Harvesting Online Ontologies for Ontology Evolution

Fouad Zablith

Submitted in partial fulfilment of the

requirements for the degree of

Doctor of Philosophy

The Knowledge Media Institute (KMi)

The Open University

02 August 2011

Milton Keynes

The United Kindgom

Abstract

Ontologies need to evolve to keep their domain representation adequate. However, the process of identifying new domain changes, and applying them to the ontology is tedious and time-consuming.

Our hypothesis is that online ontologies can provide background knowledge to decrease user efforts during ontology evolution, by integrating new domain concepts through automated relation discovery and relevance assessment techniques, while resulting in ontologies of similar qualities to when the ontology engineers' knowledge is solely used.

We propose, implement and evaluate solutions that exploit the conceptual connections and structure of online ontologies to first, automatically suggest new additions to the ontology in the form of concepts derived from domain data, and their corresponding connections to existing elements in the ontology; and second, to automatically evaluate the proposed changes in terms of relevance with respect to the ontology under evolution, by relying on a novel pattern-based technique for relevance assessment. We also present in this thesis various experiments to test the feasibility of each proposed approach separately, in addition to an overall evaluation that validates our hypothesis that user time during evolution is indeed decreased through the use of online ontologies, with comparable results to a fully manual ontology evolution.

Acknowledgements

I am delighted to be writing this part of my thesis, which I kept till the end, as it made me look back and realise how lucky I have been to be surrounded by people who made my PhD journey possible.

Going through my PhD, would have not been possible without the support of my wife Rania. She decided to leave everything behind and join me to start a new life in the UK. She has always believed (and kept reminding me) that this was for our best. I really appreciate her patience and efforts to keep our life in balance. Thank you for everything. I couldn't have made it without you.

I would like to express my gratitude to my parents, who gave me the freedom of choice and full support in whatever I do. They really encouraged me to pursue this thesis. Through my father Rodolph, I saw how someone can be self motivated, organised and diligent. Key features that tremendously help during a PhD. I hope one day I will be able to achieve a fraction of what he did for us. The continuous care and worry of my mother Sonia, gave me the drive to keep going. At many occasions I felt the need to comfort her that things are going ok. I am so lucky to have you both. I would also like to thank my sister Nadine and brother Jad for letting me know that they were always there if I needed anything.

I was first introduced to the domain of Semantic Web by Iyad Rahwan, my MSc. supervisor at the British University in Dubai. Not only he introduced me to this very interesting area, but also to the world of research. I still remember his words about our paper published together with Chris Reed in the Artificial Intelligence Journal, on how it will have a big impact; and it really had, at least on me. Through him I saw how exciting research can be, and started considering doing a bit more of it. Iyad was the one who sent me the advert and encouraged me to apply for the potential scholarship grant at the Knowledge Media Institute. Thank you Iyad, and I hope you will keep inspiring people the way you inspired me.

I am greatly thankful to my supervisors Enrico Motta, Mathieu d'Aquin and Marta Sabou. From day one I realised how fortunate I am to be guided by them. Their combined knowledge and different style in handling research activities worked well to my advantage. Between them, they supplied me with everything a PhD candidate could possibility ask for. Needless to say, they offered me continuous support throughout my PhD, and provided me with the adequate feedback whenever needed. Their guidance to publish my work at various stages of my PhD was key in directing the flow of my research. This helped in identifying ways to improve my research directions at early stages, and addressing them in the appropriate way. Their knowledge, advice, and challenging ideas were just priceless.

Another factor that helped me throughout my PhD is the great atmosphere we have in KMi. I would like to thank all my colleagues who made it a great experience in every way. I felt part of a family. I also can not forget how useful the postgraduate forums led by Marian Petre and Trevor Collins were during the first year of my PhD. We had the opportunity to discuss and share PhD related issues (among others) with fellow students. Special thanks to Mark Gaved for sharing joyful conversations when things were getting complicated, and for always reminding me that “a PhD is a marathon, not a sprint”.

I would also like to thank the KMi IT team Damian Dadswell, Lewis McCann, Paul Alexander and Robbie Bays for their swift IT support to my research and experimental work. In addition, I would like to thank our administration team for their assistance with the conference trips and administrative work related to my PhD. And many thanks to Harriett Cornish and Peter Devine for their creative design support in making my PhD posters and website look prettier.

The first three years of my PhD were co-funded by the NeOn project. I am thankful not only for the grant provided by the project, but also for the opportunity to meet and collaborate with great people.

I am also indebted to all who helped me in performing the evaluations. I would like to thank: Alessandro Adamou, Carlo Allocca, Sofia Angeletou, Claudio Baldassarre, Noam Bercovici, Eva Blomqvist, Enrico Daga, Miriam

Fernandez, Holger Lewen, Haiming Liu, Ainhoa Llorente, Vanessa Lopez, Maria Maleshkova, Andriy Nikolov, Raul Palma, Matthew Rowe, Mari Carmen Surez-Figueroa, Thomas Ullmann and Boris Villazon. Without their invaluable time and effort, it would have been hard to validate the work I did in this dissertation.

I would like to thank Nathalie Aussenac-Gilles and Nathalie Hernandez for giving me the opportunity to visit their research group in Toulouse for three weeks. I had the chance to present my research approaches to the team, and get constructive feedback about my work. We also had very useful discussions about potential collaboration ideas based on combining different techniques related to my thesis.

I would like to close with a special credit to my daughter Emma, who just turned one year old. I once read in the Zen Book of Meditation that "when things are in total disarray, play with joyful children". Emma you showed me what this means. Thank you for your smiles, cheerful attitude and serenity. You really helped me getting done with my thesis. I am so blessed to have you.

Publications

Parts of the thesis contributed to the following list of publications:

- F. Zablith, G. Antoniou, M. d'Aquin, G. Flouris, H. Kondylakis, E. Motta, D. Plexousakis and M. Sabou. Ontology Evolution: A Process Centric Survey. Submitted in 2011 to 'The Knowledge Engineering Review'.

- R. Palma, F. Zablith, P. Haase and O. Corcho (2011). Ontology Evolution. To appear in: 'Ontology Engineering in a Networked World', eds. Surez-Figueroa, M. C. et.al, Springer Verlag.

- F. Zablith, M. d'Aquin, M. Sabou and E. Motta (2010). Using Ontological Contexts to Assess the Relevance of Statements in Ontology Evolution, *in* 'Proceedings of the 17th Conference on Knowledge Engineering and Knowledge Management by the Masses (EKAW)', Lisbon, Portugal, Springer Verlag, pp. 226-240.

- F. Zablith, M. d'Aquin, M. Sabou and E. Motta (2009). Investigating the Use of Background Knowledge for Assessing the Relevance of Statements to an Ontology in Ontology Evolution, *in* 'Proceedings of the International Workshop on Ontology Dynamics (IWOD) at ISWC'. Northern Virginia, USA

- F. Zablith (2009). Evolva: A Comprehensive Approach to Ontology Evolution, *in* 'Proceedings of the 6th European Semantic Web Conference (ESWC) PhD Symposium LNCS 5554', eds. L. Aroyo et al., Springer-Verlag, Berlin, Heidelberg, pp. 944-948.

- F. Zablith, M. Sabou, M. d'Aquin and E. Motta (2009). Ontology Evolution with Evolva. Demo *in* 'Proceedings of the 6th European Semantic Web Conference (ESWC) LNCS 5554', eds. L. Aroyo et al., Springer-Verlag, Berlin, Heidelberg, pp. 908-912.

- F. Zablith (2009). Ontology Evolution: A Practical Approach. Poster *in* 'Proceedings of the AISB Workshop on Meaning and Matching (WMM)', Edinburgh, UK.

- M. Surez-Figueroa, E. Blomqvist, M. d'Aquin, M. Espinoza, A. Gmez-Prez, H. Lewen, I. Mozetic, R. Palma, M. Poveda, M. Sini, B. Villazon, F. Zablith and M. Dzbor (2009). D5.4.2 Revision and Extension of the NeOn Methodology for Building Contextualized Ontology Networks, NeOn Project Deliverable D5.4.2, Universidad Politécnica de Madrid.

- D. Maynard, N. Aswani, W. Peters, F. Zablith and M. d'Aquin (2009). D1.5.3 Advanced Methods for Change Propagation between Networked Ontologies and Metadata, NeOn Project Deliverable D1.5.3, University of Sheffield.

- M. d'Aquin, M. Sabou, E. Motta, S. Angeletou, L. Gridinoc, V. Lopez and F. Zablith (2008). What can be done with the Semantic Web? An Overview of Watson-based Applications, *in* '5th Workshop on Semantic Web Applications and Perspectives (SWAP)', Rome, Italy

- F. Zablith, M. Sabou, M. d'Aquin and E. Motta (2008). Using Background Knowledge for Ontology Evolution, *in* 'International Workshop on Ontology Dynamics (IWOD) at ISWC', Karlsruhe, Germany.

- F. Zablith (2008). Dynamic Ontology Evolution, *in* 'International Semantic Web Conference (ISWC) Doctoral Consortium', Karlsruhe, Germany.

- F. Zablith (2008). Evolva: Towards Automatic Ontology Evolution. KMi Technical Report: kmi-08-04.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The World Wide Web (Berners-Lee, 1999) was initially designed by placing humans as primary consumers of information. Thus, making sense of such information was solely dependent on users, who are able to find and interpret information based on their cognitive abilities. Recently, there was a push towards achieving more efficient and sophisticated information processing and access across the web. For example, instead of relying on keyword-based search engines on the web to find the required information, it would be favourable to supply more complex and adequate queries that can include for example finding researchers in a city working on related topics. In order to achieve this, it was identified that a more granular and structured representation of information at the data level is needed. Such a representation would allow a systematic and coherent interpretation of information using machine process-able techniques.

This idea of moving "from documents to data and information" (Shadbolt et al., 2006) is being concretised through extensive research and development performed under the umbrella of the Semantic Web (Berners-Lee et al., 2001).

To provide the required semantics behind web documents, data is modelled in what is called a Semantic Web *ontology*. Ontologies are formal representation of specific domains, and enable the modelling of a domain's conceptual entities with their corresponding relations (Gruber, 1993). They provide a well defined vocabulary to ensure that the modelled data connects in a coherent way not only within one ontology, but also with other ontologies across the web. Using such ontologies is a key requirement to a successful Semantic Web, for ensuring a seamless exchange of data with well defined meanings. It is at the ontology level where we specify for example that "student" is a concept of type "person", and that a "person" has a name, etc. Formally, an ontology is a set of statements, which we manipulate as a graph. Figure 1.1 shows a simplistic view of a part of an ontology in the academic domain, depicting how concepts can be connected through sub-class relations (i.e., *isa* relation), properties (e.g., a person *has a name*), and other named relations (e.g., a person *attends* an event). Such relations, which are also referred to as statements, are of the form $< subject, relation, object >$.

With all the benefits introduced by the Semantic Web, we witness an in-

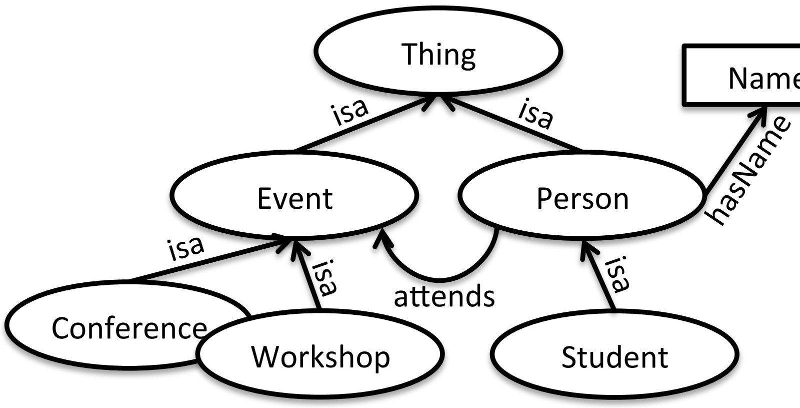


Figure 1.1: Example of a part of an ontology in the academic domain, showing the connections between the concepts in terms of subsumption relations (i.e., isa), named relations (e.g., attends) and properties (e.g., hasName).

crease in the interest of representing existing domain data in the form of Semantic Web ontologies (Ding et al., 2007). However, using ontologies comes with the cost of keeping them up-to-date with the emerging entities from the modelled domain. Such entities, for example new concepts, should be appropriately modelled in a timely manner within the ontology, to keep the domain representation as accurate as possible. This process is what we refer to as *ontology evolution*.

Ontology evolution involves many tasks, including: (1) detecting the need for evolution based on domain data or usage; (2) suggesting ontology changes by relying on unstructured or structured knowledge sources; (3) validating ontology changes based on domain information or formal properties methods;

(4) assessing the evolution impact through usage analysis or formal criteria; and (5) managing changes that include ontology versioning mechanisms and recording such changes. With all the tasks to deal with, among which some are complex and cognitively challenging, ontology evolution is a tedious and time-consuming task.

In this thesis, we focus on specific parts of the ontology evolution process. Our aim is to support users in (a) automatically identifying new elements to add to the ontology in the form of domain concepts and corresponding relations; and (b) automatically evaluating the relevance of the new proposed statements to add. Our hypothesis is that:

> Online ontologies can provide background knowledge to decrease the need for user involvement during ontology evolution, by integrating new domain concepts through automated relation discovery and relevance assessment techniques, while resulting in ontologies of similar qualities to when the ontology engineers' knowledge is solely used.

Our target users in this research are ontology engineers, responsible for building and maintaining ontologies to use in specific environments. For example in the biology domain, where the user has to constantly translate biology related information and findings into ontology compatible entities. Another example can be in Linked Data scenarios, for example in the academic context “`http://data.open.ac.uk`” (Zablith et al., 2011), where the ontology needs to evolve to adequately encapsulate the university’s conceptual data layer. Our aim is to decrease the time needed from the ontology engineer in such scenarios, while preserving the quality of the ontology under evolution, compared to when the user is required to perform the same tasks without relying on our proposed techniques.

1.1 Thesis Contributions

By focusing on finding a solution to solve the aforementioned challenge in ontology evolution, this thesis contributes to the following key aspects in the field:

1. Devising a technique to automatically identify potential ontology changes by integrating new emerging concepts from external domain data, based on the background knowledge provided by existing structured sources (especially, online ontologies).

2. Providing a novel approach to automatically assess the relevance of potential ontology changes by analysing the contexts of online ontologies from where they are derived, versus the ontology under evolution.

3. Putting together a methodology for evaluating and comparing the structural differences between different evolution settings of the same ontol-

ogy, through empirical observations supported by metric-based assessment techniques.

1.2 Problem and Research Questions

Ontology evolution is a tedious and time-consuming process, especially at the levels of detecting and evaluating new domain entities to add to the ontology, the core focus of this thesis. It requires the user first to identify domain changes, i.e., new concepts in our case, and relate them to existing knowledge in the ontology. Second, it is expected that the user evolves the ontology by integrating only changes that are relevant to the ontology under evolution. These prerequisites require the user to have a good understanding and expertise in the domain that the ontology represents, and how to model the changes appropriately in the ontology. In addition to those skills, the tasks of identifying domain changes and analysing which ones are worth to be added to ontology are cognitively challenging and time-consuming. With all these barriers, it is highly probable that ontologies are left outdated, making them unusable in their applied domains.

This thesis aims to answer the following research question:

How to support users in the process of ontology evolution?

After investigating the tasks where user input is the most needed during ontology evolution, we focus our research interests to answer the following sub-questions:

1. *How to assist users in identifying ontology change opportunities?* One of the challenges of ontology evolution is to identify what is new in the domain, and to transform it into a candidate change of the ontology. To tackle this question, we aim to provide a technique to automatically detect new concepts that emerge in the domain. Moreover, we investigate how to resolve the appropriate relations that link such detected concepts to existing concepts in the ontology.

2. *How to assess the relevance of ontology changes?* The problem with automatically identifying new ontology changes is that a substantial part of the changes might be irrelevant to add to the ontology. Hence the user will have to manually check a large number of changes, to decide whether or not they are worth to apply on the ontology. We focus in our research on devising a technique to automatically assess the relevance of ontology changes with respect to an ontology. This will serve as a support to users in the evaluation process of the changes to apply during ontology evolution.

In addition, in order to substantiate the value of the proposed approaches, we need to realise them in a coherent and usable tool. While not a core research contribution, the realisation of such a tool provides the basis for the validation and evaluation of the integration of the two proposed techniques, as well as, more generally, a reusable environment for the ontology evolution research.

1.3 Methodology

We begin our research by reviewing the existing works that tackle the various tasks involved in ontology evolution. This helps us highlight two gaps where user input is still highly needed, and define our research interests. We proceed with identifying our hypothesis, which lead to expanding and proposing our approaches to tackle the two gaps identified. Then we describe the implementation of our proposed approaches in a reusable tool, in order to conduct an experiment to evaluate our approaches and support our hypothesis.

1.3.1 Reviewing Existing Works and Identifying the Gaps in the Literature

We begin our research by performing a literature review. Based on the analysis of various frameworks in ontology evolution, we identify the tasks involved in the ontology evolution process, and represent the tasks in an evolution cycle in the form of a diagram. Then we analyse existing works performed in each of the tasks.

Our review highlights two major gaps in the literature, hampering the evolution process. Firstly, the process of automatically identifying domain changes is mainly limited to the content of existing domain data (e.g., text corpora). We argue that such domain data do not contain a sufficiently rich source of domain background knowledge to resolve the needed additions to the ontology, as not all content and conceptual connections are explicitly elicited in the text. The second identified gap is that the evaluation of the relevance of changes with respect to the ontology under evolution is usually left to the user. Due to the substantial amount of ontology additions proposed by automated techniques, users will be left dealing with a new burden of checking a big amount of ontology changes. These two gaps form the core starting points of our research focus.

1.3.2 Proposing our Research Approach

We believe that online ontologies, with the appropriate techniques in processing them, can support a higher degree of automation for some of the tasks involved in ontology evolution. In this thesis, we focus on three of the evolution tasks mentioned previously in the introduction. This includes detecting the need for ontology evolution from domain data (i.e., Task 1), suggesting ontology changes by relying on structured knowledge sources (i.e., Task 2) and validating ontology changes based on domain data (i.e., Task 3).

We propose an approach to derive domain changes by reusing available knowledge (mainly through online ontologies), in order to integrate new domain entities (identified from existing domain data such as text documents) into the ontology under evolution. While ontology changes can include adding, removing and modifying existing entities (i.e., concepts, instances and relations), we focus our investigations on adding new concepts along with new relations to the ontology. We support this approach by an experiment that we conduct in the academic domain, to test and evaluate the use of the lexical database WordNet and online ontologies as background knowledge to link new domain entities identified from the text documents, to existing concepts in the ontology. The experiment helped us evaluate the correctness of the automatically generated relations that link the new domain concepts to existing ones in the ontology, and also observe how the relation discovery process can be improved based on the analysis of the results.

Then we tackle the second gap we identified, for which we present and compare two approaches for identifying the relevance of statements (i.e., ontology changes) with respect to an ontology. We propose using the contexts provided by online ontologies from where the statements are identified, and contrast them with the context of the ontology under evolution. Based on the analysis of various graph-based context examples, we identify structural patterns that give an indication of relevance of statements. We also conduct an evaluation, which we perform in three different domains showing the accuracy of this approach.

1.3.3 Implementing our Approaches

After testing our techniques individually, we concretise our research investigations in a coherent ontology evolution tool, in which we implement our techniques for assisting the user in finding and evaluating ontology changes in the form of statements to be added to the ontology. This tool proved to be essential in evaluating our proposed research approaches.

1.3.4 Conducting an Overall Evaluation

Finally we perform an overall evaluation of our approach, by comparing and contrasting three ontology evolution modes: manual, semi-automatic and unsupervised. We log the time taken by the experts, and the entities added to each ontology in the three modes. This enables us to get a direct indication of the impact of using online ontologies on the process of ontology evolution in terms of user time, and the resulting ontologies' characteristics.

1.4 Overview of the Approach

The starting point of our process is an ontology. We place the ontology under evolution at the centre of our approach, to serve as guide in the change discovery and evaluation process. Our assumption is that if a domain entity connects to existing elements in the ontology, it can potentially be part of the evolution process. Next we give an overview of how we identify new domain entities from existing domain data such as text documents (Section 1.4.1). This is followed by a brief introduction to online ontologies, and how they can be accessed to provide background knowledge to propose changes to the ontology in order to integrate the newly identified domain entities (Section 1.4.2). Then we present our approach on assessing the relevance of a change with respect to the ontology under evolution (Section 1.4.3). Thereafter we introduce Evolva, our ontology evolution tool which makes use of our proposed techniques, with the aim to decrease user input during the evolution process through a unified and integrated platform (Section 1.4.4).

1.4.1 Discovering new Domain Entities

The first task we investigate focuses on identifying new domain changes. Domain information abundantly exist in various formats. This can be for example in the form of text documents accessible through the web or internally within an organisation, databases that have been used to store information in a structured way, or simply a list of already identified terms or folksonomies. Hence one way to identify new domain entities is through the analysis of the content of such sources, and contrasting it to the knowledge available within the ontology to evolve (an extended overview of methods for detecting the need for change is discussed in Section 2.3.1). For example processing a corpus of text documents in a research institution highlights the presence of the terms "researcher" and "tutorial" (among others) from the following text snippet:

> The researcher is writing a proposal for a tutorial to be given at the conference.

If we require to evolve the ontology modelled in Figure 1.1 based on this text snippet, a comparison of the identified terms from the text, with respect to existing elements in the ontology highlights that such terms (i.e., researcher and tutorial) are new.

1.4.2 Using Online Ontologies for Automatic Change Discovery

After identifying new terms from domain data, the next task is to integrate them in the ontology using the appropriate links to existing ontology concepts. In our example, how should the concepts "tutorial" and "researcher" be added to the ontology? Traditionally, the expert's knowledge in the domain is the main source of input at this level. Hence for a given term, experts would rely on their own knowledge of the domain to identify, in the ontology under evolution, elements related to the term, as well as the actual relations they share. This creates a bottleneck in the evolution process, hampering the integration of domain changes in a timely manner.

Recently various approaches have focused on the task of suggesting new ontology changes (more details are discussed in Section 2.3.2). In particular, some techniques investigated the use of the information residing in the text documents as a source of identifying links between domain concepts (Cimiano and Volker, 2005; Maynard et al., 2009). However, there are limitations in such approaches, due to the fact that not all background knowledge can be derived from within the text in focus. Moreover, it is very unlikely that obvious relations, e.g., specifying that researcher is of type person, can be found in text documents. While this is easy for a human to catch, further information should be provided in order to achieve this process automatically.

To solve this issue, (Cimiano and Volker, 2005) propose using lexical databases (i.e., WordNet (Fellbaum, 1998)) to improve relation finding between concepts identified within text sources. We argue that lexical databases have limitations, and we propose using online ontologies as a source of identifying the relations between new concepts identified from the sources, and existing concepts in the ontology.

Online ontologies are ontologies openly accessible through the web. Various tools have been developed for users to search and locate such ontologies (Ding et al., 2004; Oren et al., 2008). In particular, one tool that goes beyond the ontology search features, is Watson (d'Aquin and Motta, 2011), which provides also mechanisms to consume and reuse the knowledge directly from online ontologies. Such features enable a direct exploitation of online ontologies to perform various tasks that proved very useful for achieving our goals.

This source of background knowledge provides a natural, yet effective, constantly evolving body of interconnected conceptual entities. It serves as a substantial solution to our task of linking new concepts to the existing entities in the ontology. For that, we propose using Scarlet (Sabou et al., 2008), a relation discovery engine that relies on Semantic Web ontologies to connect two terms. Figure 1.2 highlights the idea of how online ontologies (depicted in the cloud of the figure), provide knowledge to link "researcher" as a type of "person", and "tutorial" as a type of "event". The advantage is that such relations can be directly exploited, as they are already represented in an ontology-compatible format.

1.4.3 Automatically Assessing the Relevance of Statements

The next critical task that we focus on in our research is helping the user in the selection process of relevant changes to take into account for evolving the ontology. The abundance of domain data available to process often results in having a significant number of new terms to add to the ontology, which means a considerable amount of new changes to apply to the ontology. This will create a new burden at the user level, who will have to go through a

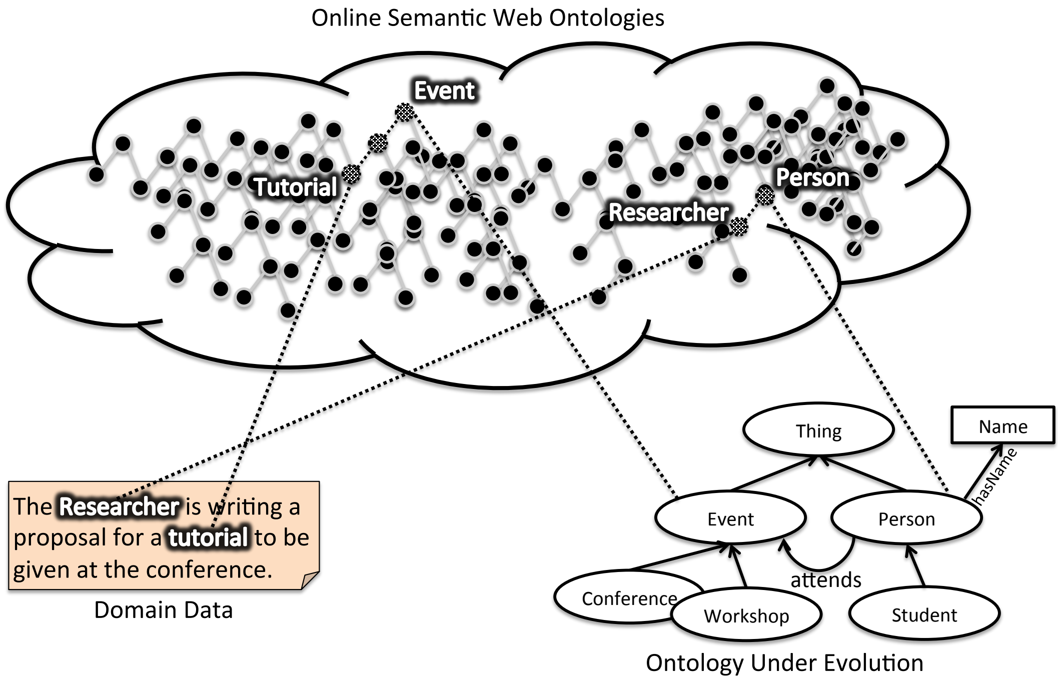


Figure 1.2: Online ontologies used as background knowledge to provide relations between terms identified from domain data, and existing concepts in the ontology under evolution.

big number of changes, and select only the ones that fit in the ontology. For example, consider the case where "birth" is identified as a new term from one of the domain data. Subsequently, an online ontology containing information about "birth", could result in adding it as a type of "event" to our ontology. How can we automatically identify that adding "tutorial" as an "event", is much more relevant than adding "birth" as an event to our ontology in the academic context?

We tackle this process by relying on the analysis of the context of the ontology to evolve, with respect to the context of the online ontology from where the relation is derived. The context provides further information about the neighbouring entities of the statement (i.e., new change proposed), making it feasible to automatically analyse the situation in which the statement is used. Figure 1.3 illustrates an example of possible surrounding entities (i.e., contexts) of the statements $<Tutorial, subClass, Event>$ and $<Birth, subClass, Event>$ (Contexts A and B respectively). The example clearly shows that Context A, where entities like "Workshop" and "Conference" exist, aligns better with the ontology under evolution, than Context B, which is mainly surrounded by other types of events e.g., "Wedding", "Death", etc.

We propose a pattern-based relevance detection mechanism, which takes into account the structure of both ontologies including the shared concepts and their position with respect to the new change to add to the ontology. We define five relevance patterns, each supported by a confidence formula to reflect the degree of relevance of a change with respect to the ontology. For example, one of the patterns detects the presence of shared siblings between the two contexts, which makes the statement to add relevant to be part of the ontology. As we show later in this dissertation, our pattern-based technique outperforms an overlap-based mechanism that takes into account only the overlap between the contexts, without any structural analysis.

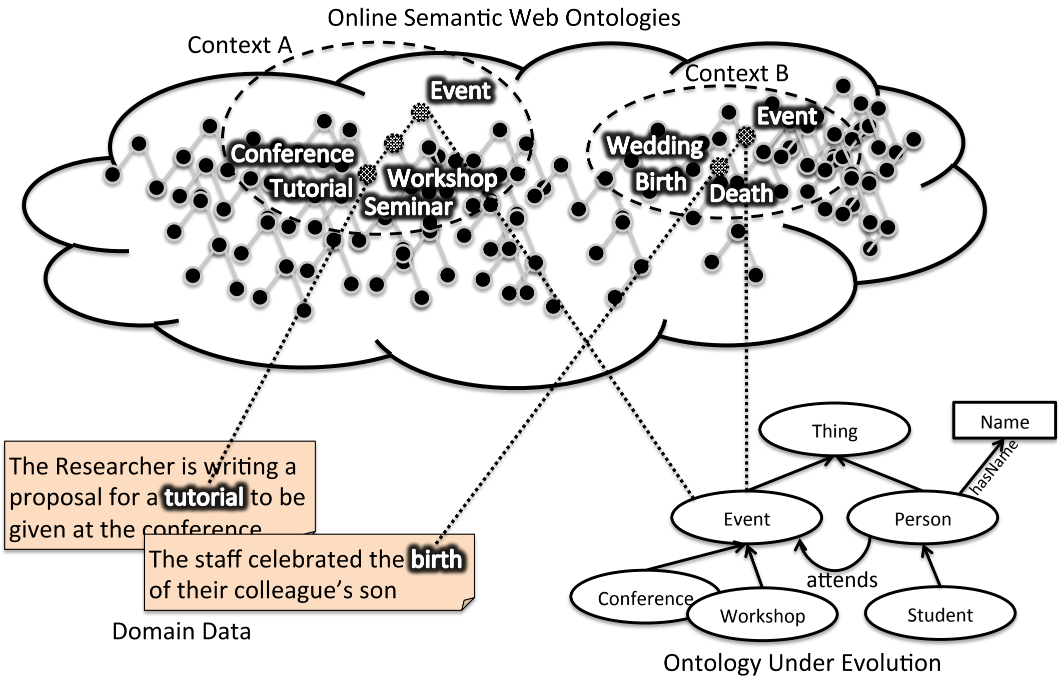


Figure 1.3: Illustration of the contexts derived from the surrounding elements in online ontologies for the statements <*Tutorial, subClass, Event*> (Context A) and <*Birth, subClass, Event*> (Context B).

1.4.4 A Tool for Ontology Evolution

After our investigations on decreasing user inputs at the levels of finding domain changes, and assessing the relevance of such changes using ontological contexts, we implement our ideas in one integrated tool called Evolva[1]. Evolva is a step-by-step tool that assists users in evolving ontologies. We implement it as part of the NeOn Toolkit[2], an ontology engineering tool that supports

[1]`http://evolva.kmi.open.ac.uk`

[2]`http://www.neon-toolkit.org`

ontologies' life-cycles.

Evolva enables users to select an ontology of interest and provide domain data including text corpora, list of terms and RSS feeds from where domain entities are automatically identified. The entities are then processed to spot new elements that do not exist in the ontology, in addition to filtering out terms that fall below certain quality criteria (e.g., term length). The entities are displayed within the interface for the user to check. Furthermore, Evolva provides users with a customisable access to background knowledge offered through WordNet and online ontologies, which are used to automatically link the new entities to the ontology as described earlier. The user is able to see the list of all proposed changes, along with the source from where they have been derived. At this level we integrate our work on the relevance assessment, based on which changes can be ranked in the interface. In addition, the user has the ability to visualise the contexts of the ontologies and how they match, to check how the relevance is calculated. This visualisation has also customisable parameters through the interface of Evolva. Finally the user will see the list of all approved changes, and has the option to add the changes to the source ontology directly, or create a new detached version. The new ontology can be directly accessible and ready to be used within the NeOn Toolkit.

In brief, Evolva helps in bootstrapping the process of evolving ontologies by analysing existing domain data, and relying on online ontologies as source of background knowledge. We show in our thesis how this can save ontology engineers a substantial amount of time in keeping the ontology up-to-date with respect to the domain.

1.5 Structure of the Thesis

In Chapter 2 we present a complete cycle of the tasks involved in ontology evolution. We use this cycle to review the existing works involved in each of the tasks. This state of the art helped us identify two major gaps in the literature. The first gap lies at the level of identifying new ontology changes, where the use of online ontologies to support this process is not well explored in the context of ontology evolution. The second gap is at the level of assessing the relevance of changes with respect to an ontology, which has been traditionally left for the user to do.

We explore in Chapter 3 the use of online ontologies and WordNet as background knowledge to identify ontology changes. We perform an experiment in the academic context to evaluate the feasibility of automatically generating ontology changes. We discuss and analyse our results, based on which we observe initial room for improvements in our approach.

Our work on assessing the relevance of changes with respect to the ontology is presented in Chapter 4. We setup an experiment in three different domains, to evaluate our automated approach in relevance detection based on the analysis of ontological contexts. The results show that our technique can be effectively used to rank the changes according to their relevance values, useful for supporting the user at the level of change evaluation.

In Chapter 5 we discuss our ontology evolution tool Evolva. We first present the framework on top of which it is implemented, followed by its implementation details as a plugin to the NeOn Toolkit.

In Chapter 6 we present an evaluation of the impact of using online ontologies in the process of ontology evolution. We perform the evaluation in the computing and IT services domain. We use the collected data to compare the performance in three different evolution scenarios: (1) a in fully manual mode where the evaluator is responsible for linking the identified concepts to the ontology, without relying on online ontologies; (2) a semi-automatic mode where online ontologies are used to identify and assess the relevance of ontology changes, which are then checked by the evaluator; and (3) an unsupervised mode where online ontologies are used to find ontology changes and assess their relevance without relying on user input and validation. Our main finding is that online ontologies can substantially decrease user time during the ontology evolution process, without having a major effect on the quality of the resulting ontology, compared to the user in fully manual mode.

Finally we conclude our thesis with Chapter 7 by revisiting the research problems we investigated, and giving an overview of our proposed approaches to solve them. We also discuss a vision related to the future directions in the field of ontology evolution, and the long term use and impact of Evolva on the production and maintenance side of ontologies' life-cycles.

Chapter 2

State of the Art in Ontology Evolution

Ontology evolution is not a simple process. It involves various tasks that fulfil specific requirements to achieve a successful evolution, reflecting the domain needs, with the aim to keep the ontology usable in its applied context. Such tasks range from the detection of the need for evolution, which can be derived for example from external data sources or usage patterns, to the management side of the evolution, which include recording changes applied to the ontology and handling various ontology versions. The chain of required tasks forms what we call the ontology evolution cycle. The extensive research done within each of these steps reflects the importance of this matter to reach a potential solution that supports the process of ontology evolution and maintenance.

In this chapter, we present an overview of the approaches involved in ontology evolution. We analyse them in terms of their support to the main tasks required to perform a complete ontology evolution cycle. We start by describing the diagram of processes involved in the ontology evolution cycle illustrated in Figure 2.1 (Section 2.1). This is followed by discussing related work involved in ontology evolution frameworks (Section 2.2). Then we give an overview of existing approaches in each ontology evolution task, with an emphasis on the work performed within the scope of our research questions (Section 2.3).

2.1 Ontology Evolution Cycle

In this part, we present a complete ontology evolution cycle that we illustrate in Figure 2.1. This cycle is the result of the analysis of the various approaches and frameworks that contribute to the area of ontology evolution, and covers the tasks involved in ontology evolution. We extend the existing tasks in the literature to cover a more granular representation of the possible steps in ontology evolution. This helped in (1) highlighting what is required to evolve an ontology, and (2) identifying the tasks where online ontologies can provide background knowledge to the process to decrease user input.

The starting point of the ontology evolution process is to identify what triggers the need for evolution (Task 1). For example, some works focus on

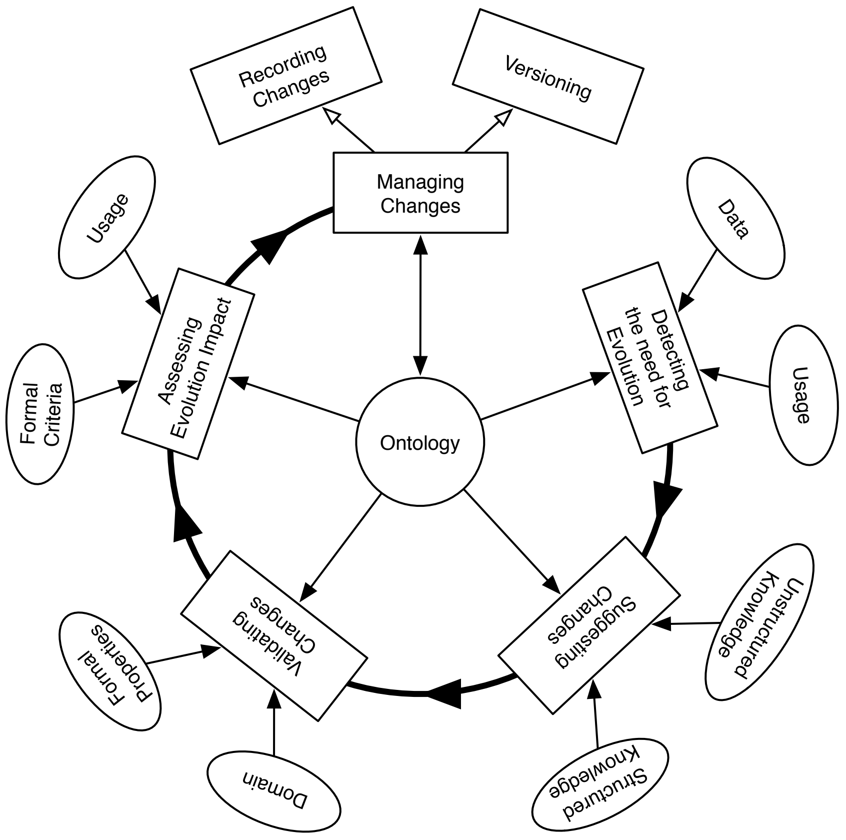


Figure 2.1: Ontology evolution cycle.

deriving changes through the analysis of user behaviour (Alani et al., 2006; Stojanovic, 2004), while others rely on data (e.g., text, or documents' annotations) to trigger the process (Bloehdorn et al., 2006; Cimiano and Volker, 2005; Maynard et al., 2009, 2007; Novacek et al., 2007; Ottens et al., 2007; Velardi et al., 2001). After identifying the need for changing the ontology, changes are represented and suggested to be applied to the ontology (Task 2). Some approaches handle this task by relying on unstructured knowledge (e.g., text documents by applying textual patterns to a corpus content representing the domain) (Cimiano and Volker, 2005; Maynard et al., 2009), while structured data sources are also used at this level, such as lexical databases (also proposed as extensions in (Cimiano and Volker, 2005; Maynard et al., 2009)), to refine the appropriate changes. The following step of the evolution cycle is validating the changes (Task 3). For that, there exist approaches that handle both a formal validation of changes making sure the ontology is logically consistent as per specified constraints (Carroll et al., 2004; Flouris et al., 2006; Haase et al., 2005; Huang et al., 2005; Ji et al., 2009; Schlobach and Cornet, 2003; Sirin et al., 2007), and a domain validation of changes taking into account the contextual information available (Cimiano and Volker, 2005; d'Aquin, 2009; Maynard et al., 2009; Sabou et al., 2009). After validating the changes, the impact of applying such changes to the ontology is measured (Task 4). In the state of the art, some approaches are based on a formal evaluation to assess the effect of a change on the ontology (Haase and Stojanovic, 2005; Palmisano et al., 2008; Pammer et al., 2009), while others measure the effect of such a change at the application and usage levels (Kondylakis et al., 2009; Lei et al., 2006; Liang et al., 2006; Wang et al., 2008), based for example on the ability to answer specified queries. The final step is the management of the changes of the ontology over time (Task 5), which is done through recording changes (Klein, 2004; Maedche et al., 2002; Noy et al., 2006, 2004; Rogozan and Paquette, 2005), implementing them and keeping track of the various versions of the ontology (Klein and Fensel, 2001; Obst and Chan, 2005).

With the cycle in place, we identify that using online ontologies as background knowledge can clearly help at the level of *suggesting changes* and *validating changes* tasks. This will form the main focus of our work, targeting our research questions.

2.2 Ontology Evolution Frameworks

Various works in the field include an ontology evolution. Table 2.1 maps our evolution cycle, the frameworks that we discuss next in this section.

Table 2.1: Relations between tasks of the ontology evolution cycle and components of existing ontology evolution framework

Ref. Work	**Detecting Need for Evolution**	**Suggesting Changes**	**Validating Changes**	**Assessing Impact**	**Managing Changes**
Stojanovic (2004)	Change capturing	Representation	Semantics of change/Valid.	Propagation	Implementation of changes
Klein and Noy (2003)		Data transformation	Consistency Verif./approval		Update
Noy et al. (2006)			Examining changes	Auditing	Accept/reject Recording
Vrandecic et al. (2005)	Local changes		Revision		Local adaptation

In her thesis (Stojanovic, 2004), Stojanovic proposed a framework for ontology evolution. The framework is a six phase cyclic process starting with the *change capturing* phase where changes to be applied to the ontology are identified. This is followed by the *representation phase* where the changes are represented following a specific model that the author name as the "evolution ontology". The third phase is the *semantics of change* phase, during which syntactic and semantic inconsistencies that could arise as a result of the changes are addressed (Tamma and Bench-Capon, 2001). A syntactic inconsistency covers cases such as violating constraints or using entities and concepts that have not been defined in the ontology. A semantic inconsistency is when an entity's meaning changes during the evolution process (Tamma and Bench-Capon, 2001). The fourth phase is the *implementation of change* phase coupled with user interaction for approving or cancelling changes. *Change propagation* is the fifth phase, allowing the update of outdated instances as well as recursively reflecting changes in referenced ontologies in the case of interconnected ontologies. The final phase is the *validation phase* where it is insured that the performed changes led to a valid (or desirable) result, and allows the user to undo such changes if this is not the case.

As discussed later in this chapter, this approach defines what is called *usage-driven* change discovery (i.e., derived from user behaviour), *data-driven* discovery (i.e., derived from the ontology's instances) and *structure-driven* change discovery where changes are derived from the analysis on the ontology's structure. Hence we observe that this evolution framework treats the ontology as a closed entity by initiating the evolution from the analysis performed on the ontology itself, without opening it to external domain data that exist in the form of text corpora or other form of data repositories.

In (Klein and Noy, 2003), the authors present a framework to support users when an ontology evolves from one version to another. The framework is component-based, and targets the following ontology evolution tasks: *Data transformation*, where data in the old ontology version are transformed into a format compatible with the new ontology version; *ontology update* where changes are propagated to the ontology under evolution; the third task deals with *consistent reasoning* to keep the ontology under evolution consistent; the final task is *verification and approval* where ontology developers perform final checks. The focus in this approach is mainly on the versioning side of the ontology, as an effect of the evolution.

(Noy et al., 2006) describe a framework for ontology evolution in collaborative environments. The framework is scenario-based and consists of various Protégé[1] plugins. It includes the following tasks: *examining changes* between ontology versions, presented for example in the form of a table; *accepting and rejecting changes* helpful in curated ontology evolution, where changes are approved or rejected with the change action recorded; *providing auditing*

[1] `http://protege.stanford.edu`

information where authors' information (e.g., time of change, number of concepts changed) are compiled. The framework in this case serves as a means to manage collaborative changes to be performed on an ontology, where the source of changes come from the ontology curators.

DILIGENT (Vrandecic et al., 2005) is a decentralised user-centric methodology proposing an ontology engineering process targeting "user-driven" ontology evolution, rather than its initial design. At a glance, the process starts by having a *core ontology collaboratively built* by users. After the building step, the ontology will be locally adapted without changing the core ontology. A board of users will then *analyse the local changes*, in order to come up with the changes that need to be incorporated in the shared ontology. The requests of changes are supported by arguments using an argumentation framework in order to come up with a balanced decision reflecting all the evolution requests. The *changes are revised* by the board of knowledge experts in order to maintain compatibility between different versions. The evolution of the ontology is a result of the decided changes to apply. Finally the shared evolved ontology is locally adapted at the different involved locations.

2.3 Ontology Evolution Tasks

In this section we discuss the approaches that deal with the core ontology evolution tasks derived from the cycle in Figure 2.1. Based on our research focus, we place special attention on the tasks of (1) detecting the need of change from data sources, (2) suggesting changes by relying on structured knowledge and (3) validating changes by processing domain information.

2.3.1 Detecting the Need for Evolution

Ontology evolution starts with the detection of the need for evolution. Being used in various scenarios, an ontology can become obsolete due to changes that occur in the surrounding environment. This can be caused for example by changes in the underlying data (e.g., entities represented within the ontology, or external domain data), or changes in usage patterns (e.g., related to changes in user or application needs).

2.3.1.1 Detecting the Need for Evolution from Data

The need for evolution can be initiated from the analysis of existing data. While some approaches limit the data analysis to information available within the ontology, for example in (Stojanovic, 2004), other tools identify ontology changes by analysing external data sources including unstructured sources e.g. text documents (Bloehdorn et al., 2006; Cimiano and Volker, 2005; Maynard et al., 2009; Novacek et al., 2007; Ottens et al., 2007; Velardi et al., 2001) and metadata (Maynard et al., 2007), or structured data e.g. databases (Haase and Sure, 2004). (Stojanovic, 2004) defines the *data-driven* ontology evolution as the process of discovering ontology changes based on the analysis of the ontology's instances by relying for example on data mining techniques. Another type of change detection defined by Stojanovic is *structure-driven*, where the evolution is initiated based on the analysis performed on the ontology structure using a set of heuristics. For example, “if all subconcepts have the same property, the property may be moved to the parent concept”, or “a concept with a single subconcept should be merged with its subconcept” (Stojanovic, 2004). This type of evolution is also referred to as *bottom-up* ontology evolution (Stojanovic et al., 2002).

The second data source for detecting the need for evolution can exist in the form of domain data, external to the ontology under evolution. Such more “traditional” form of storing information about the domain, often contain valuable knowledge that should be encapsulated in the ontology itself. With respect to our work, we rely on such source of information to identify new domain concepts that can potentially be added to the ontology. (Bloehdorn et al., 2006) based their work on the six phases ontology evolution process proposed by (Stojanovic, 2004). The authors identify that valuable information reside in databases and documents, but require better structuring and easy accessibility through the use of ontologies. Unlike (Stojanovic, 2004) who considers data-driven ontology evolution as the evolution triggered from the ontology's instances, they consider *data-driven changes* as changes happening in external data sources such as the addition and deletion of documents in a corpus, as well as changes occurring in databases (Bloehdorn et al., 2006; Haase and Sure, 2004). Other tools that initiate ontology changes from text documents include the ontology learning tools Text2Onto (Cimiano and Volker, 2005) and SPRAT (Semantic Pattern Recognition and Annotation Tool) (Maynard et al., 2009). Moreover, DINO (Laera et al., 2008; Novacek et al., 2007) is a framework for integrating ontologies which are learned from text, and Dynamo (Ottens et al., 2009) is a multi-agent system based approach that falls in this category of tools as well. We discuss in more details the processes involved within these tools in the next sections.

2.3.1.2 Detecting the Need for Evolution from Usage

In addition to using data analysis as a starting point for detecting the need for evolution, some approaches rely on the study of usage patterns to which the ontology is subject to. For example (Alani et al., 2006) propose that, based on what parts of the ontology are mostly used by applications, the ontology can shrink to better fit its purpose in the environment. In addition to application usage, user behaviour is studied to detect the need for evolution, which is called *usage-driven* ontology evolution (Stojanovic, 2004). In (Bloehdorn et al., 2006), a usage-log, which is a record kept of the interaction between the user and the ontology (e.g., user behaviour and contextual search history), is used to analyse and detect the need for evolution. Such log can store information about what has been queried, which elements in the ontology have been viewed by the user, etc, and used to derive usage preferences and suggest changes to the ontology.

2.3.2 Suggesting Changes

The task of detecting the need for ontology evolution is followed by suggesting the required changes. Various approaches derive changes by limiting their focus on the content available in unstructured documents (e.g., text documents). Other works rely on external structured knowledge sources (e.g., usage logs, lexical databases or online ontologies) to support ontology change suggestions. In this section, we discuss those two sets of approaches, by relating to the existing works performed in each area.

2.3.2.1 Suggesting Changes by Relying on Unstructured Knowledge

Text2Onto (Cimiano and Volker, 2005) derives ontology changes through processing text documents and extracting ontological entities. It is designed to overcome the limitations of existing tools in terms of domain dependency, lack of user interaction during the ontology learning process and running the learning process from scratch whenever a change occurs in the corpus. Text2Onto uses a Probabilistic Ontology Model (POM), proposing a model with a degree of certainty attached. This is coupled with data-driven change discovery that enables specific changes detection without processing all the documents again. In addition to the extraction of concepts and instances, Text2Onto includes algorithms to extract various types of relations including "Instance-of", "Subclass-of", "Part-of" and other general relations. Such relations are inferred through a set of predefined patterns. This tool relies on the GATE framework (Cunningham et al., 2002) to process the text documents.

Similar to Text2Onto, SPRAT identifies ontology changes from text documents (Maynard et al., 2009). It combines existing ontology based information extraction (OBIE) techniques, named entity recognition and relation extraction from text. It provides additional patterns to refine the process of entity identification and relations between them, and to transform them into ontological entities. SPRAT relies on lexico-syntactic patterns applied on text documents to identify terms and their corresponding relations.

Furthermore, (Bloehdorn et al., 2006) propose an architecture applied in a digital library domain or other electronic repositories. They make use of ontology learning algorithms to extract document contents.

Another tool developed with the aim to detect changes from text documents and merging ontologies is DINO (Laera et al., 2008; Novacek et al., 2007). DINO is a framework for integrating ontologies. Part of its processes includes a semi automatic integration of learned ontologies with a master ontology built by ontology designers. It relies on the use of ontology alignment, coupled with agent-negotiation techniques, to generate and select mappings between learned ontologies from text and the base ontology. In more details, Text2Onto is used to extract information from documents in the DINO framework. The learning algorithms of Text2Onto are customised through a user interface, and the confidence values of extracted terms are fed to an ontology alignment/negotiator wrapper (Novacek et al., 2007). The learned ontology representing new concepts, and the master ontology collaboratively developed by the knowledge experts are aligned, i.e. a set of mappings between the classes, entities and relations of the two ontologies are set using the alignment wrapper. The agreement of the semantics used is reached through negotiation using the negotiation wrapper. An axiom ontology, which contains the merging statements between the learned and the master ontology is created.

Dynamo (Ottens et al., 2009) is another tool that falls in the category of exploiting external data sources for building ontologies. It consists of a multi-agent system for dynamic ontology construction from domain specific set of text documents. Dynamo relies on an adaptive multi-agent system architecture, within a framework where the ontology designer interacts with the system during the process of building the ontology. The system considers the extracted entities from text sources as separate agents, which are related to other entities (agents) through a certain relationship. In other words, an ontology is treated as a multi-agent system.

2.3.2.2 Suggesting Changes by Relying on Structured Knowledge

Structured knowledge represents and defines conceptual information entities, connected through explicit relations. Such representation allows reusing knowledge entities with less effort compared to the unstructured knowledge sources. Structured sources can exist in different formats, including lexical databases and online ontologies. In this section we present approaches that make use of both, as support for suggesting ontology changes during ontology evolution.

In (Maedche et al., 2002), the authors propose the use of lexical databases, e.g., WordNet (Fellbaum, 1998), to improve semantic bridging and similarity computation. WordNet is one of the major lexical databases used in the literature and provides a wealth of entities interconnected with subsumption links represented in the form of hyponyms and hypernyms, in addition to other types of relations including meronymy and holonymy links. WordNet is used to support various tasks including word sense disambiguation (Banerjee and Pedersen, 2002; Ide and Vronis, 1998; Li et al., 1995), information retrieval (Li et al., 1995) and question answering (Clark et al., 2008; Pasca and Harabagiu, 2001). In addition to that, SPRAT (Maynard et al., 2009) and Text2Onto (Cimiano and Volker, 2005), which are textual pattern based tools that support ontology learning, propose using WordNet as a means to improve finding connections with their pattern extraction mechanism.

Online ontologies form a ready to reuse body of knowledge. They are used to perform a variety of tasks including for example ontology matching (Sabou et al., 2008) and development (Alani, 2006), question answering (Lopez et al., 2009), folksonomy enrichment (Angeletou et al., 2008) and word sense disambiguation (Gracia et al., 2009). This is mainly due to the increase of the availability of online ontologies and the presence of tools, such as Watson (d'Aquin and Motta, 2011), Swoogle (Ding et al., 2005) and Sindice (Oren et al., 2008) that help in discovering and consuming them.

In (Alani, 2006), some initial ideas for a methodology to build an ontology by reusing online ontologies are proposed. The author describes the steps needed within the process, and the required functionalities that would help achieving the target of building the ontology. The process starts with the ontology engineer putting a list of terms that need to be in the ontology. Then, based on such terms, a search mechanism (e.g., based on Swoogle (Ding et al., 2005)) is needed to find potential online ontologies that represent the terms. The next proposed step is ontology ranking, which is beneficial in case many ontologies are found online. Ranking can be based on different criteria, e.g., user ratings (Supekar, 2005), evaluation tests (Guarino and Welty, 2002) and concept representation (Alani and Brewster, 2005). This is followed by the segmentation of the selected ontologies, to limit the knowledge to be reused from the ontology. The extracted segments are then compared and merged to form the needed ontology. Furthermore, the author (Alani, 2006) proposes using existing semi-automatic tools, such as the PROMPT suite (Noy and Musen, 2003) and Chimeara (McGuinness et al., 2000), to generate the ontology that should be finally evaluated. In addition to that, the author presents a list of challenges, along with possible solutions, to build such a system and implement the presented ideas: (1) Semantic Web tools available are not mature enough yet, questioning their reliability to be reused in such a system; (2) Not all Semantic Web ontologies generated are made available online. This may affect the availability of online ontologies accessible by the system; (3) large ontologies can provide big segments, resulting in a big messy ontology that is hard to clean by the user; (4) the quality of the online ontologies would affect the generated segments, hence affecting the overall quality of the resulting ontology. Also a segment of a good ontology does not necessarily preserve the quality of the source ontology.

The raised questions are very relevant to our research, as we place the use of online ontologies at the core of our processes for change discovery and evaluation. We intend to show, through the study and evaluations performed in this research and related work (d'Aquin et al., 2008c), the increased maturity and the availability of Semantic Web tools and ontologies that support the process of ontology evolution. In particular, we focus in our research on two solutions, which provide mechanisms to access and process online ontologies:

Watson [2] (d'Aquin and Motta, 2011) is a gateway to online Semantic Web ontologies. It (1) gathers semantic content available on the Web, (2) processes the content to identify indexes and useful metadata, and (3) implements queries to efficiently extract the collected data. Watson provides two core functionalities for ontology consumers: (1) a customisable search interface for end-users to search and locate online ontologies based on user specified keywords; (2) a set of APIs that make processing online ontologies possible directly within applications, for example to extract on the fly specific metadata, content or metrics from the online bodies of knowledge, or execute SPARQL queries directly from within the API. Such APIs have proved to be useful to build various Semantic Web based applications (d'Aquin et al., 2008c), including Scarlet (Sabou et al., 2008) that we discuss next. For our relevance assessment scenario in Chapter 4, we make use of the API for example to locate a concept in a specific online ontology, and implement a technique to extract the neighbouring concepts and relations up to certain depth (which will be used as part of the context of statements to compute the relevance). Implementation details used in our case are presented in further details in Chapter 5.

[2]http://watson.kmi.open.ac.uk

Scarlet [3] (Sabou et al., 2008) relies on the Watson API to explore online ontologies, with the aim to identify potential relations between two concepts. The authors argue that the growth of the Semantic Web will directly benefit the task of ontology matching, as ontologies, in contrast to databases, provide explicit semantics and vocabularies between entities, which increase the performance of matching approaches (Sabou et al., 2008). The proposed matching technique identifies, at run time, the available online ontologies that provide information (e.g., using their labels or URI namespace) about the two terms, with potential relations that connect them. Two strategies are provided through this technique. The first strategy exploits the mappings from one ontology, while the second provides a mechanism to identify mappings spread across several ontologies. In our implementation and testings, as mentioned in Chapter 3, we focus on the relations between two concepts identified from one ontology. The first step of the matching process is *Anchoring*, to identify the online ontology where the concepts to relate occur. The second step involves the *Ontology Selection* to select the ontology in case multiple ontologies provide mapping information about the two concepts. The third step is the *Derivation Rules*, to set whether only direct relations are enough, or indirect with inference ones are preferred. The fourth

[3]http://scarlet.open.ac.uk

and final step is *Combining Mappings*, where a resolution of the cases where multiple or contradictory mappings are identified between the two concepts. We test and rely on Scarlet to identify statements that link new domain concepts to existing concepts in the ontology, leading to the identification of new changes to apply to the ontology. We present our results and analysis in Chapter 3.

2.3.3 Validating Changes

Detected changes to be added to the ontology are not always valid with respect to the ontology. The task of validating ontology changes is crucial for preventing the evolution of the ontology to go into undesirable directions. Changes can be validated using domain-based and formal-based techniques. The first relies on existing domain data to evaluate whether the change aligns with the existing content of the ontology, in order to check whether the change is important to be performed. The latter makes sure that the change does not invalidate specified constraints, e.g., in terms of consistency or coherence, using formal techniques.

2.3.3.1 Validating Changes Based on Domain Information

Domain-based ontology changes validation uses existing domain data to evaluate changes. This includes for example analysing text documents to check the occurrences of a term to add to the ontology as a concept or instance. Text2Onto (Cimiano and Volker, 2005) assigns a degree of certainty about the learning process making it easier for the user to interact. For that, it relies on the statistical measures (e.g., TF-IDF) to derive the confidence of a *term* with respect to the corpus. SPRAT (Maynard et al., 2009) also uses TD-IDF to give an overview of the importance of concepts and instances proposed to be added to the ontology or used to create it. This has a downside as such statistical measures are dependent on the size of the corpus. Moreover, the assessment takes into account the number of occurrence of entities with respect to the corpus, without taking the ontology into consideration.

In DINO (Novacek et al., 2007), the changes are sorted according to a relevance measure, leaving the choice of only presenting highly relevant changes to the users. Even though the authors present the need for a relevance measure of triples, the relevance relies on a string similarity measure between the entities in a triple, and a set of wanted or unwanted words specified by users. The drawback of this approach is that (1) it is expected that users manually create a list of words that reflect relevance, making it a tedious process to maintain, and (2) it does not take the ontology into consideration to check the relevance. In other words, this approach is purely based on matching the labels in the triple with the user defined sets, without performing any structural content analysis that the triple brings to the ontology.

Additional work has been done in checking the correctness of ontology statements. One approach measures the level of agreement and disagreement within online ontologies on how to represent specific statements (d'Aquin, 2009). Another work checks the correctness of statements on the Semantic Web, allowing the prediction of how two concepts should be correctly linked (Sabou et al., 2009). Such work would be of valuable input in validating changes in ontology evolution, as it could filter out those relations that are invalid and should not be added to the ontology in the first place.

In our work, we identify that online ontologies can provide structured context in which a change is used. We show in Chapter 4 that when such a context is compared to the ontology under evolution, a relevance value reflecting the importance of a change with respect to the ontology can be deduced. Our experiments show that pattern-based techniques, perform better than overlap-based context comparisons.

2.3.3.2 Validating Changes using Formal Properties Methods

In this part we give a brief overview of existing approaches for validating ontology changes using formal properties. (Flouris et al., 2006) propose a framework for dealing with inconsistencies and ontology changes. Different levels of inconsistencies are identified within DL-based ontologies, and a formal theory of ontology change is studied within this work. Another relevant work in this category is performed by (Haase et al., 2005), where the authors focus on handling inconsistencies of ontologies under evolution. A formal study is conducted to compare four approaches in handling inconsistent ontologies: (1) consistent ontology evolution (Haase and Stojanovic, 2005), where ontology changes are managed by fulfilling consistency conditions; (2) repairing inconsistencies, where inconsistencies are first detected, and subsequently repaired (Schlobach and Cornet, 2003); (3) reasoning in the presence of inconsistencies, which does not aim to repair inconsistencies, but keep reasoning and querying possible, even with the presence of inconsistencies (Huang et al., 2005); and (4) multi-version reasoning, where relations between different versions of an ontology are managed, and a study of compatibility, including inconsistency are dealt with (Huang and Stuckenschmidt, 2005). In (Haase et al., 2005), a framework is proposed to combine the four approaches.

In addition to the theoretical work, different tools have emerged to handle ontology validation based on formal properties. For example RaDON (Ji et al., 2009) is a tool for ontology consistency checking. It uses reasoning mechanisms to detect inconsistencies within an ontology or a network of ontologies, and supports repairing algorithms to come up with suggestions for resolving such inconsistencies. Pellet (Sirin et al., 2007) and Jena (Carroll et al., 2004) provide reasoning functionalities for inconsistency detection over ontologies. Jena is proposed to be used in DINO's reasoning and management wrapper (Novacek et al., 2007). Such tools are valuable to the ontology evolution community, as they provide out of the box solutions to be reused in environments where ontologies are under constant changes, with a risk of becoming inconsistent.

2.3.4 Assessing the Impact of the Evolution

Following a change, an important task is to assess the impact of the evolution that resulted from this change. This task is responsible for measuring the effect of an evolution from the application and usage perspectives, as well as from the formal criteria perspective. The impact on application and usage determines whether the evolution would have an effect on the systems that rely on the ontology to perform their process; the formal criteria aim to give a quantifiable measure of the impact of a change by using formal properties as the basis of the approach.

2.3.4.1 Assessing the Impact Based on Application and Usage

While ontology evolution is crucial, one of its main objectives is to keep its usage within applications possible. For that, assessing the impact of the evolution of an ontology on its dependent applications is important. This has been identified by (Liang et al., 2006), where one of the research questions is about "which methods are required to maintain the services of the deployed applications while updating the underlying ontologies?" The proposed approach is to make use of a log ontology to store and manage changes, and a prototype system that modifies incoming queries to the ontology based on the logged changes to ensure that queries would remain answerable.

(Lei et al., 2006) target the issue of acquiring and keeping semantic metadata in domains up-to-date. As part of this proposed approach, they create a set of mappings to link knowledge entities in data sources, to the domain ontology. This makes the domain ontology easily updatable whenever entities discovered from knowledge sources are mapped to ontology. However the main drawback in this approach lies at the level of the maintenance of the mappings, which should be manually updated and created.

(Kondylakis et al., 2009) present an overview of existing work on ontology evolution in the data integration field. They identify that existing approaches do not take the evolution of mappings between the data sources and the ontology into account, when the corresponding ontology is highly dynamic. They argue that for the data integration application environment, they need a system that fulfils specific requirements. Some of the requirements relevant here are that (1) the system has to deal with changes at the ontology level, (2) data queries should relate to the evolution of the ontology, and (3) experts should be able to validate the mappings.

(Wang et al., 2008) highlight the importance of managing the dependency between the ontology and its dependent applications. This would help in preventing the occurrence of inconsistencies between the ontology and its applications, when the ontology evolves. They propose a systematic way of propagating an ontology change on its application. In this case, they consider that the impact on the application is dependent on the range of ontology actions. (Wang et al., 2008) also propose having a registration server mechanism to check the impact on the information system environment whenever an ontology change occurs.

2.3.4.2 Assessing the Impact based on Formal Criteria

Evaluating the impact of changes on ontologies has been evaluated in terms of assertional effects (Pammer et al., 2009). Assertional effects measure what is gained or lost after performing an ontology change. This work is meant to aid the user to have a quick overview of a change impact, in order to make a decision about whether the change should be applied or not, while preserving conceptual consistency. The work formally describes the assertional effects, and an implementation is supplied as a support for the users during ontology development (Pammer et al., 2010).

Another approach proposes the evaluation of changes in ontology evolution using an impact function, which computes the cost involved in performing the change (Palmisano et al., 2008). This cost is aimed for agents using and changing the ontology, to make a better decision whether to apply the change or not. The authors propose an approach to compute such costs without the use of reasoning, but by identifying the parts of the ontology that will be affected as a result of the change. The impact takes into consideration the number of axioms involved in the change, and the expressivity of the parts.

In (Haase and Stojanovic, 2005), the authors present the notion of *minimal impact*, a concept dependent on user requirements. The idea is based on selecting and implementing the minimum number of ontology changes, that result in a “maximal consistent subontology”. The authors define the concept of maximal consistent subontology, as the part of the ontology to which you can not add any axiom, without loosing its consistency.

2.3.5 Managing Changes

Modifying and evolving an ontology raise the need for managing the changes applied. Managing changes involves the tasks of recording the ontology changes, as well as generating and managing various ontology versions. This would help for example in retracting the ontology to a previous version when needed, or tracing back the origin of the source of entities within the ontology, as well as helping in scenarios where the ontology is built collaboratively.

2.3.5.1 Recording Changes

Tracing ontology changes forms a substantial part in the research dealing with ontology evolution and versioning. Ontologies should not be treated as text documents while tracing their evolution, since structural and semantic changes are the important components to track. Recording changes requires a clear and well defined representation of ontology changes. For that, (Klein, 2004) defines what is called an "ontology of change operations", where a set of predefined ontology changes are represented in a unified way[4]. This helps is applying changes consistently in different scenarios. This ontology forms a core element in the evolution framework described in (Klein, 2004).

Similarly, (Stojanovic, 2004) proposes an "evolution ontology", which is a meta-ontology for representing different types of changes that can be performed on an ontology. It is created in order to unify the representation of changes across the evolution environment. Elementary and composite changes are defined. The core concept in the evolution ontology is the "Change" concept. For example, adding a concept to an ontology is represented in the ontology as a sub-class of the concept "Change" as follows:

$AddConcept \sqsubseteq AdditiveChange \sqsubseteq ElementaryChange \sqsubseteq Change$

The evolution ontology represents dependencies using the "causesChange" property, which enables to reverse the effects of the applied changes, and going

[4]The complete list of change operations can be found in the Appendix B of (Klein, 2004).

back to a previous state of the ontology.

A core feature of the framework presented by (Noy et al., 2006) is a change and annotation ontology (ChAO). This ontology enables tracking synchronous and asynchronous changes. Whenever a change is performed on the ontology using the framework tools (i.e., Protégé and the change management plugins), an instance is created in ChAO to represent the change and its meta information including for example who preformed the change, the time the change was applied, and other relevant data. While if the ontology has evolved using other means and tools, the instances of ChAO are generated through a structural *diff* between the two ontology versions. Recording such changes provides users with the ability to accept, reject or revise a change through the tool's interface.

Moreover, it is proposed having an evolution log where all changes happening to an ontology are recorded (Liang et al., 2006; Stojanovic, 2004). This could make tracing and rolling back changes easier. Ontologging (Maedche et al., 2002) is another example of evolution tracer tools. It also analyses the effects of changes performed on the ontology. Databases have been as well used in managing ontology evolutions, by storing in tables the changes of the ontology itself or the ontology's metadata level (Ceravolo et al., 2004). OntoAnalyzer is another tool that traces complex ontological changes (Rogozan and Paquette, 2005). The authors state that OntoAnalyzer has an advantage over the KAON framework (Volz et al., 2003), which is able to handle only elementary changes. Comparing it to PromptDiff (Noy et al., 2004) and OntoView (Klein, Kiryakov, Ognyanov and Fensel, 2002), which identify the changes in different ontology versions without taking complex changes into account, OntoAnalyzer allows the tracing of complex changes by keeping a log of the operations performed on the ontology. In Text2Onto (Cimiano and Volker, 2005), a pointer is used to keep track of the data changes with their provenance from the text, giving users the ability to check the source of changes.

2.3.5.2 Versioning

(Klein and Fensel, 2001) define ontology versioning as "the ability to handle changes in ontologies by creating and managing different variants of it". One of the aims is to the keep knowledge transfer interoperable among different ontology versions (Klein, Fensel, Kiryakov and Ognyanov, 2002). (Klein, 2004) targets the management side of distributed ontologies. The author initiates some of the versioning ideas by drawing comparisons to versioning work done in the database field, based on which a wish list to be applied on versioning Semantic Web ontologies is deduced. He stated that dealing with ontology evolution raises the support of: Transforming data from the old ontology version to the new one; continuous data access even if the data has not yet been transferred from the old to new ontology version; propagating the changes to remote ontologies; consistency between ontologies versions; and lastly users verification and approval (Klein, 2004). This helped in creating a component-based framework discussed in Section 2.2. Part of the outcome of this work includes OntoView (mentioned previously), which enables to store various ontology versions on the web, and helps users in detecting relations between different ontology versions (Klein, Fensel, Kiryakov and Ognyanov, 2002).

PromptDiff (Noy et al., 2004) is another ontology-versioning tool supporting knowledge engineers in collaborative development environments. It tracks structural changes in ontologies and flags as well if a mapping should be updated when one of the mapped ontologies has changed. PromptDiff has an API to hook external applications for comparing ontologies, with the ability for the ontology editor to accept or reject the changes.

(Obst and Chan, 2005) propose a generic ontology versioning framework. It draws on the idea of the minor/major ontology versions proposed in (Klein and Fensel, 2001), to fulfil what they call the *monotonicity* property, which ensures that the conceptualisations of versioned ontologies are not removed. Additional component properties are proposed, which serve as a starting point towards an object-oriented approach for building the versioning framework.

2.4 Discussion

We presented in this chapter an ontology evolution cycle, covering the tasks involved in the evolution process. The literature review conducted in our work shows the extensive research done at the various levels of ontology evolution. With this in place, we identify two major gaps in the area of ontology evolution.

Firstly, having analysed some of the tools that aim to help users in the process of ontology evolution, we realise that most of the approaches limit the integration of new knowledge entities from external sources to the information contained in the data source itself (e.g., text documents). Even though some approaches, for example Text2Onto (Cimiano and Volker, 2005) and SPRAT (Maynard et al., 2009), mention the use of WordNet for additional information, their main source of relations (particularly named relations) between concepts is based on the lexical syntactic patterns found in texts (in the case where a corpus is used to evolve the ontology), without reusing existing knowledge available in the form of online ontologies.

Secondly, another gap in the literature lies at the level of ontology change evaluation. Existing techniques for change evaluation are generally limited to checking the effect of the change in terms of consistency and impact. This is usually done through formal analysis and by measuring the cost of a change, as well as checking the logical consistency. However assessing the importance and relevance of a change to the ontology is not taken into account. While statistical techniques that measure the importance of terms with respect to a corpus of documents are proposed using for example TF-IDF, the ontology in this case is not taken into consideration.

We propose in our research to use online ontologies as background knowledge to decrease user input and time during the process of ontology evolution at two levels: first, online ontologies can help in discovering relations that link new entities identified from domain data to existing entities in the ontology under evolution (discussed in Chapter 3); second, they provide key elements (i.e., in terms of structure that we use as a context) to make the assessment of the relevance of statements with respect to the ontology in focus possible (discussed in Chapter 4).

Chapter 3

Knowledge Reuse for Ontology Change Discovery

In this chapter we focus on the task of supporting the user in identifying ontology changes based on new emerging domain entities. This is the problem highlighted through our first research question: *How to assist users in identifying ontology change opportunities?* As mentioned in Chapter 2, new domain entities can be identified from existing domain data such as text documents. The discovery of such new entities triggers the need for evolution based on data (see Section 2.3.1.1). However, the specific problem we are solving here lies at the level of the generation of the appropriate ontology changes (i.e., the task of *suggesting changes*, see Chapter 2), to integrate the newly identified domain elements in the ontology under evolution, with the appropriate connections to its existing entities.

To achieve this task, we propose the reuse of available knowledge to identify new ontology changes, by discovering relations between new terms identified in external domain data (e.g., text corpus or list of terms) on one side, and, on the other side, existing concepts in the ontology. In this chapter, we exploit WordNet and online Semantic Web ontologies as background knowledge to resolve the possible relations. We perform an experiment within the academic context, in which we use news articles from the Knowledge Media Institute[1], to evolve the AKT reference ontology[2]. We analyse and discuss the outcomes of the experiments, showing the feasibility of the approach and pointing out possible ways to better exploit external sources of knowledge to support the ontology evolution process.

3.1 Background Knowledge Availability

We have identified several potential sources of background knowledge to be used in our context. For example, **lexical databases** such as WordNet have been long used as a reference resource for establishing relations (e.g., subclass or synonymy) between concepts. Because WordNet's dictionary can be downloaded and accessed locally by the system and because a variety of re-

[1]`http://news.kmi.open.ac.uk`

[2]`http://www.aktors.org/publications/ontology`

lation discovery techniques have been proposed and optimised, exploring this resource is generally fast. However some of the disadvantages of WordNet are that firstly it mainly provides sub-class relations. Secondly, the evolution of the content of WordNet is slow, compared to information made available through online ontologies that can be created and published by entities around the world.

Online ontologies constitute another source of background knowledge which has been recently explored to support various tasks such as ontology matching (Sabou et al., 2008) and development (Alani, 2006). Unlike WordNet, online ontologies provide a richer source of relations, where, in addition to subsumption relations, named relations can also be derived. Furthermore, given the fact that more ontologies are constantly made available online, this body of knowledge would expand at a faster rate than WordNet.

Finally, the **web** itself has been recognised as a vast source of information that can be exploited for relation discovery through the use of so-called lexico-syntactic patterns (Cimiano et al., 2004; Hearst, 1992). Because they rely on unstructured, textual sources, these techniques are more likely to introduce noise than the previously mentioned techniques that rely on already formalised knowledge. Additionally, these techniques are time consuming given that they operate at web scale.

It would be good to exploit and test all three sources of background knowledge including the unstructured web documents. However we focus in our experiment on using WordNet and online ontologies. This is due to two main reasons: firstly, the web involves challenges at a different scale, such as processing unstructured online text documents to identify connections at the entities' level, which are out of the scope of our research interests. Secondly, we identify a potential key contribution provided at the level of reusing the **structure** provided by the other sources (mainly through online ontologies), to evaluate the identified relations (see Chapter 4).

3.2 Exploiting Structured Information Sources for Change Discovery

The change discovery process starts by having a list of terms that can be identified from domain data. We use text documents in this experiment to extract potential terms to add to the ontology. This can be achieved by using any existing named entity recognition technique. For this experiment, we reused the one provided by the Text2Onto implementation (Cimiano and Volker, 2005). In this scenario, we focus on adding new elements at the schema level of the ontology under evolution. In other words, any term that already appears in the ontology under evolution as a concept should be ignored. For that we created a mechanism to detect existing terms to be filtered out, in order to generate the list of new potential terms (e.g., [Researcher, Tutorial] in Figure 1.2).

The list of new terms is used, along with the concepts available in the ontology under evolution (e.g., [Thing, Event, Person, Conference, Workshop] in Figure 1.2), to identify potential relations between them. Each new term is checked against all existing concepts in the ontology to find possible connections through WordNet and online ontologies (e.g., $< Tutorial, subClass, Event >$ identified from an online ontology[3]). These connections are the ones which are potentially transformed into ontological changes to apply and evolve the ontology.

In the remainder of this section, we present the techniques we use for our experiment to process WordNet and online ontologies, to discover relations between the new terms and existing ontology entities.

3.2.1 Using WordNet for Relation Discovery

In this experiment, we first used WordNet as the source of relation discovery between new terms and the ontological entities. We processed WordNet by first identifying the closest entity in the ontology to the newly identified term, with respect to similarity. Then we take the closest similar terms and find potential relations between them. To compute the similarity between terms, we used the Wu and Palmer similarity (Wu and Palmer, 1994) measure. This

[3]Example of an online ontology containing this relation: `http://www.ifi.unizh.ch/ddis/fileadmin/pdf/service_broker/iswc.daml`

measure is computed according to the following formula:

$$Sim(C1, C2) = \frac{2*N3}{N1+N2+2*N3}$$

where $C1$ and $C2$ are the concepts to check for similarity, $N1$ is the number of nodes on the path from $C1$ to the least common superconcept ($C3$) of $C1$ and $C2$, $N2$ is the number of nodes between $C2$ and $C3$, and $N3$ is the number of nodes on the path from $C3$ to the root (Wu and Palmer, 1994). For those terms that are most closely related to each other, we derive a relation by exploring WordNet's hierarchy. This will result in a relation between the terms, as well as an inference path which lead to its discovery. The relation path in our case is the list of relations that connect two concepts, taking into consideration the intermediary nodes that exist in between.

Applying this measure on the new term *Researcher*, against the existing concepts *Thing, Event, Person, Conference, Workshop*, from our example in Figure 1.2, we get: $Sim(Researcher, Person) = 0.83$, while all the remaining similarities are of value zero. Based on this outcome, the connection between *Researcher* and *Person* is checked, leading to $< Researcher, sub-Class, Person >$ as a potential relation to use for integrating *Researcher* in the ontology.

3.2.2 Using Online Ontologies for Relation Discovery

The new terms that could not be related to the ontology through WordNet are processed through the use of online ontologies. As briefly introduced in Chapter 1 and 2, we rely on the Scarlet relation discovery engine[4] to derive relations from online ontologies. It is worth to note that we handle ontologies at the level of individual statements, rather than as complete models. Thus we focus on knowledge reuse without taking care of the validation of the sources as a whole with respect to the base ontology. This would leave room for processing, even if partially, ontologies containing statements that contradict statements in the base ontology. Scarlet uses the Semantic Web gateway Watson (d'Aquin and Motta, 2011), and automatically selects and explores online ontologies *to discover relations between two given concepts*. For example, when relating the two previously mentioned concepts labeled *Tutorial* and *Event*, Scarlet 1) identifies online ontologies that can provide information about how these two concepts inter-relate and then 2) combines this information to infer their relation. (Sabou et al., 2008) describe two increasingly sophisticated strategies to identify and to exploit online ontologies for relation discovery. Hereby, we rely on the first strategy that derives a relation between two concepts if this relation is defined within a single online ontology, e.g., stating that $< Tutorial, subClass, Event >$. Besides subsumption rela-

[4]`http://scarlet.open.ac.uk`

tions, Scarlet is also able to identify disjoint and named relations. All relations are obtained by using derivation rules which explore not only direct relations but also relations deduced by applying subsumption reasoning within a given ontology.

In more details, lets take for example the two concepts *Lecturer* and *Employee*. Scarlet discovers through Watson anchor terms that exist in online ontologies, based on which relations are discovered. In this case, the SWRC[5] ontology provides the appropriate anchors, which are connected through this chain of relations: *Lecturer* $\sqsubseteq$ *AcademicStaff* $\sqsubseteq$ *Employee*, based on which a subsumption relation can be inferred between *Lecturer* and *Employee*. Note, that as in the case of WordNet, the derived relations are accompanied by a path of inferences that lead to them.

Taken from (Sabou et al., 2008), Figure 3.1 depicts the strategy when one ontology is used for relation discovery. In the example, three ontologies are identified (O_1, O_2, O_3), which contain the concepts A' and B' corresponding to A and B. While O_1 does not contain a relation between the anchor terms, O_2 and O_3 provide a subsumption relation between the terms.

In addition to the ability to retrieve direct relations between concepts, Scarlets provides the identification of inferred relations, as the one relating *Lecturer* to *Employee*.

[5]`http://ontobroker.semanticweb.org/ontologies/swrc-onto-2001-12-11.daml`

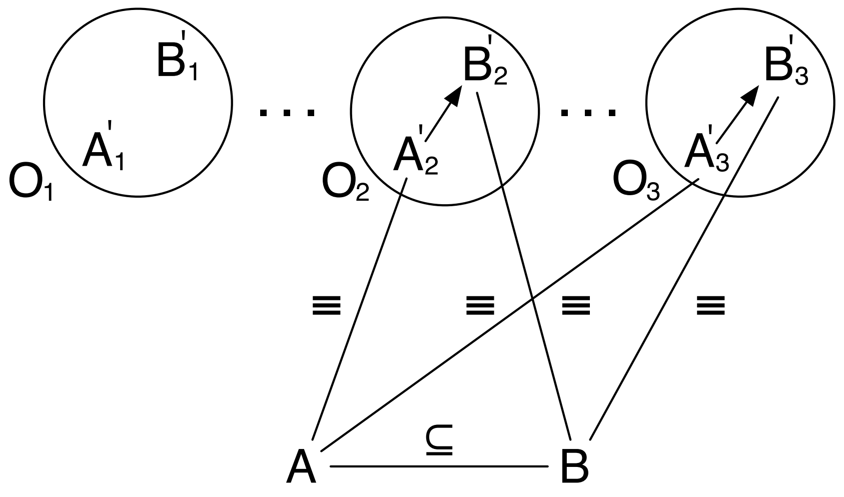


Figure 3.1: Scarlet relation discovery strategy used in our experiment.

3.3 Relation Discovery Evaluation

We performed an experimental evaluation of the proposed relation discovery process, with the goal to answer three main questions. Firstly, we wanted to get an insight into the efficiency, in particular in terms of precision, of the relation discovery relying on our two main background knowledge sources: WordNet and online ontologies. Secondly, we wished to understand the main reasons behind the incorrect relations, leading to ways for identifying these automatically. Tackling these issues would further increase the precision of the identified relations and bring us closer to a full automation of this task. Finally, as a preparation for implementing Evolva's algorithm for performing ontology changes, we also wanted to identify a few typical cases of relations to integrate into the ontology under evolution.

3.3.1 Evaluation Method

To perform our evaluation, we start by collecting the data in the context of the Knowledge Media Institute (KMi) news system[6], to evolve the ontology used to support the KMi Semantic Web portal[7] (i.e., the AKT Reference Ontology[8]). We generate the statements by extracting concepts from KMi news documents, and use WordNet and online ontologies as background knowledge to propose relations that link the extracted concepts, to the existing concept in the ontology.

We pass the collected data for evaluation by three different users working in KMi. The selection was based on their knowledge about the KMi domain, enough the make a decision about the correctness of the statements in the domain. The users were asked to perform the task of an ontology engineers, with a high level of knowledge about the domain and conceptual connections. The guidelines were to, given a set of statements, decide whether the statement is correct by itself, and whether it's useful to be added to the ontology in focus. Users were given the list of statements, with the AKT ontology. The evaluation was conducted independently by each user, and the generated data are analysed to derive the experimental observations and error analysis.

[6]`http://news.kmi.open.ac.uk`

[7]`http://kmi.open.ac.uk/technologies/name/the-kmi-semantic-web`

[8]`http://www.aktors.org/publications/ontology`

3.3.2 Experimental Data

We run our experiment in the academic context of the Knowledge Media Institute news articles to evolve its relevant ontology. We relied on 20 documents from KMi's news repository[9] as a source of potentially new information. We used Text2Onto's concepts extraction algorithm (Cimiano and Volker, 2005) and discovered 520 unique terms in these text documents.

The chosen ontology that we wish to evolve is the AKT Reference Ontology[10] that contains 256 concepts. The first part of the experiment is to perform a string matching between the extracted terms and the existing ontology elements to identify the new terms that do not exist in the ontology. We rely on the Jaro distance metric similarity (Cohen et al., 2003) which takes into account the number and positions of the common characters between a term and an ontology concept label. This string similarity technique performs well on short strings, and offers a way to find a match between strings that are slightly different only because of typos or the use of different naming conventions. The newly discovered terms trigger the need for evolution based on data, the first task in the ontology evolution cycle (see Chapter 2).

By using the Jaro matcher we identified that 21 of the extracted terms have exact correspondences within the base ontology and that 7 are closely matching to some concepts. Closely matching terms means that their Jaro

[9]`http://news.kmi.open.ac.uk`

[10]`http://www.aktors.org/publications/ontology`

Table 3.1: Examples of relations derived by using WordNet.

Extracted Term	**Ontology Concept**	**Relation**	**Relation Path**
Contact	Person	$\sqsubseteq$	contact $\sqsubseteq$ representative $\sqsubseteq$ negotiator $\sqsubseteq$ communicator $\sqsubseteq$ person
Business	Partnership	$\sqsubseteq$	business $\sqsubseteq$ partnership
Child	Person	$\sqsubseteq$	child $\sqsubseteq$ person

similarity coefficient is above the similarity threshold, which we set to 0.92.

3.3.3 Evaluation of the WordNet Based Relation Discovery

Out of the 492 new terms, 162 have been related to concepts of the ontology thanks to the WordNet based relation discovery process. Some of these relations were duplicates as they related the same pair of term and concept through different paths. For evaluation purposes, we eliminated duplicate relations and obtained 413 distinct relations. Table 3.1 shows examples of relations linking extracted terms to ontology concepts, along with the relation types and their path.

We evaluated a sample of randomly selected 205 relations (i.e., half of the total) in three parallel evaluations performed by three evaluators. This man-

Table 3.2: Evaluation results for the relations derived from WordNet.

	Evaluator 1	**Evaluator 2**	**Evaluator 3**	**Agreed by all**
Correct	107	137	132	76
False	96	53	73	26
Don't know	2	15	0	0
Precision	53 %	73 %	65 %	75 %

ual evaluation[11] helped us in identifying those relations which we considered correct or false, as well as those for which we could not decide on a correctness value ("Don't know"). Our results are shown in Table 3.2. We compute a precision value for each evaluator, however, because there was a considerable variation between these, we decided to also compute a precision value on the sample on which they all agreed. Even though, because of the rather high disagreement level between evaluators (more than 50%), we cannot draw a generally valid conclusion from these values. Nevertheless, they already give us an indication that, even in the worst case scenario, more than half of the obtained relations would be correct. Moreover, this experiment helped us to identify typical incorrect relations that could be filtered out automatically. These will be discussed in Section 3.3.6.

[11] To our knowledge, there were no benchmarks of similar experimental data against which our results could be compared.

3.3.4 Evaluation Results for Online Ontologies

The Scarlet based relation discovery processed the 327 terms for which no relation has been found in WordNet. It identified 786 relations of different types (subsumption, disjointness, named relations) for 68 of these terms (see some examples in Table 3.3). Some of these relations were duplicates, as the same relation can often be derived from several online ontologies. Duplicate elimination led to 478 distinct relations.

Table 3.3: Examples of relations discovered using online ontologies.

	Extracted Term	**Ontology Concept**	**Relation**	**Relation Path**
1	Funding	Grant	$\sqsubseteq$	funding $\sqsubseteq$ grant
2	Region	Event	occurredIn	region $\sqsubseteq$ place ←occurredIn- event
3	Hour	Duration	$\sqsubseteq$	hour $\sqsubseteq$ duration
4	Broker	Person	isOccupationOf	broker -isOccupationOf→ person
5	Lecturer	Book	editor	lecturer $\sqsubseteq$ academicStaff $\sqsubseteq$ employee $\sqsubseteq$ person←editor-book
6	Innovation	Event	$\sqsubseteq$	innovation $\sqsubseteq$ activity $\sqsubseteq$ event

For the evaluation, we randomly selected 240 of the distinct relations (i.e., 50% of them). They were then evaluated in the same setting as the WordNet-based relations. Our results are shown in Table 3.4, where, as in the case of the WordNet-based relations, precision values were computed both individually

Table 3.4: Evaluation results for the relations derived from online ontologies.

	Evaluator 1	**Evaluator 2**	**Evaluator 3**	**Agreed by all**
Correct	118	126	81	62
False	96	56	57	17
Don't know	11	47	102	8
Precision	56 %	70 %	59 %	79 %

and for the jointly agreed relations. These values were in the same ranges as for WordNet. One particular issue we faced here was the evaluation of the named relations. These proved difficult because , unlike subsumption relations where the semantics are clearer to interpret, the names of the relations did not always make their meanings clear. This is evidently reflected by the increase of the number of "Don't Knows", especially for Evaluator 3. Moreover, different evaluators provided different interpretations for these and thus increased the disagreement levels. Moreover, named relations are harder to interpret as they are highly dependent on the purpose of the ontology. This made the task even more challenging for the evaluators, hence affecting their decisions. Therefore, again, we cannot provide a definitive conclusion of the performance of this particular algorithm. Nevertheless, each evaluator identified more correct than incorrect relations.

3.3.5 Observations on Integrating Relations into the Ontology under Evolution

A particularity of the use of Scarlet is that different relations are derived from different online ontologies, reflecting various perspectives and subscribing to different design decisions.

One side effect of exploring multiple knowledge sources is that the derived knowledge is sometimes redundant. Duplicates often appear when two or more ontologies state the same relation between two concepts. These are easy to eliminate for subsumption and disjoint relations, but become non-trivial for named relations.

Another side effect is that we can derive contradictory relations between the same pair of concepts originating from different ontologies. For example, between *Process* and *Event* we found three different relations: “disjoint”, “sub-class” and “super-class”. Such a case is a clear indication that at least one of the relations should be discarded, as they cannot be all integrated into the ontology. We leave this matter to be resolved by using external consistency checking tools that can easily detect and provide solutions for such inconsistencies.

As we mentioned previously, both our methods provide a relation as well as the inference path that is used to derive it. This makes the integration with the base ontology easier as more information is available.

An interesting situation arises when part of the path supporting the relation contradicts the base ontology. For example, the second relation in the path relating *Innovation* to *Event*, Row 6 of Table 3.3, contradicts the base ontology where *Event* and *Activity* are siblings. This is a nice illustration of how the base ontology can be used as a context for checking the validity of a relation. Indeed, we could envision a mechanism that increases the confidence value for those paths which have a high correlation with the ontology (i.e., when they "agree" at least on some parts).

In the process of matching a path to an ontology, we can encounter situations where some elements of the path only have a partial syntactic match with the labels of some ontology concepts. Referring to Row 5 of Table 3.3, some of the terms in the relation path connecting *Lecturer* to *Book* partially map to labels in the subsumption hierarchy of the base ontology:

$$\begin{aligned}&LecturerInAcademia \sqsubseteq AcademicStaffMember \sqsubseteq\\&\quad HigherEducationalOrganizationEmployee \sqsubseteq EducationalEmployee \sqsubseteq\\&\quad Employee \sqsubseteq AffiliatedPerson \sqsubseteq Person\end{aligned}$$

While our Jaro based matcher could not identify a match between *Lecturer* and $LecturerInAcademia$, this association can be done by taking into account the discovered path and the base ontology, therefore avoiding the addition of already existing concepts, and giving further indications on the way to integrate the discovered relations.

A final interesting observation relates to the appropriate abstraction level where a named relation should be added. We listed in Row 5 of Table 3.3 a relation path where *Lecturer* inherits a named relation to *Book* from its superclass, *Person*. Because *Person* also exists in the base ontology, we think that it is more appropriate to add the relation to this concept rather than to the more specific concept.

3.3.6 Error Analysis and Implications

One of the main goals of this experiment was to identify typical errors and to envisage ways to avoid them. We hereby describe some of our observations.

As already mentioned, in addition to the actual relation discovered between a new term and an ontology concept, our method also provides the path that lead to this relation, derived from WordNet or the external online ontology. Related to that, a straightforward observation was that there seem to be a correlation between the length of this path and the correctness of the relation, i.e. relations derived form longer paths are more likely to be incorrect. To verify this intuition, we inspected groups of relations with different path lengths and for each computed the percentage of correct, false, un-ranked relations, as well as the relations on which an agreement was not reached. These re-

Table 3.5: Correlation between the length of the path and the correctness of a relation.

Relation Path Length	**True**	**False**	**Don't Know**	**No agreement**
1	33 %	10 %	0 %	58 %
2	26 %	8 %	4 %	64 %
3	30 %	5 %	5 %	63 %
4	23 %	10 %	2 %	67 %
5	20 %	3 %	9 %	69 %

sults are shown in Table 3.5. As expected, we observe that the percentage of correct relations decreases for relations with longer paths (although, a similar observation cannot be derived for the incorrect relations). We also note that the percentages of relations which were not ranked and of those on which no agreement was reached are higher for relations established through a longer path. This indicates that relations generated from longer paths are more difficult to interpret, and so, may be less suitable for automatic integration. We address this issue in our work by providing a customisable length threshold that the user can set, as suitable within the application.

After further analysis of the sub-class relations generated through WordNet, we discovered some of the limitations at the level of using the Wu and

Palmer technique. This is due to the reason that two terms might be highly similar (e.g., *Stage* and *Year*), however the connection between such terms is not a direct relationship. For example they might be siblings from a direct node, or even at a different level. Such cases make the interpretation and integration of sub-class relations within the ontology more complex. Hence we modify our implementation of this method to extract all relations that connect the terms through a direct sub-class relation, without going through the similarity measures. It is worth to note that there exist other types of relations in WordNet that we do not process in our approach. For example, antonyms and has-stuff can also be derived from WordNet, however they are less prominent than hypernym relations, and are harder to translate into conceptual relations at the ontology level.

It became evident that relations established with abstract concepts or concepts that are poorly related to the base ontology have a low relevance. For example, several relations were derived for the $Thing$ concept (e.g., $< Lecturer, subClass, Thing >$). While these relations cannot be considered incorrect, they are of little relevance for the domain ontology, as they would not contribute in making it evolve in a useful way. Furthermore, we also identified a set of relations to concepts that are not relevant for the academic domain (e.g., death, doubt). While they sometimes lead to correct relations (e.g., $< Death, subClass, Event >$), these were rather irrelevant for the domain and thus should be avoided. We concluded that it would be beneficial to include a filtering step that eliminates, prior to the relation discovery step, those terms which are less relevant for the base ontology. Therefore, they should simply be discarded.

To deal with this issue, we identify the need to provide users with the following features: first, it should be possible for users to control the parts for the ontology to evolve. For example, it should be possible to ignore certain concepts that are not needed to relate to, or unfavourable to take into consideration. Second, having the possibility to check the new terms identified from the data sources can give users more control on what to consider for relation discovery and what to ignore. Thirdly, the ultimate solution would be to automatically assess the relevance of statements, and later allow users to check such relations before adding them to the ontology. Hence, instead of checking a big list of terms one by one to potentially add to the ontology, users can focus on checking statements that propose a relevant way of adding such terms to the ontology.

3.4 Discussion

In this chapter, we showed that reusing available structured knowledge can support the automation of the process of relation discovery. In our experiment, we explored the use of WordNet and online Semantic Web ontologies to discover relations between new terms identified from domain data, and existing concepts in the ontology under evolution.

While the experiment presented in this chapter has shown the feasibility of exploiting external background knowledge sources to automate, at least partially, the ontology evolution process, we identify different directions for improving relation discovery:

- First a different combination of knowledge sources might lead to different results. Currently the linear approach used in this experiment, i.e., WordNet followed by online ontologies, limits the set of concepts found in WordNet to only sub-class relations, without any named relations that are detected through Scarlet. An alternative to this is to check WordNet in parallel to online ontologies, to enable discovering all types of relations that apply to concepts.

- Second, online ontologies are dependent and directly affected with what is accessible through Watson. It might be useful to check the coverage of ontologies crawled by Watson, with respect to other knowledge sources available online. For example, DBpedia (Bizer et al., 2009) was not available in Watson when the experiment was performed. DBpedia provides access to structured content extracted from Wikipedia[12], and covers a wide selection of topics. While the processing time might be af-

[12] `http://www.wikipedia.org`

fected with the presence of DBpedia in Watson, the richness of relations discovery would definitely increase. It will be interesting in the future to check the effect of having DBpedia as part of our background knowledge sources used in terms of time and quality of relations generated, and can be easily done as soon as Watson crawls the DBpedia content. This is considered one of the main advantages of using online ontologies, which would indirectly improve the relation discovery process whenever new ontologies are crawled by Watson.

- Third, one element not considered at this level concerns the computational performance of our approach to ontology evolution. With the current settings performed in the experiment, accessing and processing online ontologies is time consuming as each relation is detected along with its path for one pair of terms at a time. We address this problem by creating a batch mechanism that processes the pair of terms in advance, first checking if they appear together in the same ontology, get the related entities in a cached file and then resolve the path of the relation.

While we take the majority of the discussed observations into consideration within our work and implementation, the most interesting and challenging question we get out of this experiment is: now that we have suggested relations linking new terms to existing concepts in the ontology, how do we validate them in terms of relevance? One of the main observations from our experiment is that the base ontology itself can be used for validating the correctness and relevance of a relation. Indeed, an overlap between the statements in the path and the base ontology is an indication that the relation is likely to be correct, and, inversely, if contradictions exist between the path and the ontology, the relation should be discarded. We propose in our next chapter a pattern-based technique, which takes into account structural situations that occur between the online ontology from where the relation is derived, and the ontology under evolution. Such patterns are used to detect the relevance of statements, along with a confidence value, with the aim to provide users support in the process of evaluating statements before applying them to the ontology.

Chapter 4

Using Ontological Contexts to Assess the Relevance of Statements

Our previous chapter highlights the need for automatically identifying ontology change requirements from domain data sources. Furthermore, as discussed in Chapter 2, we witness an increase in the availability of tools that automatically suggest new additions to be applied to ontologies in the form of statements (Cimiano and Volker, 2005; Maynard et al., 2009; Ottens et al., 2009). Nevertheless, although such tools support the automatic identification of ontology changes, they have introduced a new burden on users: inspecting the quality of a large number of proposed statements, mainly in terms of relevance with respect to the ontology. In this chapter we present our approach to answer our research question: *How to assess the relevance of ontology changes?*

4.1 Statement Relevance with Respect to an Ontology

There exist many tools that can be used to manage and preserve the consistency of an ontology after adding new statements (Ji et al., 2009; Sirin et al., 2007). However, assessing the relevance of a statement with respect to an ontology is not a trivial task, and is usually left to the user. For example, introducing *Concert* as a type of *Event* in an academic related ontology might not result in any logical conflict to the ontology, but it does not constitute a valuable addition to the ontology, where events are mainly about conferences, seminars, workshops, etc. With the abundance of existing approaches that deal with the consistency of the ontology, we solely focus in our research on the relevance aspect of ontology changes. We propose and evaluate in this chapter an approach for automatically assessing the relevance of statements with respect to an ontology.

We understand statement relevance with respect to an ontology as an indication of how well it fits in the ontology. Relevance is a core subject of interest in various domains including Artificial Intelligence, Cognitive Science (Sperber and Wilson, 1986; Sternberg, 1990) and Information Retrieval (Bruza and Huibers, 1996; Mizzaro, 1997). However, this problem is not very well explored in the domain of ontology evolution. As Wilson and Sperber noted in their work on relevance theory (Sperber and Wilson, 1986), two entities communicating and in exchange of knowledge, require a kind of agreement on the choice of context in which the conversation occurs. Moreover, they argue that "an input is relevant to an individual when it connects with background information he has available to yield conclusions that matter to him." (Sperber and Wilson, 1986)

Based on these key ideas, we present in this chapter an approach towards automatically assessing the relevance of statements with respect to an ontology. As identified by (Sperber and Wilson, 1986) that the context forms a key element in assessing the relevance of a piece of information, we define in our approach the context of a statement *as the set of neighbouring concepts and relations of the statement in the ontology in which it is used, up to a certain depth.* Online ontologies are also used at this level as the source of background knowledge, but in this case to generate such contexts. Our process starts by identifying the context of a statement, by exploiting the online ontology in which it appears (Section 4.2). This context is matched to the ontology to derive the shared concepts. We initially investigate a naive overlap approach that takes into account the number of shared concepts (Section 4.3). It is based on the idea that the more shared concepts exist between the target ontology and the external online ontology defining the context of a statement, the more relevant a statement is. With the various limitations of this technique, we point out the need for a more sophisticated approach that takes into account not only the shared entities, but also the structure surrounding them. We accomplish this by identifying a set of patterns (Section 4.4), where each pattern has specific application conditions and a confidence value. When a pattern occurs (i.e., when the application conditions can be fulfilled) at the intersection area of the statement context and the target ontology, a certain degree of confidence can be calculated. We back our work by an experiment that we perform in three domains (Section 4.5), showing that the pattern-based technique outperforms the naive overlap approach in terms of precision and recall, and can be used to support users in the selection of relevant statements during the process of ontology evolution (Section 4.6).

4.2 Overview of the Relevance Assessment Process

The relevance assessment process (Figure 4.1) starts with identifying a context C for the statement s from its source online ontologies. Subsequently, the context C is matched to the target ontology (i.e., under evolution) O_t

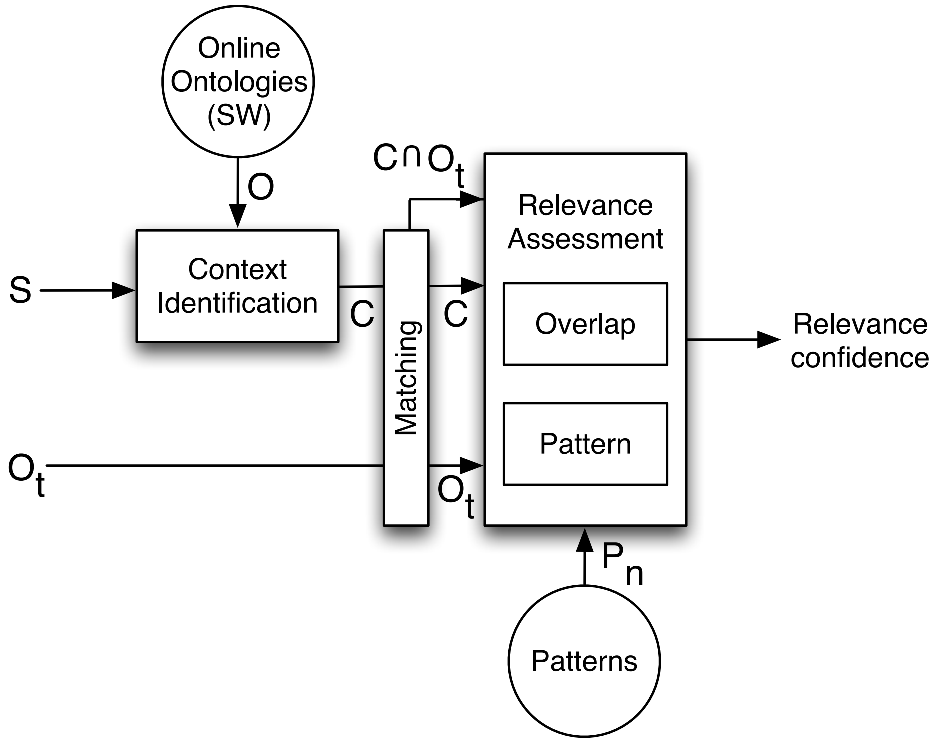


Figure 4.1: Checking the relevance of statement s with respect to target ontology O_t, where C is the context of the statement, and P_n is the pattern n applied.

to identify shared concepts, which result from the intersection of the graphs C and O_t, and used for the relevance assessment. For now, we focus on the subsumption relations, as they are less ambiguous than the named ones, of which relevance and correctness are much harder to assess even by users (as mentioned in Chapter 3).

4.2.1 Identifying the Context of a Statement

One way to assess whether a statement is relevant to a particular ontology is to rely on additional information provided by background knowledge sources. More specifically, we consider that such background knowledge can be given by the contexts in which this statement has been represented and applied. The main idea of our methodology is to explore the ontological contexts of statements–i.e., the contexts in which they are applied in other, external ontologies–to identify factors allowing to assess their relevance. Similarly to our approach in finding relations between terms (discussed in Chapter 3), and to other tools that exploit the open Semantic Web for performing a variety of tasks (d'Aquin et al., 2008b), our approach uses online ontologies as background knowledge to provide contextual information for a statement.

To find online ontologies in which the statement s appears, we use Scarlet, the relation discovery engine on the Semantic Web introduced in Chapter 3 (Sabou et al., 2008). We use the $subject$ and $object$ of s as input to Scarlet, which returns a list of relations that exist between the two entities, along with information about the source ontologies from where the relations have been identified. Once the online ontology is located, we use Watson to extract the needed elements from the specified ontology, forming the context of the relation. The extraction is performed based on a recursive function that exploits the links to $subject$ and $object$ up to a certain depth. More details about the implementation of the extraction procedure are discussed in Chapter 5.

As a context example, Figure 4.2 shows the result of extracting the context of the statement $< \ Project, has{-}funding, Grant \ >$ from the AKT ontology available online[1]. The extracted context is formed of elements linked up to depth 1.

4.2.2 Matching the Statement Context with the Target Ontology

The statement context is matched to the target ontology to detect their shared concepts. In our case, we consider the matching as a parameter, i.e., we do not impose any specific matching technique. In our implementation, we perform the matching between the concepts' names using the Jaro-Winkler string similarity metric (Cohen et al., 2003). We define the function $e(G)$ to extract the set of nodes n_i that exist in the graph G. We use the matching to generate the intersection of the statement context and the target ontology:

$$e(C) \cap e(O_t) = \{n_i \mid n_i \in e(C) \wedge n_i \in e(O_t)\}$$

In order to analyse the mappings between the statements' contexts and the ontology to evolve, we developed a tool to visualise such a mapping, clearly showing the intersection as well as the differences between the two graphs.

[1]`http://www.mindswap.org/2004/SSSW04/aktive-portal-ontology-latest.owl`

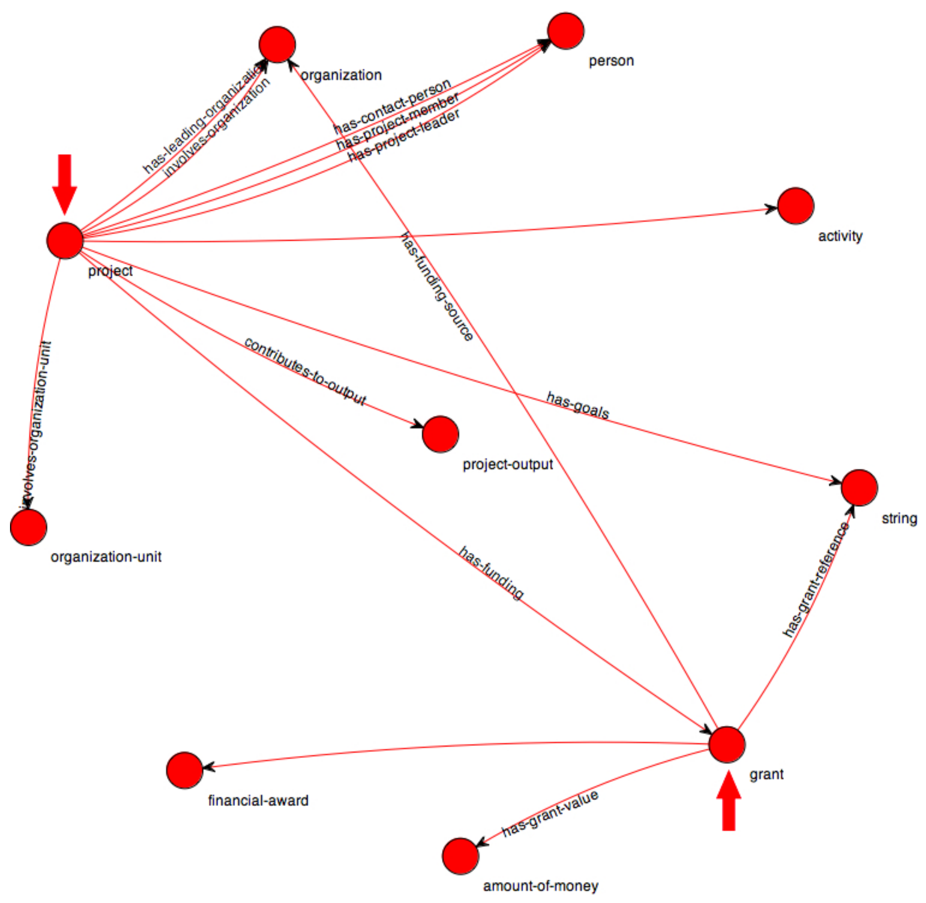


Figure 4.2: Example of ontological-context for the <*Project,hasfunding,grant*> statement.

The mappings resulting from the process described above allow us to divide the elements of the context and the ontology into three groups: 1- entities that are common to both C and O_t, 2- entities that are only present in C and 3- entities that are only present in O_t. Our visualisation displays a unique graph based on these three groups of elements, using different shapes and colours to distinguish them. Entities from the first group are merged and displayed in green, and represented as star-shaped nodes. Entities from the second group are represented in red, with round-shaped nodes. Entities from the third group are represented in blue, with square-shaped nodes.

The visualisation has customisable parameters that enable for example to only display the shared nodes with their connected entities up to a certain depth, and hide or show the target or online ontology. Figure 4.3 shows a visualisation of the context of $< proposal, subClass, document >$ extracted from the online OntoSem ontology[2], and the SWRC target ontology[3].

4.3 Assessing Relevance Based on Overlap Analysis

We investigate a first naive approach based on the idea that the more overlapping the statement context and ontology are, the more relevant the statement is. The relevance confidence in this case is based on the ratio of the number of shared concepts, to the number of concepts in O_t, as calculated using the following formula:

$$conf_{overlap}(s, C, O_t) = \frac{|e(C) \cap e(O_t)|}{|e(O_t)|}$$

[2]`http://morpheus.cs.umbc.edu/aks1/ontosem.owl`

[3]`http://kmi-web05.open.ac.uk:81/cache/6/98b/5ca1/94b45/7e29980b0f/dfc4e24088dffe851`

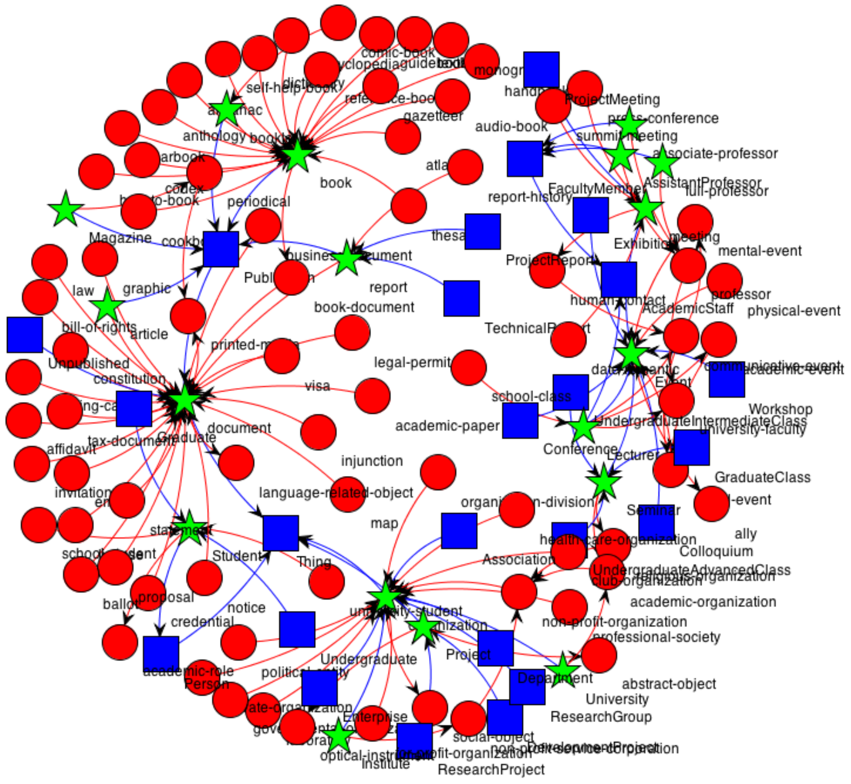


Figure 4.3: Visualisation of the overlap section between the OntoSem context of $<\ proposal, subClass, document\ >$ and the target SWRC ontology, where the star shaped nodes are shared, round nodes belong to the statement context, and square nodes belong to the target ontology.

For example in Figure 4.3, with the string similarity threshold value of 0.96, there are 18 shared concepts between the context of $< proposal, subClass,$ $document >$, and the SWRC ontology that includes 71 concepts. Thus the confidence of the overlap in this case is 0.2535 (i.e., $\frac{18}{71}$).

However, the drawback of this approach is that it does not take into consideration how the ontological entities connect with each other, as it focuses on the number of shared nodes only, without any additional analysis. As a side effect, all the statements used in the context will be treated with the same relevance confidence. With big ontologies that are not domain focussed such as OntoSem[4] or Cyc[5], it will cause the overlap technique to misjudge relevance. For example the statement $< \ capture, subClass, event \ >$ is extracted from OntoSem as well, but not relevant to add to the SWRC ontology. However, it has the same confidence value as the relevant statement $< \ proposal, subClass, document \ >$.

4.4 Pattern-Based Relevance Assessment

Given the limitations of the naive overlap technique, a more sophisticated approach is needed, which takes into account not only the overlap at the level of entity names, but also the way these entities are structured, giving a better indication of how the context fits in the ontology.

Relevance patterns are structural situations of interlinked nodes. When the surrounding entities in the matching graph around the statement s trigger such patterns, a degree of relevance can be identified. This lifts the problem of the overlap method that only matches the concepts' names, by providing further elements to analyse and hence a better relevance judgement. For example, a

[4]`http://morpheus.cs.umbc.edu/aks1/ontosem.owl`

[5]`http://www.cyc.com`

shared concept that is a sibling of an entity in s has a better influence on the relevance of s, than a shared concept which is not related to the elements of s. We create relevance patterns to detect such conditions and help assessing relevance. A clear visualisation of the context and its intersection with the ontology (as shown in Figure 4.3) helped in identifying the relevance patterns that we present in this part.

We discuss next how we identified the first relevance patterns (Section 4.4.1). Then we present the collection of our experimental data (Section 4.4.2) that helped in the discovery and evaluation of further relevance patterns. Then we show how we refined the generation of the statements' context (Section 4.4.3), and finally present the relevance patterns (Section 4.4.4).

4.4.1 Identifying the First Relevance Patterns

We started our investigations by the analysis of various graph examples. Using our visualisation tool, we generated a set of graphs showing how the context of relevant statements matched with the target ontology, and compared them to the matching of a set of graphs of irrelevant statements. For example, in the academic context, the graph of the relevant statement $< deliverable, subClass, report >$ highlighted the presence of siblings to *deliverable*, which are shared with the target ontology. This led to the creation of the first pattern (Pattern 1 described later in this chapter), which detects the availability of siblings shared with the target ontology. While the graph of the statement $< player, subClass, person >$, which is irrelevant to the academic context did not show similar connections with the target ontology.

Another pattern that emerged from our analysed example is when the statement is introducing a new parent to the ontology, and this parent is the parent of other shared concepts (Pattern 5 described later). During the analysis of the examples, one thing that became clear is that we need to have more cases to look at, as there was obviously more patterns to discover.

4.4.2 Gathering Experimental Data

To refine and discover further relevance patterns, we needed a gold standard of statements assessed in terms of relevance that would serve as the basis of our analysis and tests. As such a gold standard does not exist yet, we created a set of statements evaluated by experts for relevance in three different domains: academic, music and fishery. A total of 12 experts were selected to evaluate the statements. We assign three different experts for each dataset, with two academic datasets, given the expertise and availability of our evaluators in this area, and one dataset for each of the music and fishery domains. The experts were chosen from KMi and other research institutions (including the Universitad Politécnica de Madrid[6], Karlsruhe Institute für Technology[7], among

[6]`http://www.oeg-upm.net`

[7]`http://www.kit.edu`

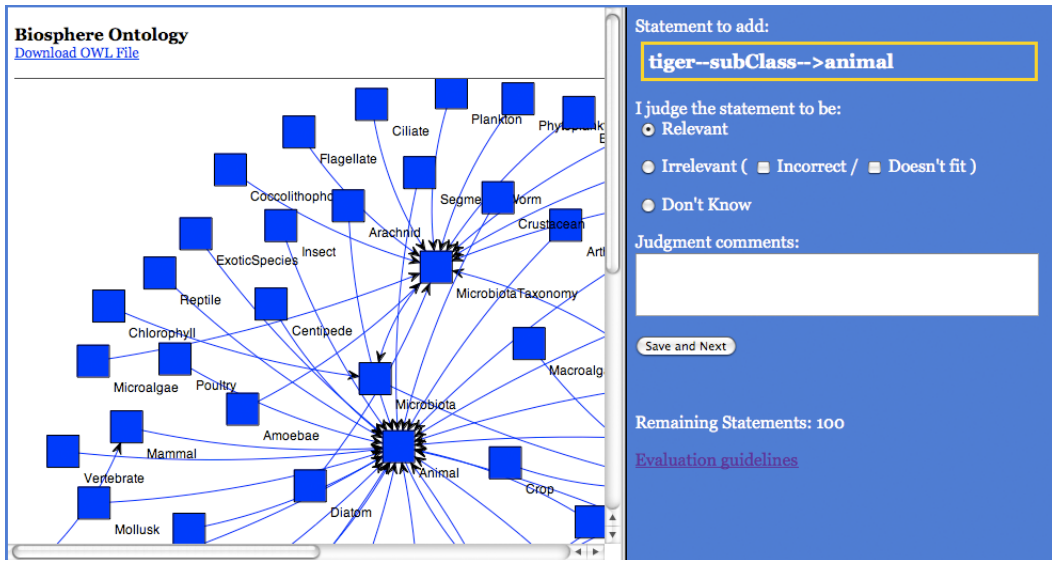
others). The assignment of datasets was based on their topic of preference. Similarly to the previous experiment described in Chapter 3, the evaluators were asked to play the role of ontology engineers.

Data collection of experts' evaluation was accomplished through a web interface, shown in Figure 4.4, and was conducted independently. It supplied experts with a visualisation of the target ontology, along with the options to select whether a statement is relevant, irrelevant or if relevance can not be judged from the given information ("Don't Know"), in addition to the possibility of leaving comments when needed. Experts were also given guidelines[8] describing the evaluation process, with some clarifications on what is meant by relevance supported by examples.

We use the same technique presented in Chapter 3, to generate the set of statements to add to the ontologies of each domain. In this case we use online ontologies as a source of background knowledge, which link new concepts extracted from text to existing ones in the ontology in the form of statements.

In the academic domain, we randomly pick 30 news articles published on the Knowledge Media Institute's website. For the fishery domain, we extract 108 online web documents that include information about fishes and fishery stock. For the music domain, we extract 20 music blog pages that have on average seven blog post headers each. Table 4.1 lists the domains, the tar-

[8]`http://evolva.kmi.open.ac.uk/experiments/statementrelevance/guidelines.php`


Biosphere Ontology

Download OWL File

Statement to add:

tiger--subClass-->animal

I judge the statement to be:

[✓ Relevant]

Irrelevant (Incorrect / Doesn't fit)

Don't Know

Judgment comments:

Remaining Statements: 100

Evaluation guidelines

Figure 4.4: Statement relevance evaluation web interface, showing the ontology to evolve on the left, the statement to evaluate on the right, with the assessment options.

get ontology to evolve, the corpus used and the total number of statements suggested.

We apply a filter on the generated statements to 1) select only the subsumption relations (cf. Section 4.2), and 2) remove generic relations, as our previous investigations show that statements linked to generic terms (e.g. *thing*, *object*, etc.) are mostly irrelevant (see Section 3.3.6). We generate random selections of 100 statements in each domain to form the datasets for the experts to evaluate.

Table 4.1: Statements generation setup for relevance assessment evaluation.

Domain	**Target Ontology**	**Corpus**	**Total** s
Academic	SWRC: http://kmi-web05.open.ac.uk:81/cache/6/98b/5ca1/94b45/7e29980b0f/dfc4e24088dffe851	KMi_News: http://news.kmi.open.ac.uk/	251
Fishery	Biosphere: http://kmi-web06.open.ac.uk:8081/cupboard/ontology/Experiment1/biosphere?rdf	Fishery_Website: http://fishonline.org/	124
Music	Music: http://pingthesemanticweb.com/ontology/mo/musicontology.rdfs	Music_Blog: http://blog.allmusic.com/	341

4.4.3 Statement Context Generation Revisited

A first improvement we introduce, following the analysis of the naive overlap approach, concerns the context generation. Instead of dealing with the online ontology as a whole to define the context, we generate the context of the statement based on the surrounding entities of the statement up to a certain depth within the ontology. This will help in focussing the usage of the statement by analysing the close entities only. For that, we use $context(s, O, d) = C$, a recursive function that generates a sub-graph, formed of nodes related through subsumption and other types of relations to the *subject* and *object* of s in O, up to a depth d (set to 1 in our implementation). This function is similar to the Prompt ontology view extraction (Noy and Musen, 2004) or some ontology modularisation techniques (d'Aquin et al., 2009). The sub-graph generated forms the context C of the statement in the specified ontology.

4.4.4 The Set of Relevance Patterns

The statement relevance evaluation based on expert users in concrete domains contributed to spotting further undetected relevant statements, which improved our selection and definition of patterns.

Each pattern has specific *application conditions*, supported by a *confidence value*. Application conditions are defined in a way that makes the patterns mutually exclusive, to facilitate their performance analysis. Based on our analysed data, we identified five different patterns. At a glance, Pattern 1 identifies direct shared siblings of the *subject* in s; Pattern 2 detects whether s introduces a new leaf to the ontology; Pattern 3 identifies shared ancestors of the *object* of s; and Pattern 4 detects shared siblings that occur at different levels of depth in the context and the target ontology. As per our analysis, shared ancestors (Pattern 3) gave better relevance indications then the other patterns, thus our application conditions are defined in a way to favour Pattern 3 over Patterns 1, 2 and 4. The last pattern, Pattern 5, is applied when s introduces a new parent to the target ontology.

Pattern 1: Direct Siblings. One core indication of relevance is when a new concept to add to the target ontology is surrounded by shared siblings between the statement context and target ontology. Shared siblings show that the concept in focus is missing in the target ontology, giving the statement adding it a high relevance. Pattern 1 detects shared siblings of the introduced concept, shown in Figure 4.5, where the statement to assess is in the dashed oval, round and square nodes belong to the context and target ontology respectively, and star nodes are the ones shared by both.

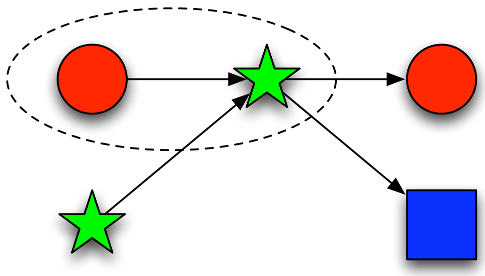


Figure 4.5: Pattern 1: Direct Siblings.

This is illustrated in Figure 4.6, where the statement in focus is $< tutorial, subClass, event >$ (in the dashed oval), in the context of the ISWC ontology[9]. This context shares with the SWRC target ontology the concepts *workshop* and *conference*. Those concepts show that the new concept *tutorial* is important to add to the SWRC ontology. *Application conditions*:

1. $\exists\, n_a \mid n_a \in e(C) \cap e(O_t) \wedge\ < n_a, subClass, object > \in C \cup O_t$

2. $\neg\exists\, n_b \mid n_b \in e(C) \cap e(O_t)\ \wedge C \models <\ object, subClass, n_b >$

Condition 1 ensures that the *subject* of s has direct siblings, while Condition 2

[9]`http://annotation.semanticweb.org/ontologies/iswc.owl`

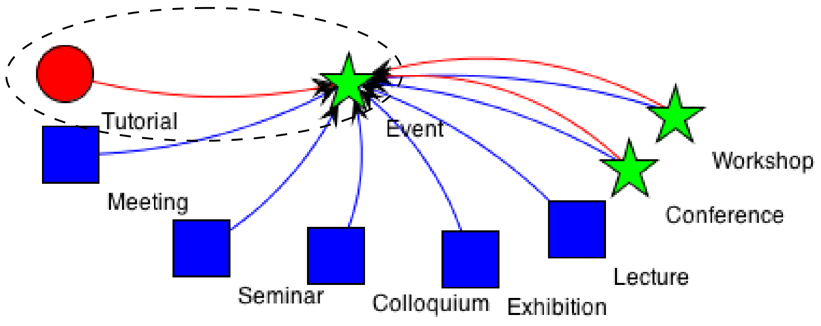


Figure 4.6: Pattern 1 detected on $s = < tutorial, subClass, event >$, $C = ISWC.owl$ and $O_t = SWRC.owl$.

checks that there are no shared ancestors, thus giving priority to Pattern 3. The *pattern confidence* formula is:

$$conf_{p1}(s, C, O_t) = \frac{|dSubC(object, C) \cap dSubC(object, O_t)|}{|dSubC(object, C)| - 1}$$

where $dSubC(n, G) = \{x_i \mid < x_i, subClass, n > \in G\}$, is a function to extract the set of direct sub-classes of a node in a graph. The confidence in this case is the ratio of the number of shared siblings (the numerator in the $conf_{p1}$ formula), to the total number of siblings in the context of s. If we apply the formula on $s_1 = < tutorial, subClass, event >$ in Figure 4.6, the confidence is:

$$conf_{p1}(s_1, ISWC, SWRC) = \frac{|\{Workshop, Conference\}|}{|\{Workshop, Conference, Tutorial\}| - 1} = 1$$

Even though Pattern 1 is one of the most intuitive patterns, it occurred on average only 11.25% of the statement cases (including relevant and irrelevant), in our four test datasets.

Pattern 2: New Leaf. As Pattern 1 relies on the shared siblings of the *subject* of s, it will fail when the *object* of s is a leaf in the target ontology, because there will be no shared siblings in this case. This is where Pattern 2 (Figure 4.7) called *New Leaf* comes in place, to detect the *subject* added as a new leaf to the target ontology. This pattern happened to be common in detecting

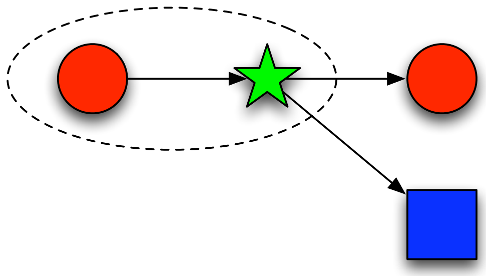


Figure 4.7: Pattern 2: New Leaf.

relevant statements in the music domain, where many statements introduce new ontology levels, for example statements $< duet, subClass, performer>$ and $< quartet, subClass, performer>$[10] link *duet* and *quartet* as sub-classes to *performer*, an existing leaf in the target ontology, as depicted in Figure 4.8. On average, this pattern occurred 10.25% of the cases in our tested statements. *Application conditions*:

1. $\neg\exists\, n_a \mid < n_a, subClass, object > \in O_t$

2. $\neg\exists\, n_b \mid n_b \in e(C) \cap e(O_t)\ \wedge C \models < object, subClass, n_b >$

[10]Statement contexts extracted from: `http://maciej.janik/test`

Condition 1 ensures that the *object* of s does not have children (i.e., it's a leaf in the target ontology), and Condition 2 confirms that the *object* doesn't have common ancestors. With the absence of close relatives (i.e., shared parents, ancestors and siblings), the confidence of the new leaf pattern is based on the ratio of the overlap of the target ontology O_t cut to a specified depth d around the *object* of s and the context of s, to the cut of O_t. Thus the *pattern confidence* formula is:

$$conf_{p2}(s, C, O_t) = \frac{|e(C) \cap e(context(s, O_t, d))|}{|e(context(s, O_t, d))|}$$

For the example depicted in Figure 4.8, the confidence value is 0.5714 (i.e., $\frac{4}{7}$).

Pattern 3: Shared Ancestors. The *Shared Ancestors* pattern (Figure 4.9) relies on the condition that the relevance of a statement with respect to a target ontology increases if the shared *object* in s has shared ancestors between the target ontology and the context in which it is used. This situation was very common in the fishery domain, where Pattern 3 applied to 50% of the statements identified. For example, for the statement $< cod, subClass, fish >$, $fish$ has the ancestor *animal* in C[11], which is shared with the biosphere ontology O_t. The example is visualised in Figure 4.10. This reflects a degree of common representations of animal species in online ontologies, where top

[11]The context of the statement is derived from: `http://morpheus.cs.umbc.edu/aks1/ontosem.owl`

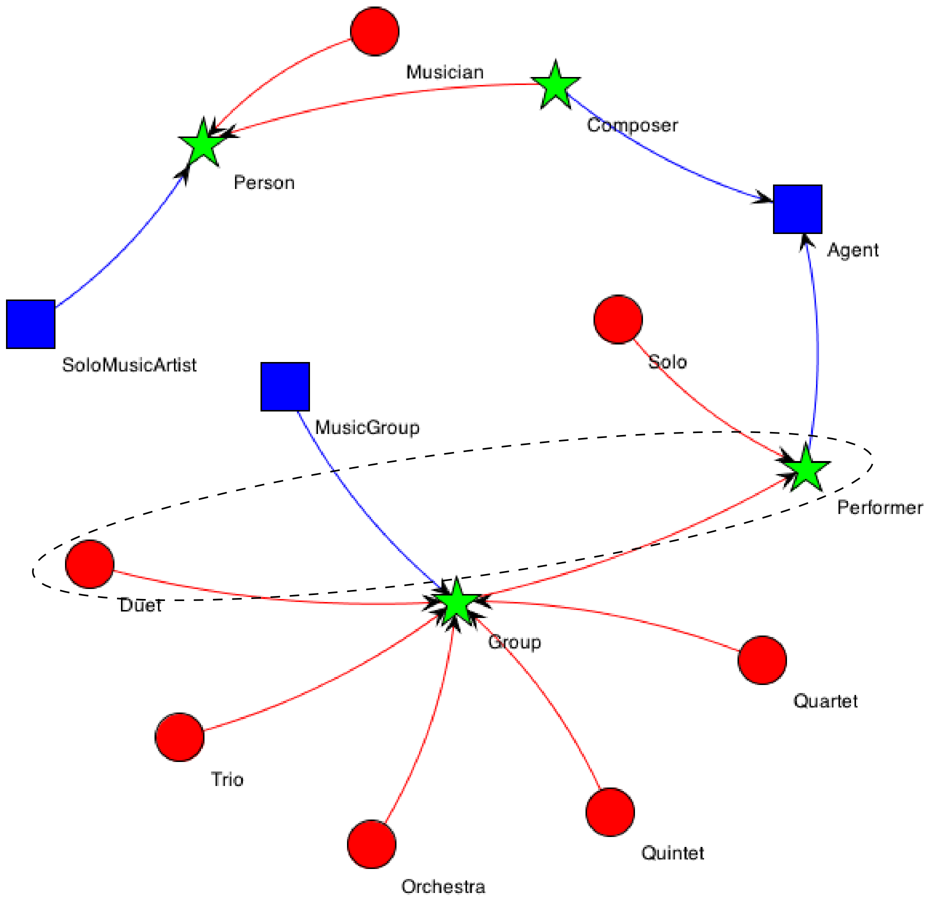


Figure 4.8: Pattern 2 detected on $s = < duet, subClass, performer >$, $C =$ `http://maciej.janik/test` and $O_t = MusicOntology$.

levels in many ontologies tend to be more aligned than in the other domains. On average this pattern occurred in 20% of our analysed statements datasets.

Application condition:

1. $\exists\, n_a \mid n_a \in e(C) \cap e(O_t) \ \wedge C \models < object, subClass, n_a >$

The *pattern confidence* formula is:

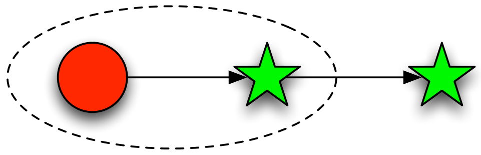


Figure 4.9: Pattern 3: Shared Ancestors.

$$conf_{p3}(s, C, O_t) = \frac{|aSupC(object, C) \cap e(O_t)|}{|aSupC(object, C)|}$$

based on the ratio of shared ancestors of the *object* of s, to the total number of ancestors of *object* in C. $aSupC(n, G) = \{x_i \mid G \models < x_i, superClass, n >\}$ extracts all the (direct and inferred) super-classes of a node n in a graph G. In the case of $s_3 = < cod, subClass, fish >$, visualised in Figure 4.10, the shared ancestors are $\{Vertebrate, Animal\}$, which number is equal to the total number of superclasses of $fish$ in the context. Hence the confidence in this case is:

$$conf_{p3}(s_3, OntoSem, Biosphere) = \frac{|\{Vertebrate, Animal\}|}{|\{Vertebrate, Animal\}|} = 1$$

Pattern 4. Granularity Mismatch. As ontologies are used in different application contexts, design decisions such as the level of granularity often vary from an ontology to another. This affects the performance of Pattern 1, which checks only the *direct* shared siblings of the *subject* in s. Pattern 4 (Figure 4.11), called *Granularity Mismatch*, identifies such situations. With the highest occurrence of 41.75% of the cases, this pattern shows that granu-

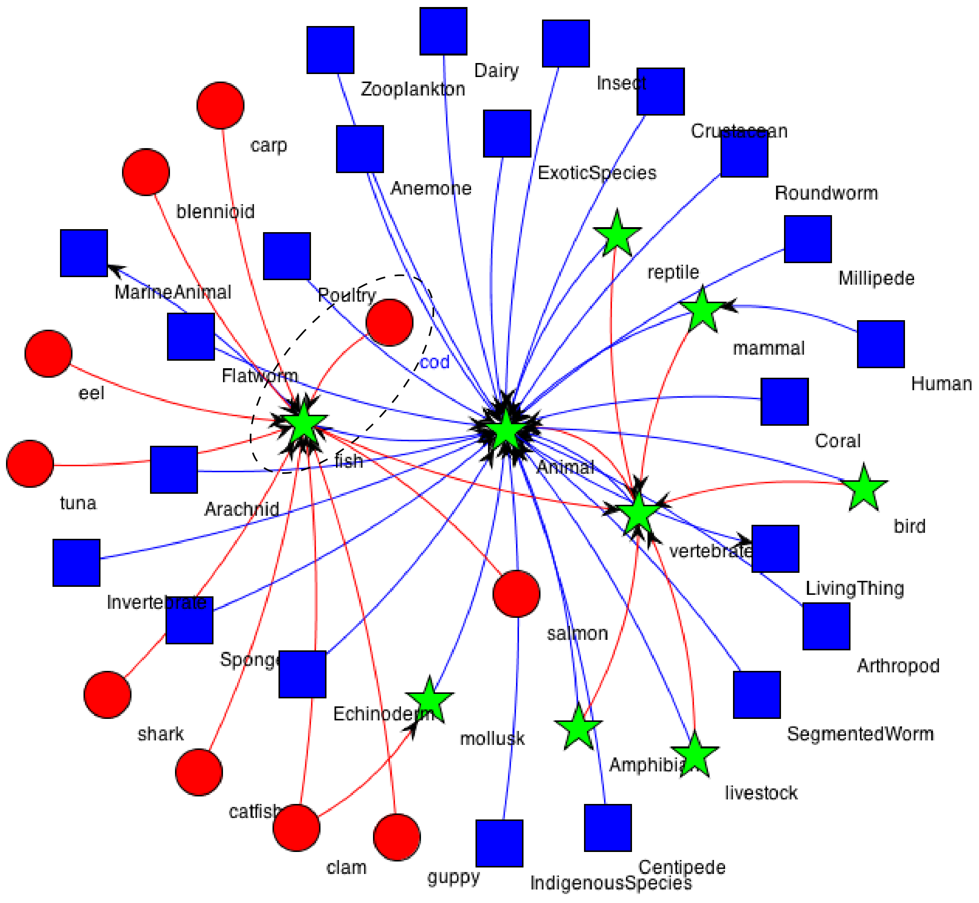


Figure 4.10: Pattern 3 detected on $s = < cod, subClass, fish >$, $C =$ `http://morpheus.cs.umbc.edu/aks1/ontosem.owl` and $O_t =$ *Biosphere Ontology*.

larity differences in concept representation when designing ontologies is a very common case. With our tests performed on the datasets, we have set this pattern to be applied as a last resort if Patterns 1, 2, and 3 are not detected.

Application conditions:

1. $\neg\exists\, n_a \mid n_a \in e(C) \cap e(O_t) \wedge < n_a, subClass, object > \in C \cup O_t$

2. $\exists n_a, n_b | n_a \in e(C) \cap e(O_t) \wedge n_b \in e(C) \ominus e(O_t) \wedge (n_b \in aSupC(n_a, C) \vee n_b \in$

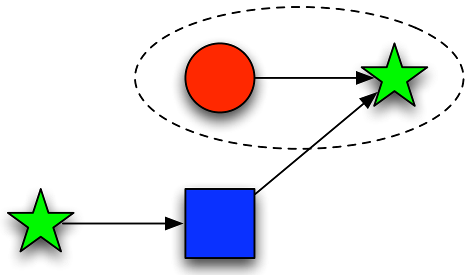


Figure 4.11: Pattern 4: Granularity Mismatch.

$aSupC(n_a, O_t)) \wedge object \in aSupC(n_a, C) \cap aSupC(n_a, O_t)$

3. $\neg\exists\, n_a \mid n_a \in e(C) \cap e(O_t) \wedge < object, subClass, n_a > \in C$

where Condition 1 is for ruling out the presence of Pattern 1, and Condition 2 checks for the presence of shared siblings (including the inferred ones) that fall at different levels in depth than *subject* of s with respect to the *object* of s through a non-shared concept (i.e., a concept in the symmetric difference of C and O_t denoted by the symbol $\ominus$). Condition 3 rules out the presence of shared ancestors, for which Pattern 3 should be applied. This *pattern confidence* is:

$$conf_{p4}(s, C, O_t) = \frac{|aSubC(object, C) \cap aSubC(object, O_t)|}{|aSubC(object, C)|}$$

which takes the ratio of all the shared sub-classes of *object* in C and O_t, to the total number of all sub-classes of *object* in C. The function $aSubC(n, G)$ extracts all (direct and inferred) sub-classes of a concept n in G. Figure 4.12 shows an example in the food domain, where concepts are modelled at different granularity levels. In the target SmartPoduct food ontology[12], the concept *sauces* is defined as type of $food$, while in the online context derived from the *ontosem* ontology, *sauces* is a type of *preparedFood*, which is a type of $food$. In this case, Pattern 4 applies on the statement $s_3 = < broth, subClass, food >$, due to the shared sub-classes of $food$: *sauces*, *pasta* and *soup*. The confidence in this example is 0.375 (i.e., $\frac{3}{8}$).

Pattern 5. New Parent. In cases where s links $subject$ to $object$ through a super-class relation, i.e. s is introducing $object$ as a new parent to the ontology, Pattern 5 is applied (Figure 4.13). There is indication of relevance in this case if $object$ is a parent of other shared concepts between the statement context and the target ontology. The *application condition* of this pattern is solely limited to checking whether the type of relationship linking $subject$ to $object$ is super-class. The *pattern confidence* is based on the following formula:

$$conf_{p5}(s, C, O_t) = \frac{|aSubC(subject, C) \cap e(O_t)|}{|aSubC(subject, C)|}$$

The numerator in the fraction detects the number of shared concepts between C and O_t that are children of $subject$ in C.

In the datasets, the number of statements discovered with super-class relations is much lower than the sub-class relations. On average, only 16.75% of the total number of statements are super-classes. Furthermore, the percentage

[12]The ontology is available at: `http://projects.kmi.open.ac.uk/smartproducts/ontologies/food.owl`

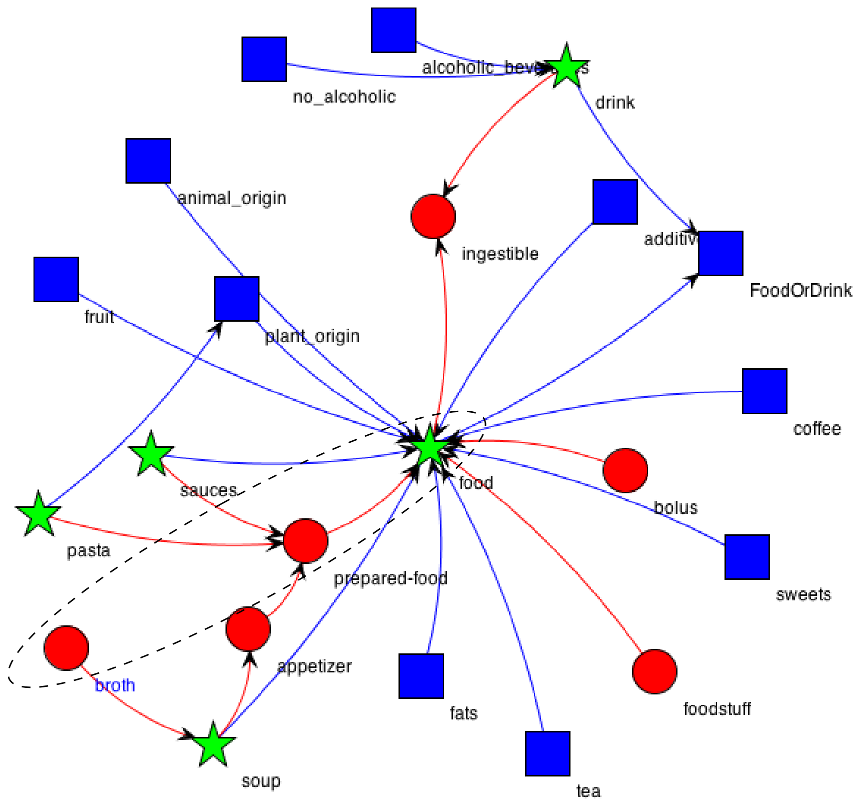


Figure 4.12: Pattern 4 detected on $s = < broth, subClass, food >$, $C =$ `http://morpheus.cs.umbc.edu/aks1/ontosem.owl` and $O_t =$ `http://projects.kmi.open.ac.uk/smartproducts/ontologies/food.owl`.

of relevance judgment correctness of this pattern is high in the four datasets. Thus one pattern dealing with super-class relations proved to be enough for our domains. Figure 4.14 shows an example in which we are introducing *staff* as a $superClass$ of lecturer to the $SWRC$ ontology. In this context, *staff* is a *superClass* of other shared concepts *administrative-staff*, making it relevant to be added.

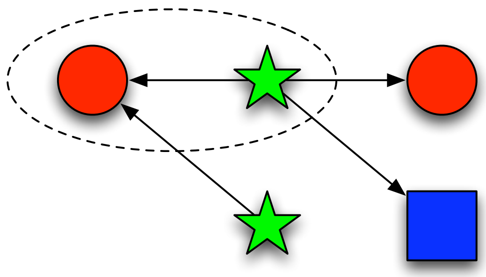


Figure 4.13: Pattern 5: New Parent.

4.5 Evaluation of Relevance Assessment

In order to evaluate the discussed approaches, we analyse and compare the performance of the naive overlap approach, versus the pattern-based approach. We use the experts' statements evaluation datasets in the three domains as the basis of our evaluation, which we present in this section.

4.5.1 Experiment Measures

Statement relevance being in many cases subjective, we made sure that each statement is evaluated independently by three experts per domain, having in total 12 experts for the four datasets. To get an idea of the level of agreement between the experts, we employ in this case the use of coefficient Kappa (Cohen, 1960), an agreement calculator that takes the element of chance into account. Given that the original Kappa was initially designed to take only two evaluators into account, we use the multiraters Kappa version, where evaluators are not forced to assign a certain number of cases to each evalua-

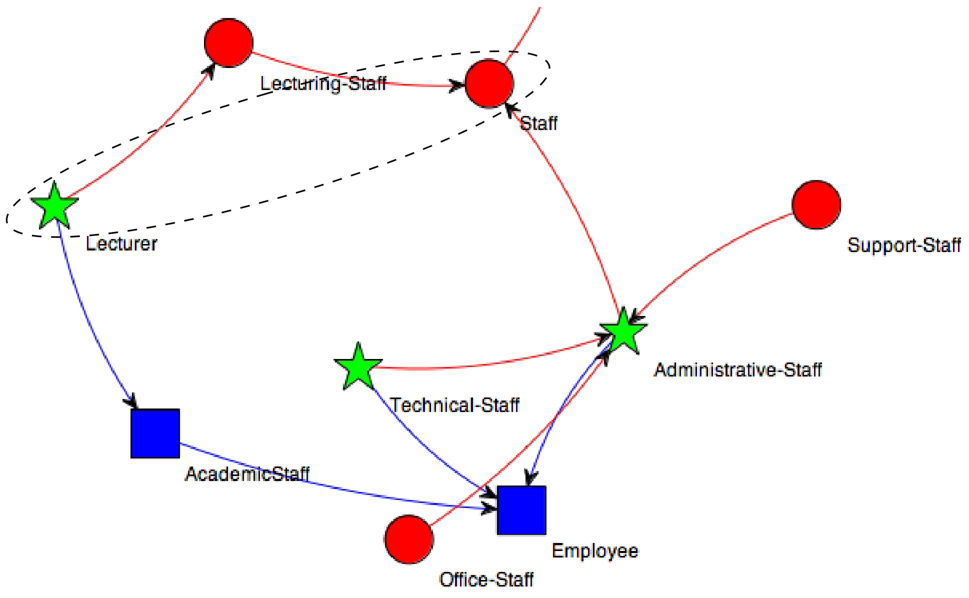


Figure 4.14: Pattern 5 detected on $s = <staff, superClass, lecturer>$, $C =$ `http://www.atl.external.lmco.com/projects/ontology/ontologies/comsci/csB.rdf` and $O_t =$ *SWRC Ontology*.

tion category, i.e., the free-marginal multirater Kappa measure, in addition to the overall agreement levels (Randolph, 2005). We rely on the online Kappa calculator (Randolph, 2008) to compute the agreement levels across the four datasets. The results are presented in Table 4.2.

The formulas for calculating the overall agreement and the free-marginal Kappa can be found in (Randolph, 2005). As the numbers show, we can see a higher level of agreement in the academic domain, than the music and fishery ones. Given that most of our evaluators are academics, with a degree of knowledge in music and fishery, this reflects in our scenario that there's a

Table 4.2: Agreement levels between experts evaluating relevance of statements across the datasets.

Measure	**Academic-1**	**Academic-2**	**Fishery**	**Music**
Overall Agreement	0.74	0.76	0.51	0.66
Free Marginal Kappa	0.61	0.64	0.27	0.49

higher consensus over the notion of relevance, when the domain tends to be rather more generic than specialised.

Based on the intuition that "relevance is not just an all-or-none matter but a matter of degree" (Sperber and Wilson, 1986), we use a measure to assess the overall relevance of each statement. To achieve this, we assign a score for each answer type from the experts: 1 for *relevant*, 0.5 for *don't know* and 0 for *irrelevant* (cf. Section 4.4.2). We use the sum of these values as an overall relevance score:

$$overall_{rel}(s, d) = \sum_{i=1}^{3} score(e_i, s, d)$$

where $overall_{rel}(s, d)$ is a function that returns the overall relevance score of a statement s in a dataset d, and $score(e_i, s, d)$ is the score given by expert e_i to s in d. For example, if the evaluation of a statement s in d is *relevant*, *relevant* and *don't know* by experts A_d, B_d and C_d respectively, the overall relevance value of s is 2.5. We set two thresholds to handle the overall relevance measure outcome: a *relevance threshold* sets the limit above which s is considered relevant and an *irrelevant threshold* below which s is irrelevant. If the overall relevance value falls between the two thresholds, the relevance can not be determined in this case, as the experts are undecided.

Concerning the naive overlap and pattern-based algorithms output, a threshold is set to determine the relevance based on the confidence value for each algorithm, i.e., when the overlap or a pattern is applied with a confidence degree higher than the specified threshold, the corresponding statement is classified as relevant, otherwise it is irrelevant. Given the different ways that each pattern calculates confidence, we use a separate threshold for each pattern (displayed in Table 4.3[13]). As the goal of this experiment is to check the feasibility of the pattern-based approach, we empirically set the combination of thresholds that resulted with the highest performance.

One of the downsides of having a threshold set empirically, is the difficulty in replicating this method in a new domain. At this level of our testings, we demonstrated that it is feasible to determine the thresholds which lead to the highest average F-measures. One potential solution to apply this approach in a new domain, as we discuss at the end of this chapter, is to create an automatic way of threshold detection. Another solution is to assign one threshold to all the patterns, and provide a user customisable weight for each pattern, which can be changed based on the user preference in the applied domain.

[13]This is a corrected version of the thresholds used in (Zablith et al., 2010)

Table 4.3: Employed thresholds selected empirically to provide the highest average relevance and irrelevance F-measure in each dataset.

Threshold	**Academic-1**	**Academic-2**	**Fishery**	**Music**
User Relevance	2	2	2	2
User Irrelevance	1	1	1	1
Overlap	0.2	0.29	0.4	0.05
Pattern 1	0.2	0.1	1	0.08
Pattern 2	1	1	0	0
Pattern 3	1	0.4	0.05	0
Pattern 4	1	0.01	0.03	1
Pattern 5	0.8	1	0.5	1

We implement and apply the latter solution in our tool that we discuss in Chapter 5, which we also evaluate in Chapter 6.

We use *Precision*, *Recall* and *F-measure* to evaluate the performance of the relevance algorithms. We define 4 sets $REL_{ed}, IRR_{ed}, REL_{ad}$ and IRR_{ad}, where: REL_{ed} is the set of all statements evaluated as relevant by the experts in dataset d; IRR_{ed} the set of irrelevant statements as judged by experts in d; REL_{ad} and IRR_{ad} the sets of relevant and irrelevant statements respectively, as classified by the algorithm a (i.e., pattern or overlap), in dataset d. We use the following formulas:

$P_{rel}(d,a) = \frac{|REL_{ed} \cap REL_{ad}|}{|REL_{ad}|}$ $\qquad$ $R_{rel}(d,a) = \frac{|REL_{ed} \cap REL_{ad}|}{|REL_{ed}|}$

where $P_{rel}(d,a)$ and $R_{rel}(d,a)$ compute the precision and recall of relevance respectively, in dataset d as judged by algorithm a. We use the usual *F-measure* computation based on precision and recall. In the case of irrelevance, the formulas are similar to the ones of relevance, but replaced with sets related to irrelevance (i.e., IRR_{ed} and IRR_{ad}).

4.5.2 Results

The main conclusion of our experiment, as shown in Table 4.4, is that the pattern-based approach performs better than the naive overlap approach. By simply comparing the precision and recall in each dataset, patterns are able to identify more correct relevant statements as classified by experts, with a better precision than then overlap approach. Overall, the overlap relevance F-measure is in the range of [7.41%, 58.06%], while the range is higher for the pattern-based relevance F-measure [43.75%, 69.05%]. In terms of irrelevance, the range is [60.87%, 85.71%] for the overlap approach, compared to the [74.74%, 92.48%] F-measure range using the pattern-based irrelevance detection. This is mainly due to the presence of large ontologies online that tend to highly overlap with target ontologies in general, and the fact that the overlap technique treats all statements coming from such ontologies equally, leading to lower precision and recall.

Table 4.4: Evaluation results for relevance assessment.

		Overlap		**Patterns**	
		Relevance	Irrelevance	Relevance	Irrelevance
Academic-1	Statements	18	82	13	87
	Precision	05.56%	83.72%	46.15%	91.95%
	Recall	11.11%	87.80%	66.67%	93.02%
	F-measure	07.41%	85.71%	**54.52%**	**92.48%**
Academic-2	Statements	15	85	16	84
	Precision	26.67%	81.18%	43.75%	90.00%
	Recall	25.00%	86.25%	43.75%	85.71%
	F-measure	25.81%	83.64%	**43.75%**	**87.80%**
Fishery	Statements	57	43	59	41
	Precision	47.37%	74.42%	55.39%	90.24%
	Recall	75.00%	55.17%	91.67%	63.79%
	F-measure	58.06%	63.36%	**69.05%**	**74.74%**
Music	Statements	57	43	35	65
	Precision	29.82%	81.40%	42.86%	83.08%
	Recall	73.91%	48.61%	65.22%	75.00%
	F-measure	42.49%	60.87%	**51.73%**	**78.83%**

Note that, in the context of validating changes for ontology evolution, identifying irrelevant statements is equally important as identifying relevant ones. Moreover, our experiment shows that in most datasets, the proportion of irrelevant statements is higher than the one of relevant statements. Thus having a high precision and recall on the bigger portion of the datasets (formed of irrelevant statements) reflect that the pattern-based approach would successfully act as a filter of irrelevant statements, reducing the workload on the user in the process of statement selection during ontology evolution.

To put the results in perspective, we rank the outcomes based on the confidence values of the overlap and pattern-based approaches, and compare them to the randomly ordered statements generated initially during relation discovery (see example in Figure 4.15). Due to the pattern specific threshold and confidence calculations, a direct ranking based on the confidence is not possible. Thus we normalise the pattern-based confidence values to a target unified threshold of 0.5, based on which we perform the ranking. As Figure 4.15 shows, the ranking based on the pattern technique groups relevant statements more towards the top of the list, meaning that ontology engineers could more confidently select most of the top statements, while safely discard most of the lower ranked ones. One interesting feature to test, which is beyond our research plans, is to investigate how these results would combine with other statement evaluation techniques (i.e., based on consistency checking, impact evaluation, etc). In particular, integrating consistency checking features per statement before being added to the ontology can assist the user in the statement selection process. We foresee a potential way to achieve this is by checking each proposed statement against existing statements in the ontology. Furthermore, it will be required to check the consistency of the statements to add among themselves, to ensure the consistency preservation of the ontology under evolution, when multiple statements are considered for addition. However we believe that doing such checks individually would be an expensive process, especially if the amount of changes to apply is substantial. Hence an alternative would be to check the consistency of the ontology after evolution, which can be easily done by off-the-shelf consistency checking tools. Such tools usually provide inconsistency justification features, by identifying the set of axioms causing the inconsistency and making it easier for ontology engineers to resolve it (Ji et al., 2009).

4.6 Discussion

We presented in this chapter an approach towards the automatic assessment of the relevance of statements with respect to ontologies. This approach is based on the analysis of the context in which the statement occurs, and how it compares to the considered ontology. A set of relevance patterns in the graph merging the context with the ontology are identified, with the aim to

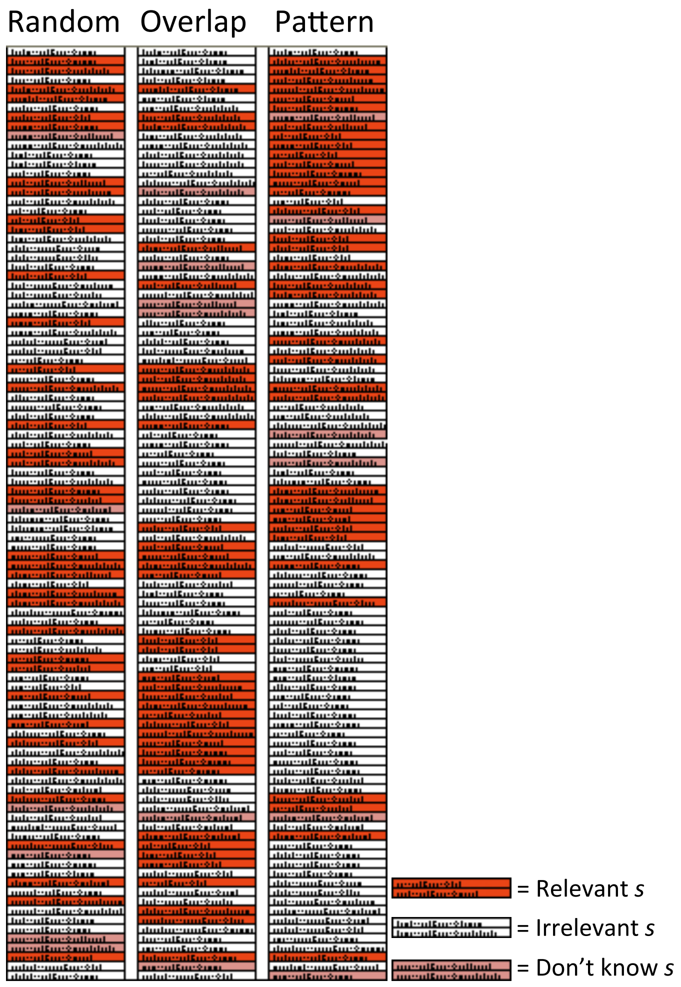


Figure 4.15: Visualised ranking of 100 statements in the fishery domain, comparing the results of the random order on the left, overlap approach in the middle and pattern-based approach on the right side of the figure.

provide indications of the level of relevance of the statement, by showing how the context fits in the ontology.

This experiment shows a potential in assessing the relevance of statements with respect to an ontology, a task which was traditionally left for the user to perform. Exploiting the structure of online ontologies and of the ontology under evolution were key for obtaining the improvement over a solution based on a pure matching technique. In addition, our experiment reflects the subjective nature of relevance, observed from the level of disagreement between the evaluators. Even though the evaluation of our approach shows promising results, we identify potential improvements that could be applied to our methodology. Firstly, we are aware that the relevance patterns identified are not exhaustive, and that further data analysis applied in different domains would highlight additional patterns. Moreover, we design our approach to have mutually exclusive patterns. A modification can be done at this level to check whether a combination of patterns would have any effect on the performance. Secondly, instead of using the first online ontology returned by Scarlet as the statement context, an alternative method is to select the context that returns the highest relevance confidence. In addition to that, the current way thresholds are set is empirical. An interesting study would be in the direction of investigating a methodology to automatically identify the optimal set of thresholds applicable in a domain. One potential way to do so is to take the set of statements that have been lately added to the ontology under evolution as a base case of the threshold values calculation. The idea is based on the fact that such statements are already assessed relevant by the user who added, or approved the addition of the statements to the ontology.

The outcome of this work is integrated in our ontology evolution tool Evolva, which we discuss in the following Chapter 5. However, it is worth to note that this work can be used in other application environments, for example, within the Watson plugin for the NeOn Toolkit[14]. This plugin supplies ontology developers with the ability to check online statements that relate to a specific concept. This is achieved by the user selecting the concept in the ontology, then searching through Watson to get the list of potential statements. Currently, the user has to scan through all the relations in order to select the appropriate ones to add. Having our work integrated in Watson would give the ability to rank the results based on relevance, helping users by scanning through the most relevant ones first. In addition to the evaluation conducted in this chapter, we put our approach to the test within an evolution scenario that we discuss in Chapter 6.

[14]`http://www.neon-toolkit.org`

Chapter 5

Evolva

With the aim to support users in the process of ontology evolution, we apply our research outcomes within one unified ontology evolution tool: Evolva[1]. This tool presents a solution to evolve an ontology starting from external data sources, by giving the user a degree of control during the evolution steps. In addition to providing users the ability to directly reuse our research solutions, Evolva played a key role in the evaluation of our proposed approaches in this thesis. This tool enabled us to evolve an ontology over a month period, in order to collect data for analysis and evaluation. Furthermore, Evolva provided an interface with customisable parameters, which enabled us to easily setup different evolution modes (i.e., manual, semi-automatic and unsupervised) that we test, evaluate and present in Chapter 6.

[1]http://evolva.kmi.open.ac.uk

One of the key features of Evolva, is that it uses customisable sources of background knowledge, to automate part of its processes at the level of change discovery and evaluation. We discuss in this chapter the evolution framework based on which Evolva is designed (Section 5.1). Then, we present the interface of Evolva (Section 5.2), followed by the implementation details within the NeOn Toolkit[2] (Section 5.3).

5.1 Evolution Framework

The framework on which Evolva is based went through different iterations during our research. It is formed of five components, to handle the discovery of domain information from external data sources, and support users in the process of the identification and validation of ontological changes to apply to the ontology. This framework is an instantiation of the complete evolution cycle discussed in Chapter 2.

5.1.1 Information discovery

Detecting the need for evolution (the first step of the evolution cycle in Figure 2.1), is applied through the information discovery component by contrasting existing knowledge in the ontology to information available in external domain and application specific sources, such as text corpora, databases or

[2] `http://www.neon-toolkit.org`

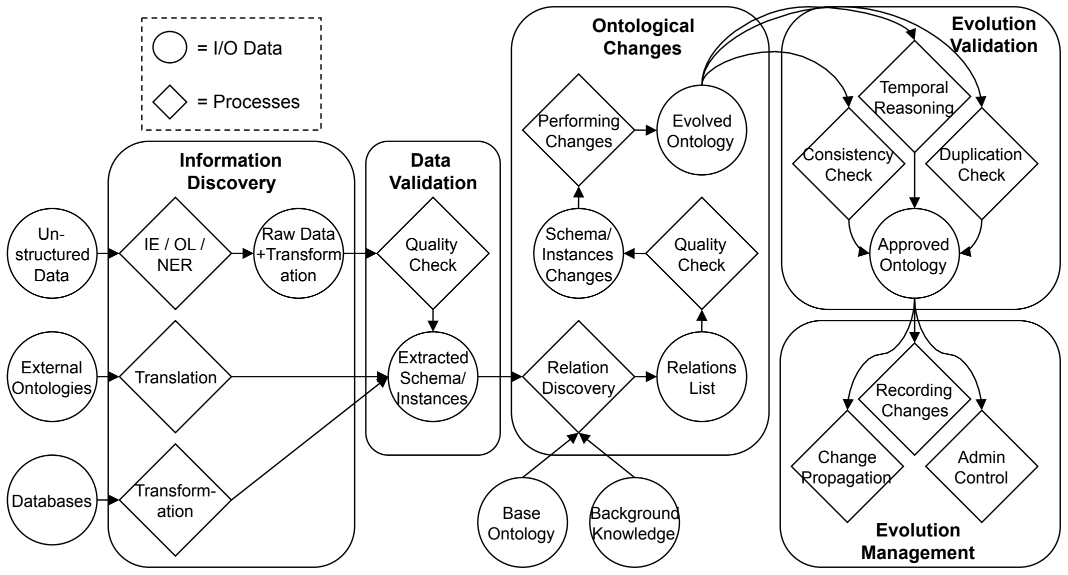


Figure 5.1: Ontology evolution framework.

other ontologies.

The framework is designed to handle unstructured sources including text documents, a list of raw terms or folksonomy entities. Text documents require certain processing techniques such as information extraction (e.g., using Gate's ANNIE component (Cunningham et al., 2002)), or ontology learning mechanisms (Cimiano and Volker, 2005) and named entity recognition (Maynard et al., 2008; Zhu et al., 2005). External ontologies and databases present a more structured source of information, where concepts, relations and instances are explicitly encoded in a well-defined structure. However, a translation should be applied on exploited ontologies to ensure language compatibility with the base ontology under evolution. In the case of databases, a transformation should be performed to encapsulate the database schema and entities in an ontology compatible language (Tirmizi et al., 2008).

5.1.2 Data validation

The information discovery component is likely to introduce a lot of noise in the generated data. This includes for example one or two letter words identified as terms, or terms that include unrecognised characters due to differences in the text encodings. One way to validate discovered information is by applying a set of heuristic rules. Ontologies and databases do not need this kind of low level quality check as the content structure is more trusted.

Another level of validation is the identification of entities that already exist in the ontology for example in the form of concepts. The data validation component is responsible for cleaning and identifying entities worth passing to the ontological changes component for further processing.

5.1.3 Ontological changes

This component is the core component in which we apply the core outcomes of our research: the relation discovery step that identifies links between the new terms and existing entities in the ontology (discussed in Chapter 3), and the quality check that includes the relevance assessment of statements (discussed in Chapter 4). In terms of the evolution cycle, this component covers the *suggesting changes* and *validating changes* steps.

The validated entities are passed to the relation discovery process, for resolving the links to existing knowledge in the ontology. This process relies on the various sources of background knowledge, and outputs the list of possible relations linking new entities to existing ones in the ontology.

The list of relations is then passed for quality check, in which relations are assessed in terms of relevance with respect to the ontology. Another potential way of validating the relations can be at the level of impact for example in terms of cost (Palmisano et al., 2008) or assertional effect (Pammer et al., 2010) of the relations on the ontology. It is worth to note that in the framework, quality checking is different from consistency checking, which occurs after applying changes to the ontology.

Checked relations are then passed to the performing changes step that integrates the new changes into the ontology to produce a new evolved version.

5.1.4 Evolution validation

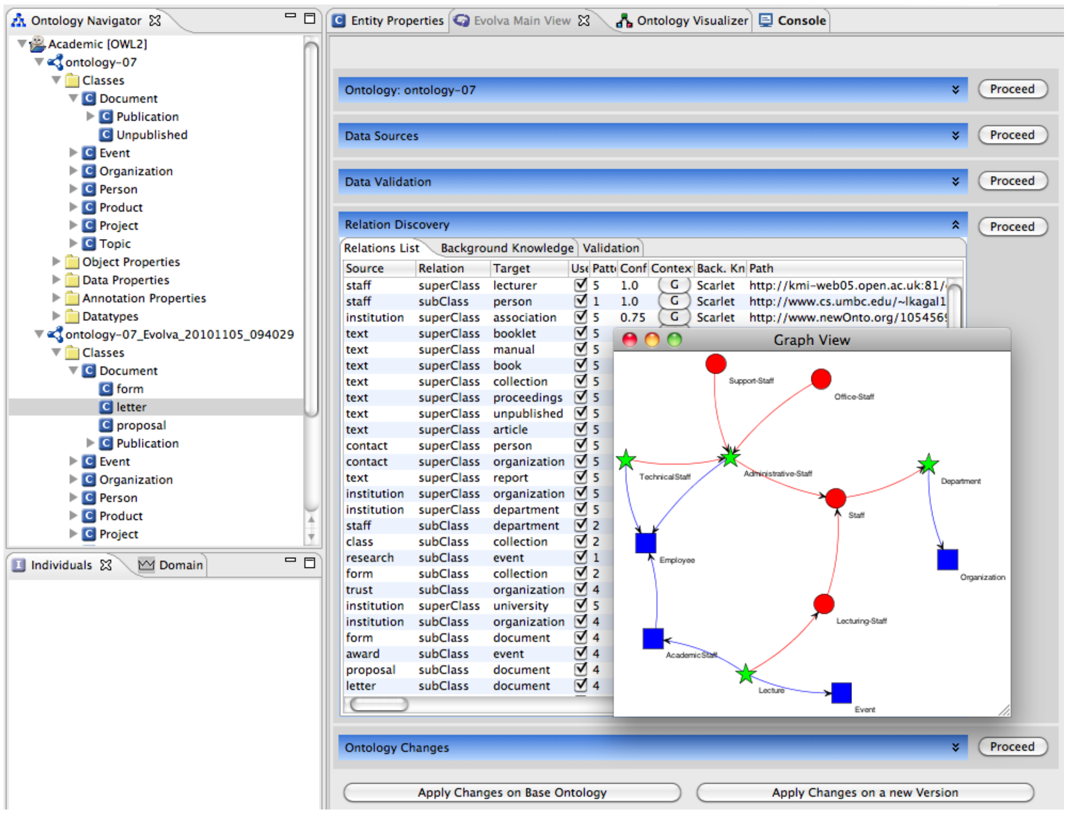
Performing ontological changes could generate some problems such as conflicting statements, data duplication and time related inconsistencies. The evolution validation component deals with these issues in order to produce an approved ontology in a reusable state. The consistency checking process evaluates the consistency of the new generated ontology, with an attempt to propose changes to be fixed in case of inconsistencies. This is part of the *validating changes* step using formal properties of the ontology evolution cycle (Flouris et al., 2006; Haase et al., 2005; Ji et al., 2009). The temporal reasoning process detects whether statements are temporally incorrect in the ontology, with respect to the data sources. For example the job status of a person in a certain company might change with time, and this should be captured correctly from the data sources into the ontology. The duplication check is an additional check of the occurrence of duplicates (e.g., *car* and *automobile*) resulting from the evolution.

5.1.5 Evolution management

The approved ontology is passed to the evolution management component, part of the *managing changes* step in the evolution cycle. In this component, the changes performed on the ontology are recorded to ensure functionalities such as tracing or rolling back changes (Maedche et al., 2002; Noy et al., 2004; Rogozan and Paquette, 2005). The changes should be propagated to dependent ontologies and applications. Administrator control is supplied for monitoring purposes, setting the evolution parameters and resolving any additional problem that might arise.

5.2 The Interface of Evolva

The interface of Evolva is sequential, with the ability for users to come back to previous steps and perform changes to their selected options. The described framework is not the architecture of our evolution tool (i.e., not all the framework components are implemented as part of the tool). However, Evolva follows an instantiation of our proposed framework. Figure 5.2 shows a screenshot of Evolva, where the steps required are depicted within the separating bars. The plugin has a generic *preferences* setting window to specify the



```
▼ Academic [OWL2]
  ▼ ontology-07
    ▼ Classes
      ▼ Document
        ▶ Publication
          Unpublished
      ▶ Event
      ▶ Organization
      ▶ Person
      ▶ Product
      ▶ Project
      ▶ Topic
    ▶ Object Properties
    ▶ Data Properties
    ▶ Annotation Properties
    ▶ Datatypes
  ▼ ontology-07_Evolva_20101105_094029
    ▼ Classes
      ▼ Document
          form
          letter
          proposal
        ▶ Publication
      ▶ Event
      ▶ Organization
      ▶ Person
      ▶ Product
      ▶ Project
```

Source	Relation	Target	Use	Patt	Conf	Contex	Back. Kn	Path
staff	superClass	lecturer	✓	5	1.0	G	Scarlet	http://kmi-web05.open.ac.uk:81/
staff	subClass	person	✓	1	1.0	G	Scarlet	http://www.cs.umbc.edu/~lkagal1
institution	superClass	association	✓	5	0.75	G	Scarlet	http://www.newOnto.org/105456
text	superClass	booklet	✓	5				
text	superClass	manual	✓	5				
text	superClass	book	✓	5				
text	superClass	collection	✓	5				
text	superClass	proceedings	✓	5				
text	superClass	unpublished	✓	5				
text	superClass	article	✓	5				
contact	superClass	person	✓	5				
contact	superClass	organization	✓	5				
text	superClass	report	✓	5				
institution	superClass	organization	✓	5				
institution	superClass	department	✓	5				
staff	subClass	department	✓	2				
class	subClass	collection	✓	2				
research	subClass	event	✓	1				
form	subClass	collection	✓	2				
trust	subClass	organization	✓	4				
institution	superClass	university	✓	5				
institution	subClass	organization	✓	4				
form	subClass	document	✓	4				
award	subClass	event	✓	4				
proposal	subClass	document	✓	4				
letter	subClass	document	✓	4				

Figure 5.2: A screenshot of Evolva.

parameters of the path to the WordNet dictionary, and whether an automatic evolution is needed without having the user to stop and validate each step. In addition to the generic preferences settings, each step has its own specific customisable parameters where applicable. The motive behind this design is to have task specific settings that users can change, and directly check the effect of the settings before moving to the next step in the process.

5.2.1 Setting the Ontology to Evolve

Starting with the *base ontology* step in the interface of Evolva, the user can specify which concepts to take into consideration during evolution. This is targeted for the cases where evolving only part of the ontology is needed. It is useful for example in situations where the ontology size is substantial, and the user is interested in evolving only part of the ontology. This also helps users to ignore generic terms in the ontology (e.g., Thing), which, as identified in the observations from our relation discovery experiment in Chapter 3, tend to generate irrelevant relations with respect to the base ontology.

5.2.2 Specifying the Domain Data Sources

In the *data sources* step, the user selects the domain data sources to process. The currently supported sources are text corpora, a list of terms, and RSS feeds. For a text corpus, the user selects a directory in which the text documents exist. In the case of a list of terms, the accepted input is in the form of a file, in which each term is on a separate line. For the RSS feed source, the URL of the feed should be entered. As RSS feeds are meant to evolve continuously, and the user not having control of the items to process, we implemented a caching mechanism that keeps track of the RSS items that have been already processed for evolution. Given the fact that RSS feeds sequentially publish new content whenever available in the domain, hooking Evolva to such feeds makes it easier to follow the evolution of the domain content, and evolve the ontology accordingly.

5.2.3 Validating the Identified Terms

The *data validation* step displays to the user the list of potential terms identified from the data sources. The validation at this level includes the automatic detection of whether the term already exists as a concept in the ontology, or whether the length of the term falls below a threshold under which it should be considered as noise. Both validation methods are customisable from the interface by setting the similarity threshold between extracted terms and concept names in the ontology, and the minimum length of the term to take into consideration. Moreover, the user has the ability to manually identify terms that have to be ignored during the evolution process, a need we identified in our relation discovery experiment (Chapter 3). At this level, the validation of terms is minimal. If time is not an issue, the user can keep the whole set of terms for the next step, where an additional more rigorous level of validation is performed, by taking the structure of the ontology into consideration.

5.2.4 Identifying and Evaluating new Statements

The *relation discovery* step is where the core outcomes of our research are implemented. This is where relations between the new concepts and existing concepts in the ontology are identified (based on our work presented in Chapter 3) and evaluated in terms of relevance (based on our proposed approach in Chapter 4). At this level, the user is presented with a list of relations that include the *source* entity, the *relation type*, the *target* entity, the option to *use* or discard, the *relevance pattern* that applies, the $relevance\ confidence$, the option to *visualise the contexts* based on which the relevance is assessed, the type of *background knowledge* used and the *relation path*.

The relation discovery process is customisable. The user can specify the type of background knowledge to use including WordNet, online ontologies, or both. The maximal length of the relation to discover can also be set. This proved to be useful as a result of our experiment discussed in Chapter 3, which showed that the correctness and relevance of a relation decreases with the increase of its length. Another customisable parameter provides the option to discover relations between the new entities identified from the data source among themselves, in addition to relations between new entities and existing concepts in the ontology. This is useful if the user aims to have a more complete taxonomical representation in the ontology using Evolva. To better illustrate the idea, suppose we are evolving an ontology that represent the animal domain. The ontology includes the concept *Animal* and Evolva identifies in the data sources the terms *Dog* and *Mammal*. The following relations are discovered: $< Dog, subClass, Animal >$ and $< Mammal, subClass, Animal >$. If the option to relate new entities among themselves is enabled, the following additional relation would be identified: $< Dog, subClass, Mammal >$, which if added to the ontology, will result in a more complete taxonomy.

In addition to the background knowledge parameters, the validation of relations is customisable as well. Evolva provides the feature to automatically remove duplicate taxonomical relations. Moreover, it is possible to hide or show relations that have been previously ignored during evolution. This is done by keeping track of the choices of the user regarding what relations have not been applied to the ontology. This helps reducing the amount of relations to check during the evolution process. Other available options come at the level of relevance validation. It is possible to change the weight applied to each pattern's confidence value. This option comes as a result of our experiment in Chapter 4, where we saw that some of the patterns would act differently in different domains. Having customisable pattern related weights helps the users to fine-tune the ranking of relations according to their relevance. Another set of options comes at the level of the context graph visualisation (see e.g., the small window displayed on top in Figure 5.2). For that, users can (1) display or hide the ontology under evolution, (2) display or hide the context of the relation derived from online ontologies, (3) limit the visualisation up to depth 1 around the shared nodes, (4) display the shared nodes only (the green starred nodes), and (5) display or hide the ontology's named relations.

5.2.5 Reviewing Changes and Applying them to the Ontology

The last step in Evolva's interface is the *ontology changes*. At this level, the list of all changes to apply to the ontology is displayed based on the selected relations in the relation discovery step. This acts as a final check of changes, and gives the user the possibility to go back to the relation discovery step in case settings need to be modified, or missed relations need to be considered or discarded. After this step, the options of applying changes to the ontology itself, or to a new version are enabled at the bottom of Evolva's interface. If the user chooses to *apply changes to the base ontology*, the changes are applied on the initial ontology. In case the user chooses to *apply changes to a new version*, a new version detached will be created, and directly accessible within the ontology navigator window of the NeOn Toolkit (the left part in

Figure 5.2). A tag with the string pattern "*_Evolva_EvolutionDate*" is added at the end of the ontology URI, to help tracing back the different ontology versions.

5.3 The Implementation of Evolva

We implemented Evolva as a plugin for the NeOn Toolkit (d'Aquin et al., 2008a). This helped in getting direct feedback whether our proposed ideas were feasible, and whether such ideas can concretise into usable solutions. The NeOn Toolkit provides a solution to handle the ontology engineering life-cycle, and various plugins have been implemented targeting different areas of the life-cycle (e.g., ontology development, consistency checking, modularisation, etc.). The toolkit is based on the Eclipse open source development platform (Budinsky et al., 2003). Eclipse provides the means for developing software projects as plugins that act as extension points to other plugins. In addition to Evolva supplying the toolkit with the feature for evolving ontologies starting from external data sources, Evolva benefited from reusing other plugins' functionalities to achieve some of its framework requirements. For example, for consistency checking, the RaDON (Ji et al., 2009) plugin can be used to spot inconsistencies in an evolved ontology. Another plugin for change logging (Palma et al., 2009) records changes applied on the ontology, and allows users to confirm or discard changes after being applied. An additional plugin offering relevant functionality for Evolva is the Gate Web Services[3] plugin (discussed later). This plugin processes text documents to identify terms and named entities.

5.3.1 Data Sources Processing and Validation

In the initial versions of Evolva, we used the Text2Onto's extraction algorithms (Cimiano and Volker, 2005) to process text documents. However this proved to be a burden at the installation level, as various prerequisite tools and settings were required. This included for example a local access to Gate[4], with specific parameters fetched from local files.

During the investigation of alternative options available to process text documents, the Gate Web Services[5] plugin for the NeOn Toolkit was released. This plugin offers a mechanism to process text documents and extract occurring terms through the *TermRaider* component. Being developed within the NeOn Toolkit, the plugin was easy to handle by exposing the needed java packages to be accessed by Evolva. As discussed previously, this is one of the advantages of the NeOn Toolkit: offering a unified environment where plugins can interact and share their functionalities. Having the Gate Web Services

[3] `http://gate.ac.uk/projects/neon/webservices-plugin.html`

[4] `http://gate.ac.uk/download`

[5] Description and installation guidelines are available at: `http://gate.ac.uk/projects/neon/webservices-plugin.html`

improved the installation of Evolva, as users do not have to install further applications to be able to run Evolva any more.

When the user specifies a *text corpus* as a starting point, the corpus directory is passed to the TermRaider component in Gate Web Services, which outputs the extraction results in a temporary file. Evolva processes the file to extract the terms, and visualises the results in the Data Validation section. In the case of an *RSS feed* selection, Evolva accesses the feed and processes its existing items, and downloads the related files content to a temporary directory that is processed in a similar way as the text corpus. With the user having less control over the content of the RSS feed, Evolva keeps track of the items that have been processed through an ontology dependent caching mechanism. The caching is applied at the final step when the user applies the changes on the ontology. The *terms list* is an option supplied to users who already have a list of terms that they want to process. This proved to be useful for example to process a list of food ingredients, to evolve the taxonomy of a food ontology. In this case, the user supplies the terms in a plain ".txt" file in which each term is on one line. Evolva scans each line and selects the available term. The terms identified from the different data sources are stored in a dynamic list that contains term related information, for example notes whether the term exists in the ontology, or if it has been manually ignored by the user.

In the *Data Validation* step, terms extracted from data sources are prepared before the relation discovery step. At this level, terms are checked against the existing concepts in the ontology, for detecting if they already exist. This detection is based on the Jaro-Winkler (Cohen et al., 2003) string similarity to allow a similarity-based matching, rather than an exact matching (as discussed in Chapter 3). The customisable threshold allows users to control the degree of similarity needed. The higher the threshold, the stricter the similarity between terms would be. We use the concept's URI local name in the ontology under evolution, to match against the extracted terms. This implementation worked well in our scenarios (e.g., used in Chapter 6), where the concepts' URI local names contained the concepts' information. However, this can not be applied to ontologies where the concepts' information are stored for example as labels, rather than in the URI local names that can contain pure reference numbers. This implementation can be easily extended in the future to handle such cases.

5.3.2 Relation Discovery

The *Relations Discovery* step relies on WordNet and online ontologies to link new entities to existing ones in the ontology. We implement a WordNet based relation identification method by using the Java WordNet Library[6]. Given two terms, the method scans the WordNet dictionary to find a subsumption relation (i.e., linked through a hypernym or hyponym property) between the

[6]`http://sourceforge.net/projects/jwordnet`

two terms. The dictionary is included in the WordNet application[7]. Evolva stores the path of the relation, with the gloss of the terms along the path. The path and the gloss help the user in the validation of the relation. For example suggesting the unusual relation $< Bag, subClass, Person >$ is explained by the gloss of *Bag* being *An ugly or ill-tempered woman*. This method also allows the extraction of relations up to a user specified depth, set from the interface.

To process online ontologies, Evolva relies on Scarlet that depends on the Watson gateway, as introduced in Chapter 3. Scarlet provides a java library[8], and supports the functionality of identifying relations between two terms from online ontologies. The function accepts several parameters, including for example a filter based on specific relation types, the length of the path, and whether to identify relations that spread across several ontologies, or limit them to one.

Since Scarlet is not designed to process many relations checking at once, the initial implementation using Scarlet proved to be very inefficient, as each pair of terms was checked for existing relation between them one at a time online and sequentially. To target this issue and allow processing more terms in one go, a new method was implemented within Scarlet to batch process a

[7]WordNet can be downloaded from: `http://wordnet.princeton.edu/wordnet/download`

[8]The Scarlet API can be downloaded from: `http://scarlet.open.ac.uk/download.php`

source of terms, against a given target terms. This methods initially checks if the pair appears together in one ontology using Watson, if it does, post-process the ontology to find and extract the relation and its corresponding path. Similarly to WordNet, the path of the relation is used as a validation by the user.

5.3.3 Statement Relevance Assessment

Within Evolva, relations generated through online ontologies retain the source ontology from where the statement s is identified. For the relevance checking, we focus on relations coming from online ontologies, leaving the assessment of WordNet relations for the user. We rely on the Watson API[9] to process online ontologies, and extract the entities that form the context used to analyse with respect to the ontology under evolution.

The statement context is extracted by generating the set of elements surrounding the statement in a specific online ontology. This include *super-classes*, *sub-classes* and *named relations* connected to the *Subject* and *Object* of s. We implement a recursive function that iteratively extracts the elements of the online ontology up to a certain depth. We then pass the entities of s, along with the online ontology, to get the elements through the *get-RelationType* functions of the Watson API, where *RelationType* can be the named relations

[9]The Watson API can be downloaded from: `http://watson.kmi.open.ac.uk/WS_and_API.html`

to and *from* the entity, *sub-classes* and *super-classes*. We collect the entities into a vector of strings, forming the context C that we use for the pattern identification, confidence calculation and context visualisation.

We process the ontology O to evolve using the OWL API[10] to extract the entities' identifiers, mainly based on the URI local names, which will be used to match against the entities in C. In our implementation, we match the strings using the Jaro-Winkler string similarity with a specified threshold, above which concepts will be considered similar. As mentioned in Chapter 4, relevance patterns occur when specific *application conditions* (i.e., structural forms) apply. We code the conditions in such a way to have one relevance pattern that applies for each statement. We develop a series of functions for detecting specific conditions. For example, in the case of Pattern 1, one condition is that the *Subject* of s needs to have shared direct siblings. For that, we develop a function that returns the number of siblings of a concept shared between C and O. Another example, used in Pattern 3, the condition is applied through a function that returns the number of shared concepts between C and O, which are super-classes of *Object* in C and O respectively. We reuse these functions to code each pattern's application conditions.

The relevance confidence values are implemented based on the formulas discussed in Chapter 4. Counters that keep track of shared entities are im-

[10]`http://owlapi.sourceforge.net`

plemented and reused depending on the pattern applied. The weight (a value between [0, 1] of the patterns, set through the Evolva parameters), is multiplied with the corresponding pattern confidence and affect the ranking of statements.

5.3.4 Context Visualisation

For the generation of the context visualisation, we use the JUNG API[11]. We developed a method that takes as input the context C and ontology O. Based on the Jaro-Winkler matcher, we represent the shared entities as green star shaped nodes. This can be specified by the JUNG API, which provides customisable features in terms of colours, shapes and node connections. The nodes from the ontology are generated as blue squares, and the nodes from the statement context as red circles. In addition to colour and shape customisations, JUNG provides different graph layout implementations. We use the Kamada-Kawai "spring-embedder" layout (Kamada and Kawai, 1989), implemented as the "KKLayout" class in JUNG. When the user inspects the button to visualise the graph, the context is extracted on the fly through Watson, matched against entities in the ontology, and drawn in a separate window on top of the Evolva plugin. The graph includes zooming in and out functionalities, to focus on and inspect specific parts of the graph, in addition to selecting elements

[11]`http://jung.sourceforge.net`

and dragging them to change their position in case the graph is cluttered.

5.3.5 Changes Generation and Implementation

The list of selected relations generated are converted into ontology axioms, using the OWL API. Such axioms are processed and passed to a function within the Evolva plugin, which applies the changes depending on the user selection. In case the user selects to apply the changes on the ontology itself, the function directs the axioms to be applied on the ontology model. In the other case where a new ontology version is needed, the function first creates a clone of the initial ontology, integrates the new axioms, and generates a new ontology having in its URI the Evolva tag, as well as the evolution date. The new ontology files are saved within the NeOn Toolkit workspace.

5.4 Discussion

In this chapter, we presented an overview of Evolva, a tool that assists users in the process of ontology evolution. Based on an evolution framework to handle evolution from external domain data, Evolva is implemented as a plugin to the NeOn Toolkit. Details about the installation and usage of the tool are available in Appendix A and on Evolva's website[12].

It is worth to note that the experiment we present in Chapter 6 helped us

[12]`http://evolva.kmi.open.ac.uk`

improve Evolva and introduce additional features. This is because, as part of the experiment, we evolved the ontology in a scenario spreading over several days. This provided the ability to spot glitches that were rather undetectable when running a one phase evolution based on one set of documents.

For example, one of the things Evolva was missing is a caching mechanism to store unselected concepts and relations, so that they do not appear again to the user in the following versions. This is already implemented and added for the experiment, in addition to a feature that shows the progress of the relation discovery step, to give an insight of the status to the user.

While we discussed in this chapter our framework and implementation details of Evolva, we identify at this level the following potential room for improvements:

- Firstly, relation discovery within WordNet can be made faster. For example, instead of passing the newly identified terms along with existing ontology entities as pairs, a more efficient way is to get the related hypernyms and hyponyms of a new term from WordNet, then check against all existing terms in the ontology in one go. This would reduce the number of queries performed on WordNet, and improve processing time.

- Secondly, the process of handling online ontologies can be refined. Currently, the graphs of statement contexts and their matching to the ontology are generated on the fly. If the contexts are cached at the level of relevance assessment during the pattern detection process, relevance graphs can be visualised in a much faster way.

- Thirdly, another improvement can be introduced at the level of background knowledge management and availability. It is feasible (by using Cupboard (d'Aquin and Lewen, 2009) for example) to provide users with a customisable space of online ontologies to be used as specialised background knowledge sources. This is useful in domain specific ontology evolution, where it is hard to find publicly available ontologies.

While such improvements are interesting to investigate, we keep them out of the scope of our work, as their impact will mainly be in terms of processing time and ontologies' availability online.

Chapter 6

Evaluating the Impact of Using Online Ontologies as Background Knowledge for Ontology Evolution

In this chapter, we focus on measuring the impact of using online ontologies on (1) the user involvement time during ontology evolution, and (2) on the quality of the resulting evolved ontology. To realise that, we conduct an experiment within the computer and related services sector of the "Tenders Electronic Daily" (TED)[1] portal, and use our ontology evolution tool Evolva

[1] http://ted.europa.eu

for processing the domain data and evolve the ontology. We compare the case where online ontologies are semi-automatically used to provide background knowledge to support the ontology evolution process, to the case where an expert manually evolves the ontology based on his/her own knowledge about the domain, and to the case where the process is done in an entirely unsupervised manner.

We analyse these results to support our claim that the use of online ontologies in a semi-automatic process can reduce significantly the time required for ontology evolution, with a minimal impact on the quality of the resulting ontology. Hence, with the right balance of user input and our proposed techniques, the use of online ontologies provides the best trade-off between quality and expert time, compared to the manual and unsupervised ontology evolution modes.

In the next section we discuss the evaluation methodology. This is followed by the preparation of the ontology to evolve and the data used for our experiment (Section 6.2). In Section 6.3, we present the pilot experiments that we conducted. Then in Section 6.4 we analyse our results, which consist of the analysis of: (1) user time, (2) entities discovered and added to the ontologies, (3) the effect of using online ontologies for relevance checking, and (4) the ontologies generated out of the three evolution modes, supported by evaluation metrics.

6.1 Evaluation Methodology

In order to evaluate the added value of using online ontologies as a source of background knowledge for supporting users in the process of ontology evolution, we mainly rely on an empirical methodology, coupled with metric supported ontology analysis to evaluate the resulting ontologies. We derive some of our ideas from the empirical methods for artificial intelligence (Cohen, 1995). We start by the claim that

> the use of online ontologies through the Evolva tool can substantially decrease the effort required by users to evolve ontologies, with little negative effect on the quality of the resulting ontology compared to a manually evolved one.

To check the feasibility of our claim, we setup an experiment that will serve as the basis our analysis. We start by preparing the ontology to evolve. Then, we collect the domain data to use for evolving the ontology. By relying on Evolva, we run the pilot experiments in three different ways: a manual, semi-automatic and unsupervised evolution modes. The main requirements needed for this evaluation were: (1) evaluators with ontology building knowledge, (2) knowledge in the computer and related services domain, and (3) running the experiment over a month period. Two evaluators from the Knowledge Media Institute (KMi) that fulfilled the requirements were selected, one for the manual evolution mode, and the other for the automatic mode. They ran the experiments independently. As in the previous experiments performed in Chapters 3 and 4, the evaluators were asked to perform the task as ontology engineers, who are required to keep the ontology up-to-date. The background information about the scenario were clearly agreed on: the ontology should evolve, to reflect the new emerging elements identified from new tenders documents published over a period of 30 days in the computer and related services domain. For that, new concepts appearing in the documents should be added with the appropriate relations to existing ones in the ontology. For each mode, the evaluator was clearly aware of which features of Evolva they are allowed to use. I.e., the evaluator in manual mode was only allowed to use the "Data Discovery" component of Evolva to identify terms from documents, while the evaluator in semi-automatic mode was given the ability to use all the provided functionalities of Evolva, with the requirement to validate the proposed relations, and maintain the final structural state of the ontology. We log and analyse the resulting data, to highlight the impact of using online ontologies on the evolution process.

6.2 Data Preparation

We conduct our experiment in the computer and services domain of the "Tenders Electronic Daily" portal, where daily information is published about tenders availability across Europe. We collect the tenders data related to this sector, published within the United Kingdom. We make use of the functionalities provided by Evolva, which can be customised to extract information from the specified data sources.

6.2.1 Initial Ontology

After searching for an existing ontology describing software and computer components that we can use as a starting point for evolution, we could not find an appropriate one that fits out intentions: to represent hardware and software products to use for tenders in the UK. Hence we created our own initial ontology, which includes basic representation of computer related devices, programming languages and software types[2].

6.2.2 Domain Data Collection

The TED portal provides RSS feeds access for automatic tenders update[3]. This source fits very well in our evolution scenario, where new domain information is published through the feeds daily. The RSS feed we used describes the availability of new tenders, documents and further information available in the computer and services industry within the UK[4]. With the extensive amount

[2]`http://evolva.kmi.open.ac.uk/ontologies/tedcomponto.owl`

[3]`http://ted.europa.eu/TED/rss/rssFeed.do`

[4]`feed://ted.europa.eu/TED/rss/en/RSS_comp_UK.xml`

of data published daily, keeping up with domain changes and evolving the ontology accordingly is time-consuming and requires a lot of effort from the users. This is because users will have to go through the published documents daily, identify relevant new concepts to add, and find the appropriate changes to perform on the ontology.

We use Evolva to extract the RSS feed items using its *data sources* component (Section 5.2.2). We ran the process daily over a period of 30 days. A total of 232 web documents were collected as a result, grouped into 21 sets of documents corresponding to 21 different days (no documents were published during certain days, especially over weekends).

6.3 Pilot Experiments Design

For our experiment, we setup three different ontology evolution modes: manual, semi-automatic, and unsupervised. The first focuses on measuring the effort required from an ontology engineer in evolving the ontology based on the list of terms identified from the RSS feed, without the support of online ontologies. The semi-automatic mode measures the effort required from another ontology engineer, who is assisted by the change discovery and evaluation features of Evolva. The unsupervised mode represents the case where online ontologies are employed without any user intervention.

6.3.1 Manual Mode

In the manual mode, the expert is given the domain data collected in the form of web documents, and Evolva running within the NeOn Toolkit. This mode serves as the baseline against which the other modes will be compared. The task here is to manually integrate new concepts detected from the data in the ontology. To perform this, the expert is required to load the ontology within the NeOn Toolkit, and feed the data to Evolva in order to extract the list of terms identified. The support of Evolva here is at the level of (1) the identification of terms (i.e., the *information discovery* component), and (2) the selection of new terms that do not exist in the ontology, with noise cleaning based on the terms' length (i.e., the *quality check* component). This enabled us to run the three pilot experiments on the same set of terms.

Evolva will return the list of terms to the expert, who will go through each term and check whether the term is relevant to be added to the ontology. In the case of relevance, the expert is required to add it to the ontology, with the appropriate relations to existing concepts in the ontology. Then a new ontology version will be saved by integrating new entities from the set of documents, resulting in having a sequence of ontology versions (i.e., 21 in total based on the number of document sets). Keeping a trace of all versions helped in identifying what has been added to each version. The time to perform this task is recorded for each ontology version, in addition to the number of concepts and relations added at each level, with further details including for example what has been discarded.

6.3.2 Semi-Automatic Mode

The second experimental mode is semi-automatic. This is where the expert is supported by the use of online ontologies. Similarly to the manual mode, the expert will be using Evolva's features to process and clean the domain data to extract the available terms. However, in this case, the relations linking new concepts to the existing ones are left for Evolva to discover using online ontologies. Hence the focus of the expert here is on going through the identified statements that already connect the new terms to existing ones in the ontology, rather than checking each term detected from the domain data individually.

In this mode, Evolva is customised to use online ontologies as the source of background knowledge. Online ontologies not only provide the feature of automatic relation discovery (discussed in Chapter 3), but also a mechanism for statement relevance assessment with respect to the ontology (discussed in Chapter 4). After identifying new concepts and linking them to the ontology, the expert will have the list of relations to check, with a ranking based on the confidence value of the statements' relevance. After being approved, relations are transformed into ontological changes, and applied on new ontology versions to create a sequence of ontologies as in the manual mode.

Throughout the process of using Evolva, the expert's time in validating relations is logged, along with what has been added and discarded in terms of concepts and relations during the evolution. This data is then compared to the other evolution modes.

6.3.3 Unsupervised Mode

The third evolution mode analysed is where there's no user intervention in the evolution steps. In this mode, only online ontologies are used to discover and assess the relevance of changes to add to the ontology, without the user validation. The aim of this scenario is to judge the importance of having the expert's supervision as part of the process.

In this unsupervised mode, we run the evolution on the same initial ontology. We select the option in Evolva's preference pane to automatically evolve the ontology. In this case, the relation discovery step suggests changes, and evaluates them in terms of relevance. Then all statements with relevance greater than zero are added to the ontology. Similarly to the previous two modes, 21 ontology versions are created.

6.4 Experiment Analysis

In this section we focus on the analysis of the collected data from the three evolution modes, in order to assess the impact of the use of online ontologies on ontology evolution. For this analysis, we are interested in (1) checking the amount of the overall time taken by the expert to evolve the ontology, (2) comparing what has been added to the ontologies in the three modes (3) measuring the effect of the automatic relevance assessment on the selection of statements to add and (4) comparing the final ontologies of the three modes based on our observations, supported by objective metrics.

6.4.1 Time Performance

The first impact of the use of online ontologies is on the time taken by the expert to evolve the ontology. The graph in Figure 6.1 plots the time taken by the second expert in the semi-automatic mode, versus the one in the manual mode to generate the ontologies. This shows that overall, the time spent by the expert supported by the use of online ontologies (i.e., in semi-automatic mode) is less than the time taken by the expert who is manually integrating the concepts in the ontology.

The accumulated time taken by the expert in manual mode is 481 minutes (i.e., 8.02 hours), while it took 149 minutes for the expert in the semi-automatic mode to perform the evolution. However, we found that in some cases, mainly days 2 and 4, the timing is not very accurate. After further investigations, the expert in semi-automatic mode highlighted that at these two days, the process has been interrupted without stopping the time logging. Hence if we look at

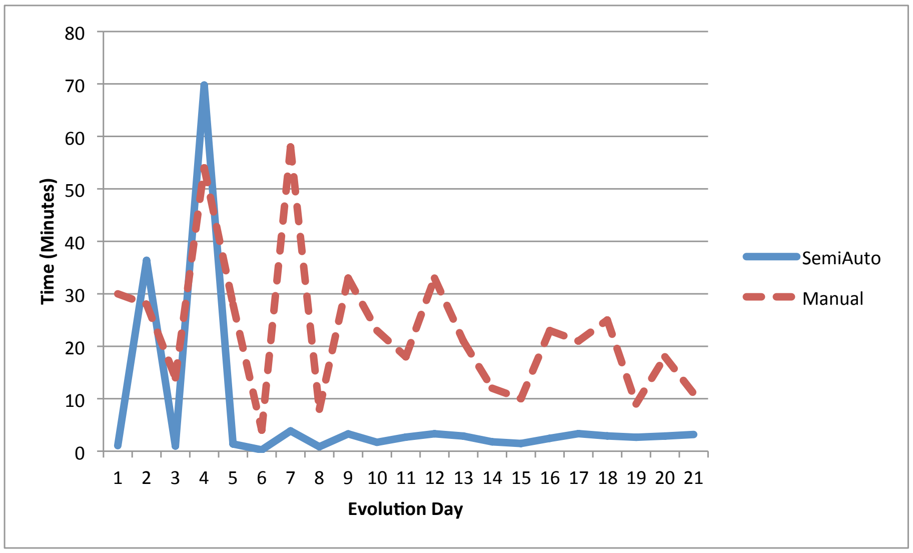


Figure 6.1: Graph of the time in minutes spent by the expert in the semi-automatic mode (full line), versus the time of the second expert in manual mode (dashed line).

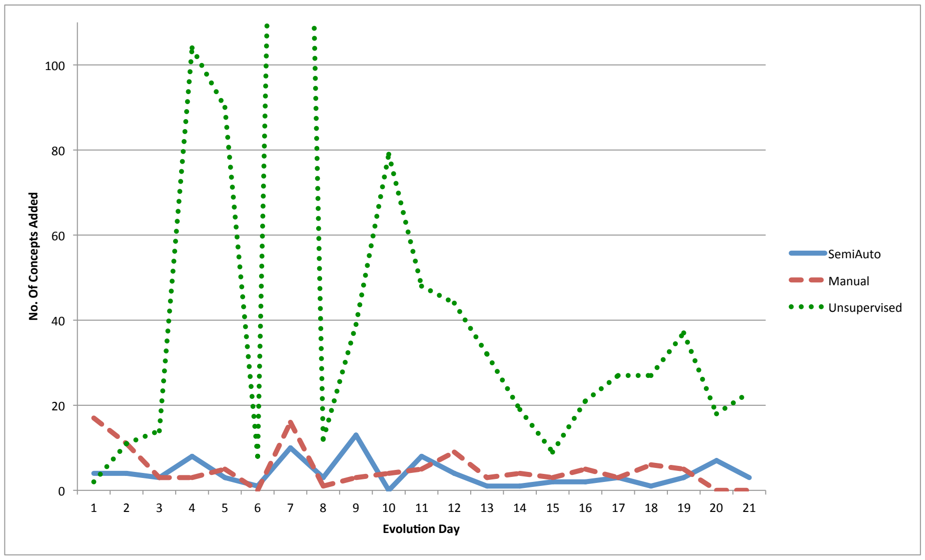


Figure 6.2: Graph of the number of new concepts added in semi-automatic mode (full line), versus the manual (dashed line) and unsupervised modes (dotted line).

days 5 until 21, the values are more realistic. In this case, the accumulated time for the expert in manual mode will be 355 minutes, versus 41 minutes for the user in semi-automatic mode. So the time decrease factor would be 8.66 $(\frac{355}{41})$.

Another observation from the graph, is that the time spent in evolving the ontology in both modes tends to decrease with time. This is due to various factors, including that the experts were getting more used to the domain ontology, making faster judgements whether an entity is worth adding to the ontology. Another reason is related to filtering out the set of already existing terms in the ontology. The more concepts are added to the ontology, the less terms the expert in manual mode has to check, as such terms will be automatically filtered out if they appear again in the set of documents. Another finding coming from the graph, is that the time in semi-automatic mode tends to drop faster than the manual time. This is mainly due to the reason that in the manual mode, the expert will have to check the terms one at a time, thinking about appropriate relations. In the semi-automatic mode, the expert can focus on checking relations, which already provide a degree of context on possible integrations with existing concepts in the ontology. This is also due to the fact that the expert in the semi-automatic process is supported by caching mechanism to hide statements that have been previously discarded. The time graph does not apply to the unsupervised mode, because there was no user involvement in this case.

Figure 6.2 shows the number of concepts added to each ontology version. This reflects that at the beginning (first two days), we see that more concepts are added manually, than semi-automatically and without user involvement. This is because the initial ontology contains gaps in the represented knowledge that the expert can easily identify. However online ontologies can extend the ontology only if they can connect to existing concepts in the ontology.

One thing we were expecting from our experiment is that the noise generated from the domain data sources (i.e., number of irrelevant terms proposed) would have a bigger effect on the expert time taken during manual mode, than on the expert time in the semi-automatic mode. This expectation is based on the idea that the more terms are identified from domain data, the more time the expert in manual mode will have to spend checking the terms. However the data proved us wrong. Contrary to our expectations, the noise has a similar effect on the user time in the semi-automatic mode. To check this observation, we calculate the ratio of concepts added to the concepts discovered from the data (i.e., $Ratio_{concepts} = \frac{Concepts\ added\ to\ ontology}{Terms\ extracted\ from\ domain\ data}$). Figure 6.3 plots the ratio of the expert time, to the $Ratio_{Concepts}$ for each ontology version. To a certain extent, we can see in the graph that there's a correlation between the two cases. This shows that irrelevant terms still have a significant impact on the user time in the semi-automatic mode, even if this impact is reduced in comparison to the user in manual mode.

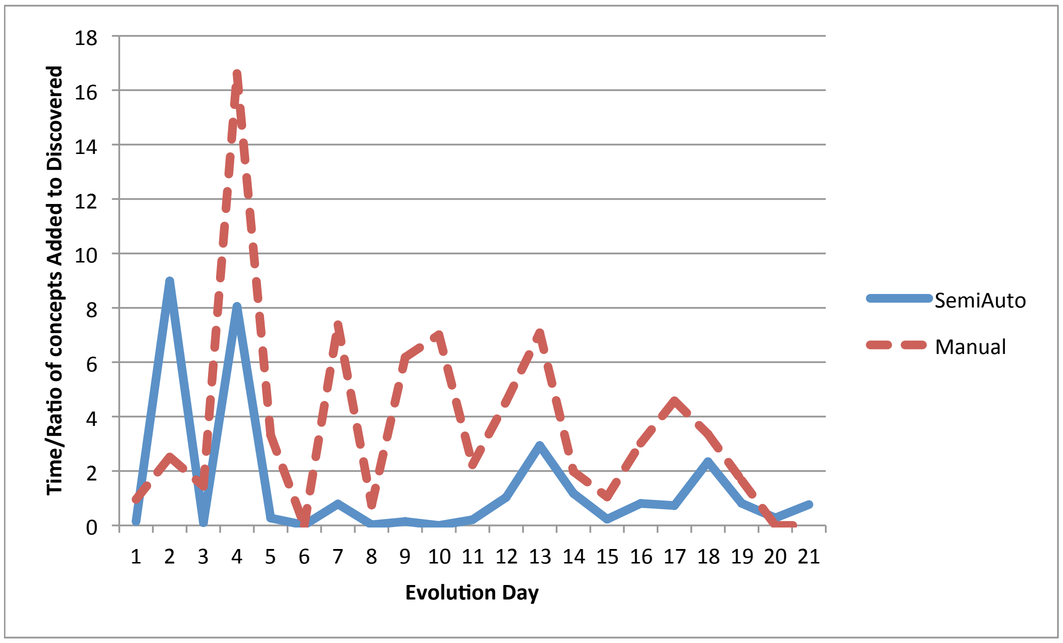


Figure 6.3: Chart showing the ratio of the expert time to the ratio of concepts added to discovered of the semi-automatic mode (full line), versus the manual mode (dashed line).

This is due to the fact that irrelevant terms will result in proposing more relations for the user to check. Even if these relations might be ranked lower by our relevance detection technique, the expert still had to spend some time checking them.

One limitation of our comparison is that we are comparing the times of two different experts, as other elements can contribute to the difference in times, such as user efficiency or experience. However, the reason of choosing

Table 6.1: Comparison of the number of entities added to the final ontologies in the semi-automatic, manual and unsupervised modes

Entity Type Added	**SemiAuto Mode**	**Manual Mode**	**Unsupervised Mode**
Concepts	84	104	1029
Subsumption Relations	91	96	2090
Named Relations	6	19	0

two different experts is that we wanted to compare the performance of the evaluators working on the task from scratch. Because if we require the same expert to do one mode, then run the other mode at a later stage, he/she will already have an idea of potential concepts and relations that should be added to the ontology. Hence, this will favour the second run mode, producing a degree of bias and incomparable results.

6.4.2 Entities Discovered and Added to the Ontology

From the domain data accessed through the RSS feed, a total of 15,609 terms have been identified. From this set, Table 6.1 shows a summary of what has been added to the ontologies. A total of 84 concepts have been integrated in the semi-automatic mode through 97 relations (91 sub-class and 6 named relations), while a total of 104 concepts are integrated in the manual mode through 115 relations (96 sub-class and 19 named relations), and 1029 concepts have been automatically integrated through 2090 subsumption relations in the unsupervised mode. There are no named relations added in the unsupervised mode because the relevance process is based only on sub-class relations, filtering out all named relations. One thing to take into account is the design perspectives that affected the decision of each expert (excluding the unsupervised mode), and subsequently what is added to the ontology. For example, at some point, we realised that the expert in the manual mode started adding concepts related to tenders and offers representation, while this is not the case for the semi-automatic mode, in which the expert considered this area irrelevant, and focused on representing computer and related services products.

6.4.3 Online Ontologies Used to Assess the Relevance of Statements

As discussed in Chapter 4, we provided a technique, through the use of online ontologies, to support users in validating the statements to add to an ontology under evolution. This is mainly aimed to support users in semi-automatic mode, by automatically evaluating the relevance of statements based on the ontological context. Within Evolva, identified relations are ranked based on the relevance confidence calculation, hence placing the most relevant at the top of the list. Even though we performed an evaluation in Chapter 4, we can provide additional evidence regarding the performance of our approach on evolving ontologies by processing sets of documents collected over several days. In this experiment, we log the application of relevance assessment, and what has been actually selected by the expert. This helps answering the following questions: (1) to what extend selected statements are chosen amongst the ones detected as relevant? And (2) by how much the user effort is decreased thanks to relevance detection?

Table 6.2 reveals the tracing of the relevance detection in the semi-automatic mode. It compares the number of sub-class relations discovered, the number classified as relevant, the number selected by the expert, and the number of relations selected classified as relevant. The expert was advised to select relations from the whole set of the proposed statements. A relevant statement in this case is a statement to which a relevance pattern applies, with a confidence value greater than zero.

The results in the table show that the system has classified 57% of the proposed subsumption relations as relevant. Moreover, 81% of the statements selected by the expert were classified as relevant. This means that if the expert was asked to focus only on the 57% supplied by the system, we would get 81% of the taxonomy results selected. Hence with the use of relevance assessment, we could decrease user effort by 43% in extending the taxonomy of the ontology, while keeping 81% of the relevant results.

Table 6.2: Number of Selected Relevant relations by the expert in semi-automatic mode, from the total Selected Relations, with the number of Relevant Relations detected by the tool, from the subsumption relations for each Evolution Day.

Evolution Day	**Subsumption Rel. Discovered**	**Relevant Relations**	**Selected Relations**	**Selected Relevant**
1	23	2	6	2
2	48	13	4	4
3	22	6	3	2
4	37	19	7	6
5	30	12	4	2
6	4	0	1	0
7	165	88	9	6
8	15	8	3	3
9	81	44	15	12
10	60	35	0	0
11	76	43	10	10
12	112	69	4	4
13	113	68	3	3
14	74	41	1	1
15	47	29	2	2
16	86	52	2	1
17	96	66	3	2
18	129	75	1	1
19	99	69	3	3
20	89	60	7	7
21	98	68	3	3
Total	1504	857	91	74
	Average	57%	**Recall**	81%

To check whether there's another potential threshold (other than zero) that could lead to a better selection of relevant relations, we gradually increment the threshold by 0.1, and check the effects on the numbers in the table above. We plot the values traced in the graph of Figure 6.4.

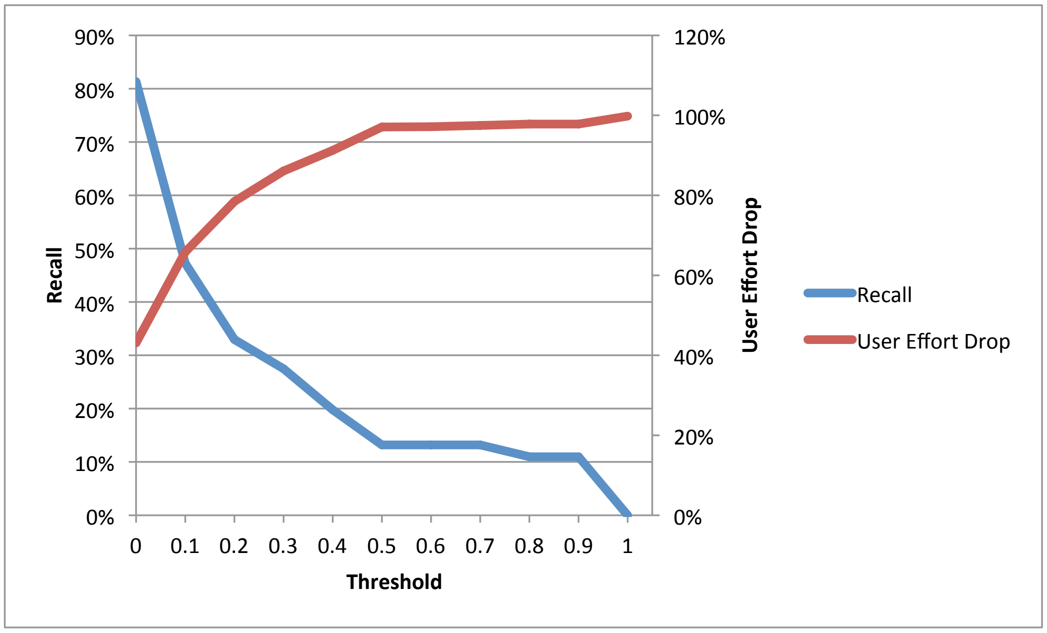


Figure 6.4: Graph of the threshold values effect on the recall and user effort drop.

We observe that once we increased the value of the threshold to 0.1, the user effort can be decreased by 66% (compared to 43% with a zero threshold), however the recall drops to 47% (compared to 81% with a zero threshold). The more the threshold is increased, the less user effort is needed but with a high tradeoff with the recall. As in our scenario we focused on maximising the number of relevant relations to identify (i.e., maximising the recall), the threshold of value zero offered the best trade-off in this case. In the situation where time is favoured, the threshold increase to 0.1 would work well, decreasing the effort of users to 66%, with a 47% recall in this scenario.

6.4.4 Observations and Analysis of the Resulting Ontologies

In order to get a more understanding of the final ontologies generated in the different modes, we analyse the ontologies at two levels. We observe some of the key differences in the ontologies in terms of structure and design, and we apply a set of metrics and analyse to what extent they support our observations.

We select the evaluation metrics from the ontology evaluation community, in addition to some measures that we feel appropriate to our scenario. (Gangemi et al., 2006) define three types of ontology evaluation measures: the *structural dimension* where the analysis of the ontology as a graph is considered, the *functional dimension* where the usage and intended functionality of the ontology is considered, and finally the *usability-profiling* that takes the annotations that define ontology profiles and its metadata into consideration, to address the ontology's communication context.

In our case, the selection of the metrics from the community is based on our goal: to get an insight of the major structural differences between the final ontologies generated in the three different evolution modes. Hence we focus on the *structural dimension* of the ontology. Such measures are not to be taken as absolute indication of quality, but they are meant to provide a relative indication of differences between the three differently generated ontologies and check to what extent they support our initial observations. To compute and generate the metrics, we implemented our own ontology analysis component to work out the elements involved to apply the formulas.

Observation 1

At first glance, we realise that the ontologies produced in the three modes have all been added significantly more roots with respect to the original ontology (see Figure 6.5). In addition, we observe that the number of roots of the

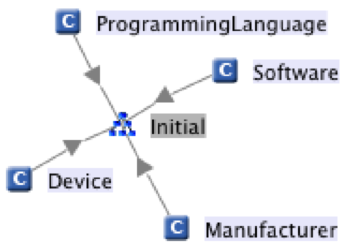


Figure 6.5: Concepts at the root level of the initial ontology.

ontology in the manual mode (see Figure 6.6) is larger than the one created in the semi-automatic mode (see Figure 6.7). Moreover, the ontology generated in the unsupervised mode has substantially more roots (see Figure 6.8) than the other two ontologies. This gives an indication that the expert in

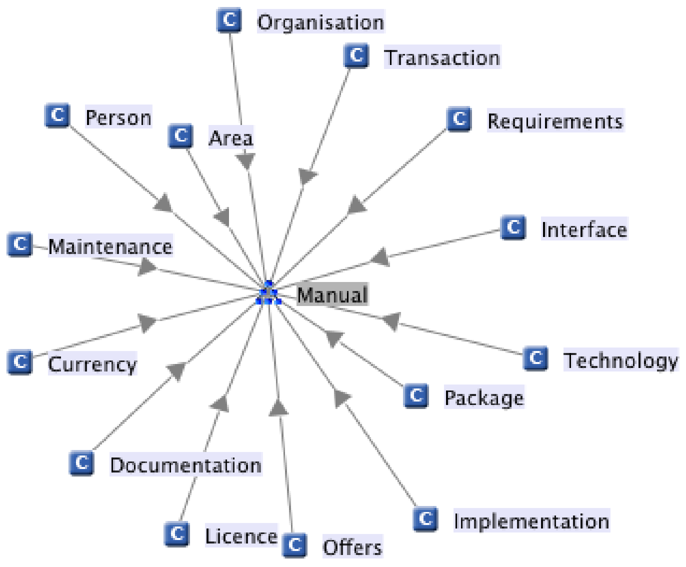


Figure 6.6: Concepts at the root level of the ontology created in the manual mode.

manual mode created more branches touching other contexts than the one in semi-automatic mode, and the unsupervised mode resulted in a more diverse ontology. For example, the expert in manual mode created a "Documentation" root class, which the expert in the semi-automatic mode considered irrelevant. However, in the unsupervised mode, there are concepts such as "Fluid" and "Identity" that would be classified by the experts as irrelevant to the ontology.

Metrics Used

Counting as roots the classes without parents in the ontology, we obtain $Roots_{Initial} = 4$, $Roots_{SemiAuto} = 9$, $Roots_{Manual} = 14$ and $Roots_{Unsupervised} = 41$. The numbers support the observation of the increase in the number of

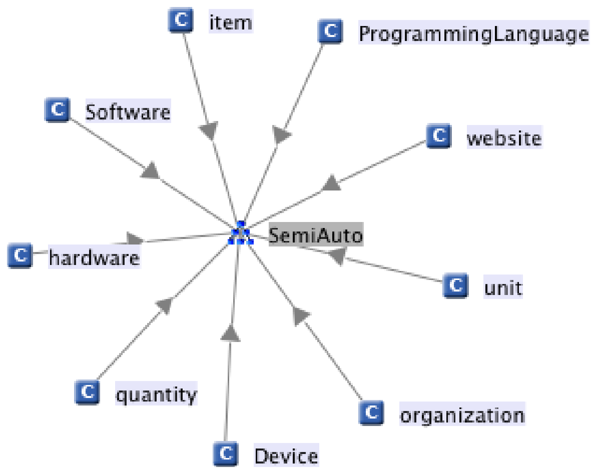


Figure 6.7: Concepts at the root level of the ontology created in the semi-automatic mode.

roots.

Observation 2

Going deeper in the ontologies, we realise that the manually evolved ontology has more hierarchical levels than the ontology generated in the semi-automatic mode. While the one produced in the unsupervised mode has definitely more levels than both. In the latter case, it was harder to check as in some branches of the ontology the path to reach the end was too long.

Metrics Used

We make use of the **average depth** and **maximal depth** metrics (Gangemi et al., 2006) to get a quantitative feedback for our observation.

The **average depth** measure takes into account the number of paths in the ontology to get the average depth value based on this formula:

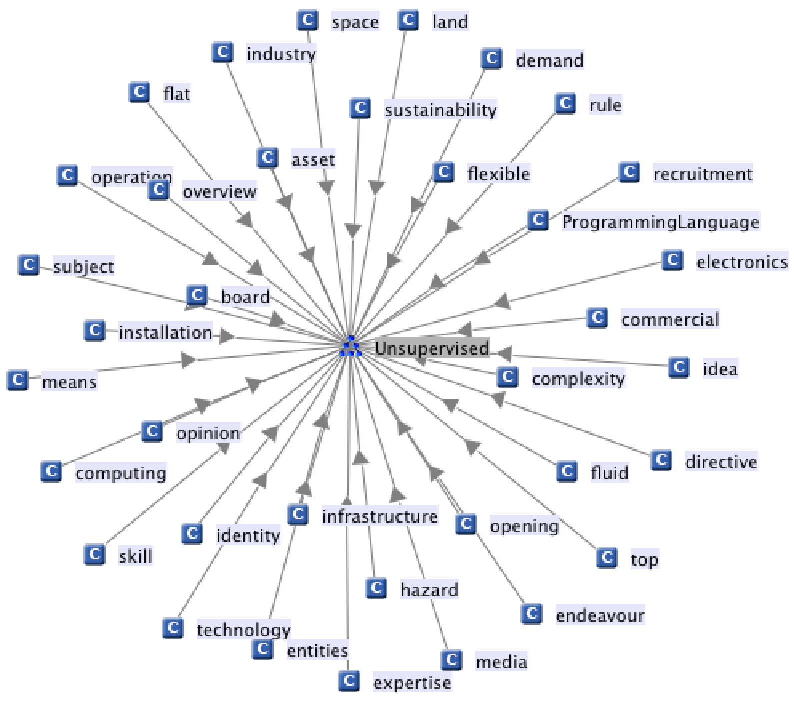


Figure 6.8: Concepts at the root level of the ontology created in the unsupervised mode.

$$AvgDepth = \frac{\sum_{j}^{P} N_{j \in P}}{n_{P \subseteq g}}$$

"where $N_{j \in P}$ is the cardinality of each path j from the set of paths P in a graph g, and $n_{P \subseteq g}$ is the cardinality of P" (Gangemi et al., 2006). Applying the measure in our case, we get: $AvgDepth_{Initial} = 1.75$, $AvgDepth_{SemiAuto} = 1.33$, $AvgDepth_{Manual} = 2.18$ and $AvgDepth_{Unsupervised} = 5.75$. This shows that the semi-automatic ontology depth shrank on average compared to the initial ontology. For the manual mode, we see that the average depth has slightly increased, however it highly increased in the unsupervised mode. While the ontology increased in size in the three modes (as reflected by the number of added entities in Table 6.1), the semi-automatic and manual modes did not focus on making the ontology more specific as the depth did not extend significantly. But in the unsupervised mode, we see a considerable increase in the specificity level of the ontology.

The **maximal depth** selects the longest path in the ontology. Its calculation is based on the following formula:

$$MaxDepth = N_{j \in P} | \forall i \exists j (N_{j \in P} \geq N_{i \in P})$$

"where $N_{j \in P}$ and $N_{i \in P}$ are the cardinalities of any path i or j from the set of paths P in a graph g" (Gangemi et al., 2006). In our experiment, $MaxDepth_{Initial} = 3$, $MaxDepth_{SemiAuto} = 3$, $MaxDepth_{Manual} = 4$ and $MaxDepth_{Unsupervised} = 15$. The values support our observation that the ontology of the manual mode is deeper than the one evolved semi-automatically. This also shows how close the results of the semi-automatic and manual modes are, while the unsupervised mode extended the same taxonomy to a larger depth, confirming the above outcome.

Observation 3

After further investigations while checking the ontology branches, we realise that the taxonomy in the unsupervised ontology differs from the other two modes. Due to some classes having more than one parent in the ontology of the unsupervised mode, (e.g., "Computer" is at the same time a "Device", "Individual", "Item", "Infrastructure", "Electronics" and "Products"), this created a rather complex structure to browse and analyse. In contrast, the other ontologies produced in the manual and semi-automatic modes have a tree shaped structure, with most concepts having one specific parent. We also realise that some concepts in the ontology produced in the semi-automatic mode have more than one parent, but were much fewer than the cases in the ontology of the unsupervised mode.

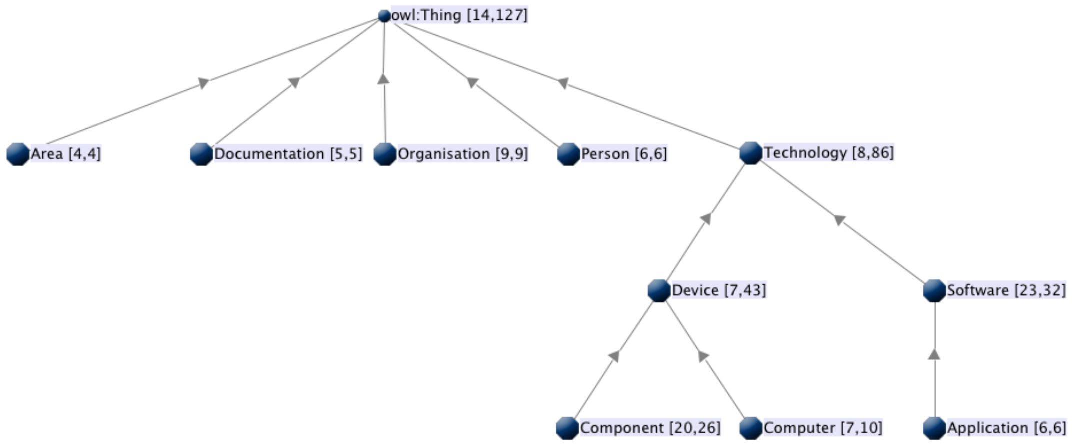


Figure 6.9: Summary view of the ontology evolved in the manual mode showing the 10 key concepts.

Due to the size of the ontologies, especially the one evolved in an unsupervised way, we loaded a summary view of the evolved ontologies using the KC-Viz visualisation tool of the NeOn Toolkit (Peroni et al., 2008). We set the tool to get the 10 key concepts of the three evolved ontologies. Figure 6.9

shows the 10 key concepts of the ontology produced in the manual mode, where the first value next to the concept reflects the number of direct sub-classes, and the second value is the number inferred sub-classes.

Figure 6.10 displays the summary view of the ontology evolved semi-automatically, by showing the 10 key concepts of the ontology. Figure 6.11 shows

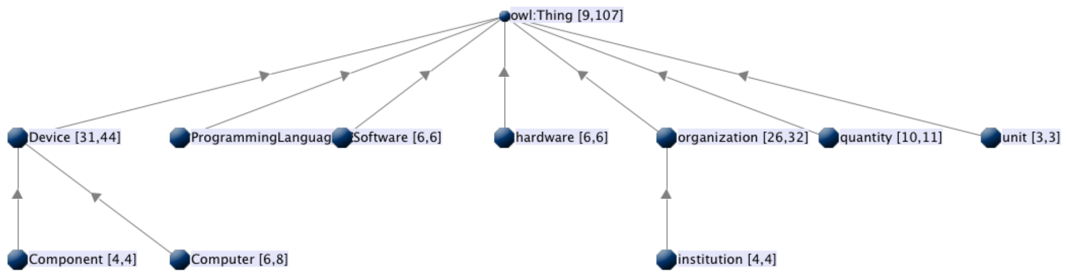


Figure 6.10: Summary view of the ontology evolved in the semi-automatic mode showing the 10 key concepts.

the summary view of the 10 key concepts of the ontology produced in the unsupervised mode. Indeed it is clear that, based on the high level views in Figures 6.9 and 6.10, the structures of the ontologies produced manually and semi-automatically are tree-shaped. However, the graph of the ontology of the unsupervised mode shown in Figure 6.11 reflects the fact that the ontology does not have a clearly defined structure.

Metrics Used

To support this view, we use the **tangledness** measure defined by (Gangemi et al., 2005). This measure takes into consideration the concepts in the ontol-

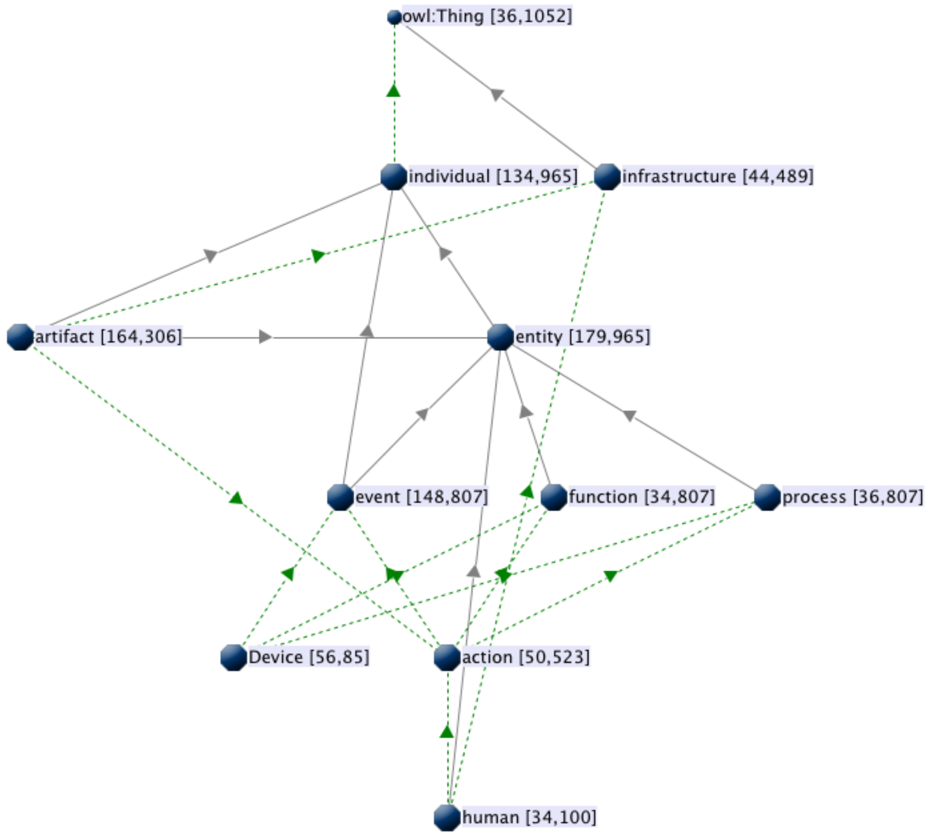


Figure 6.11: Summary view of the ontology evolved in the unsupervised mode showing the 10 key concepts, where the dotted arrows depict inferred sub-class relations.

ogy which have more than one parent. The formula that we use:

$$Tangledness = \frac{t_{\in G \wedge a_1, a_2 (isa(m,a_1) \wedge isa(m,a_2))}}{n_G}$$

"where $t_{\in G \wedge a_1, a_2 (isa(m,a_1) \wedge isa(m,a_2))}$ is the cardinality of the set of nodes with more than one outgoing isa arc in g." Note that we have inverted the formula given in (Gangemi et al., 2005), so that we can interpret it as the ratio of concepts with more than one parent to the total number of concepts in the ontology. This means that when the *tangledness* value is zero, the ontology has a perfect tree structure, with all concepts having at most one parent, while when the value is closer to one means that most concepts within the ontology have more than one parent.

Applying the formula to our ontologies, we get $Tangledness_{Initial} = 0$, $Tangledness_{SemiAuto} = 0.1$, $Tangledness_{Manual} = 0$, $Tangledness_{Unsupervised} = 0.53$. This shows that the initial ontology and the manually generated one are perfect tree structures, where no concepts have more than one parent. In the case of the semi-automatic one, there exist 11 out of 107 classes with more than one super-class. Having a closer look, we realise that three out of eleven classes (Bank, Hospital and College), have redundant classifications that can be inferred. For example, the statements $< Bank, subClass, Company >$, $< Bank, subClass, Organisation >$ and $< Company, subClass, Organisation >$ are all present in the ontology. While the formula treats this as a tangled case, $< Bank, subClass, Organisation >$ is redundant and can be easily removed automatically. After removing such cases, we get $Tangledness_{SemiAuto} = 0.07$. In the case of the unsupervised mode, the measure shows that around 50% of the ontology concepts have more than one parents. This supports our observation that the ontology is highly tangled, resulting with a rather complex and taxonomically confusing ontology structure. Applying a mechanism to detect most of the redundant cases for the ontology in the unsupervised mode, we get $Tangledness_{Unsupervised} = 0.19$, which is still high compared to the size of the ontology. The summary view of the ontology visualised in Figure 6.11 shows the tangledness of the ontology, clearly visible among the key concepts in the ontology. We can deduce at this level that the manual mode kept a very clean and nice ontology structure, and that the semi-automatic mode achieved a very similar result, while the unsupervised mode resulted with a highly tangled ontology.

Observation 4

While it is clear that the most explored concepts in terms of sub-classes are related to "Device" in both semi-automatically and manually evolved ontologies, it was harder to judge for the case of the unsupervised ontology. This is mainly due to the generic concepts being added (e.g., "Individual" and "Entity"), with many sub concepts being attached to them. This is also confirmed by the key concepts view in Figure 6.11, where such generic concepts are among the key ones identified. With the presence of generic concepts, and adding many sub-classes at the same level, the ontology evolved in the unsupervised mode is wider than the other two ontologies. We can see that online ontologies play a positive role in the identification of granular (i.e., sub-classes) concepts in the semi-automatic evolution mode. With well explored concepts in online ontologies (e.g., organisation), it was easy to find corresponding sub concepts to integrate in the ontology evolved in the semi-automatic mode. This made it easier to expand concepts using popular entities in online ontologies, than the user in manual mode having to make all the connections, hence resulting in the ontology evolved semi-automatically to be wider than the one manually evolved. While we can see that some of the specialised concepts (added manually to the ontology) detected from the domain data (e.g., EDRMS - Electronic Document and Records Management System), were harder to find in online ontologies. With more ontologies being published on the web, we anticipate that the conceptual representation will keep increasing with time, which will enhance the relation discovery process.

Metrics Used

The **breadth** metrics fit our target at this level (Gangemi et al., 2005). The **average breadth** measure computes the average breadth per generation within the ontology, using this formula:

$$AvgBreadth = \frac{\sum_j^L N_{j \in L}}{n_{L \subseteq g}}$$

"where $N_{j \in L}$ is the cardinality of each generation j from the set of generations L in a digraph g, and $n_{L \subseteq g}$ is the cardinality of L." In our experiment, $AvgBreadth_{Initial} = 5.75$, $AvgBreadth_{SemiAuto} = 26.75$, $AvgBreadth_{Manual} = 25.4$ and $AvgBreadth_{Unsupervised} = 65.75$. This confirms what we observed earlier (Observation 2) that the ontology produced semi-automatically is on average flatter than the ontology produced manually, while staying close to each other in terms of structure compared to the unsupervised ontology.

The **maximal breadth** selects the widest generation available in the ontology:

$$MaxBreadth = N_{j \in L} | \forall i \exists j (N_{j \in L} \geq N_{i \in L})$$

"where $N_{j \in L}$ and $N_{i \in L}$ are the cardinalities of any generation i or j from the set of generations L." $MaxBreadth_{Initial} = 9$, $MaxBreadth_{SemiAuto} = 81$, $MaxBreadth_{Manual} = 36$ and $MaxBreadth_{Unsupervised} = 515$. The values show that the semi-automatic mode resulted in an ontology which is more expanded horizontally than the manually evolved ontology, also supporting our previous findings.

Observation 5

In terms of the relations that exist in the ontologies, we can clearly see from browsing the ontologies that more named relations have been added to the manually created ontology than the one of the semi-automatic mode. This is mainly due to the fact that the expert, relying on his background knowledge, tends to more easily find connections between concepts, than online ontologies, which in some cases, do not contain very granular relations.

Metrics Used

We rely on the **relationship richness** and **inheritance richness** metrics (Tartir et al., 2005) to get an insight of the differences at the level of relations. The relationship richness is based on the (RR) formula:

$$RR = \frac{|P|}{|SC| + |P|}$$

which is the ratio of named relations (P) to the sum of sub-class relations (SC) and (P). This would give an idea of the *richness* of named relations with respect to all the relations in the ontology, showing the degree of diversity in relationships in the ontology. In the case of the initial ontology, $RR_{Initial} = \frac{1}{19+1} = 0.05$. Applying the formula on our ontologies, we get $RR_{SemiAuto} = \frac{7}{110+7} = 0.06$ for the semi-automatic mode, and $RR_{Manual} = \frac{20}{115+20} = 0.15$ for the manual mode. The values show that in both modes, the number of sub-class relations added is significantly higher than the number of named relations. This also shows that the ontology evolved manually tends to be more diverse in terms of relations, than the ontology produced semi-automatically, as per our observation. In the case of unsupervised evolution, we cannot deduce the (RR) as selected relations do not include named relations $(P = 0)$, leading to $RR_{Unsupervised} = 0$.

The second metric we employ is the inheritance richness measure showing the "average number of sub-classes per class" (Tartir et al., 2005) in the ontology. This gives an indication whether the ontology tends to be more flat or vertical. The larger (IR) is, the more horizontal the ontology is. While the lower it is, the more vertical it is. The formula (IR) for the inheritance richness is:

$$IR = \frac{\sum_{C_1 \in C} |H^C(C_1, C_i)|}{|C|}$$

where "the number of sub-classes (C_1) for a class (C_i) is defined as $|H^C(C_1, C_i)|$".

Applying this formula to our context, we get $IR_{Initial} = \frac{19}{23} = 0.83$, $IR_{SemiAuto} = \frac{110}{107} = 1.03$, $IR_{Manual} = \frac{115}{127} = 0.91$ and $IR_{Unsupervised} = \frac{2109}{1052} = 2$. This means that we have more sub-classes per class in the ontology generated semi-automatically than the one generated manually, while it is around the double in the unsupervised mode. This confirms as well our previous observations that the semi-automatically evolved ontology expanded slightly more horizontally that the ontology in the manual mode.

Table 6.3: Metric-based values, applied to the initial and resulting ontologies in the three evolution modes.

Measure Applied	**Initial Ontology**	**SemiAuto Mode**	**Manual Mode**	**Unsupervised Mode**
Total Roots	4	9	14	41
Average Depth	1.75	1.33	2.18	5.75
Maximal Depth	3	3	4	15
Tangledness	0	0.07	0	0.19
Average Breadth	5.75	26.75	25.4	65.75
Maximal Breadth	9	81	36	515
Relationship Richness	0.05	0.06	0.15	0
Inheritance Richness	0.83	1.03	0.91	2

Table 6.3 shows an aggregated view of the metrics used in our structural ontology analysis. Such metrics show that the ontology evolved in the semi-automatic mode did not diverge much from the ontology that evolved manually. However, the ontology that evolved in the unsupervised mode expanded in an uncontrollable way. The graph in Figure 6.12 puts in perspective the average depth and breadth, tangledness and inheritance richness of the initial ontology, compared to the three ontologies of the different evolution modes. We take the average depth and breadth metrics as they reflect more the overall shape of the ontology, hence reducing the anomalies created by certain ontology branches. We did not integrate the relationship richness in the graph as the unsupervised mode did not produce named relations during the evolution of the ontology. Indeed, the resulting graph shows that the characteristics of the ontologies evolved manually and semi-automatically are very close, clearly expanding the initial ontology. In contrast, the unsupervised evolution mode produced an ontology that increased in the four dimensions at a much higher scale, compared to the other two modes.

6.5 Discussion

In this chapter, we evaluated the impact of the use of online ontologies as background knowledge sources to support users in the process of ontology evolution. This evaluation puts the feasibility of our proposed approaches

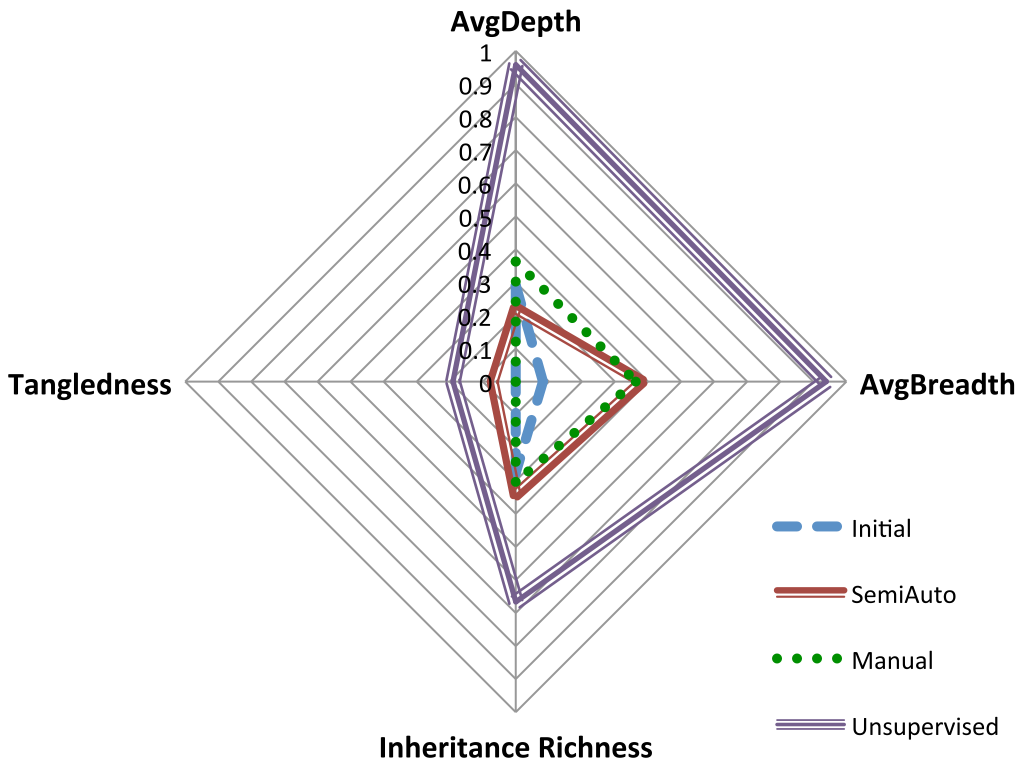


Figure 6.12: Graph comparing the average depth and breadth, tangledness and inheritance richness metrics of the four ontologies.

combined to the test. For that, we chose to study and analyse the evolution of an ontology in the computer and services domain. We collected the domain data, and setup an experiment to evolve an ontology in three different modes: a manual mode where no support is provided to the user to integrate new terms to the ontology; a semi-automatic mode where online ontologies are used to support the user during the evolution; and an unsupervised mode where no user input is involved.

We recorded the time and logged all entities added to the ontology in the three different modes, and used the data to get a direct feedback of the main differences between the evolved ontologies. The data representing the recorded time showed an overall decrease in the time spent by the expert using online ontologies, compared to the time spent by the expert in manual mode.

With the challenges imposed to perform a reasonable ontology evaluation, we used a mix of empirical observations through browsing and visualising the ontologies, backed by a set of ontology evaluation metrics (focusing on the structural measures). This combination showed that the ontology that evolved semi-automatically did not diverge significantly from the one that evolved manually, while the one evolved in the unsupervised mode created a much more complex ontology. In other words, no major issues were introduced by the use of online ontologies in the semi-automatic evolution mode. While the use of online ontologies in the unsupervised mode lead to an explosion in adding new concepts to the ontology, due to the integration of some generic terms at some point during the evolution, leading to a more tangled and complex ontology. Our ontology analysis coupled with the time logging, therefore support our initial hypothesis: "the use of online ontologies through the Evolva tool can substantially decrease the effort required by users to evolve ontologies, with little negative effect on the quality of the resulting ontology compared to a manually evolved one."

Chapter 7

Conclusion

This thesis had three main contributions to the field of ontology evolution. Firstly, we devised a technique to automatically identify potential ontology changes by integrating new emerging concepts from external domain data, based on the background knowledge provided by existing structured sources, discussed in Chapter 3. Secondly, we provided a novel approach to automatically assess the relevance of potential ontology changes by analysing the contexts of online ontologies from where they are derived, versus the ontology under evolution. This approach was presented and evaluated in Chapter 4. Thirdly, we put together a methodology for evaluating and comparing the structural differences between ontologies under evolution, through a combination of empirical observations supported by metric based assessment techniques. This contribution came as a result of our work in Chapter 6, where we employed this methodology as part of the overall evaluation of our proposed approaches using the Evolva tool described in Chapter 5.

Next we recall our proposed approaches that answer the research questions raised in this dissertation. Then we discuss and identify potential ways of improving these approaches and overcoming some of their limitations, before we finally close with potential future directions in the field of ontology evolution.

7.1 Approaches Revisited

With ontology evolution being a tedious and time-consuming task, we aimed in this thesis to answer the following research question that we initially presented in Chapter 1:

How to support users in the process of ontology evolution?

Our literature review conducted around the tasks involved in the complete ontology evolution cycle (Chapter 2), highlights some of the gaps that are hampering the process of ontology evolution. Based on the existing gaps, we identify two sub-questions that we tackled along this thesis.

One of the identified gaps raises the issue of identifying new domain changes, which connect appropriately with existing knowledge in the ontology. Hence the first sub-question that we deal with is: *how to assist users in identifying ontology change opportunities?*

We presented in Chapter 3 our approach based on the use of lexical databases and online ontologies, to automatically derive statements that link newly identified domain concepts to existing ontology concepts. Such new concepts can be derived from external domain data such as text corpora or RSS feeds. While lexical databases provide support in doing this task, they only partially fulfil the requirements, as the statements that can be explored are mainly sub-class relations, in addition to the fact that the knowledge does not get updated frequently. Distinctively, online ontologies provide a richer source of relations that can be of any type including subsumption, named and disjoint relations. With the use of Semantic Web gateways such as Watson, new knowledge becomes available as soon as a new ontology is published and crawled by the gateway. The use of such tools to consume and reuse available knowledge on the web is a good sign towards the maturity of the existing Semantic Web tools. We devise a technique to process WordNet in order to find the list of sub-class relations to link new concepts to the ontology's concepts. In the case of online ontologies, we reuse the relation discovery engine Scarlet, to identify statements through Watson. In our tests, we process WordNet first, then online ontologies for concepts not found in WordNet. This resulted in limiting the relations of concepts found in WordNet to sub-class relations only. This issue can be addressed by giving the ontology engineer the choice of processing all concepts through both WordNet and online ontologies at the tool level.

Our evaluation proved the feasibility of using external sources as background knowledge to ontology evolution, with an average precision of 77%. With this approach in place, the user input at the level of identifying domain changes, and integrating these changes in the ontology is reduced. This validated the extent of our first contribution to provide a mechanism to automatically identify potential ontology changes opportunities, starting from concepts detected in external data sources, supported by the background knowledge supplied by structured sources.

Our experiment shows that automatically identifying ontology changes tends to generate a lot of statements, among which a good portion are irrelevant to the ontology under evolution. This leads to the user having to do additional efforts at the level of evaluating the changes before adding them to the ontology. Hence our second research question is: *how to assess the relevance of ontology changes?*

Our study, presented in Chapter 4, shows that the contexts supplied by online ontologies from which relations are derived can play a role in assessing the relevance of a statement with respect to an ontology. After further investigations, we realise that relevance can be derived by performing an overlap between such contexts and the ontology under evolution. However, with the presence of large ontologies online (e.g., cyc[1]) that can cover a wide range of

[1]`http://www.cyc.com`

contexts, the relevance calculation can be misleading. Hence we proposed a new pattern-based approach that takes into account the structure surrounding the new statement to add, with what is shared between the online ontology and the ontology under evolution. Each pattern has specific application conditions, and a corresponding confidence measure, based on which the new statements to add are ranked, enabling the users to focus on the top ranked relevant statements. Our experiment shows that our pattern-based relevance detection approach outperforms the overlap-based technique, in both detecting relevant and irrelevant statements. The F-measure of the overlap relevance tested on three different domains is in the range of [7.41%, 58.06%], compared to the higher [43.75%, 69.05%] F-measure range based on the pattern technique. In the case of irrelevance, the overlap F-measure range of [60.87%, 85.71%], is also lower than the pattern-based F-measure performance range of [74.74%, 92.48%] across the three domains. The efficiency of our relevance approach is also supported by the overall evaluation we conducted in Chapter 6, which shows that user effort can be reduced by 43%, while preserving 81% of the relevant statements. This study reflects the potentials behind our second research contribution.

In addition to finding approaches that tackle specific problems in ontology evolution, we considered in this dissertation the integration and realisation of our proposed approaches into a coherent and usable system, in order to use the implemented system to perform an overall evaluation of our solutions and proposed techniques. In Chapter 5 we presented an ontology evolution framework to handle: (a) discovering information from external domain data, (b) validating the discovered information, (c) identifying the appropriate ontology changes by relying on background knowledge sources, (d) validating the evolution and finally (e) managing it. On top of this generic framework, we built an ontology evolution tool, Evolva, in which we implement our key research outcomes. Evolva is available for download within the NeOn Toolkit. It enables users to load an ontology, and initiate the evolution starting from text corpora, RSS feeds, or a raw list of terms. The identified terms from the sources are then cleaned and the list of new terms that do not exist in the ontology is returned to the user. In the relation discovery component, Evolva identifies the appropriate relations that link the new concepts to existing ones in the ontology (based on our work in Chapter 3), and ranks the relations according to their relevance (based on our work in Chapter 4). The user has the option to check the statements, based on which a list of ontology changes is generated. The changes are then used to create a new version of the ontology, which is directly accessible within the NeOn Toolkit.

In order to get an insight into the usability of Evolva, and of the overall effectiveness of our approaches relying on the use of online ontologies as background knowledge for ontology evolution, we conduct an experiment presented in Chapter 6. We performed this experiment in the computing and related services domain over an ontology that we developed for the task, and collected data over a 30 day period from the RSS feed provided by the "EU Tenders Electronic Daily" (TED) portal. We setup three ontology evolution modes: manual in which the user is the only source of background knowledge, semi-automatic mode in which another user is assisted by online ontologies to identify and evaluate ontology changes, and unsupervised mode where the evolution is performed automatically without user input. We log and analyse the time taken by users in the process with and without using the online ontologies as background knowledge, in addition to recording the elements added to the ontologies in the three modes. Furthermore, we analyse and compare the ontologies generated, supported by metric-based measures to get an idea of the key differences between the resulting ontologies. Our main conclusions out of our experiment are:

- **User time.** We noticed a tremendous decrease in time (around 8 times less) for the ontology engineer who relied on online ontologies, versus the other ontology engineer who is manually adding new elements to the ontology.

- **Resulting ontologies.** Based on our observations and ontology evaluation metrics, we realised that the ontologies produced manually and semi-automatically are very close in terms of structure and quality features. However, the ontology produced in the unsupervised mode is much more complex. This is mainly due to the fact that highly abstract concepts, which were added throughout the evolution stages, attracted many statements that created inappropriate connections between concepts at different levels of the ontology.

Our findings support our initial hypothesis that the use of online ontologies can substantially reduce user efforts both at the levels of identifying new ontology changes and evaluating the changes, while producing results of comparable qualities to the manual approach.

7.2 Discussion

As proved by the experiments we performed throughout this thesis, our approaches can contribute to assisting users in the process of ontology evolution. However, there are obviously some limitations and further room for improvements that we highlighted previously in different sections in the thesis. We revisit some of them which we believe can bring additional enhancements to our techniques:

- **Handling other types of ontology changes.** We focus in our approach on adding new elements to the ontology. However there exist other types of changes that occur during ontology evolution such as deletion or modification of existing concepts. Different approaches can be introduced to handle such operations. For example, the work on belief revision applied on ontology evolution can pick up change suggestions based on new information that become available, while keeping the ontology consistent (Flouris, 2006). This could enable dealing with cases where the domain knowledge modelled in an ontology changes (e.g., Pluto ceasing to be classified as a main planet, but as a dwarf planet). Other approaches propose removing ontology elements based on usage analysis. The removal of parts in this case aims to increase efficiency by shrinking the ontology to a more "fit for purpose size" (Alani et al., 2006).

- **Online ontologies availability.** Our techniques heavily depend on what's available and accessible online in terms of structured knowledge. This is considered one of the strongest features, as our techniques will continuously improve with the constant increase of availability of online ontologies. However, this can be a drawback in certain situations, where for example very specialised domains do not have relevant ontologies available online yet. A potential solution to tackle this limitation is by providing the users with the ability to specify their own private background knowledge sources by using Cupboard (d'Aquin and Lewen, 2009) for example, in addition to the ones that can be available online.

- **Tuning the relevance assessment technique.** Even though we set the relevance pattern thresholds empirically in Chapter 4, we later proved in Chapter 6 that the technique still contributes, to a large extent, to decreasing user input when these thresholds are set to the default values. However, for our technique to work optimally, the user will have to perform a high degree of tuning per domain where the ontology evolution is performed. A potential way to decrease this level of tuning is by devising an automatic threshold detection mechanism. This can be achieved for example by using the existing statements in the ontology as a base case of relevant statements, and computing the relevance thresholds accordingly.

- **Improving the efficiency of the process.** Accessing and processing online ontologies are time consuming tasks. Our approaches can benefit from further optimisation techniques. For example, caching of ontologies can be used to improve the access times, when the same ontology is accessed repetitively. In addition, efficiency can be improved through introducing clustering mechanisms at the level of Watson, and limit the relation discovery and assessment processes to ontologies in the cluster relevant to the ontology under evolution. For example, when evolving an ontology in the music domain, Watson can respond with ontologies that are evaluated a priori as relevant to this domain, making the process of retrieving and processing these ontologies faster and more efficient.

7.3 Closing Notes and Future Directions

We believe that research around the field of ontology evolution will continue to thrive. However, we foresee that the research directions will not only be limited to the tasks directly involved in the evolution process, but will also extend to further external areas. We discuss here some potential future research directions.

- **From ontology evolution to a meta-domain evolution.** We foresee that ontology evolution will have an impact on the analysis of trends resulting from the evolution of domains. Instead of relying on the analysis of text documents, ontology evolution can enable the tracing of trends at a more granular level, based on the conceptual evolution of domains and their subsequent connections. For example, the evolution of the advances in the computer and communication industry can be based on tracing what has been added in terms of concepts and relations to the ontology in focus, and at which period of time. This would enable a more structured analysis, which can make such tasks easier for example to businesses or news industries.

- **Supporting the knowledge backbone of smart systems.** An additional angle to look at the evolution of knowledge is from the perspective of smart systems, which purpose is to solve task-based problems. For example, expert systems provide user support in specific domains, based on questions asked by end users. Such systems have to be frequently updated based on new facts that need to be fed into the knowledge base in the appropriate manner. This requires special techniques in capturing and modelling knowledge not only based on the data connections at the conceptual level, but also taking into account the facts and task-based requirements.

- **Long term use of Evolva and impact on Semantic Web knowledge availability.** Another interesting direction to consider is the analysis of the long term use of Evolva, and the evaluation of its impact on the vision of the Semantic Web. Having Evolva assisting users to rely on existing domain data to evolve ontologies by reusing available knowledge, could have an effect on at least two levels: firstly, it would contribute to disseminate more Semantic Web knowledge by moving from documents to information and data; secondly, by reusing existing knowledge, it is contributing towards a coherent data integration mechanism. These two effects are part of the core requirements to achieve the aims of the Semantic Web (Shadbolt et al., 2006). One possibility to achieve this is by connecting Evolva to various domain related sources to evolve ontologies, and publishing the resulting ontologies online through Semantic Web gateways such as Watson. Having the ontologies as part of the online gateways creates the ability to monitor the growth of knowledge in the domain in focus with respect to other existing domains, hence having direct feedback on the potential impact created by Evolva. This can be achieved through the use of specific metrics such as cluster growth around concepts, or the absolute increase in published knowledge in a certain domain. Furthermore, a high level view of existing ontologies in the gateway could help highlighting the expansion in focus before and after the use of Evolva to evolve specific domain ontologies.

Appendix A

Evolva User Manual

In this appendix, we provide guidelines for installing and using Evolva in an applied scenario.

A.1 Installation

As mentioned in Chapter 5, Evolva is implemented as a plugin for the NeOn Toolkit. Its installation is performed directly from within the toolkit.

A.1.1 Prerequisites

To install Evolva, the following list of requirements are needed:

– **The NeOn Toolkit:** The latest version of the NeOn Toolkit is available for download from the toolkit website[1]. At the time of writing, Evolva

[1]`http://neon-toolkit.org/wiki/Download`

was running on the latest version of the NeOn Toolkit v2.5[2]. The most up-to-date information about the toolkit compatibility can be found on Evolva's website[3].

– **WorNet 2.0 (optional):** Even though Evolva can run by relying only on online ontologies, the user has the option to use WordNet for the relation discovery process. If this is required, WordNet can be downloaded from the download section of its website[4]. We performed our tests on the dictionary version 2.0 of WordNet.

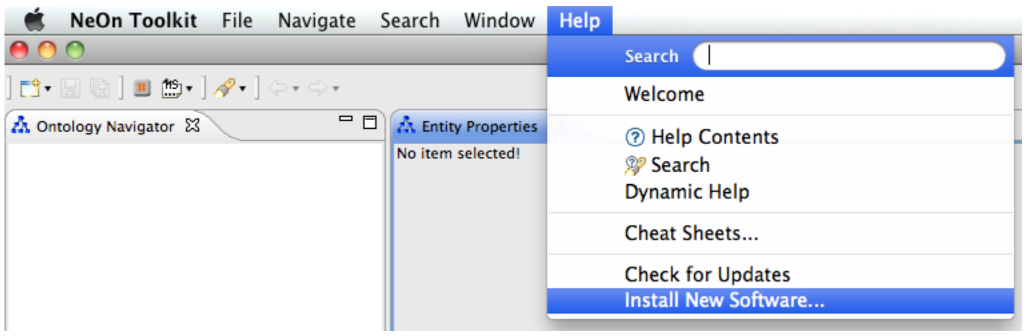


Figure A.1: Installing Evolva within the NeOn Toolkit.

A.1.2 Installation Steps

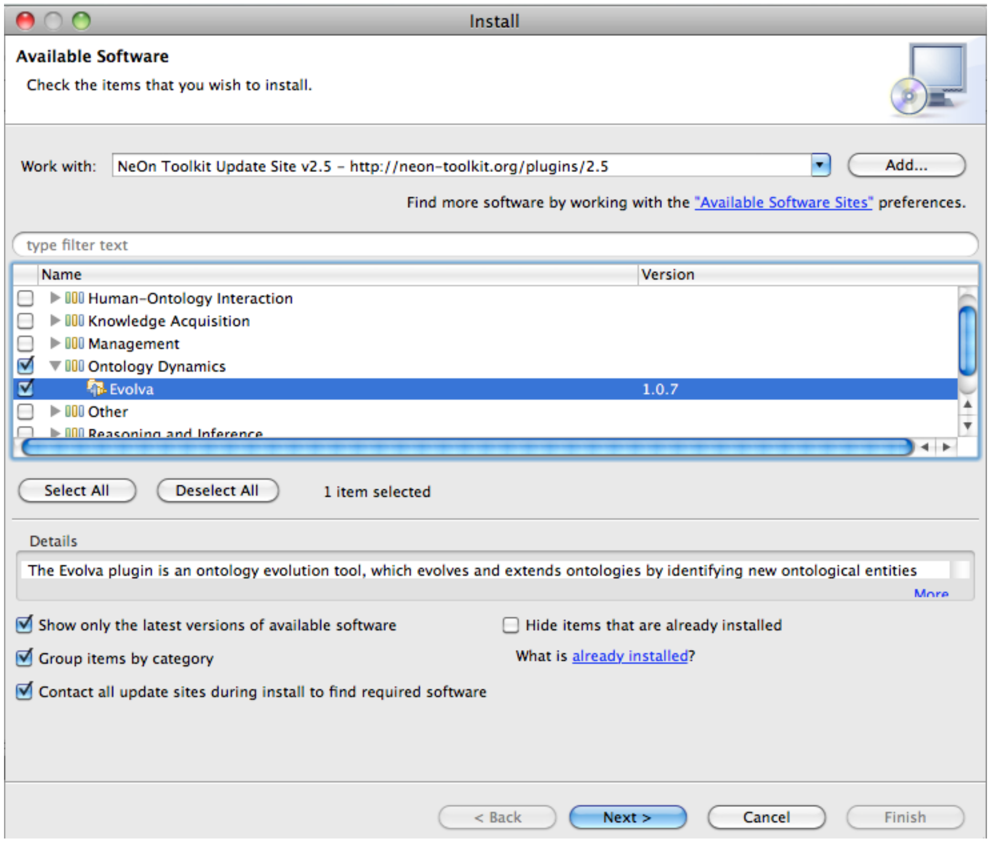
After fulfilling the above prerequisites, Evolva can be installed following the below steps:

[2]`http://neon-toolkit.org/wiki/Download/2.5`

[3]`http://evolva.kmi.open.ac.uk`

[4]`http://wordnet.princeton.edu/wordnet/download`

1. After installing and launching the NeOn Toolkit, from the toolbar, the user should go to ``Help >> Install New Software...'' (see Figure A.1).


Install

Available Software

Check the items that you wish to install.

Work with: NeOn Toolkit Update Site v2.5 - http://neon-toolkit.org/plugins/2.5

Find more software by working with the "Available Software Sites" preferences.

type filter text

	Name	Version
☐	▶ Human-Ontology Interaction	
☐	▶ Knowledge Acquisition	
☐	▶ Management	
☑	▼ Ontology Dynamics	
☑	Evolva	1.0.7
☐	▶ Other	
☐	▶ Reasoning and Inference	

1 item selected

Details

The Evolva plugin is an ontology evolution tool, which evolves and extends ontologies by identifying new ontological entities

☑ Show only the latest versions of available software

☐ Hide items that are already installed

☑ Group items by category

What is already installed?

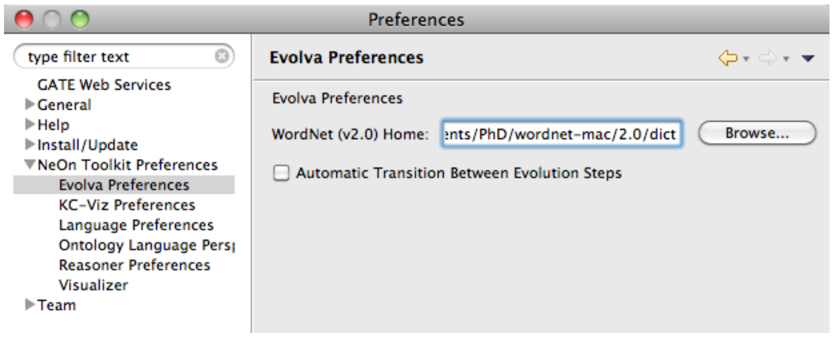
☑ Contact all update sites during install to find required software

Figure A.2: Selecting Evolva from the list of NeOn Toolkit plugins.

2. Then the ``NeOn Toolkit Update Site vX.X'' option should be selected, followed by choosing ``Evolva'' below the ``Ontology Dynamics'' heading (see Figure A.2), and proceeding with the screen instructions. The *Gate Web Services* plugin (required by Evolva's information

discovery step) will be automatically installed at this level. After the installation is complete, the user will be asked to restart the NeOn Toolkit.

3. It is possible to setup the generic Evolva preferences (optional) by going to `‘‘NeOn Toolkit Preferences >> Evolva Preferences’’` (see Figure A.3). At this level, the path to the WordNet dictionary has to be specified. In addition, depending on the scenario, users can choose to have an automatic transition between the evolution steps after specifying the domain data.



Evolva Preferences

WordNet (v2.0) Home: nts/PhD/wordnet-mac/2.0/dict

☐ Automatic Transition Between Evolution Steps

Figure A.3: Generic Evolva preferences.

A.2 Use-case Scenario

In this section we present Evolva applied in a specific scenario. Consider we need to evolve the academic related Semantic Web for Research Communities

(SWRC) ontology[5], by identifying potential terms from the Knowledge Media Institute (KMi) news feed[6].

A.2.1 Loading the Ontology and Starting Evolva

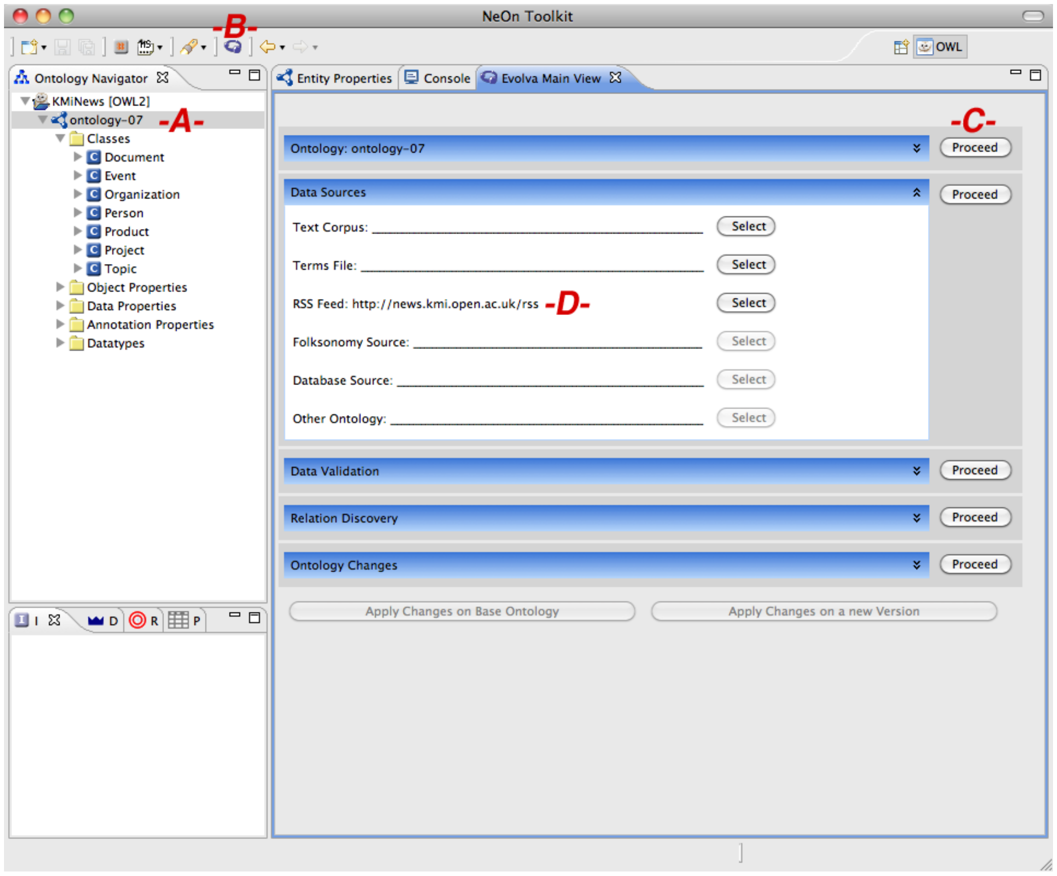
The NeOn Toolkit provides the option to either import ontologies from the web, or load it from a file accessible locally from the system. After completing this task, the ontology can be accessed and browsed through the ontology navigator section (see Figure A.4 (-A-)). After selecting the ontology, Evolva can be launched by pressing on the Evolva button in the toolbar (see Figure A.4 (-B-)). The view of Evolva will appear on the right window panel, and consists of five steps that we follow below.

A.2.2 Providing the Domain Data

In the *Ontology* step (the first bar of Evolva's steps in Figure A.4), the user has the option to select the concepts of the ontology to consider for evolution (i.e., the concepts which we want the new terms to potentially link to). For this scenario, we kept all the ontology concepts selected, and pressed on the *Proceed* button (see Figure A.4 (-C-)). In the *Data Sources* step, we enter the KMi news RSS feed address, from where news items will be pulled (see

[5]The ontology is available at: `http://ontoware.org/swrc/swrc/SWRCOWL/swrc_updated_v0.7.1.owl`

[6]`http://news.kmi.open.ac.uk/rss`



Text Corpus: ______

Terms File: ______

RSS Feed: http://news.kmi.open.ac.uk/rss

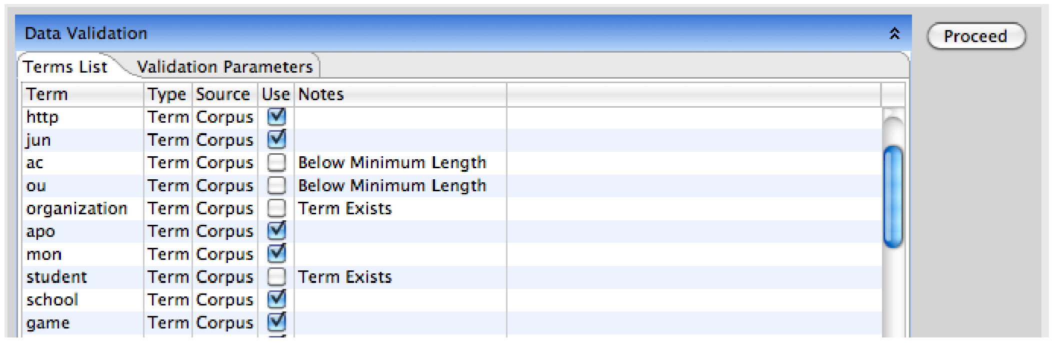
Folksonomy Source: ______

Database Source: ______

Other Ontology: ______

Figure A.4: Starting Evolva and supplying domain data sources.

Figure A.4 (-D-)). In the case where a *text corpus* is needed, users should specify the folder location where the text documents are stored. In the case of *terms file*, users should enter the location of the file containing one term on each line to take into account. The latter case is useful for example when users already have a list of terms identified by other entity extraction tools.



Term	Type	Source	Use	Notes
http	Term	Corpus	✓	
jun	Term	Corpus	✓	
ac	Term	Corpus	☐	Below Minimum Length
ou	Term	Corpus	☐	Below Minimum Length
organization	Term	Corpus	☐	Term Exists
apo	Term	Corpus	✓	
mon	Term	Corpus	✓	
student	Term	Corpus	☐	Term Exists
school	Term	Corpus	✓	
game	Term	Corpus	✓	

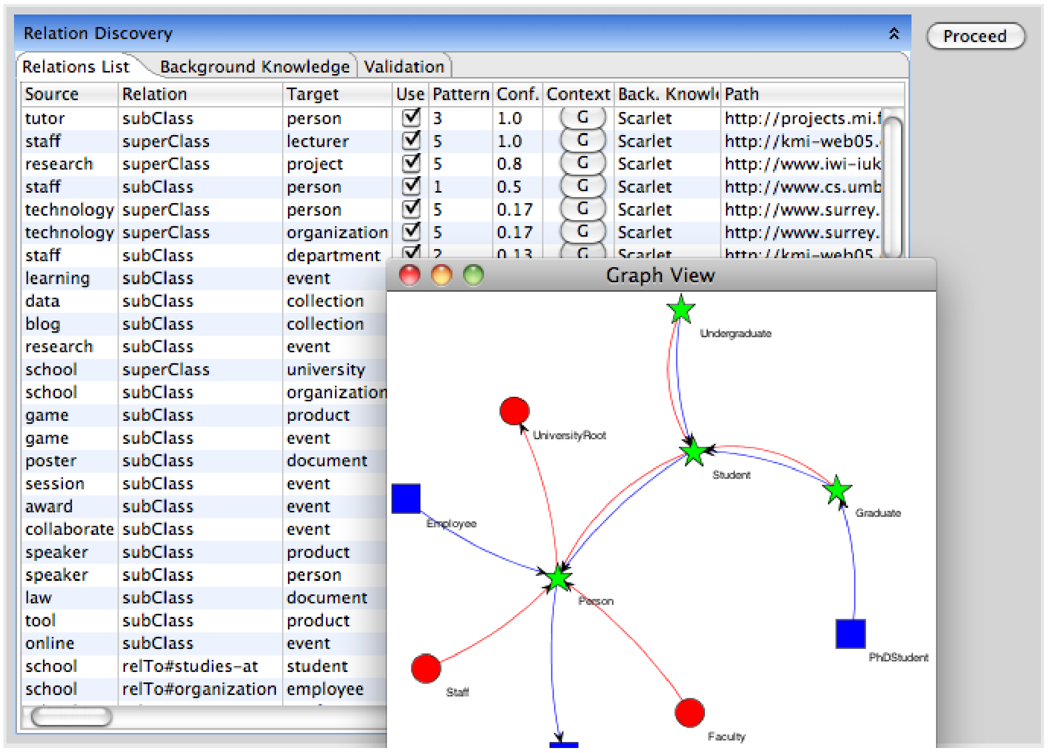
Figure A.5: Snippet of identified terms from the KMi news RSS feed.

A.2.3 Validating Extracted Terms

In the *Data Validation* step, the list of identified terms from the data sources is displayed. The automatic validation process based on the length of terms and whether the term already exist in the ontology, unselects the terms to avoid taking them to the next step (see Figure A.5). In this scenario, we keep all the other terms selected.

A.2.4 Performing Relation Discovery

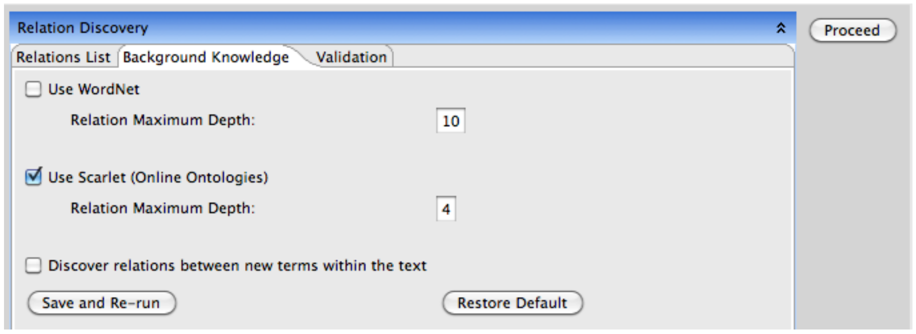
After getting the list of terms, the relation discovery step is for identifying the potential relations that link the new terms, to existing concepts in the ontology. Figure A.6 shows part of the list of relations in our scenario. For each relation, the user can check the following: the source concept (i.e., new concept retrieved from the data source), the relation type, the target concept (i.e., concept in the ontology), the option to select it, the relevance pattern detected, the relevance



Source	Relation	Target	Use	Pattern	Conf.	Context	Back. Knowl	Path
tutor	subClass	person	✓	3	1.0	G	Scarlet	http://projects.mi.f
staff	superClass	lecturer	✓	5	1.0	G	Scarlet	http://kmi-web05.
research	superClass	project	✓	5	0.8	G	Scarlet	http://www.iwi-iuk
staff	subClass	person	✓	1	0.5	G	Scarlet	http://www.cs.umb
technology	superClass	person	✓	5	0.17	G	Scarlet	http://www.surrey.
technology	superClass	organization	✓	5	0.17	G	Scarlet	http://www.surrey.
staff	subClass	department	✓	2	0.13	G	Scarlet	http://kmi-web05
learning	subClass	event						
data	subClass	collection						
blog	subClass	collection						
research	subClass	event						
school	superClass	university						
school	subClass	organization						
game	subClass	product						
game	subClass	event						
poster	subClass	document						
session	subClass	event						
award	subClass	event						
collaborate	subClass	event						
speaker	subClass	product						
speaker	subClass	person						
law	subClass	document						
tool	subClass	product						
online	subClass	event						
school	relTo#studies-at	student						
school	relTo#organization	employee						

Figure A.6: Example of potential relations that link new terms to existing concepts in the ontology.

confidence, the option to visualise the graph by pressing the *G* button (e.g., the graph displayed on top in Figure A.6), the background knowledge used (i.e., Scarlet for online ontologies or WordNet), and the relation path details. By default, online ontologies are selected as the source of background knowledge. However if WordNet is needed, the user can specify so in the *Background Knowledge* tab (see Figure A.7), where also the maximum length of relations to discover can be set. The relations are ranked based on the relevance confidence value. The weight of patterns can be adjusted, as well as the settings for the



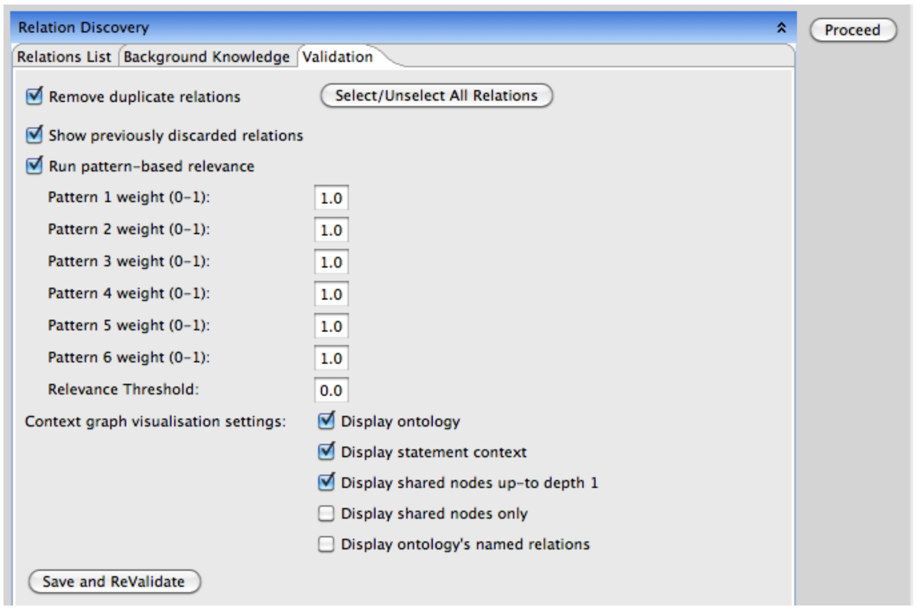
[ ] Use WordNet
Relation Maximum Depth: 10

[✓] Use Scarlet (Online Ontologies)
Relation Maximum Depth: 4

[ ] Discover relations between new terms within the text

Figure A.7: Background knowledge selection settings.

graph visualisation in the validation tab (see Figure A.8). After checking and confirming the relations, the user has to press proceed, and move to the next step.



[✓] Remove duplicate relations

[✓] Show previously discarded relations

[✓] Run pattern-based relevance

Pattern 1 weight (0-1):	1.0
Pattern 2 weight (0-1):	1.0
Pattern 3 weight (0-1):	1.0
Pattern 4 weight (0-1):	1.0
Pattern 5 weight (0-1):	1.0
Pattern 6 weight (0-1):	1.0
Relevance Threshold:	0.0

Context graph visualisation settings:
[✓] Display ontology
[✓] Display statement context
[✓] Display shared nodes up-to depth 1
[ ] Display shared nodes only
[ ] Display ontology's named relations

Figure A.8: Relevance assessment and graph visualisation settings.

A.2.5 Applying Changes and Creating a New Ontology Version

After approving the relations, the user will see the list of changes under the *Ontology Changes* tab, which is the last step or the process in Figure A.4. After approving the changes, we press in our case on the *Apply changes on a new version* button, to create a new detached ontology version that will directly appear in the ontology navigation (Figure A.9). This new version is ready to be used and modified further as needed within the toolkit. The other option is to apply the changes on the original ontology, depending on the scenario and user preference.

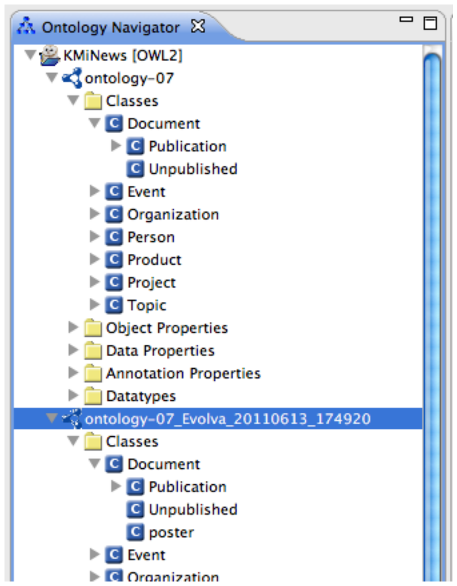


Figure A.9: New ontology version appearing in the navigator, and containing the new changes selected.

References

Alani, H. (2006), Position paper: ontology construction from online ontologies, *in* 'Proceedings of the 15th international conference on WWW', ACM New York, NY, USA, pp. 491–495.

Alani, H. and Brewster, C. (2005), Ontology ranking based on the analysis of concept structures, *in* 'Proceedings of the 3rd international conference on Knowledge capture (K-CAP)', ACM Press, Banff, Alberta, Canada, pp. 51–58.

Alani, H., Harris, S. and O'Neil, B. (2006), Winnowing ontologies based on application use, *in* 'Proceedings of 3rd European Semantic Web Conference (ESWC)', Springer, Budva, Montenegro.

Angeletou, S., Sabou, M. and Motta, E. (2008), Semantically enriching folksonomies with FLOR, *in* 'Proceedings of the 1st International Workshop on Collective Semantics: Collective Intelligence & the Semantic Web (CISWeb)', Tenerife, Spain.

Banerjee, S. and Pedersen, T. (2002), An adapted lesk algorithm for word sense disambiguation using WordNet, *in* 'Proceedings of the 3rd International Conference on Computational Linguistics and Intelligent Text Processing (CICLing)', Springer-Verlag, pp. 136–145.

Berners-Lee, T. (1999), *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor*, Harper San Francisco.

Berners-Lee, T., Hendler, J. and Lassila, O. (2001), 'The semantic web', *Scientific american* **284**(5), 28–37.

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R. and Hellmann, S. (2009), 'DBpedia-A crystallization point for the web of data', *Web Semantics: Science, Services and Agents on the World Wide Web* **7**(3), 154–165.

Bloehdorn, S., Haase, P., Sure, Y. and Voelker, J. (2006), Ontology evolution, *in* 'Semantic Web Technologies - Trends and Research in Ontology-based Systems', John Wiley & Sons, pp. 51–70.

Bruza, P. D. and Huibers, T. W. C. (1996), 'A study of aboutness in information retrieval', *Artificial Intelligence Review* **10**(5), 381–407.

Budinsky, F., Brodsky, S. A. and Merks, E. (2003), *Eclipse modeling framework*, Pearson Education.

Carroll, J. J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A. and Wilkinson, K. (2004), Jena: implementing the semantic web recommendations, *in* 'Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters', pp. 74–83.

Ceravolo, P., Corallo, A., Elia, G. and Zilli, A. (2004), Managing ontology evolution via relational constraints, *in* 'Proceedings of the 8th International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES)', Wellington, New Zealand.

Cimiano, P., Handschuh, S. and Staab, S. (2004), 'Towards the self-annotating web', *Proceedings of the 13th international conference on World Wide Web* pp. 462–471.

Cimiano, P. and Volker, J. (2005), Text2Onto a framework for ontology learning and data-driven change discovery, *in* 'Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB)', Springer Berlin / Heidelberg, Alicante, Spain, pp. 227–238.

Clark, P., Fellbaum, C. and Hobbs, J. (2008), Using and extending wordnet to support question-answering, *in* 'Proceedings of the 4th Global WordNet Conference (GWC)', Hungary, pp. 111–119.

Cohen, J. (1960), ‘A coefficient of agreement for nominal scales’, *Educational and psychological measurement* **20**(1), 37–46.

Cohen, P. R. (1995), *Empirical methods for artificial intelligence*, MIT press.

Cohen, W. W., Ravikumar, P. and Fienberg, S. E. (2003), A comparison of string distance metrics for name-matching tasks, *in* ‘Proceedings of the IJCAI Workshop on Information Integration on the Web (IIWeb)’.

Cunningham, D. H., Maynard, D. D., Bontcheva, D. K. and Tablan, M. V. (2002), GATE: a framework and graphical development environment for robust NLP tools and applications, *in* ‘Proceedings of the 40th Annual Meeting of the ACL’.

d’Aquin, M. (2009), Formally measuring agreement and disagreement in ontologies, *in* ‘Proceedings of the Fifth International Conference on Knowledge Capture (K-CAP)’, California, USA, pp. 145–152.

d’Aquin, M., Gomez-Perez, J. and Haase, P. (2008a), NeOn - lifecycle support for networked ontologies - case studies in the pharmaceutical industry, *in* ‘Proceedings of the 2nd European Semantic Technology Conference (ESTC)’, Vienna, Austria.

d’Aquin, M. and Lewen, H. (2009), Cupboard-a place to expose your ontologies to applications and the community, *in* ‘Proceedings of The 6th European Semantic Web Conference (Demo) ESWC’, Heraklion, Greece, pp. 913–918.

d'Aquin, M. and Motta, E. (2011), 'Watson, more than a semantic web search engine', *Semantic Web Journal* **1**(2), 55–63.

d'Aquin, M., Motta, E., Sabou, M., Angeletou, S., Gridinoc, L., Lopez, V. and Guidi, D. (2008b), 'Toward a new generation of semantic web applications', *IEEE Intelligent Systems* **23**(3), 20–28.

d'Aquin, M., Sabou, M., Motta, E., Angeletou, S., Gridinoc, L., Lopez, V. and Zablith, F. (2008c), What can be done with the semantic web? an overview of watson-based applications, *in* 'Proceedings of the 5th Workshop on Semantic Web Applications and Perspectives (SWAP)', Rome, Italy.

d'Aquin, M., Schlicht, A., Stuckenschmidt, H. and Sabou, M. (2009), Criteria and evaluation for ontology modularization techniques, *in* 'Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization', Springer-Verlag, pp. 67–89.

Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R. S., Peng, Y., Reddivari, P., Doshi, V. and Sachs, J. (2004), Swoogle: a search and metadata engine for the semantic web, *in* 'Proceedings of the 13th ACM international conference on Information and knowledge management (CIKM)', Washington D.C., USA, pp. 652–659.

Ding, L., Kolari, P., Ding, Z. and Avancha, S. (2007), Using ontologies in the semantic web: A survey, *in* 'Ontologies: A Handbook of Principles, Concepts and Applications in Information Systems', Vol. 14, Springer US, Boston, MA, pp. 79–113.

Ding, L., Pan, R., Finin, T., Joshi, A., Peng, Y. and Kolari, P. (2005), Finding and ranking knowledge on the semantic web, *in* 'Proceedings of the 4th International Semantic Web Conference (ISWC)', Galway, Ireland, pp. 156–170.

Fellbaum, C. (1998), *Wordnet: An Electronic Lexical Database*, MIT Press.

Flouris, G. (2006), On Belief Change and Ontology Evolution, PhD thesis, Department of Computer Science, University of Crete.

Flouris, G., Huang, Z., Pan, J. Z., Plexousakis, D. and Wache, H. (2006), Inconsistencies, negations and changes in ontologies, *in* 'Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)', Vol. 21, Boston, Massachusetts, pp. 1295–1300.

Gangemi, A., Catenacci, C., Ciaramita, M. and Lehmann, J. (2005), Ontology evaluation and validation: An integrated formal model for the quality diagnostic task, Technical report.

Gangemi, A., Catenacci, C., Ciaramita, M. and Lehmann, J. (2006), Modelling ontology evaluation and validation, *in* 'Proceedings of the 3rd European Semantic Web Conference (ESWC)', Springer, Budva, Montenegro, pp. 140–154.

Gracia, J., d'Aquin, M. and Mena, E. (2009), Large scale integration of senses for the semantic web, *in* 'Proceedings of the 18th international conference on World Wide Web (WWW)', ACM, Madrid, Spain, pp. 611–620.

Gruber, T. R. (1993), 'A translation approach to portable ontology specifications', *Knowledge acquisition* **5**(2), 199–220.

Guarino, N. and Welty, C. (2002), 'Evaluating ontological decisions with OntoClean', *Communications of the ACM* **45**(2), 65.

Haase, P., Harmelen, F., Huang, Z., Stuckenschmidt, H. and Sure, Y. (2005), A framework for handling inconsistency in changing ontologies, *in* 'Proceedings of the 4th International Semantic Web Conference (ISWC)', Galway, Ireland, pp. 353–367.

Haase, P. and Stojanovic, L. (2005), Consistent evolution of OWL ontologies, *in* 'Proceedings of the 2nd European Semantic Web Conference (ESWC)', Heraklion, Greece.

Haase, P. and Sure, Y. (2004), 'D3. 1.1. b state of the art on ontology evolution', *SEKT Deliverable* .

Hearst, M. A. (1992), Automatic acquisition of hyponyms from large text corpora, *in* 'Proceedings of the 14th Conference on Computational Linguistics', Vol. 2, pp. 539–545.

Huang, Z. and Stuckenschmidt, H. (2005), Reasoning with multi-version ontologies: A temporal logic approach, *in* 'Proceedings of the 4th International Semantic Web Conference (ISWC)', Galway, Ireland, pp. 398–412.

Huang, Z., Van Harmelen, F. and Teije, A. (2005), Reasoning with inconsistent ontologies, *in* 'Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI)', Vol. 19, Edinburgh, Scotland, pp. 454–459.

Ide, N. and Vronis, J. (1998), 'Introduction to the special issue on word sense disambiguation: the state of the art', *Computational Linguistics* **24**(1), 2–40.

Ji, Q., Haase, P., Qi, G., Hitzler, P. and Stadtmuller, S. (2009), RaDON-repair and diagnosis in ontology networks, *in* 'Proceedings of The 6th European Semantic Web Conference (ESWC)', Heraklion, Greece, pp. 863–867.

Kamada, T. and Kawai, S. (1989), 'An algorithm for drawing general undirected graphs', *Information Processing Letters* **31**(12), 7–15.

Klein, M. (2004), Change Management for Distributed Ontologies, PhD thesis, Vrije Universiteit in Amsterdam.

Klein, M. and Fensel, D. (2001), Ontology versioning on the semantic web, *in* 'Proceedings of the International Semantic Web Working Symposium (SWWS)', California, USA, pp. 75–91.

Klein, M., Fensel, D., Kiryakov, A. and Ognyanov, D. (2002), Ontology versioning and change detection on the web, *in* 'Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web - EKAW', pp. 247–259.

Klein, M., Kiryakov, A., Ognyanov, D. and Fensel, D. (2002), Finding and characterizing changes in ontologies, *in* 'Proceedings of the 21st International Conference on Conceptual Modeling', Tampere, Finland.

Klein, M. and Noy, N. F. (2003), A Component-Based framework for ontology evolution, *in* 'Proceedings of the Workshop on Ontologies and Distributed Systems', Acapulco, Mexico, pp. 135–144.

Kondylakis, H., Flouris, G. and Plexousakis, D. (2009), Ontology and schema evolution in data integration: Review and assessment, *in* 'On the Move to Meaningful Internet Systems (OTM)', Vilamoura, Algarve, Portugal, pp. 932–947.

Laera, L., Handschuh, S., Zemanek, J., Volkel, M., Bendaoud, R., Hacene, M. R., Toussaint, Y., Delecroix, B. and Napoli, A. (2008), D2.3.8 v2 report and prototype of dynamics in the ontology lifecycle, Technical report.

Lei, Y., Sabou, M., Lopez, V., Zhu, J., Uren, V. and Motta, E. (2006), An infrastructure for acquiring high quality semantic metadata, *in* 'Proceedings of the 3rd European Semantic Web Conference (ESWC)', Budva, Montenegro.

Li, X., Szpakowicz, S. and Matwin, S. (1995), A WordNet-based algorithm for word sense disambiguation, *in* 'Proceedings of the International Joint Conference on Artificial Intelligence', Vol. 17-24, Montreal, Canada, pp. 1368–1374.

Liang, Y., Alani, H. and Shadbolt, N. (2006), Enabling active ontology change management within semantic web-based applications. mini-thesis: PhD upgrade report, Technical report.

Lopez, V., Uren, V., Sabou, M. and Motta, E. (2009), Cross ontology query answering on the semantic web, *in* 'Proceedings of the fifth international conference on Knowledge capture - K-CAP', Redondo Beach, California, USA, p. 17.

Maedche, A., Motik, B., Stojanovic, L., Studer, R. and Volz, R. (2002), Managing multiple ontologies and ontology evolution in ontologging, *in* 'Proceedings of the Conference on Intelligent Information Processing, World Computer Congress', Kluwer, pp. 51–63.

Maynard, D., Funk, A. and Peters, W. (2009), SPRAT: a tool for automatic semantic pattern based ontology population, *in* 'Proceedings of the International Conference for Digital Libraries and the Semantic Web', Trento, Italy.

Maynard, D., Li, Y. and Peters, W. (2008), NLP techniques for term extraction and ontology population, *in* 'Proceedings of the Conference on Ontology Learning and Population: Bridging the Gap Between Text and Knowledge', IOS Press Amsterdam, pp. 107–127.

Maynard, D., Peters, W., d'Aquin, M. and Sabou, M. (2007), Change management for metadata evolution, *in* 'Proceedings of the International Workshop on Ontology Dynamics (IWOD)', Innsbruck, Austria.

McGuinness, D. L., Fikes, R., Rice, J. and Wilder, S. (2000), An environment for merging and testing large ontologies, *in* 'Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR)', Morgan Kaufmann, Colorado, USA, pp. 483–493.

Mizzaro, S. (1997), 'Relevance: the whole history', *Journal of the American Society for Information Science archive* **48**(9), 810–832.

Novacek, V., Laera, L. and Handschuh, S. (2007), Semi-automatic integration of learned ontologies into a collaborative framework, *in* 'Proceedings of the International Workshop on Ontology Dynamics (IWOD)', Innsbruck, Austria.

Noy, N. F., Chugh, A., Liu, W. and Musen, M. (2006), A framework for ontology evolution in collaborative environments, *in* 'Proceedings of the 6th International Semantic Web Conference (ISWC)', Springer, Athens, Georgia, pp. 544–558.

Noy, N. F., Kunnatur, S., Klein, M. and Musen, M. A. (2004), Tracking changes during ontology evolution, *in* 'Proceedings of the 3rd International Conference on the Semantic Web (ISWC)', Springer, Hiroshima, Japan, pp. 259–273.

Noy, N. F. and Musen, M. A. (2003), 'The PROMPT suite: interactive tools for ontology merging and mapping', *International Journal of Human-Computer Studies* **59**(6), 983–1024.

Noy, N. F. and Musen, M. A. (2004), Specifying ontology views by traversal, *in* 'Proceedings of the 3rd International Conference on the Semantic Web (ISWC)', Springer, Hiroshima, Japan, pp. 713–725.

Obst, D. and Chan, C. (2005), Towards a framework for ontology evolution, *in* 'Proceedings of the Canadian Conference on Electrical and Computer Engineering', pp. 2191–2194.

Oren, E., Delbru, R., Catasta, M., Cyganiak, R., Stenzhorn, H. and Tummarello, G. (2008), 'Sindice.com: A document-oriented lookup index for open linked data', *International Journal of Metadata, Semantics and Ontologies* **3**(1), 37–52.

Ottens, K., Gleizes, M. and Glize, P. (2007), A multi-agent system for building dynamic ontologies, *in* 'Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems', ACM, Honolulu, Hawaii.

Ottens, K., Hernandez, N., Gleizes, M. P. and Aussenac-Gilles, N. (2009), 'A Multi-Agent system for dynamic ontologies', *Journal of Logic and Computation Special Issue: Recent Advances in Ontology Dynamics* **19**(5).

Palma, R., Haase, P. and Qui, J. (2009), D1.3.2 change management to support collaborative workflows, Technical report, NeOn Project Deliverable.

Palmisano, I., Tamma, V., Iannone, L., Payne, T. and Doran, P. (2008), Dynamic change evaluation for ontology evolution in the semantic web, *in* 'Proceedings of the International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)', Vol. 1, pp. 34–40.

Pammer, V., Ghidini, C., Rospocher, M., Serafini, L. and Lindstaedt, S. (2010), Automatic support for formative ontology evaluation, *in* 'Poster at EKAW 2010', Lisbon, Portugal.

Pammer, V., Serafini, L. and Lindstaedt, M. (2009), Highlighting assertional effects of ontology editing activities in OWL, *in* 'Proceedings of the ISWC International Workshop on Ontology Dynamics (IWOD)', USA.

Pasca, M. and Harabagiu, S. (2001), The informative role of WordNet in open-domain question answering, *in* 'Proceedings of NAACL Workshop on WordNet and Other Lexical Resources', pp. 138–143.

Peroni, S., Motta, E. and D'Aquin, M. (2008), Identifying key concepts in an ontology, through the integration of cognitive principles with statistical and topological measures, *in* 'Proceedings of the 3rd Asian Semantic Web Conference on The Semantic Web', Springer-Verlag, Bangkok, Thailand, pp. 242–256.

Randolph, J. (2005), Free-Marginal multirater kappa: An alternative to fleiss Fixed-Marginal multirater kappa, *in* 'Joensuu Learning and Instruction Symposium'.

Randolph, J. (2008), 'Online kappa calculator, http://justus.randolph.name/kappa'.

Rogozan, D. and Paquette, G. (2005), Managing ontology changes on the semantic web, *in* 'Proceedings of the International Conference on Web Intelligence (WI)', Compiegne, France, pp. 430–433.

Sabou, M., d'Aquin, M. and Motta, E. (2008), 'Exploring the semantic web as background knowledge for ontology matching', *Journal on Data Semantics XI - Lecture Notes in Computer Science* **5383**, 156–190.

Sabou, M., Fernandez, M. and Motta, E. (2009), Evaluating semantic relations by exploring ontologies on the semantic web, *in* 'Proceedings of the 14th International Conference on Applications of Natural Language to Information Systems (NLDB)', Springer Verlag, Saarbrcken, Germany.

Schlobach, S. and Cornet, R. (2003), Non-standard reasoning services for the debugging of description logic terminologies, *in* 'Proceedings of the 18th International Joint Conference on Artificial Intelligence', Morgan Kaufmann, Acapulco, Mexico, pp. 355–362.

Shadbolt, N., Hall, W. and Berners-Lee, T. (2006), 'The semantic web revisited', *Intelligent Systems, IEEE* **21**(3), 96–101.

Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A. and Katz, Y. (2007), 'Pellet: A practical OWL-DL reasoner', *Web Semantics: Science, Services and Agents on the World Wide Web* **5**(2), 51–53.

Sperber, D. and Wilson, D. (1986), *Relevance: Communication and cognition*, MIT Press, Cambridge, MA.

Sternberg, R. J. (1990), *Metaphors of mind*, Cambridge University Press, New York.

Stojanovic, L. (2004), Methods and Tools for Ontology Evolution, PhD thesis, FZI - Research Center for Information Technologies at the University of Karslruhe.

Stojanovic, L., Maedche, A., Motik, B. and Stojanovic, N. (2002), User-driven ontology evolution management, *in* 'Proceedings of the 13th European Conference on Knowledge Engineering and Knowledge Management (EKAW)', Siguenza, Spain.

Supekar, K. (2005), A peer-review approach for ontology evaluation, *in* '8th International Protege Conference', Madrid, Spain, pp. 77–79.

Tamma, V. and Bench-Capon, T. (2001), A conceptual model to facilitate knowledge sharing in multi-agent systems, *in* 'Proceedings of the Autonomous Agents 2001 Workshop on Ontologies in Agent Systems (OAS)', Montreal, Canada, pp. 69–76.

Tartir, S., Arpinar, I. B., Moore, M., Sheth, A. P. and Aleman-Meza, B. (2005), OntoQA: metric-based ontology quality analysis, *in* 'IEEE Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources', Vol. 9.

Tirmizi, S., Sequeda, J. and Miranker, D. (2008), Translating sql applications to the semantic web, *in* 'Proceedings of the 19th International Databases and Expert Systems Application Conference (DEXA)', Turin, Italy, pp. 450–464.

Velardi, P., Fabriani, P. and Missikoff, M. (2001), Using text processing techniques to automatically enrich a domain ontology, *in* 'Proceedings of the 2nd International Conference on Formal Ontology in Information Systems', Ogunquit, ME, USA, pp. 270–284.

Volz, R., Oberle, D., Staab, S. and Motik, B. (2003), KAON SERVER-A semantic web management system, *in* 'Alternate Track Proceedings of the

Twelfth International World Wide Web Conference (WWW)', Budapest, Hungary, pp. 20–24.

Vrandecic, D., Pinto, H. S., Sure, Y. and Tempich, C. (2005), 'The DILIGENT knowledge processes', *Journal of Knowledge Management* **9**(5), 85–96.

Wang, Y., Liu, X. and Ye, R. (2008), Ontology evolution issues in adaptable information management systems, *in* 'Proceedings of the IEEE International Conference on e-Business Engineering', Vol. 0, IEEE Computer Society, Los Alamitos, CA, USA, pp. 753–758.

Wu, Z. and Palmer, M. (1994), Verb semantics and lexical selection, *in* 'Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics', Association for Computational Linguistics, Las Cruces, New Mexico, pp. 133–138.

Zablith, F., d'Aquin, M., Sabou, M. and Motta, E. (2010), Using ontological contexts to assess the relevance of statements in ontology evolution, *in* 'Knowledge Engineering and Knowledge Management by the Masses (EKAW)', Springer-Verlag, Lisbon, Portugal.

Zablith, F., Fernandez, M. and Rowe, M. (2011), The OU linked open data: production and consumption, *in* 'Proceedings of the 1st International Workshop on eLearning Approaches for the Linked Data Age (Linked Learning) at ESWC', Heraklion, Greece.

Zhu, J., Uren, V. and Motta, E. (2005), ESpotter: adaptive named entity recognition for web browsing, *in* ‘Proceedings of the Workshop on IT Tools for Knowledge Management Systems at WM Conference’, Kaiserslautern, Germany, pp. 11–13.