USE OF ONTOLOGIES IN INFORMATION EXTRACTION

by

DAYA CHINTHANA WIMALASURIYA

A DISSERTATION

Presented to the Department of Computer and Information Science
and the Graduate School of the University of Oregon
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

March 2011

DISSERTATION APPROVAL PAGE

Student: Daya Chinthana Wimalasuriya

Title: Use of Ontologies in Information Extraction

This dissertation has been accepted and approved in partial fulfillment of the requirements for the Doctor of Philosophy degree in the Department of Computer and Information Science by:

Dr. Dejing Dou	Chair
Dr. Arthur Farley	Member
Dr. Michal Young	Member
Dr. Monte Westerfield	Outside Member

and

Richard Linton	Vice President for Research and Graduate Studies/ Dean of the Graduate School

Original approval signatures are on file with the University of Oregon Graduate School.

Degree awarded March 2011

DISSERTATION ABSTRACT

Daya Chinthana Wimalasuriya

Doctor of Philosophy

Department of Computer and Information Science

March 2011

Title: Use of Ontologies in Information Extraction

Information extraction (IE) aims to recognize and retrieve certain types of information from natural language text. For instance, an information extraction system may extract key geopolitical indicators about countries from a set of web pages while ignoring other types of information. IE has existed as a research field for a few decades, and ontology-based information extraction (OBIE) has recently emerged as one of its subfields. Here, the general idea is to use ontologies - which provide formal and explicit specifications of shared conceptualizations - to guide the information extraction process. This dissertation presents two novel directions for ontology-based information extraction in which ontologies are used to improve the information extraction process.

First, I describe how a component-based approach for information extraction can be designed through the use of ontologies in information extraction. A key idea in this approach is identifying components of information extraction systems which make extractions with respect to specific ontological concepts. These components are termed "information extractors". The component-based approach explores how information extractors as well as other types of components can

be used in developing information extraction systems. This approach has the potential to make a significant contribution towards the widespread usage and commercialization of information extraction.

Second, I describe how an ontology-based information extraction system can make use of multiple ontologies. Almost all previous systems use a single ontology, although multiple ontologies are available for most domains. Using multiple ontologies in information extraction has the potential to extract more information from text and thus leads to an improvement in performance measures. The concept of information extractor, conceived in the component-based approach for information extraction, is used in designing the principles for accommodating multiple ontologies in an ontology-based information extraction system.

CURRICULUM VITAE

NAME OF AUTHOR: Daya Chinthana Wimalasuriya

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

University of Oregon, Eugene, Oregon, USA

University of Moratuwa, Moratuwa, Sri Lanka

DEGREES AWARDED:

Doctor of Philosophy, University of Oregon, Computer and Information Science, 2011.

Master of Science, University of Oregon, Computer and Information Science, 2006.

Bachelor of Science, University of Moratuwa, Engineering, 2002.

ACKNOWLEDGEMENTS

Although I am the author of this dissertation, it would not have been produced if not for the efforts of so many people. I am heartily thankful to all of them. First and foremost, I would like thank my advisor, Dr. Dejing Dou, who provided guidance to me through all the steps that led to this dissertation. In addition to rigourously assessing my work and helping me improve as a researcher, he provided motivational support whenever there was a serious difficulty. I am also grateful to my dissertation committee members, who provided immense support in conducting research work and in preparing this dissertation. I would also like to thank the faculty and staff of the Computer and Information Science (CIS) Department as well as my fellow students who helped me in countless ways. Last but not least, I would like to thank my parents and my wife who supported me through all the years I spent on graduate studies.

Dedicated to Dilini Nisansala Wimalasuriya.

TABLE OF CONTENTS

Chapter Page

LIST OF FIGURES

Figure Page

LIST OF TABLES

Table Page

CHAPTER I

INTRODUCTION

The objective of information extraction (IE) is identifying and retrieving or *extracting* some information from natural language text. It does not attempt to comprehensively analyze and understand natural language as it is concerned with only certain types of information. Information extraction systems ignore other types of information present in the text. Because of this restricted focus, information extraction is a much more manageable task than comprehending natural language, which is the aim of natural language understanding (NLU).

Russell and Norvig, in their widely used textbook on artificial intelligence [88], state that information extraction aims to process natural language text to retrieve occurrences of a particular class of objects or events and occurrences of relationships among them. Presenting a similar view Riloff states that information extraction is a form of natural language processing in which certain types of information must be recognized and extracted from text [85]. The focus of information extraction is natural language, as identified by these definitions, and as such it is seen as a subfield of natural language processing (NLP).

A system that processes a set of web pages and extracts information regarding countries and their political, economic and social indicators can be presented as an example information extraction system. Some kind of model that specifies what to look for (e.g., country name, population, capital, main cities, etc.) is needed to guide this process. The system will attempt to find information matching this model and ignore other types of data.

Information extraction systems are different from information retrieval (IR) systems. The objective of information retrieval is finding documents that are relevant to a user's needs from a collection of documents. Search engines of the Web generally function as information retrieval systems. Information extraction systems go one step further by extracting the information from the text and presenting them to the user instead of returning a link to a document and leaving the task of looking up and finding the information to the user. Returning to the example of countries and their important indicators presented in the previous paragraph, for a query on the capital of US, an information retrieval system would return the links to a set of web pages containing the information (e.g., US government pages and Wikipedia pages) whereas an information extraction system would directly present the answer as "Washington D.C.".[1]

Information extraction has existed as a field for a few decades and has experienced a significant development since 1990's partly due to the Message Understanding Conferences (MUC). These conferences have provided standard extraction tasks and evaluation criteria and has led to an objective evaluation of different information extraction techniques. As a result, researchers have concentrated on the promising information extraction techniques developing better systems over the years. Out of such research work, two techniques have emerged as the dominant techniques for information extraction, namely *machine learning* and *extraction rules*.

In using machine learning for information extraction, classification is the technique generally applied. Here, information extraction is converted into a set of classification problems. For instance, information extraction can be performed on a

[1]For this particular example, most search engines provide the answer in addition to the links. This is an indication that they operate as information extraction systems for certain queries.

document by first identifying the sentences that contain the required information and then identifying the relevant words within sentences. It can be seen that classification can handle both these tasks.

The extraction rules technique relies on patterns expressed as regular expressions to extract the required information from text. For example, the expression `(belonged|belongs) to <NP>`, where `<NP>` denotes a noun phrase, might capture the names of organizations in a set of news articles. Such extraction rules are normally hand-crafted but there have been some work on employing machine learning techniques to discover them.

There has been surge of interest in information extraction in the recent past as evidenced by the advanced information extraction systems that have been developed recently. Such systems include TextRunner [38], KIM [84], Kylin [100] and C-PANKOW [45]. It is reasonable to expect that automatically processing text and finding useful information, which is performed by information extraction, will become more and more important in the future given the exponential growth of the World Wide Web and the textual data held by organizations. It should be noted that according to some estimates [86], around 80% of data in a typical corporation are in natural language.

The emergence of ontology-based information extraction (OBIE) as a subfield of information extraction can also be seen as a result of the recent rapid development of the field. Here, the general idea is to use ontologies to guide the information extraction process. In computer science, the study of ontologies has originated from the field of knowledge representation (KR). It is the subfield of artificial intelligence concerned with designing and using systems for storing knowledge [7]. The formal conceptualizations provided by ontologies are directly

usable by such systems. The concept of an ontology has originated in the field of philosophy, where ontologies have been used hundreds or thousands of years before they were adopted by knowledge representation.

An ontology has been defined as a formal and explicit specification of a shared conceptualization [55, 91]. Generally, ontologies are specified for specific domains, which are known as "domain ontologies." Since information extraction is essentially concerned with the task of retrieving information for a particular domain, formally and explicitly specifying the concepts of that domain through an ontology can be helpful to this process. For example, a geopolitical ontology that defines concepts like country, province and city can be used to guide the information extraction system described earlier. This is the general idea behind ontology-based information extraction.

One of the most important potentials of ontology-based information extraction is its ability to automatically generate *semantic contents* for the Semantic Web. As envisioned by Berners-Lee *et al.* [39], the goal of the Semantic Web is to bring meaning to the web, creating an environment where software agents roaming from page to page can readily carry out sophisticated tasks for the users. For this vision to realize, *semantic contents* that can be processed by such agents should be made available. Information contained in ontologies fall under this category because Semantic Web agents are expected to process them automatically. This has been pointed out by several authors including Wu and Weld [101] and Cimiano *et al.* [44].

Since ontology-based information extraction is new field, many interesting research directions related to it are yet to be explored. This dissertation explores two such research directions, which have the potential to make significant

contributions towards improving the information extraction process carried out by ontology-based information extraction systems. Brief description on these two directions are presented below.

1. **Designing a component-based approach for information extraction:**

   A domain ontology contains *classes*, which represent entity sets in the domain (e.g., "Country" and "City" in a geopolitical ontology), and *properties*, which represent relationships between classes (e.g., "has capital" in a geopolitical ontology). Ontology-based information extraction systems can be designed in such a manner that they use independently deployable components with clear interfaces to make extractions with respect to specific classes and properties. We call such components *information extractors*. Once information extractors are implemented in this manner they can be *reused* in other ontology-based information extraction systems, which either uses the same ontological concept (class or property) it has been designed for or a concept that has some relationship with the original concept. Such relationships between ontological concepts are known as *mappings*. This is one key idea behind the designed component-based approach for information extraction. Another key idea is separating domain, corpus and concept specific information from underlying information extraction techniques resulting in what are called *platforms for information extraction*. This makes the reuse of components for information extraction structured and straight-forward.

2. **Using multiple ontologies in information extraction:**

   Almost all previous ontology-based information extraction systems make use of a single ontology although multiple ontologies are available for most

domains. Such multiple ontologies either specialize on sub-domains or provide different perspectives on the same domain, which can be seen as two scenarios for having multiple ontologies for the same domain. Using more than one ontology in an information extraction system is interesting because it has the potential to make more extractions and thus lead to an improvement in performance measures. In designing information extraction systems that use multiple ontologies, principles have to be developed to accommodate multiple ontologies and to facilitate interaction between them. The concept of information extractor, conceived under the component-based approach for information extraction, is used in this exercise.

We call the designed component-based approach for information extraction, "OBCIE (Ontology-Based Components for Information Extraction)". The term "MOBIE (Multiple Ontology-Based Information Extraction)" is used for the studies on the use of multiple ontologies in information extraction.

The remainder of this dissertation is organized as follows. In Chapter II, we discuss the background areas related to the original research work presented. The main contributions of this dissertation are presented in Chapters III, IV, V and VI. Chapters III and IV discuss the theory and the implementation details of the component-based approach for information extraction. Similarly, Chapters V and VI are dedicated to the theory and implementation details related to the use of multiple ontologies in information extraction. Finally, in Chapter VII, we discuss future directions for the research work and provide some concluding remarks.

CHAPTER II

BACKGROUND

This chapter covers the background areas and related work necessary to understand the contributions of this dissertation. It discusses the current state in the field of information extraction as well as its subfield ontology-based information extraction. In addition, it describes the basic concepts of the Semantic Web, which is an important area of application for information extraction. We conclude with a high-level summary of these background areas.

2.1. Information Extraction

Current State of Information Extraction

It is generally agreed that the Message Understanding Conferences (MUC) conducted in 1990's made a significant contribution towards the development of information extraction as a research field. These conferences, which were initiated and financed by the Defense Advanced Research Projects Agency (DARPA), provided standard text corpora and standard evaluation criteria and invited participants to develop information extraction systems to compete with each other. This led to an objective evaluation of information extraction systems and techniques and allowed researchers to identify effective information extraction techniques. Most of the currently used information extraction techniques have their origins in the Message Understanding Conferences. For example, the "extraction rules" information extraction technique mentioned in Chapter I, which relies on patterns expressed as regular expressions to make extractions, attracted the

attention of researchers after the FASTUS system [35], which was largely based on this technique, came in the second place in the 4th MUC.

In the early days of information extraction, it was seen as useful mostly in military applications. This is evident by the funding of the MUCs by DARPA as well as the topics selected for the first few MUCs: the first two MUCs were on fleet operations while the next two were on terrorist activities of Latin American countries [92]. However, it was soon realized that information extraction is very useful for business organizations. As such, business-related domains were used by subsequent MUCs as well as the Automated Content Extraction (ACE) conferences, which are the successor to the Message Understanding Conferences. These applications are being seen as an important area in the emerging field of business intelligence, which is defined as the process of finding, gathering, aggregating, and analyzing information for decision making, typically in a corporate environment [89]. Having an effective business intelligence system is seen as providing a competitive advantage to a business organization. In addition, information extraction systems are being increasingly used in bioinformatics as a result of the exponential growth of bioinformatics literature and the resulting difficulty of processing all of them manually. To address this situation, several competitions such as BioNLP [4] and BioCreative [3] have been organized for information extractions systems that target bioinformatics text.

The growth of the World Wide Web has also had a significant impact on information extraction. Since a large portion of the information available in the web takes the form of natural language text, the necessity to convert them into a machine processable format has been recognized from the early days of the web. For instance, in late 1990's there have been several attempts to develop

what have been called *wrappers* for web pages, which extract some structured information from them [36, 67]. These efforts did not catchup with development of the Web because of its rapid growth as well as the difficulty and cost of developing effective wrappers for web pages. Still, the enormous amount of text available in the web has continued to be seen as both an important target for information extraction systems as well as a resource that can be used in developing new information extraction techniques. For instance, there have been attempts to develop information extractions systems that leverage the information available in Wikipedia (the Kylin system [101]) and to develop a generic information extraction technique that uses the web as a corpus (the PANKOW system [44]). These are described in detail later in this chapter.

The emergence of the Semantic Web as a research area as well as a practical application has added another dimension to the development of information extraction. As envisioned by Berners-Lee *et al.*, the goal of the Semantic Web is to bring meaning to the web, creating an environment where software agents roaming from page to page can readily carry out sophisticated tasks for the users [39]. It can be seen that information extraction will be a useful tool in realizing the vision of the Semantic Web. As mentioned in Chapter I, several authors have pointed out that information extraction can be used to generate semantic contents for the Semantic Web. This task is best handled by its new subfield ontology-based information extraction. To a certain extent, this can be seen as a repetition of the earlier attempts to develop *wrappers* for web pages. However, due to the development of information extraction and the formality provided by ontologies, this challenge can be addressed more effectively now.

Before concluding this section, we will discuss the main performance measures used in information extraction, which are *precision*, *recall* and *F1-measure*. Precision shows the fraction of correct extractions out of all the extractions made whereas recall shows the fraction of correct extractions made out of all possible correct extractions. With $\{Relevant\}$ and $\{Retrieved\}$ representing the set of all possible correct extractions and the set of extractions made respectively, precision and recall are defined as follows [59].

Definition 2.1 (Precision)**.**

$$Precision = \frac{|\{Relevant\}| \cap |\{Retrieved\}|}{|\{Retrieved\}|}$$

Definition 2.2 (Recall)**.**

$$Recall = \frac{|\{Relevant\}| \cap |\{Retrieved\}|}{|\{Relevant\}|}$$

The metrics of precision and recall are used in information retrieval as well, where $\{Relevant\}$ and $\{Retrieved\}$ denote the sets of all relevant documents, which should ideally be returned by an information retrieval systems in response to a query, and the set of retrieved documents respectively.

It can be seen that a system can improve recall at the expense of precision by making many extractions. Similarly, precision can be improved at the expense of recall by making only few extractions that are highly likely to be accurate. Hence, a metric that takes both precision and recall into consideration is necessary to accurately evaluate the performance of an information extraction system. F1-score,

which is the harmonic mean between precision and recall, is widely used for this purpose. It is defined as follows [59].

Definition 2.3 (F1-score)**.**

$$F1 - Score \;=\; 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Challenges in Information Extraction

Despite its rapid development as a research field, information extraction faces some serious challenges. The following are two of such important challenges.

1. Achieving widespread usage and commercialization
2. Coming up with appropriate models for representing the extracted information

The *first challenge* is a result of more than one factor and represents the obstacles faced by the field in developing real world applications as opposed to research-oriented systems. It can be seen that information extraction still does not enjoy widespread usage and commercialization especially when compared with the field of information retrieval. As mentioned earlier, the goal of information retrieval is finding the documents related to a user's request from a large collection of documents. This field has given rise to many widely used systems including the search engines of the web. It can be argued that information extraction systems should be more useful than information retrieval system because they provide the required information itself instead of a pointer to a document. Yet, their usage is very low when compared with information retrieval systems.

The costs and complexity associated with setting up an information extraction system in a new domain or a new text corpora can be seen as one main factor hindering the widespread usage of information extraction. This results in serious difficulties in applying information extraction techniques in new situations as well as in developing large scale information extraction systems that work on different domains and different text corpora.

The following can be identified as two factors that give rise to the costs and complexity associated with the implementation of information extraction systems.

1. The requirement of *templates*: As described by Wilks and Brewster [99], information extraction systems normally require a *set of templates* for a particular domain and merely attempt to fill the templates. Generating these *templates* manually beforehand is not practical in at least some situations.

2. Bundling domain and corpus specific information with the information extraction techniques: Information extraction systems are often built as monolithic systems where no clear separation is made between domain and corpus specific information used by the implementation and the underlying information extraction technique. This makes the application of the information extraction system in a new situation difficult.

The first problem mentioned above is targeted by the emerging paradigm of "open information extraction," which aims to discover relations of interest from text instead of being provided in advance [38]. For instance, the TextRunner system [38], which is based on this paradigm, is capable of extracting relations using some grammatical structures and data tuples that fit into these relations in a single pass over the corpus. Open information extraction avoids the second problem

by using information extraction techniques that do not use any domain or corpus specific information.

Although open information extraction constitutes a significant improvement, it does not provide a complete solution for the problems that prevent information extraction systems being widely used and commercialized. One main issue here is its reliance on domain and corpus independent information extraction techniques. Our component-based approach for information extraction attempts to provide a different solution for these problems.

The *second challenge* is related to making use of the extracted information. In many cases, the extracted information is intended to be used by an application and for this to happen the information should be stored in a format that can be efficiently and effectively processed by such applications. In the early days of information extraction, the relational model was seen as the natural choice for this purpose. Due to the prevalence of this view, information extraction has even been described as the process of creating database entries from text [88]. With the development of the field, it has been realized that relational models are not sufficient for some applications that make use of the extracted information. One option used in this situation is UML (Unified Modeling Language) [33], which has been adopted by some information extraction systems such as those based on the UIMA (Unstructured Information Management Architecture) framework [2]. Another option is the use of ontologies as discussed earlier. It can be seen that the use of ontologies provides several advantages such as supporting reasoning based on logic. However, even when ontologies are used to model the extracted information, finding a suitable ontology is a concern. Our work on the use of multiple ontologies

in information extraction attempts to addresses this challenge by accommodating the use of more than one ontology to represent the information.

2.2. Ontology-Based Information Extraction

In this section we provide a thorough review of ontology-based information extraction. Since this dissertation is primarily in this area, such a review is necessary to understand its contributions. We begin this review by providing a definition for an ontology-based information extraction system and then move onto describe a common architecture for these systems. Then we categorize existing ontology-based information extraction systems along a set of dimensions we have identified. We conclude this section by discussing some important implementation details of current systems and some metrics designed specifically for ontology-based information extraction.

A Definition

The general idea behind ontology-based information extraction is using ontologies to guide the information extraction process, as mentioned earlier. Since this is a new field only few years old, it appears that researchers have not completely agreed on the definition for an ontology-based information extraction system. As such we attempt to arrive at such a definition by reviewing the literature of the field.

A formal definition for an ontology has to be adopted in providing a definition for an ontology-based information extraction system. We use the following widely used definition provided by Studer *et al.* [91]. It is a refinement of a definition provided by Gruber [55].

Definition 2.4 (Ontology)**.** An ontology is a formal and explicit specification of a shared conceptualization.

We have attempted to arrive at a definition for an ontology-based information extraction system by identifying the key characteristics of OBIE systems discussed in the literature, concentrating on the factors that make OBIE systems different from general IE systems. These are presented below.

1. *Process unstructured or semi-structured natural language text:* Since ontology-based information extraction is a subfield of information extraction, which is seen as a subfield natural language processing, it is reasonable to limit the inputs to natural language text. They can be either unstructured (e.g., text files) or semi-structured (e.g., web pages using a particular template such as pages from Wikipedia). Systems that use images, diagrams or videos as input cannot thus be categorized as ontology-based information extraction systems.

2. *Present the output using ontologies:* Li *et al.* [69] identify the use of a formal ontology as one of the system inputs and as the target output as an important characteristic that distinguishes ontology-based information extraction systems from information extraction systems. While this statement holds true for most ontology-based information extraction systems, there are some systems that construct the ontology to be used through the information extraction process itself instead of treating it as an input (e.g., the Kylin system [101]). Since constructing an ontology in this manner should not disqualify a system from being an OBIE system, we believe that it is prudent to remove the requirement to have an ontology as an input for the system.

However, the requirement to represent the output using ontologies appears to be reasonable.

3. *Use an information extraction process guided by an ontology:* We believe that "guide" is a suitable word to describe the interaction between the ontology and the information extraction process in an OBIE system: in all OBIE systems, the information extraction process is guided by the ontology to extract things such as classes, properties and instances. This means that no new information extraction method is invented but an existing method is oriented to identify the components of an ontology.

Combing these factors with the definition for information extraction provided by Riloff [85], we provide the following definition for an ontology-based information extraction system.

Definition 2.5 (Ontology-Based Information Extraction System)**.** An ontology-based information extraction system is a system that processes unstructured or semi-structured natural language text through a mechanism guided by ontologies to extract certain types of information and presents the output using ontologies.

A Common Architecture

Although the implementation details of individual ontology-based information extraction systems are different from each other, a common architecture of such systems can be identified from a higher level. Figure 1 schematically represents this common architecture. This figure represents the union of different components found in different ontology-based information extraction systems. As such, some ontology-based information extraction systems do not contain all the components of

this architecture. For example, the systems that use an ontology defined by others instead of constructing an ontology internally (as discussed later in this section) do not have the "ontology generator" component.

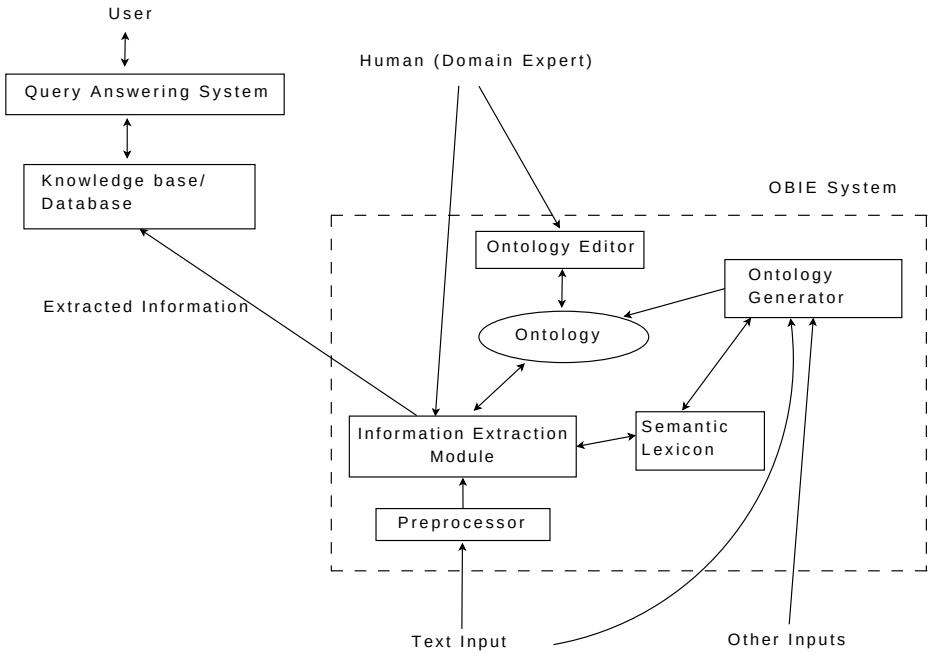


FIGURE 1: Common architecture of OBIE systems

It should also be noted that in some implementations, the ontology-based information extraction system is a part of a larger system that answers user queries based on the information extracted by the system. Figure 1 shows these components as well. But they should not be construed as parts of an ontology-based information extraction system.

As represented by Figure 1, the textual input of an ontology-based information extraction system first goes through a preprocessor component, which converts the text to a format that can be handled by the information extraction

module. For example, this might involve removing tags from an HTML file and converting it into a pure text file.

The information extraction module is where the actual information extraction takes places. This can be implemented using any information extraction technique that is described later in this section. No matter what information extraction technique is used, it is guided by an ontology. A *semantic lexicon* for the language in concern is often used to support this purpose. For example, the WordNet [53] toolkit is widely used for the English language. It groups English words into sets of synonyms called *synsets* and provides semantic relationships between them including a taxonomy.

The ontology that is used by the system may be generated internally by an ontology generator component. This process too might make use of a semantic lexicon. In addition, humans may assist the system in the ontology generation process. This is typically carried out through an ontology editor such as Protégé [19]. Humans may also be involved in the information extraction process in some systems that operate in a semi-automatic manner.

The output of the ontology-based information extraction system consists of the information extracted from the text. They can be represented using an ontology definition language such as the Web Ontology Language (OWL) [16]. In addition, the output might also include links to text documents from which the information was extracted. This is useful in providing a justification for an answer given to a user relying on the extracted information.

As mentioned earlier, the ontology-based information extraction system is a part of a larger query-answering system in some implementations. In such systems, the output of the ontology-based information extraction process is often stored in a

database or a knowledge base. The query answering system makes use of the stored information and answers user queries. It many also include a reasoning component. The nature of the interface provided by the query answering system to the users depends on the particular implementation.

It is insightful to analyze how some ontology-based information extraction systems fit into this architecture. For example, the ontology-based information extraction system implemented by Saggion *et al.* [89] operates as a part of the larger EU MUSING project. The output of the OBIE system is stored in a knowledge base, which is then used by MUSING applications that constitute the query answering system in this case. The ontology to be used by the system is manually defined by domain experts and as such this system does not have an ontology generator component. The information extraction module of this system uses extraction rules and gazetteer lists (which are described later in this chapter). Information extraction operates in a semi-automatic manner, where incorrect extractions made by the process are corrected by humans. Note that the general architecture accommodates this.

For the Kylin information extraction system [101], the input consists of a set of web pages of Wikipedia. Kylin can be considered an ontology-based information extraction system because it extracts information with respect to the structures of "infoboxes" of Wikipedia, which are organized into an ontology. (An infobox presents a summary of the content of a page in Wikipedia.) The selected files of Wikipedia go through a preprocessor before being used by the information extraction module. In this case, the information extraction module employs classification information extraction technique. The ontology is constructed by a special component named "Kylin Ontology Generator". This ontology generator

makes use of the structures of Wikipedia infoboxes and WordNet [53], which is a semantic lexicon for the English language as mentioned earlier. The information extracted by Kylin are not expected to be directly used in a knowledge base or a database but they are used in developing a "communal correction" system for Wikipedia, which allows the users to verify and correct the extractions of Kylin [98]. Note that this can also be viewed as an instance where humans interact with the information extraction module as allowed by the architecture.

Classification of Current OBIE systems

In this section, we provide a classification of current ontology-based information extraction systems along a set of dimensions that we have identified in order to obtain a better understanding of their operation. The following are the dimensions we use.

1. Information extraction technique
2. Ontology construction and updated
3. Components of the ontology extracted
4. Types of sources

The first dimension represents different information extraction techniques that have been designed over the years. Moens has presented a comprehensive categorization and analysis of these techniques in the form of a textbook [80]. Most of these techniques have been adopted by OBIE systems. The following are the main information techniques used by the OBIE systems we have reviewed.

1. *Extraction rules:* As mentioned in Chapter I, this is one of the two dominant information extraction techniques used by information extraction systems. Here, the general idea is to use regular expressions as patterns to extract information. Returning to the example we have discussed in Chapter I, the expression `(belonged|belongs) to <NP>`, where `<NP>` denotes a noun phrase, might capture the names of organizations in a set of news articles. Some incorrect extractions will clearly be made but by carefully selecting appropriate rules good performance measures can be achieved when using this technique. The set of regular expressions representing the rules are often implemented using finite-state transducers which consist of a series of finite-state automata.

   The FASTUS information extraction system [35], which was implemented in 1993 and participated in the 4th Message Understanding Conference, appears to be one of the earliest systems to use this method. The General Architecture for Text Engineering (GATE) [8], which is a widely used NLP framework, provides an easy to use platform to employ this technique.

   Embley's OBIE systems [52] appear to be some of the first OBIE systems to use this technique. Following Embley, Yildiz and Miksch have employed a similar technique in their ontoX system [103]. Extraction rules are also used by the Textpresso system [81] for biological literature, which is more of an information retrieval system but can be seen as an ontology-based information extraction system as well. The same principle is employed to construct an ontology in the implementation by Hwang [65]. In addition, ontology-based information extraction systems that use the GATE architecture, such as

KIM [84] and the implementation by Saggion *et al.* [89] rely at least partly on this technique.

The above mentioned ontology-based information extraction systems use hand-crafted (manually identified) rules. This means that a person or a group of persons have to read all the documents of the corpus in concern and figure out suitable extraction rules. It can be seen that this a tedious and time consuming exercise which does not scale well. In order to address this issue, some systems have aimed to automatically mine extraction rules from text. Vargas-Vera *et al.* have designed and implemented an OBIE system that operates on these principles back in 2000 [96]. They have used a dictionary induction tool named Crystal to identify extraction rules. This tool operates on the principles of the inductive learning algorithm and searches for the most specific generalization that covers all positive instances. Other techniques for inducting extraction rules have been designed by Romano *et al.* [87], who have approached this problem using an algorithm for the longest common subsequence problem (the problem of finding the longest subsequence common to all sequences in a set of sequences) and Ciravegna [46], who has designed an algorithm named LP^2 that makes extensive use of machine learning techniques for this purpose.

2. *Gazetteers:* This technique relies on finite-state automata just like extraction rules but recognizes individual words or phrases instead of patterns. The words to be recognized are provided to the system in the form of a list, known as a gazetteer. This technique is widely used in the *named-entity recognition* task, which can be seen as a subtask of information extraction. It is concerned with identifying individual entities of a particular category.

Typically, well-known categories such as the states of the US or countries of the world are used in named entity recognition.

This technique is used by several OBIE systems. These systems often use gazetteer lists containing all the instances of some classes of the ontology. They have been used in the SOBA system [42] to get details about soccer games and in the implementation by Saggion *et al.* [89] to get details about countries and regions.

It can be seen that gazetteers are most effective as an information extraction technique when the information being extracted are instances of a well-known entity type for which an exhaustive list of instance names is available. In addition, if a list of instances is publicly available for a particular category, it can be used to help the information extraction process even if the list is not exhaustive. In such cases, the gazetteer (list) is more useful when used in combination with some other information rather than on its own. For instance, such a gazetteer can be used as one feature in classification-based information extraction (described next).

3. *Classification techniques:* As mentioned in Chapter I, machine learning techniques constitute the other dominant information extraction technique other than extraction rules. Typically, classification is the machine learning technique used for this purpose. Different classification techniques have been used in this manner. Moens provides a comprehensive review of these techniques and categorizes them as "Supervised Classification" techniques [80]. These include sequence tagging techniques such as Hidden Markov Models (HMM) and Conditional Random Fields (CRF) as well.

To carry out information extraction using classification, the problem of information extraction has to be converted into a series of classification problems. Li *et al.* [70] have used two binary classifiers for each entity type (class) being extracted: one determines whether a given word is the start of an instance of the class in concern while the other determines whether it is the end. The Kylin system [101] also uses two binary classifiers for each entity type but adopts a different approach: the first classifier determines whether an instance is found in a sentence and the second classifier determines whether a particular word within a sentence represents an instance. This approach can be termed *two-phase classification* because it operates in two phases, namely sentence level and word level.

Different classification techniques can be used in information extraction. Li *et al.* [70] have used uneven margins support vector machines (SVM) and perceptron techniques. The Kylin system uses the maximum entropy model at sentence level and the conditional random fields (CRF) technique at word level [101]. In addition, the implementation by Li and Bontcheva [69] uses the Hieron large margin algorithm for hierarchical classification [48].

Different features such as Part-Of-Speech (POS) tags, capitalization information, individual words and gazetteers related to the concept in concern can be used as features in classification. Different information extraction systems use different sets of features. It can be seen that the set of features used in information extraction is a valuable resource because they constitute the result of *feature selection*, which is the process of selecting relevant features out of the larger set of candidate features for building learning

models [37]. In this case, a particular model built is used to make extractions with respective to a particular concept.

4. *Analyzing HTML/XML tags:* Information extraction and ontology-based information extraction systems that use HTML or XML pages as input can extract certain types of information using the tags of these documents. For example, a system that is aware of the HTML tags for tables can extract information from tables present in HTML pages. Using this technique, the SOBA ontology-based information extraction system [42] extracts information related to soccer games from the tables of a set of web pages. XML documents would provide more opportunities to extract information in this manner because they allow users to define their own tags.

5. *Web-based search:* Using queries on web-based search engines for information extraction appears to be a new technique. It has not been recognized as an information extraction technique even in the review of information extraction techniques compiled by Moens in 2006 [80]. The general idea behind this approach is using the web as a big corpus.

   Cimiano *et al.* have implemented an OBIE system named "Pattern-based Annotation through Knowledge on the Web (PANKOW)" that semantically annotates a given web page using web-based searches only [44]. It conducts web searches for every combination of identified proper nouns in the document with all the concepts of the ontology for a set of linguistic patterns. Such patterns include Hearst patterns [61] like `<CONCEPT>s such as <INSTANCE>`. The concept labels for the proper nouns are determined based on the aggregate number of hits recorded for each concept. The C-PANKOW

system operates on the same principles but improves performance by taking the context into consideration [45]. The OntoSyphon system uses a similar approach but aims to learn all possible information about some ontological concepts instead of extracting information from a given document [79].

6. *Construction of partial parse trees:* A small number of ontology-based information extraction systems construct a semantically annotated parse tree for the text as a part of the information extraction process. The constructed parse trees are not meant to comprehensively represent the semantic content of the text as aimed by text understanding systems such as TACITUS [62]. Hence, this type of processing can still be categorized under shallow NLP, typically used by information extraction systems, as opposed to deep NLP used by text understanding systems, although they conduct more analysis than other information extraction systems.

   A representative system for OBIE systems that employ this approach is the implementation by Maedche *et al.* [72]. This system, which has been developed for the German language, makes use of a NLP toolkit for the German language named Saarbücker Message Extracting System (SMES). The SMES system consists of several components and operates at the lexical level (words) as well at the clause level. It produces an under-specified dependency structure as the output, which is basically a partial parse tree. This structure is used for information extraction. The Text-To-Onto system developed by Maedche and Staab [73], which uses the SMES system to construct an ontology, is another OBIE system that adopts this approach.

   The Vulcain OBIE system developed by Todirascu *et al.* also makes use of partial parse trees [94]. In addition, this system uses lexical entries for

concepts of the ontology which can be categorized as extraction rules. Therefore, the information extraction technique used by the Vulcain system can be seen as a combination of linguistic extraction rules and partial parse trees.

The second dimension classifies information extraction systems on how the used ontologies are acquired and whether they are updated. In terms of acquiring ontologies, one approach is to consider the ontology as an input to the system. Under this approach, the ontology can be constructed manually or an *off-the-shelf* ontology constructed by others can be used. Most OBIE systems appear to adopt this approach. Such systems include SOBA [42], KIM [84], the implementation by Li and Bontcheva [69], the implementation by Saggion *et al.* and PANKOW [44].

The other approach is to construct an ontology as a part of the ontology-based information extraction process. Ontology construction can be carried out by building an ontology from scratch or by using an existing ontology as the base. Some OBIE systems only construct an ontology and do not extract instances. Such systems include Text-To-Onto [74] and the implementation by Hwang [65]. Kylin (through Kylin Ontology Generator) and the implementation by Maedche *et al.* [72] construct an ontology as a part of the process although their main aim is to identify new instances for the concepts of the ontology.

In addition, it is possible to update the ontology by adding new classes and properties through the information extraction process. Identifying instances and their property values are not considered updates to the ontology here. Such updates can be conducted for both cases mentioned above. However, only few systems update the ontology in this manner. Such systems include the implementations by Maedche *et al.* [72] and Dung and Kameyama [51].

The third dimension classifies information extraction systems based on the components of an ontology that they extract. An ontology consists of several components. These include *classes*, *properties*, *instances* and *property values*. For instance "country" is a class whereas "hasPopulation" and "hasCapital" are properties. Properties are of two types, namely *datatype properties* (e.g., hasPopulation) and *object properties* (e.g., hasCapital assuming that "City" is a class and that a capital has to be a city). Object properties include taxonomical relationships (e.g., a country *is a* political entity) as well as non-taxonomical relationships (e.g., hasCapital). Instances or individuals represent specific objects of classes (e.g., United States is an instance of the country class) while property values represent values for properties for specific instances (e.g., United States has capital Washington D.C.). Based on this background, ontology-based information extraction systems can be categorized on whether they extract classes, properties, individuals, property values or some combination of these.

Some ontology-based information extraction systems only extract classes and properties. These are often termed *ontology induction systems* or *ontology construction systems* [83]. Among such systems, the implementation by Hwang [65], extracts class names and the taxonomy (class hierarchy) only. In contrast, Text-To-Onto [74] discovers class names, taxonomical relationships as well as non-taxonomical relationships.

Most systems only extract individuals and property values and such systems are often termed *ontology population systems* [47, 83, 96]. Many of these systems only extract instance identifiers (names). Such systems include the implementation by Li and Bontcheva [69], PANKOW [44] and OntoSyphon [79]. Some systems

extract property values of the instances as well. Such systems include SOBA [42], the implementation by Embley [52] and the implementation by Saggion *et al.* [89].

Some systems extract classes and properties as well as individuals and property values. Using the terminology of *ontology construction* and *ontology population*, these systems can be described as both constructing and populating the ontology. The Kylin system [101] and the implementation by Maedche *et al.* [72] are two systems falling under this category. They extract class names, taxonomical relationships, non-taxonomical relationships, individuals as well as property values of individuals.

The fourth dimension classifies information extraction systems on the types of sources they use. Although all OBIE systems extract information from natural language text, the sources used by them can be quite different. Some systems are capable of handling any type of natural language text while others have specific requirements for the document structure or target specific web sites.

Many OBIE systems can handle any type of documents including web pages and word-processed documents but require that they be related to a particular domain. Such systems include the implementation by Maedche *et al.* [72], the implementation by Embley [52] and the implementation by Saggion *et al.*[89]. In contrast, SOBA retrieves the web pages that it processes using its own web crawler. It can only handle HTML pages as it makes use of HTML tags in the information extraction process. The Kylin system has been designed specifically for Wikipedia. It makes use of structures specific to Wikipedia pages like infoboxes.

We end this section by providing a summary of the ontology-based information extraction systems discussed throughout this section. Table 1 shows how they are classified on the four dimensions we have discussed.

TABLE 1: Summary of the classification of OBIE systems

System	**Information Extraction Technique**	**Ontology Construction & Update**	**Ontology Components Extracted**	**Types of Sources**
Kylin [101]	Classification, Web-based search	Constructed; not updated.	Classes, taxonomy, data type properties, instances, property values	Wikipedia pages
PANKOW [44]	Web-based Search	Off-the-shelf; not updated	Instances	No restriction
OntoSyphon [79]	Web-based Search	Off-the-shelf; not updated	Instances	No restriction
Maedche *et al.* [72]	Partial parse trees	Constructed; updated	Classes, taxonomy, data type properties, instances, property values	Documents from a domain
Text-To-Onto [74]	Partial parse trees	Constructed; not updated	Classes, taxonomy, other relationships	Documents from a domain
SOBA [42]	Extraction rules, Gazetteer lists, Analyzing tags	Off-the-shelf; not updated	Instances, property values	HTML pages from a domain
Embley [52]	Extraction rules	Manually defined; not updated	Instances, property values	Documents from a domain
Saggion *et al.* [89]	Extraction rules and Gazetteer lists	Manually defined; not updated	Instances, property values	Documents from a domain
Li and Bontcheva. [69]	Classification	Off-the-shelf; not updated	Instances	Documents from a domain
Hwang [65]	Extraction rules	Constructed; not updated	Classes, taxonomy, properties	Documents from a domain

Table 1 – continued

System	Information Extraction Technique	Ontology Construction & Update	Ontology Components Extracted	Types of Sources
ontoX [103]	Extraction rules	Manually defined; not updated	Instances, data type property values	Documents from a domain
Vulcain [94]	Partial parse trees Extraction Rules	Manually defined; not updated	Instances, property values	Documents from a domain
Vargas-Vera *et al.* [96]	Extraction Rules	Manually defined; not updated	Instances, property values	Web pages from a site
KIM [84]	Extraction Rules Gazetteer lists	Manually defined; not updated	Instances, property values	Documents from a domain

Implementation Details and Performance Evaluation

In this section, we provide a summary of the important tools used by ontology-based information extraction systems and the performance measures designed specifically for ontology-based information extraction.

One main category of tools used by OBIE systems is *shallow* NLP tools. The term *shallow* distinguishes these tools from text understanding systems that perform a deeper analysis of natural language. These tools perform functions such as Part-Of-Speech (POS) tagging, sentence splitting and identifying occurrences of regular expressions. They are used by almost all information extraction systems. For example, extraction rules represented by regular expressions can be directly implemented using these tools whereas the features that are used for classification can be extracted using them. Widely used such tools include GATE [8], OpenNLP [15], SProUT [26] and the tools developed by the Stanford Natural Language Processing Group [32]. In addition, the Saarbücker Message Extracting System (SMES) used by Maedche and his group [72, 74] can be categorized as a shallow NLP system. However, it conducts more analysis than other shallow NLP systems as mentioned earlier.

Semantic lexicons, also known as lexical-semantic databases and lexical-semantic nets, also play an important role in many OBIE systems. These tools organize words of a natural language according to meanings and identify relationships between the words. Such relationships related to subsumption are sometimes seen as giving rise to a *lexical ontology*. The information contained in semantic lexicons are utilized in different manners by OBIE systems. For example, the Kylin Ontology Generator uses them to assist the construction of an ontology [102]. For the English language, WordNet [53] is the most widely

used semantic lexicon. Similar tools are available for some other languages such as GermaNet [10] for German and Hindi Wordnet [11] for Hindi.

Ontology editors are also used by most OBIE systems. These tools can be used to manually construct an ontology which is later used by the OBIE system. They can also be used to correct the output of the information extraction process in systems that operate in a semi-automatic manner. Protégé [19] is one of the most widely used tools for this purpose. Other tools include OBO-Edit [13], OntoStudio [14] and the ontology editor provided by the GATE toolkit [8].

As mentioned earlier in this chapter, *precision*, *recall* and *F1-measure* are the most widely used performance metrics in information extraction. However, as pointed out by Maynard *et al.* [77], using these performance measures directly in ontology-based information extraction is problematic to a certain extent because these metrics are *binary* in nature. They expect each answer to be categorized as correct or incorrect. Ideally, OBIE systems should be evaluated in a *scalar* manner, allowing different degrees of correctness [77]. Such metrics are especially useful in evaluating the accuracy in identifying instances of classes from text; the score can be based on the closeness of the assigned class to the correct class in the taxonomy of the ontology. A similar approach can be adopted with respect to property values as well.

For the task of identifying instances of an ontology Cimiano *et al.* have used a performance measure named "Learning Accuracy (LA)" [45]. They have adopted this metric from a work by Hahn *et al.* [57]. This measures the closeness of the assigned class label to the correct class label based on the subsumption hierarchy of the ontology. It gives a number between 0 and 1 where 1 indicates that all assignments are correct. Learning accuracy is computed as follows.

For each candidate pair (i, c) of the output, where i is an instance and c is the assigned class label, there is a pair $(i, gold(i))$ in the gold standard and $c, gold(i) \in O$, where O is the set of classes in the ontology used.

The least common superconcept (lcs) between two classes a and b is defined by:

$$lcs(a, b) = \arg\min_{c \in O}(\delta(a, c) + \delta(b, c) + \delta(top, c))$$

where $\delta(a, b)$ is the number of edges on the shortest path between a and b and top is the root class. Now, the taxonomy similarity T_{sim} between two classes x and y is defined as:

$$T_{sim}(x, y) = \frac{\delta(top, z) + 1}{\delta(top, z) + \delta(x, z) + \delta(y, z) + 1}$$

where $z = lcs(x, y)$.

Then learning accuracy for a set of instance - class label pairs, X is defined as follows.

$$LA(X) = \frac{1}{\mid X \mid} \sum_{(i,c) \in X} T_{sim}(c, gold(i))$$

Maynard *et al.* have defined two metrics called Augmented Precision (AP) and Augmented Recall (AR) that can be used for ontology-based information extraction systems [77]. Based on the results of some experiments, they have concluded that these measure are at least as effective as Learning Accuracy (LA). Augmented Recall in particular should be a useful tool because Learning Accuracy is basically a measure of precision.

Although these specialized performance metrics have been designed for ontology-based information extraction systems, most authors measure the performance of their systems using the classical measures of precision, recall and F1. This makes it easier to compare the performance of systems with other information extraction systems. The specialized measures described above are sometimes used in combination with the classical measures.

It should also be noted that both classical and specialized performance measures are used only for the ontology population task (extracting individuals and property values). Evaluating the results of ontology construction (identifying classes, taxonomy and properties) is inevitably subjective because it is very difficult to come up with a gold standard.

2.3. The Semantic Web

The Semantic Web is one major application area for ontology-based information extraction since OBIE systems can generate semantic contents that can be used by software agents of the Semantic Web, as described earlier in this chapter. In this section, we provide an overview of the various technologies related to the Semantic Web.

Since the vision of the Semantic Web was laid out by Berners-Lee *et al.* in 2001 [39], there have been a lot of work on developing software agents that use logic to perform complex tasks as envisioned as well as on representing information in a format that can be used by such agents. The rise of interest on ontologies as a mechanism to formally specify conceptualizations is also related to these studies. As a result of the research work in this area, the Web Ontology Language (OWL) [16], which is an extension of the Resource Description Framework (RDF)

language [22], has emerged as the de-facto standard to specify ontologies while SPARQL [25] is increasingly gaining acceptance as an ontology query language. In addition, the efforts to produce contents that can be used by software agents of the Semantic Web are also beginning to show some progress. For instance, the Linked Open Data (LOD) project, which is a collaborative effort by different research groups and some business organizations, currently provides billions of RDF triples related to hundreds of ontologies [40]. Taken together, these works indicate that the Semantic Web is slowly becoming a reality. In the remainder of this section provides brief introductions on these topics.

The Resource Description Framework (RDF) [22] provides a mechanism for representing information about resources in the World Wide Web. Initially it was intended to represent metadata about resources in the Web such as author and modification date of a web page but its application has been generalized to include conceptualizations that can be modeled or implemented using web resources. In order to apply this generalization, concepts are often represented by Uniform Resource Identifiers (URI). Through this mechanism, RDF can be used to represent almost any conceptualization or an ontology.

The RDF model describes web resources in statements consisting of a *subject*, a *predicate* and an *object*. Such a statement is known as *triple*. The subject and the predicate of a triple are represented by URIs. The object can be a URI or a string literal. For instance, the following RDF triple represents that Tim Berners-Lee is a member of MIT Decentralized Information Group (as identified by the URIs used) [40].

```
Subject: http://dig.csail.mit.edu/data#DIG
Predicate: http://xmlns.com/foaf/0.1/member
Object: http://www.w3.org/People/Berners-Lee/card#i
```

The abstract definition of RDF triples are not tied to any particular representation but a syntax known as RDF/XML is generally used to store or *serialize* RDF triples as XML documents [21]. In addition, formats known as Notation-3 (N3), Turtle and N-Triples can be used for this purpose.

The Web Ontology Language (OWL) [16] introduces additional constructs and constraints on top of the RDF model to effectively represent ontologies. It is a W3C standard and was published after years of research work aimed at developing an effective ontology representation language. It adopts some concepts from RDF-Schema [20] and DAML+OIL [30] languages, which were earlier attempts to develop an ontology representation language. OWL is actually a family of languages consisting of three languages, namely OWL-Lite, OWL-DL and OWL-Full, which are sometimes known as "species" of OWL. Moving from OWL-Lite through OWL-DL to OWL-Full results in more expressive power and increased difficulty in reasoning. OWL-DL, which is the most widely used species of OWL languages, is based on description logic and guarantees decidable reasoning [16].

The W3C OWL specification [16] provides formal definitions of different components of an OWL ontology. These correspond to the ontology components that we informally described earlier in classifying OBIE systems based on the ontology components that they extract. Namely, these components are classes, properties (with datatype properties and object properties as subtypes), individuals and property values. In addition, an OWL ontology could contain additional constraints or axioms. For instance, in the ontology for countries that we have

described earlier in this chapter, an axiom might state that every country should have exactly one capital. Axioms are typically not extracted by ontology-based information extraction systems.

OWL ontologies can be represented using various formats including the above mentioned representations for RDF triples. RDF/XML is the most widely used format since it is an official W3C recommendation [21]. Once serialized using one of these formats, OWL ontologies can be manipulated using ontology editing tools such as Protégé [19] as mentioned earlier.

The power of the Semantic Web comes from querying and reasoning on ontologies. This allows the agents of the Semantic Web to perform sophisticated tasks for users as envisioned by Berners-Lee *et al.* [39]. Over the years several ontology query languages have been designed and SPARQL [25] is the most widely used language among these. SPARQL operates on RDF triples and is a W3C recommendation. A SPARQL query consists of triple patterns, conjunctions, disjunctions, and optional patterns and returns the RDF triples matching the specified pattern. Multiple implementations exist for the SPARQL language.
In addition to ontology query languages, several advanced tools that perform reasoning on OWL ontologies have been developed. Such tools include Pellet [17], SNePS [41] and FACT++ [95].

As mentioned earlier, semantic contents should be available in ontologies for the agents of the Semantic Web to operate on. There has been some progress in creating such semantic contents as well. One area where there has been a visible progress on this regards is in encoding scientific knowledge in ontologies. This exercise has been particularly successful in bioinformatics where ontologies containing very large number of terms such as Gene Ontology [82] and Phenotypic

Quality Ontology (PATO) [18] are widely used. In addition, there has been some progress in converting existing data sets of non-profit and commercial organizations, distributable under open licenses, into semantic contents. Such data are typically taken from relational databases. The Linking Open Data (LOD) project, which is a grass root community effort supported by W3C has led the way on this regard [40]. It provides best practices for publishing data as RDF triples and makes it easier for organizations to publish their data. Several organizations such as BBC, Thomson Reuters, US Census Bureau and the Library of Congress have published some of their data through LOD [40].

The scarcity of suitable, high quality ontologies is also a concern in realizing the Semantic Web [63]. Other than domain ontolgies developed for scientific domains (e.g., Gene Ontology), many of the other domain ontologies do not provide a detailed description of the domain in concern. Some of these ontologies are categorized as *toy ontologies* because they contain very few concepts. Ontology-based information is useful on this regards as well since it can be used to evaluate the quality of of ontologies, as pointed out by Kietz *et al.* [66] and Maynard *et al.* [76]. If a given domain ontology can be successfully used by an OBIE system to extract the semantic contents from a set of documents related to that domain, it can be deduced that the ontology is a good representation of the domain. Moreover, the weaknesses of the ontology can be identified by analyzing the types of semantic content it has failed to extract. In addition, ontology construction systems can be used to generate ontologies from text. These applications of ontology-based information extraction replace or complement the efforts of domain experts and ontology engineers in designing high quality domain ontologies.

Despite theses developments, realizing the Semantic Web is still an ongoing effort. Significant improvements are necessary in several areas including creating ontologies, reasoning on ontologies, querying ontologies and creating semantic contents for the vision of the Semantic Web laid out Berners-Lee *et al.* back in 2001 to fully realize.

2.4. Summary

The research work presented in this dissertation falls within the field of ontology-based information extraction (OBIE), which is a sub-field of information extraction (IE). Information extraction has existed as a research field for several decades and several information extraction techniques have been designed over the years. Among these, extraction rules and machine learning are the dominant techniques. Research work in information extraction has produced advanced systems but widespread usage and commercialization remain difficult goals for this field. Ontology-based information extraction has recently emerged as a subfield of information extraction, where ontologies are used to guide the information extraction process. Although the implementation details of OBIE systems are different from each other a common architecture for them can be identified from a higher level. Differences among OBIE systems are mainly in the dimensions of information extraction techniques used, ontology construction and update, ontology components extracted and types of sources. One of the most important potentials of ontology-based information extraction is its ability to generate semantic contents for the Semantic Web from natural language text. Hence, the Semantic Web can be seen as one major application area for ontology-based information extraction. Research work on the Semantic Web have been steadily progressing in the last

decade in terms of representing knowledge in ontologies, querying and reasoning on ontologies and converting existing knowledge into ontologies. However, a lot of work is yet to be done and ontology-based information extraction can be expected to make a significant contribution on this regard.

CHAPTER III

OBCIE

The development of a component-based approach for information extraction is one of the two new directions for ontology-based information extraction presented in this dissertation, as mentioned in Chapter I. In this chapter, we discuss the intuition and theory behind this component-based approach. The details of its implementation and case studies conducted on it are presented in the next chapter. We identify this approach by the name "Ontology-Based Components for Information Extraction (OBCIE)" to highlight the fact that the use of ontologies is critical in it.

The first section of this chapter describes why it makes sense to use a component-based approach for information extraction. In the next section, we present a high-level overview of the OBCIE approach describing its key concepts. Then we move onto the formal specifications of the OBCIE approach and the details on its operation. Next, we present a discussion on the relationship between the OBCIE approach and some other studies closely related to it. We conclude this chapter with a summary of the intuition and theory of the OBCIE approach.

3.1. Reasons for a Component-Based Approach in IE

In Chapter II, it was mentioned that widespread usage and commercialization remain elusive goals for information extraction despite its success as a research field. This was pointed out as a serious challenge faced by the field. Our main reason for advocating the use of a component-based approach for information

extraction is its potential to make a significant contribution towards overcoming this challenge.

We identified the costs and complexity associated with setting up an information extraction system in a new domain as a major factor hindering the development of widely used information extraction systems in Chapter II. It was seen that this affects the development of large scale information extraction systems that operate in many domains and many text corpora as well. If *software components* are available to perform information extraction for certain concepts or certain tasks associated with information extraction, it can be seen that the task of setting up an information extraction system becomes much easier. In an ideal situation, the setting up of the information extraction system would be similar to constructing a complex machine using individual components as routinely done in fields such as mechanical engineering. This is also the objective of the field of component-based software engineering, where the software development process is expected to be carried out using software components with clear interfaces and clear functionalities. Since information extraction system can be considered software systems, the development of component-based approaches for information extraction can be directly related to component-based software engineering.

However, since information extraction systems face some unique circumstances different from typical software systems, the development of a component-based approach for information extraction cannot simply be achieved by directly applying the techniques developed in component-based software engineering. For instance, information extraction systems are not expected to be 100% accurate in its task (some incorrect extractions are tolerated) while typical software systems are expected to be completely accurate. This shows that new

approaches have to be adopted in developing a component-based approach for information extraction.

In Chapter II, we also identified two major factors giving rise to the costs and complexity associated with setting up an information extraction system. These factors are the following.

1. The requirement of *templates*: Information extraction systems normally require a *set of templates* for a particular domain and merely attempt to fill the templates. Generating these *templates* manually beforehand is not practical in at least some situations.

2. Bundling domain and corpus specific information with the information extraction techniques: Information extraction systems are often built as monolithic systems where no clear separation is made between domain and corpus specific information used by the implementation and the underlying information extraction technique. This makes the application of the information extraction system in a new situation difficult.

A successful component-based approach for information extraction should effectively handle these two factors. We describe how the OBCIE approach achieves this objective in the next section.

It can be seen that a component-based approach for information extraction attempts to perform the difficult task of information extraction by decomposing it into smaller parts or components. As we discuss in the following sections, the OBCIE approach mainly divides an information extraction system along ontological concepts. This approach has been adopted in the field of image retrieval, which aims to identify images relevant to a user query from a large set of images,

giving rise to a subfield known an *ontology-based image retrieval* [97]. These ontology-based image retrieval systems have shown strong results. Since image retrieval is an area of study falling under artificial intelligence just like information extraction, this provides additional support for the development of component-based approaches for information extraction. Moreover, this shows that identifying components based on ontologies, as carried out by the OBCIE approach, can be successful in problems in the field of artificial intelligence.

3.2. An Overview

It Chapter I, it was mentioned that designing independently deployable components that make extractions with respect to specific classes or properties of an ontology and separating domain, corpus and concept specific information from the underlying information extraction techniques are two key concepts in the OBCIE approach. In addition to these, identifying different operations and sub-operations of information extraction and using different types of components for these operations as well as using the ontology construction process to discover concepts that fit into a coherent conceptual framework play key roles in the OBCIE approach. All four of these concepts are described in detail in separate subsections below.

The OBCIE approach operates by decomposing an information extraction system along ontological concepts. In addition, the output of a system that has been developed under the OBCIE approach is presented in ontologies. Therefore, it can be seen that these systems fall within Definition 2.5 provided in Chapter II for ontology-based information extraction systems: the information extraction process in these systems are guided by ontologies and the output is presented through

ontologies. As such, system developed under the OBCIE approach are ontology-based information extraction systems.

Information Extractors

The concept of an information extractor is central to the OBCIE approach. As described in Chapter I, an information extractor makes extractions with respect to a specific class or a property of an ontology. In other words, it identifies individuals for a class or property values for a property. For instance, in an ontology-based information extraction system for a geopolitical ontology, information extractors might exist to identify countries (individuals for the "Country" class) or capitals of countries (property values for the "has Capital" property). We formally define an information extractor as follows.

Definition 3.1 (Information Extractor)**.** An information extractor is a component of an information extraction system that makes extractions with respect to a particular class or a property of an ontology. It has clear interfaces and is independently deployable.

When an information extraction system consists of information extractors, it can effectively be decomposed along ontological concepts (classes and properties). Individual components, which can be independently deployed, can be identified for individual classes and properties. These components can be *reused* in making extractions with respect to either the same concept or a concept that has a mapping with the original concept in a different corpus. The two ontologies might even be for different domains although there should be some overlap between the two domains for the mappings to exist. This is one of the key ideas of the OBCIE

approach. Different mechanisms can be adopted in reusing information extractors as described in section 3.4.

Platforms for Information Extraction

As described in Chapter I, another key idea of the OBCIE approach is separating domain, corpus and concept specific information from the underlying information extraction techniques. This is achieved by separating the contents of an information extractor into a *platform of information extraction* and *metadata of the information extractor*. In other words, an information extractor consists of a platform for information extraction and some metadata. They can be defined as follows.

Definition 3.2 (Platform for Information Extraction)**.** A platform for information extraction is a domain, concept and corpus independent implementation of an information extraction technique.

Definition 3.3 (Metadata of an information extractor)**.** Metadata of an information extractor are any domain, concept or corpus specific information contained in an information extractor.

The types of metadata depend on the information extraction technique. In the extraction rules technique, these will be the particular rules used for a particular concept whereas under the classification technique these will be the features used for classification. Irrespective of the information extraction technique, these amount to valuable information in information extraction. As described in Chapter II, features used for classification are the results of the *feature selection* process whereas extraction rules generally represent the results of several hours of

manual work. Metadata of information extractors provide a standard mechanism to represent these.

The separation of metdata and platforms for information extraction provides a structured mechanism to reuse information extractors. In addition, this allows making changes in the metadata when reusing an information extractor based on a mapping between two concepts of different ontologies. This amounts to *customizing* a component and is very difficult to achieve otherwise. Moreover, the separation of metadata and platforms handles the problem of bundling domain and corpus specific information with information extraction techniques described in section 3.1.. Because this issue is handled in a structured manner, OBCIE approach can make use of information extraction techniques that make use of domain, concept and corpus specific information unlike the systems that follow the paradigm of open information extraction, which are restricted to techniques that do not use any such information.

Operations in Information Extraction

As described in Chapter II, ontology construction and ontology population are the two main operations in an ontology-based information extraction system. It can be seen that information extractors, consisting of platforms for information extraction and metadata, are related to the ontology population operation. However, the ontology population operation is not completed by information extractors. Clearly, the output of individual information extractors have to be combined and the ontology has to be updated to include the new individuals and property values. It might be necessary to perform tasks such as reference reconciliation, which refers to the problem of determining whether two individuals

refer to the same real world entity [50], when performing this task. This can be considered a sub-operation of ontology population. Under the OBCIE approach, this operation is performed by a separate type of components called "aggregators". In addition, two other types of components called "preprocessors" and "formatters" can be used in ontology population. Preprocessors, as their name suggests, do some processing in the source text to prepare them to be used by information extractors. Formatters make some adjustments in the outputs produced by information extractors before they are handed over to aggregators. For instance, in identifying gene names it might be necessary to ensure that all identified gene names exactly represent standard gene names.

In summary, the OBCIE approach divides the ontology population operation into four sub-operations, namely preprocessing, making extractions with respect to specific ontological concepts, formatting the extractions and aggregating. Different types of components are used for these operations namely preprocessors, information extractors, formatters and aggregators. The ontology construction operation is not divided into sub-operations in this manner and it is expected to be conducted as a single operation. More details on this is presented in the next subsection.

Ontology Construction

As described in Chapter II, ontology construction is concerned with identifying classes and properties of ontologies. This includes identifying a class hierarchy or a taxonomy as well as identifying non-taxonomical relationships. According to the terminology used by OWL [16], ontology construction operation identifies classes, datatype properties as well as object properties. It is possible

for an ontology construction process to identify other types of axioms such as constraints but this normally does not take place. Practically, ontology construction is normally carried out by starting with few high-level concepts, which are sometimes called *seed concepts* [65], and discovering more concepts based on them because constructing an ontology from scratch is very difficult.

It can be seen that ontology construction operation addresses the problem of templates for information extraction described in section 3.1. It was stated that information extraction systems often require *templates* and just attempt to fill values for these templates. This is seen as a factor hindering the widespread usage of information extraction systems. It was also mentioned that open information extraction handles this problem by discovering relations of interest from text itself. In ontology-based information extraction, the templates are *classes* and *properties* of the ontologies. Values fitting into these templates are discovered through the ontology population process. Hence, it can be seen that the ontology construction process generates templates for the information extraction system. It can also be seen that the ontology construction process conducted in this manner ensures that the discovered concepts fit into a coherent conceptualization. This is often not the case in the relation discovery process carried out by open information extraction, which just discovers all types of relations present in the corpus.

3.3. The Formal Specification

This section provides a formal specification of the OBCIE approach. It is primarily based on the Z notation [90], which is a widely used formal specification language. First, a specification is provided for a generic ontology-based information extraction system. Then we show how this specification can be *refined* into a

specification for an ontology-based information extraction system that follows the OBCIE approach. This shows that the OBCIE approach is functionally correct.

The specifications are also provided in the Alloy specification language [1], which is viewed as a subset of the Z notation, and which has better tool support for consistency checking. These Alloy specifications are used to check the consistency of the refinement of the specification of a generic OBIE system into a one following the OBCIE approach. This provide additional support for the functional correctness of the OBCIE approach.

The Z Specification

In order to provide a specification for a generic OBIE system, we need a formal specification of an ontology. We have conceived the following specification for this purpose, which identifies different types of components in an ontology and provides formal specifications for each of these types.

Definition 3.4 (Ontology)**.** An ontology O is a quintuple, $O = (C, P, I, V, A)$ where C, P, I, V, and A are the sets of classes, properties, individuals, property values and other axioms respectively. These sets are defined as follows.

$C = \{c \mid c \textit{ is a unary predicate}\}$

$P = \{p \mid p \textit{ is a binary predicate}\}$

$I = \{c(i) \mid c \in C \ \wedge \ i \textit{ is a ground term}\}$

$V = \{p(x, y) \mid p \in P \ \wedge \ x \textit{ and } y \textit{ are ground terms} \ \wedge \ (\exists c \ c \in C \wedge c(x))\}$

$A = \{a \mid a \textit{ is an assertion}\}$

It should be noted that the above definition does not contradict the definition for an ontology provided by Studer *et al.* [91], discussed in Chapter II (Definition 2.4). We do not dispute that ontologies provide formal and explicit

specifications of shared conceptualizations. Rather, we formalize the components that can be included in such a specification. In this sense, our definition is also consistent with the OWL specification [16], because the five sets listed above cover all the components of an OWL ontology.

We now provide a Z specification for an ontology-based information extraction system. This is done by providing formal specifications for the two operations of such a system, named *Populate* and *Construct*, which correspond to ontology population and ontology construction respectively.

[*Document*, *UnaryPredicate*, *BinaryPredicate*, *AssertionsOnUnaryPredicate*, *AssertionsOnBinaryPredicate*, *Assertion*]

Corpus

$$documents : \mathbb{P}\, Document$$

$$\#documents > 1$$

$$Response \triangleq success \mid empty\_ontology$$

Ontology

$$classes : \mathbb{P}\, UnaryPredicate$$
$$properties : \mathbb{P}\, BinaryPredicate$$
$$individuals : \mathbb{P}\, AssertionsOnUnaryPredicate$$
$$values : \mathbb{P}\, AssertionsOnBinaryPredicate$$
$$axioms : \mathbb{P}\, Assertion$$

InitOntology

$classes'$
$properties'$
$individuals'$
$values'$
$axioms'$

InitCorpus

$documents'$

$\#documents > 1$

PopulateOk

$\Delta Ontology$
$\Xi Corpus$
$r! : Response$
$i! : \mathbb{P}\, AssertionsOnUnaryPredicate$
$v! : \mathbb{P}\, AssertionsOnBinaryPredicate$

$(\#classes > 0) \lor (\#properties > 0)$
$classes' = classes$
$properties' = properties$
$individuals' = individuals \cup i!$
$values' = values \cup v!$
$axioms' = axioms$
$r! = success$

PopulateEmpty

$\Xi Ontology$
$\Xi Corpus$
$r! : Response$

$(\#classes = 0) \land (\#properties = 0)$
$r! = empty\_ontology$

$Populate \triangleq PopulateOk \lor PopulateEmpty$

$$
\begin{array}{|l}
\textit{Construct} \\
\hline
\Delta Ontology \\
\Xi Corpus \\
r! : Response \\
c! : \mathbb{P}\ UnaryPredicate \\
p! : \mathbb{P}\ BinaryPredicate \\
a! : \mathbb{P}\ Assertion \\
\hline
classes' = classes \cup c! \\
properties' = properties \cup p! \\
individuals' = individuals \\
values' = values \\
axioms' = axioms \cup a! \\
r! = success \\
(\#classes' > \#classes) \lor \\
(\#properties' > \#properties) \\
\hline
\end{array}
$$

Now we move onto refining this specification into a specification of an ontology-based information extraction system that follows the OBCIE approach. Under OBCIE, we keep the ontology construction operation unchanged because it is expected to be performed using a single component. However, we expect to use a series of components to perform the ontology population operation. This can be achieved by refining the *PopulateOk* operation into a *pipeline* of four separate operations as follows.

$PopulateOk \triangleq Preprocess >> Extract >> Format >> Aggregate$

These four operations correspond to different types of components namely *preprocessors*, *information extractors*, *formatters* and *aggregators*. As described in section 3.2., preprocessors change the format of the documents into a format that can be used by information extractors; information extractors make extractions with respect to specific classes or properties as mentioned earlier; formatters make some adjustments on the extractions made by information extractors to make them

more accurate; aggregators combine the output produced by different information extractors and produce ontologies, generally in the OWL format, as the final output of the system.

Z specifications can be provided for these individual operations as follows.

$[PreprocessedDocument]$

Preprocess

$$\Xi Ontology$$
$$\Xi Corpus$$
$$intermediate! : \mathbb{P}\ PreprocessedDocument$$

$$(\#classes > 0) \lor (\#properties > 0)$$
$$\#intermediate! = \#documents$$

Extract

$$\Xi Ontology$$
$$\Xi Corpus$$
$$intermediate? : \mathbb{P}\ PreprocessedDocument$$
$$ex\_inds! : \mathbb{P}\ AssertionsOnUnaryPredicate$$
$$ex\_values! : \mathbb{P}\ AssertionsOnBinaryPredicate$$

$$(\#classes > 0) \lor (\#properties > 0)$$
$$\#intermediate? = \#documents$$

The *Extract* operation is further refined into a set of operations that make extractions with respect to specific classes and properties as follows. It is assumed that extractions are made for m number of concepts (classes and properties).

$$Extract \triangleq ExtractInd >> CombineExt$$

$$ExtractInd \triangleq ExtractInd_1 \land ExtractInd_2 \land ... \land ExtractInd_m$$

Each operation $ExtractInd_i (1 \leqslant i \leqslant m)$ is defined as follows.

$ExtractInd_i$

$\Xi Ontology$
$\Xi Corpus$
$intermediate? : \mathbb{P}\ PreprocessedDocument$
$ex\_inds\_i! : \mathbb{P}\ AssertionsOnUnaryPredicate$
$ex\_values\_i! : \mathbb{P}\ AssertionsOnBinaryPredicate$

$(\#classes > 0) \lor (\#properties > 0)$
$\#intermediate? = \#documents$
$(\#ex\_inds\_i! = 0) \lor (\#ex\_values\_i = 0)$

CombineExt Operation is defined as follows.

$CombineExt$

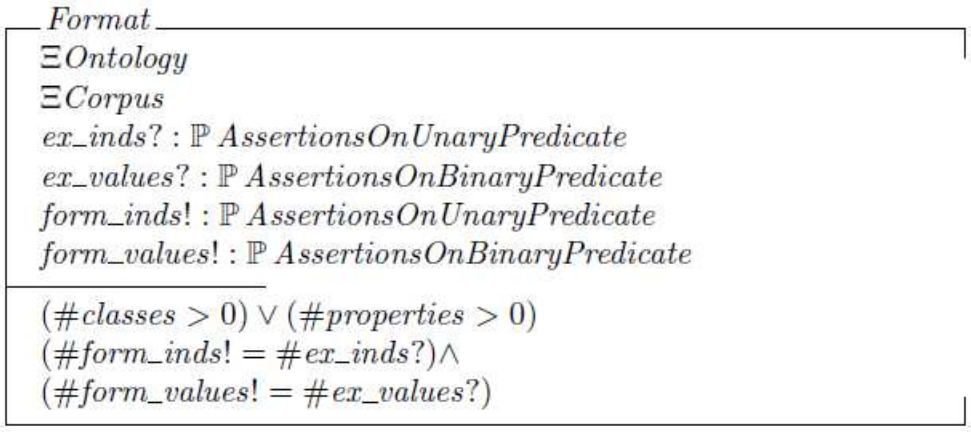
$\Xi Ontology$
$\Xi Corpus$
$ex\_inds\_1? : \mathbb{P}\ AssertionsOnUnaryPredicate$
$ex\_values\_1? : \mathbb{P}\ AssertionsOnBinaryPredicate$
$ex\_inds\_2? : \mathbb{P}\ AssertionsOnUnaryPredicate$
$ex\_values\_2? : \mathbb{P}\ AssertionsOnBinaryPredicate$
.
.
.
$ex\_inds\_m? : \mathbb{P}\ AssertionsOnUnaryPredicate$
$ex\_values\_m? : \mathbb{P}\ AssertionsOnBinaryPredicate$
$ex\_inds! : \mathbb{P}\ AssertionsOnUnaryPredicate$
$ex\_values! : \mathbb{P}\ AssertionsOnBinaryPredicate$

$ex\_inds! = \bigcup_{i=1}^{m} ex\_inds\_i?$
$ex\_values! = \bigcup_{i=1}^{m} ex\_values\_i?$

Each $ExtractInd_i$ corresponds to an individual information extractor. The *CombineExt* operation is used for the correctness of the specification. There is no component related to this operation because it is not necessary to combine

the output of individual information extractors before reaching the aggregator operation.

The *Format* operation is defined as follows.



The *Format* operation is also further refined into a set of operations. These individual operations format the results related to specific classes and properties. They use the same m concepts (classes and properties) used by individual information extractor operations.

$$Format \triangleq FormatInd >> CombineFmt$$

$$FormatInd \triangleq FormatInd_1 \wedge FormatInd_2 \wedge ... \wedge FormatInd_m$$

Specifications of individual $FormatInd_i$ operations and the $CombineFmt$ operation are analogous to the specifications of individual $ExtractInd_i$ and $CombineExt$ operations respectively.

The *Aggregate* operation is defined as follows.

Aggregate

$\Delta Ontology$
$\Xi Corpus$
$form\_inds? : \mathbb{P}\ AssertionsOnUnaryPredicate$
$form\_values? : \mathbb{P}\ AssertionsOnBinaryPredicate$
$r! : Response$
$i! : \mathbb{P}\ AssertionsOnUnaryPredicate$
$v! : \mathbb{P}\ AssertionsOnBinaryPredicate$

$(\#classes > 0) \lor (\#properties > 0)$
$classes' = classes$
$properties' = properties$
$individuals' = individuals \cup i!$
$values' = values \cup v!$
$axioms' = axioms$
$r! = success$

It should be noted that the Z specifications presented in this section are *non-deterministic* in that they do not specify what extractions are made by the system. However, these specifications capture the essential functionality of ontology-based information extraction systems and show that the OBCIE approach preserves this functionality.

The Alloy Specification

The main reason for coming up with an Alloy specification in addition to the Z specification is checking the consistency of the refinement of the specification from a generic ontology-based information extraction system into a system under the OBCIE approach, as mentioned earlier. This can be achieved by representing the refinement as an *assertion* in Alloy and checking the consistency of this assertion using a tool named *Alloy Analyzer* developed for the Alloy language [1].

Alloy analyzer does not use a theorem prover to verify that an assertion or a model is consistent. Instead it searches for counter-examples, which show that the model is invalid within a finite search space. Since counter-examples are found for most inconsistent models within a small search space, absence of any counter-examples within a finite search space is normally good enough for a model to be considered consistent.

The correctness of the refinement of the *PopulateOk* operation into a pipeline consisting of four different operations was verified using the process described above. The Alloy specifications used in this exercise are shown in Appendix A.

3.4. The Operation of the Approach

In this section, we discuss how the OBCIE approach operates. First, we show a schematic diagram that represent the composition of ontology-based information extraction systems under the OBCIE approach. Then we describe the architecture designed for the reuse of information extractors. This includes the different reuse mechanisms that can be used for information extractors.

System Composition

A schematic diagram that represents ontology-based information extraction systems that follow the OBCIE approach can be drawn based on the formal specifications presented in the previous section. Figures 2 presents this diagram.

Ontology-based information extraction systems under the OBCIE approach consist of different types of components as shown by this diagram. We focus our attention on the information extractors because we believe that they contain the

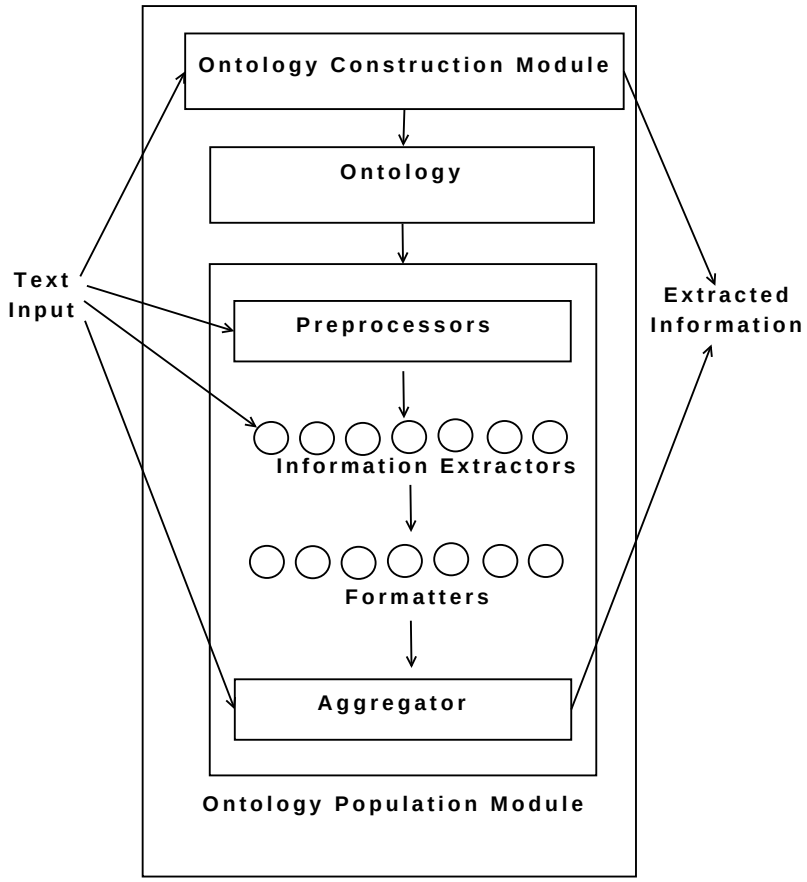


FIGURE 2: A single-ontology OBIE system under the OBCIE approach

most valuable information captured by information extraction systems. But other components also perform very important tasks in an information extraction system.

Architecture for Reuse

For our component-based approach to work, there should be standard mechanism to store the different types of information associated with information extractors and to represent the links between them. We have designed a three-layered architecture, consisting of ontologies, metadata for information extractors and platforms for this purpose. Figure 3 represents this architecture.

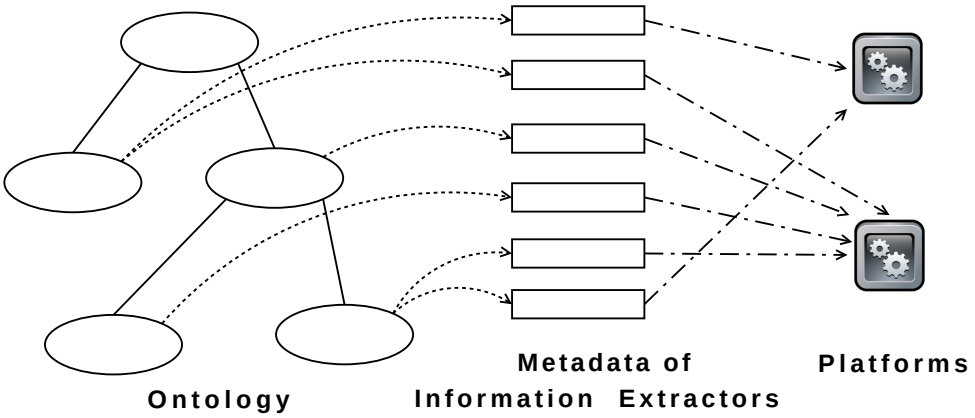


FIGURE 3: The architecture for reuse of information extractors

The ontology represents the domain ontology for which an information extraction system has been developed. Since the Web Ontology Language (OWL) [16] is increasingly being seen as the standard for defining ontologies, ontologies are represented using it. The metadata of information extractors and the details of platforms are stored in *XML files* and made available through *URIs*. The links between classes and properties of ontologies and the information extractors

that have been developed for them are represented as *OWL annotation properties* in the ontologies. An XML file for the metadata of an information extractor contains *an element* that represents the URI of the platform that it uses. The XML file for the platform contains a link to the executable file for the platform.

OWL annotation properties provide a mechanism to include the links between ontological concepts and information extractors in ontologies while excluding these links from reasoning. These are typically used to include metadata about concepts such as version, creator and comments and are ignored by software agents that perform reasoning on the ontologies. The information extractors developed for classes and properties of ontologies can be considered a type of such metadata. Moreover, OWL annotation properties can be used with both classes and properties and can be specified as URIs. These properties match nicely with our requirement to specify the URIs of information extractors for both classes and properties. OWL annotation properties have also been used by Yildiz and Miksch [103] to include extraction rules in ontologies. Their work can be seen as a previous attempt to develop a component-based approach for information extraction and we discuss it together with some other work that follow the same approach later in section 3.5. The OBCIE approach stores URIs of information extractors instead of extraction rules in OWL annotation properties but is driven by similar objectives as in the work by Yildiz and Miksch [103].

XML was selected as the format for representing metadata of information extractors and platforms because it is a lightweight machine processable format that is widely used for information exchange. Using ontologies for this purpose was considered but it was quite clear that advanced features of ontologies are not

necessary here. XML schemata for creating XML files for platforms and metadata have been created. There are shown in Appendices B.

For the OBCIE approach to operate, mappings between concepts of different ontologies have to be identified since the reuse of information extractors is based on these mappings. This is not a trivial task and there have been several works on discovering mappings between ontologies in an automatic or semi-automatic manner [34, 49, 56, 60, 64]. In some cases it is possible to identify mappings manually, but automatic mapping discovery techniques are required to make the OBCIE approach scalable.

Only certain types of mappings are useful in reusing information extractors. There can be complex mappings between ontologies, often involving more than two concepts and such mappings cannot be used for the purpose of reusing information extractors. For instance, a mapping which states that a person's full name under one ontology consists of a first name and a last name of another ontology, does not lead to a reuse of information extractors.

We use the term "direct mapping" to identify mappings that are useful in reusing information extractors. They can be defined as follows.

Definition 3.5 (Direct Mapping)**.** A direct mapping $M(X_{a,i}, X_{b,j})$ exists between two concepts $X_{a,i}$ and $X_{b,j}$ of two different ontologies O_a and O_b ($X_{a,i} \in C_a \cup P_a$ and $X_{b,j} \in C_b \cup P_b$ with usual definitions for O_a and O_b), if and only if, $val(X_{a,i}) \equiv val(X_{b,j})$ or $val(X_{a,i}) \subset val(X_{b,j})$ or $val(X_{a,i}) \supset val(X_{b,j})$, where $val(X_{a,i})$ and $val(X_{b,j})$ represent the sets of individuals/property values of $X_{a,i}$ and $X_{b,j}$ respectively.

Once direct mappings between ontological concepts are discovered, reuse of information extractors can be performed in different ways as shown below.

1. *Black Box Reuse:* This refers to directly reusing an existing information extractor for a particular class or a property in a new system.

2. *Clear Box Reuse:* Here, the metadata of the information extractor, which contain domain and corpus specific information, are changed before applying it in a new corpus or a new domain. The platform is not changed.

3. *Simple Combination:* Here, more than one information extractor is used for the same concept. The results produced by them are combined using set operations. The union operator is selected for this purpose since it has the potential to result in more correct extractions.

4. *Advanced Combination:* More advanced techniques, such as techniques used under ensemble learning can be used to combine different information extractors instead of set operators.

It is interesting to note that the terms *black box reuse* and *clear box reuse* are also used in component-based software engineering with similar meanings. The relationship between component-based software engineering and the OBCIE approach is discussed in detail in the following section.

3.5. Discussion

In this section, we discuss the details of some studies closely related to the OBCIE approach. First, we discuss component-based software engineering which has influenced the development of the OBCIE approach. Next, we discuss the details of UIMA (Unstructured Information Management Architecture), which is a significant previous attempt to develop a component-based approach for information extraction. Finally, we present the details of some other work which

can be seen as being related to the development of a component-based approach for information extraction.

Component-Based Software Engineering

Component-Based Software Engineering (CBSE) studies how component-based approaches can be used in developing software systems. This field has been in existence for more than a decade and has been successful in many domains. Since information extraction systems can be viewed as software systems, the field of component-based software engineering is related to the development of a component-based approach for information extraction.

Szyperski [93] provides a simple commonsense justification for the use of a component-based approach in software engineering in his textbook on this subject by stating "components are the way to go because all other engineering disciplines introduced components as they became mature and still use them." He concedes that from a purely formal view, there is nothing that could be done with components that could not be done without them. The differences are in goal-driven factors such as reusability, time to market, quality and viability. It can be seen that these arguments are more or less valid in the field of information extraction as well: a component-based approach, while not doing anything that cannot be done using existing techniques, has the potential to improve the information extraction process quantitatively and qualitatively.

In component-based software engineering, it is generally agreed that software components have clear interfaces and functionalities. It can be seen that ontology constructors, preprocessors, information extractors, formatters and aggregators described earlier satisfy these requirements. Further, Szyperski states that software

components may contain "meta-data and resources" and even concedes that some *degenerate* components may only consist of such meta-data and resources. Hence, it can be seen that even the metadata of information extractors can be considered independent components. This implies that an information extractor is a component that consists of two sub-components, namely a platform and a metadata component.

UIMA: Unstructured Information Management Architecture

UIMA (Unstructured Information Management Architecture) [2] is a component-software architecture for analyzing unstructured information. It is not restricted to text and targets audio, video and images as well. It has started as an IBM project and later donated to Apache by IBM. Now it is conducted as an Apache open source project.

UIMA components have been mostly developed for general NLP tasks such as sentence splitting, tokenization and POS tagging but some components have been developed for information extraction tasks. For example, some UIMA components have been developed to identify gene names. Frameworks have been developed in Java in C++ to deploy UIMA components. They are expected to be distributed through *UIMA component repositories*.

A key idea of UIMA is that of a Common Analysis Structure (CAS), which contains the source being processed as well as the extractions that have been made on it. It can be kept in the memory or serialized into XML. A related concept is *Sofa*, which stands for "Subject of Analysis", which is a view on a source. The source can be a text document, image file or a video file.

Because of the significant effort that has led to the development of UIMA, its frameworks are technologically sophisticated and there are some tools that support the development of UIMA components. For instance, there is an UIMA plug-in for the popular Java IDE (Integrated Development Environment) Eclipse, which makes it easier to develop and deploy UIMA components. This technical sophistication is clearly a strength of UIMA.

However, UIMA does not appear to have been very successful as a component-based approach for information extraction. There are only a handful of repositories for UIMA components and the few existing one have been implemented by research groups to distribute their own components. As mentioned earlier, most of these components are for generic NLP tasks and not for information extraction.

The OBCIE approach differs from UIMA in the following ways. Because of these differences, it can be seen as being fundamentally different from UIMA although both advocate the use of components.

1. OBCIE separates the domain, corpus and concept specific information from the underlying information extraction techniques, resulting in generic platforms. UIMA does not have a comparable mechanism.

2. It can be seen that UIMA implicitly assumes that the developed components are *interoperable*. In contrast, OBCIE investigates the circumstances that lead to successful reuse through different mechanisms such as black box reuse and clear box reuse.

3. OBCIE uses ontologies to represent the concepts being extracted whereas UIMA uses "type systems" based on UML. Ontologies should be the better option because of their ability to generate semantic contents and the ability

to support reasoning. In addition, OBCIE provide links to information extractors from ontologies, whereas UIMA stores the type systems in the Common Analysis Structure (CAS).

4. UIMA requires centralized component repositories. OBCIE is based on a decentralized architecture where the components are made available through the Web and links to them are provided from the ontologies.

As a final note regarding UIMA, it can be argued that UIMA has put too much emphasis on UML and web services (UIMA components can be deployed as web services), which were very popular few years ago but have since lost some of their appeal. Ontologies and more open access mechanisms such as RDF link dereferencing are gaining popularity and have shown their potential in applications such as linked data [40]. The OBCIE approach is in a better position to make use of these techniques.

Other Related Work

Embley [52], Maedche *et al.* [72] and Yildiz and Miksch [103] have independently worked on including *extraction rules* used for extracting information related to some concepts in the ontologies themselves. An ontology that contains such rules are identified as extraction ontologies by Embley [52] and as concrete ontologies by Maedche *et al.* [72]. As mentioned in section 3.4., Yildiz and Miksch have included these extraction rules in OWL annotation properties [103].

It can be seen that including extraction rules directly in the ontology violate the requirement that ontologies be formal, as stipulated by the generally accepted definition for ontologies given by Gruber and Studer *et al.* [55, 91]. Therefore, the extraction ontologies and concrete ontologies proposed by Embley and Maedche

et al. can not be seen as consistent with the standard definition of an ontology. The approach taken by Yildiz and Miksch [103] does not directly contradict this definition because they have used OWL annotation properties, which are not used for reasoning in ontologies, to include extraction rules. However, it can be seen that this is quite cumbersome because there can be numerous extraction rules for a concept and some of them can be long, which would result in very long annotation properties.

Despite these limitations, these works can be seen as previous attempts to develop a component-based approach for information extraction. They all share the objective of using some kind of components for information extraction although the term *component* has not been used by them. One limitation of these approaches is that they are restricted to one information extraction technique, namely extraction rules. The OBCIE approach is based on similar insights as these works but accommodates different information extraction techniques instead of being restricted to just one.

3.6. Summary

OBCIE, which stands for Ontology-Based Components for Information Extraction, is a component-based approach for information extraction. One of its key ideas is the concept of information extractor, which is an independently deployable component of an information extraction system that makes extractions with respect to a particular class or a property of an ontology. An information extractor consists of a platform for information extraction, which is domain, corpus and concept independent, and some metadata. Other components used in the OBCIE approach are preprocessors, formatters, aggregators and ontology

constructors. The ontology population task is performed together by preprocessors, information extractors, formatters and aggregators. The functional correctness of the OBCIE approach can be proven using Z and Alloy specifications. The main focus of these specifications is proving that the refinement of a specification for a generic ontology-based information extraction system into a one following the OBCIE approach is consistent.

A three-layered architecture consisting of OWL ontologies and XML files for metadata and platforms have been designed for reusing information extractors. XML schema have been designed for XML files that store the details of metadata and platforms. Different mechanisms can be used for the task of reusing information extractors. These include black box reuse, clear box reuse, simple combination and advanced combination.

The OBCIE approach is directly related to the field of component-based software engineering, which studies how components can be used in software development. There have been some previous attempts to develop a component-based approach for information extraction and UIMA is the most important among them. It provides a sophisticated framework for developing components but has some key differences with the OBCIE approach.

CHAPTER IV

CASE STUDIES ON OBCIE

In this chapter, we present the implementation details related to the OBCIE (Ontology-Based Components for Information Extraction) approach. These include the details of reusable platforms and components developed under the OBCIE approach as well as the details of a case study conducted to evaluate how it operates in practical situations.

We begin this chapter by presenting the details of two platforms for information extraction, which were developed following the principles of the OBCIE approach. Then we present the details of two aggregator components that suit two different situations. Next we present the details of a case study conducted using the OBCIE approach. We conclude this chapter with a discussion on the results and general observations of the implementation work carried out.

4.1. Platforms for Information Extraction

A platform for information extraction is a domain, corpus and concept independent implementation of an information extraction technique as described in Chapter III. Developing platforms for information extraction makes the reuse of information extractors structured and straight-forward. Moreover, they make it easier to change or customize an information extractor to suit a slightly different situation than the one it was developed for.

In this section we present the details of platforms developed for two information extraction techniques, namely classification and extraction rules.

As described in Chapter II, these are the two dominant techniques used for information extraction.

Two-Phase Classification

The problem of information extraction can be converted into a set of classification problems using different techniques as described in Chapter II. The technique known as two-phase classification was selected for implementing a platform for information extraction. This technique works by converting the problem of extracting information related to a particular concept into two classification problems: identifying sentences in which the relevant information is present and identifying words within sentences that represent the information. One classifier is used at the sentence-level while another is used at the word-level. Both are binary classifiers because they have to classify each sentence or word as containing the relevant information or not. Two such classifiers have to be used for each class or property on which information is extracted. This technique has been used by the Kylin information extraction system [101].

There are two different techniques to combine the results of the two classifiers as described below.

1. *Pipelining:* The word-level classifier only operates on the words of sentence that are classified as containing the required information by the sentence-level classifier.

2. *Combination:* The word-level classifier operates on all words but uses the classification made by the sentence-level classifier as one feature for building the classification model.

Both techniques were attempted in the experiments that were conducted in developing this platform. Better results were observed in these experiments when the combination approach was used and as such it was incorporated into the platform.

Since this information extraction technique is based on classification, it requires a training set in addition to the corpus on which information extraction is to be performed, which can be considered the test set. Instead of requiring an annotated corpus indicating the positions where the particular concept is found, the platform allows providing key files specified for each file in the corpus. Then it internally annotates the text files with the keys provided based on string matching (allowing some prefixes and suffixes such as " ‘ " and " 's "). While this reduces the accuracy of the annotations, it also significantly reduces the effort required to create the training corpus. For the test set, it is not necessary to provide keys but the accuracy of the platform can be automatically measured if keys are provided.

Experiments were conducted with different classification techniques for the two classification phases and it was found that Bayesian techniques (specifically Naïve Bayes model) used with bagging produced best results in sentence-level and that Conditional Random Fields (CRF), which is a sequence tagging technique, produced best results at the word-level. Therefore, these two techniques were incorporated into the platform. However, the user has the option of selecting a different classification technique for the sentence-level classification. The Mallet system [78] was used for the CRF technique while the Weka system [58] was used for other classification techniques.

The sentence-level classifier uses several domain and corpus independent features such as the word count for each POS tag. Similarly, we use domain and

corpus independent features such as POS tags, stop words, the half of the sentence a word belong to (first or second) and capitalization information at word-level. Some of such features were adopted from the Kylin system. In addition, we use specific words, WordNet [53] synsets of some words and some gazetteers as concept-specific information. (Specific examples are provided later in this chapter.) These features represent the metadata of the information extractor and are included in the XML file for the metadata. The platform requires such an XML file with metadata as an input.

The platform was implemented using the Java programming language. It programmatically calls Weka and Mallet systems as necessary. It uses the Widows installation of Mallet and as such operates only in Windows operating systems.

Extraction Rules

As described in Chapter II, this information extraction technique operates by using regular expressions to extract occurrences of some concept. Since a set of regular expressions is used for extracting information related to a particular concept, arranging the regular expressions used based on the concepts they are used for is quite natural for this information extraction technique. Under the OBCIE approach, such regular expressions developed for a concept are considered metadata of the information extractor of that concept. Since all regular expressions, which are known as extraction rules, are specific to the concept in concern, no domain and corpus independent rules are used by the platform. This is a difference of this platform from the platform for two-phase classification which uses some domain and corpus independent features for all concepts as mentioned above.

This platform does not require a training set to make extractions on a corpus because the extraction rules can be directly applied on the target corpus. However, a training set would be required when first developing the extraction rules. As described in Chapter II, this is normally carried out manually although some tools are available to discover rules in an automatic or semi-automatic manner. Further, when the clear box reuse technique is used, a training set is required for the corpus for which information is extracted because it is necessary to adjust the rules to fit the target corpus better. As in the platform for two-phase classification, keys for the test set (corpus for which information extraction is carried out) are not required but performance can be measure automatically if keys are provided.

This platform was implemented using the General Architecture for Text Engineering (GATE) [8], which a widely used NLP toolkit that can be used to directly deploy extraction rules. Here, the rules have to written in a format known as Java Annotation Patterns Engine (JAPE), which is interpreted by GATE. As such, these JAPE rules are included in the metadata of the information extractor. In addition, the metadata also include gazetteers, which can be used by the JAPE rules. As in the case of the platform for two-phase classification, the platform requires an XML file with metadata as an input. This platform does not depend on a particular operating system and it has been successfully tested in Windows and Unix environments.

In developing these platforms, it was observed that the implementation is much neater than a regular information extraction system because only one type of extractions had to be considered instead of a set of different types. However, the platforms have to be executed separately for each concept. Parallel processing is the natural choice for improving efficiency on this regard.

4.2. Aggregator Components

As described in Chapter III, the task of an aggregator component is combining the output of different information extractors and producing OWL files as output. It was seen that a single aggragator component cannot be used for all information extraction tasks. In particular, different aggregator components have to be used in the following scenarios.

1. *Relation-based information extraction:* As described by Russel and Norvig [88], this refers to a situation where all the extractions made are related to a single class. When the relational model is used for representing the information, the extracted information are related to individual tuples of a relation. In ontology-based information extraction, this refers to a situation where information extraction is carried out to identify instances of a single class and their property values.

2. *Multi-relation information extraction:* This refers to a situation where extractions are made with respect to different classes. In the relational model, this would correspond to multiple relations. In ontology-based information extraction, extractions will be made to identify instances of multiple classes and their property values.

Two different aggregators were developed for these two scenarios.

It can be seen that the development of an aggregator component for relation-based information extraction is straight-forward. However, aggregating results in the multi-relation information extraction is a complex task since relationships between different classes have to be considered. Therefore, in developing an aggregator component for this scenario, the following condition was used: *all object*

properties should have a single main class as the domain. When this assumption is used, aggregation can be carried out based on this main class. Although all ontologies clearly do not satisfy condition, many do including the ones used in the case studies presented later in this chapter. Figure 4 schematically represents the structure of ontologies that can be used for relation-based information extraction and multi-relation information extraction with the simplifying condition used.

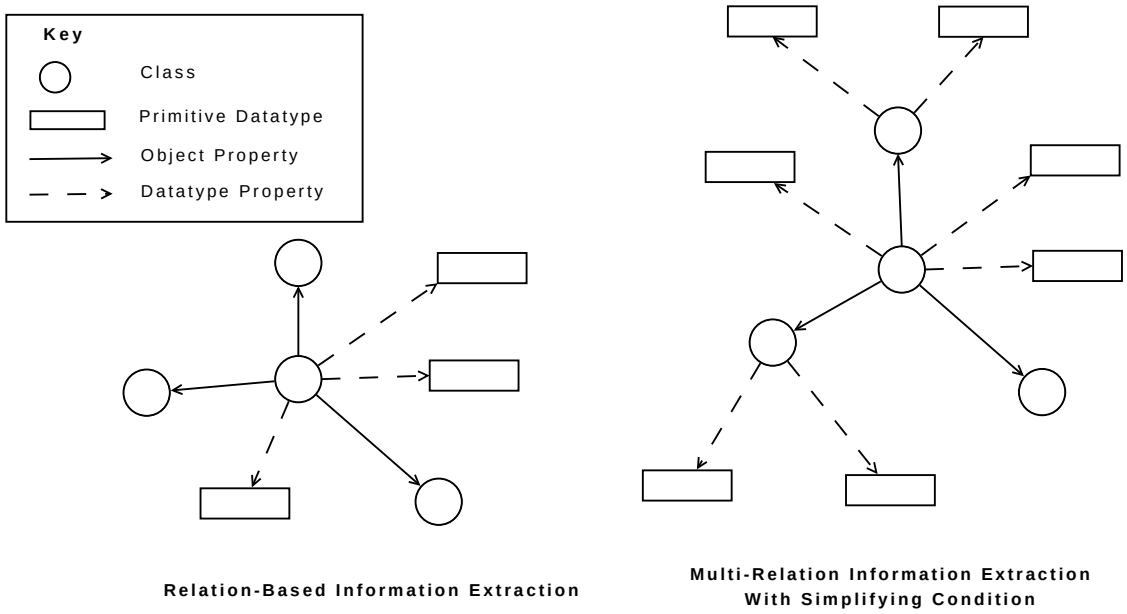


FIGURE 4: The structure of ontologies handled by aggregators

These aggregator components were implemented using the Java programming language. They take output files produced by information extractors, which contain the extractions for classes and properties as input. A popular Java API for OWL [31] was used for creating OWL files as output.

4.3. A Case Study

This section presents the details of a case study conducted to evaluate the OBCIE approach. The domain of terrorist attacks was used for the case study. The first phase of the case study was developing information extractors for a corpus in this domain. The selected corpus was a section of the corpus used by a previous Message Understanding Conference. Information extractors were developed using this corpus for two different ontologies for the domain of terrorist attacks. Then these information extractors were reused in a different corpus related to terrorist attacks for another ontology on the domain of terrorist attacks. A corpus of Wikipedia pages was used in this phase. It can be seen that the success of the OBCIE approach in this case study can be evaluated by comparing the results for the reuse of information extractors in the second corpus against the results obtained by them in the first corpus.

The first corpus was derived from the corpus used by the 4th Message Understanding Conference (MUC 4). This conference has used a set of news articles related to terrorist activities of Latin American countries as its corpus. This corps as well as the keys (gold standard) for the documents are publicly available [23]. The corpus consists of 1700 articles, 1300 in the training set and 400 in the test set. The first 200 articles of the training set were used as the corpus in this case study, 160 for the training set and 40 for the test set.

As mentioned earlier, two ontologies that provide different perspectives on the domain of terrorist attacks were used in this case study. One ontology was adopted from the structure of MUC 4 key files itself. Each key file presents the details of a particular terrorist attack and these details consist of 24 *slots*. "Incident: Location", "Incident: Stage of Execution", "Hum Tgt: Name" and "Hum Tgt:

Description" are some of such slots. It was seen that these slots can be easily converted into an OWL ontology. The clear specifications on the relationships between slots (e.g., Hum Tgt: Description may be *cross referenced* to a Hum Tgt: Name) provided by MUC 4 documentation were also helpful in this exercise. We use the term *MUC 4 ontology* to refer to the ontology constructed in this manner.

The other ontology was an ontology developed by the Mindswap group of the University of Maryland [12]. This group has conducted some research on the terrorism domain in developing this ontology [75]. Some minor changes were made in this ontology in adopting it for this case study. For example, the constraints in the original ontology which stated that start dates and end dates should be known for all "terrorist events" were removed since these were not known for some events described in the MUC 4 corpus. We identify this ontology by the term *Mindswap ontology*.

In developing information extractors for concepts of MUC4 and Mindswap ontologies, two different platforms were used with the two ontologies. The two-phase classification platform was used with the MUC4 ontology and the extraction rules platform was used with the Mindswap ontology. A set of classes and properties were selected from each ontology to use in information extraction. In order to apply the two platforms, specific words, WordNet synsets and gazetteers were identified for each concept used with the two-phase classification platform and extraction rules (in JAPE format) and gazetteers were identified for each concept used with the extraction rules platform. Techniques were also developed for identifying these information, the details of which are presented later in this section. These features were then written into XML files conforming to the XML schema for metadata. Next platforms were executed using the XML files and the

performance of the observed results were measured. As described in section 4.1., this could be done automatically once the key files for the test set were provided to the platforms.

The next step of the case study was reusing the information extractors developed for the MUC4 corpus in a different corpus and a different ontology. For this, a corpus of Wikipedia pages on terrorist attacks was compiled. This corpus consists of 100 Wikipedia pages and it was randomly split into a training set of 70 pages and a test set of 30 pages. A simple ontology for terrorist attacks was also constructed based on the fields of infoboxes[1] of the selected Wikipedia pages (which we call *Wikipedia ontology*). The keys for the files were also derived from the infoboxes. Then, mappings between the concepts of this Wikipedia ontology and the MUC4 and Mindswap ontologies were identified. Based on these mappings, the XML files containing the metadata of the information extractors were reused together with the platforms to perform information extraction on the Wikipedia corpus.

Figure 5 shows sections of the ontologies used and some mappings between them. The next subsection presents the details of developing information extractors for the MUC4 corpus. The details of their reuse in the Wikipedia corpus is presented in the subsection that follows it. The final subsection presents the results of the case study.

[1] An infobox of a Wikipedia page, located in the top right hand corner, provides a summary of the contents of the page.

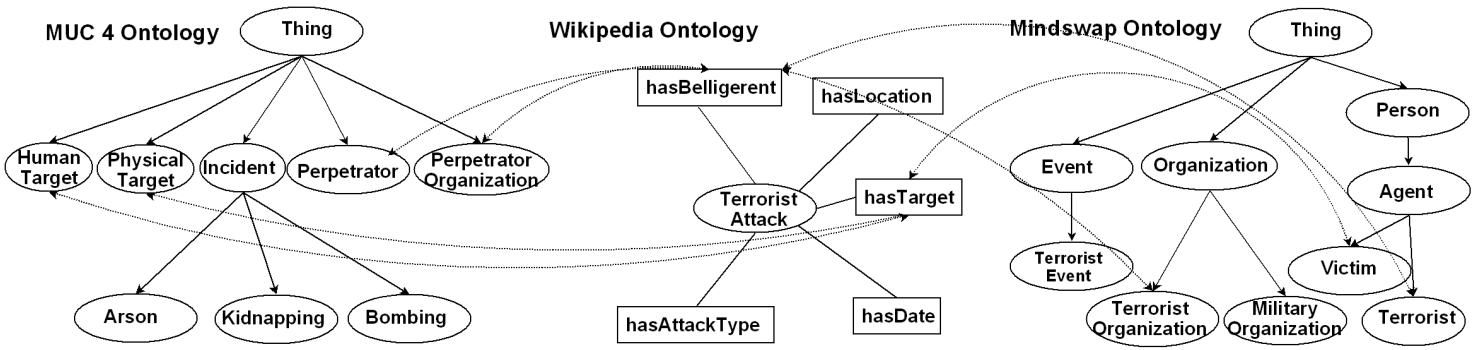


FIGURE 5: Different ontologies on terrorist attacks

The MUC4 Corpus

In order to discover the words and WordNet synsets to be used with the two-phase classification platform, human knowledge as well as a statistical approach were used. Certain words and their WordNet synsets were selected based on the general understanding of the concepts in concern. For instance, the words "Kill" and "Kidnap," as well as their WordNet synsets were selected as features for the class "Human Target." In addition, the following statistical technique was employed. First, the sentences that contain key values were selected from the training set. Then the words that are found in more than a predefined fraction of these sentences (5% was selected as the threshold after some experiments) were identified. Next correlation analysis (using the frequency of these words among *all* sentences) was used to ensure that the selected words have a positive correlation with keys instead of just being frequent words. The statistical measure of *lift* was used for this purpose and words having a lift of more than 1, which mean that they have a positive correlation with key sentences, were selected. Still, common words such as "the" and "to" were often included in the selected set of words and they were excluded from the final set of features. There was some overlap between the words selected based on human knowledge and words mined using the statistical technique but many new words were also discovered by the statistical technique.

Following the generally accepted practice on the use of gazetteers, standard sets of lists for certain concepts were selected as gazetteers. For instance, a set of Spanish first names provided by a website[2] was used as a gazetteer. In addition, some lists provided by the support material of the MUC 4 conference were used as gazetteers.

[2] `http://www.zelo.com/firstnames/names/spanish.asp`

The extraction rules to be used with the Mindswap ontology were discovered by first separating the sentences containing keys from the training files and then manually going through these key sentences to identify extraction patterns. Correlation analysis was used, in a manner similar to its application described above, to ensure that the discovered patterns are useful. In addition, gazetteers were identified in the manner described above. Most of these gazetteers were the same ones used with the classification platform.

As mentioned earlier, key files for the MUC4 ontology were adopted from the publicly available key files of the MUC 4 conference. For the Mindswap ontology, it was necessary to manually annotate the corpus for keys. This was the main reason for not using the entire MUC 4 corpus in the case study.

The aggregator component developed for multi-relation information extraction was used to combine the outputs of individual information extractors and produce OWL files as output.

The Wikipedia Corpus

The Wikipedia pages on terrorist attacks were identified from a list of terrorist incidents provided by Wikipedia. The majority of selected pages were on terrorist activities in the decade of 2000.

It was seen that infobox structure is not uniform among different Wikipedia pages since authors of pages can add their own attributes and delete existing one from the templates of infoboxes. As such only the attributes that are found in at least 20% of the pages were included in the Wikipedia terrorism ontology. A similar approach has been adopted by the Kylin [100] system. In addition, it was seen that the infoboxes of most of the pages had to be manually refined before being used as

gold standards. Often this included removing descriptions such as "(as reported)" and removing fields such as "Belligerent: Unknown." In some situations, missing information was manually filled.

As mentioned earlier, we need the mappings between ontologies in order to reuse the information extractors. Here, we need the mappings between the MUC4 and Wikipedia ontologies as well as the mappings between the Mindswap and Wikipedia ontologies. In order to discover these mappings, Anchor Flood [60] and Falcon-AO [64] systems were used, which are two recently developed mapping discovery systems. The precision of the discovered mappings were quite high (close to 80%) but although the recall was also high (close to 70%), it was observed that some important mappings that were planned to be used were not discovered by these systems. For instance, both systems failed to discover the mappings between *Belligerent* class of the Wikipedia ontology and *Perpetrator* and *Perpetrator Organization* classes of the MUC4 ontology shown in figure 5 above. As we discuss in the following sections, our component-based approach can detect incorrect mappings to a certain extent but in the case of missing mappings there is no alternative other than manually reviewing the entire ontologies to discover mappings.

In addition to the mappings shown in figure 5, the following mappings were used.

- Between class *Location* of MUC4 and class *Location* of Wikipedia
- Between classes *City*, *Country* and *Location* of Mindswap and class *Location* of Wikipedia

– Between classes *Instrument* and *Instrument Type* of MUC4 and class *Weapon* of Wikipedia

In reusing the information extractors of the MUC4 corpus with the Wikipedia corpus, the black-box reuse, clear-box reuse and simple combination techniques described in Chapter III were used. When reusing clear-box reuse, words and extraction rules that can be used as features were identified using the statistical analysis techniques described above. In addition, the gazetteers used were changed to take the new domain into consideration. For instance, the gazetteer of Spanish first names was replaced with a list of common Indian, Arabic, U.S. and Spanish first names because the Wikipedia corpus described terrorist attacks in countries all over the world while the MUC4 corpus was restricted to Latin American countries. The platforms were not applied on the Wikipedia corpus while ignoring the MUC4 information extractors, because it was seen that this would be quite similar to clear-box reuse in most cases.

The aggregator component developed for relation-level information extraction was used to combine the outputs of individual information extractors and produce OWL files as output.

Results

In evaluating the results, a scorer that compares extractions made with the gold standard provided in the key files for the test set was used. It operates based on string matching (while allowing some prefixes and suffixes such as " ‘ " and " 's ") and counting words. The figures calculated using this scorer for precision, recall and F1 are shown in Tables 2 and 3 as percentages. The first column of each table shows the concept of the MUC4 or Mindswap ontology.

From the results shown in Tables 2 and 3, it can be seen that the reuse of information extractors has been generally successful. While the performance measurements have recorded a drop when the information extractors are reused, they have normally recorded a F1 in the range of 25%- 30%. Not surprisingly, clear-box reuse has shown better results than black-box reuse (because it adds some features better suited for the Wikipedia corpus).

The exception to this are the results for reuse based on mappings for the *Target* class of Wikipedia. For this mapping, the F1 is lower than 5%. In analyzing the reasons for this, it was found that targets specified by Wikipedia infoboxes are very different from the human targets (victims) and physical targets specified for MUC4 corpus. They contained very few person names, effectively invalidating the reuse of information extractors for the *Human Target* class of MUC4 and the *Victim* class of Mindswap. On the first glance, it appeared that there was a somewhat stronger relationship between the *Target* class of Wikipedia and the *Physical Target* class of MUC4 but a closer inspection revealed that even this relationship is questionable. Targets of Wikipedia often contained phrases such as "Moscow Metro," which were quite different from the physical targets identified in the MUC4 corpus. In other words, the mappings identified between the *Target* class of the Wikipedia ontology and classes of other ontologies appear to be contradicted by the actual data of the corpora although the mappings make sense intuitively. This means that even manually identified mappings can not be considered 100% accurate. Further, it can be seen that a drastic drop in performance when an information extractor is reused based on a mapping indicates that the mapping in concern might be incorrect.

For the application of each platform in each ontology, aggregate performance measures can be computed as follows.

Assume that $C = \{C_1, C_2, ..., C_n\}$ denotes the set of concepts (classes and properties) for which extractions are made. For each concept C_i, let $correct(C_i), total(C_i)$ and $all(C_i)$ denote the number of correct extractions, total number of extractions and the number of extractions in the gold standard respectively. Then aggregate measures for precision and recall can be calculated as follows. F1 will be the harmonic mean between precision and recall as usual.

$$Precision = \frac{\sum_{i=1}^{n} correct(C_i)}{\sum_{i=1}^{n} total(C_i)}$$

$$Recall = \frac{\sum_{i=1}^{n} correct(C_i)}{\sum_{i=1}^{n} all(C_i)}$$

Tables 4 and 5 present these results. The *Target* class has been excluded from the calculations for the Wikipedia corpus. These results further highlight that the reuse of information extractors has been generally successful. It can also be seen that the reusability of information extractors is higher with the classification platform. This can be expected because it uses words and WordNet synsets as features instead of hand-crafted rules for a particular corpus.

As mentioned earlier, an attempt was made to use the simple combination technique described in Chapter III. For instance, the output of the information extractors for the *Perpetrator* and *Perpetrator Organization* classes from the MUC4 ontology can be combined to get the results for the *Belligerent* class of Wikipedia and the output of this combination can be combined again with the output produced by combining the output for *Terrorist Organization* and *Terrorist* classes of Mindswap. As described in section 3.4., the union operator was selected

TABLE 4: Aggregate results for classification

Ontology/Method	Precision	Recall	F1
MUC4/Direct	23.59	42.77	30.41
Wikipedia/Black-Box	19.68	35.60	25.35
Wikipedia/Clear-Box	21.54	41.75	28.42

TABLE 5: Aggregate results for extraction rules

Ontology/Method	Precision	Recall	F1
Mindswap/Direct	39.02	57.46	46.48
Wikipedia/Black-Box	23.70	17.42	20.08
Wikipedia/Clear-Box	22.76	30.31	26.00

for combining the outputs since it has the potential to result in more correct extractions. However, it was observed that F1 measure drops by around 2% - 5%, when compared with the best original information extractor when results are combined in this manner. The possible reason for this is the increase of errors when the information extractors are reused in a different corpus.

4.4. Discussion

The performance measures for information extraction obtained in the above mentioned case study is slightly below the measures obtained by comparable information extraction systems. For instance, the Kylin system [100] has recorded F1 measures in the range of 40% - 45% for certain concepts when extracting information from Wikipedia pages. For the classification platform, the main reason for this appears to be the insufficiency of training data. Kylin has faced the same problem and has employed special mechanisms to add new training data to improve performance [100]. For the extraction rules platform, the only way to improve results is to identify better extraction rules. This typically requires more manual involvement.

It is worth highlighting again that one of the most important contributions of the OBCIE approach is providing a standard mechanism to store domain and corpus specific information used by information extraction techniques. This facilitates their reuse in a structured manner and ensures that these information are not lost over time. As mentioned earlier, extraction rules developed for specific concepts are the domain and corpus specific information in the extraction rules information extraction technique. We obtained some evidence that extraction rules developed for concepts can get lost over time as hypothesized through our communication with the authors of the FASTUS system [35]. As described in Chapter II, this system has participated in the 4th Message Understanding Conference and has used the extraction rules technique. By communicating with the authors of this system, we learnt that it has been superseded by another system and that even the developers of the system do not have easy access to the extraction rules, which are considered *source* of the program. While conceding that the situation might be different in some other information extraction systems, we believe that this provides an indication of what happens to the useful information captured by information extraction systems. Such information (extraction rules in this case), get mixed up with source code of the programs and often it is difficult to retrieve them to be used in a new application. A component-based approach with structured mechanisms to store such information would correct this situation.

This chapter presented the details of information extractors, platforms for information extraction and aggregators that have been developed under the OBCIE approach. The implementations of platforms do not require any preprocessor components although tasks such as POS tagging can be delegated to preprocessors. Formatter components, which make changes in the extractions made by individual

information extractors have also not been used in the implementations. Formatters are not very important in the domain of terrorist attacks but would be very important in some domains such as bioinformatics where the extractions for certain concepts are required to have standard names. For instance, gene names extracted from an article should match standard gene names. Moreover, ontology constructors have not been used in this case study since ontologies developed by other groups have been used. Development of ontology constructor components is a critical task for the development of the OBCIE approach. Therefore, it amounts to an important future work for the OBCIE approach.

CHAPTER V

MOBIE

In this chapter, we discuss how multiple ontologies can be used in an ontology-based information extraction system. Here, we explore how the components for information extraction described under the OBCIE approach can be used in a multiple-ontology environment. We use the acronym *MOBIE*, which stands for "Multiple Ontology-Based Information Extraction" to identify this research area. In this chapter we discuss the intuition and reasoning behind MOBIE and present the principles for its application. The details of case studies are presented in the next chapter.

The first section of this chapter discusses the reasons for the use of multiple ontologies in artificial intelligence. In the next section we present an overview on the use of multiple ontologies in information extraction. This includes the reasons for using multiple ontologies in information extraction, an overview of how multiple ontologies can be accommodated in information extraction systems as well as how this can be done for systems developed under the OBCIE approach (described in Chapters III and IV). The next section presents the details of operational principles that can be used to improve the information extraction process through the use of multiple ontologies. We conclude with a summary on the use of multiple ontologies in information extraction.

5.1. Multiple Ontologies for a Domain

In the early days of using ontologies in knowledge representation, there have been some efforts to develop an all-encompassing ontology that is capable of storing

all the knowledge of the mankind. The Cyc project [29] was the most prominent one among these efforts. They have largely failed because of the shear size and complexity of the human knowledge. Since then researchers have concentrated on developing two types of ontologies, namely *domain ontologies* and *upper level ontologies*. They are described below.

1. *Domain Ontologies:* These ontologies describe a particular area or a domain. For instance, domain ontologies can be developed to represent classes and properties in areas such as universities, countries and genes. A domain ontology for the domain of universities may contain classes such as "professor" and "student" and properties such as "takes class".

2. *Upper Ontologies:* These ontologies describe higher level classes and properties that can be found in several domains. As such these ontologies are expected to be referred by many ontologies. They contain classes such as "thing", "person" and "event" and properties such as "has name".

Multiple ontologies have been designed both for the case of upper ontologies as well as the case of domain ontologies. Different upper ontologies developed by different research groups include the Suggested Upper Merged Ontology (SUMO) [27], Dublin Core [6] and the General Formal Ontology (GFO) [9]. In terms of domain ontologies, several ontologies for the same domain can be identified from ontology repositories as well as by using search tools such as Swoogle [28], that search different ontology repositories. For instance, one ontology repository contains more than 10 ontologies for the tourism domain [5].

Generally speaking, the existence of multiple ontologies can be attributed to the complexity of the human knowledge. The existence of multiple ontologies

in the case of upper ontologies indicates that people cannot agree on a unified representation even for the so called higher level concepts. The same applies for individual domains, which may be even more complex than higher level concepts. As such, the existence of multiple ontologies has become the norm in the field of knowledge representation. Therefore, issues related to the use of multiple ontologies such as integrating multiple ontologies and discovering and using mappings between different ontologies (mappings refer to relationships between concepts of different ontologies as described in Chapter II) has attracted the attention of researchers as evidenced by the research papers published on these topics [43, 71].

Since ontology-based information extraction systems generally make extractions with respect to a particular domain, domain ontologies are more relevant to this field than upper ontologies. As such we concentrate our attention on multiple ontologies in the case of domain ontologies. By studying multiple ontologies defined for the same domain, we have identified the following as the two main scenarios for this case.

1. *Specializing in sub-domains:* For example in the domain of universities, several sub-domains can be identified such as North American universities, British universities and universities with a religious background. For each of these sub-domains, specific ontologies can be developed by paying special attention to the classes and properties unique to it.

2. *Providing different perspectives:* For example, one ontology for the domain of marriages might define two classes named "husband" and "wife", while another might define an object property named "is spouse of". These different ontologies look at the same domain from different angles or perspectives.

5.2. Multiple Ontologies in Information Extraction

Reasons

The existence of multiple ontologies for a particular domain has become the norm in knowledge representation as described in the previous section. However, almost all ontology-based information extraction systems developed so far make use of a single ontology for a particular domain. For example, each OBIE system discussed in Chapter II operates using a single ontology.

Since multiple ontologies is the norm rather than an exception, it cannot be conceived that there are any rules that prevent their usage in information extraction. Moreover, it can be hypothesized that there are certain opportunities in using multiple ontologies in information extraction. The following are the two main opportunities we perceive on this regard.

1. *Improvement in recall:*

   As defined in Chapter II, recall shows the number of correctly identified items as a fraction of the total number of correct items available. Recall and precision, which shows the number of correctly identified items as a fraction of the total number of items identified, are the two main performance metrics used in information extraction.

   When using multiple ontologies that provide different perspectives, it can be hypothesized that the information extraction processes guided by concepts of different ontologies would make more extractions together than what is possible by a single ontology, thus resulting in a higher recall. For instance, in the marriage ontologies described above, extractions made based on the "is spouse of" property would capture homosexual marriages in addition to some

heterosexual marriages while extractions based on "husband" and "wife" classes are likely to be more successful in retrieving instances of heterosexual marriages. Using both ontologies might therefore extract more instances than what either one is capable on its own. Similarly, when using ontologies that specialize on particular sub-domains, each ontology can be expected to be more successful in making extractions in its own sub-domain. Hence, a set of specialized ontologies can be expected to make more correct extractions than what is possible under a common ontology.

If the resulting multi-ontology system is more accurate as a whole than the single-ontology systems, the precision would also increase. On the other hand if there is some loss in accuracy when making more predictions, a drop in precision can be anticipated. We expect that greater improvements in recall would offset such losses in terms of the overall performance measure for the information extraction system. As mentioned in Chapter II, the F1-measure, which represents the harmonic mean between precision and recall can be used to measure the overall performance of an OBIE system. Hence, we expect an improvement if F1-measure even when there is reduction in precision due to a larger increase in recall.

2. *Supporting multiple perspectives:*

   Since each ontology directly represents a particular conceptualization or a perspective of the domain in concern, using multiple ontologies implies that the system is capable of handling the perspectives related to each of the ontologies. This means that the output of the system can be used to answer queries based on different perspectives. For example, the output of an ontology-based information extraction system for the marriage domain that

uses both marriage ontologies described above can be used to answer different queries such as "Is person A a husband?" and "Who is person A's spouse?".

Ontology-based information extraction systems that make use of multiple ontologies have to be developed in order to realize the opportunities described above. In next subsection, we present a high-level overview of how this can be done.

An Overview

An ontology-based information extraction system that uses a single ontology may either extract individuals and property values for that ontology (perform ontology population) or extract classes and properties (perform ontology construction) or perform both operations. We extend this idea in to a system that uses multiple ontologies by describing such a system as *a set of single ontology information extraction systems*. The only condition is that the set of ontologies be related to the same domain. Formally, this can be represented as follows.

Definition 5.1 (Multiple-Ontology Information Extraction System)**.** A multiple ontology information extraction for a set S of ontologies ($S = \{O_1, O_2, ..., O_m\}$), $EM(S)$ is given by, $EM(S) = \{E(O_1), E(O_2), ..., E(O_m)\}$, where each $E(O_i)$, $1 \leq i \leq m$ is a single-ontology information extraction system for ontology O_i. All the ontologies in the set S of ontologies are related to the same domain.

On the face of it, this definition is simple and not very interesting: we consider a multiple-ontology system to be a just set of single-ontology systems. The interestingness in the use of multiple ontologies arises from the *interaction* between ontologies in the system, as we describe in the remainder of this section.

The mechanisms in which these interactions take place cannot be included in the definition for a multiple-ontology information extraction system since they differ for different scenarios. As such, we use the above generic definition to represent a multiple-ontology information extraction system.

As described in the previous chapters, the two main operations of an information extraction system are ontology construction and ontology population. We concentrate on improving the ontology population through the interaction between ontologies. This is directly related to the hypothesized opportunity of improving recall through the use of multiple ontologies described in the previous subsection since we expect to make more correct extractions for individuals and property values through the interaction between ontologies. We do not plan to improve the ontology construction process through a similar interaction among ontologies. As such, if a multiple-ontology system attempts to construct more than one ontology, the multiple ontology construction operations would be executed completely independent of each other.

In describing the principles for the interaction among ontologies in ontology population, we use the concept of information extractor, defined under the OBCIE approach. As described in Chapter III, an information extractor is an independently deployable component of an information extraction system that makes extractions with respect to a particular class or a property of an ontology. Therefore, these principles for the interaction among different ontologies in a multiple-ontology system, presented in section 5.3., can be considered as being designed for systems that follow the OBCIE approach. One caveat here is that the principles only make use of the outputs produced by information extractors (extractions made for individual classes and individual properties). As such,

they can be even be applied for multiple-ontology systems that do not follow the OBCIE approach as long as the output for each class and property is clearly distinguishable. We describe this point in more detail in section 5.3.

It should be noted that the focus of the use of multiple ontologies in an information extraction system is improving the results of information extraction on the *same corpus*. When multiple ontologies are use for the same corpus, it is expected that they would make more correct extractions together than either one of them used in isolation. This is different from reusing components for information extraction in different corpora as carried out by the OBCIE approach.

In the next subsection, we describe how multiple-ontology information extraction systems that fit into the definition for a multiple ontology system (Definition 5.1) can be developed under the OBCIE approach. In such systems, the components for information extraction used with different ontolgies, operate on the same corpus.

MOBIE under OBCIE

Since Definition 5.1 describes a multiple-ontology information extraction system as a set of single-ontology information extraction systems, the specifications for an ontology-based information extraction system under OBCIE presented in section 3.3. of Chapter III can be easily extended in to the multiple ontology case. Figure 6 schematically represents such a system. To satisfy the conditions of Definition 5.1, the set of ontologies should be related to the same domain.

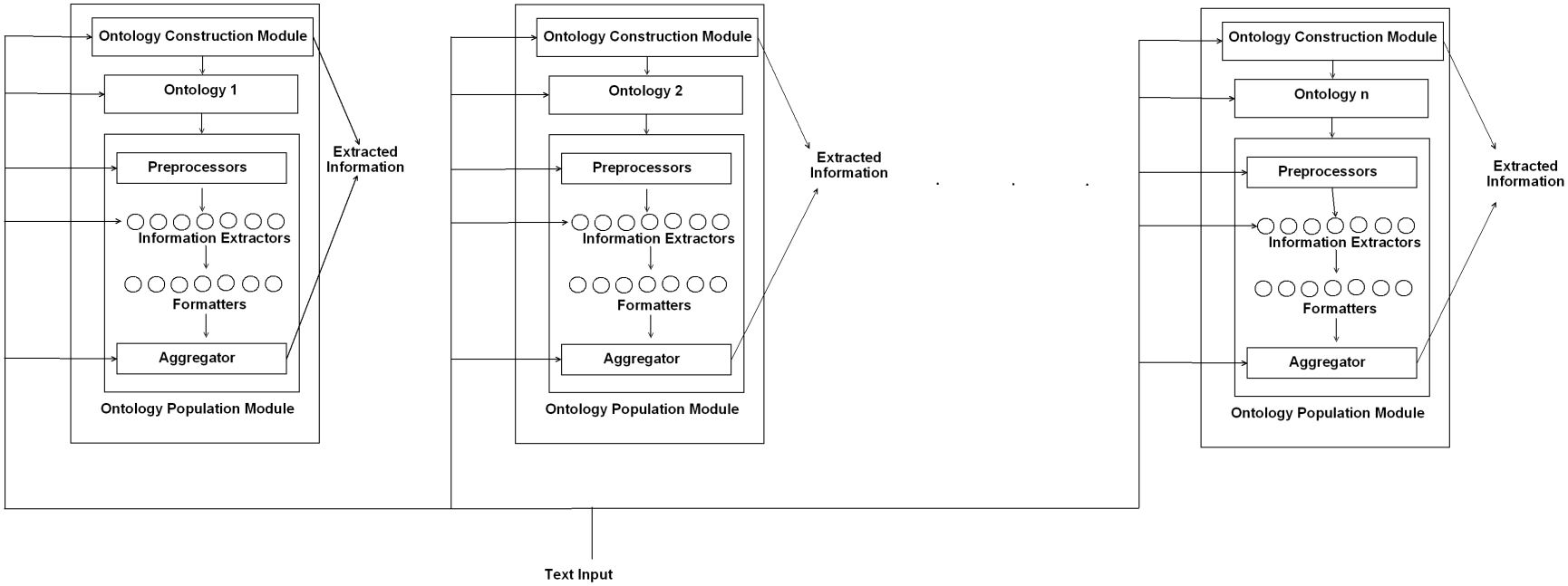


FIGURE 6: A multiple-ontology OBIE system under the OBCIE approach

The Z specifications for a system under the OBCIE approach, presented in section 3.3., can be extended into the multiple ontology case. Assuming that the multiple-ontology system uses n ontologies, this can be done by using n schemata for the n ontologies and a set of n operations for each operation. The same approach can be adopted with respect to the Alloy specifications presented in Appendix A.

5.3. Operational Principles

In this section, we present the principles that can be used for the interaction among different ontologies in a multiple-ontology system for the ontology population operation. As mentioned earlier, these principles are designed assuming that the systems in concern follow the OBCIE approach. We first revisit how ontology population takes place for a single-ontology system under the OBCIE approach. Then, we show how this can be extended for the multiple-ontology case while allowing for the interaction among different ontologies. As described earlier, the interaction between different ontologies is carried out with the objective improving performance measures, mainly recall.

Ontology Population for a Single Ontology

Under the OBCIE approach, information extractors are used to make extractions with respect to specific classes and properties of ontologies. Each information extractor either identifies individuals for a class or values for a property. In a single-ontology system, all these classes and properties belong to a single ontology.

Lets assume that a single-ontology information extraction system under the OBCIE approach makes extractions with respect to n ontological concepts (classes and properties) in an ontology O. As usual, $O = (C, P, I, V, A)$ where $C, P, I, V,$ *and* A are the sets of classes, properties, individuals, property values and other axioms respectively.

In this case, there will be a set of n information extractors, which we denote by $I(O)$. We denote the individual information extractors by $E(O, X_i)$ $1 \leq i \leq n$.
In other words,
$I(O) = \{E(O, X_1), E(O, X_2), ..., E(O, X_n)\}$
where $\forall\ i\ (1 \leq i \leq n),\ X_i \in\ C$ *or* $X_i \in\ P$.

For a given corpus D, each extractor $E(O, X_i)$ would make a set of extractions $R(E(O, X_i), D)$, which according to its predictions are either individuals or property values. Some of these extractions may be incorrect.

We denote the actual individuals and property values found in D (often known as the *gold standard* or the *key*) by $k(I, D)$ and $k(V, D)$ respectively. It is assumed that all these actual individuals and property values are included in I and V. Formally,

$$k(I, D) \subset I \text{ and } k(V, D) \subset V$$

Using these definitions, we can obtain formulae for precision and recall of the set of information extractors $I(O)$. These are shown below.

$$Precision(I(O)) = \frac{|\bigcup_{i=1}^{n} R(E(O, X_i), D) \cap \{k(I, D) \cup k(V, D)\}|}{|\bigcup_{i=1}^{n} R(E(O, X_i), D)|}$$

$$Recall(I(O)) \ = \frac{|\bigcup_{i=1}^{n} R(E(O, X_i), D) \cap \{k(I, D) \cup k(V, D)\}|}{|\{k(I, D) \cup k(V, D)\}|}$$

Ontology Population for Multiple Ontologies

Since a multiple-ontology information extraction system is defined simply as a set of single-ontology systems, ontology population in the multiple-ontology case can be carried out by just using a set of information extractors with respect to each ontology in concern. This would, however, not lead to any improvement in the information extraction process. As mentioned earlier, the key here is the interaction between different ontologies. We use different interaction mechanisms for the two scenarios for having multiple ontologies for the same domain, described in section 5.1., namely specializing on sub-domains and providing different perspectives on the entire domain. They are presented separately below.

First we handle the scenario of *multiple ontologies specializing on sub-domains*. In this case, we have a generic (common) ontology O_c and a set of m specialized ontologies S given by $S \ = \ \{O_1, O_2, ..., O_m\}$.
Let $O_c \ = (C_c, P_c, I_c, V_c, A_c)$ and $\forall \ i \ (1 \leq i \leq m), \ O_i \ = (C_i, P_i, I_i, V_i, A_i)$

In performing information extraction on a given corpus D, the single-ontology information extraction system would use the common ontology O_c. A single ontology system can use only one ontology and this is the logical choice for an ontology: all other ontologies are specialized in a sub-domain and the common ontology is for the entire domain. The set of information extractors used by this system can be represented by $I(O_c)$. Assuming that extractions are made for n_s concepts, there will be a n_s number of information extractors. They can be represented as follows.

$I(O_c) = \{E(O_c, X_{c,1}), E(O_c, X_{c,2}), ..., E(O_c, X_{c,n_s})\}$
where $\forall\ i\ (1 \leq i \leq n_s),\ X_{c,i} \in\ C_c\ or\ X_{c,i} \in\ P_c$

The multiple-ontology system would use the set of specialized ontologies. It has the capability of handling more than one ontology and thus it makes sense to use the set of specialized ontologies with the objective of improving the information extraction process as we have discussed earlier in this chapter. In this case, we expect the system use only a *single ontology* for each document of the corpus. Since we have a set of specialized ontologies, it makes sense to determine which ontology is best suited for each document and process it using the selected ontology. This operation is carried out as follows.

For each of the K documents of the corpus D_j $(1 \leq j \leq K)$, this system has to determine which ontology to use. It would try to use the most suitable ontology for each document. This selection would be performed by an *ontology selector* component and it can be represented by a function **os**, which returns the number of the selected specialized ontology.
$\forall\ j\ (1 \leq j \leq K),\ os(D_j)\ \in\ \{1, 2, ..., m\}$

Once an ontology is assigned to a document, it is processed using the information extractors of that ontology. Figure 7 schematically represents this process. Based on this functionality, we can obtain formulae for precision and recall for the set of information extractors. These are presented below.
Using $\alpha =\ os(D_j),\ 1 \leq j \leq K$,

$$Precision(IM(S))\ = \frac{\sum_{j=1}^{K} |\bigcup_{i=1}^{r_\alpha} R(E(O_\alpha, X_{\alpha,i}), D_j) \cap \{k(I_\alpha, D_j) \cup k(V_\alpha, D_j)\}|}{\sum_{j=1}^{K} |\bigcup_{i=1}^{r_\alpha} R(E(O_\alpha, X_{\alpha,i}), D_j)|}$$

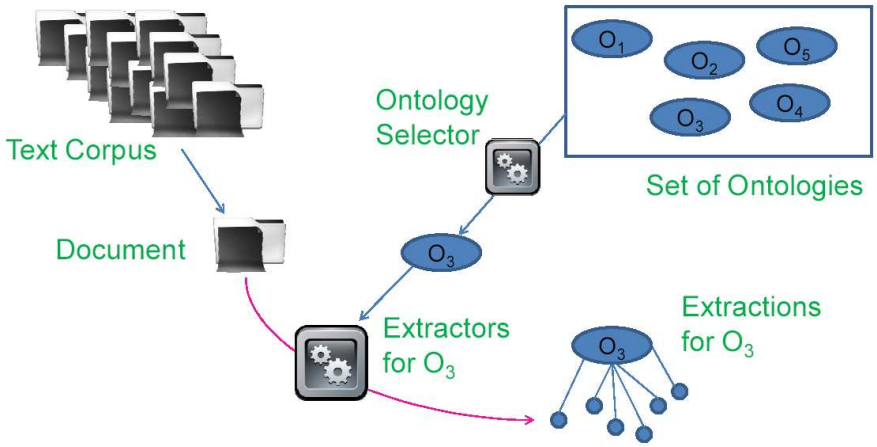


FIGURE 7: MOBIE for ontologies specializing on sub-domains

$$Recall(IM(S)) = \frac{\sum_{j=1}^{K} |\bigcup_{i=1}^{r_\alpha} R(E(O_\alpha, X_{\alpha,i}), D_j) \cap \{k(I_\alpha, D_j) \cup k(V_\alpha, D_j)\}|}{\sum_{j=1}^{K} |\{k(I_\alpha, D_j) \cup k(V_\alpha, D_j)\}|}$$

The precision and recall for the single-ontology system can be obtained using the formulae given in the previous section. When calculating the recall, these formulae will only consider the instances and property values found with respect to the common ontology. However, more instances and property values will exist with respect to the specialized ontologies used by the multiple-ontology system. It is possible to compute a separate measure of recall with respect to these. We call this measure *global recall* and call the recall computed with respect to the common ontology, which is the standard measure of recall, *local recall*. The formula for global recall is shown below.

$$Global\ Recall(I(O_c))) = \frac{|\bigcup_{i=1}^{n_s} R(E(O_c, X_{c,i}), D) \cap \{k(I_c, D) \cup k(V_c, D)\}|}{\sum_{j=1}^{K} |\{k(I_\alpha, D_j) \cup k(V_\alpha, D_j)\}|}$$

Here it is assumed that, $k(I_c, D) \subset \bigcup_{j=1}^{K} k(I_\alpha, D_j)$ and $k(V_c, D) \subset \bigcup_{j=1}^{K} k(V_\alpha, D_j)$.

In other words, the common ontology would only contain classes and properties common to all the specialized ontologies. This is what one would normally expect from a common ontology.

Now, we move on to the scenario of multiple ontologies providing different perspectives for the entire domain. Here, we have a set of m ontologies, $S = (O_1, O_2, ..., O_m)$ which have the same definitions as presented above. Since none of these ontologies can be seen as a "common ontology", there will be a set of single-ontology systems. Their sets of information extractors can be represented by $I(O_i)$, $1 \leq i \leq m$. This set of single-ontology systems can also be considered a multiple-ontology system according to the definition presented above.

It should be noted that it does not make sense to use an ontology selector component in this scenario. Each ontology is for the entire domain and as such a document can not be meaningfully assigned to a single ontology. Therefore, for a given corpus D, each document of the corpus will have to be processed with respect to each and every ontology.

We can improve the performance of the multiple-ontology system in this scenario through the use of *mappings* between concepts of different ontologies. The mappings we use here are the mappings falling under the category of "direct mappings" as defined in Definition 3.5. The general idea here is that there should be an equivalence or subset relationship between the sets of individuals or values falling under the two concepts (classes and properties) in concern.

Assuming that single-ontology systems have already been developed for individual ontologies, the multiple-ontology system can make use of the information extractors of these systems. In developing an information extractor to identify individuals of a given class or property values of a given property, the multiple-

ontology system can use the extractors of *more than one* single-ontology system. The intuition behind this approach is implementing a better information extractor by combining a set of different information extractors. For example, for the marriage ontologies we have discussed earlier, the extractor for the "spouse" class in the multiple-ontology system can use not only the results for the "spouse" class but also the results of the "husband" and "wife" classes of a different ontology. Here, the general idea is to use the information extractors for all the concepts that have some *mapping* with the concept in concern. This process can be represented as follows.

Let $X_{i,j} \in \{C_i \cup P_i\},\ 1 \leq i \leq m \ and\ 1 \leq j \leq r_i$ be a class or a property of ontology O_i. (m is the number of ontologies and r_i is the number of information extractors for ontology O_i)

Let $\overline{X} = \{\overline{X_1}, \overline{X_2}, ..., \overline{X_n}\}$ be the set of n properties or classes of other ontologies, which have a mapping $M(X_{i,j}, \overline{X_l}), 1 \leq l \leq n$ and let $o(\overline{X_l}) \in \{1, 2, ..., m\}, 1 \leq l \leq n$ denote the number of the ontology for $\overline{X_l}$.

The information extractor for $X_{i,j}$ in the multiple-ontology system can make use of not only the information extractor for $X_{i,j}$ but also of the information extractors for elements of $\overline{X}$ in the single-ontology systems. The extractions made by the information extractor $E(O_i, X_{i,j})$ for the corpus D in the multiple-ontology system depends on a set of single-ontology information extractors as follows.

$R(E(O_i, X_{i,j}), D) = f_{i,j}(R(E(O_i, X_{i,j}), D), R(E(O_{o(\overline{X_1})}, \overline{X_1}), D), ...,$
$R(E(O_{o(\overline{X_n})}, \overline{X_n}), D))$

Here function $f_{i,j}$ presents the operation used to combine the outputs of different information extractors. We expect to use the *union set operator* for this purpose. This means that the output of the information extractor of the multiple ontology system would be the union of the outputs of information extractors of the single ontology systems. The reasoning behind the use of the union operator here is the same as the reasoning provided for the use of the union operator in the simple combination technique under the OBCIE approach, as described in Chapter III (section 3.4.): it has the potential to result in more correct extractions.

Figure 8 schematically represents the operation a multiple ontology system for the scenario of ontologies providing different perspectives. Based on the specified functionality, we can obtain formulae for the precision and recall of the set of information extractors as shown below.

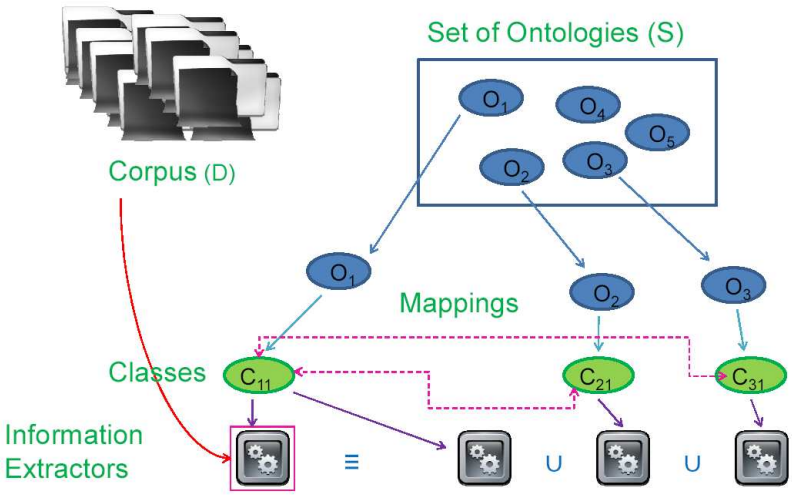


FIGURE 8: MOBIE for ontologies providing different perspectives

$$Precision(IM(S)) \ = \frac{\sum_{i=1}^{m} |\bigcup_{j=1}^{r_i} R(E(O_i, X_{i,j}), D) \cap \{k(I_i, D) \cup k(V_i, D)\}|}{\sum_{i=1}^{m} |\bigcup_{j=1}^{r_i} R(E(O_i, X_{i,j}), D)|}$$

$$Recall(IM(S)) \ = \frac{\sum_{i=1}^{m} |\bigcup_{j=1}^{r_i} R(E(O_i, X_{i,j}), D) \cap \{k(I_i, D) \cup k(V_i, D)\}|}{\sum_{i=1}^{m} |\{k(I_i, D) \cup k(V_i, D)\}|}$$

For the single ontology systems, precision and recall can be defined using the formulae specified above. We can also define a formula for global recall as follows.

$$Global\ Recall(I(O_i)) \ = \frac{|\bigcup_{j=1}^{r_i} R(E(O_i, X_{i,j}), D) \cap \{k(I_i, D) \cup k(V_i, D)\}|}{\sum_{i=1}^{m} |\{k(I_i, D) \cup k(V_i, D)\}|}$$

Here the number of information extractors in the single-ontology system $I(O_i)$ is r_i because both the multiple-ontology system and this system has the same number of information extractors for ontology O_i, corresponding to the total number of classes and properties for which information extraction is carried out.

In reviewing the principles presented above, it can be seen that only the *outputs* of the information extractors are needed to employ them. For ontologies specializing on sub-domains, information extractor are not really relevant since a single document is processed with respect to only one ontology. The information extraction system may even me a monolithic system as long as it produces output for each concept separately for results to be evaluated. For ontologies providing different perspectives on the entire domain, only the outputs of individual information extractors are needed since the combination is carried out through the

union operator. Again the requirement is to have the output for each class and property separately.

Because the requirement for the operation of the principles is only to have the outputs separately for each class and property, even systems that do not follow the OBCIE approach can be used for multiple-ontology based information extraction. These systems would simply be assumed to follow the OBCIE approach for the purpose of applying the principles.

5.4. Summary

MOBIE, which stands for "Multiple Ontology-Based Information Extraction", is the area of study on the use of multiple ontologies in information extraction systems. It is has the potential to improve the information extraction process, most notably by improving recall. Multiple domain ontologies that are available for most domains can be used in this exercise.

We define a generic multiple ontology-based information extraction system simply as a set of single-ontology systems. The advantages of using multiple ontologies arise from the interaction between different ontologies. Different mechanisms are used for this interaction for the two scenarios for having multiple ontologies for the same domain, namely specializing on sub-domains and providing different perspectives on the entire domain. These mechanisms are only used to improve the ontology population operation.

The principles for the interaction between different ontologies have been designed using the concept of information extractor, which refers to an independently deployable component of information extraction system that makes extractions with respect to specific ontological concepts, defined under the OBCIE

approach. However, these principles only require the outputs of information extractors to be distinguishable from each other. As such they can even be applied on systems that do not follow the OBCIE approach, as long as the output related to individual classes and properties of the ontoloiges can be separately identified.

CHAPTER VI

CASE STUDIES ON MOBIE

In this chapter, we discuss the details of two case studies conducted for MOBIE (Multiple Ontology-Based Information Extraction) based on the principles presented in Chapter V. One of these case studies is for the scenario of multiple ontologies specializing on sub-domains while the other is for the scenario of multiple ontologies providing different perspectives on the entire domain. The domain of universities was selected for the case study on ontologies specializing on sub-domains while the domain of terrorist attacks was selected for the case study on ontologies providing different perspectives on the entire domain. The text corpora for the case studies were either compiled from public sources or adopted from publicly available corpora. In both case studies, the corpus was divided into a training set and a test set. The training set was expected to be used to develop the information extraction systems whose performance could then be measured using the test set.

In the remainder of this chapter, we discuss the details of each case study. A separate section is used for each case study with subsections for corpus and ontologies used, design and implementation details and the results obtained. We conclude with a discussion on the results s well as some general observations on the case studies.

6.1. Ontologies Specializing on Sub-Domains

Corpus and Ontologies

As mentioned earlier, the domain of universities was used in this case study. The corpus consisted of web pages of 100 universities, 50 from North America and 50 from other parts of the world. From each group, 30 were selected for the training set and 20 were used as the test set. Since the set of all documents of a university website is typically very large and contains many pages irrelevant to the task of extracting information about the university (such as personal websites), only a selected set of webpages was included in the corpus. A programming interface to the Google search engine was used for this purpose. This program takes the domain name of a university as the input and selects a set of webpages from that domain by searching for certain key words (e.g., "about us", "statistics").

Two specialized ontologies and one common ontology were used in this case study. The specialized ontologies were on the sub-domains of North American universities and universities of the other parts of the world. These ontologies were developed by studying the documents of the training sets and other university ontologies. An ontology developed by the Simple HTML Ontology Extensions (SHOE) project [24] was quite useful in developing the North American ontology. The development of the non-North American ontology was primarily based on documents of the training set. A common university ontology was designed by identifying the concepts common to the two specialized ontologies.

In designing ontologies for North American and non-North American universities, we tried to capture the differences between two types of universities whenever possible. For example, an object property named "hasPresident"

was defined in the North American ontology while a property named "hasViceChancellor" was defined in the non-North American ontology. The corresponding property was named "hasFunctionalHead" in the common ontology since it was expected to represent the common characteristics of all universities.

The ontologies were defined in OWL using Protégé [19] ontology editor. Figure 9 shows a section of class hierarchy of the common ontology. Figures 10 and 11 show sections of class hierarchies related to employees of a university in the ontologies for North American and non-North American universities respectively.

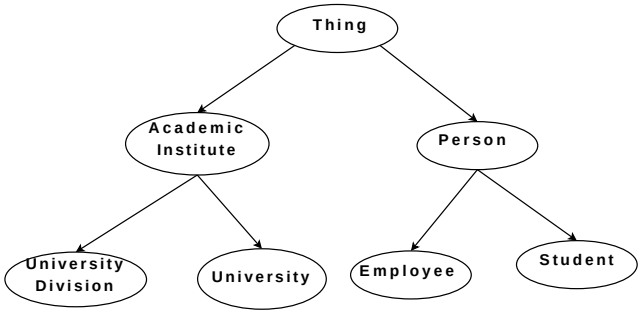


FIGURE 9: A section of the common university ontology

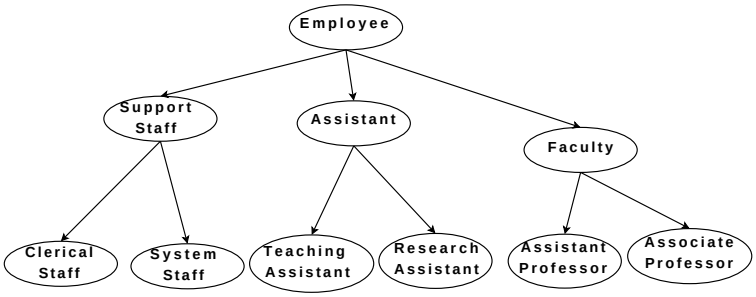


FIGURE 10: A section of the ontology for North American universities

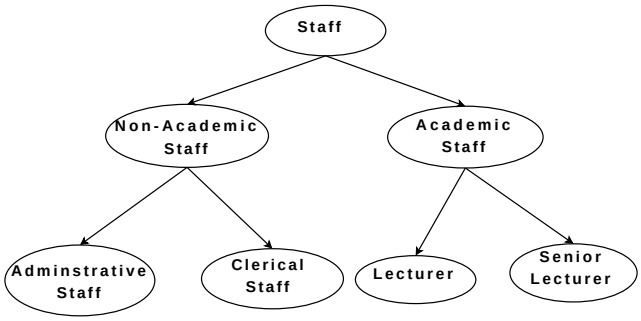


FIGURE 11: A section of the ontology for non-North American universities

Design and Implementation

In this case study, extraction rules were used as the information extraction technique. As described in Chapter II, this technique is based on the use of regular expressions that capture certain types of information. Specific words, phrases and linguistic features such as Part-Of-Speech (POS) tags were used in the extraction rules. As in the case studies on OBCIE presented in Chapter IV, these extraction rules were deployed using the The General Architecture for Text Engineering (GATE) [8]. The extraction rules were specified in the JAPE format for this purpose.

Information extraction was performed for a selected set of classes and properties in each ontology. The common university ontology was selected for the single-ontology system according to the principles described in Chapter V. The multiple-ontology system uses the two ontologies specializing in sub-domains, namely North American universities and non-North American universities.
As described in Chapter V, the multiple-ontology system requires an *ontology selector* component which assigns a specialized ontology for each document. This component was designed to make use of the URLs of university websites. The

documents from domains .edu, .ca and .us are assigned to the North American ontology while the others are assigned to the non-North American ontology.

This case study was conducted before the platform for extraction rules under the OBCIE approach was developed and as such the developed information extraction systems do not fall under OBCIE. Each system used a single JAPE file containing all the extraction rules used by the system instead of storing these rules in an XML file as done under the OBCIE approach. The outputs of produced by JAPE rules are written to a set of files, which are later processed by another programming module to produce the final output in the OWL format. This module makes use of the same Java OWL API [31] used by the aggregator components described in Chapter IV. However, it is designed specifically for the set of classes and properties used and as such does not amount to a reusable aggregator component.

Although the developed information extraction system do not follow the OBCIE approach, the output for each class and property can be distinguished from the OWL files produced. Therefore, the operational principles of MOBIE can be applied in this case study, as described in Chapter V. In order to evaluate the performance of the systems, gold standards were manually created for the documents in the test set.

Some classes and properties selected for information extraction are presented below. For each class or property, the ontologies in which it is found, either directly or by a concept directly mapped into it, are shown within parenthesis (using the symbols NA, NNA and C to denote North American, non-North American and Common ontologies respectively). Note that some concepts are only found in specialized ontologies.

– Classes *University* (NA,NNA,C) and *UniversitySystem* (NA)

– Object properties *hasFunctionalHead* (NA,NNA,C) and *hasCeremonialHead* (NA,NNA,C)

– Datatype properties *isFoundedOn* (NA,NNA,C) and *isReligiousUniversity* (NA,NNA,C)

The regular expressions used for information extraction were manually written by studying the documents of the training set. In some cases, different regular expressions were used for concepts of different ontologies that were directly mapped to each other. For example, for the *isReligiousUniversity* datatype property, patterns based on the words "Christian" and "Catholic" were used for North American universities while patterns based on the word "Islamic" were also used for non-North American universities.

Results

Table 6 shows the summary of the results obtained. It shows the precision, recall and F1 measure as percentages for each sub-domain as well as for the entire domain. Note that the figures for the entire domain are not the averages of the corresponding figures for the two sub-domains because the number of extractions made for the two sub-domains are different. It can be seen that the multiple-ontology system has shown improvements in all three measures. The improvement in recall is somewhat higher than the improvement in precision. Altogether, the multiple ontology system has shown an improvement of about 5% in F1 measure for the entire corpus.

TABLE 6: Summary of the results obtained for the university domain

System	Domain	Precision	Recall	F1	Global Recall
Single Ontology	North American	52.86	37.00	43.53	34.91
	Non-North American	47.83	52.38	50.00	52.38
	All	**50.86**	**41.55**	**45.74**	**39.86**
Multiple Ontology	North American	54.65	44.34	48.96	44.34
	Non-North American	52.17	57.14	54.54	57.14
	All	**53.79**	**47.97**	**50.71**	**47.97**

We have also computed the global recall of each system according to the definitions presented in Chapter V. It can be seen that the global recall is slightly lower than the standard recall (local recall) for North American universities in the single-ontology system. This is because some concepts specific to the North American university ontology (such as the class for university systems) were used by the multiple-ontology system. No such concepts were used for non-North American ontology and as such the figure for global recall is the same as local recall for these universities in the single-ontology system.

6.2. Ontologies Providing Different Perspectives

Corpus and Ontologies

The domain of terrorist attacks was used in this case study. The corpus used is the same corpus used in the case study on the OBCIE approach, described in Chapter IV. As described there, this corpus consists of 200 news articles on terrorist activities in Latin American countries, taken from the training set of the MUC 4 conference. As in the case study described in Chapter IV, 160 articles were used as the training set while the remaining 40 were used as the test set.

Two ontologies that provide different perspectives on the domain of terrorist attacks were used in the case study. They were the MUC4 and Mindswap terrorism ontologies described in Chapter IV. As mentioned in Chapter IV, the MUC4 ontology was derived from the structure of key files in the MUC4 conference while the Mindswap ontology is an ontology developed by the Mindswap group of University of Maryland [12]. As described in Chapter IV, the key files for the MUC4 ontology were available from public references of the conference while the manual annotations were made for keys with respect to the Mindswap ontology.

Sections of the MUC 4 and Mindswap ontologies relevant to this case study are shown in figure 12. This figure also shows some mappings between the two ontologies.

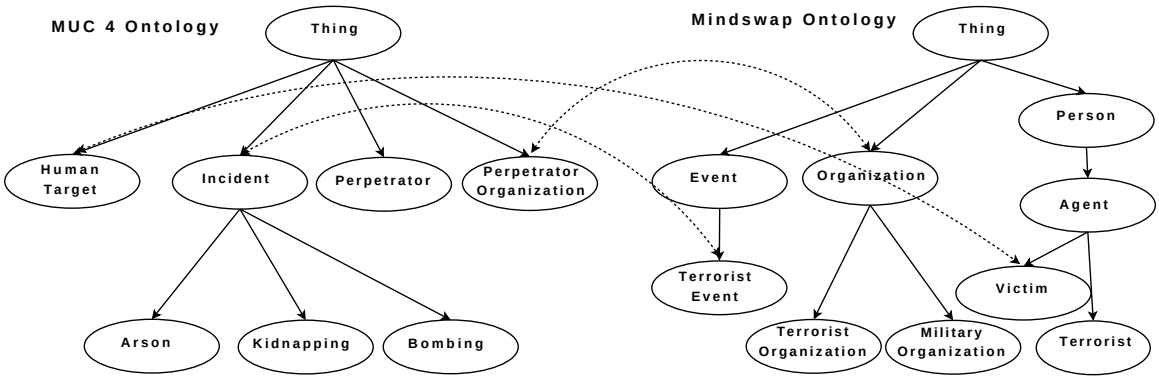


FIGURE 12: Sections of MUC 4 and Mindswap terrorism ontologies

Design and Implementation

This case study was conducted before the platforms for information extraction of the OBCIE approach were developed and as such the systems developed for it do not fall under the OBCIE approach, similar to the case study on ontologies specializing on sub-domains presented in section 6.1. It was possible to apply the operational principles of MOBIE for these systems because it was possible to clearly distinguish the outputs made for each class and property.

Once the platforms for information extraction were developed under the OBCIE approach and information extractors were developed for some classes and properties of MUC4 and Mindswap ontologies, it was possible to apply the principles of MOBIE for some of these information extractors as well. We also present the details of this application.

In systems developed prior to the development of platforms for information extraction, information extraction was restricted to identifying sentences in which concepts in concern (individuals of classes or property values of properties) are found. This is often used as an intermediate step in information extraction, especially when classification is used as the information extraction technique (e.g., the Kylin OBIE system [101]). The next step is to identify the words within a sentence that represent the concept in concern. It is possible to evaluate the effects of using multiple ontologies instead of a single ontology in information extraction by comparing the results for the two cases at the sentence level as well as the word level. In this case study, the results at the sentence level were compared.

In developing these information extraction systems, the classification technique was used for the MUC4 ontology while extraction rules were used for the Mindswap ontology. Following the operational principles presented in Chapter V,

the multiple-ontology information extraction system was developed by combining the outputs of the two single ontology systems through the use of mappings.

The classification-based information extraction system for the MUC4 ontology was developed using a set of features including specific key words, WordNet [53] synsets for key words and Part-Of-Speech tags. Classification was carried out using the Weka [58] system. Different classification techniques were used to find out the techniques that produce best results. In addition, the techniques that address the problem of imbalanced classification encountered in using classification for information extraction such as the use of weights, oversampling and bagging/boosting [68] were also used. For the Mindswap ontology, the extraction rules were specified by studying the training set as in the case study on university ontologies.

As in the case study on ontologies specializing on sub-domains, information extraction was performed only on a selected set classes and properties instead of covering all the concepts of the ontologies. For the MUC 4 ontology, extractions were made for the following properties.

- *hasName* and *hasDescription* for *HumTgt* class
- *hasPerpInd* for *Perpetrator* class
- *hasName* for *PerpetratorOrganization* class
- *hasName* and *hasInstrumentType* for *Instrument* class

For the Mindswap ontology, the values for the *hasName* datatype property of the *Agent* class and its subclasses (*Government Agent*, *Terrorist* and *Victim*) and the Organization class and its subclasses (*Terrorist Organization*, *Government*, *Government Organization* and *Military Organization*) were extracted.

In developing the multiple-ontology information extraction system, the following mappings were used.

1. $val(@MUC4 : hasName(HumTgt)) \subset val(@Mindswap : hasName(Victim))$
2. $val(@MUC4 : hasName(PerpetratorOrganization)) \subset val(@Mindswap : hasName(Organization))$

We denote the MUC 4 and Mindswap ontologies by $@MUC4$ and $@Mindswap$ respectively. What follows after ":" is the class or property represented.

The first mapping states that each human target name in the MUC 4 ontology is also a name of a victim in the MindSwap ontology. The second mapping states that each name of a perpetrator organization in the MUC 4 ontology is also a name of an organization is the Mindswap ontology. These mappings can also be expresses in First-Order Logic. For example, the following FOL statement represents the first mapping.

$\forall x, y\ @MUC4 : HumTgt(x) \wedge @MUC4 : hasName(x, y) \rightarrow \exists z\ @Mindswap : Vitim(z) \wedge @Mindswap : hasName(z, y)$

These mappings were manually identified along with some others such as the mapping between "incident" and "terrorist event" classes of MUC 4 and Mindswap ontologies which were not used because those classes and properties were not selected for information extraction. In the first mapping, a subset relationship is used instead of an equivalence relationship because the key files provided by MUC 4 are more restrictive in identifying terrorist incidents and victims than the keys for the Mindswap ontology. Regarding the second mapping, the "perpetrator organization" class of the MUC 4 ontology includes terrorist organizations as well as military organizations. MUC 4 does not have specialized classes for these

different types of organization. Therefore, the mapping can be made only to the "organization" class of MindSwap ontology even though specialized classes are available there.

These mappings were used in generating extractions for the Mindswap ontology in the multiple-ontology system. As described in Chapter V, the union of the outputs for the two related concepts was used as the output for multiple-ontology system. This can be represented as follows using the notation used in Chapter V. Here E_m, E_s denote the information extractors of the single and multiple ontology systems while D denotes the corpus.

$R(E_m(Mindswap, hasName(Victim), D)) =$
$R(E_s(Mindswap, hasName(Victim), D)) \cup$
$R(E_s(MUC4, hasName(HumanTarget), D))$
$R(E_m(Mindswap, hasName(Organization), D)) =$
$R(E_s(Mindswap, hasName(Organization), D)) \cup$
$R(E_s(MUC4, hasName(PerpetratorOrganization), D))$

The extractions for the MUC 4 ontology in the multiple-ontology system were the same as those made by the single-onotlogy system for MUC 4 because the identified mappings cannot be used to improve them.

So far we have only described the application of operational principles of MOBIE in systems that were developed prior to the development of the platforms for information extraction under the OBCIE approach. As mentioned earlier, these principles were used for combining the output of information extractors described in Chapter IV as well. The first mapping presented above was used for this purpose. Information extractors were developed for "victim" class of Mindswap and "human target" class of MUC4 in the case study on OBCIE and the first

TABLE 7: Results for different classification techniques for the MUC 4 ontology

IE Technique	**Precision**	**Recall**	**F1**
Bayes Net	28.33	49.28	35.27
Naïve Bayes	25.72	54.36	34.41
Naïve Bayes Updateable	25.72	54.36	34.41
Bagging - Bayes Net	28.13	47.31	34.60
Bagging - Naïve Bayes	26.77	52.09	34.80
Bagging - Naïve Bayes Updateable	26.35	52.09	34.44

mapping was used to update the extractions made for the "victim" class of the Mindswap ontology.

Results

Different classification techniques were used for the single-ontology system for the MUC 4 ontology as mentioned earlier. Here, Bayesian techniques consistently showed better results than other classification techniques. Bagging improved results in some cases while other techniques used to address data imbalance problem such as oversampling and the use of weights did not significantly improve performance (or resulted in a deterioration). Table 7 shows a summary of the results obtained.

It can be seen that the Bayesian network classification technique has shown the highest F1-measure. Therefore, the results given by this technique are used as the results of the single-ontology information extraction system for the MUC 4 ontology. Since mappings were not used to modify the results for the MUC 4 ontology in the multiple ontology system, these figures also represent its results for the MUC 4 ontology. These were also used in the multiple-ontology system for the Mindswap ontology based on mappings between concepts of different ontologies.

Tables 8 and 9 show summaries of the results obtained for the Mindswap ontology in the single and multiple ontology systems respectively. They show results separately for the "agent" class and its subclasses and the "organization" class and its subclasses in addition to the results for all the classes. They also show figures obtained for precision, recall and F1-measure using the standard definitions and using the concept of *taxonomy similarity* [45, 57].

As described in Chapter II, taxonomy similarity is used in defining the performance measure *learning accuracy*, which is designed specifically for ontology-based information extraction. Hence, the figures in Tables 8 and 9 for precision that have been obtained using taxonomy similarity show the learning accuracy. In addition, we calculate recall and F1 measure using the figures provided by taxonomy similarity. The general idea behind taxonomy similarity is assigning a score between 0 and 1 for an extraction based on the closeness of the assigned class label to the correct class label in terms of the subsumption hierarchy of the ontology. For example, in the Mindswap ontology, this scheme assigns a score of 2/3 when the name of a *terrorist organization* is identified as the name of an *organization*. Under standard definitions no score will be awarded for this extraction. Here, we use these scores in calculating recall in addition to using them in calculating precision as advocated by learning accuracy. In situations where the same extraction is made for more than one class, it is assigned to the class that gives it the highest score for the purpose of calculating recall. The F1 measure is the harmonic mean between precision and recall as usual.

Note that S and T in Tables 8 and 9 denote the figures obtained using standard definitions and taxonomy similarity respectively.

TABLE 8: Results for Mindswap ontology in the single ontology systems

Scope	Precision (%)		Recall (%)		F1 (%)	
	S	T	S	T	S	T
Agent	26.40	31.73	34.74	41.75	30.00	36.06
Organization	54.79	59.36	26.14	28.32	35.39	38.35
All Classes	**36.87**	**41.92**	**29.44**	**33.47**	**32.74**	**37.22**

TABLE 9: Results for Mindswap ontology in the multiple ontology system

Scope	Precision (%)		Recall (%)		F1 (%)	
	S	T	S	T	S	T
Agent	29.27	32.72	50.53	56.49	37.07	41.44
Organization	36.67	51.94	28.76	40.74	32.24	45.66
All Classes	**32.39**	**40.85**	**37.10**	**46.77**	**34.59**	**43.61**

From the results shown in Tables 8 and 9, it can be seen that the multiple-ontology system has generally produced better results. It can also be seen that the multiple-ontology system has recorded a larger improvement for the "agent" class and its sub-classes than for the "organization" class and its subclasses. This means that the first mapping mentioned above has produced better results than the second mapping. This can be expected because the first mapping directly identifies names of victims whereas second mapping identifies the names of terrorist organizations and military organizations as names of organizations (which is the super class for the two types of organizations). Therefore, the advantage using the second mapping is made clear only when performance is measured using taxonomy similarity. It has resulted in a slight drop in F1 measure when standard definitions are used. Altogether, the precision has slightly dropped in the multiple-ontology system while the recall has improved by a larger margin. As a result, F1 measure has increased by about 2% when standard definitions are used and by about 6% when taxonomy similarity is used.

The figures for recall shown above represent local recall, since only the local ontology was considered in establishing the gold standard. Figures for global recall can be computed using the total number of facts available in the keys for the two ontologies as described in Chapter V. Table 10 shows these figures. The total figure for the single-ontology system is shown in order to provide a baseline for the multiple-ontology system. Since the two single-ontology systems are separate there is no such system that works on both ontologies.

The results for combining the outputs of information extractors developed under the OBCIE approach were consistent with the above results. As presented in Table 3 of Chapter IV, the performance measures for the "victim" class of the Mindswap ontology are as follows.

Precision = 33.71%, Recall = 44.36%, F1 = 38.31%

These figures were obtained using the platform for extraction rules. According to the first mapping shown above, the output of this information extractor can be combined with the output of the information extractor for the "human target" class of MUC4, which uses the two-phase classification platform. Following the operational principles of MOBIE, the union operator was used for this purpose. The performance measures changed as follows when this was done.

Precision = 31.84%, Recall = 58.65%, F1 = 41.27%

It can be seen that precision has dropped slightly while recall has improved by a larger margin resulting in a net gain of F1 measure by about 3%.

All the details of these two case studies including the OWL files for ontologies, text corpora, programs used for information extraction and full result sets are available from the project website[1].

[1] `http://aimlab.cs.uoregon.edu/obie/`

TABLE 10: Global recall for MUC4 and Mindswap ontologies

System	**MUC 4**	**Mindswap**	**Total**
Single-Ontology	20.09	17.26	37.35
Multiple-Ontology	20.09	21.75	41.84

6.3. Discussion

It can be seen that the multiple ontology-based information extraction system has shown a higher recall in both case studies. This supports our hypothesis that the use of multiple ontologies in ontology-based information extraction leads to a better recall. In terms of precision, the multiple-ontology system has recorded a higher figure when ontologies represent specialized sub-domains while a slightly lower figure has been observed when ontologies provide different perspectives. It can be hypothesized that the improvement in precision when ontologies represent specialized sub-domains is a result of the information extraction systems for the specialized ontologies being more accurate in their narrower domains than the information extraction system for the common ontology, which is designed for a broader domain. The slight drop in overall precision in the case study on terrorism data is mainly due to the drop of precision resulting from the mapping between *Organization* and *Perpetrator Organization* classes. Hence, it appears that the effect of the use of multiple ontologies on precision depends on the type of mappings used: if mappings are exact (directly between concepts), precision slightly improves whereas if the mappings are "rough" (based on subsumption hierarchy) the precision slightly deteriorates. Altogether, F1-measure has increased by significant amounts in both case studies mainly due to the improvement in recall. This represents the net benefit in using multiple ontologies.

In describing the operational principles of MOBIE in Chapter V, it was stated that they can be used even with systems that do not follow the OBCIE approach as long as the outputs can be clearly distinguished for for individual classes and properties. The importance of this point is highlighted by the case studies presented in this chapter where these principles are used to improve the performance of systems that do not follow the OBCIE approach. This means that the advantages of accommodating multiple ontologies in an information extraction system can be realized even for systems that do not follow the OBCIE approach.

CHAPTER VII

CONCLUSION

This dissertation presented two novel directions for the field of ontology-based information extraction. Firstly, it presented a component-based approach for information extraction designed through the use of ontologies in information extraction. The key concepts of this approach are decomposing an information extraction system along ontological concepts and separating domain, corpus and concept information from underlying information extraction techniques. Secondly, it showed how the information extraction process can be improved through the use of multiple ontologies in an information extraction system. For this purpose, a simple, generic definition was provided for a multiple ontology-based information extraction system and principles were designed for the interaction between ontologies in such systems. The concept of an information extractor, which is an independently deployable component that makes extractions with respect to a specific class or a property of an ontology, was used in defining the principles of multiple-ontology based information extraction.

This dissertation also presented the details of some case studies that have validated the hypotheses used in designing the component-based approach for information extraction and the principles for multiple ontology-based information extraction. With respect to the component-based approach, platforms for information extraction, which are domain, corpus and concept independent implementations of information extraction techniques, have been developed for the two dominant information extraction techniques, namely extraction rules and classification. Information extractors developed using these platforms have been

successfully reused, showing that the component-based approach is functional. With respect to multiple ontology-based information extraction, the principles designed for improving the information extraction process have been successfully applied in the two scenario for having multiple ontologies for the same domain, namely ontologies specializing in sub-domains and ontologies providing different perspectives on the entire domain.

7.1. Future Work

The research work presented in this dissertation can be extended in several directions. The following are the main directions we have identified.

Developing More Platforms

Although the use of machine learning techniques and extraction rules appear to be the two dominant information extraction techniques, several other information extraction techniques have been developed. As described in Chapter II, these include constructing partial parse trees and web-based search. Our component-based approach for information extraction can be made more effective by developing platforms for these techniques.

Unique challenges can be encountered in developing platforms for different information extraction techniques. This is mainly because a clear separation has to be made between the domain, corpus and concept specific information used by an information extraction technique and the underlying generic technique in developing platforms. Making this separation may not be straight-forward for some information extraction techniques such as constructing partial parse trees since the

same parse tree is used for making extractions with respect to different ontological concepts.

New Ways of Combining Information Extractors

Currently, we are only using the simple combination technique, which relies on the union operator, to combine different information extractors. We have recognized that different and probably more effective techniques can be designed for this purpose, for instance by using ensemble learning techniques, but have not thoroughly explored such techniques. This is related to the studies on using multiple ontologies in information extraction as well since the same techniques can be used to combine different information extractors when ontologies provide different perspectives on the entire domain.

Ensuring that the proposed techniques work in different situations may be a challenge in designing more effective combination techniques for information extractors. The use of the union operator has the advantage that there is some reasoning for its usage based on its potential to make more extractions. It may be necessary to come up with similar reasons for the use of more advanced combination techniques.

Handling Uncertainty of Mappings

Uncertainty associated with mappings, which are relationships between concepts of different ontologies, is currently gaining the attention of researchers because such uncertain mappings are generated in large numbers by automatic mapping discovery techniques. Manually verifying these mappings is not practical especially when ontologies in concern are large. How to make use of such uncertain

mappings in the component-based approach for information extraction as well in multiple ontology-based information extraction is an important research question.

One technique that can be used in this situation is validating the mappings using information extractors designed for the concepts in concern. As discussed in Chapter IV, the observation of very low performance measures when an information extractor is reused for a different concept can be construed as providing some evidence that the mapping in concern is incorrect. This idea can be extended into a mapping validation technique through the use of a training set. In addition, it may be possible to design special techniques to improve the results of the information extraction process when there is prior evidence that the mappings used for the reuse of information extractors are uncertain.

Developing Ontology Constructors

In the component-based approach for information extraction, we have focused our attention on the ontology population operation. Ontology construction operation, which is concerned with identifying classes and properties from text is an equally important task. Developing components for this task as well as investigating whether this process can be improved through the interaction between multiple ontologies are important research issues.

As described in Chapter III, the ontology construction process is normally carried out by using a set of high-level concepts often known as "seed concepts". Based on this idea, it may be possible to develop "platforms for ontology construction" analogous to the platforms we have used for the ontology population task. Such platforms for ontology construction will be domain, corpus and concept

independent implementations of ontology construction techniques and will have to be used together with some domain, corpus and concept specific information.

Scalability Issues

The designed component-based approach for information extraction as well as the principles for using multiple ontologies in information extraction are generic enough to be applied for any number of ontologies containing any number of concepts. However, practical problems can arise when the number of ontologies or the number of concepts in ontologies gets larger. For instance, the complexity of the aggregation task, performed by aggregator components under the OBCIE approach, will dramatically increase with an increase in the number of concepts in an ontology. Reuse of information extractors will also get more complex if a large number of information extractors have a mapping with the concept in concern. Discovering mappings between ontologies will also be more difficult in such situations. These scalability issues will have to be methodically handled in applying the OBCIE approach and the principles of multiple ontology-based information extraction with a large number of ontologies or with ontologies having a large number of concepts.

It should be noted that certain domains either have ontologies that contain a large number of concepts or a large number of ontologies. For instance, there are more than 30,000 classes in the well-known Gene Ontology [82] while Ghazvinian *et al.* have worked on the problem of discovering mappings among more than 200 ontologies for the domain of biomedicine [54]. This shows that the scalability issues mentioned above can occur in practice.

7.2. Concluding Remarks

Although there has been a surge in interest in information extraction, experts appear to have very different opinions on its future. Some are convinced that it will have a wide variety of applications while others suspect whether it would be good for anything. Such diametrically opposing views were expressed in a panel discussion in a leading conference in 2009[1]. Much of the criticisms of information extraction target its inflexibility and apparent brittleness. This situation shows the need to seriously look at different approaches for information extraction in addition to attempting to make incremental improvements in existing techniques.

We believe that the designed principles for a component-based approach for information extraction and multiple-ontology based information extraction amount to such new approaches for the field of information extraction. As such they should be considered the most important contributions of this dissertation.

[1] http://www.comp.polyu.edu.hk/conference/cikm2009/program/panels.htm

APPENDIX A

THE ALLOY SPECIFICATION

The following is the Alloy specification for an ontology-based information extraction system that uses a single ontology.

```
sig Document{}

sig Corpus{documents: some Document}

fact no_dangling_documents{
    all d:Document | some c:Corpus | d in c.documents
}

sig UnaryPredicate{}
sig BinaryPredicate{}
sig AssertionOnUnaryPredicate{}
sig AssertionOnBinaryPredicate{}
sig Axiom{}

sig ExtractedIndividuals{ext_individuals: set  AssertionOnUnaryPredicate}
fact{one ExtractedIndividuals}

sig ExtractedValues{ext_values: set  AssertionOnBinaryPredicate}
fact{one ExtractedValues}

sig ExtractedClasses{ext_classes: set  UnaryPredicate}
fact{one ExtractedClasses}

sig ExtractedProperties{ext_properties: set  BinaryPredicate}
fact{one ExtractedProperties}

sig ExtractedAxioms{ext_axioms: set  Axiom}
fact{one ExtractedAxioms}

sig Ontology{
    classes: set UnaryPredicate,
    properties: set BinaryPredicate,
    individuals: set  AssertionOnUnaryPredicate,
    values: set AssertionOnBinaryPredicate,
    axioms: set Axiom
}
```

```
fact no_dangling_ontology_elements{
(all a:UnaryPredicate | some o:Ontology | a in o.classes) and
(all a:BinaryPredicate | some o:Ontology | a in o.properties) and
(all a:AssertionOnUnaryPredicate | some o:Ontology |
        a in o.individuals) and
    (all a:AssertionOnBinaryPredicate | some o:Ontology |
        a in o.values)
    and (all a:Axiom | some o:Ontology | a in o.axioms)
}

pred populate(o, o':Ontology,
    c,c':Corpus, i:ExtractedIndividuals, v: ExtractedValues) {
    o'.individuals = o.individuals  + i.ext_individuals
    o'.values = o.values + v.ext_values
    o'.classes = o.classes
    o'.properties = o.properties
    o'.axioms = o.axioms
    c'.documents = c.documents
}

pred construct(o, o':Ontology,
    c,c':Corpus, l:ExtractedClasses, p:ExtractedProperties,
        a:ExtractedAxioms) {
    o'.individuals = o.individuals
    o'.values = o.values
    o'.classes = o.classes + l.ext_classes
    o'.properties = o.properties + p.ext_properties
    o'.axioms = o.axioms + a.ext_axioms
    c'.documents = c.documents
}
```

We use the following additional *predicates* to represent the refinement of `populate` operation into a series of operations under the OBCIE approach.

```
sig PreprocessedDocument{}

sig PreprocessedCorpus{documents: some PreprocessedDocument}

pred preprocess(o, o':Ontology, c, c':Corpus, pc: PreprocessedCorpus){
    o'.individuals = o.individuals
```

```
    o'.values = o.values
    o'.classes = o.classes
    o'.properties = o.properties
    o'.axioms = o.axioms
    c'.documents = c.documents
    #c.documents = #pc.documents
}

pred extract(o, o':Ontology, c, c':Corpus, i, i':ExtractedIndividuals,
    v, v':ExtractedValues){
    o'.individuals = o.individuals
    o'.values = o.values
    o'.classes = o.classes
    o'.properties = o.properties
    o'.axioms = o.axioms
    c'.documents = c.documents
    i'.ext_individuals = i.ext_individuals
    v'.ext_values = v.ext_values
}

pred format(o, o':Ontology, c, c':Corpus, i, i':ExtractedIndividuals,
    v, v':ExtractedValues){
    o'.individuals = o.individuals
    o'.values = o.values
    o'.classes = o.classes
    o'.properties = o.properties
    o'.axioms = o.axioms
    c'.documents = c.documents
    #i'.ext_individuals = #i.ext_individuals
    #v'.ext_values = #v.ext_values
}

pred aggregate(o, o':Ontology, c, c':Corpus,  i,
     i':ExtractedIndividuals, v, v':ExtractedValues){
    o'.individuals = o.individuals + i'.ext_individuals
    o'.values = o.values + v'.ext_values
    o'.classes = o.classes
    o'.properties = o.properties
    o'.axioms = o.axioms
    c'.documents = c.documents
}
```

Now, the consistency of the refinement of the `populate` operation can be checked using the following *assertion*.

```
assert populateRefinementCheck{
    all o,o',o1,o2,o3,o4:Ontology, c,c',c1,c2,c3,c4:Corpus,
    i,i1,i2,i3:ExtractedIndividuals, v,v1,v2,v3:ExtractedValues,
    pc:PreprocessedCorpus | populate[o,o',c,c',i,v] &&
    preprocess[o, o1, c, c1, pc] &&
    extract[o1, o2, c1, c2, i1, i2, v1, v2] &&
    format[o2, o3, c2, c3, i2,i3, v2, v3] &&
    aggregate[o3, o4, c3, c4, i3, i, v3, v] implies
        o'.individuals = o4.individuals and
        o'.values = o4.values and
        o'.properties = o4.properties and
        o'.axioms = o4.axioms and
        o'.classes = o4.classes  and
        c'.documents = c4.documents
}

check populateRefinementCheck for 5
```

When Alloy Analyzer is used to check the consistency of this assertion, it reports that no counter-examples were found for up to 5 instances. This is good enough evidence that the refinement is correct.

The specifications presented above can be extended to the multiple ontology case by using a set of signatures to represent different ontologies and a set of predicates for each predicate used. This is the same approach adopted in extending the Z specifications for the multiple-ontology case.

APPENDIX B

THE XML SCHEMATA

B.1. The XML Schema for Platforms

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="http://aimlab.cs.uoregon.edu/obie/StandardXMLSchema/1.0/"
targetNamespace=
"http://aimlab.cs.uoregon.edu/obie/StandardXMLSchema/1.0/"
elementFormDefault="qualified">

<xs:element name="Platform">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Name" type="xs:string"/>
      <xs:element name="Description" type="xs:string"/>
      <xs:element name="FeatureMeaning" type="xs:string"/>
      <xs:element name="Executable" type="xs:anyURI"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

B.2. The XML Schema for Metadata

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="http://aimlab.cs.uoregon.edu/obie/StandardXMLSchema/1.0/"
targetNamespace=
"http://aimlab.cs.uoregon.edu/obie/StandardXMLSchema/1.0/"
elementFormDefault="qualified">

<xs:element name="InformationExtractor">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="Language" type="xs:string"/>
      <xs:element name="IETechnique" type="xs:anyURI"/>
      <xs:element name="Concept" type="xs:string"/>
      <xs:element name="ConceptType">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:enumeration value="Class"/>
            <xs:enumeration value="Property"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
      <xs:element name="IdentifierName" type="xs:string"/>
      <xs:element name="Feature" type="xs:string"
        minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

REFERENCES CITED

[1] Alloy Community.
http://alloy.mit.edu/community/.

[2] Apache UIMA.
http://uima.apache.org/.

[3] BioCreative - Critical Assessment for Information Extraction in Biology.
http://biocreative.sourceforge.net/.

[4] BioNLP'09 Shared Task on Event Extraction.
http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/SharedTask/.

[5] DAML Ontology Library.
http://www.daml.org/ontologies/.

[6] Dublin Core Metadata Initiative.
http://dublincore.org/.

[7] FOLDOC: Free On-Line Directory of Computing.
http://foldoc.org/.

[8] General Architecture for Text Engineering (GATE).
http://www.gate.ac.uk/.

[9] General Formal Ontology (GFO).
http://www.onto-med.de/ontologies/gfo/.

[10] GermaNet.
http://www.sfs.uni-tuebingen.de/GermaNet/.

[11] Hindi Wordnet.
http://www.cfilt.iitb.ac.in/wordnet/webhwn/.

[12] Mindswap Terrorism Ontology.
http://www.mindswap.org/2003/owl/swint/terrorism.

[13] OBO-Edit: The OBO Ontology Editor.
http://oboedit.org/.

[14] OntoStudio.
http://www.ontoprise.de/en/home/products/ontostudio/.

[15] OpenNLP.
http://opennlp.sourceforge.net/.

[16] OWL Web Ontology Language.
http://www.w3.org/TR/owl-ref/.

[17] Pellet: OWL 2 Reasoner for Java.
http://clarkparsia.com/pellet/.

[18] Phenotypic Quality Ontology.
http://www.bioontology.org/wiki/ index.php/PATO:Main_Page.

[19] The Protégé Ontology Editor and Knowledge Acquisition System.
http://protege.stanford.edu/.

[20] RDF Schema.
http://www.w3.org/TR/rdf-schema/.

[21] RDF/XML Syntax Specification.
http://www.w3.org/TR/REC-rdf-syntax/.

[22] Resource Description Framework.
http://www.w3.org/RDF/.

[23] SAIC Information Extraction.
http://www.itl.nist.gov/iaui/894.02/related_projects/muc/.

[24] Simple HTML Ontology Extensions (SHOE) University Ontology.
http://www.cs.umd.edu/projects/plus/SHOE/onts/univ1.0.html.

[25] SPARQL Query Language for RDF.
http://www.w3.org/TR/rdf-sparql-query/.

[26] SProUT (Shallow Processing with Unification and Typed Feature Structures).
http://sprout.dfki.de/.

[27] SUMO: Suggested Upper Merged Ontology.
http://www.ontologyportal.org/.

[28] Swoogle Semantic Web Search Engine.
http://swoogle.umbc.edu/.

[29] The Cyc ontology project.
http://www.cyc.com/.

[30] The DARPA Agent Markup Language.
http://www.daml.org/.

[31] The Java OWL API.
http://owlapi.sourceforge.net/index.html.

[32] The Stanford Natural Language Processing Group.
`http://nlp.stanford.edu/software/index.shtml`.

[33] Unified Modeling Language.
`http://www.uml.org/`.

[34] Y. An, A. Borgida, R. J. Miller, and J. Mylopoulos. A semantic approach to discovering schema mapping expressions. In *ICDE*, pages 206–215, 2007.

[35] D. E. Appelt, J. R. Hobbs, J. Bear, D. J. Israel, and M. Tyson. FASTUS: A finite-state processor for information extraction from real-world text. In *IJCAI*, pages 1172–1178, 1993.

[36] N. Ashish and C. A. Knoblock. Wrapper generation for semi-structured internet sources. *SIGMOD Record*, 26(4):8–15, 1997.

[37] M. Attik. Using ensemble feature selection approach in selecting subset with relevant features. In *ISNN (1)*, pages 1359–1366, 2006.

[38] M. Banko, M. J. Cafarella, S. Soderland, O. Etzioni, and M. Broadhead. Open information extraction from the web. In *IJCAI*, pages 2670–2676, 2007.

[39] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284(5), May 2001.

[40] C. Bizer, T. Heath, and T. Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.

[41] J. Bona and S. C. Shapiro. SNePS as an ontological reasoning tool. In *Proceedings of the International Conference on Biomedical Ontologies (ICBO)*, page 160, 2009.

[42] P. Buitelaar and M. Siegel. Ontology-based information extraction with SOBA. In *LREC*, pages 2321–2324, 2006.

[43] N. Choi, I.-Y. Song, and H. Han. A survey on ontology mapping. *SIGMOD Rec.*, 35(3):34–41, 2006.

[44] P. Cimiano, S. Handschuh, and S. Staab. Towards the self-annotating web. In *WWW*, pages 462–471, 2004.

[45] P. Cimiano, G. Ladwig, and S. Staab. Gimme' the context: context-driven automatic semantic annotation with C-PANKOW. In *WWW*, pages 332–341, 2005.

[46] F. Ciravegna. Adaptive information extraction from text by rule induction and generalisation. In *IJCAI'01*, pages 1251–1256. Morgan Kaufmann, 2001.

[47] T. Declerck, C. Federmann, B. Kiefer, and H.-U. Krieger. Ontology-based information extraction and reasoning for business intelligence applications. In *KI*, pages 389–390, 2008.

[48] O. Dekel, J. Keshet, and Y. Singer. Large margin hierarchical classification. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 27. ACM, 2004.

[49] A. Doan, P. Domingos, and A. Y. Halevy. Reconciling Schemas of Disparate Data Sources: A Machine-Learning Approach. In *Proceedings of the ACM Conference on Management of Data*, 2001.

[50] X. Dong, A. Y. Halevy, and J. Madhavan. Reference reconciliation in complex information spaces. In *SIGMOD Conference*, pages 85–96, 2005.

[51] T. Q. Dung and W. Kameyama. Ontology-based information extraction and information retrieval in health care domain. In *DaWaK*, pages 323–333, 2007.

[52] D. W. Embley. Toward semantic understanding: an approach based on information extraction ontologies. In *ADC*, pages 3–12, 2004.

[53] C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[54] A. Ghazvinian, N. F. Noy, C. Jonquet, N. H. Shah, and M. A. Musen. What four million mappings can tell you about two hundred ontologies. In *International Semantic Web Conference*, pages 229–242, 2009.

[55] T. Gruber. Ontolingua: A translation approach to providing portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

[56] L. M. Haas, M. A. Hernandez, H. Ho, L. Popa, and M. Roth. Clio Grows Up: From Research Prototype to Industrial Tool. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, pages 805–810, 2005.

[57] U. Hahn and K. Schnattinger. Towards text knowledge engineering. In *AAAI '98/IAAI '98*, pages 524–531, 1998.

[58] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

[59] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*, pages 614–628. Morgan Kaufmann Publishers, second edition, 2006.

[60] M. S. Hanif and M. Aono. Alignment results of Anchor-Flood algorithm for OAEI-2008. In *OM*, 2008.

[61] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics*, pages 539–545, Morristown, NJ, USA, 1992. Association for Computational Linguistics.

[62] J. R. Hobbs, M. E. Stickel, D. E. Appelt, and P. A. Martin. Interpretation as abduction. *Artif. Intell.*, 63(1-2):69–142, 1993.

[63] I. Horrocks. Ontologies and the semantic web. *Commun. ACM*, 51(12):58–67, 2008.

[64] W. Hu and Y. Qu. Falcon-ao: A practical ontology matching system. *J. Web Sem.*, 6(3):237–239, 2008.

[65] C. H. Hwang. Incompletely and imprecisely speaking: Using dynamic ontologies for representing and retrieving information. In *KRDB*, pages 14–20, 1999.

[66] J. Kietz, A. Maedche, and R. Volz. A method for semi-automatic ontology acquisition from a corporate intranet. *EKAW-2000 Workshop "Ontologies and Text"*, 2000.

[67] A. H. F. Laender, B. A. Ribeiro-neto, A. S. da Silva, and J. S. Teixeira. A brief survey of web data extraction tools. *SIGMOD Record*, 31:84–93, 2002.

[68] C. Li. Classifying imbalanced data using a bagging ensemble variation (BEV). In *ACM-SE 45*, pages 203–208, 2007.

[69] Y. Li and K. Bontcheva. Hierarchical, perceptron-like learning for ontology-based information extraction. In *WWW*, pages 777–786, 2007.

[70] Y. Li, K. Bontcheva, and H. Cunningham. Using uneven margins svm and perceptron for information extraction. In *In Proceedings of Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, 2005.

[71] A. Maedche, B. Motik, and L. Stojanovic. Managing multiple and distributed ontologies on the semantic web. *The VLDB Journal*, 12(4):286–302, 2003.

[72] A. Maedche, G. Neumann, and S. Staab. Bootstrapping an ontology-based information extraction system. *Intelligent exploration of the web*, pages 345–359, 2003.

[73] A. Maedche and S. Staab. Discovering conceptual relations from text. In *ECAI*, pages 321–325, 2000.

[74] A. Maedche and S. Staab. The Text-To-Onto Ontology Learning Environment. *Software Demonstration at ICCS-2000-Eight International Conference on Conceptual Structures. August*, pages 14–18, 2000.

[75] A. Mannes and J. Golbeck. Building a terrorism ontology. Technical report, MINDSWAP,University of Maryland, 2005.

[76] D. Maynard. Metrics for evaluation of ontology-based information extraction. In *In WWW 2006 Workshop on Evaluation of Ontologies for the Web*, 2006.

[77] D. Maynard, W. Peters, and Y. Li. Metrics for evaluation of ontology-based information extraction. In *WWW 2006 Workshop on Evaluation of Ontologies for the Web (EON)*, 2006.

[78] A. K. McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[79] L. McDowell and M. J. Cafarella. Ontology-driven information extraction with ontosyphon. In *International Semantic Web Conference*, pages 428–444, 2006.

[80] M.-F. Moens. *Information Extraction: Algorithms and Prospects in a Retrieval Context (The Information Retrieval Series)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[81] H. M. Müller, E. E. Kenny, and P. W. Sternberg. Textpresso: an ontology-based information retrieval and extraction system for biological literature. *PLoS Biology*, 2(11), November 2004.

[82] G. Ontology Consortium. Creating the Gene Ontology Resource: Design and Implementation. *Genome Research*, 11(8):1425–1433, 2001.

[83] H. Poon and P. Domingos. Unsupervised ontology induction from text. In *ACL '10: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 296–305. Association for Computational Linguistics, 2010.

[84] B. Popov, A. Kiryakov, D. Ognyanoff, D. Manov, and A. Kirilov. KIM – a semantic platform for information extraction and retrieval. *Nat. Lang. Eng.*, 10(3-4):375–392, 2004.

[85] E. Riloff. Information extraction as a stepping stone toward story understanding. *Understanding language understanding: computational models of reading*, pages 435–460, 1999.

[86] J. J. Ritsko and D. I. Seidman. Preface. *IBM Systems Journal*, 43(3):449–450, 2004.

[87] R. Romano, L. Rokach, and O. Maimon. Automatic discovery of regular expression patterns representing negated findings in medical narrative reports. In *NGITS*, pages 300–311, 2006.

[88] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition edition, pages 848-850, 2003.

[89] H. Saggion, A. Funk, D. Maynard, and K. Bontcheva. Ontology-based information extraction for business intelligence. In *ISWC/ASWC*, pages 843–856, 2007.

[90] J. M. Spivey. *The Z notation: a reference manual*. Prentice Hall International (UK) Ltd., Hertfordshire, UK, 1992.

[91] R. Studer, V. R. Benjamins, and D. Fensel. Knowledge engineering: principles and methods. *Data Knowledge Engineering*, 25(1-2):161–197, 1998.

[92] B. M. Sundheim and N. A. Chinchor. Survey of the message understanding conferences. In *HLT '93: Proceedings of the workshop on Human Language Technology*, pages 56–60, Morristown, NJ, USA, 1993. Association for Computational Linguistics.

[93] C. Szyperski. *Component Software*. Addison-Wesley Professional. pages xv-xix, 3-12, 35-47, 2002.

[94] A. Todirascu, L. Romary, and D. Bekhouche. Vulcain - an ontology-based information extraction system. In *NLDB '02: Proceedings of the 6th International Conference on Applications of Natural Language to Information Systems-Revised Papers*, pages 64–75. Springer-Verlag, 2002.

[95] D. Tsarkov and I. Horrocks. FaCT++ description logic reasoner: System description. In U. Furbach and N. Shankar, editors, *Automated Reasoning*, volume 4130 of *Lecture Notes in Computer Science*, pages 292–297. Springer Berlin / Heidelberg, 2006.

[96] M. Vargas-vera, E. Motta, J. Domingue, S. B. Shum, and M. Lanzoni. Knowledge extraction by using an ontology-based annotation tool. In *In K-CAP 2001 workshop on Knowledge Markup and Semantic Annotation*, pages 5–12, 2001.

[97] H. Wang, S. Liu, and L.-T. Chia. Does ontology help in image retrieval?: a comparison between keyword, text ontology and multi-modality ontology approaches. In *ACM Multimedia*, pages 109–112, 2006.

[98] D. S. Weld, R. Hoffmann, and F. Wu. Using wikipedia to bootstrap open information extraction. *SIGMOD Rec.*, 37(4):62–68, 2008.

[99] Y. Wilks and C. Brewster. Natural language processing as a foundation of the semantic web. *Foundations and Trends in Web Science*, 1(3-4):199–327, 2009.

[100] F. Wu, R. Hoffmann, and D. S. Weld. Information extraction from Wikipedia: moving down the long tail. In *KDD*, pages 731–739. ACM, 2008.

[101] F. Wu and D. S. Weld. Autonomously semantifying Wikipedia. In *CIKM*, pages 41–50. ACM, 2007.

[102] F. Wu and D. S. Weld. Automatically refining the Wikipedia infobox ontology. In *WWW*, pages 635–644, 2008.

[103] B. Yildiz and S. Miksch. ontox - a method for ontology-driven information extraction. In *ICCSA (3)*, pages 660–673, 2007.