	foaf_2010-08-09.n3	foaf_2014-01-14.n3	http://xmlns.com/foaf/0.1/
osspr_2010-04-01.n3	osspr_2013-09-04.n3	dcterms_2008-01-14.n3	dcterms_2012-06-14.n3	http://purl.org/dc/terms/
osspr_2010-04-01.n3	osspr_2013-09-04.n3	dce_2008-01-14.n3	dce_2012-06-14.n3	http://purl.org/dc/elements/1.1/
scovo_2011-08-05.n3	scovo_2012-08-09.n3	dcterms_2010-10-11.n3	dcterms_2012-06-14.n3	http://purl.org/dc/terms/
scovo_2011-08-05.n3	scovo_2012-08-09.n3	dce_2010-10-11.n3	dce_2012-06-14.n3	http://purl.org/dc/elements/1.1/
prv_2009-11-28.n3	prv_2010-04-04.n3	foaf_2007-10-02.n3	foaf_2010-01-01.n3	http://xmlns.com/foaf/0.1/
prv_2010-07-10.n3	prv_2011-01-25.n3	foaf_2010-01-01.n3	foaf_2010-08-09.n3	http://xmlns.com/foaf/0.1/
prv_2010-07-10.n3	prv_2011-01-25.n3	dcterms_2008-01-14.n3	dcterms_2010-10-11.n3	http://purl.org/dc/terms/
gn_2010-10-05.n3	gn_2012-02-14.n3	dcterms_2008-01-14.n3	dcterms_2010-10-11.n3	http://purl.org/dc/terms/
gn_2012-02-14.n3	gn_2012-10-29.n3	dcterms_2010-10-11.n3	dcterms_2012-06-14.n3	http://purl.org/dc/terms/
dctype_2008-01-14.n3	dctype_2010-10-11.n3	dcterms_2008-01-14.n3	dcterms_2010-10-11.n3	http://purl.org/dc/terms/
dctype_2010-10-11.n3	dctype_2012-06-14.n3	dcterms_2010-10-11.n3	dcterms_2012-06-14.n3	http://purl.org/dc/terms/
dctype_2008-01-14.n3	dctype_2010-10-11.n3	dcam_2008-01-14.n3	dcam_2010-10-11.n3	http://purl.org/dc/dcam/
dctype_2010-10-11.n3	dctype_2012-06-14.n3	dcam_2010-10-11.n3	dcam_2012-06-14.n3	http://purl.org/dc/dcam/
vcard_2010-01-20.n3	vcard_2013-05-25.n3	dce_2008-01-14.n3	dce_2012-06-14.n3	http://purl.org/dc/elements/1.1/
schema_2012-04-27.n3	schema_2012-06-26.n3	dcterms_2010-10-11.n3	dcterms_2012-06-14.n3	http://purl.org/dc/terms/

edm__2010-07-30.n3	edm__2012-01-23.n3	dctype__2008-01-14.n3	dctype__2010-10-11.n3	http://purl.org/dc/dcmitype/
edm__2010-07-30.n3	edm__2012-01-23.n3	dcterms__2008-01-14.n3	dcterms__2010-10-11.n3	http://purl.org/dc/terms/
edm__2010-07-30.n3	edm__2012-01-23.n3	foaf__2010-01-01.n3	foaf__2010-08-09.n3	http://xmlns.com/foaf/0.1/
edm__2010-07-30.n3	edm__2012-01-23.n3	dce__2008-01-14.n3	dce__2010-10-11.n3	http://purl.org/dc/elements/1.1/
edm__2012-01-23.n3	edm__2013-05-20.n3	dctype__2010-10-11.n3	dctype__2012-06-14.n3	http://purl.org/dc/dcmitype/
edm__2012-01-23.n3	edm__2013-05-20.n3	dcterms__2010-10-11.n3	dcterms__2012-06-14.n3	http://purl.org/dc/terms/
edm__2012-01-23.n3	edm__2013-05-20.n3	dce__2010-10-11.n3	dce__2012-06-14.n3	http://purl.org/dc/elements/1.1/
mo__2010-11-28.n3	mo__2013-07-22.n3	bio__2010-04-20.n3	bio__2011-06-14.n3	http://purl.org/vocab/bio/0.1/
mo__2010-11-28.n3	mo__2013-07-22.n3	dcterms__2010-10-11.n3	dcterms__2012-06-14.n3	http://purl.org/dc/terms/
mo__2010-11-28.n3	mo__2013-07-22.n3	dce__2010-10-11.n3	dce__2012-06-14.n3	http://purl.org/dc/elements/1.1/
org__2010-06-06.n3	org__2010-10-08.n3	foaf__2010-01-01.n3	foaf__2010-08-09.n3	http://xmlns.com/foaf/0.1/
org__2010-10-08.n3	org__2012-09-30.n3	dcterms__2008-01-14.n3	dcterms__2012-06-14.n3	http://purl.org/dc/terms/
org__2013-02-15.n3	org__2013-12-16.n3	prov__2012-07-24.n3	prov__2013-04-30.n3	http://www.w3.org/ns/prov#
org__2014-01-02.n3	org__2014-01-25.n3	foaf__2010-08-09.n3	foaf__2014-01-14.n3	http://xmlns.com/foaf/0.1/
bio__2009-05-19.n3	bio__2010-04-20.n3	foaf__2007-10-02.n3	foaf__2010-01-01.n3	http://xmlns.com/foaf/0.1/
bio__2010-04-20.n3	bio__2011-06-14.n3	dcterms__2008-01-14.n3	dcterms__2010-10-11.n3	http://purl.org/dc/terms/
bio__2010-04-20.n3	bio__2011-06-14.n3	foaf__2010-01-01.n3	foaf__2010-08-09.n3	http://xmlns.com/foaf/0.1/
dcam__2008-01-14.n3	dcam__2010-10-11.n3	dcterms__2008-01-14.n3	dcterms__2010-10-11.n3	http://purl.org/dc/terms/
dcam__2010-10-11.n3	dcam__2012-06-14.n3	dcterms__2010-10-11.n3	dcterms__2012-06-14.n3	http://purl.org/dc/terms/
rdag1__2012-04-09.n3	rdag1__2012-08-30.n3	dcterms__2010-10-11.n3	dcterms__2012-06-14.n3	http://purl.org/dc/terms/
rdag1__2012-04-09.n3	rdag1__2012-08-30.n3	dce__2010-10-11.n3	dce__2012-06-14.n3	http://purl.org/dc/elements/1.1/
adms__2012-06-25.n3	adms__2013-05-24.n3	schema__2012-04-27.n3	schema__2013-04-05.n3	http://schema.org/
adms__2013-05-24.n3	adms__2013-09-16.n3	schema__2013-04-05.n3	schema__2013-08-07.n3	http://schema.org/
adms__2013-09-16.n3	adms__2013-12-21.n3	schema__2013-08-07.n3	schema__2013-12-04.n3	http://schema.org/
adms__2013-12-21.n3	adms__2015-07-22.n3	schema__2013-12-04.n3	schema__2015-05-12.n3	http://schema.org/
adms__2013-12-21.n3	adms__2015-07-22.n3	foaf__2010-08-09.n3	foaf__2014-01-14.n3	http://xmlns.com/foaf/0.1/
mads__2012-05-10.n3	mads__2015-10-28.n3	foaf__2010-08-09.n3	foaf__2014-01-14.n3	http://xmlns.com/foaf/0.1/
cito__2010-03-26.n3	cito__2011-05-05.n3	dce__2008-01-14.n3	dce__2010-10-11.n3	http://purl.org/dc/elements/1.1/
cito__2011-12-09.n3	cito__2012-07-03.n3	dce__2010-10-11.n3	dce__2012-06-14.n3	http://purl.org/dc/elements/1.1/
txn__2012-05-25.n3	txn__2012-07-05.n3	dcterms__2010-10-11.n3	dcterms__2012-06-14.n3	http://purl.org/dc/terms/
txn__2012-05-25.n3	txn__2012-07-05.n3	dce__2010-10-11.n3	dce__2012-06-14.n3	http://purl.org/dc/elements/1.1/
voaf__2011-11-16.n3	voaf__2012-07-03.n3	dcterms__2010-10-11.n3	dcterms__2012-06-14.n3	http://purl.org/dc/terms/
wlo__2010-02-19.n3	wlo__2013-12-18.n3	dctype__2008-01-14.n3	dctype__2012-06-14.n3	http://purl.org/dc/dcmitype/
wlo__2010-02-19.n3	wlo__2013-12-18.n3	dcterms__2008-01-14.n3	dcterms__2012-06-14.n3	http://purl.org/dc/terms/
wlo__2010-02-19.n3	wlo__2013-12-18.n3	foaf__2010-01-01.n3	foaf__2010-08-09.n3	http://xmlns.com/foaf/0.1/
rov__2013-01-08.n3	rov__2013-12-21.n3	schema__2012-11-08.n3	schema__2013-12-04.n3	http://schema.org/
rov__2013-01-08.n3	rov__2013-12-21.n3	org__2012-10-06.n3	org__2013-12-16.n3	http://www.w3.org/ns/org#
lingvo__2013-03-20.n3	lingvo__2013-04-18.n3	schema__2012-11-08.n3	schema__2013-04-05.n3	http://schema.org/
lingvo__2013-04-18.n3	lingvo__2013-12-19.n3	schema__2013-04-05.n3	schema__2013-12-04.n3	http://schema.org/
lingvo__2013-12-20.n3	lingvo__2014-01-17.n3	foaf__2010-08-09.n3	foaf__2014-01-14.n3	http://xmlns.com/foaf/0.1/
lingvo__2014-01-20.n3	lingvo__2014-03-20.n3	schema__2013-12-04.n3	schema__2014-02-05.n3	http://schema.org/
lingvo__2014-03-20.n3	lingvo__2014-08-11.n3	schema__2014-02-05.n3	schema__2014-07-28.n3	http://schema.org/

locn__2013-11-25.n3	locn__2013-12-21.n3	schema__2013-11-19.n3	schema__2013-12-04.n3	http://schema.org/
locn__2013-12-21.n3	locn__2015-03-23.n3	schema__2013-12-04.n3	schema__2015-02-04.n3	http://schema.org/
locn__2013-12-21.n3	locn__2015-03-23.n3	foaf__2010-08-09.n3	foaf__2014-01-14.n3	http://xmlns.com/foaf/0.1/
frbrer__2012-02-29.n3	frbrer__2015-07-14.n3	foaf__2010-08-09.n3	foaf__2014-01-14.n3	http://xmlns.com/foaf/0.1/
frbrer__2012-02-29.n3	frbrer__2015-07-14.n3	dce__2010-10-11.n3	dce__2012-06-14.n3	http://purl.org/dc/elements/1.1/
osadm__2010-04-01.n3	osadm__2013-09-04.n3	osspr__2010-04-01.n3	osspr__2013-09-04.n3	http://data.ordnancesurvey.co.uk/ontology/spatialrelations/
osadm__2010-04-01.n3	osadm__2013-09-04.n3	dcterms__2008-01-14.n3	dcterms__2012-06-14.n3	http://purl.org/dc/terms/
osadm__2010-04-01.n3	osadm__2013-09-04.n3	dce__2008-01-14.n3	dce__2012-06-14.n3	http://purl.org/dc/elements/1.1/

C.2 The set of co-evolution cases from BioPortal

v_1	v_2	v_1'	v_2'	namespace
SIO__2015-06-24.owl	SIO__2015-09-02.owl	cito__2010-03-26.n3	cito__2015-07-03.n3	http://purl.org/spar/cito/
SIO__2016-06-28.owl	SIO__2016-08-16.owl	schema__2016-05-04.n3	schema__2016-08-09.n3	http://schema.org/
OBOREL__2018-01-05.owl	OBOREL__2018-03-13.owl	cito__2015-07-03.n3	cito__2018-02-16.n3	http://purl.org/spar/cito/
SCHEMA__2014-10-30.owl	SCHEMA__2017-05-19.owl	schema__2012-04-27.n3	schema__2017-03-23.n3	http://schema.org/
SCHEMA__2017-05-19.owl	SCHEMA__2018-10-11.owl	schema__2017-03-23.n3	schema__2018-06-14.n3	http://schema.org/
CLO__2018-02-07.owl	CLO__2018-10-15.owl	cito__2015-07-03.n3	cito__2018-02-16.n3	http://purl.org/spar/cito/
GFVO__2014-12-12.owl	GFVO__2015-05-26.owl	sio__2013-12-04.n3	sio__2015-04-21.n3	http://semanticscience.org/resource/
GFVO__2015-10-13.owl	GFVO__2016-10-14.owl	sio__2015-04-21.n3	sio__2016-09-27.n3	http://semanticscience.org/resource/
PLANA__2017-09-27.owl	PLANA__2018-07-14.owl	cito__2015-07-03.n3	cito__2018-02-16.n3	http://purl.org/spar/cito/
OBI_BCGO__2014-12-10.owl	OBI_BCGO__2015-04-08.owl	sio__2013-12-04.n3	sio__2015-01-11.n3	http://semanticscience.org/resource/
DCO__2015-01-09.owl	DCO__2015-02-25.owl	schema__2012-04-27.n3	schema__2015-02-04.n3	http://schema.org/
ONS__2017-12-15.owl	ONS__2018-03-12.owl	cito__2015-07-03.n3	cito__2018-02-16.n3	http://purl.org/spar/cito/
SIO__2017-02-12.owl	SIO__2017-10-02.owl	schema__2016-08-09.n3	schema__2017-08-14.n3	http://schema.org/
SIO__2018-02-15.owl	SIO__2018-04-12.owl	cito__2015-07-03.n3	cito__2018-02-16.n3	http://purl.org/spar/cito/

Bibliography

[Abdel-Qader et al., 2018] Abdel-Qader, M., Scherp, A., and Vagliano, I. (2018). Analyzing the evolution of vocabulary terms and their impact on the LOD cloud. In Gangemi, A., Navigli, R., Vidal, M., Hitzler, P., Troncy, R., Hollink, L., Tordai, A., and Alam, M., editors, *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 1–16. Springer.

[Abgaz et al., 2012] Abgaz, Y. M., Javed, M., and Pahl, C. (2012). Analyzing impacts of change operations in evolving ontologies. In [Groza et al., 2012].

[Agirre et al., 2000] Agirre, E., Ansa, O., Hovy, E. H., and Martínez, D. (2000). Enriching very large ontologies using the WWW. In Staab, S., Maedche, A., Nedellec, C., and Wiemer-Hastings, P. M., editors, *ECAI'2000 Workshop on Ontology Learning, Proceedings of the First Workshop on Ontology Learning OL'2000, Berlin, Germany, August 25, 2000. Held in conjunction with the 14th European Conference on Artificial Intelligence ECAI'2000, Berlin, Germany*, volume 31 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Alani et al., 2006] Alani, H., Brewster, C., and Shadbolt, N. (2006). Ranking ontologies with aktiverank. In [Cruz et al., 2006], pages 1–15.

[Auer et al., 2007] Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. G. (2007). Dbpedia: A nucleus for a web of open data. In Aberer, K., Choi, K., Noy, N. F., Allemang, D., Lee, K., Nixon, L. J. B., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., and Cudré-Mauroux, P., editors, *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735. Springer.

[Balakrishna and Moldovan, 2013] Balakrishna, M. and Moldovan, D. I. (2013). Automatic building of semantically rich domain models from unstructured data. In Boonthum-Denecke, C. and Youngblood, G. M., editors, *Proceedings of the Twenty-Sixth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2013, St. Pete Beach, Florida. May 22-24, 2013.* AAAI Press.

[Balakrishna and Srikanth, 2008] Balakrishna, M. and Srikanth, M. (2008). Automatic ontology creation from text for national intelligence priorities framework

(NIPF). In Laskey, K. B. and Wijesekera, D., editors, *Towards Effective Exploitation and Integration of Intelligence Resources, Proceedings of the Third International Ontology for the Intelligence Community Conference, OIC 2008, Fairfax, VA, USA, December 3-4, 2008.*, volume 440 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Beckett, 2014] Beckett, D. (2014). Rdf 1.1 n-triples. *URL: https://www. w3. org/TR/n-triples*.

[Beckett et al., 2014] Beckett, D., Berners-Lee, T., Prud'hommeaux, E., and Carothers, G. (2014). Rdf 1.1 turtle. *World Wide Web Consortium*.

[Beckett and McBride, 2004] Beckett, D. and McBride, B. (2004). Rdf/xml syntax specification (revised). *W3C recommendation*, 10(2.3).

[Bedini and Nguyen, 2007] Bedini, I. and Nguyen, B. (2007). Automatic ontology generation: State of the art. *PRiSM Laboratory Technical Report. University of Versailles*.

[Bernaras et al., 1996] Bernaras, A., Laresgoiti, I., and Corera, J. M. (1996). Building and reusing ontologies for electrical network applications. In Wahlster, W., editor, *12th European Conference on Artificial Intelligence, Budapest, Hungary, August 11-16, 1996, Proceedings*, pages 298–302. John Wiley and Sons, Chichester.

[Berrueta et al., 2008] Berrueta, D., Phipps, J., Miles, A., Baker, T., and Swick, R. (2008). Best practice recipes for publishing rdf vocabularies. *Working draft, W3C*.

[Białecki et al., 2012] Białecki, A., Muir, R., Ingersoll, G., and Imagination, L. (2012). Apache lucene 4. In *SIGIR 2012 workshop on open source information retrieval*, page 17.

[Bloehdorn et al., 2006] Bloehdorn, S., Haase, P., Sure, Y., and Voelker, J. (2006). Ontology evolution. *Semantic Web Technologies: Trends and Research in Ontology-based Systems*, pages 51–70.

[Brickley et al., 2014] Brickley, D., Guha, R. V., and McBride, B. (2014). Rdf schema 1.1. *W3C recommendation*, 25:2004–2014.

[Brickley and Miller, 2010] Brickley, D. and Miller, L. (2010). Foaf vocabulary specification 0.91.

[Burton-Jones et al., 2005] Burton-Jones, A., Storey, V. C., Sugumaran, V., and Ahluwalia, P. (2005). A semiotic metrics suite for assessing the quality of ontologies. *Data Knowl. Eng.*, 55(1):84–102.

[Cahyani and Wasito, 2017] Cahyani, D. E. and Wasito, I. (2017). Automatic ontology construction using text corpora and ontology design patterns (odps) in alzheimer's disease. *Jurnal Ilmu Komputer dan Informasi*, 10(2):59–66.

[Carlson et al., 2010] Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Jr., E. R. H., and Mitchell, T. M. (2010). Toward an architecture for never-ending language learning. In Fox, M. and Poole, D., editors, *Proceedings of the Twenty-Fourth*

AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010. AAAI Press.

[Castano et al., 2006] Castano, S., Ferrara, A., and Hess, G. N. (2006). Discovery-driven ontology evolution. In Tummarello, G., Bouquet, P., and Signore, O., editors, *SWAP 2006 - Semantic Web Applications and Perspectives, Proceedings of the 3rd Italian Semantic Web Workshop, Scuola Normale Superiore, Pisa, Italy, 18-20 December, 2006*, volume 201 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Cimiano and Völker, 2005] Cimiano, P. and Völker, J. (2005). Text2onto. In Montoyo, A., Muñoz, R., and Métais, E., editors, *Natural Language Processing and Information Systems, 10th International Conference on Applications of Natural Language to Information Systems, NLDB 2005, Alicante, Spain, June 15-17, 2005, Proceedings*, volume 3513 of *Lecture Notes in Computer Science*, pages 227–238. Springer.

[Confort et al., 2015] Confort, V. T. F., Revoredo, K., Baião, F. A., and Santoro, F. M. (2015). Learning ontology from text: A storytelling exploratory case study. In Uden, L., Hericko, M., and Ting, I., editors, *Knowledge Management in Organizations - 10th International Conference, KMO 2015, Maribor, Slovenia, August 24-28, 2015, Proceedings*, volume 224 of *Lecture Notes in Business Information Processing*, pages 477–491. Springer.

[Cristani and Cuel, 2005] Cristani, M. and Cuel, R. (2005). A survey on ontology creation methodologies. *Int. J. Semantic Web Inf. Syst.*, 1(2):49–69.

[Cruz et al., 2006] Cruz, I. F., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., and Aroyo, L., editors (2006). *The Semantic Web - ISWC 2006, 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006, Proceedings*, volume 4273 of *Lecture Notes in Computer Science*. Springer.

[Dahab et al., 2008] Dahab, M. Y., Hassan, H. A., and Rafea, A. A. (2008). Textontoex: Automatic ontology construction from natural english text. *Expert Syst. Appl.*, 34(2):1474–1480.

[Doran et al., 2007] Doran, P., Tamma, V. A. M., and Iannone, L. (2007). Ontology module extraction for ontology reuse: an ontology engineering perspective. In Silva, M. J., Laender, A. H. F., Baeza-Yates, R. A., McGuinness, D. L., Olstad, B., Olsen, Ø. H., and Falcão, A. O., editors, *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, CIKM 2007, Lisbon, Portugal, November 6-10, 2007*, pages 61–70. ACM.

[Dragoni and Ghidini, 2012] Dragoni, M. and Ghidini, C. (2012). Evaluating the impact of ontology evolution patterns on the effectiveness of resources retrieval. In [Groza et al., 2012].

[Dragoni et al., 2017] Dragoni, M., Poveda-Villalón, M., and Jiménez-Ruiz, E., editors (2017). *OWL: - Experiences and Directions - Reasoner Evaluation - 13th International Workshop, OWLED 2016, and 5th International Workshop, ORE*

2016, Bologna, Italy, November 20, 2016, Revised Selected Papers, volume 10161 of *Lecture Notes in Computer Science*. Springer.

[ETSI, 2019a] ETSI, T. (2019a). 103 608 v1. 1.1 (2019-07). *Published, SmartM2M*.

[ETSI, 2019b] ETSI, T. (2019b). 103 673 v1.1.1 (2020-04). *Published, SmartM2M*.

[Fernández-López et al., 1997] Fernández-López, M., Gómez-Pérez, A., and Juristo, N. (1997). Methontology: from ontological art towards ontological engineering. *AAAI Technical Report*.

[Gaudet and Dessimoz, 2017] Gaudet, P. and Dessimoz, C. (2017). Gene ontology: pitfalls, biases, and remedies. In *The Gene Ontology Handbook*, pages 189–205. Humana Press, New York, NY.

[Groß et al., 2012] Groß, A., Hartung, M., Prüfer, K., Kelso, J., and Rahm, E. (2012). Impact of ontology evolution on functional analyses. *Bioinformatics*, 28(20):2671–2677.

[Groza et al., 2012] Groza, T., Plexousakis, D., and Novácek, V., editors (2012). *Proceedings of the 2nd Joint Workshop on Knowledge Evolution and Ontology Dynamics, Boston, MA, USA, November 12, 2012*, volume 890 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Gruber, 1993] Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2):199–220.

[Guha et al., 2016] Guha, R. V., Brickley, D., and Macbeth, S. (2016). Schema. org: evolution of structured data on the web. *Communications of the ACM*, 59(2):44–51.

[Gyrard et al., 2015] Gyrard, A., Serrano, M., and Atemezing, G. A. (2015). Semantic web methodologies, best practices and ontology engineering applied to internet of things. In *2nd IEEE World Forum on Internet of Things, WF-IoT 2015, Milan, Italy, December 14-16, 2015*, pages 412–417. IEEE Computer Society.

[Haase et al., 2006] Haase, P., Rudolph, S., Wang, Y., Brockmans, S., Palma, R., Euzenat, J., and d'Aquin, M. (2006). D1. 1.1 networked ontology model. *NeOn Deliverable*, 1(1).

[Harris et al., 2013] Harris, S., Seaborne, A., and Prud'hommeaux, E. (2013). Sparql 1.1 query language. *W3C recommendation*, 21(10).

[Hartung et al., 2012] Hartung, M., Groß, A., and Rahm, E. (2012). CODEX: exploration of semantic changes between ontology versions. *Bioinformatics*, 28(6):895–896.

[Hartung et al., 2013] Hartung, M., Groß, A., and Rahm, E. (2013). Conto-diff: generation of complex evolution mappings for life science ontologies. *Journal of Biomedical Informatics*, 46(1):15–32.

[Hazber et al., 2016] Hazber, M. A., Li, R., Gu, X., and Xu, G. (2016). Integration mapping rules: Transforming relational database to semantic web ontology. *Appl. Math*, 10(3):1–21.

[Heath and Bizer, 2011] Heath, T. and Bizer, C. (2011). Linked data: Evolving the web into a global data space. *Synthesis lectures on the semantic web: theory and technology*, 1(1):1–136.

[Huang et al., 2016] Huang, J., Lee, K., Choi, K., and Kim, Y. K. (2016). Extract reliable relations from wikipedia texts for practical ontology construction. *Computación y Sistemas*, 20(3):467–476.

[Iqbal et al., 2013] Iqbal, R., Murad, M. A. A., Mustapha, A., and Sharef, N. M. (2013). An analysis of ontology engineering methodologies: A literature review. *Research journal of applied sciences, engineering and technology*, 6(16):2993–3000.

[Janowicz et al., 2014] Janowicz, K., Hitzler, P., Adams, B., Kolas, D., and Vardeman, C. (2014). Five stars of linked data vocabulary use. *Semantic Web*, 5(3):173–176.

[Javed et al., 2011] Javed, M., Abgaz, Y. M., and Pahl, C. (2011). Graph-based discovery of ontology change patterns. In *In Proceedings of the Joint Workshop on Knowledge Evolution and Ontology Dynamic*. CEUR.

[Jones et al., 1998] Jones, D., Bench-Capon, T., and Visser, P. (1998). Methodologies for ontology development. *na*.

[Jonquet, 2018] Jonquet, C. (2018). Fair data requires fair ontologies, how do we do? *F1000Research*, 7.

[Jonquet et al., 2018a] Jonquet, C., Toulet, A., Arnaud, E., Aubin, S., Kaboré, E. D. Y., Emonet, V., Graybeal, J., Laporte, M., Musen, M. A., Pesce, V., and Larmande, P. (2018a). Agroportal: A vocabulary and ontology repository for agronomy. *Computers and Electronics in Agriculture*, 144:126–143.

[Jonquet et al., 2018b] Jonquet, C., Toulet, A., Dutta, B., and Emonet, V. (2018b). Harnessing the power of unified metadata in an ontology repository: the case of agroportal. *Journal on Data Semantics*, 7(4):191–221.

[Kietz et al., 2000] Kietz, J.-U., Maedche, A., and Volz, R. (2000). A method for semi-automatic ontology acquisition from a corporate intranet. In *EKAW-2000 Workshop "Ontologies and Text", Juan-Les-Pins, France, October 2000*.

[Kirsten et al., 2009] Kirsten, T., Hartung, M., Groß, A., and Rahm, E. (2009). Efficient management of biomedical ontology versions. In Meersman, R., Herrero, P., and Dillon, T. S., editors, *On the Move to Meaningful Internet Systems: OTM 2009 Workshops, Confederated International Workshops and Posters, ADI, CAMS, EI2N, ISDE, IWSSA, MONET, OnToContent, ODIS, ORM, OTM Academy, SWWS, SEMELS, Beyond SAWSDL, and COMBEK 2009, Vilamoura, Portugal, November 1-6, 2009. Proceedings*, volume 5872 of *Lecture Notes in Computer Science*, pages 574–583. Springer.

[Klein and Fensel, 2001] Klein, M. C. A. and Fensel, D. (2001). Ontology versioning on the semantic web. In Cruz, I. F., Decker, S., Euzenat, J., and McGuinness, D. L.,

editors, *Proceedings of SWWS'01, The first Semantic Web Working Symposium, Stanford University, California, USA, July 30 - August 1, 2001*, pages 75–91.

[Kong et al., 2006] Kong, H., Hwang, M., and Kim, P. (2006). Design of the automatic ontology building system about the specific domain knowledge. In *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, volume 2, pages 4–pp. IEEE.

[Konstantinidis et al., 2008] Konstantinidis, G., Flouris, G., Antoniou, G., and Christophides, V. (2008). A formal approach for RDF/S ontology evolution. In Ghallab, M., Spyropoulos, C. D., Fakotakis, N., and Avouris, N. M., editors, *ECAI 2008 - 18th European Conference on Artificial Intelligence, Patras, Greece, July 21-25, 2008, Proceedings*, volume 178 of *Frontiers in Artificial Intelligence and Applications*, pages 70–74. IOS Press.

[Kumar et al., 2016] Kumar, N., Kumar, M., and Singh, M. (2016). Automated ontology generation from a plain text using statistical and nlp techniques. *International Journal of System Assurance Engineering and Management*, 7(1):282–293.

[Kupfer and Eckstein, 2006] Kupfer, A. and Eckstein, S. (2006). Coevolution of database schemas and associated ontologies in biological context. In *22nd British National Conference on Databases*.

[Kupfer et al., 2006] Kupfer, A., Eckstein, S., Neumann, K., and Mathiak, B. (2006). A coevolution approach for database schemas and related ontologies. In *19th IEEE International Symposium on Computer-Based Medical Systems (CBMS 2006), Salt Lake City, Utah, USA*, pages 605–610. IEEE Computer Society.

[Landis and Koch, 1977] Landis, J. R. and Koch, G. G. (1977). The measurement of observer agreement for categorical data. *biometrics*, pages 159–174.

[Lebo et al., 2013] Lebo, T., Sahoo, S., McGuinness, D., Belhajjame, K., Cheney, J., Corsar, D., Garijo, D., Soiland-Reyes, S., Zednik, S., and Zhao, J. (2013). Prov-o: The prov ontology. *W3C recommendation*, 30.

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

[Lossio-Ventura et al., 2016] Lossio-Ventura, J. A., Roche, M., Jonquet, C., and Teisseire, M. (2016). A way to automatically enrich biomedical ontologies. In Pitoura, E., Maabout, S., Koutrika, G., Marian, A., Tanca, L., Manolescu, I., and Stefanidis, K., editors, *Proceedings of the 19th International Conference on Extending Database Technology, EDBT 2016, Bordeaux, France, March 15-16, 2016, Bordeaux, France, March 15-16, 2016*, pages 676–677. OpenProceedings.org.

[Maedche and Staab, 2002] Maedche, A. and Staab, S. (2002). Measuring similarity between ontologies. In Gómez-Pérez, A. and Benjamins, V. R., editors, *Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1-4, 2002,*

Proceedings, volume 2473 of *Lecture Notes in Computer Science*, pages 251–263. Springer.

[Manola et al., 2004] Manola, F., Miller, E., McBride, B., et al. (2004). Rdf primer. *W3C recommendation*, 10(1-107):6.

[Maynard et al., 2009] Maynard, D., Funk, A., and Peters, W. (2009). Sprat: a tool for automatic semantic pattern-based ontology population. In *International conference for digital libraries and the semantic web, Trento, Italy.*

[McCord et al., 2012] McCord, M. C., Murdock, J. W., and Boguraev, B. (2012). Deep parsing in watson. *IBM Journal of Research and Development*, 56(3):3.

[Mihindukulasooriya et al., 2016] Mihindukulasooriya, N., Poveda-Villalón, M., García-Castro, R., and Gómez-Pérez, A. (2016). Collaborative ontology evolution and data quality - an empirical analysis. In [Dragoni et al., 2017], pages 95–114.

[Miller, 1995] Miller, G. A. (1995). Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

[Moldovan et al., 2007] Moldovan, D., Srikanth, M., and Badulescu, A. (2007). Synergist: Topic and user knowledge bases from textual sources for collaborative intelligence analysis. In *CASE PI Conference.*

[Moldovan and Girju, 2000] Moldovan, D. I. and Girju, R. (2000). Domain-specific knowledge acquisition and classification using wordnet. In Etheredge, J. N. and Manaris, B. Z., editors, *Proceedings of the Thirteenth International Florida Artificial Intelligence Research Society Conference, May 22-24, 2000, Orlando, Florida, USA*, pages 224–228. AAAI Press.

[Motik et al., 2009] Motik, B., Patel-Schneider, P. F., Parsia, B., Bock, C., Fokoue, A., Haase, P., Hoekstra, R., Horrocks, I., Ruttenberg, A., Sattler, U., et al. (2009). Owl 2 web ontology language: Structural specification and functional-style syntax. *W3C recommendation*, 27(65):159.

[Mukherjee et al., 2014] Mukherjee, S., Ajmera, J., and Joshi, S. (2014). Domain cartridge: Unsupervised framework for shallow domain ontology construction from corpus. In Li, J., Wang, X. S., Garofalakis, M. N., Soboroff, I., Suel, T., and Wang, M., editors, *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3-7, 2014*, pages 929–938. ACM.

[Navigli and Ponzetto, 2010] Navigli, R. and Ponzetto, S. P. (2010). Babelnet: Building a very large multilingual semantic network. In Hajic, J., Carberry, S., and Clark, S., editors, *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*, pages 216–225. The Association for Computer Linguistics.

[Nicola and Missikoff, 2016] Nicola, A. D. and Missikoff, M. (2016). A lightweight methodology for rapid ontology engineering. *Commun. ACM*, 59(3):79–86.

[Novacek and Handschuh, 2007] Novacek, V. and Handschuh, S. (2007). Semi-automatic integration of learned ontologies into a collaborative framework. *Proceedings of International Workshop on Ontology Dynamics (IWOD-07)*.

[Noy et al., 2006] Noy, N. F., Chugh, A., Liu, W., and Musen, M. A. (2006). A framework for ontology evolution in collaborative environments. In [Cruz et al., 2006], pages 544–558.

[Noy et al., 2003] Noy, N. F., Crubézy, M., Fergerson, R. W., Knublauch, H., Tu, S. W., Vendetti, J., Musen, M. A., et al. (2003). Protege-2000: an open-source ontology-development and knowledge-acquisition environment. In *AMIA Annu Symp Proc*, volume 953, page 953.

[Noy et al., 2001] Noy, N. F., McGuinness, D. L., et al. (2001). Ontology development 101: A guide to creating your first ontology.

[Noy and Musen, 2002] Noy, N. F. and Musen, M. A. (2002). PROMPTDIFF: A fixed-point algorithm for comparing ontology versions. In Dechter, R., Kearns, M. J., and Sutton, R. S., editors, *The Eighteenth National Conference on AI and Fourteenth Conference on Innovative Applications of AI, Canada.* AAAI Press / The MIT Press.

[Noy and Musen, 2004] Noy, N. F. and Musen, M. A. (2004). Ontology versioning in an ontology management framework. *IEEE Intell. Syst.*, 19(4):6–13.

[Ottens et al., 2007] Ottens, K., Aussenac-Gilles, N., Gleizes, M. P., and Camps, V. (2007). Dynamic ontology co-evolution from texts: Principles and case study. In Chen, L., Cudré-Mauroux, P., Haase, P., Hotho, A., and Ong, E., editors, *Proceedings of the First International Workshop on Emergent Semantics and Ontology Evolution, ESOE 2007, co-located with ISWC 2007 + ASWC 2007, Busan, Korea*, volume 292 of *CEUR Workshop Proceedings*, pages 70–83. CEUR-WS.org.

[Papavasileiou et al., 2013] Papavasileiou, V., Flouris, G., Fundulaki, I., Kotzinos, D., and Christophides, V. (2013). High-level change detection in RDF(S) kbs. *ACM Trans. Database Syst.*, 38(1):1:1–1:42.

[Papavassiliou et al., 2009] Papavassiliou, V., Flouris, G., Fundulaki, I., Kotzinos, D., and Christophides, V. (2009). On detecting high-level changes in RDF/S kbs. In Bernstein, A., Karger, D. R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., and Thirunarayan, K., editors, *ISWC 2009, 8th International Semantic Web Conference,USA.*, volume 5823 of *Lecture Notes in Computer Science*, pages 473–488. Springer.

[Peroni, 2016] Peroni, S. (2016). A simplified agile methodology for ontology development. In [Dragoni et al., 2017], pages 55–69.

[Pinto et al., 2004] Pinto, H. S., Staab, S., and Tempich, C. (2004). DILIGENT: towards a fine-grained methodology for distributed, loosely-controlled and evolving engineering of ontologies. In de Mántaras, R. L. and Saitta, L., editors, *Proceedings of the 16th Eureopean Conference on Artificial Intelligence, ECAI'2004, including*

Prestigious Applicants of Intelligent Systems, PAIS 2004, Valencia, Spain, August 22-27, 2004, pages 393–397. IOS Press.

[Poveda Villalón, 2016] Poveda Villalón, M. (2016). *Ontology Evaluation: a pitfall-based approach to ontology diagnosis*. PhD thesis, ETSI_Informatica.

[Poveda-Villalón et al., 2014] Poveda-Villalón, M., Gómez-Pérez, A., and Suárez-Figueroa, M. C. (2014). Oops! (ontology pitfall scanner!): An on-line tool for ontology evaluation. *Int. J. Semantic Web Inf. Syst.*, 10(2):7–34.

[Poveda Villalón et al., 2012] Poveda Villalón, M., Suárez-Figueroa, M. C., and Gómez-Pérez, A. (2012). The landscape of ontology reuse in linked data. *Proceedings Ontology Engineering in a Data-driven World (OEDW 2012)*.

[Pruski et al., 2011] Pruski, C., Guelfi, N., and Reynaud, C. (2011). Adaptive ontology-based web information retrieval: The TARGET framework. *IJWP*, 3(3):41–58.

[Qawasmeh, 2018] Qawasmeh, O. (2018). A collaborative framework for ontology and instance data co-evolution and extraction. In Hollink, L. and Osborne, F., editors, *Proceedings of the EKAW Doctoral Consortium 2018 co-located with the 21st International Conference on Knowledge Engineering and Knowledge Management (EKAW 2018), Nancy, France, November 13, 2018.*, volume 2306 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Qawasmeh et al., 2018] Qawasmeh, O., Lefrançois, M., Zimmermann, A., and Maret, P. (2018). Computer-assisted ontology construction system: Focus on bootstrapping capabilities. In Gangemi, A., Gentile, A. L., Nuzzolese, A. G., Rudolph, S., Maleshkova, M., Paulheim, H., Pan, J. Z., and Alam, M., editors, *The Semantic Web: ESWC 2018 Satellite Events - ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers*, volume 11155 of *Lecture Notes in Computer Science*, pages 60–65. Springer.

[Qawasmeh et al., 2019] Qawasmeh, O., Lefrançois, M., Zimmermann, A., and Maret, P. (2019). Observing the impact and adaptation to the evolution of an imported ontology. In Dietz, J. L. G., Aveiro, D., and Filipe, J., editors, *Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, IC3K 2019, Volume 2: KEOD, Vienna, Austria, September 17-19, 2019.*, pages 76–86. ScitePress.

[Rapoza, 2006] Rapoza, J. (2006). Sparql will make the web shine. *eWeek. com*, 2.

[Redmond et al., 2008] Redmond, T., Smith, M., Drummond, N., and Tudorache, T. (2008). Managing change: An ontology version control system. In Dolbear, C., Ruttenberg, A., and Sattler, U., editors, *Proceedings of the Fifth OWLED Workshop on OWL: Experiences and Directions, collocated with the 7th International Semantic Web Conference (ISWC-2008), Karlsruhe, Germany, October 26-27, 2008*, volume 432 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Rieß et al., 2010] Rieß, C., Heino, N., Tramp, S., and Auer, S. (2010). Evopat - pattern-based evolution and refactoring of RDF knowledge bases. In Patel-Schneider, P. F., Pan, Y., Hitzler, P., Mika, P., Zhang, L., Pan, J. Z., Horrocks, I., and Glimm, B., editors, *The Semantic Web - ISWC 2010 - 9th International Semantic Web Conference, ISWC 2010, Shanghai, China, November 7-11, 2010, Revised Selected Papers, Part I*, volume 6496 of *Lecture Notes in Computer Science*, pages 647–662. Springer.

[Rogozan and Paquette, 2005] Rogozan, D. and Paquette, G. (2005). Managing ontology changes on the semantic web. In Skowron, A., Agrawal, R., Luck, M., Yamaguchi, T., Morizet-Mahoudeaux, P., Liu, J., and Zhong, N., editors, *2005 IEEE / WIC / ACM International Conference on Web Intelligence (WI 2005), 19-22 September 2005, Compiegne, France*, pages 430–433. IEEE Computer Society.

[Sabou and Fernández, 2012] Sabou, M. and Fernández, M. (2012). Ontology (network) evaluation. In [Suárez-Figueroa et al., 2012c], pages 193–212.

[Sahlgren, 2005] Sahlgren, M. (2005). An introduction to random indexing. *na*.

[Savic et al., 2019] Savic, M., Ivanovic, M., and Jain, L. C. (2019). *Complex Networks in Software, Knowledge, and Social Systems*, volume 148 of *Intelligent Systems Reference Library*. Springer.

[Schreiber and Raimond, 2014] Schreiber, A. T. and Raimond, Y. (2014). Rdf 1.1 primer. *World-Wide Web Consortium*.

[Segaran et al., 2009] Segaran, T., Evans, C., and Taylor, J. (2009). *Programming the Semantic Web - Build Flexible Applications with Graph Data*. O'Reilly.

[Simperl and Luczak-Rösch, 2014] Simperl, E. and Luczak-Rösch, M. (2014). Collaborative ontology engineering: a survey. *Knowledge Eng. Review*, 29(1):101–131.

[Simperl, 2009] Simperl, E. P. B. (2009). Reusing ontologies on the semantic web: A feasibility study. *Data Knowl. Eng.*, 68(10):905–925.

[SmartM2M, 2019] SmartM2M, E. (2019). Saref consolidation with new reference ontology patterns, based on the experience from the seas project. 2019 jul. report no.: Ts 103 548 v1. 1.1.

[Stojanovic, 2004] Stojanovic, L. (2004). *Methods and tools for ontology evolution*. PhD thesis, Karlsruhe Institute of Technology, Germany.

[Suárez-Figueroa et al., 2011] Suárez-Figueroa, M. C., García-Castro, R., Villazón-Terrazas, B., and Gómez-Pérez, A. (2011). Essentials in ontology engineering: methodologies, languages, and tools. *na*, pages 9–21.

[Suárez-Figueroa et al., 2012a] Suárez-Figueroa, M. C., Gómez-Pérez, A., and Fernández-López, M. (2012a). The neon methodology for ontology engineering. In [Suárez-Figueroa et al., 2012c], pages 9–34.

[Suárez-Figueroa et al., 2012b] Suárez-Figueroa, M. C., Gómez-Pérez, A., Motta, E., and Gangemi, A. (2012b). Introduction: Ontology engineering in a networked world. In [Suárez-Figueroa et al., 2012c], pages 1–6.

[Suárez-Figueroa et al., 2012c] Suárez-Figueroa, M. C., Gómez-Pérez, A., Motta, E., and Gangemi, A., editors (2012c). *Ontology Engineering in a Networked World*. Springer.

[Suchanek et al., 2007] Suchanek, F. M., Kasneci, G., and Weikum, G. (2007). Yago: a core of semantic knowledge. In Williamson, C. L., Zurko, M. E., Patel-Schneider, P. F., and Shenoy, P. J., editors, *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, pages 697–706. ACM.

[Sure et al., 2004] Sure, Y., Staab, S., and Studer, R. (2004). On-to-knowledge methodology (OTKM). In Staab, S. and Studer, R., editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 117–132. Springer.

[Tartir et al., 2005] Tartir, S., Arpinar, I. B., Moore, M., Sheth, A. P., and Aleman-Meza, B. (2005). Ontoqa: Metric-based ontology quality analysis. *https://corescholar.libraries.wright.edu/knoesis/660*.

[Tartir et al., 2010] Tartir, S., Arpinar, I. B., and Sheth, A. P. (2010). Ontological evaluation and validation. In *Theory and applications of ontology: Computer applications*, pages 115–130. Springer.

[Tastle and Wierman, 2007] Tastle, W. J. and Wierman, M. J. (2007). Consensus and dissention: A measure of ordinal dispersion. *Int. J. Approx. Reasoning*, 45(3):531–545.

[Toulet et al., 2018] Toulet, A., Dutta, B., and Jonquet, C. (2018). Assessing the practice of ontology metadata: A survey result. *Research Report, University of Montpellier*.

[Vandenbussche et al., 2017] Vandenbussche, P., Atemezing, G., Poveda-Villalón, M., and Vatant, B. (2017). Linked open vocabularies (LOV): A gateway to reusable semantic vocabularies on the web. *Semantic Web*, 8(3):437–452.

[Vigo et al., 2014] Vigo, M., Bail, S., Jay, C., and Stevens, R. D. (2014). Overcoming the pitfalls of ontology authoring: Strategies and implications for tool design. *Int. J. Hum.-Comput. Stud.*, 72(12):835–845.

[Villazón-Terrazas, 2012] Villazón-Terrazas, B. (2012). *A Method for Reusing and Re-engineering Non-ontological Resources for Building Ontologies*, volume 12 of *Studies on the Semantic Web*. IOS Press.

[Vrandecic and Krötzsch, 2014] Vrandecic, D. and Krötzsch, M. (2014). Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

[Weigel et al., 2014] Weigel, T., Kindermann, S., and Lautenschlager, M. (2014). Actionable persistent identifier collections. *Data Science Journal*, 12:191–206.

[Whetzel et al., 2011] Whetzel, P. L., Noy, N. F., Shah, N., Alexander, P. R., Dorf, M., Fergerson, R. W., Storey, M. D., Smith, B., Chute, C. G., and Musen, M. A. (2011). Bioportal: Ontologies and integrated data resources at the click of a mouse. In Bodenreider, O., Martone, M. E., and Ruttenberg, A., editors, *Proceedings of the 2nd International Conference on Biomedical Ontology, Buffalo, NY, USA, July 26-30, 2011*, volume 833 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Wilkinson et al., 2016] Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., et al. (2016). The fair guiding principles for scientific data management and stewardship. *Scientific data*, 3.

[Zablith, 2009] Zablith, F. (2009). Ontology evolution: A practical approach. In *Workshop on Matching and Meaning at Artificial Intelligence and Simulation of Behaviour*.

[Zablith et al., 2015] Zablith, F., Antoniou, G., d'Aquin, M., Flouris, G., Kondylakis, H., Motta, E., Plexousakis, D., and Sabou, M. (2015). Ontology evolution: a process-centric survey. *Knowledge Eng. Review*, 30(1):45–75.

[Zablith et al., 2010] Zablith, F., d'Aquin, M., Sabou, M., and Motta, E. (2010). Using ontological contexts to assess the relevance of statements in ontology evolution. In Cimiano, P. and Pinto, H. S., editors, *Knowledge Engineering and Management by the Masses - 17th International Conference, EKAW 2010, Lisbon, Portugal, October 11-15, 2010. Proceedings*, volume 6317 of *Lecture Notes in Computer Science*, pages 226–240. Springer.

[Zablith et al., 2009] Zablith, F., Sabou, M., d'Aquin, M., and Motta, E. (2009). Ontology evolution with evolva. In Aroyo, L., Traverso, P., Ciravegna, F., Cimiano, P., Heath, T., Hyvönen, E., Mizoguchi, R., Oren, E., Sabou, M., and Simperl, E. P. B., editors, *The Semantic Web: Research and Applications, 6th European Semantic Web Conference, ESWC 2009, Heraklion, Crete, Greece, May 31-June 4, 2009, Proceedings*, volume 5554 of *Lecture Notes in Computer Science*, pages 908–912. Springer.

[Zarembo, 2015] Zarembo, I. (2015). Automatic transformation of relational database schema into owl ontologies. In *Proceedings of the 10th International Scientific and Practical Conference. Volume III*, volume 217, page 222.

[Zhang et al., 2015] Zhang, Y., Tudorache, T., Horridge, M., and Musen, M. A. (2015). Helping users bootstrap ontologies: An empirical investigation. In Begole, B., Kim, J., Inkpen, K., and Woo, W., editors, *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI 2015, Seoul, Republic of Korea, April 18-23, 2015*, pages 3395–3398. ACM.