KNOWLEDGE MEDIA
KMi
INSTITUTE

Evolva: Towards Automatic Ontology Evolution

Technical Report kmi-08-04
October 2008

Fouad Zablith

The Open University

Abstract

Ontologies form the core of Semantic Web systems, and as such, they need to evolve to meet the changing needs of the system and its users. Information is exponentially increasing in organizations' intranets as well as on the web, especially with the increased popularity of tools facilitating content generation such as wikis, blogs and social software. In such dynamic environments, evolving ontologies should be agile, i.e. with the least knowledge experts' input, for reflecting fast changes occurring in repositories, and keeping Semantic Web systems up-to-date. Most of current ontology evolution frameworks mainly rely on user input throughout their evolution process.

We propose Evolva, an ontology evolution framework, aiming to substantially reduce or even eliminate user input through exploiting various background knowledge sources. Background knowledge exists in various forms including lexical databases, web pages and Semantic Web ontologies. Evolva has five main components: information discovery, data validation, ontological changes, evolution validation and evolution management. We present in this report an overview of the current work on ontology evolution, followed by our ontology evolution approach and pilot study conducted so far, and we finally conclude with a discussion and our future directions.

Contents

1 Introduction **6**
1.1 Motivation 6
1.2 Outline 8

2 Literature Review **9**
2.1 Ontology Evolution Requirements 10
2.2 Sources of Ontology Changes and Evolution 11
2.2.1 Evolution as the Management of Changes 11
2.2.2 Evolution from External Data Sources 14
2.3 Conclusion and Major Gaps 16

3 Research Proposal **18**
3.1 Research Question 18
3.2 Proposed Approach 19
3.2.1 Ontology Evolution Framework: Evolva 19
3.2.2 Background Knowledge Supporting Evolva 22

4 Pilot Study **24**
4.1 Scenario Description 24
4.2 Information Extraction 25
4.3 Concepts Extracted Vs. Base Ontology Concepts 26
4.4 Relation Discovery Experiment 27
4.4.1 Experimental Data 29
4.4.2 Evaluation of the WordNet based Relation Discovery 29
4.4.3 Evaluation Results for Scarlet 30
4.4.4 Error Analysis 31
4.5 Observations on Integrating Relations into the Base Ontology 33
4.6 Experiment Conclusion 34
4.7 System Implementation 35
4.7.1 String Matching 35
4.7.2 Text2Onto Integration 36

4.7.3 WordNet Relationship Path 36
4.7.4 Scarlet Relation Discovery Integration 37

5 Discussion and Future Work **40**

References **42**

List of Figures

1.1 Data Integration in Semantic Web Ontology 7

3.1 Evolva for Evolving Ontologies from External Data Sources 20
3.2 Evolva Framework Main Components 20
3.3 Evolva Framework: Detailed View 21
3.4 Finding Relations Between New Terms and The Base Ontology in Evolva . 23

4.1 Extracted Concepts Compared to the Base Ontology 27
4.2 The Wu and Palmer Concept Similarity Measure 28
4.3 Pilot System Current Status Covering Shaded Nodes 35
4.4 Screenshot: String Matching Results Example 36
4.5 Screenshot: Reading Corpus Directory 36
4.6 Screenshot: Activating WordNet Matching 37
4.7 Relation Discovery with Scarlet 38

List of Tables

2.1 Existing Work Vs. Requirements Coverage 17

4.1 Text2Onto Evaluation on 20 KMi News Articles 25
4.2 Concepts Relevance Based on SubClassOf Relation with Exact Matchings . 26
4.3 Examples of Relations Derived by Using WordNet 29
4.4 Evaluation Results for the Relations Derived from WordNet. 30
4.5 Examples of Relations Discovered Using Scarlet 30
4.6 Evaluation Results for the Relations Derived with Scarlet 31
4.7 Correlation Between the Length of the Path and the Correctness of a Relation 32
4.8 Bad Examples of WordNet Relationship Paths 33

Part 1

Introduction

Ontology evolution is defined in [10] as the "timely adaptation of an ontology to the arisen changes and the consistent management of these changes". Ontologies form the core of Semantic Web systems, and as such, they need to evolve to meet the changing needs of the system and its users (e.g. new data or functionalities introduced). This report goes through the analysis of current approaches on ontology evolution, and presents our research proposal, coupled with our experimental and implementation work under process. Parts of this report will be published in [40] and [41].

1.1 Motivation

It is well recognized today that the use of ontologies could improve data reuse and accessibility. Thus the integration of the content from various data sources into an ontology, as displayed in Figure 1.1, is the target of many organizations nowadays. A motivating use case is the KMi Semantic Web portal[1] based on the extended AKT reference ontology[2]. The content of this ontology, both in terms of structure and instances, needs to be kept up-to-date by painstaking manual efforts. A mix of structural and instance changes in the ontology could be triggered from the regular publication of KMi news articles that lead to the continuous introduction of new concepts and instances. Another source of ontology change could be from new projects or technologies arising in KMi databases, leading to instance changes. Those changes will surely be coupled with structural changes through, for example, allocating existing/new resources (e.g. KMi people, external organizations) to the technology or project introduced, requiring the creation of corresponding links and relations. These are just a few examples showing the extensive work needed at the ontology level, to reflect the KMi domain everyday changes.

Evolving an ontology is a time consuming task, and relies on considerable input from

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

[2]http://kmi.open.ac.uk/projects/akt/ref-onto/index.html

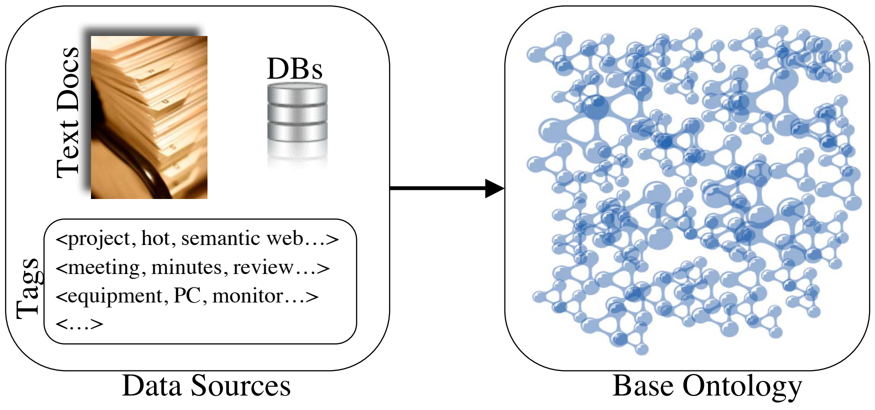

Text Docs
DBs
Tags
<project, hot, semantic web…>
<meeting, minutes, review…>
<equipment, PC, monitor…>
<…>
Data Sources
Base Ontology

Figure 1.1: Data Integration in Semantic Web Ontology

a user with knowledge representation skills. Various works target the issue of updating or evolving ontologies from external data sources [22, 26]. However, user input is usually required during evolution, especially at the level of performing changes. This is often due to the limited information residing in sources from which the evolution of ontology is performed, as well as the additional reasoning required to resolve the right links between entities, and complete information. Our research is motivated by the hypothesis that:

> (Online) knowledge and various data sources could be explored to significantly decrease or even eliminate user input in the evolution process, thus rendering it dynamic.

We understand background knowledge as a range of data sources with various levels of formality including web pages formed of unstructured text such as Wikipedia, lexical databases such as WordNet [9] and FrameNet [2], and online ontologies.

Background knowledge can support finding relationships between new knowledge discovered, and existing knowledge in the domain ontology, which we call "base ontology". For example, the lexical database WordNet identifies "European Union", a new instance discovered in the KMi news articles, as a kind of "Organization", a concept defined in the base ontology.

Background knowledge helps as well in completing information not available to the scope of the system. Consider the case of adding to the base ontology a publication co-authored by Enrico Motta and Peter Haase. KMi's data sources hold significant information about Enrico Motta, who works at KMi. However, Peter Haase who is not part of KMi, requires additional effort to identify him as a person with additional details such as current position, address, other publications, etc. These details are, however, available in online Semantic Web data.

1.2 Outline

In the next part of this report we present our literature review in which we talk about the requirements of ontology evolution, sources of ontology changes, evolution as the management of changes as well as from external data sources. In Part 3 we talk about our research problem and proposed approach, followed by our pilot study in Part 4 that includes as well an experiment on relation discovery. Then (in Part 5) we conclude the report with our research plan and discussion.

Part 2

Literature Review

We present in this part related work in our research field, and identify existing gaps in current approaches. Ontology evolution is not only about performing changes, but it handles as well the management of changes, while maintaining the ontology's consistency. Many work targeted ontology changes from different perspectives through ontology population, learning or versioning, and sometimes used the term evolution in their methods, even if their work covers only a portion of the processes that happen during evolution.

Ontology population deals with adding instances to an ontology starting from structured or semi-structured external data sources, without dealing with structural changes of the corresponding ontology. Ontology population could be performed by creating a set of mappings between the data source and the ontology [18]. The possibility of having inconsistencies as a result of ontology population is low, as the source and target of data population are well defined. Hence dealing with inconsistencies at the level of ontology population is not taken into consideration.

Ontology learning is the process of extracting information from data sources, and transforming them into ontology concepts, instances and relations. Examples of ontology learning tools are Text2Onto [6] and OntoLearn [35], which learn ontologies from text. Ontology learning tools do not deal with extensive management of ontology changes. Extracted terms are usually supplied with a domain relevance value that could be based on the Term Frequency Inverse Document Frequency (TFIDF) in the case of Text2Onto, or a domain relevance formula specific to OntoLearn.

Ontology versioning handles the creation and management of ontology versions, taking into account the consistency and compatibility between different versions. Various works involve ontology versioning, such as the work on ontologies in dynamic environments based on the SHOE [19] knowledge representation language [13].

We understand *ontology evolution* as the structural and instances adaptation of the ontology to changes occurring in the domain, coupled with the management of these changes. We consider ontology evolution as a process starting from data sources that contain valuable information about the represented domain, as well as aiming to keep the ontology operational and consistent. Our target is to make the evolution process occur with the least user input possible.

2.1 Ontology Evolution Requirements

Various work analyzed ontology evolution, and identified a series of requirements for making the evolution more efficient. First, an ontology should remain consistent throughout the evolution process [32]. Second, the option to be a supervised evolution [33] should be available, as users sometimes prefer to have control over changes happening. A third requirement is to be semi-automated and to notify the users when an evolution should occur.

Applying changes to an ontology can sometimes be very hard to maintain, especially in cases where changes could flag a chain of changes across the system. This is where change propagation [16, 30] is required to update all the ontology dependent components. Tools have been developed to assist users in spotting all the required changes to apply in order to avoid inconsistency [33]. Some works suggest resolving inconsistencies by applying different ontology revisions [12]. But this would lead to problems in merging information from the different ontology versions. The latter issue is tackled by the use of backward-compatibility that is implemented in SHOE [19], a "prototype ontology language", to detect whether the models represented in two different ontology versions are compatible [12]. It is introduced to detect the consistency in the meanings of two versions of an ontology, making it safe as well to merge information between both versions.

Through our scenario analysis and literature review, we identify the following additional requirements for ontology evolution: (1) We believe that in most of the cases, domains are represented by a mix of different data sources. With today's diversity of tools and data repositories, ontology evolution should handle knowledge discovery from different data sources ranging from unstructured data such as text or tags, to structured data such as other ontologies and databases. (2) In order to make the evolution as agile as possible, performing ontology changes should be performed with the least or even without user input. (3) As ontology evolution is an ongoing process, time related knowledge should be handled correctly for avoiding conflicts in data. (4) With the presence of multiple-data sources, it's likely to have duplication in data during evolution, raising the need for automated or semi-automated duplication check.

Ontology evolution requirements can be summarized in the following compiled list:

R1. The ontology should remain consistent during evolution, through resolving for example time related and duplication inconsistencies.

R2. Relevant and validated information should be extracted from different types of data sources: structured, semi-structured and unstructured.

R3. For a fast adaptation to changes, ontology evolution should occur with the least or even without user input i.e. dynamic.

R4. Changes should be recorded and propagated to the ontology dependent components.

R5. Users should have a degree of control over the evolution.

R6. Ontology evolution should be as domain independent as possible.

2.2 Sources of Ontology Changes and Evolution

We realize that handling ontology changes is viewed from two different perspectives. The first set of approaches looked at the evolution of ontologies as the management of changes performed by knowledge engineers. In this case ontologies are considered closed entities, without exploring external data sources to identify the need for changes. A second set of approaches made use of information residing in external data sources to perform automated ontology changes. We discuss these two lines of work in the next subsections.

2.2.1 Evolution as the Management of Changes

Various work has been done on the analysis of ontology changes and evolution, and viewed the ontology as a closed entity, undergoing changes from the ontology designer. They mainly dealt with the management of changes for supporting knowledge engineers. In this section we present various approaches that looked at ontology evolution from this perspective.

Ontologies should not be treated as text documents while tracing their evolution, but structural and semantic changes are the important components to track. There are many tools that track changes in an ontology such as PromptDiff [25].

PromptDiff is an 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. This tool can be downloaded from the Protégé plugins website. However, it seems that PromptDiff captures only the structural changes without handling instance changes [25].

Other approaches propose having an evolution log where all changes happening to an ontology are recorded [30]. This could make tracing and rolling back changes easier. Ontologging [20] is another example of evolution tracer tools, which analyses as well the effects of changes performed on the ontology. Databases have been as well used in managing ontology evolutions, by storing in tables the changes done at the ontology level or the ontology's metadata level [4].

In Text2Onto [6], a pointer is used to keep track of the data changes, presenting an ontology evolution traceability feature to the end-user. In this model it is even possible to keep track of various inconsistent scenarios, and giving the users the ability to assess "the" consistent ontology.

OntoAnalyzer is a tool to trace complex ontological changes. The authors of the paper [27] state that OntoAnalyzer has an advantage over the KAON framework [37], which is able to handle only elementary changes. Comparing it to PromptDiff [25] and OntoView [15], 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 his thesis [14], Klein targets the management side of distributed ontologies. The author uses existing approaches in other fields such as databases, and applies them to Semantic Web ontologies. The way change is applied to ontologies in Kleins thesis is through an ontology of change, where possible changes to ontologies are instances of the ontology of change. They are represented in RDF data. The thesis mainly deals with evolution management, and how it could assist ontology engineers maintaining the ontology. Klein makes use of ontology versioning for managing changes and spotting inconsistencies. 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 is 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.

Stojanovic [31], identifies two types of ontology ontology evolution: top-down and bottom-up evolution. The first occurs when changes are done in the requirements of users such as adding new functionalities to the system. A bottom-up approach is when changes come from within the ontology such as a new inferred relation coming from relation analysis performed on the ontology intself. Additionally, Stojanovic [30] identifies three different sources of ontology change discovery:

– *Usage-driven* change discovery derived from the end-user behaviour representing the likes/dislikes.

– *Data-driven* change discovery derived from the analysis of existing ontologys instances by using for example data-mining techniques.

– *Structure-driven* change discovery based on the ontology structure analysis.

Stojanovic [30] proposed a framework for evolving ontologies mainly triggered by internal sources of change from within the ontology. The framework is a six phases cyclic process starting with the *change capturing* phase where changes to be applied to the ontology are identified. Then there is the *representation phase* where the changes are represented in the ontology. The third phase is the *semantics of change* phase including syntactic and semantic inconsistencies that could arise as a result of ontological changes [34]. 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 [31]. 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 having networked ontologies. The final phase is the *validation phase* to insure that the changes propagated are valid, and allows the user to undo the changes performed on the ontology. Stojanovic implements the six phases framework without capturing changes from external data sources.

Stojanovic [31] defines various evolution strategies. Building a strategy consists of a set of specific evolution cases such as adding a class, with the set of effects, and the action to take once triggered. Changes in ontologies have been categorized in a tabular representation throughout various papers [24, 31]. This representation could help in specifying the needed evolution "actions" in the ontology. For example there are cases to define when to delete the instances of a deleted class C, or when to keep its instances: should instances of type C not be deleted when C got a superclass [24]? Possible advanced evolution strategies [31] are (1) *structure-driven* which takes into account structure preferences (such as a deep or shallow hierarchy), (2) *process-driven* by setting specific constraints such as time or maximum number of instances, (3) *instance-driven* in the cases where instances are created heavily and (4) *frequency-driven* strategy that takes into consideration the most or last-used strategies.

Similar to Klein's ontology of change [14], Stojavonic proposes an evolution ontology, which is a meta-ontology for representing different changes transactions on an ontology. It is created in order to unify the representation of changes across the evolution environment. The core concept in the evolution ontology is the "Change" concept with the following sub-concepts:

$$AddConcept \sqsubseteq AdditiveChanges \sqsubseteq ElementaryChanges \sqsubseteq Change$$

The evolution ontology represents dependencies using the "causesChange" property, which enables as well to reverse effects of the applied change. Other properties as well help going to a previous state of the ontology [30].

Formalizing ontology changes is highly important for our research, especially that we are working on performing the changes dynamically. Such ontology of changes could be reused in our framework.

Noy et al. [23] describe a framework for ontology evolution in collaborative environments. The framework is scenario based and consists of various Protégé[1] plugins. Siorpaes [29] describes as well the need of a community-driven lightweight ontology evolution, hav-

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

ing a Wiki based technique for collaboratively building ontologies. However, even though Google, Wikipedia or WordNet are proposed as sources for background enrichment, the source of ontology evolution is limited to the collaborative input of the users, without dealing with ontology learning from external sources.

DILIGENT [38] is a decentralized user-centric framework proposing an ontology engineering methodology 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 *analyze 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 evolving 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.

The management of ontology changes is crucial in ontology evolution. We will reuse various approaches in the field and investigate their added-values to our work at a later stage our research.

2.2.2 Evolution from External Data Sources

Various approaches focus on ontological changes from external data sources such as text documents [3, 6, 22, 26, 35], metadata [21] and databases [11]. Most of the work involved in ontology changes from external sources are categorized as ontology learning tools such as Text2Onto [6] and OntoLearn [35]. These tools are not complete evolution systems as the management of changes is not handled to the extend to present high quality ontologies. Additionally, users do not have a well developed degree of control during the evolution.

Text2Onto [6] is designed to overcome the limitations of ontology learning tools from text, such as: 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 new Probabilistic Ontology Model, coupled with data-driven change discovery that enables specific changes detection without processing all the documents again. In this case data-driven changes are changes occurring in text documents. This tool assigns a degree of certainty about the learning process making it easier for the user to interact. 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.

Bloehdorn et al. [3] based their work on the six phases ontology evolution process proposed by Stojanovic [30]. However, 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 [11]. The authors identify that valuable information reside in databases and documents, but require better structuring and easy accessibility through the use of ontologies. They propose an architecture applied in a digital library domain or other electronic repositories, and they make use of ontology learning algorithm for extracting document contents. In addition to the data-driven changes, they integrate a usage-driven source of ontology changes, dealing with the analysis of user behavior and contextual search history for refining the ontology.

However the process is semi-automatic, and returns back suggestions to the user about potential ontology changes coming from ontology learning algorithms.

DINO [17, 22] is a framework for integrating ontologies. Part of its processes include a semi automatic integration of learned ontologies with a master ontology built by ontology designers. It includes 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 customized through a user interface, and the confidence values of extracted terms are fed to an ontology alignment/negotiator wrapper [22].

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 an alignment wrapper. The agreement of the semantics used is reached through negotiation using a negotiation wrapper. An axiom ontology, which contains the merging statements between the learned and the master ontology is created.

The statements in the axiom ontology are used as input for a reasoning and management wrapper, responsible for merging the learned ontology with the master ontology. Inconsistencies are checked at this level using the Jena OWL DL reasoner, giving the priority to the master ontology statements. Solving inconsistencies is elaborated more in the paper. In the case of having many possible candidates to be removed, the confidence returned by Text2Onto is relied on [22].

The resulting ontology is passed to the ontology diff wrapper that spots the changes brought to the master ontology using the SemVersion [36] library. The changes are sorted according to a relevance measure, leaving the choice of only presenting highly relevant changes to the users. The authors propose a mapping between the changing triples to natural language, in order to represent changes to users who are not familiar with technical OWL representations [22].

However, relying on the Text2Onto confidence for deciding which candidate to remove has a downside. This is because the confidence is based on the TFIDF measure, which does not perform well in cases when the number of documents in the corpus is low. Another limitation of the DINO approach is that the final knowledge integration decision is not automated, but left to the user.

Dynamo [26] is another tool that falls in the exploration of 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 is a semi-automated tool that requires in its process, next to text sources, an ontology designer. It uses adaptive multi-agent system architecture, and proposes 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. In addition to be domain specific, another limitation of Dynamo is that it generates simple concept hierarchy and not extended semantic hierarchies. Another drawback of the system is that the output depends on the order of the input data fed in the system. Handling taxonomy maintenance is also not well developed, as the system addresses only the low level structure of concept hierarchy and not the middle layer.

2.3 Conclusion and Major Gaps

We realize that automating knowledge modeling in the form of ontologies is approached from different perspectives: Ontology population and learning that deal with discovering knowledge from external data sources. Ontology versioning dealing with the management of different ontology versions, and ontology evolution through which some of the approaches considered it as a management of changes, while others included external data sources in the evolution process.

The literature review shows that there's no complete ontology evolution framework that tries to meet all the requirements mentioned earlier in section 2.1. Table 2.1 lists the previously mentioned work and how they meet the ontology evolution requirements. The last row of the table shows Evolva, our ontology evolution framework, with the aim to cover all the requirements in the list. We talk about Evolva in more details in the research proposal Part 3.

Our overview about ontology evolution and the tools available in the domain helped identifying two major gaps in our research field:

1. All described systems mainly rely on user input during the ontology evolution process.
2. Moreover, no system considers the use of background knowledge to support ontology evolution.

	Consistency Check	*External Data Sources*	*Change Propagation*	*User Control*	*Evolution Tracing*	*Automatic Evolution*
Stojanovik	[30, 10]		[30]	[30]	[30, 10]	
Klein	[14]		[14]	[14]	[14, 25]	
Noy				[23]	[23, 25]	
Haase	[10]	[11]	[10]			
DILIGENT			[38]	[38]		
DINO	[22, 17]	[22, 17]				
Dynamo		[26]				
Text2Onto	[6]	[6]		[6]		
OntoLearn		[35]				
OntoAnalyzer	[27]				[27]	
PromptDiff					[25]	
Ontologging	[20]				[20]	
Evolva's Target	✓	✓	✓	✓	✓	✓

Table 2.1: Existing Work Vs. Requirements Coverage

Part 3

Research Proposal

In this part we present our research proposal. We start with our research question and break it into four sub-questions. Then we discuss our proposed approach that includes Evolva, our ontology evolution framework, supported by a gradual background knowledge harvesting technique.

3.1 Research Question

Our research question is

> How to perform ontology evolution starting from heterogeneous data sources, and support the evolution by various background knowledge for substantially reducing, or even eliminating user input?

Our research question can be divided into four more focussed sub-questions:

Q.1. How to extract new and relevant information? A first step in an ontology evolution process is the identification of new and relevant information that should be added to the base ontology. Such information can reside in text corpora, databases and ontologies, making them a good source of discovering potential changes. These data sources should be processed for identifying and extracting concepts, relations and entities, relevant to the base ontology.

Q.2. How to perform ontological changes dynamically? Extracted information from data sources should be added to the base ontology without relying on user input, with the corresponding relations to existing knowledge in the base ontology. Relations should be discovered automatically by exploring background knowledge sources, and by taking into account the related entities' contextual meaning. For example, "degree" could refer to an academic degree in the education context, or to a unit-of-measure in the physics context with different types of relations.

Q.3. How to validate the evolved ontology? During evolution, inconsistencies could occur due to conflicts in statements, data duplication and temporal related facts. These phenomena are particularly likely to arise in cases when evolution is informed by knowledge extracted from multiple heterogeneous data sources, with various degrees of quality. A temporal inconsistency example is when some of the current KMi news articles mention Peter Scott as KMi's director, conflicting with news published during the period when Enrico Motta was the director of KMi. Such inconsistencies should be identified and resolved.

Q.4. How to manage the evolution? After validating the evolved ontology, its dependent components, such as other ontologies or applications, should be notified with the performed changes for ensuring compatibility. Another requirement is to be able to follow-up the evolution process, and present to the user a degree of control for monitoring and spotting unresolved problems.

3.2 Proposed Approach

We propose Evolva, an ontology evolution framework that explores various background knowledge sources to evolve ontologies and reduce user input.

3.2.1 Ontology Evolution Framework: Evolva

Evolva will play the role of mediator between the traditional data repositories, and the base ontology as displayed in Figure 3.1. Our framework is formed of five components visualized in Figure 3.2. We present a detailed view of our framework in Figure 3.3. In the rest of this part, we discuss the components of our ontology evolution framework.

Information discovery. One way to detect new knowledge to be added to the base ontology is by contrasting it to information contained in external domain and application specific sources, such as text corpora, databases or other ontologies. Unstructured data such as text documents or tags, require information extraction or ontology learning tools such as Text2Onto [6]. Text2Onto has many features including the extraction of concepts, instances and relations from text. 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. In the case of databases, a transformation should be performed to encapsulate the database schema and entities in an ontology compatible language.

Data validation. The discovered information are validated by applying a set of heuristic rules. For example, most of the two-letter concepts extracted by Text2Onto from KMi's news corpora such as "cu" and "th" are meaningless and should be discarded.

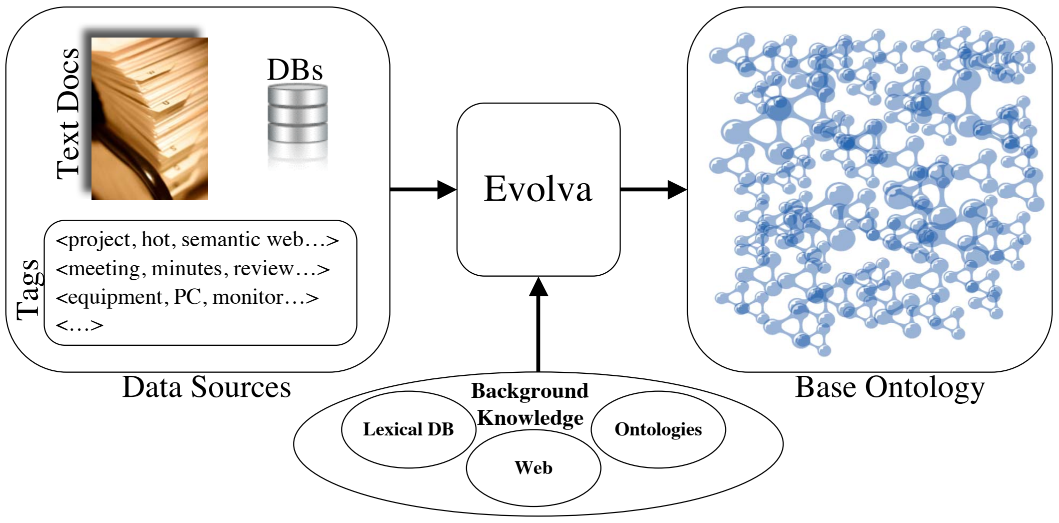


Figure 3.1: Evolva for Evolving Ontologies from External Data Sources

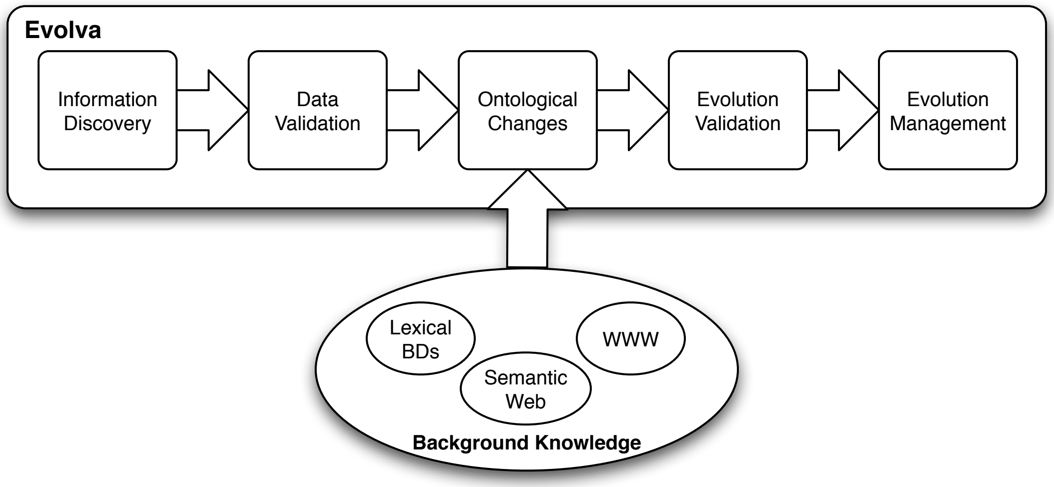


Figure 3.2: Evolva Framework Main Components

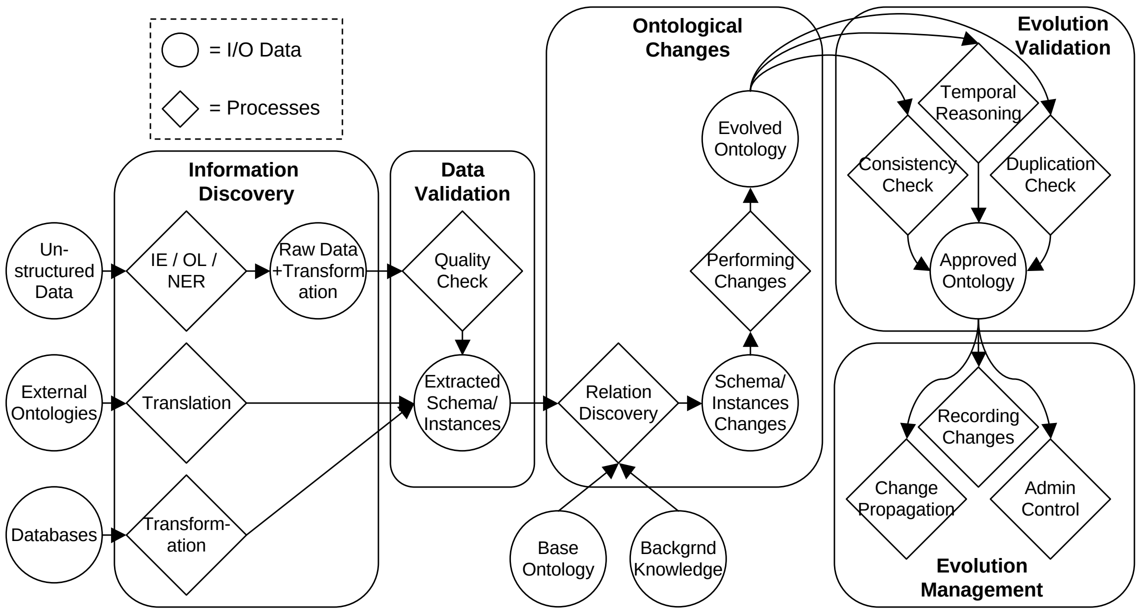


Figure 3.3: Evolva Framework: Detailed View

Ontologies and databases do not need this kind of low level quality check as the content structure is more trusted, and the type of information is explicitly defined.

Ontological changes. This component deals with performing changes to the ontology. Validated information are linked to the base ontology through a relation discovery mechanism, which relies of a set of background knowledge sources. After resolving the relations, corresponding changes are performed on the base ontology using a predefined set of encapsulated ontology changes as described by Klein [14] and Stojanovic [30].

Evolution validation. As mentioned earlier, performing ontological changes could generate some problems such as conflicting statements, data duplication and time related inconsistencies. We deal with these problems at the level of the evolution validation component, formed of the consistency and duplication checks, as well as the temporal reasoning process.

Evolution management. The approved ontology is passed to the evolution management component. In this component, the changes performed on the ontology are recorded to ensure functionalities such as tracing or rolling back changes. The changes are then propagated to dependent ontologies and applications. The administrator control is supplied for monitoring purposes, setting the evolution parameters and resolving any additional problem.

3.2.2 Background Knowledge Supporting Evolva

A core task in most ontology evolution scenarios is the integration of new knowledge into the base ontology. We focus on those scenarios in which such new knowledge is extracted as a set of emerging terms from textual corpora, databases, or domain ontologies. Traditionally this process of integrating a new set of emerging terms is performed by the ontology curator. For a given term, he/she would rely on his/her own knowledge of the domain to identify, in the base ontology, elements related to the term, as well as the actual relations they share. As such, it is a time consuming process, which requires the ontology curator to know well the ontology, as well as being an expert in the domain it covers.

Evolva makes use of various background knowledge sources to identify relations between new terms and ontology elements. The hypothesis is that a large part of the process of updating an ontology with new terms can be automated by using these sources as an alternative to the curator's domain knowledge. We have identified several potential sources of background knowledge. For example, thesauri such as WordNet have been long used as a reference resources for establishing relations between two given concepts, based on the relation that exists between their synsets. Because WordNet's dictionary can be downloaded and accessed locally by the system and because a variety of relation discovery techniques have been proposed and optimized, exploring this resource is quite fast. Online ontologies constitute another source of background knowledge which has been recently explored to support various tasks such as ontology matching [28] or enrichment [1]. While the initial results in employing these ontologies are encouraging, these techniques are still novel and in need of further optimizations (in particular regarding time-performance). Finally, the Web itself has been recognized as a vast source of information that can be exploited for relation discovery through the use of so-called lexico-syntactic patterns [5]. Because they rely on unstructured, textual sources, these techniques are more likely to introduce noise than the previously mentioned techniques which rely on already formalized knowledge. Additionally, these techniques are also time consuming given that they operate at Web scale.

Taking into account these considerations, we devised a relation discovery process that combines various background knowledge sources with the goal of optimizing time-performance and precision. As shown in Figure 3.4, the relation discovery starts from quick methods that are likely to return good results, and continues with slower methods which are likely to introduce a higher percentage of noise: (1) The process begins with string matching for detecting already existing terms in the ontology. This will identify equivalence relations between the new terms and the ontology elements. (2) Extracted elements that do not exist in the base ontology are passed to a module that performs relation discovery by exploring WordNet's synset hierarchy. (3) Terms that could not be incorporated by using WordNet are passed to the next module which explores Semantic Web ontologies. (4) If no relation is found, we resort to the slower and more noisy methods which explore the Web itself through search engines' APIs and lexical syntactic patterns [5]. In case no relation is found at the final level, the extracted term is discarded or, optionally, forwarded for manual check.

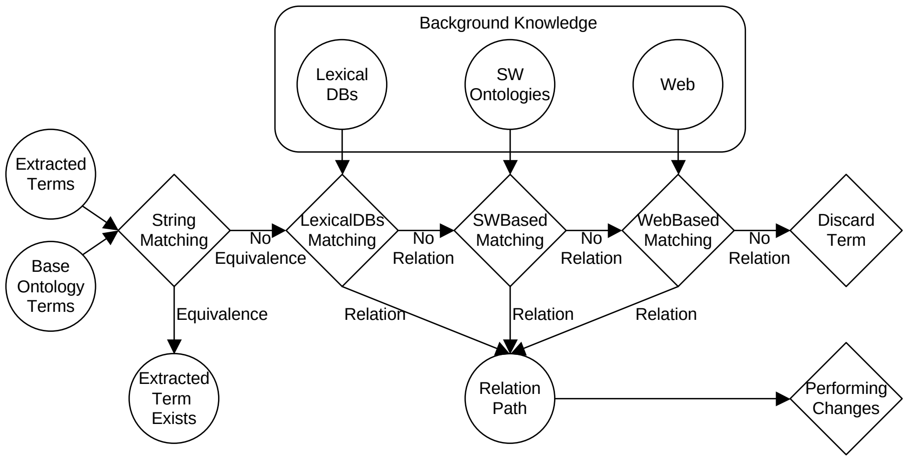


Figure 3.4: Finding Relations Between New Terms and The Base Ontology in Evolva

Part 4

Pilot Study

We are applying our initial methods for information discovery and data validation on the KMi portal data to evolve the underlying AKT ontology. We use Text2Onto [6] to extract information from the KMi Planet News web-pages[1]. We then perform string matching between the extracted concepts and the current KMi ontology, based on the Jaro distance metric [7] algorithm. We are currently testing the potential use of WordNet [9] and online ontologies to find possible relationships between the newly discovered terms and the base ontology's terms.

Next to experiments performed on information extraction, data validation and relation discovery, we are in parallel implementing our framework's components that we discuss later in this part.

4.1 Scenario Description

The Knowledge Media Institute's (KMi) Semantic Web portal is a typical example of an ontology based information system. The portal provides access to various data sources (e.g. staff and publication databases, news stories) by relying on an ontology that represents the academic domain, namely the AKT ontology. The ontology has been originally built manually and is automatically populated by relying on a set of manually established mapping rules [18].

However, apart from the population process, the evolution of this ontology was performed entirely manually. Indeed, as in this scenario ontology population is bound by strict and limited mapping expressions, when a new type of term (i.e. not covered by the mapping rules) is extracted, the intervention of the ontology administrator is required to modify the mappings. Moreover, the ontology schema can only be updated by the administrator. Finally, with no mechanism to support recording and managing changes, it is difficult to maintain a proper versioning of the ontology. Therefore, as this manual evolution of the

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

	Extracted Concepts	**Relevant Concepts**	**Average Confidence**
Exact Matching	26	26	0.0353
Close Matching	37	37	0.0253
New Concepts	474	97	0.0218
Overall	537	160	0.0227

Table 4.1: Text2Onto Evaluation on 20 KMi News Articles

ontology could not follow the changes in the underlying data (which happen on a daily basis), the ontology was finally left outdated.

4.2 Information Extraction

First we discuss a *manual* evaluation of Text2Onto performing information extraction on a small set of KMi's news articles. This experiment was performed manually to get an insight information about the capabilities of this tool. We select a 20 random news articles, and run Text2Onto to extract concepts, instances and relations. In section 4.3 we show how the data validation component takes care of evaluating the information extracted from all the corpus using an automated process.

The first evaluation of Text2Onto is about pointing out extracted concepts that are exactly matching concepts in the KMi ontology, closely matching or newly discovered concepts not available in the KMi ontology. Then we check if the newly extracted concepts are relevant to KMi's domain, and whether there's a correlation between the degree of matching, relevance and the confidence value generated by Text2Onto. The results are depicted in Table 4.1.

The confidence of extraction assigned by Text2Onto is taken into consideration to check if there is a correlation between the confidence, and the matching done with the KMi ontology. The confidence of concepts is the normalized value of the TFIDF between [0..1], reflecting the probability of how the concept extraction is confident with respect to the domain.

Based on the numbers in the table, the precision of Text2Onto is 29.8%. The average confidences are relatively close to each other, probably due to the small number of documents in the corpus, making the relative TFIDF measures not very accurate relatively to the KMi domain.

It is worth to note that the relevance of the concepts is directly related, but not limited

SubClassOf Relation	**Number of Relations**	**Relevant**
ExactMatch↔CloseMatch	7	1
ExactMatch↔NewConcepts	21	6

Table 4.2: Concepts Relevance Based on SubClassOf Relation with Exact Matchings

to the domain of the ontology, application layer features i.e. the usage, tasks fulfilled and functionalities of the ontology, and finally to the degree of topic coverage in a domain.

Another test performed is the evaluation of the "subClassOf" relation between the exact matching and the other extracted concepts. The "subClassOf" relation is also generated by Text2Onto. The idea behind this is to test the possibility of validating new and closely matching concepts out of their "subClassOf" relation with exactly matching concepts. The results are displayed in Table 4.2. We can deduce that they are not very promising, and that we have to rely on other techniques to decide on the relevance of new discovered concepts.

Concluding Text2Onto's manual evalution of the extraction performed on 20 news articles:

– Information extracted from unstructured sources contain a certain amount of noise.

– Determining the relevance of extracted terms to the domain is not an isolated process. However, various issues should be taken into consideration such as the ontology functionality and domain coverage.

4.3 Concepts Extracted Vs. Base Ontology Concepts

After the manual evaluation of information extraction on a small number of documents, we performed the extraction of concepts from all the KMi news articles corpus. At the time of experiment, the corpus includes 986 KMi news articles. They have been extracted from the web using a web content downloader tool called httrack[2]. We have sequentially batch processed 40 news articles, as Text2Onto could not handle all the corpus at one time. The concepts are extracted based on the Term Frequency Inverse Document Frequency (TFIDF) algorithm. A total of 4979 concepts are extracted.

The *string matching* process automatically analyzes the output of Text2Onto concept extraction based on the Jaro distance metric string similarity with a 0.92 threshold. The Jaro distance metric similarity is based on the number and positions of the characters in common between two strings. This string similarity technique performs well on short strings [7].

[2]http://www.httrack.com/

The comparison is performed against the base ontology concepts. Results show that out of 256 concepts in the base ontology, 50 concepts exactly match and 22 concepts are closely matching (with a similarity between 0.92 and 1) with the extracted concepts. This leaves us with 4907 concepts newly discovered by Text2Onto. Results are visualized in Figure 4.1.

In the coming section we present our experiment for finding the links between the new discovered concept, and the base ontology concepts.

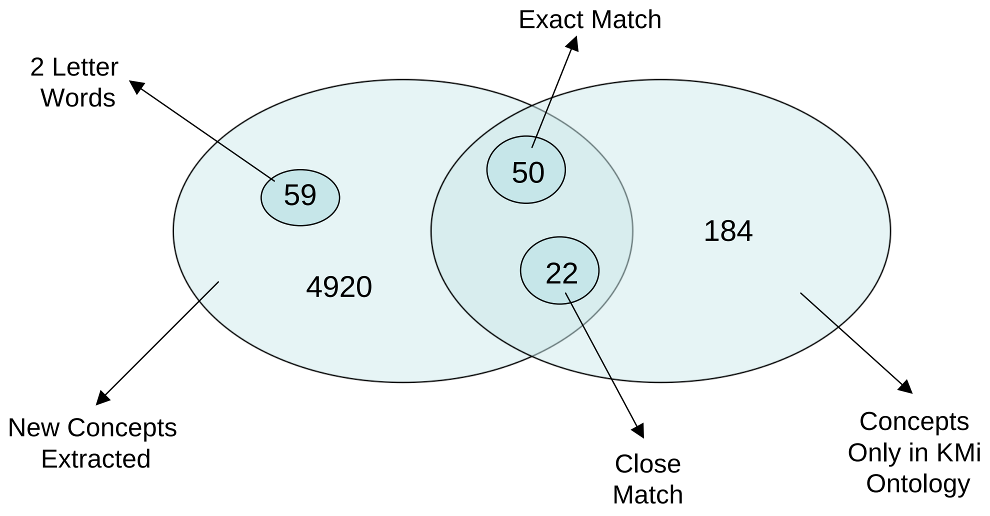


Figure 4.1: Extracted Concepts Compared to the Base Ontology

4.4 Relation Discovery Experiment

We performed an experimental evaluation of the current implementation of the relation discovery module on the data sets provided by the KMi scenario[3]. Our goal was to answer three main questions. First, 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. Second, we wished to understand the main reasons behind the incorrect relations, leading to ways of 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 base ontology.

The WordNet based relation discovery makes use of the Wu and Palmer similarity [39]

[3]Major parts of this section will appear in [41]

for identifying the best similarity measure between the two terms. As per Figure 4.2 This measure is computed according to the following formula:

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

where N1 is the number of nodes on the path from C1 to C3 (the least common superconcept 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 [39]. For those terms that are most closely related to each other, we derive a subsumption relation by exploring WordNet's hierarchy using a functionality built into its Java library[4]. This will result in a relation between a term, as well as an inference path which lead to its discovery.

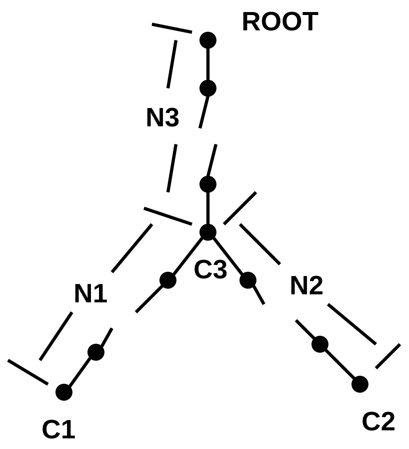


Figure 4.2: The Wu and Palmer Concept Similarity Measure

The terms that could not be related to the base ontology are forwarded to the next module which makes use of online ontologies. For this component, we rely on the Scarlet relation discovery engine[5]. Scarlet [28] automatically selects and explores online ontologies *to discover relations between two given concepts.* For example, when relating two concepts labeled *Researcher* and *AcademicStaff*, Scarlet 1) identifies (at run-time) online ontologies that can provide information about how these two concepts inter-relate and then 2) combines this information to infer their relation. [28] describes 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 *Researcher* $\sqsubseteq$ *AcademicStaff.* Besides

[4]http://jwordnet.sourceforge.net/
[5]http://scarlet.open.ac.uk/

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

Table 4.3: Examples of Relations Derived by Using WordNet

subsumption relations, 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. For example, when matching two concepts labeled *Drinking Water* and *tap_water*, appropriate anchor terms are discovered in the TAP ontology and the following subsumption chain in the external ontology is used to deduce a subsumption relation: *DrinkingWater* $\sqsubseteq$ *FlatDrinkingWater* $\sqsubseteq$ *TapWater*. Note, that as in the case of WordNet, the derived relations are accompanied by a path of inferences that lead to them.

4.4.1 Experimental Data

We relied on randomly selected 20 documents from KMi's news repository as a source for potentially new information. Text2Onto's extraction algorithm [6] discovered 520 terms from these texts.

As already mentioned, the base ontology which we wish to evolve (i.e. KMi's ontology) currently contains 256 concepts, although it has not been updated for well over one year, since April 2007. 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 related to some concepts (i.e. their similarity coefficient is above the threshold of 0.92).

4.4.2 Evaluation of the WordNet based Relation Discovery

Out of the 492 remaining new terms, 162 have been related to concepts of the ontology thanks to the WordNet based relation discovery module. Some of these relations were duplicated as they related the same pair of term and concept through the relation of different synsets. For evaluation purposes, we eliminated duplicate relations and obtained 413 distinct relations (see examples in Table 4.3).

We evaluated a sample of randomly selected 205 relations (i.e. half of the total) in three parallel evaluations. We identified those relations which are considered correct or false, as well as those for which a correctness value could not be decided on ("Don't know"). Results are shown in Table 4.4. We computed a precision value for each evaluator, however, because

	Evaluator 1	**Evaluator 2**	**Evaluator 3**	**Agreed by all**
Correct	106	137	132	76
False	96	53	73	26
Don't know	2	15	0	0
Precision	53 %	73 %	65 %	75 %

Table 4.4: Evaluation Results for the Relations Derived from WordNet.

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 4.4.4.

4.4.3 Evaluation Results for Scarlet

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 4.5.). Some of these relations were duplicates, as the same relation can often be derived from several online ontologies. Duplicate elimination lead to 478 distinct relations.

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.

No.	**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

Table 4.5: Examples of Relations Discovered Using Scarlet

	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 %

Table 4.6: Evaluation Results for the Relations Derived with Scarlet

Our results are shown in Table 4.6, where, as in the case of the WordNet-based relations, precision values were computed both individually 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 the names of the relations did not always make their meanings clear. Different evaluators provided different interpretations for these and thus increased the disagreement levels. Therefore, again, we cannot provide a definitive conclusion of the performance of this particular algorithm. Nevertheless, each evaluator identified more correct than incorrect relations.

4.4.4 Error Analysis

One of the main goals of our 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, either in the WordNet synset hierarchy or in the external ontology in the case of Scarlet. 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 results are shown in Table 4.7. 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.

Another observation was that several relations were derived for the *Thing* concept (e.g. *Lecturer* $\sqsubseteq$ *Thing*). While these relations cannot be considered incorrect, they are of

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 %

Table 4.7: Correlation Between the Length of the Path and the Correctness of a Relation

little relevance for the domain ontology, as they would not contribute in making it evolve in a useful way. Therefore, they should simply be discarded. Similarly, relations that contained in their path abstract concepts such as *Event*, *Individual* or *Resource* tended to be incorrect.

The WordNet experiment helped identifying some of the matching problems we need to handle. Incorrect matching examples in WordNet are listed in table 4.8, which opened additional questions we need to investigate:

- How to determine the context of terms occurring in the corpus? Keeping the link between the extracted term and its corresponding document, with the analysis of other terms in the document could be useful in order to perform word sense disambiguation.

- How to determine the context of terms in the base ontology?

- How to match the term to its right sense in WordNet?

- How to solve the matching of compound extracted term? i.e. how to prevent the matching of "area chart" with "country", due to wrongly anchoring "area chart" to the "area" term in WordNet, which has a relationship path with "country"? We currently have a partial solution to this issue by matching only terms that exist in WordNet, without breaking compound terms.

Finally, during the evaluation we also identified a set of relations to concepts that are not relevant for the domain (e.g. death, doubt). While they sometimes lead to correct relations (e.g. $Death \sqsubseteq Event$), 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.

New Concept	**Existing Concept**	**Wu & Palmer Similarity**	**Relationship Path Based on WordNet**	**Problem Source**
actor	grant	0.941	actor $\sqsupseteq$ grant (Gary Grant)	Wrong sense
area chart	country	1	area chart $\equiv$ country	Wrong anchoring
system	organization	1	system $\equiv$ organization	Wrong sense

Table 4.8: Bad Examples of WordNet Relationship Paths

4.5 Observations on Integrating Relations into the Base Ontology

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", "subClassOf" and "superClassOf". 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.

As we mentioned previously, both our methods provide both a relation and an inference path that lead to its derivation. This makes the integration with the base ontology easier as more information is available.

An interesting situation arises when a 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 4.5, 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 back to Row 5 of Table 4.5, some of the terms in the relation path connecting *Lecturer* to *Book* partially map to labels in the subsumption hierarchy of the base ontology:

$$LecturerInAcademia \sqsubseteq AcademicStaffMember \sqsubseteq HigherEducationalOrganizationEmployee \sqsubseteq EducationalEmployee \sqsubseteq Employee \sqsubseteq AffiliatedPerson \sqsubseteq Person$$

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 4.5 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.

4.6 Experiment Conclusion

Our relation discovery experiments using WordNet and Scarlet helped identifying various possible ways in which the overall quality of the process can be improved. First, it became evident that relations established with abstract concepts or concepts that are poorly related to the base ontology have a low relevance. We could avoid deriving such relations in the first place by simply maintaining a list of common abstract concepts (e.g. $Thing$, *Resource*) that should be avoided. Another approach would be to check the relevance of the terms with respect to the ontology, by measuring their co-occurence in Web documents with concepts from the base ontology.

In addition, the observation of the result of the relation discovery process can lead to the design of a number of heuristic-based methods to improve the general quality of the output. For example, we observed a correlation between the length of the path from which a relation was derived and its quality. Note, however, that more in-depth analysis is needed to verify such hypothesis.

Finally, and most interestingly, the base ontology itself can be used for validating the correctness 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 have also shown cases where during the integration of a path in the ontology, some concepts can be considered equivalent even if their labels only partially match at a syntactic level.

4.7 System Implementation

We are implementing our framework in Java using the Eclipse open development platform[6]. We have so far partially covered information discovery, data validation, and relation discovery in the ontological changes components. At a glance, the current pilot system implementation targets the shaded nodes of Evolva's framework in Figure 4.3. We are not currently focusing on the graphical user interface part as we are planning to integrate our system as a plugin in the NeOn toolkit[7].

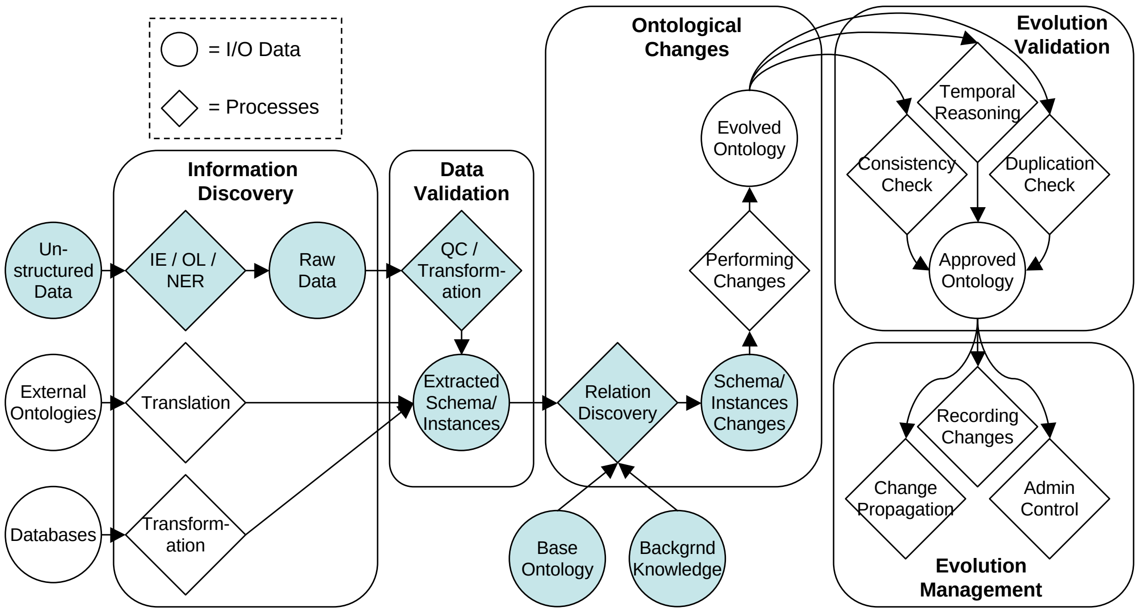
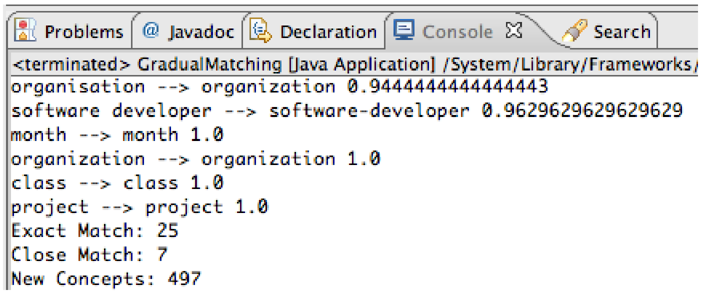


Figure 4.3: Pilot System Current Status Covering Shaded Nodes

4.7.1 String Matching

The implementation started by performing Jaro distance metric string similarity between the list of extracted terms by Text2Onto, and the list of concepts in the KMi ontology. At this stage we used the stand alone application of Text2Onto to process the news documents, and we got the output in an OWL ontology file. Then we performed text parsing for identifying the concepts of Text2Onto output file. This automated string matching helped to quickly identify the new terms extracted by the Text2Onto. In Figure 4.4 we show a sample output of processing the list of concepts occurring in 20 news documents in our system.

[6]http://www.eclipse.org
[7]http://neon-toolkit.org



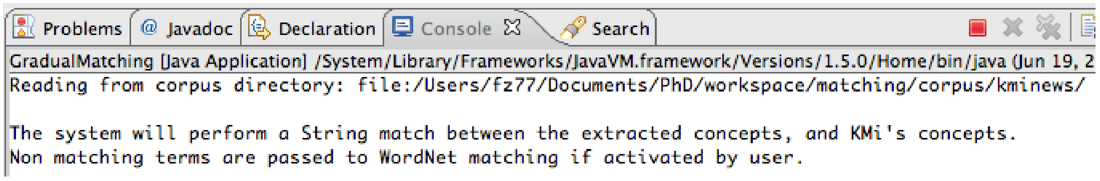
```
<terminated> GradualMatching [Java Application] /System/Library/Frameworks/
organisation --> organization 0.9444444444444443
software developer --> software-developer 0.9629629629629629
month --> month 1.0
organization --> organization 1.0
class --> class 1.0
project --> project 1.0
Exact Match: 25
Close Match: 7
New Concepts: 497
```

Figure 4.4: Screenshot: String Matching Results Example

4.7.2 Text2Onto Integration

In order to present one stand-alone application, we integrated Text2Onto API in our framework. This gave us more flexibility at the level of processing the output of Text2Onto, as now we are able to directly get the concepts and instances or other extracted relations in separate lists. This functionality enabled us to read the corpus documents directly from a directory as shown in Figure 4.5, process the documents, and read the extracted terms for our gradual matching functionality.

We faced some difficulties during the integration, mainly due to compatibility issues between the GATE[8] libraries and the Eclipse environment, as well as other 3rd parties' libraries.



```
GradualMatching [Java Application] /System/Library/Frameworks/JavaVM.framework/Versions/1.5.0/Home/bin/java (Jun 19, 2
Reading from corpus directory: file:/Users/fz77/Documents/PhD/workspace/matching/corpus/kminews/

The system will perform a String match between the extracted concepts, and KMi's concepts.
Non matching terms are passed to WordNet matching if activated by user.
```

Figure 4.5: Screenshot: Reading Corpus Directory

4.7.3 WordNet Relationship Path

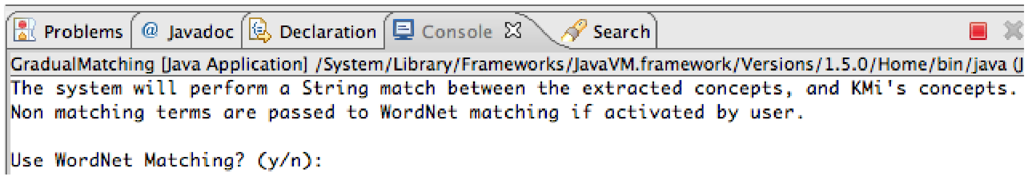
Then we have integrated WordNet for discovering possible relations between *new* concepts discovered from the previous string matching process, and the list of the KMi ontology concepts. As mentioned previously, the Wu and Palmer similarity formula is used, and we only select the best relationships between the terms. The user is asked if he/she would like to use WordNet matching as in Figure 4.6. The output of WordNet matching is sent to a text file for further analysis. The analysis enabled us to spot anchoring and duplication

[8]http://gate.ac.uk

problems, that after solving, we were able to reduce the 3527 WordNet similarities we initially got, to 328 similarities. Below are two examples showing the relationship details between first *blog* and *journal*, and second between *presentation* and *activity*:

```
blog --> journal - Similarity: 0.9333333333333333
[Words: web_log, blog -- (a shared on-line journal where people can post
   diary entries about their personal experiences and hobbies)]
   [PointerType: hypernym]
[Words: diary, journal -- (a daily written record of (usually personal)
   experiences and observations)]

presentation --> activity - Similarity: 0.6666666666666666
[Words: presentation -- (the activity of formally presenting something
   (as a prize or reward); "she gave the trophy but he made the
   presentation")] [PointerType: hypernym]
[Words: ceremony -- (the proper or conventional behavior on some solemn
   occasion; "an inaugural ceremony")] [PointerType: hypernym]
[Words: activity -- (any specific activity; "they avoided all
   recreational activity")]
```



```
GradualMatching [Java Application] /System/Library/Frameworks/JavaVM.framework/Versions/1.5.0/Home/bin/java (J
The system will perform a String match between the extracted concepts, and KMi's concepts.
Non matching terms are passed to WordNet matching if activated by user.

Use WordNet Matching? (y/n):
```

Figure 4.6: Screenshot: Activating WordNet Matching

The Wu and Palmer similarity is good for measuring similarities, but sometimes returns similar terms which do not fall on a direct path. This would make performing ontology changes tricky and not a straight forward process. Thus we put on our implementation schedule the option to resolve only direct relationship paths.

4.7.4 Scarlet Relation Discovery Integration

Scarlet [28] relies on the WATSON [8] libraries for accessing online ontologies. Figure 4.7 illustrates the strategy used with an example where three ontologies are discovered (O_1, O_2, O_3) containing the concepts `A'` and `B'` corresponding to `A` and `B`. The first ontology contains no relation between the anchor concepts, while the other two ontologies declare a subsumption relation. For a given ontology (O_i) the following derivation rules are used:

– if $A'_i \equiv B'_i$ then derive $A \xrightarrow{\equiv} B$;

– if $A_i' \sqsubseteq B_i'$ then derive $A \xrightarrow{\sqsubseteq} B$;

– if $A_i' \sqsupseteq B_i'$ then derive $A \xrightarrow{\sqsupseteq} B$;

– if $A_i' \perp B_i'$ then derive $A \xrightarrow{\perp} B$;

– if $R(A_i', B_i')$ then derive $A \xrightarrow{R} B$.

These rules explore not only direct relations but also relations deduced by applying subsumption reasoning with a given ontology.

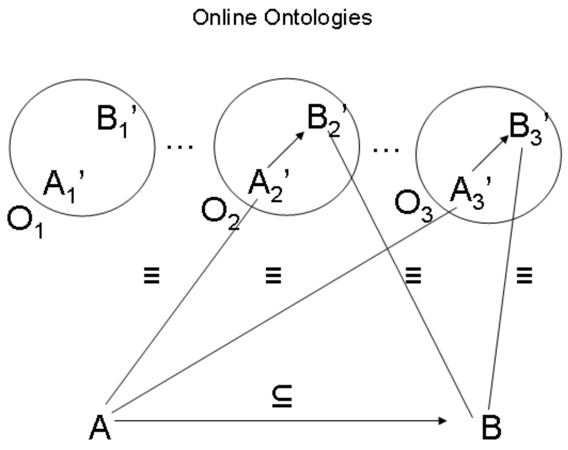


Figure 4.7: Relation Discovery with Scarlet

In Evolva, only the list of terms that are not linked to the base ontology through a WordNet relation are passed for checking using Scarlet. Two relation examples discovered through Scarlet:

```
Relation 1:
PERSON --R_TO-http://travel.org/russia#has_birthplace--> TOWN
Path:
http://travel.org/russia#TOWN--subClassOf-->
http://...#political_region <--R_TO-http://...#has_birthplace--
http://travel.org/russia#PERSON

Relation 2:
MANAGER --subClass--> PERSON
Path:
http://www.aifb.uni-karlsruhe.de/ontology#MANAGER--subClassOf-->
http://www.aifb.uni-karlsruhe.de/ontology#Employee--subClassOf-->
http://www.aifb.uni-karlsruhe.de/ontology#PERSON
```

After implementation, we realized that Scarlet consumes a lot of time, mainly due to the complete path discovery of relations. We decided to later implement a method for batch processing related terms first, without checking all the relation path, and then extract the path of terms that we are sure they are related.

Part 5

Discussion and Future Work

Ontology evolution is a tedious and time consuming task, especially at the level of introducing new knowledge to the ontology. Most of current ontology evolution approaches rely on the ontology curator's expertise to come up with the right integration decisions. Our proposed framework, Evolva, is a solution towards tremendously decreasing or even eliminating user input during ontology evolution. Our work in progress shows preliminary results on information discovery, data validation, as well as relation discovery for performing ontological changes. Our shortcoming plan is to enhance our process of relation discovery in terms of time and precision through:

- Implementation of a filter to exclude very generic terms, not relevant to our ontology.
- Extracting only subsumption relations from WordNet without the use of the Wu and Palmer similarity.
- Using the Web to check the degree of relatedness between the extracted terms and the base ontology entities. This will serve as an automated checker for the relevance of terms with respect to the ontology.
- Try different combinations of background knowledge such as a *parallel* combination instead of *linear* as per our experiment.
- In terms of time performance, we will implement a batch processing method in Scarlet, that will highly improve computational time.

Then our plan is to proceed with our gradual matching technique by exploiting the whole Web as a source of background knowledge. Then we tackle the issue of performing the changes dynamically to the base ontology, and finally focus on the component for validating and managing the evolution. After the implementation phase, we plan to test Evolva in other environments such as the UN's Food and Agriculture Organization (FAO)[1]

[1]http://www.fao.org

domain. We will then work on setting the basis of evaluation procedures, for example by testing the accuracy and degree of coverage of our system by comparing it to an ontology engineer's performance. This will enable further extensions and improvements to Evolva.

Going back to our list of requirements, we discuss in this part how our work is oriented to tackle each point in the list:

R1. The ontology should remain consistent during evolution, through resolving for example time related and duplication inconsistencies.

Evolva's evolution validation component includes a consistency, temporal related and duplication checkers for spotting evolution anomalies. However we acknowledge the fact that not all consistencies could be automatically resolved, but we are aiming at least to spot and return them to the user.

R2. Relevant and validated information should be extracted from various types of data sources: structured, semi-structured and unstructured.

The information discovery in Evolva is designed to extract information from various data sources keeping in mind that each source, with a differing level of structure, should be handled differently. Handling varies both at the level of discovery as well as at the validation level. We are planning to integrate a trust level for our data and background knowledge sources that could help in the data validation process.

R3. For a fast adaptation to changes, ontology evolution should occur with the least or even without user input i.e. dynamic.

Evolva resolves the relationship paths between new terms to integrate in the base ontology through exploiting various background knowledge sources. This is one of the main contributions of our research, and we hope that this will be a step towards performing dynamic ontology evolution. The performance of changes is handled at the ontological changes component level.

R4. Changes should be recorded and propagated to the ontology dependent components, and **R5.** Users should have a degree of control over the evolution.

Ontology changes will be recorded and propagated by Evolva in its evolution management component. Requirement R5 is tackled in the evolution management component as well, by giving a degree of control for the users through the administrator control panel.

R6. Ontology evolution should be as domain independent as possible.

We are aware how challenging is the task of having a domain independent ontology evolution system. However, using a generic approach for harvesting knowledge, we are optimistic about Evolva performing well in different domains. Towards the end of our research, as mentioned in our plan, we will test the performance of Evolva in the FAO domain, and study the level of domain independence.

References

[1] Angeletou, S., Sabou, M., Specia, L., and Motta, E. Bridging the gap between folksonomies and the semantic web: An experience report. *Proc. of the ESWC Workshop on Bridging the Gap between Semantic Web and Web 2.0 2* (2007).

[2] Baker, C. F., Fillmore, C. J., and Lowe, J. B. The berkeley framenet project. *Proceedings of the COLING-ACL* (1998).

[3] Bloehdorn, S., Haase, P., Sure, Y., and Voelker, J. *Ontology Evolution.* John Wiley & Sons, June 2006, pp. 51–70.

[4] Ceravolo, P., Corallo, A., Elia, G., and Zilli, A. Managing ontology evolution via relational constraints. *Proceedings of the Eighth International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES'04), Wellington, New Zealand, September* (2004).

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

[6] Cimiano, P., and Völker, J. Text2onto - a framework for ontology learning and data-driven change discovery. *Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB 2005), Alicante* (2005), 15–17.

[7] Cohen, W. W., Ravikumar, P., and Fienberg, S. E. A comparison of string distance metrics for name-matching tasks. *Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web (IIWeb-03)* (2003).

[8] d'Aquin, M., Baldassarre, C., Gridinoc, L., Sabou, M., Angeletou, S., and Motta, E. Watson: Supporting next generation semantic web applications. *Proc. of WWW/Internet conference* (2007).

[9] Fellbaum, C. *Wordnet: An Electronic Lexical Database.* MIT Press, 1998.

[10] Haase, P., and Stojanovic, L. Consistent evolution of owl ontologies. *Proceedings of the Second European Semantic Web Conference, Heraklion, Greece, 2005* (2005), 182–197.

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

[12] Heflin, J. *Towards the Semantic Web: Knowledge Representation in a Dynamic, Distributed Environment.* PhD thesis, University of Maryland, College Park, 2001.

[13] Heflin, J., and Hendler, J. A. Dynamic ontologies on the web. pp. 443–449.

[14] Klein, M. *Change Management for Distributed Ontologies.* PhD thesis, Vrije Universiteit in Amsterdam, 2004.

[15] Klein, M., Kiryakov, A., Ognyanov, D., and Fensel, D. Finding and characterizing changes in ontologies. *Conceptual Modeling-ER 2002: 21st International Conference on Conceptual Modeling, Tampere, Finland* (2002).

[16] Klein, M., and Noy, N. F. A component-based framework for ontology evolution. *Ontologies and Distributed Systems* (2003).

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

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

[19] Luke, S., Spector, L., Rager, D., and Handler, J. Ontology-based web agents. W. L. Johnson and B. Hayes-Roth, Eds., ACM Press, pp. 59–68.

[20] Maedche, A., Motik, B., Stojanovic, L., Studer, R., and Volz, R. Managing multiple ontologies and ontology evolution in ontologging. *Proceedings of the Conference on Intelligent Information Processing, World Computer Congress* (2002).

[21] Maynard, D., Peters, W., d'Aquin, M., and Sabou, M. Change management for metadata evolution. *International Workshop on Ontology Dynamics (IWOD-07)* (2007).

[22] Novacek, V., Laera, L., and Handschuh, S. Semi-automatic integration of learned ontologies into a collaborative framework. *International Workshop on Ontology Dynamics (IWOD-07)* (2007).

[23] Noy, N. F., Chugh, A., Liu, W., and Musen, M. A. A framework for ontology evolution in collaborative environments. *Proceedings of the 5th Int. Semantic Web Conference (ISWC?06). Athens, Georgia, USA* (2006), 544–558.

[24] Noy, N. F., and Klein, M. Ontology evolution: Not the same as schema evolution. *Knowledge and Information Systems 6* (2004), 428–440.

[25] Noy, N. F., Kunnatur, S., Klein, M., and Musen, M. A. Tracking changes during ontology evolution. *Proceedings of the Third International Conference on the Semantic Web (ISWC'04)* (2004).

[26] Ottens, K., and Glize, P. A multi-agent system for building dynamic ontologies. *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems* (2007).

[27] Rogozan, D., and Paquette, G. Managing ontology changes on the semantic web. *Proceedings of the International Conference on Web Intelligence (WI'05)* (2005), 430–433.

[28] Sabou, M., d'Aquin, M., and Motta, E. Exploring the semantic web as background knowledge for ontology matching. *Journal on Data Semantics* (2008).

[29] Siorpaes, K. Lightweight community-driven ontology evolution. *Doctoral Symposium of the ISWC 4825* (2007), 951.

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

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

[32] Stojanovic, L., Stojanovic, N., and Handschuh, S. Evolution of the metadata in the ontology-based knowledge management systems. *German Workshop on Experience Management* (2002), 65–77.

[33] Tallis, M., and Gil, Y. Designing scripts to guide users in modifying knowledge-based systems. *Proceedings of the Sixteenth National Conference on Artificial Intelligence* (1999), 242–249.

[34] Tamma, V., and Bench-Capon, T. A conceptual model to facilitate knowledge sharing in multi-agent systems. *Proceedings of the OAS* (2001), 69–76.

[35] Velardi, P., Fabriani, P., and Missikoff, M. Using text processing techniques to automatically enrich a domain ontology. *Proceedings of the international conference on Formal Ontology in Information Systems-Volume 2001* (2001), 270–284.

[36] Volkel, M., and Groza, T. Semversion: An rdf-based ontology versioning system. *Proceedings of the IADIS International Conference WWW/Internet (ICWI 2006), Murcia, Spain* (2006).

[37] Volz, R., Oberle, D., Staab, S., and Motik, B. Kaon server-a semantic web management system. *Alternate Track Proceedings of the Twelfth International World Wide Web Conference, WWW2003, Budapest, Hungary* (2003), 20–24.

[38] Vrandecic, D., Pinto, H. S., Sure, Y., and Tempich, C. The diligent knowledge processes. *Journal of Knowledge Management 9* (2005), 85–96.

[39] Wu, Z., and Palmer, M. Verb semantics and lexical selection. *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics* (1994), 133–138.

[40] Zablith, F. Dynamic ontology evolution. *ISWC Doctoral Consortium* (2008).

[41] Zablith, F., Sabou, M., d'Aquin, M., and Motta, E. Using background knowledge for ontology evolution. *To appear in: Proceedings of the ISWC International Workshop on Ontology Dynamics (IWOD)* (2008).