An Ontology-based Knowledge Support System for Requirements Analysis

A thesis presented
in partial fulfilment of the requirements
for the degree of
Doctor of Philosophy
in Computer Science
at Massey University, Manawatū,
New Zealand

Gang (Jason) Liu

2013

Abstract

An Ontology-based Knowledge Support System for Requirements Analysis (OKSSRA) is proposed and developed, in order to help requirements analysts obtain the preliminary business knowledge.

Requirements Engineering (RE) is a sub-discipline of Software Engineering (SE). It is involved in the whole software life cycle from the very first step throughout the process of software development. Thus, the performances of requirements analysts are crucial to RE outcomes since requirements analysts bridge the communication gap between business stakeholders and development teams.

However, normally, there is a knowledge gap between requirements analysts and business stakeholders, especially when analysts work for out-sourcing contractors. The existence of this knowledge gap may seriously lower analysts' efficiency of communicating with business stakeholders and hinder their performances on preparing requirements documentation. Obviously, preliminary business knowledge related to the project will help analysts to narrow down the knowledge gap and improve their performances. However, based on our survey, there is no existing RE tools providing such knowledge support to analysts. Therefore, we proposed and developed OKSSRA to help analysts obtain the preliminary business knowledge for narrowing down the knowledge gap.

There are three key modules in OKSSRA: (i) a semantic similarity measure module, (ii) an ontology mapping module, and (iii) an automatic use case generating module.

In the semantic similarity measure module of OKSSRA, we proposed and developed a new semantic similarity measure utilising WordNet and

Normalised Google Distance (NGD). In the new measure, NGD will be used to calculate a unique length for each edge in the shortest path between two candidate concepts in the WordNet graph. The semantic similarity measure enables our system (i) to assign related concepts for a user's queries to extend the queries; and (ii) to identify the related business processes from the business knowledge repository.

The ontology mapping module of OKSSRA employs a newly developed ontology mapping method based on MIMapper (Kaza and Chen 2008). In this new ontology mapping method, our newly proposed semantic similarity measure will be used to matching class names and to locate the most informative instances of their class. The ontology mapping module enables our system (i) to update the ontology-based repositories in the system, and (ii) to integrate the ontology-based repositories with other repositories.

In the OKSSRA module for automatically generating Use Cases, we propose a set of mapping rules for the system to automatically generating Use Cases based on the information retrieved from business processes. The set of mapping rules specified how the components of a business process are transformed into use case elements, e.g., actors, goals, and steps of scenarios. With this module, our system is able to generate essential Use Cases automatically using business processes retrieved from MIT Process Handbook.

A set of three test use scenarios and a questionnaire has been carefully designed to evaluate the efficiency and effectiveness of OKSSRA. The experimental results show that (i) our system is useful for obtaining business knowledge, (ii) our system is more effective than existing system developed for similar purpose, and (iii) our system is able to provide a pleasant user experiences.

Dedicated to my parents

for their everlasting love, encouragement and support

Acknowledgement

I would like to take this opportunity to express my appreciation and gratitude to those people who have supported me to achieve this qualification.

My first sincere thanks go to my supervisor, Dr. Ruili Wang, for his invaluable guidance and tremendous support throughout this research. Without his tireless directions and continuing encouragement, it would have been unfeasible for me to achieve my PhD degree. His enthusiasm, support and encouragement have influenced me far beyond this research.

I also wish to express my deep gratitude to my co-supervisors Dr. Jeremy Buckley, Dr. Dave Parry, Dr. Helen Zhou, and A/Prof. Elizabeth Kemp, for the time and effort they have spent with me, during my PhD study. I appreciate their valuable suggestions and constructive comments.

I thank all the participants for their help in the evaluations. Thanks to Jingli, Frank, and Ali, and other friends at Massey University, for their valuable support and friendship.

I gratefully acknowledge the funding from the Foundation for Research, Science and Technology towards my PhD study.

Last but not least, my special thanks go to my parents for their everlasting support, understanding and encouragement.

Contents

List of Figures

List of Tables

Chapter 1.

Introduction and Scope

1.1 Introduction

In this research, an Ontology-based Knowledge Support System for Requirements Analysis (OKSSRA) is proposed and developed to help requirements analysts gain preliminary business knowledge. A requirements analyst has an important role in a software project team. They are in charge of eliciting business requirements from business stakeholders and composing the requirements documentation for developing software (Salem and Darter 2006). Business knowledge is crucial for requirements analysts to perform their function well and produce requirements documentation of high quality (Kaiya and Saeki 2006).

Requirements Engineering (RE) is a sub-discipline of Software Engineering (SE). It is involved in the whole software life cycle from the very first step throughout the process of software development. Activities of RE include identifying stakeholders, eliciting customers' and users' expectations toward software products, documenting and analysing requirements, and verifying and validating requirements (Robertson and Robertson 2006). Requirements analysts are involved in the whole process of RE. Various obstacles may hinder requirements analysts' performance. One of the most common obstacles is the knowledge gap between Software Developers (SD) and business stakeholders.

The existence of this knowledge gap between SD and business stakeholders is well known (Al-Karaghouli et al. 2000 and Merunka 2006). The business

stakeholders include sponsors and users. The sponsors are the people paying for the system, while the users are the people who will actually use the system.

The knowledge gap between SD and business stakeholders is a serious hindrance for requirements analysts working in outsourced SD environment. There are usually two kinds of SD: in-house and outsourcing. The in-house SD and business stakeholders belong to the same company while the outsourcing SD normally belong to a different organisation from that of the business stakeholders. The project manager of an in-house project can act as a domain expert who may have sufficient knowledge for both business area and technical area. She or he can ensure the correctness and efficiency of the communication between in-house SD and business stakeholders.

However, it is difficult to find a domain expert for an outsourcing project in either the customer's organisation or outsourcing contractor's organisation (Robertson and Robertson 2006). Requirements analysts working in outsourced projects are usually unfamiliar with the business background and do not have the requisite business knowledge (Kaiya and Saeki 2006). Furthermore, a common problem of outsourcing projects is that the business stakeholders only have a vague idea about the system-to-be (McConnell 1996). As a result, it is difficult for them to get their ideas through to the requirements analysts (Pouloudi 1999).

Therefore, for requirements analysts (particularly for an outsourced project), the knowledge gap between the SD and business stakeholders can ultimately reduce the level of their performance, and which may lead to requirements documentation of poor quality.

The success of a software product highly depends on the quality of the requirements documentation (Wiegers 2003). In the course of developing

software, developers directly use requirements documentation (including software specifications) as guideline for constructing a software product.

Recognising the importance of requirements documentation, numerous Computer Aided Requirements Engineering (CARE) tools have been developed to provide assistance in whole or in part of the process of preparing and composing requirements documentation. However, according to our survey of existing RE tools (please refer to Section 2.2), no such CARE tool has been specially developed for requirements analysts to gain preliminary business background and relevant knowledge.

Business knowledge repositories can be utilised to help requirements analysts gain preliminary business knowledge. In order to narrow down the knowledge gap, requirements analysts will normally obtain preliminary information and business knowledge with regard to their project in advance. One way to obtain the business process knowledge is from the MIT Process Handbook (MITPH). MITPH is a business knowledge repository built by MIT Business School. It has long been utilised as a knowledge source by business analysts for gaining explicit understanding of business processes.

Therefore, in this research, in order to help requirements analysts gain preliminary business knowledge, we propose and develop an Ontology-based Knowledge Support System for Requirements Analysis (OKSSRA).

Requirements analysts can use OKSSRA to (i) gain preliminary business knowledge for narrowing down the knowledge gap between them and business stakeholders, and (ii) obtain (essential) Use Cases automatically generated on the basis of business processes retrieved from MITPH.

1.2 Scope of this thesis

This thesis is organised as follows:

Chapter 2: *Motivation and Research Objectives*. In this chapter, we illustrate the importance for requirements analysts to have preliminary business knowledge. We then propose OKSSRA, which is aimed to help requirements analysts gain preliminary business knowledge. The research issues of OKSSRA are presented along with proposed solutions.

Chapter 3: *Semantic Similarity Measure Module*. In this chapter, we investigate previous research on semantic similarity measurement. We then propose a new semantic similarity measure, which take advantages of the information retrieved from semantic networks (e.g., WordNet) and the Internet. The Normalised Google Distance (NGD) is employed to extract Internet knowledge for calculating the length of the shortest path in WordNet between two words of interest. NGD is a similarity measure using the statistical information of Google search results to determine the semantic relatedness between words. The new measurement is evaluated against existing measurements and the results are presented.

Chapter 4: *Ontology Mapping Module*. In this chapter, a review of previous work on ontology mapping is presented. A new ontology mapping method is then proposed utilising: (i) our newly proposed semantic similarity measure to perform linguistic based matching, and (ii) NGD to conduct instance based matching. The new ontology mapping method is evaluated against existing mapping methods and the results are presented.

Chapter 5: *Automatic Use Case Generating Module*. In this chapter, we investigate previous research on generating Use Cases from business knowledge. We then propose a new set of mapping rules to allow OKSSRA generate Use Cases automatically from the business processes retrieved from

the MITPH. The evaluation experiments and the results of our automatically generated Use Cases are presented.

Chapter 6: *System Evaluation*. In order to assess the effectiveness and efficiency of OKSSRA on helping junior requirements analysts gain preliminary business knowledge, we design a test use of the system and a questionnaire sheet for evaluating OKSSRA. The results of the evaluation are presented, followed by the discussion of possible improvements of the system.

Chapter 7: *Conclusion and Future Work*. In this chapter, we summarise and conclude the main findings of this research. We also outline the contribution this research has made and a discussion of possible future research directions.

Chapter 2.

Motivation and Research Objectives

In this chapter, we discuss the reasons why we focus our research on helping requirements analysts gain preliminary business knowledge, and define our research objectives.

In Section 2.1, we demonstrate the significance and difficulties for helping requirements analysts gain preliminary business knowledge. Section 2.2 proposes an Ontology-based Knowledge Support System for Requirements Analysis (OKSSRA) to help requirements analyst gain business knowledge. The research issues and proposed solutions for developing OKSSRA are also presented in Section 2.2. Section 2.3 summaries this chapter.

2.1 Motivations

Requirements Engineering (RE) is a sub-discipline of software engineering that is responsible for the derivation of software requirements. Generally, there are four phases in RE: (i) stakeholder identification, (ii) requirements elicitation, (iii) analysis and documentation of the requirements, and (iv) verification and validation of the requirements. Requirements analysts are involved in all the four RE phases. Related business knowledge is important for improving the requirements analysts' performance throughout the RE process, especially during the second and third phases.

2.1.1 Importance of business knowledge for requirements analysts

Improve requirements elicitation

Requirements Elicitation involves the gathering of requirements from business stakeholders in order to define the functionality to be delivered by the system under development (Wiegers 2003). The activities include identifying business stakeholders, eliciting business stakeholders' motivations, objectives and goals and modelling initial requirements to get more complete and detailed specifications.

Understanding the business context can help requirements analysts systematically identify business stakeholders under complex circumstances. Both Pouloudi (1999) and Sharp et al. (1999) believe that this, along with understanding the interactions between stakeholders, is useful in the identification of business stakeholders, especially those that are not obvious.

Business knowledge is considered to be useful for requirements analysts to thoroughly collect business stakeholders' expectations of the system as initial requirements. Aoyama (2005) suggests that understanding and modelling the interacting between crucial users can help identify unknown users, and also assist users organise their thoughts and ideas purposefully.

Improve quality of requirements documentation

There are two types of defects in requirements documentation: defects of presentation and defects of logic (Nuseibeh and Easterbrook 2000).

Presentation defects are mostly language related, such as ambiguity and inconsistency. Berry and Kamsties (2004), Fantechi et al. (2002), Kaiya and Saeki (2006), Sawyer et al. (2006), and Wasson (2006) studied how to locate

ambiguities utilising different support mechanisms, such as ontologies, linguistic techniques, and language enhancement.

With preliminary business knowledge, the requirements analyst can gain a better understanding of the business context. They will be able to address requirements in a more tenable way with a correct and unambiguous understanding of those requirements. As a result, both ambiguity and inconsistency may be reduced from requirements documentation.

Logic defects include unexpected interactions among requirements, barrier of requirements satisfaction, and absent assumption.

With knowledge support from a business knowledge repository, requirements analysts will be able to have a better chance of discovering possible situations that may be encountered by the system. As a result, there will be less chance for analysts missing assumptions.

Thus, we believe that a knowledge support system will be able to increase the requirements analysts' level of performances through the various RE activities by helping them better understand the related business context.

2.1.2 Review of current Requirements Engineering tools

In this section, we briefly review nine existing requirements engineering tools (The detailed reviews are presented in Appendix C). These tools are developed by various vendors or educational organisations. We try to review every well-known requirements engineering tool that we can gain access to. However, some of the more well-known tools, such as Accept 360°, C.A.R.E and Blueprint, have not been included in this section, since these tools do not have free trial version.

The nine requirements engineering tools that we have reviewed are: (i) Caliber DefineIT, (ii) Caliber RM, (iii) IBM Rational RequisitePro, (iv)DOORS, (v) GatherSpace, (vi) Accompa, (vii) Statestep, (viii) Tiger Pro, and (ix)WIBNI.

Caliber DefineIT helps requirements analysts to communicate with business stakeholders to obtain their expectations of the software being developed. It acts as a "digital white board" to improve efficiency of the discussion between analysts and business stakeholders, and to help elicit requirements from the resulting discussions.

CaliberRM has features to help development teams communicate and agree on requirements. It also provides a repository to store resultant requirements and related information. In addition it provides tools to support the assessment of the progress and to estimate the cost of the project.

RequisitePro has similar features for eliciting requirements as CaliberRM does. It provides better support than CaliberRM on extracting requirements from different file formats, such as MSWord and CSV documents. RequisitePro can also correlate requirements documentation stored in its database. However, it does not have similar tools for managing project as CaliberRM does.

Similar to CaliberRM and RequisitePro, *DOORS* provides wide range of features for requirements elicitation. A distinguishing feature of DOORS is that it provides a C-like description language called DXL. With this feature, DOORS provides more flexibility to fit itself into what customers really need.

GatherSpace is an internet browser based online requirements management tool. It provides support for relatively simple tasks such as gathering requirements and drafting Use Cases. GatherSpace can also automatically generate reports based on store requirements.

Accompa is also a light weight online requirements management tool similar to GatherSpace. It allows users to upload requirements documentation, which is a function that GatherSpace does not support. However Accompa cannot generate reports from the stored requirements.

Statestep is a RE tool using a formal method to deduct requirements and discover inconsistencies.

Tiger Pro can help users generate requirements using a pre-determined structure to reduce ambiguities that may occur in natural language. It also examines requirements using light weight natural language processing methods for the purpose of reducing ambiguities.

WIBNI is a light weight requirements organising tool based on Microsoft Access. It allows users to assign attributes to requirements, such as priority, type, and dependency. Similar to Accompa, WIBNI can also generate reports on stored requirements.

The features of the nine RE tools presented above are summarised in Table 2.1. From Table 2.1, we can see that none of nine reviewed tools provides knowledge support for requirements analysts to smooth the process of eliciting requirements. Most of the nine RE tools provide support for composing requirements and have a database for the purpose of storing requirements and the properties of requirements, such as, attribute values, change history, and related Use Cases.

Only four of the nine tools support the communication between analysts and business stakeholders, and provide help on easing this communication. However, there is little support offered by these tools to help analysts to ask the right questions, so that they can get all business stakeholders' expectations on the software in an efficient and effective way. We believe that a knowledge

Table 2. 1 The functionalities of reivewed RE tools

Attributes \ Software	Caliber DefineIT	CaliberRM	IBM RequisitePro	DOORS	GatherSpace	Accompa	Statestep	Tiger Pro	WIBNI
Providing assistances on requirements elicitation	Y	Y	Y	Y					
Providing knowledge support for initial requirements elicitation									
Possessing a requirements repository		Y	Y	Y	Y	Y			Y
Supporting requirements analysis	Y	Y	Y	Y			Y	Y	
Tracing and analysing requirements changes		Y	Y	Y					
Configuring requirements		Y	Y	Y	Y	Y		Y	Y
Groupware		Y	Y	Y	Y	Y			
Allowing importing documents		Y	Y	Y	Y	Y		Y	Y
Ability of integrating with other tools		Y	Y	Y					
Summarising and generating reports		Y	Y	Y	Y			Y	Y

support system providing preliminary business knowledge to requirements analysts will fill this gap and help them elicit requirements in a more efficient and effective way, and, ultimately, improve the quality of resultant requirements documentation.

2.2 Ontology-based Knowledge Support System for Requirements Analysis (OKSSRA)

Considering the importance for requirements analysts to have business knowledge and the lack of RE tools that help analysts gain business

knowledge, we propose an Ontology-based Knowledge Support System for Requirements Analysis (OKSSRA). This system aims to help junior requirements analysts to gain preliminary business knowledge efficiently and effectively so that the analyst can improve their performance and enhance the quality of requirements documentation.

2.2.1 High level system structure

OKSSRA has five components, as illustrated in Figure.2.1: (i) an ontology-based repository of business knowledge, (ii) a Use Cases repository, (iii) a general knowledge repository, (iv) a deductive engine, and (v) a user interface.

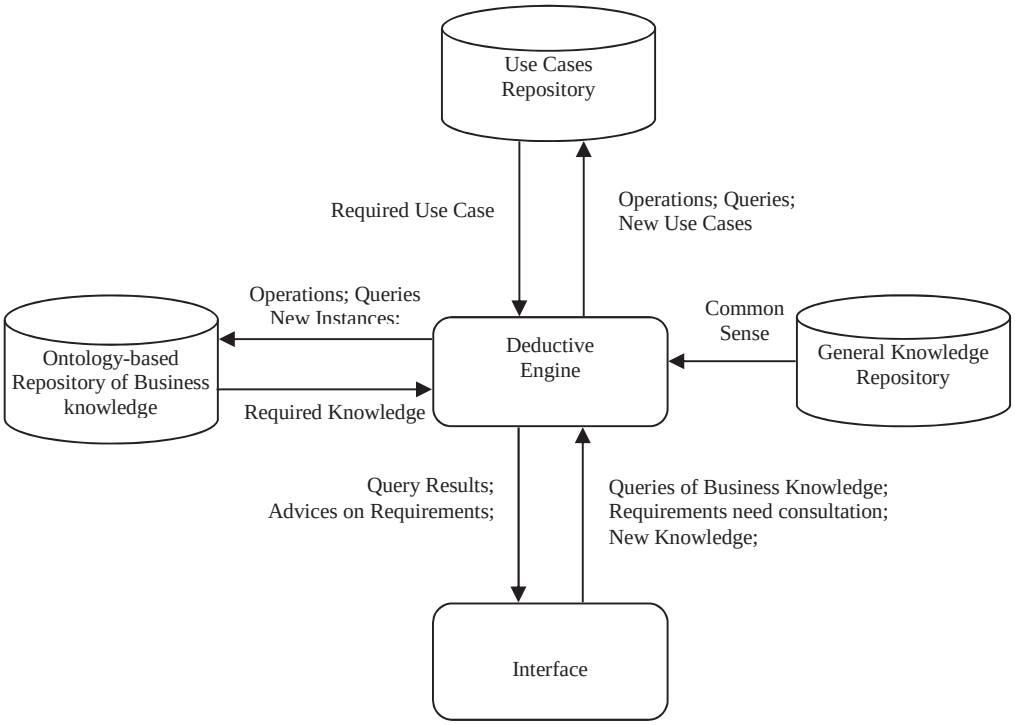

Figure 2. 1 The architecture of proposed system

The *general knowledge repository* provides semantic meaning of words for the deductive engine. The semantic meaning of words can help the deductive engine to (i) process and extend queries, (ii) enhance the efficiency of

querying ontologies, (iii) improve the accuracy of mapping ontologies, and (iv) provide knowledge support to our semantic similarity measure. We use WordNet (Fellbaum 1998) as our general knowledge repository. WordNet is one of the most popular semantic networks used for estimating semantic similarities (Resnik1995, Lin 1998, Jiang and Conrath 1998). Semantic networks are considered an ideal choice for estimating semantic similarity over other lexical resources (Budanitsky and Hirst 2006).

The *deductive engine* performs operations on and with the repositories. Three modules are included in the deductive engine: (i) a semantic similarity measure module, (ii) an ontology mapping module, and (iii) the automatic Use Case generation module.

The semantic similarity measurement module is in charge of: (i) identifying related words of each key word in a query, (ii) utilising related words to extend the query according to their semantic distance to the key words, and (iii) providing semantic similarity measurement to support the linguistic matching for the ontology mapping.

The ontology mapping module can help the system to (i) merge new knowledge into the business knowledge repository or/and the general knowledge repository, and (ii) unify current business knowledge repository or/and the general knowledge repository with newly coming repositories.

The automatic Use Case generation module is in charge of generating Use Cases automatically based on the processes retrieved by the system from the MITPH.

The *User interface* is the component that defines the interaction between the system and users of the system: (i) acquiring queries; (ii) processing queries with light weight NLP techniques to get key words of queries, e.g., running a

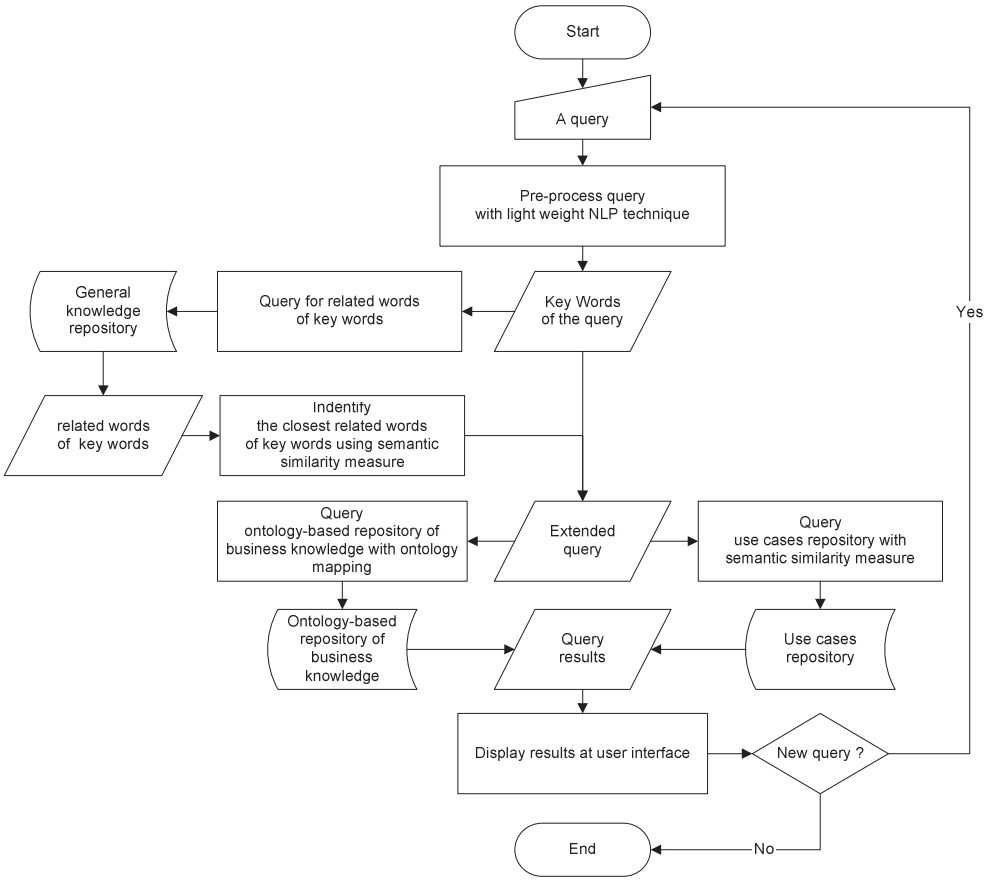

Figure 2. 2 The flow chart of one query thread

Porter Stemming Algorithm (Porter 1980); and (iii) presenting results given by deductive engine in different styles.

A flow chart of one query thread in our system is presented in the Figure 2.2. The process can be described as follows: (i) a query is pre-processed by the user interface to get its key words; (ii) the set of closest related words of those key words are selected from the general knowledge repository using the semantic similarity measure; (iii) the query is extended by the set of closest related words; (iv) the extended query is used to enquiry the business knowledge repository and Use Case repository; and (v) the query results are displayed to the user.

2.2.2 Research issues and proposed solutions

In order to develop the system successfully, we identified the following four research issues:

1. How to develop a new semantic similarity measure, which will help improve effectiveness of queries and provide assistance to ontology mapping;
2. How to develop a new ontology mapping method, which will help merge ontologies;
3. How to generate essential Use Cases automatically from retrieved business processes;
4. How to evaluate the new methods proposed and our system.

Semantic similarity measure

Semantic similarity measures have been widely employed by various applications in the area of Natural Language Processing (NLP). Examples of this are word sense disambiguation (Navigli 2009), malapropism (Lin 1998), information retrieval (Richardson and Smeaton 1995) and natural language learning (Leacock et. al. 1998).

With the help from a semantic similarity measure, our system will be able to (i) locate the closest related concepts from the general knowledge repository for the purpose of extending queries, and (ii) identify the related business processes from the business knowledge repository, and (iii) identify the related Use Case from the Use Case repository.

We propose a new semantic similarity measure utilising WordNet and the Normalised Google Distance (NGD). Current WordNet-based semantic similarity measures all have their own problems (for details please refer to Section 3.1). For example, the node counting methods ignore the structure information of WordNet (e.g., type of relations, semantic distance between

adjacent concepts), which may bring coarse results; and most edge counting methods equally treat all relations between adjacent concepts, which may cause inaccuracy. In our newly proposed measure, NGD will be used to calculate a unique length for each edge in the shortest path between candidate concepts in the WordNet graph. The details of our semantic similarity measure are presented in Chapter 3.

Ontology mapping

Ontology mapping is the key component of ontology merging. Ontology merging is crucial in ontology research, since it enables the sharing and integration knowledge across various ontologies.

In our system, ontology merging will be employed when there are new repositories are available to be integrated with our ontology-based repositories (i.e., the business knowledge repository and the general knowledge repository).

We propose a new ontology mapping method for our system based on MIMapper (Kaza and Chen 2008). Kaza and Chen (2008) claimed that MIMapper is one of the best ontology mapping methods so far. Our new ontology mapping method improves on the quality of mapping, which is needed for updating and integrating ontology-based repositories in our system. In our mapping method, our semantic similarity measure will be used to matching class names and to locate the most informative instances of their respective classes.

The details of our new ontology mapping method are presented in Chapter 4.

Automatically generating essential Use Cases

Essential Use Cases are first introduced by Constantine (1997) for describing the intended interactions between actors and systems at an abstract level (Unhelkar 2005). Essential Use Cases are normally used at the beginning of requirements elicitation as a set of conceptual Use Cases. They can be extended into conventional ones as the analyst gains a better knowledge of the system under analysis.

Santander and Castro (2002) proposed an approach for deriving Use Cases from organisational models under goal-oriented analysis. Since organisational models can be seen as collections of business processes, based on Santander and Castro's guide lines, we propose to automatically generate essential Use Cases from the business processes stored in the MIT Process Handbook. To achieve this goal, we propose a set of mapping rules for our system to automatically retrieve the information required for generating Use Cases (e.g., actors, goals, and steps of Use Case flows) from the components included in the business processes. Those essential Use Cases should help analysts develop conventional Use Cases more efficiently.

The details of automatically generating essential Use Cases are presented in Chapter 5.

Evaluation methods

Two methods will be utilised to evaluate our semantic similarity measure. One method of comparison is with human judgement. Two popularly used human judgement benchmark data will be employed, i.e., Rubenstein and Goodenough (1965) List, Miller and Charles (1991) List. Another method is to evaluate the newly proposed measure with given NLP applications. Two kinds of NLP applications can be employed to conduct the evaluation: (i)

word sense disambiguation and (ii) malapropism detection. Our evaluation is detailed in Section 3.3.

We will evaluate our ontology mapping method through calculating the precision, recall and F-measure of the mapping results. The values of these criteria will be compared to the corresponding values calculated from results of other mapping methods, such as, PROMPT, LOM, Chimaera, and MIMapper. This evaluation is detailed in Section 4.4.

In order to evaluate the method for automatically generating essential Use Cases, we will generate a set of essential Use Cases from the MIT Process Handbook, and ask requirements analysts to give feedback about the quality of the generated Use Cases. This evaluation is detailed in Section 5.4.

The three following hypotheses will be tested in the evaluation of our system: (i) the system can effectively retrieve the relevant business process from MITPH, (ii) the system is more helpful in retrieving relevant business processes from MITPH compared with the online MITPH, and (iii) the system is easy to use. Three scenarios will be designed for participants to test use our system. The participants will be asked to complete a questionnaire for collecting their opinions towards the system. The details the evaluation of our system is presented in Chapter 6.

2.2.3 Application scope

Kaindl et al. (2000) suggest a taxonomy for requirements engineering tools (as illustrated in Figure 5.1), we classify our system as a tool/system for the elicitation of requirements, i.e., the coloured box in Figure.2.3.

Two groups of requirements analysts that may benefit from our system are:

- Analysts who are not familiar with the business domain of her or his project, and does not have enough support from business domain experts. With help from our ontology-based repository, they can effectively gain a quantity of preliminary knowledge of the business domain. As a result, they will be able to communicate with business stakeholders more efficiently and understand their expressions more accurately.

- Analysts who apply goal-oriented methods during requirements elicitation. The categorised relations stored in our repository will use the business context of initials goals to help them ask "How" and "Why" questions systematically in an effective way; and thus, the efficiency and robustness of the goal oriented approach is supposed to be enhanced.

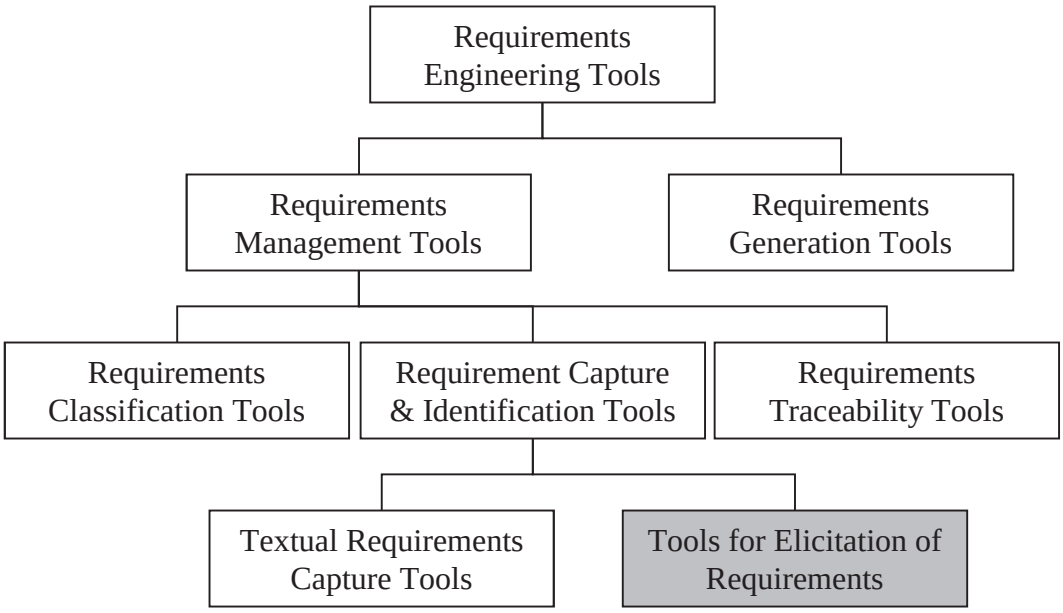

Figure 2. 3 The Requirements Engineering tool taxonomy

2.3 Summary

In this chapter, we have demonstrated the importance for requirements analysts to possess preliminary business knowledge. A survey of nine existing requirements engineering tools has been presented. According to our review, various requirements engineering tools have been developed to enhance the performance of requirements analysts, yet there is no tool that has been developed to help the junior requirements analyst gain preliminary business knowledge.

We propose an Ontology-based Knowledge Support System for Requirements Analysis (OKSSRA) to help junior requirements analysts gain preliminary business knowledge. A high level system structure of OKSSRA has been outlined. We also defined four research issues for developing OKSSRA, followed by proposed solutions to those issues. The application scope of OKSSRA was also presented.

Chapter 3.

Semantic Similarity Measure Module

A semantic similarity measurement module is introduced into OKSSRA so that it can (i) locate the closest related concepts from the general knowledge repository for the extension of queries; (ii) identify semantically related business processes from the business knowledge repository; and (iii) identify semantically related Use Cases from the Use Cases repository. .

A new semantic similarity measure is proposed. The new measure makes use of the information retrieved from a semantic network (in our case we are using WordNet) and the Internet. In particular, the structural information from the semantic network and the statistical information obtained from the Internet are combined to quantify the semantic similarity between words.

This chapter is organised as follows: Section 3.1 gives a brief introduction to WordNet and provides a review on previous WordNet-based semantic similarity measures. Pros and cons of these measures will be pointed out to motivate our newly proposed measure. In Section 3.2, we propose a new WordNet-based semantic similarity measure and the motivations of using the Internet as a corpus and Normalised Google Distance (NGD) as a tool to estimate semantic relatedness between concepts. Section 3.3 presents the evaluation experiments and associated results by which we employ two experimental methods: (i) comparing the rating results with human judgement; and (ii) applying the semantic similarity measure to a given Natural Language Processing (NLP) task (in this case, malapropism detection). Section 3.4 concludes this chapter and proposes possible extensions of our new measure.

3.1 Previous research on semantic similarity measure

Measuring semantic similarity between words plays an important role in many research areas including Artificial Intelligence, Linguistics, Cognitive Science and Knowledge Engineering (Turney 2006). The semantic similarity between words can be utilised to assist computers disambiguating word senses (Navigli 2009), to help detect malapropism (Lin 1998) and to improve the accuracy of mapping ontology entities (Kaza and Chen 2008). Accurate estimation of semantic similarity can benefit all these applications in the aforementioned areas and can enhance their performances in some extent.

Measuring the semantic similarity or distance between words is a process of quantifying the relatedness between the words utilising some information obtained from certain information sources. These information sources can be (i) lexical recourses such as dictionaries, thesauri and semantic networks (Pirrò 2009); (ii) collections of documents such as corpus (Turney 2006), and (iii) the Internet (Cilibrasi, and Vitányi2007, Gabrilovich and Markovitch 2007).

Semantic networks are considered a better choice for estimating semantic similarity than other lexical resources (Budanitsky and Hirst 2006). WordNet (Fellbaum 1998) is one of the most popularly used semantic networks for estimating semantic similarity (Banerjee and Pedersen 2003, Budanitsky and Hirst 2006, Hirst and St-Onge 1998, Jiang and Conrath 1998, Leacock and Chodorow 1998, Lin 1998, Pirrò 2009, Resnik 1995, Richardson and Smeaton 1995, Sussna 1993, Yang and Powers 2005).

3.1.1 WordNet

WordNet (Fellbaum 1998) is a semantic network developed by the Cognitive Science Laboratory at Princeton University. It has been widely used as a lexical database for English words. Different from other traditional lexicons,

WordNet organises words based on relationships among words instead of lexical forms of words. The purpose of structuring WordNet in this way is to resemble the way of human being organising their lexical knowledge.

In WordNet, one word is represented by a set of unique wordsense(s), and each *wordsense* represents a single meaning of this word. Wordsenses that share similar meaning are represented as *synsets* ("synonym set"). Subsequently WordNet is a graph whose nodes are the wordsenses and synsets and whose edges relate wordsenses to synsets and conceptual relationship among synsets. For example, {*car, auto, automobile, motorcar*} all have one wordsense belonging to the synset of *automobile*. More than two thirds of the nodes in the semantic network of WordNet are synsets.

Synsets and wordsenses are connected by *containWordsense* relations. *hyponymOf* is the key relation for the noun synsets in WordNet (Seco 2005), which has been widely used to estimate the semantic relatedness among nouns. It is also our choice in this research.

WordNet has been commonly used to measure semantic similarity among words since it has the inherent advantages of being structured in the way of simulating human recognition behaviours (Fellbaum 1998). The reminder of this section will give a brief review on previous WordNet-based semantic similarity measures organised in the aforementioned three categories.

3.1.2 WordNet-based semantic similarity measures

The WordNet-based semantic similarity measures can be classified into three categories based on the way that WordNet is utilised: (i) node-based, (ii) edge-based, and (iii) hybrid.

Node-based methods estimate the semantic similarity by computing the amount of information contained by related words in WordNet (e.g., Resnik

(1995)). The amount of the information of a word is normally measured by the *Information Content (IC)* of the word. The IC of a word is computed by the probability of the word appearing in a corpus. There are several drawbacks to this kind of method: (i) it is normally a time-consuming to analyse a corpora for estimating the IC value of a word; (ii) unbalanced contents of the corpora employed may significantly decrease the accuracy of the obtained IC values.

Edge-based methods assess the semantic similarity by counting the number of edges on the shortest path between the words in WordNet (e.g., Yang and Powers (2005)). This kind of methods simulates the way humans determine semantic similarity (Fellbaum 1998). However, the accuracy of edge-based methods is significantly affected by lack of considering the variety of semantic distances between adjacent words, which is caused by the uneven densities in WordNet.

Hybrid methods combine the information content theory and the structure information from WordNet to estimate the semantic similarity between words (e.g., Jiang and Conrath (1998)). According to previous results (Budanitsky and Hirst 2006, Pirrò 2009), this kind of methods is able to achieve better evaluation results compared with the node-based methods and the edge-based methods, though the accuracy of their results may affected by the insufficiency of the employed corpora, e.g., unbalanced contents.

Node-based methods

Node-based methods use the amount of information contained by related nodes (i.e., related concepts) in WordNet to estimate semantic similarity between the concepts of interest, i.e., c_1 and c_2. Thus, these kinds of methods are also referred to as *information-based methods*.

Most of these methods employ the information content of a concept to quantify the amount of information that the concept contains. According the

definition in the information theory (Rubenstein and Goodenough 1965), the information content (*IC*) of a concept *c* can be quantified by *IC(c)*=-log*(P(c))*, where *P*(*c*) is the probability of *c* appearing in a corpus.

Resnik (1995) believes that the similarity of concepts c_1 and c_2 is determined by the closest common superordinate concept (i.e., hypernym) of c_1 and c_2 in WordNet. Thus, Resnik proposed the following equation for calculating the semantic relatedness of c_1 and c_2:

$$rel(c_1, c_2) = \max_{c \in S(c_1, c_2)} [-\log P(c)]$$

where $S(c_1,c_2)$ is the set of lowest subsumers of c_1 and c_2; $P(c)$ is calculated by

$$P(c) = \sum_{w \in senses(c)} freq(w)$$

where *senses(c)* is the set of word senses of *c*.

Richardson and Smeaton (1995) amended Resnik (1995)'s method by using the following equation to calculate $P(c)$:

$$P(c) = \sum_{w \in senses(c)} \frac{freq(w)}{|senses(c)|}$$

where $|senses(c)|$ stands for the number of word senses that *c* has.

Banerjee and Pedersen (2003) developed a score schema to estimate the relatedness through cross comparing the words being used in the definition and the hypernyms of c_1 and c_2.

A common drawback of node-based methods is that the ignorance of the WordNet structure information can bring coarse results to the estimation of semantic relatedness between concepts.

Edge-based methods

Edge-based methods utilise the shortest path between concepts (i.e., c_1 and c_2) in WordNet to estimate the semantic relatedness between c_1 and c_2. Lengths of

all edges on the shortest path are accumulated to quantify the semantic similarity. It is the way of calculating the length of an edge that differentiates methods in this category.

Sussna (1993) proposed to use the following equation to compute the semantic relatedness of c_1 and c_2:

$$rel(c_1, c_2) = \frac{wt(c_1 \rightarrow r) + wt(c_2 \rightarrow r')}{2 \times \max(depth(c_1), depth(c_2))}$$

where $depth(c_1)$ and $depth(c_2)$ are the depths of the hierarchy structure of WordNet that c_1 and c_2 are in, respectively; r and r' are the relations between c_1 and c_2 setting off from c_1 and c_2, respectively; $wt(c_1 \rightarrow r)$ is the type-specific fanout factor, which is computed by

$$wt(c_1 \rightarrow r) = \max_r - \frac{\max_r - \min_r}{edges_r(c_1)}$$

where $\max_r$ and $\min_r$ define the weight range of relation r.

Leacock and Chodorow (1998) suggested that the semantic relatedness between c_1 and c_2 can be estimated by using the edge number of the shortest path between c_1 and c_2, *length* (c_1, c_2), and the depth of the involved taxonomy tree in WordNet, D. They proposed that the semantic relatedness between c_1 and c_2 can be calculated by the following equation:

$$rel(c_1, c_2) = -\log \frac{length(c_1, c_2)}{2D}$$

Hirst and St-Onge (1998) believes that edges with different directions in the hierarchy structure of WordNet have different lengths. The direction types of relations are used to restrict the allowable paths between c_1 and c_2 in WordNet and to define the strength of turns in the allowable paths. Hirst and St-Onge uses:

$$rel(c_1, c_2) = C - length(c_1, c_2) - k \times turns(c_1, c_2)$$

to calculate the semantic relatedness of c_1 and c_2, where *length*(c_1, c_2) is the length of the shortest allowable path between c_1 and c_2; *turns*(c_1, c_2) is the number of the direction turns in the shortest path; C and k are the constants with a value of eight and one, respectively.

Yang and Powers (2005) proposed that the semantic relatedness between c_1 and c_2 can be calculated by

$$rel(c_1, c_2) = \begin{cases} \alpha_t \prod_{i=1}^{edges(c_1, c_2)} \beta_{ti}; & \text{if } edges(c_1, c_2) < \gamma \\ 0; & \text{if } edges(c_1, c_2) \geq \gamma \end{cases}$$

where t indicates the relation type of edges; edges(c_1, c_2) equals to the edge number of the shortest path; α is the type factor of edges; β denotes the depth factor of edges; γ is a constant given by the operator to define the maximum length of shortest paths.

The performance of edge-based methods is usually limited by their way of calculating the edge length of the shortest path in WordNet. For instance, Sussna (1993), and Leacock and Chodorow (1998) have not considered the variance in edge lengths caused by irregular densities within WordNet; Hirst and St-Onge (1998), and Yang and Powers (2005) ignored the diversity of semantic distance among edges in a same type.

Hybrid methods

Hybrid methods combine the information from different resources to estimate the semantic similarity between the concepts, e.g., using *IC* combined with structural information from WordNet for estimation (Jiang and Conrath 1998).

Jiang and Conrath (1998) utilise the *IC* of end nodes to define the length of each edge in the shortest path between c_1 and c_2. The edge length between concept c, a node in the shortest path, and concept p, the parent node of c in the shortest path, is calculated by $length(c, p) = \log(P(p)) - \log(P(c))$. Structural

information from WordNet such as (i) the local density of the nodes, (ii) the depth of the nodes, and (iii) the type of the edges are utilised to scale the edge length with the following equation:

$$wt(c, p) = \left(\beta + (1-\beta)\frac{\overline{E}}{E(p)}\right)\left(\frac{d(p)+1}{d(p)}\right)^{\alpha} length(c, p)\ T(c, p)$$

where $\overline{E}$ is the average local density of all nodes; *E(p)* is the local density of the parent node, i.e., the number of its children nodes; *d(p)* is the depth of the parent node, and α, β and *T(c, p)* are the factors of depth factor, density factor and the factor of edge type, respectively. The semantic relatedness between c_1 and c_2 can then be calculated by the following equation:

$$rel(c_1, c_2) = \sum_{n \in \{S(c_1, c_2) - Sol(c_1, c_2)\}} wt(n, parentOf(n))$$

where $S(c_1, c_2)$ is the set of nodes in the shortest path between c_1 and c_2 ; $Sol(c_1, c_2)$ is the set of nodes in the shortest path which have no parent nodes in the path; *n* is any node include by $S(c_1, c_2)$ but exclude by $Sol(c_1, c_2)$.

Hybrid methods are believed to have better performances than the other two kinds to methods (Budanitsky and Hirst 2006). However, hybrid methods share a common weakness with the node-based methods: the data sparseness of the corpus. This weakness could be caused by two reasons: (i) the size of the corpus is relatively undersized, and (ii) the coverage of the corpus is not well balanced, e.g., text sources of the corpus are localised to publications in certain areas. One possible way to complement the deficiency is to use a better corpus.

3.2 The semantic similarity measure in OKSSRA

Our semantic similarity measure is a hybrid method combining the statistical information acquired from the Internet and the structure information from WordNet to quantify the semantic relatedness between concepts. In our

method, the Internet is used as a corpus for estimating the semantic distance between adjacent concepts along the shortest path between the concepts of interest in WordNet.

3.2.1 The Internet as a corpus

The corpus employed is believed to be the bottle neck that limits the performance of hybrid methods, though this kind of semantic similarity measure can achieve better estimation results than other two kinds of methods, i.e., edge based methods and node based methods.

It is difficult to find a suitable corpus for hybrid method. The reasons are two-fold: (i) existing corpora may not reflect the current way of using words since the text sources of the existing corpora are mainly from the articles published long time ago, e.g., the *ACL/DCI* Linguistic Data Consortium contracted at 1993 by University of Pennsylvania, was built upon the Collins English Dictionary 1979 version and the articles published in the Wall Street Journal during 1987 and 1989, and (ii) the data in stand-alone corpora may have bias since a corpus may only use certain articles in particular fields as their text sources because of various reasons, such as, the purpose of building the corpus, and the research interests of the constructors.

We believe that the Internet is an ideal corpus for calculating *IC* of concepts. The reasons include three-folds: (i) the Internet is the largest electronic text source currently available to us; (ii) the Internet is a well-balanced knowledge source; (iii) the information in the Internet is normally updated regularly and up-to-date.

Well-performed Internet search engines, such as Google, enable us to use the Internet as a huge corpus. With help from these search engines, the frequency of a word appearing in the Internet can be easily estimated using the number of web pages that contain the word.

3.2.2 Use context words to tag word senses

Tagged word senses are one of the key requirements to achieve good results when computing semantic similarity where corpora are involved. Although the Internet has its own advantages compared with traditional large size corpora, it still shares an issue with most of traditional corpora: word senses are not tagged.

To our knowledge, highly accurate word senses tagging currently can only be done manually. This issue cannot be overcome by using large size corpora. It means that even tagging a reasonable large size corpus will require a tremendous amount of work, as a reasonable large size corpus should be comprised of millions of tagged words.

Take SemCor (Miller et al. 1993) as an example, it has more than 190,000 sense tagged words, 11 out of 150 word senses (Liu et.al. 2011) cannot be found in the corpus when we were using the Miller and Charles set (Miller and Charles 1991) to perform an evaluation.

Using context words to tag word senses is one way to circumvent the issue of word sense tagging. In word sense disambiguation area, utilising context words is widely recognised as an effective method for identifying word senses since every context word is believed to share a certain amount of linguistic information with the word of interest and can be looked as a feature vector to indicate the sense of the word of interest (Navigli 2009).

We retrieve hypernyms from WordNet as context words to tag word senses. Context words are widely believed to share linguistic information with the word of interest and can be looked as a feature vector to indicate word senses (Navigli 2009). Experiment results of Clever Search developed by Kruse et al. (2005) show that hypernyms are effective as word senses indicators for querying the Internet.

3.2.3 Normalised Google distance

Normalised Google Distance (NGD) is proposed by Cilibrasi and Vitányi (2004) for measuring similarity between words or phrases utilising their numbers of search results returned by Google.

NGD uses the count of the search results returned by Google to approximate the Kolmogorov Complexity (Atallah 1999) of words for estimating their semantic relatedness. Given two words *x* and *y*, the NGD between *x* and *y* can be calculated by the following equation:

$$NGD(x, y) = \frac{\max\{\log f(x), \log f(y)\} - \log f(x, y)}{\log N - \min\{\log f(x), \log f(y)\}}$$

where *f*(*x*), *f*(*y*), and *f*(*x*, *y*) are the numbers of search results returned by Google which contain *x*, *y*, and both *x* and *y*, respectively. *N* is a normalising factor, whose value can be the total number of web pages indexed by Google or any reasonable value larger than both *f*(*x*) and *f*(*y*).

3.2.4 Calculation of semantic similarity

We use the following structure information from WordNet for calculating the semantic similarities: (i) the number of the edges on the shortest path between c_1 and c_2; (ii) the link strength of the edges on the shortest path (i.e., the length of edges); (iii) the type of the relations regarding to the edges on the shortest path; (iv) the local density of nodes (i.e., the number of links set off from a node); (v) the depth of the nodes on the shortest path, and (vi) the maximum depth of involved hierarchy structure in WordNet.

Since Jiang and Conrath (1998)'s method outperforms other methods (Budanitsky and Hirst 2006, Pirrò 2009), we proposed an equation based on their equation for calculating weighted edge length of the shortest path in WordNet:

$$wt(c, p) = \left(\beta + (1 - \beta) \frac{\overline{E}}{E(p)} \right) \left(\frac{D}{d(p)} \right)^{\alpha} NGD(c, p)\ T(c, p)$$

where c is a node on the shortest path; p is the parent node of c; $\overline{E}$ is the average local density of all nodes; $E(p)$is the local density of p, i.e., the number of its children nodes; D is the maximum depth of the semantic hierarchy structure that c and p are in; $d(p)$ is the depth of the parent node; α, β and $T(c, p)$ are the depth factor, the density factor, and the factor of edge type, respectively, and $NGD(c, p)$ is the Normalised Google Distance between c and p.

The semantic relatedness between c_1 and c_2 can then be calculated by the following equation:

$$rel(c_1, c_2) = \sum_{n \in \{S(c_1, c_2) - Sol(c_1, c_2)\}} wt(n, parentOf(n))$$

where $S(c_1, c_2)$ is the set of nodes in the shortest path between c_1 and c_2 ; $Sol(c_1, c_2)$ is the set of nodes in the shortest path which have no parent nodes in the path; n is any node included by $S(c_1, c_2)$ but excluded by $Sol(c_1, c_2)$.

3.3 Evaluation and discussion

We employ two methods to evaluate the effectiveness of our measure: (i) comparing the rating results with human judgement; and (ii) applying the semantic similarity measure to a given NLP task, i.e., malapropism detection.

3.3.1 Comparing with human judgement

Two sets of prevalent human benchmark data are employed: (i) the Rubenstein and Goodenough data set (RG-Set) (1965) and (ii) the Miller and Charles data set (MC-Set) (1991). The Pearson Product-moment Correlation Coefficient (*PPCC*, Rodgers and Nicewander 1988) is employed to calculate the consistency between similarity ratings.

RG-Set lists 65 pairs of words regarding to 51 everyday subjects. Each of the 65 pairs of words is given with a rating of relatedness by human subjects, on the scale of 0.0 to 4.0. *MC-Set* is generated by Miller and Charles (1991) through selecting 30 pairs of words from *RG set*.

The Pearson Product-moment Correlation Coefficient is used to calculate the consistency between similarity ratings. Given two sets of semantic similarity ratings, $X = \{x_1, x_2, \ldots x_i \mid i = n\}$ and $Y = \{y_1, y_2, \ldots, y_i \mid i = n\}$, the *PPCC* can be calculated by the following equation (Rodgers and Nicewander 1988):

$$PPCC = \frac{1}{n-1} \sum_{i=1, x_i \epsilon X, y_i \epsilon Y}^{n} \left(\frac{x_i - \bar{X}}{S_X}\right)\left(\frac{y_i - \bar{Y}}{S_Y}\right)$$

where n is the number of word pairs in the list; x_i and y_i are the rating values in X and Y with regard to the same word pair i, respectively; $\bar{X}$ and $\bar{Y}$ are the mean values of X and Y, respectively; S_X and S_Y are the standard deviations of X and Y, respectively. The *PPCC* values are in the range between -1.0 to 1.0. If the value is 1.0, it means that two sets of similarity ratings are identical, while if the value is -1.0, it means that the correlation is perfect negative. If the value is 0.0, it means that there is no correlation between the ratings.

Jiang and Conrath (1998)'s method is used as an indicator to show the robustness of our method. Jiang and Conrath's method has been considered as one of the best semantic similarity measures (Budanitsky and Hirst 2006, Pirrò 2009). In other words, we can believe our method is better than the other methods if our method can outperform Jiang and Conrath's method.

Table 3.1 describes when the original human judgement of MC-Set is compared against, the correlation coefficients obtained (i) by the human judgement replication, (ii) by Jiang and Conrath's method, and (iii) by our method. The correlation of human judgement replication listed in the table was reported by Miller and Charles (1991).

Table 3. 1 The correlation coefficients on MC-Set

Method	*PPCC*
Human Judgement (replication)	0.8848
Jiang and Conrath's Method	0.7509
Our Method	0.8087

Table 3.2 reports when RG-Set is applied, the correlation coefficients obtained (i) by Jiang and Conrath's method, and (ii) by our method.

Table 3. 2 The correlation coefficients on RG-Set

Method	*PPCC*
Jiang and Conrath's Method	0.7129
Our Method	0.7634

Figure 3.1 and Figure 3.2 illustrates, when MC-Set is applied, the scatter plot with linear model fits of between human benchmark and Jiang and Conrath's estimations, and human benchmark and our estimations, respectively.

The coefficient of determinations (R^2) of the linear model fit of Jiang and Conrath's method and our method are 0.564 and 0.654, respectively. The increase of R^2 value indicates that, compared with Jiang and Conrath's estimations, our estimations have an eight percent improvement in correlation to human benchmark.

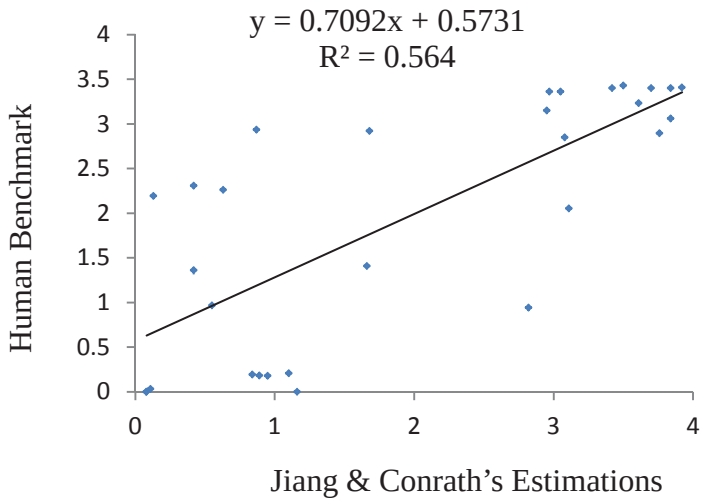

Figure 3. 1 The scatter plot with linear model fits between human benchmark and Jiang & Conrath's estimations

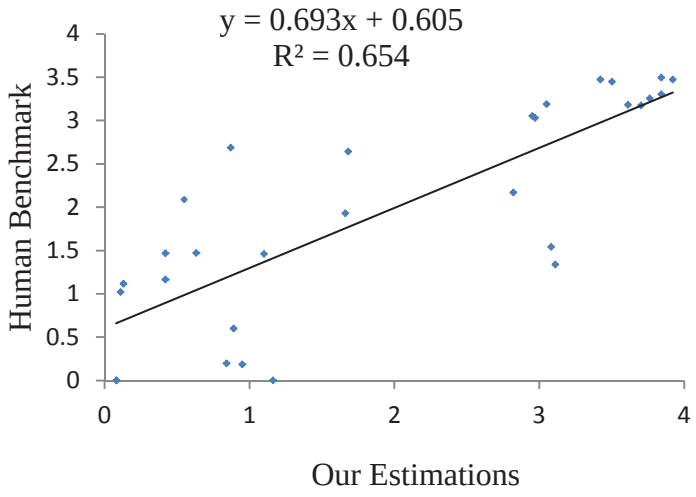

Figure 3. 2 The scatter plot with linear model fits between human benchmark and our estimations

For detailed experimental data, please refer to Appendix F. The above experiments results indicate that our method outperforms Jiang and Conrath's method. Since Jiang and Conrath's method is one of the best existing semantic similarity measures (Budanitsky and Hirst 2006, Pirrò 2009), we believe that our method can provide better estimations on semantic similarities than previous methods.

Discussion

The experimental results of calculating semantic distance between given pairs, i.e., word pairs in human benchmark data sets, confirm that our newly proposed method outperforms Jiang and Conrath's method (1998). The correlation coefficient of the replication of human judgement for MC-Set, i.e., 0.8848, is believed to be the upper bound for automated algorithms, therefore, even the small improvement made by modifying the equation should also be recognised.

Through a series of thorough experimentations, we also found that the values of α and β can affect the performance of estimating semantic similarities. Table 3.3 illustrates how the values of depth factor α and density factor β

affect the performance of our method. The data presented in this table are correlation coefficient values corresponding to various values of α and β, when MC-Set was employed as the test data set, the T_{HO} and T_{CW} were set as 1.0 and 0.1, respectively. According to this table, the value of correlation coefficient could be increased at least 0.1 with a carefully chosen values of α and β. According to the experimental results in Table 3.3, the best values of α and β are 0.2 and 0.9, respectively.

Table 3. 3 The correlation coefficients obtained by our method on MC-Set when different values of α and β are applied

	α = 0.0	α = 0.2	α = 0.4	α = 0.6	α = 0.8	α = 1.0
β =0.1	0.70740	0.75035	0.76628	0.77046	0.76484	0.75253
β =0.3	0.77984	0.79626	0.79770	0.79077	0.77800	0.76176
β =0.5	0.79521	0.80440	0.80265	0.79363	0.77972	0.76300
β =0.7	0.80091	0.80729	0.80431	0.79453	0.78024	0.76339
β =0.9	0.80350	0.80868	0.80508	0.79493	0.78046	0.76357

3.3.2 Evaluating with a given NLP application

Application to an NLP problem is another way to evaluate semantic similarity measures (Budanitsky and Hirst 2006). Various NLP problems can be used to evaluate semantic similarity measures, e.g., word sense disambiguation, and malapropism detection.

We adopt malapropism detection for this evaluation task. The malapropism detection has been considered as a standard method for evaluating semantic similarity measure (Hirst and St-Onge 1998, Lin 1998, and Budanitsky and Hirst 2006).

In computational linguistics, the *malapropism* stands for the spelling errors that results an unintended but existing word because of ill-situated characters

(Wilks et al. 1996). For instance, in the sentence of "New Zealand has *tow* main islands." The word "*two*" instead of "*tow*" was intended, and we call the word "*tow*" as a malapropism here.

Since both "*tow*" and "*two*" are existing words in dictionaries, this error is difficult to be automatically detected using light weight NLP techniques, such as spell checking. However, semantic similarity measures can be utilised to detect this kind of errors by computing the relatedness between words and their context words. If a word is semantically unrelated to its context words, we can say that it is a suspect of malapropism. Thus, we can evaluate the performance of semantic similarity measures by comparing their abilities of detecting malapropisms.

This evaluation is designed on the basis of a similar evaluation conducted by Budanitsky and Hirst (2006). The assumptions of conducting this evaluation include (Budanitsky and Hirst 2006): (i) a real-world spelling error can be simulated by replacing a word with small change of spelling; (ii) a real-world spelling error is unlikely to be semantically related to its context words; and (iii) the intended word is semantically related to its context words.

Implementation

240 pairs of nouns retrieved from WordNet are used as malapropism candidate pairs. The spellings of most of these noun pairs are identical except two of the positions of their two characters are swapped to mock malapropism errors, e.g., "conversation" and "conservation".

186 paragraphs retrieved from SemCor are used as the context documents of malapropisms. After removing proper nouns, such as "January" and "John", and stop-list words, such as "the" and "always", 192,707 words remain in these paragraphs.

22,355 lexemes of the 480 candidates are found in the 186 context paragraphs. 100 out of these identified lexemes were randomly chosen and replaced by the corresponding malapropisms, respectively.

The size of the context window is set to nine, i.e., four adjacent nouns on each side of the word of interest (*woi*). Each of these context words were used to compute the relatedness between the malapropism and its context. Leacock et al. (1998) suggests that context windows at the size of seven or nine are enough to provide the context information.

The weighted average relatedness rating of all context words, eight words in our case, is used to estimate the relatedness between the *woi* and its context words. A higher weight is assigned to the context word that it is closer to the *woi*, under the assumption that the closer a context word is to the *woi*, the stronger semantic connection is supposed between the context word and the *woi*. Given the eight context words around the *woi* ,$\{ cw_1 , cw_2 , \ldots , cw_i \mid i = 8\}$, and their relatedness rating to *woi* are $\{ rel_{cw_1} , rel_{cw_2} , \ldots , rel_{cw_i} \mid i = 8\}$, the relatedness rating between *woi* and its contexts can be calculated by the following equation:

$$rel_{context} = \frac{\sum_{i=1}^{8} \left((4.5 - |4.5 - i|) rel_{cw_i} \right)}{\sum_{i=1}^{8} (4.5 - |4.5 - i|)}$$

In other words, the relatedness rating of adjacent words at each side of *m* is given a weight of 4, and the weight of rating decreases 1 when the context word is one word further away from *woi*. The sum of all weighted ratings dividing by the sum of all weight coefficients is considered as the relatedness rating between *woi* and its contexts.

For the threshold of determining if *woi* and its contexts are related, we use the average relatedness rating of word pairs “crane”-“implement” and “brother”-“monk”. This threshold is suggested by Budanitsky and Hirst for their malapropism detection tasks (2006). Since this rating separates RG-Set words into a higher-level group and a lower-level group. The higher-level group

pairs are considered to be closely related in semantic meanings, and the lower-level group pairs are less semantically related.

Results and Discussion

Recall rates of Jiang and Conrath's method and our method are employed to determine comparatively the measures ability to detect malapropisms.

Recall is a popular mechanism to measure the performance of information retrieval tasks. Malapropism detection can be seen as an information retrieval task, in which the information of interest is the malapropisms. The *recall* can be defined by the following equation:

$$R = \frac{\text{Number of Detected Malapropisms}}{\text{Total Number of Malapropisms}}$$

The magnitude of relatedness ratings are affected by the longest distance of the data sets. Therefore, we normalised all similarity ratings by multiplying them with a scale factor, which is the quotient of the longest distances.

The detailed results are presented in Appendix D. The recall rates of detecting malapropism achieved by Jiang and Conrath's method and our method are listed in Table 3.4:

Table 3. 4 The recall rates of detecting malapropism achieved by Jiang and Conrath's method and our method

Methods	Recall
Jiang and Conrath's method	0.90
Our method	0.93

The above experiment results also confirm that our equation outperform those using Jiang and Conrath's equation (1998).

3.4 Summary

In this chapter, we propose a new hybrid method for measuring semantic similarity by combining the structural information of WordNet and the statistical information obtained from the Internet.

In our method, the length of the shortest path between two concepts in WordNet is used to measure the semantic similarity of the two concepts. The Internet is used as a corpus for estimating the length of each edge on the shortest path. Previous semantic similarity measures usually employ standalone corpora to estimate the edge lengths, and their performances are limited by the data sparseness or unbalanced knowledge of the employed corpus. Our experimental results show that the Internet can overcome the drawbacks that stand alone corpora brought to semantic similarity measures. We use context words to tag word senses and Normalised Google Distance (NGD) to calculate the length of edges on the shortest path.

The proposed semantic similarity measure has been evaluated by comparing the rating results with human benchmark data and their performance on malapropism detection. We also compare our results with the results returned by Jiang and Conrath's method, which, according to previous researches (Budanitsky and Hirst 2006, Pirrò 2009), provides one of the best results on measuring semantic relatedness. The experimental results indicate that employing the Internet as a corpus and using NGD to compute the edge lengths enable our method to outperform previous semantic similarity measures.

Chapter 4.

Ontology Mapping Module

An Ontology Mapping (OM) module is introduced to OKSSRA so that the system can conduct the ontology merging for the following purposes: (i) updating the ontology-based repositories, and (ii) integrating the ontology-based repositories in OKSSRA with other ontology-based repositories.

This chapter is organised as follows. In Section 4.1, we give a brief introduction to ontology mapping. Section 4.2 reviews the applications of ontology in computer science and current research on ontology mapping. Section 4.3 presents the design of our ontology mapping method. The experimental results and discussions are given in Section 4.4. Section 4.5 concludes this chapter.

4.1 Introduction

In order to allow our system to provide up to date business knowledge, we proposed and developed the Ontology Mapping (OM) module to allow OKSSRA (i) updating its repositories with new information and knowledge, and (ii) extending its application scope.

In this changing world, companies need to keep their information and knowledge up to date. For example, the technologies have subtly changed the way of recruiting new staff. The traditional medium (e.g., newspapers) used to be the main channel for advertising recruitment information. However, nowadays, the internet has become a more powerful media for broadcasting recruitment information. Thus, posting advertisements on the Internet should

be included as a part of the business process of recruiting new staff, which is usually not the case for those old fashioned business process repositories. In order to be capable of providing knowledge support on such new situations, OKSSRA should be able to update its knowledge repositories if it is required.

The ability to merge the current ontology-based repositories with other ontology-based repositories will allow OKSSRA to extend its application scope. It is wildly recognised that repository related systems should be able to extend their repositories (Blaha et al. 1998). The extendibility of the business process repository endowed by the OM module supports OKSSRA to enrich the contents of the business process repository. The source of the new business processes can be newly finished projects or other existing business process repositories, e.g., the BMP (Best Manufacturing Practices) developed by the University of Maryland.

Different organisations or people normally build their ontologies based on their own preferences or according to their own needs. Thus, it is common that different ontologies use various expressions to describe similar and, sometimes, identical subjects. An effective ontology mapping method gives accurate suggestions on correlating ontology entities describing similar or identical subjects across ontologies.

Ontology mapping has attracted a large amount of attention in the research area of ontology (Choi et. al. 2006). Ontology mapping is crucial for broadening the utility of existing ontologies. The ability to match entities across ontologies will enable (i) the integration of the information contained by various ontologies, and (ii) the creation of new ontologies based on existing ontologies.

The need to integrate information from multiple ontologies arises from the desire of broadening the use of existing ontologies (Kalfoglou and Schorlemmer 2003). The scope of a single existing ontology can hardly cover

all practical applications. It is also impractical and expensive to create a brand new ontology for every newly encountered situation.

A few practical situations may require help from ontology mapping to merge multiple ontologies. For instance, a company that has its own company-specific ontology may want to combine their ontology with a business process ontology, such as MIT Process Handbook (MITPH), for the purpose of reviewing or upgrading their business processes. However, it is likely that the company-specific ontology does not have the same schema as that of MITPH. Such a situation means that the ontology merging cannot be conducted by simply relating entities on the basis of their relative positions across the ontologies. The most reliable way to matching entities between heterogeneous ontologies is to manually check entities across ontologies to determine if they are equivalent, though this is impractical as it is often extremely time-consuming. In such situations, an ontology mapping method is desired to effectively match entities across ontologies according to the semantic meaning and other criteria.

Identifying the inter-ontology relationship among entities is one of the most common applications of ontology mapping (Huhns and Stephens 2002).

4.2 Previous research on ontology mapping

The word “ontology” has long been used in philosophy since the time of ancient Greek. It stems from the Greek words: *ontos* (being) and *-logy* (theory). Ontology was originally used to describe the branch of philosophy that defines entities and types of entities for existences through categorising and identifying relationships between them.

4.2.1 Ontology in Computer Science

The concept of ontology was introduced to the field of Computer Science about three decades ago. The first widely recognised ontology project, i.e., Cyc project was initiated in 1984 (Lenat and Guha 1990). However, the first widely acknowledged definition of "ontology" in Computer Science was proposed by Gruber (1993):

> *"Ontology is a formal and explicit specification of a shared conceptualisation."*

Among the sub-disciplines of Computer Science, Artificial Intelligence (AI) first employs ontology as a knowledge representation method to share common understanding and conceptualisations. As a sophisticated application, ontology has also been used as an infrastructure of the communication mechanism between agents in Multi-Agent Systems.

As one of the core components of Web 2.0 standards, ontology enables the knowledge understanding and sharing for future web pages. Furthermore, ontology has been increasingly used for knowledge representation and management in various sub-disciplines of Computer Science, e.g., Software Engineering. Recent research in the field of Software Engineering has employed ontology to (i) acquire information across multiple documents (Lee et al. 2006), (ii) improve the efficiency of requirements query (Caralt and Kim 2007), (iii) perform the semantic requirements profiling (Bhatt et al. 2007), and (iv) to classify plugin resources (Dietrich et al. 2007).

Bhatt et al. (2007) introduced an ontology-based framework for efficiently retrieving resources from a resource repository, e.g., a process template repository. The main idea of this framework is to utilise the semantic contexts of users' requests to retrieve resources of interest from a resource repository. Bhatt et al. used (i) a sub-ontology extracted from a domain knowledge model

to represent the semantic contexts of users' requests, and (ii) a relevant domain ontology to categorise knowledge in the resource repository.

Advantages of Ontology

For human beings, ontology provides an easier way to obtain new knowledge and promote understanding by comparison with other media. Bartlett (1932) conducted a research on human learning behaviours and memory. According to the experimental results, Bartlett concluded that "*without some general setting or label as we have repeatedly seen, no material can be assimilated or remembered*" (Mayer 2003). "*General setting and label been seen repeatedly*" is exactly how ontologies help human beings learn and memorise new knowledge through elucidating the relationships between new knowledge and previously gained knowledge.

Ontology enables computers to comprehend and extensively process knowledge (Liu and Fang 2006). Several other methods for knowledge representation and management, such as control vocabulary, taxonomy, and the thesaurus, are often mistaken as the synonyms of ontology.

Control vocabulary is a carefully selected list of terms that have been enumerated explicitly. A vocabulary registration authority controls the list of terms, so that each term in control vocabulary has a non-redundant, unambiguous definition.

A taxonomy is a collection of controlled vocabularies that are hierarchical in structure. The vocabularies in taxonomies are in a parent-child relation. The complexity of these relations varies according to practice as the number of parent-child relationships that are allowed may differ.

A thesaurus is a set of controlled vocabularies organised in a network like structure. The thesaurus vocabularies are in an associative relation, which is different from the parent-child relations among the taxonomy terms.

An ontology can be seen as a group of controlled vocabulary terms that are organised as a network. Different from a taxonomy or thesaurus, an ontology has a set of grammar rules to explicitly indicate how the controlled vocabularies are used to describe an object within a certain domain. As an outcome of this characteristic, ontologies have the capability of inference.

A data schema is another knowledge representation method that is commonly mistaken for being an ontology. A data schema gives a definition of structures in which data are stored. The capacity to make inference is widely accepted as the major difference between data schemas and ontologies (Ashino and Yoshizu 2005). However, a data schema can be closely related to ontologies, e.g., Resource Description Framework Schema (RDFS) is one of the key components of Web Ontology Language (OWL).

The ability to infer allows ontologies to gain new knowledge by processing stored knowledge (Pan and Shen 2005). Inferences also allow ontologies to explicitly explain why a certain experience or knowledge is not suitable for the situation at hand (Stojanovic and Stojanovic 2002).

In addition to the ability to process knowledge extensively, ontologies also demonstrate the potential for effective knowledge extraction. The knowledge acquisition bottleneck is a ubiquitous issue faced by knowledge-based systems, such as Case-Based Reasoning (CBR) systems, Decision-support systems, and Expert systems, is. The knowledge acquisition bottleneck represents the difficulty of capturing knowledge for a certain purpose (Feigenbaum and McCorduck 1983).

Using ontologies has been widely recognised as a way of overcoming the knowledge acquisition bottleneck (Buchanan and Wilkins 1993, Boicu et al. 2001). By identifying the underlying relations and connections between existing methods and tasks solved by those methods, ontology can help

maximise the efficiency of acquiring stored knowledge in the course of reasoning (Díaz-Agudo and González-Calero 2002).

4.2.2 Current research in ontology mapping

The definition of ontology mapping given by Sinir (2007) is:

> *"Ontology Mapping is the process whereby two ontologies are semantically related at the conceptual level, and the source ontology instances are transformed into the target ontology entities according to those semantic relations."*

In this section, we briefly review six popular tools that provide ontology mapping suggestions. Chimaera and iPROMPT give mapping suggestions only according to a light weight lexical matching on the class names.

AnchorPROMPT utilises the ontology structural information to predict the potential matching entities. We believe that the semantic relatedness among the name of ontology elements should be considered as well.

ONION provides a set of precise mapping suggestions based on its formal inference. However, it requires a large amount of human efforts to formalise the source ontologies.

MAFRA and LOM give mapping suggestions based on the light weight semantic similarity of the text description of entities. However, we believe that a more sophisticated semantic similarity method can be employed to provide more accurate suggestions on ontology mapping.

MIMapper matches ontology entities on the basis of (i) the semantic relatedness among entity names, and (ii) the relatedness among Most

Informative Instances (MIIs), which are identified using Point-wise Mutual Information (PMI) method.

However, we believe that a more sophisticated semantic similarity measure, e.g., our semantic similarity measure (in Section 3.2), can be used to enhance the accuracy of the class names mapping. Additionally, PMI is actually a measure of the association between a class and its instances. We argue that the semantic similarity measure between the class name and the instance names can also be used to identify the association between a class and its instances. Therefore, we propose that a new ontology mapping method can be developed using these semantic similarity measures to conduct matching activities.

A detailed review of the six ontology mapping tools is presented in Appendix E.

4.3 The ontology mapping method in OKSSRA

Lambrix and Tan (2007) believe that there are five kinds of ontology mapping strategies according to the information that they rely on: (i) linguistic-based matching, (ii) structure-based matching, (iii) constraint-based matching, (iv) instance-based matching, and (v) auxiliary-information-based matching.

Linguistic-based matching utilises the textual descriptions of entities, e.g., the entity names and the definition of entities, to identify matches among entities. This kind of matching can be conducted using the following algorithms: (i) the string matching, and (ii) the semantic matching.

Structure-based matching utilises the structural information of ontologies and previously matched entity pairs to identify relevant entities. For example, given two entities belonging to different ontologies, if their super-entities (i.e., parent nodes) and sub-entities (i.e., child nodes) are identified as

corresponding matches by previous calculations, the two given entities are likely a match pair.

Constraint-based matching uses the properties of axioms in the ontologies to help estimate the relations between entities.

Instance-based matching utilises the relevant instances to discover matching class entities across ontologies. The instances have been proven to be helpful to identify the relationships between seemly unrelated objects (Kaza et al. 2006). For example, given two address book ontologies, the class *FirstName* in one ontology is likely to share more common instances with the class *GivenName* than with the class *LastName* or class *CityName* of another ontology. Thus, the common first names (i.e., the common instances) can effectively help discover that the class *FirstName* and the class *GivenName* are a matching pair across ontologies.

Auxiliary-information-based matching utilises the information from external resources other than the ontologies themselves to help identify the relations between entities across ontologies. The external resources can be domain ontologies, dictionaries, and thesauri, etc. This strategy can be combined with other strategies. For instance, we can use WordNet, i.e., the auxiliary resource, which can support the linguistic-based matching by providing the semantic information.

Our ontology mapping method is adapted from MIMapper, which, as claimed by Kaza and Chen (2008), has one of the best performances among existing mapping methods.

MIMapper is a two-step method. In the first step, MIMapper conducts the linguistic-based matching, during which the class names are used to locate potential mappings. *Hypernym* and *hyponym* relations in WordNet are utilised to conduct the linguistic-based matching. In the second step, MIMapper

conducts the instance-based matching, during which the Most Informative Instances (MII) are used to match the classes across ontologies. A MII is the instance that shares the most information with its class's name among peer instances. In MIMapper, the MIIs are identified by using Point-wise Mutual Information (PMI, Baeza-Yates and Ribeiro-Neto 1999).

Our method uses (i) a more sophisticated method for conducting the linguistic-based matching in the first step, and (ii) a better method for the instance-based matching in the second step than the method employed by MIMapper. In the first step of our method, our newly proposed semantic similarity measure (Section 3.2), is employed to match the class names, instead of only using *Hypernym* and *Hyponym* relations in WordNet by MIMapper. In the second step, we use the Normalised Google Distance (NGD) instead of PMI to locate MIIs. The MIIs are then utilised to give suggestions on matching class entities.

4.3.1 Linguistic-based matching

In order to conduct the linguistic-based matching on class names, we employ the semantic distance between the names as the measure to calculate the relatedness between class names.

Pre-processing

The class names need to be pre-processed before calculating the semantic distance between them, as class names are usually not written in plain English. For instance, in order to avoid machine-reading error, spaces are usually removed from class names or replaced by certain punctuations such as underscores or hyphens.

In order to limit the potential error introduced by various textual styles of class names, we conducted a two-step pre-processing on class names prior to

calculating their semantic distances: (i) remove all punctuations or white spaces. Upper case characters are used as flags for separating words in class names, e.g., *FirstName*; (ii) split the class names into a set of meaningful English words. Since the semantic networks, WordNet in our case, only provide the semantic information for real English words or phrases such as *First Name*, a string of words such as *FirstName* will make no sense to WordNet.

Semantic Distance

The semantic distances between words of the class names are calculated by our semantic similarity measure introduced in Chapter 3. The calculation is conducted by the Semantic Similarity Measure (SSM) Module of our system.

We treat the class names as sentences to estimate their semantic relatedness. This decision is based on the observation that, similar to short sentences, class names are normally comprised of a small set of words describing a relatively complicated concept. The strategy we used to compute the semantic similarity between class names is adapted from a method designed to measure the semantic similarity between sentences proposed by Li et al. (2006). According to a thorough review conducted by Achananuparp et al. (2008), Li et al.'s (2006) method outperforms other algorithms on estimating the semantic similarity between sentences.

The lexical semantic vectors are employed to determine the semantic similarity among class names. Given two class names, N_1 and N_2, the semantic similarity between N_1 and N_2 is calculated using the following steps:

1. Create $J = \{w_1, w_2, ...w_m\}$, where J is a joined word set of N_1 and N_2 that contains all distinct words from N_1 and N_2. For example, for the class name *StreetName* and *PersonGivenName*, the join word set J is {*street, name, person, given*};

2. Calculate lexical semantic vector $s_1 = (\breve{s}_{11}, \breve{s}_{12}, ..., \breve{s}_{1m})$ and $s_2 = (\breve{s}_{21}, \breve{s}_{22}, ..., \breve{s}_{2m})$ for N_1 and N_2, respectively, where m is the number of entries that J has. The value of the element $\breve{s}_{1i}$ is mainly determined by ς_{1i}, which is the semantic similarity score between w_i (i.e., the ith word in J) and w_{i1} (i.e., the word in N_1 that is most similar to w_i). The information contents of w_i and w_{i1} are used to weight the significance of $\breve{s}_{1i}$. Therefore, the value of entry $\breve{s}_{1i}$ can be calculated by the following steps:
    a. If w_i and w_{i1} are identical, then the ς_{1i} is set to 1;
    b. If w_i cannot be found in N_1, the semantic similarity score between w_i and each word in N_1 is calculated. The highest score is set as the value of ς_{1i}. However, if the value of ς_{1i} is lower than the pre-set threshold for determining if two words are semantically related, the value of ς_{1i} will be set to 0;
    c. If there is a word in N_1 that is related to w_i, i.e., the value of ς_{1i} is not 0, the information content of w_i and w_{i1}, $I(w_i)$ *and* $I(w_{i1})$, will then be calculated, respectively ;
    d. Calculate the value of entry $\breve{s}_{1i}$ with the following equation:

$$\breve{s}_{1i} = \varsigma_{1i} \cdot I(w_i) \cdot I(w_{i1})$$

3. Use the cosine coefficient between s_1 and s_2 to calculate S_{12}, i.e., the semantic similarity between N_1 and N_2:

$$S_{12} = \frac{s_1 \cdot s_2}{\| s_1 \| \cdot \| s_2 \|}$$

The information content is believed to be an effective way for measuring the amount of information carried by words (Resnik 1995). As mentioned in Chapter 3, the information content of a word can be estimated by the probability of the word appearing in a corpus. A higher value of the probability implies the word contains less information. In our case, we use the hits of the word in the Internet to calculate its information content. The

number of the hits will be normalised using the estimated total number of webpages in the Internet, i.e., 1.5×10^{12} .

4.3.2 Instance-based matching

We utilise NGD to identify MII (Most Informative Instances). We believe that the MII appears more often with the class name on the Internet than its peer instances do.

MIMapper utilises PMI to quantify the relatedness between a class name and each of its instances on the basis of the concurrences of the class name and the instance within a corpus. There are two possible ways to improve this instance-based matching strategy of MIMapper: (i) using a better method to compute the concurrences of concept pairs, i.e., the class names and each of its instances; and (ii) employing a better corpus that reflects the relatedness between concepts in a more accurate way.

NGD is a better method than PMI for quantifying the relatedness between concepts on the basis of statistical information in corpora. Lindsey et al. (2007) compared the performances of NGD and PMI for measuring the semantic relatedness when various corpora are employed. According to their experimental results, NGD outperforms PMI in five out of six applied corpora.

The Internet is a better corpus than any exiting standalone corpus. As we have stated in Section 3.2.1, the Internet is an ideal corpus because of the following reasons: (i) the Internet can be considered as a well-balanced corpus since it is contributed by people who have various backgrounds and interests; (ii) Internet contents are normally up to date since thousands of millions of people are updating the content of the Internet on a frequent basis; and (iii) powerful search engines such as Google enable us to effectively retrieve information from the Internet.

In order to use NGD as a measure of the relatedness instead of a measure of the difference, each NGD value is subtracted from 1 to get its corresponding similarity value, since 1 is the maximum value for NGD.

In order to use the relatedness between the instances and their class names to help estimate the relatedness between classes, we use the following algorithm to calculate $I_{c_1c_2}$ i.e., the instance-based similarity between the classes of interest, C_1 and C_2:

1. Create $I = \{i_1, i_2, \ldots, i_m\}$, where I is a joined instance set of C_1 and C_2 that contains all the distinct instances of C_1 and C_2;
2. Construct instance relatedness vectors $IR(C_1)=(ir_{11}, ir_{12}, \ldots, ir_{1m})$ and $IR(C_2)=(ir_{21}, ir_{22}, \ldots, ir_{2m})$ for C_1 and C_2, respectively, where m is the number of instances in I, the value of ir_{1m} is determined by the NGD value between the name of C_1 and i_m. For example, if the name of C_1 is "*street name*", and the value of i_m is "*Oxford street*", the value of ir_{1m} can be computed by the following equation:
$$ir_{1m} = 1 - NGD(\text{"street name"}, \text{"Oxford Street"})$$
3. Calculate the cosine coefficient of $IR(C_1)$ and $IR(C_2)$ as the value of the instance-based similarity $I_{c_1c_2}$:
$$I_{c_1c_2} = \frac{IR(C_1) \cdot IR(C_2)}{\|IR(C_1)\| \cdot \|IR(C_2)\|}$$

After obtaining both the semantic similarity of class names $S_{c_1c_2}$ (introduced in Section 4.3.1) and the instance-based similarity $I_{c_1c_2}$ of class C_1 and C_2, we can calculate $M_{c_1c_2}$, i.e., the matching factor of C_1 and C_2, using the following equation:

$$M_{c_1c_2} = w_s \cdot S_{c_1c_2} + w_i \cdot I_{c_1c_2}$$

where w_s and w_i are the weight factors for $S_{c_1c_2}$ and $I_{c_1c_2}$, respectively, and $w_s + w_i = 1$.

As a summary, our method is a two-step method similar to MIMapper. However, each of two steps in our method is implemented differently from that of MIMapper for the purpose of achieving better mapping results: (i) a more sophisticated semantic similarity measure (i.e., our semantic similarity measure introduced in Chapter 3) is employed to perform the linguistic-based matching on class names, and (ii) the concurrences of the instances and their class names in the Internet are utilised to perform the instance-based matching.

4.4 Evaluation and discussion

This section reports the evaluations of our newly proposed Ontology Mapping (OM) method, followed by experimental results and discussions. Section 4.4.1 describes the evaluation measurements; Section 4.4.2 introduces the implementation details of the evaluations; Section 4.4.3 presents and discusses experimental results.

4.4.1 Task description

In order to evaluate the performance of our ontology mapping method, we compare the outcomes of our method with the mapping suggestions provided by other existing ontology mapping tools. Three measurements are utilised to conduct the comparison: recall, precision and F-measure.

Recall, precision, and F-measure are popularly employed to evaluate the performances of information retrieval algorithms (Baeza-Yates and Ribeiro-Neto 1999). The *recall* measures the completeness of results returned by algorithms. The recall of mapping results *R* can be calculated by the following equation:

$$R = \frac{|\{correct\ matches\} \cap \{returned\ matches\}|}{|\{correct\ matches\}|}$$

The *precision* measures the relevance level of results returned by algorithms. The precision of the mapping results P can be calculated by the following equation:

$$P = \frac{|\{correct\ matches\} \cap \{returned\ matches\}|}{|\{returned\ matches\}|}$$

The *F-measure* is the harmonic mean of recall and precision, which measures the effectiveness of algorithms (van Rijsbergen 1979). The value of F-measure can be calculated by the following equation:

$$F_\beta = \frac{(\beta^2 + 1) \cdot P \cdot R}{\beta^2 \cdot P + R}$$

where P indicates the value of precision; R denotes the value of recall; β is a factor determining the weights of precision and recall to the F-measure value. We set 1 as the value of β, which means that the value of precision has the same significance as the value of recall does to the value of F-measure.

We compare the performance of our method with that of MIMapper for the purpose of evaluation. MIMapper is able to achieve better performance than other existing ontology mapping methods, such as PROMPT, and Chimaera (Kaza and Chen 2008). Therefore, if our method can provide better mapping suggestions than MIMapper does, we believe that our method can also outperform other existing ontology mapping methods.

4.4.2 Implementation

The benchmark data set employed for this evaluation is the ISLab Instance Matching Benchmark (IIMB). This data set is provided by OKKAM European project (Bouquet et al. 2008) and contains 37 test cases designed to facilitate evaluation of ontology mapping algorithms' performances across a range of scenarios. The test case 001 is the original data for the whole data set, which is sourced from OWL/RDF data files containing information about actors, sports people and business organisations. The remaining 36 test cases are automatically generated by modifying the test case 001. In order to simulate

the real world scenarios, four different strategies are employed to modify the original data set for generating test data. The four strategies are as following:

- The test cases numbered from 002 to 010 are generated by applying value transformations, e.g., spelling errors, or altering representation forms of data. Different numbers of errors are introduced to each of the nine test cases to vary the difficulty level of performing ontology mapping.
- The test cases numbered from 011 to 019 are generated by applying structure transformations. Modifications are made to change the data structure of the ontologies, e.g., removing one or more instances from a class, or splitting a single class into several classes.
- The test cases numbered from 020 to 029 are generated by applying logical transformations. Modifications are made to introduce the logic variations or errors into ontologies, e.g., placing identical instances under different subclasses that share a super class, or introducing identical instances into unrelated classes.
- The test cases numbered from 030 to 037 are generated by combining the above three transformations.

The corresponding relationship between the entities in the modified data set and the entities in the original data set are stated in the alignment data file. The alignment data file is used as the benchmark data to compare with the matching suggestions given by the ontology mapping algorithms for estimating their performance.

4.4.3 Results and discussion

Table 4.1 presents the best experimental results achieved by our method and MIMapper with respect to each of the IIMB test cases. The set-up of our method is established as follows: (i) the weight factor of class name semantic similarity w_s is set 0.29, (ii) the weight factor of instance based similarity w_i is

Table 4. 1 The best performances of our method and MIMapper on each of the IIMB test cases

Methods	Our Method			MIMapper		
Test Cases	Precision	Recall	F-Measure	Precision	Recall	F-Measure
002	0.9688	0.958	0.96336973	0.944	0.9436	0.9438
003	0.9637	0.9519	0.95776366	0.9304	0.9641	0.94695
004	0.9743	0.9576	0.96587782	0.944	0.9469	0.945448
005	0.9666	0.9528	0.95965039	0.9344	0.944	0.939175
006	0.969	0.9429	0.95577185	0.9325	0.9431	0.93777
007	0.9622	0.9584	0.96029624	0.939	0.9484	0.943677
008	0.9731	0.9579	0.96544018	0.9482	0.9535	0.950843
009	0.9632	0.9474	0.95523467	0.9446	0.9685	0.956401
010	0.9604	0.9539	0.95713896	0.9442	0.9614	0.952722
011	0.9379	0.8351	0.88351979	0.8368	0.8399	0.838347
012	0.9235	0.839	0.8792244	0.8368	0.8253	0.83101
013	0.9204	0.8327	0.87435637	0.8344	0.8433	0.838826
014	0.9281	0.8434	0.88372514	0.824	0.8447	0.834222
015	0.9391	0.8229	0.87716843	0.8366	0.8478	0.842163
016	0.9224	0.827	0.87209878	0.827	0.82	0.823485
017	0.9219	0.8322	0.87475649	0.8394	0.8283	0.833813
018	0.9229	0.8328	0.8755381	0.8277	0.8272	0.82745
019	0.9245	0.8335	0.87664477	0.8218	0.8247	0.823247
020	0.825	0.9639	0.88905752	0.7632	0.9598	0.850284
021	0.8445	0.9595	0.89833453	0.7721	0.9742	0.861455
022	0.8258	0.9763	0.8947656	0.785	0.9626	0.864776
023	0.8442	0.9586	0.89777027	0.7889	0.956	0.864449
024	0.8341	0.9629	0.89388413	0.7649	0.9665	0.853963
025	0.8391	0.9659	0.89804619	0.7713	0.9631	0.856595
026	0.8242	0.9596	0.88676121	0.7897	0.9539	0.864068
027	0.8378	0.9577	0.89374666	0.7816	0.9546	0.859481
028	0.8389	0.9557	0.89349909	0.7779	0.9497	0.855258
029	1	1	1	1	1	1
030	0.9755	0.8583	0.91315482	0.9092	0.8687	0.888489
031	0.952	0.8718	0.91013664	0.9017	0.8728	0.887015
032	0.9601	0.857	0.90562512	0.9294	0.8654	0.896259
033	0.9632	0.8694	0.91389947	0.9114	0.8638	0.886962
034	0.9579	0.8738	0.91391933	0.9141	0.8623	0.887445
035	0.9706	0.8643	0.9143709	0.9229	0.8696	0.895458
036	0.9781	0.8668	0.91909272	0.9276	0.8821	0.904278
037	0.9591	0.8596	0.90662821	0.9061	0.8784	0.892035

set to 0.71, and (iii) the threshold of relatedness is set to 0.739. The setups of MIMapper with respect to this set of results are as follows: (i) the weight factor of the class name semantic similarity is set to 0.23, (ii) the weight factor

of the MI matching is set to 0.77, and (iii) the threshold of relatedness is set to 0.595.

Table 4.2 summarises the performance of our method and MIMapper with regard to each of the four groups of IIMB test cases. The settings of both methods are the same as the aforementioned settings.

Table 4. 2 The performances of our method and MIMapper

Methods	Our method			MIMapper		
Test Cases	Precision	Recall	F-Measure	Precision	Recall	F-Measure
002-010	0.966811	0.953422	0.960060	0.940144	0.952611	0.946309
011-019	0.926744	0.833177	0.877448	0.831611	0.833466	0.832507
020-029	0.85136	0.96601	0.904586	0.7994	0.9640	0.873032
030-037	0.964562	0.865125	0.912103	0.9153	0.870388	0.892242
Average	0.927369	0.904434	0.913549	0.871629	0.905126	0.886023

According to the results presented in Table 4.2, our method has the best performance with the first group of test cases, which are numbered from 002 to 010. The differences between related entities across ontologies are mainly typographical variations. A possible reason is that, for this group of test cases, both the linguistic-based matching and the instance-based matching can work well. Our method has the least ideal results with the test cases numbered from 011 to 019. An explanation may be that the strategy applied for generating this group of test cases has significantly affected the performance of the instance-based matching of our method, since the mapping relations between the instances and their parent classes have been disarranged by the strategy.

According to the presented evaluation results, our newly proposed method outperforms MIMapper on providing suggestions on matching entities across ontologies. MIMapper is believed to have a better performance than other existing ontology mapping methods, such as PROMPT, Chimaera (Kaza and Chen 2008). Therefore, we believe that our method can also outperform other existing ontology mapping methods.

We noticed that, in a relatively recent paper Euzenat et al. (2009) proposed two newly enhanced methods that may return better mapping suggestions than MIMapper does. We cannot conclude that our method outperforms those methods. This may need to be further investigated in the future.

After carefully analysing, we believe the reasons of Euzenat et al.'s methods outperforming MIMapper are as follows: (i) those two algorithms employ more matching strategies than MIMapper. As we aforementioned in Section 4.3, there are five strategies can be applied to provide suggestions on matching ontology entities. MIMapper only used two out of the five strategies. However, applying more strategies makes the estimation process of those two methods becomes more complicated and, as a result, significantly lowers their efficiency. We can try to integrate all five strategies into our method in the future research. (ii) Euzenat et al.'s methods introduced may have been specially targeted to achieve better experimental results for the IIMB data. For example, those methods may have been engineered to deal with a certain type of artificial errors that have been introduced to IIMB data. However, both of the possible reasons require further investigations.

4.5 Summary

This chapter reports our research on developing a new ontology mapping method for the Ontology Mapping module of the proposed system. This method is a hybrid method applying two matching strategies: (i) the linguistic-based strategy, which uses the linguistic information of class names to estimate the relatedness between classes, and (ii) the instance-based strategy, which uses the most informative instances to help identify matches between classes. In order to achieve better results for the linguistic based matching, we use a sophisticated method to calculate the semantic similarity between the words in class names. Also, we treat class names as short sentences to estimate the semantic distances between class names. Furthermore, we utilise the number of concurrences of the instances and their class names in the

Internet to locate the most informative instance for each class. The Normalised Google Distance is used to quantify the concurrences into the informative level of the instances to their parent classes. According to the experimental results, we conclude that our method outperforms most of the existing methods.

Chapter 5.

Automatic Use Case Generating Module

This chapter reports our research on Automatic Use Cases Generating (AUCG) module of OKSSRA. The AUCG module is intended to create essential Use Cases automatically utilising the business processes retrieved from a business knowledge repository, i.e., the MIT Process Handbook (MITPH).

This chapter is organised as follows. Section 5.1 gives a brief introduction to our motivation of developing the AUCG module. Section 5.2 reviews previous research on generating Use Cases on the basis of business knowledge. Section 5.3 illustrates the design of our method for automatically generating Use Cases. The evaluation results and discussions are given in Section 5.4. Section 5.5 concludes this chapter.

5.1 Introduction

The concept of use case is proposed by Jacobson (1992) as a mechanism for establishing requirements of a software system through capturing the interactions between the external actors (e.g., the system users) and the system.

Nowadays, Use Cases have been widely recognised as a *de facto* standard to represent the functional requirements of software system (Dobing and Parsons 2000). The functional requirements define the specific functionalities that a software system is supposed to accomplish, such as data manipulation and processing (Wiegers 2003). Two of the most important reasons that make Use

Cases so popular are: (i) Use Cases bring a software development team a semiformal framework for presenting requirements, and (ii) Use Cases are a powerful platform for organising project information (Adolph et al. 2003).

A large amount of efforts has been contributed to support requirements analysts creating robust Use Cases or deriving Use Cases semi-automatically from existing business documentation, such as, business models (Dietz 2003, Santander and Castro 2002) or, business documentation (Dijkman and Joosten 2002). Down to the basic of these researches, they all try to transform business knowledge from business processes into Use Cases.

It has long believed that business processes and Use Cases resemble each other (Jacobson et al. 1994, Jacobson et al. 1999, Nurcan et al. 1998, OMG 2001). The underlying relations between them can be brought out by their definitions. A use case "*specifies a sequence of actions, including variants that the system can perform, and that yield an observable result of value to a particular actor*" (Jacoboson et al. 1999). A business process is "*a structured, measured set of activities designed to produce a specified output for a particular customer or market*" (Davenport 1993).

The resemblance between their definitions suggests that the transformation between Use Cases and business processes is possible. Both Use Cases and business processes concern about behaviours, i.e., actions or activities, which aim at delivering a particular result(s) to a certain party, i.e., an actor or a customer. The difference between Use Cases and business processes, according to their definitions, is mainly the way of organising the behaviours, i.e., a use case has its actions organised in sequences but a business process organises the activities in a structured and measured way. In other words, it is possible to derive Use Cases from business processes, if the activities in a business process can be reorganised into a sequence(s) with a carefully designed mapping rules.

The benefits of deriving Use Cases from business processes include: (i) it helps collect the requirements in a more efficient way; and (ii) it can reduce the errors in the newly created Use Cases (Dietz 2003).

The process of building Use Cases can be time consuming. The common method of building Use Cases is so called “waiter strategy”, which normally requires the requirements analysts to have a set of lengthy, and maybe tedious, interviews with business stakeholders to elicit the requirements (Dietz 2003).

The requirements elicitation process for Use Cases is full of uncertainties which can bring errors to the Use Cases and, eventually, to the system. The errors can be caused by (i) the misunderstanding happens at the meeting between requirements analysts and business stakeholders; and (ii) the requirements analysts’ lack of related business knowledge to understand business stakeholders correctly. Therefore, it is believed that eliciting requirements from business stakeholders is the weakest link in the progress of composing Use Cases (Dietz 2003).

Deriving Use Cases from business processes can support requirements analysts to efficiently create Use Cases with a higher accuracy. Business processes are normally stored as a form of business models or business documentation. Utilising such existing business knowledge as the basis to create Use Cases can significantly improve the efficiency of interviewing the business stakeholders. The stored processes have usually been well examined and validated, which means these processes are more reliable information sources for requirements analysts to use to compose Use Cases.

In this research, we propose an automatic method to generate user cases from business processes. Our method, based on a set of carefully designed mapping rules, transforms business processes into essential Use Cases that can support requirements analysts to create conventional Use Cases.

5.2 Previous research on generating Use Cases

A brief introduction to Use Cases and essential Use Cases is presented in Section 5.2.1. Section 5.2.2 reviews previous research on deriving Use Cases from business processes.

5.2.1 Use Cases

A use case is a collection of events showing that how the system is expected to respond users' requests and, eventually, help the users achieve their goals. Every use case is a small play script for the software system to follow when the system encounters the situations described in the use case or the initiating requests of the use case are triggered.

It is believed that requirements in a rigid formal style are not usable, as are those requirements without a structure (Dobing and Parsons 2000). Requirements elicitation requires efforts from both business stakeholders and software developers. However, those requirements structured in a formal style are difficult to read and be understood by business stakeholders, who usually do not have a technical background. As a result, business stakeholders may not be able to give their opinions on the documented requirements. If the requirements are presented casually without an explicit structure, it will be extremely difficult to validate the requirements.

Use Cases provide a framework for elaborating requirements in a semiformal way, which is suitable for all the parties in the development team (Wiegers 2003). The semiformal structure of requirements provided by Use Cases can (i) remind the analysts to collect all necessary information for developing the system, such as alternative scenarios, and possible exceptions; and (ii) allow business stakeholders without technique background to understand what the collected requirements are.

Use Cases provide scaffolding for gathering various kinds of project information into one document. The information may belong to different problem spaces and concern different groups of stakeholders of the project. For instances, (i) the data and business rules or relevant legal issues included in the Use Cases can give the developer ideas how the collected data need to be stored, and how long it can be legally kept in the database; and (ii) the pre-conditions and post-conditions presented in Uses Cases can help testers to construct test cases.

A use case normally consists of the following key components: (i) use case name, (ii) actor(s), (iii) description, (iv) precondition(s), (v) main flow, (vi) alternative flow, and (vii) post-conditions.

Use Cases present the functional requirements of a software system, which illustrate how a system behaves in the business context that the system is developed for. Therefore, understanding the related business processes is essential for analysts to write Use Cases showing how the system will be applied in the business context.

Use Cases are usually generated manually with an eight-step process: (i) identify actors; (ii) identify goals or activities of actors; (iii) identify key components for the use case; (iv) determine the name and a short description for the use case; (v) create a main flow for the use case; (vi) create alternative flows for the use case; (vii) compose the use case documents; and (viii) review and revise. In order to collect all the necessary information for composing Use Cases, it normally requires employing existing business models and/or repeatedly interviewing business stakeholders. The whole process can be seen as a process of transforming business requirements of business stakeholders into system requirements of the system.

The Use Cases that our module generates are in a text form. The basic form of Use Cases is written in natural language, although other forms can be

employed, such as using (i) sequence charts, (ii) UML diagrams, or (iii) a programming language (Cockburn 2001). Use Cases in a text form is believed to be easier for the business stakeholders to understand (Cox 2004), which is important for stimulating the discussion between the business stakeholders and requirements analysts.

Essential Use Cases

Essential Use Cases are first introduced by Constantine (1997) for describing the intended interactions between actors and systems at an abstract level (Unhelkar 2005). Table 5.1 (Constantine and Lockwood 1999) is an example showing the differences between conventional Use Cases (also known as *system Use Cases*) and essential Use Cases (also known as *business Use Cases*).

Table 5. 1 Examples of the conventional use case and the essential use case for "getting cash from ATM" scenario

Conventional Use Case			Essential Use Case	
Actor	SYSTEM		Actor	SYSTEM
Insert card			Identify self	
	Read magnetic stripe			Verify identity
	Request PIN			Offer choices
Enter PIN			Choose	
	Verify PIN			Dispense cash
	Display trans options		Take cash	
Press key				
	Display account menu			
Press key				
	Prompt for amount			
Enter amount				
	Display amount			
Press key				
	Return card			
Take card				
	Dispense cash			
Take cash				

Essential Use Cases are normally used at the beginning phase of the requirements elicitation as a set of conceptual Use Cases. Essential Use Cases normally focus on the business context of the software system. The software system is normally treated as a black box in essential Use Cases. However, essential Use Cases can be extended into conventional ones after the analysts gaining a better understanding on the system.

The Use Cases derived from the MITPH business processes are essential Use Cases, since the business processes normally only include abstracted knowledge about the business operations instead of the details of how the concern actors interact with each other. The generated essential Use Cases may not be used directly for developers to develop the software, but are great resources for requirements analysts to use as a frame for composing the conventional Use Cases.

5.2.2 Current research in generating Use Cases from business processes

In this section, we will review previous research on deriving Use Cases from business processes.

It is a complicate procedure to elicit and compose Use Cases. This procedure normally requires the involvement of a group of: (i) at least one person who has programming background to ensure that the granularity of Use Cases can meet the requirements of developing the system; (ii) at least one person who is familiar with the business domain of the system; and (iii) at least one person who knows how the system will be used (Adolph et.al. 2003).

The three following researches aim at reducing the complication of the procedure of eliciting and composing Use Cases through making the procedure become a semi-automatic one.

Santander and Cast's Method

Santander and Castro (2002) proposed a method for deriving Use Cases from a set of organisational models, which are developed under the i* framework (Yu 1995). In this research, a set of detailed guide lines are used to illustrate the correlations between the components of the organisational models and the elements of Use Cases.

The i* framework is a modeling language that aims at presenting causalities between system requirements (e.g., motivations, intentions, and rationales) in an organisational environment. It provides two models for describing those causalities: (i) the Strategic Dependency (SD) Model, and (ii) the Strategic Rationale (SR) Model.

The SD Model is used to model relationships among the involved actors. It is mainly composed of a set of actors and four types of dependency links: (i) goal dependency, which indicates that one actor depends on another to fulfill its goal; (ii) resource dependency, which describes that one actor need physical support or information from another; (iii) task dependency, which indicates that one actor need another to partially fulfill its goals; and (iv) softgoal dependency, which means that one actor has another as "non-functional requirements".

The SR Model is a supplementary model to the SD model. It is normally utilised to model those internal relationships associated with each actor. For example, a SR Model can be used to illustrate how an actor can accomplish a task by finishing several sub-tasks. In addition to the four types of dependency links included in the SD Model, the SR Model has two more types of links: (i) means-ends, which stands for that one of the end nodes is the method to achieve the another end node; and (ii) task-decomposition, which denotes that one of the end nodes is one of component tasks of the another end node.

Santander and Castro (2002) suggested a guide line for utilising actors and dependency links in the i* framework to derive Use Cases. The guide line can be briefly described as follows (Santander and Castro 2002):

1. Identifying system actors;
2. Composing Use Cases according to the dependencies related to the actors identified at Step 1;
3. Identifying and describing scenarios of Use Cases.

Dietz's Method

Dietz (2003) proposes a method creating Use Cases from business processes using a modeling method named Demo Engineering Methodology for Organisations (DEMO). He suggested that the DEMO model should be employed as a bridge to explicit the procedure of transforming business processes into Use Cases.

Dietz (2003) believes that the DEMO method is useful for deriving Use Cases from business processes. In the DEMO theory, an organisation consists of a set of actors and transactions. The *actors* are social individuals or subjects, who perform two kinds of acts: (i) *production acts*, which fulfill the organisation's missions, (ii) *coordination acts*, which involve initiation and coordination for executing the production acts. A *transaction* is used to describe the interactions between actors in DEMO. Each of the transaction is illustrated as a recurrent pattern of coordination acts and production acts.

The process of deriving Use Cases from business processes in Dietz (2003)' method can be seen as a five-step process: (i) construct the DEMO model frames; during this step, the actors, the transaction types, and the relationship between them are identified; (ii) enrich the DEMO models; transactions in the models are enrich with the patterns of coordination acts and production acts; (iii) transform the DEMO models into use case diagrams; (iv) identify the

relationship among Use Cases and their supporting components, such as required database; (v) determine the scope of the system.

The mapping rules used by Dietz (2003) for transforming the DEMO models into Use Cases are: (i) the actors in DEMO models are the actors of Use Cases; (ii) each of the transactions in the DEMO model corresponds to a use case; (iii) each of the acts in a transaction shall be mapped into an event in the corresponding use case.

However, the DEMO method may become the barrier that stops analysts to use Dietz (2003)'s method. Building robust DEMO models on the basis of the business processes is the corner stone for Dietz (2003)'s method of deriving robust Use Cases. However, mastering the DEMO model can be a time consuming process for requirements analysts, which requires them to have a good understanding of the DEMO theory and a good knowledge on how to break the business transactions into acts.

Dijkman and Joosten's Method

Dijkman and Joosten (2002) proposed a method utilising a set of mapping rules between business process meta-models and use case diagram meta-models to transform the business processes into the use case diagrams. Their method is based on the observation that Use Cases and business processes both consist of a collection of business activities. They believe that the difference between Use Cases and business processes is only the way of organising the business activities.

Dijkman and Joosten (2002) use a simplified version of the meta-model introduced in the UML specification as their use case diagram meta-model. The components of this meta-model are as follows: (i) *actor*, (ii) *interaction*, (iii) *include* relationship, (iv) *generalisation* relationship, and (v) *extend* relationship.

The business process meta-model in Dijkman and Joosten (2002)'s method is first introduced by van Dommelen et al. (1999). This meta-model consists of the following components: (i) *task*, which is a smallest unit of work in a process, (ii) *role*, which is a group of entities who perform a task of a group of tasks with similar rights and obligations, (iii) *branches*, which is a choice of the succeed task after the current task, (iv) *transitions*, which connect tasks with their adjacent tasks, and (v) *guard*, which is a condition for the current task proceeding to a certain succeed task.

In order to ensure a derived use case contains proper number of events, i.e., business tasks, Dijkman and Joosten (2002) introduced the concept of steps in their method. A *step* stands for a proper sized sequence of tasks that are contiguously performed by the same role. Each step can make business sense and contains a proper amount of business tasks for deriving Use Cases. If business tasks retrieved from business documentation are first organised as a number of task collections, each of the collections normally contains too many tasks to be included into one use case. Therefore, Dijkman and Joosten (2002) suggest breaking the big business task collections into smaller groups, i.e., steps.

Dijkman and Joosten (2002) believe that the association between a step and the role who performs the step can be mapped as the association between a use case and its actor.

The mapping proposed by Dijkman and Joosten (2002) are illustrated in Table 5.2.

Dijkman and Joosten (2002)'s method is based on their validated assumption. Under their assumption, if a group of proper organised business tasks can be mapped into a use case, the roles conducts the group of business tasks can be mapped to the actors of the corresponding Use Cases.

Table 5. 2 The mapping proposed by Dijkman and Joosten (2002)

Business Process Concept	**Use Case Concept**
Role	Actor
Step	Use Case
Association Between Role and Step	Association between Actor and Use Case
Task	Interaction
Task in a Step	Interaction in a Use Case
Transition between Tasks in the same Step	Ordering between Interactions in the same Use Case
Guard on Transition	Constraint on Interaction
Alternative Path through a Branch	Alternative Path Description of a Use Case

The limitation of Dijkman and Joosten (2002)'s method is that each generated use case can only have one actor. This limitation is brought by the definition of the steps in Dijkman and Joosten (2002)'s model. They believe that all the tasks in one step are performed by only one role. According to their mapping rules, the derived use case from a step will then only have one actor, i.e., the role of the step.

Though Dijkman and Joosten (2002)'s method has its own limitations, their validated assumption inspired us that it is rational to map a group of business tasks into a use case, as long as the group of business tasks (i) can make a business sense, and (ii) includes a proper amount of information.

In order to automatically generate Use Cases from the ontologised business processes, we need first establish a set of mapping rules between the elements in the business processes (e.g., a certain type of relations or classes) and the components of Use Cases (e.g., actors, goals, and scenario). One method to build up the set of rules is modifying Santander and Castro's guide lines. An

alternative method is to develop our own set of rules. In this research, we propose our own set of mapping rule between business processes and Use Cases.

5.3 Automatic Use Cases generating in OKSSRA

According to the review on earlier research, the procedure of driving a use case from a business process is essentially a procedure to respectively correlate a proper segmented business process and the associations between the business process components with a use case and the associations between the use case elements.

According to the validation provided by Dijkman and Joosten (2002), it is feasible to derive a use case from a complete business process. However, in order to ensure that the generated Use Cases have a rational complexity, the business processes need to be segmented into a proper size in prior to conducting the transformation procedure.

Adolph et. al. (2003) believe that there is no best format for Use Cases, and the amount of details being put into a use case is determined by the need of the project and the development team, and the purpose of writing the use case.

Different methods have been developed to segment business processes for deriving Use Cases without jeopardising the completeness of the business processes. Santander and Castro (2002) utilised the actors and the dependency links in the *i** framework as the indicator to segment business processes stored in *i** organisational models. In Dijkman and Joosten (2002)'s method, the time continuity among business tasks is used to group tasks into proper sized processes, i.e., steps. Dietz (2003)'s method uses the DEMO method to group business tasks into transactions in a DEMO model. Each

transaction is considered as a proper sized segment of a business process for generating Use Cases.

We believe that the MITPH business processes are well segmented for deriving a use case. Each of those business processes is completed and describes how a certain business goal can be reached in a common way and, if applicable, one or multiple alternative ways.

The complexity of generated Use Cases can be controlled by limiting the depth of searching in MITPH. The business processes in MITPH are organised in the following way: (i) each *process* is described as a collection of several tasks, which show how the process is executed step by step; (ii) each of the *task*s of the process can be presented as a sub-process, which shows how the task can be accomplished; and (iii) each of the task of the *sub-processes* can be recursively broke into another set of downstream sub-processes, till each of the tasks in the downstream sub-process is a simple task and can be described by their own name.

The completeness of a process will not be jeopardised by limiting the searching depth in MITPH, as each process itself is a complete process for achieving a certain goal or accomplish a certain task. By limiting the searching depth, we only set a constraint to the granularity of the events elaborated in the use case, which, as a bonus effect, can ensure that the events in the use case are in the same level of detail.

From the business process elements to the use case components

We map the base process's *name* to the name of the use case. In the MITPH, the name of a business process (we will use *process* as a brief name for the MITPH business process in the rest of the chapter) describes its goal in a clear and brief way. The derived use case is supposed to have the same goal as the

base process from which the use case is derived. Therefore, the name of the base process can employed as the name of the derived use case.

The *description* of the base process can be used as the description of the derived use case. The reason is that, as stated above, the use case and the base process share the same goal. Thus, the description of the goal of the base process is also the goal of the derived use case.

The *is-performed-by* relationship is employed to identify the actor(s) of the use case. It is a common understanding that, for the Use Cases derived from business processes, the person or the party who performs in the source process is normally the actor of the derived use case (Dietz 2003, Dijkman and Joosten 2002).

The *is-enable-by* relationship can be used to identify the pre-condition(s) of the base process or the alternative paths for a certain event in the main flow.

If there are entities related to the base process by the *is-enable-by* relationship, it normally indicates those entities, e.g., resources or processes, are the necessary conditions to apply the base process. The association between the base process and these entities is similar to the association between a use case and its pre-conditions. Therefore, the *is-enable-by* relationship can be also used to identify preconditions of the derived use case.

However, if a MITPH entity is connected to an step in the main flow by the *is-enable-by* relationship, the *is-enable-by* relationship can then be utilised to derive alternative flows. An *is-enable-by* relationship normally indicates how a step, from which the relationship is originated, can be executed by the way or under the condition, to which that the relationship is pointed.

The *requires* and *has-input* relationship can also be used to derive the pre-conditions of Use Cases. These two relationships both indicate the entities that

are required for initiating a process. This kind of initiation requirements is the same as the pre-conditions in Use Cases. Thus, we consider the resources connected to the base process by the *requires* and *has-input* relationship as pre-conditions of the derived use case.

The *has-part* relationship is used in MITPH to break larger processes into a group of smaller and more specific sub-processes, i.e., events of the base process. These sub-processes illustrate how the base process is executed step by step, which are equivalent to the steps in the main flow of the derived use case.

The post-conditions of a use case are the outcomes or outputs after the system running the whole use case. The *has-output* relationship in MITPH indicates a similar association between the post-conditions and the use case. Thus, we use the *has-output* relationship to identify the post-conditions of the derived use case.

The *has-exception* relationship in MITPH is used to define the exceptions in the derived Use Cases. The exception flows in Use Cases describe how the system is supposed to react when errors occurs. Therefore, the entities connected to the bass process by the *has-exception* relationship are gathered to compose the exception flow of the derived Use Cases.

Based on the above observations, we propose the following mapping rules for our AUCG module to derive Use Cases from business processes automatically.

Mapping Rules for the AUCG Module

Given we already have the unique ID number of the base processes in MITPH, we developed a set of mapping rules as follows for deriving Use Cases on the basis of the MITPH business processes.

1. Building the basic elements of the use case.
    a. Retrieving the base process using the given ID number;
    b. Determining the name of the use case, which is the *name* of the base process;
    c. Composing the description of the use case, which is the *description* of the base process;
    d. Determining the actor(s) of the use case, which is the entities connected to the process by *is-performed-by* and *has-handler* relationships;
    e. Determining the pre-condition(s) of the use case, which can be the MITPH entities that are connected to the base process:
        i. The goal(s) that is connected to the base process by the *requires* relationship;
        ii. The resource(s) that is related to the base process by the *has-input* relationship;
        iii. The resource(s) that is related to the process by the *is-enabled-by* relationship;
    f. Composing the main flow of the use case, which consists of the entities that connect to the base process by the *has-part* relationship;
    g. Determining the post-condition(s) of the use case, which are the entities related to the base process by the *has-output* relationship;
2. Composing the alternative flow(s) by examining each step of the main flow and apply the following rules:
    a. Creating alternative steps on the basis of *is-enable-by* relationship: if there are *N* resources connected to a main flow step through the *is-enabled-by* relationship, there will then be *N* different ways to accomplish the step. It means that there should be *N-1* alternative ways to commit the step;
    b. Retrieving the implementation details using the *has-part* relationship: if a main flow step has *has-part* relationship towards

a set of sub-processes, these sub-processes should be listed as the implementation of the sub-process;

3. Constructing the exception flows: if there are processes are related to the base process with the *has-exception* relationship, these processes should be considered as parts of the error course.

5.4 Evaluation and Discussion

This section reports the experimental set ups and results for evaluating the Use Cases automatically derived by AUCG module. Three hypotheses are tested in this evaluation: (i) whether the generated essential Use Cases are helpful for users to understand the business background and relevant knowledge, (ii) whether the generated essential Use Cases are helpful for users to create conventional Use Cases, and (iii) whether the generated essential Use Cases are in a good quality.

5.4.1 Research on evaluating the use case quality

Various researches have been conducted on evaluating the quality of Use Cases. Those Use Cases of interest are normally generated manually after applying a set of predefined restriction rules on composing use case descriptions.

Achour et al. (1999) evaluated the quality of Use Cases from four aspects of the Use Cases: (i) the completeness, (ii) the precision, (iii) the accuracy, and (iv) consistency.

Cox and Phalp (2000) replicated Achour et al. (1999)'s experiment and concluded that it is difficult to measure the experimental results with the four aforementioned facets. Thus, they proposed to use the four following characters to define Use Cases' quality: (i) the plausibility, (ii) the readability, (iii) the consistent structure, and (iv) the alternative flow.

Anda et al. (2001) proposed a marking schema utilising seven aspects of the correctness of Use Cases to evaluate their quality. The seven aspects are as follows: (i) single diagram, (ii) actors, (iii) Use Cases, (iv) content, (v) granularity, (vi) realism, and (vii) consistency.

Phalp et al. (2007) proposed a set of seven criteria for examining the quality of Use Cases. These seven criteria are as follows: (i) coverage, (ii) cogent, (iii) coherent, (iv) consistent abstraction, (v) consistent structure, (vi) consistent grammar, and (vii) consideration of alternatives.

In summary, the following five criteria have been commonly employed to evaluate the quality of Use Cases by previous researches: (i) the understandability, (ii) the coverage, (iii) the level of detail, (iv) the consistency of structure, and (v) the consideration of alternative flows. Thus, we use the five criteria as the guideline to design the part of our questionnaire for evaluating the quality of the Use Cases derived by AUCG module.

5.4.2 Experiment design

This section introduces the design of the experimental set ups for evaluating the Use Cases derived by the AUCG module.

Questionnaire

A questionnaire is designed for investigating the helpfulness and the quality of Use Cases derived by AUCG module. The questionnaire consists of seven questions in the following two parts: (i) two questions for investigating the helpfulness of the derived Use Cases, and (ii) five questions for examining the quality of the derived Use Cases.

The two questions in the first part of the questionnaire are intended to evaluate the helpfulness of the generated Use Cases. Participants are asked to compare

the derived Use Cases and the corresponding business processes. The following participants' opinions are collected: (i) which style of the business knowledge is easier to be understood, and (ii) which style of the business knowledge is easier to be used for composing the conventional Use Cases.

The second part of the questionnaire has five questions for evaluating the quality of the derived Use Cases. The five questions are focus on investigating the following five quality criteria of the Use Cases: (i) the understandability, (ii) the coverage, (iii) the level of detail, (iv) the consistency of structure, and (v) the consideration of alternative flows. As mentioned in Section 5.4.1, these five aspects are the common interests of the previous research on evaluating the use case quality.

A classic five-point Likert scale (Likert 1932) is employed to record the participants' opinion of each question.

Participants

Ten intermediate to senior software engineers / requirements analysts were invited to participate this evaluation. Six of the participants have two to five years experiences as software engineers / requirements analysts. Four of the participants have more than five year experiences in software industry.

Procedures

Three example business processes are selected and retrieved from MITPH. AUCG module is then used to automatically generate three corresponding Use Cases on the basis of the three business processes.

Before commencing the experiment, the purpose of the evaluation was briefly introduced to the participants. A short description of the procedure is also provided to the participants.

The three retrieved business processes were first given to the participants. The participants are asked to read through the business processes and try to understand the content. After finishing the reading, the participants are required to compose a use case on the basis of each of the given business processes.

The three derived Use Cases are then provided to the participants. After reading through the provided Use Cases, the participants are asked to revisit the Use Cases that are composed by themselves on the basis of the given business processes.

Following the revisiting of their own Use Cases, the participants are asked to answer the two questions in the first part of the questionnaire. These two questions compare the helpfulness of the derived Use Cases by AUCG module with the corresponding MITPH business processes.

After completing the first two questions, the participants are asked to revisit the three Use Cases derived by AUCG module. The participants are then required to answer the five questions in the second part of the questionnaire, which are designed to investigate the quality of the derived Use Cases by AUCG module.

At the end of the evaluation, a brief interview with each participant is conducted to collect further information with regard to their answers.

5.4.3 Experimental results and discussion

This section presents the experimental results of the evaluation that investigates the quality of the Use Cases derived by AUCG module automatically.

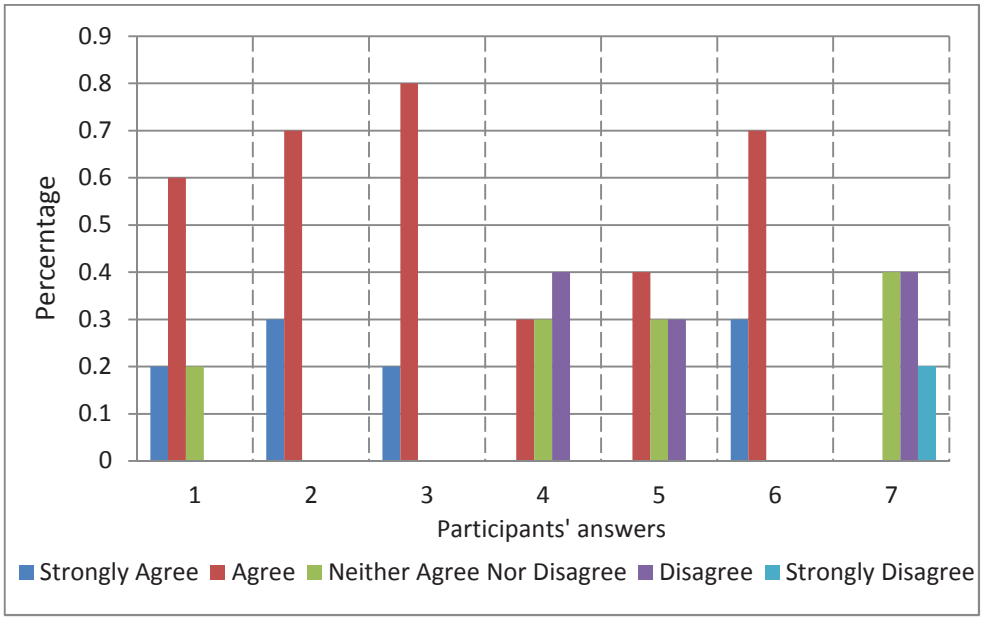

Figure 5. 1 Distribution of paticipants' opinions about the generated Use Cases by AUCG module

The statistics of participants' opinions about the helpfulness and the quality of the Use Cases derived by AUCG module are presented in Figure 5.1. According to the answers of the first two questions of the questionnaire, all participants think that the MITPH business processes are harder to be understood than the derived Use Cases. To some extent, all participants believe that the derived Use Cases are more helpful for composing conventional Use Cases comparing with the corresponding business processes.

The third to the seventh question of the questionnaire ask the participants' opinions of the quality about the derived Use Cases. All the participants believe that (i) the derived Use Cases are easy to be read and understood, and (ii) the derived Use Cases have a good consistency in structure. 60% of the participants think that the derived Use Cases provide reasonably good descriptions on the alternative events. 40% of the participants are happy with the coverage and the detail level of the derived Use Cases, though 30% of the participants hold an opposite opinion.

In order to investigate the participants' opinion by individuals, we calculate the average score of participants' answers to quantify their attitudes toward the derived Use Cases. A score is assigned to each Likert item (Likert 1932) depending on the attitude that the Likert item stands for. The Likert items representing the strong positive attitude to the strong negative attitude are correspondingly assigned a score from one to five. Thus, a lower average score of a participant's answers represents that the participant has a stronger positive attitude toward the quality of the derived Use Cases.

Table 5. 3 The average scores of subjects' opinions in AUCG module evaluation

Participant number	Average score of answers			Participant number	Average score of answers		
	Q1-Q2	Q3-Q7	Total		Q1-Q2	Q3-Q7	Total
1	1	1.6	1.43	6	2	2.4	2.29
2	1.5	1.6	1.57	7	2	2.6	2.43
3	1.5	1.6	1.57	8	2	3	2.71
4	1.5	1.8	1.71	9	2.5	3	2.86
5	2	2.4	2.29	10	2.5	3	2.86

Table 5.3 illustrates the average scores of all ten participants' answers. The average scores of (i) Question 1 to Question 2, (ii) Question 3 to Question 7, and (iii) all the seven questions, respectively relect the participants' opinions toward the following aspects of the derived Use Cases: (i) the helpfulness, (ii) the quality, and (iii) the overall feeling. According to the above definitions of the score of the Likert items, an average score of 3 means a neutral attitude, and an average score less than 3 means a positive attitude.

We can observe from the average scores in Table 5.3 that (i) all participants are happy with the helpfulness of the derived Use Cases, (ii) no participant is holding a negative attitude against the quality of the derived Use Cases, and (iii) generally, all participants are holding a positive attitude towards the derived Use Cases.

5.5 Summary

In this chapter, we proposed a new method for automatically deriving essential Use Cases using the MITPH business processes. A set of new mapping rules are introduced to explicitly map the elements of the MITPH business processes to the components of essential Use Cases.

According to their definition, the Use Cases and the business processes are two different ways of organising the business activities. It has long been believed that it is possible to map the elements of business processes to the components of Use Cases, as long as the business process elements are organised in a proper size. We first identified the MITPH business processes are in a proper size for deriving Use Cases. An analysis was presented to describe how the relationships among the MITPH entities can be employed to derive Use Cases automatically from the corresponding MITPH business processes.

Three pairs of business processes and the corresponding Use Cases are used to evaluate the helpfulness and the quality of the Use Cases automatically derived by AUCG module. Ten participants were invited participate the evaluation. According to the experimental results, the derived Use Cases by AUCG module are believed to be helpful and in a reasonable good quality.

Chapter 6.

System Evaluation

This chapter reports the experiments for evaluating OKSSRA. The hypotheses that we tested in these experiments are as follows: (i) whether OKSSRA is effective in helping junior requirements analysts and/or programmers to retrieve the relevant MITPH business processes and to gain the business knowledge that they do not have, (ii) whether OKSSRA is more helpful in retrieving relevant business processes from MITPH compared with the online MITPH, (iii) whether OKSSRA is easy to use. Potential further improvements of OKSSRA are also investigated in this evaluation.

This chapter is organised as follows: Section 6.1 describes the design of the evaluation. Section 6.2 presents and discusses the experimental results. In Section 6.3, we summarise the potential further improvements of OKSSRA and conclude this chapter.

6.1 Experiment design

6.1.1 Scenarios for test use

Three scenarios are designed for the test use of OKSSRA (The details of these scenarios are presented in Appendix A). In these three scenarios, we assume that the subject is a junior requirements analyst working for a software company, which develops bespoke software tools or systems for its clients. The key task of the subject at work is assumed to be eliciting and collecting the requirements for the system to be developed from the clients, i.e., business stakeholders. In order to make her or his meetings with the clients be more effective, we assume that the subject normally prepare herself/himself with

the preliminary business knowledge related to the software system, so that she or he can always ask clients the right questions for eliciting the requirements.

The three test use scenarios are meant to demonstrate that the effectiveness of OKSSRA using expanded queries to retrieve the MITPH business processes (For the rest of this chapter, we will use *process* to stand for the MITPH business process). It is difficult for junior users to use the online MITPH to retrieve the desired processes. This drawback is brought by the fact that the online MITPH does not process users' queries, and it only returns those results that are rigorous match to users' queries on the linguistic level. Each of the three scenarios is designed to help the subjects realise at least one major difference between OKSSRA and the online MITPH with regard to retrieving the relevant processes.

The first scenario is designed to test OKSSRA's ability to conduct the basic natural language processing on queries, such as, port trimming. In this scenario, the subjects can easily realise that the processes of interest are related to hiring employees. With the online MITPH, the subjects can get the desired processes only if they use the exact phrases such as "hire employee", or "hire part time employee". Otherwise, the online MITPH will return no results if the queries are in slightly different styles, such as using the plural nouns, the gerunds, or the verbs in past tense. However, OKSSRA can apply natural language processing techniques to users' queries, so that users can still get the desired processes that do not rigorously match the users' queries on the linguistic level.

The second scenario is aimed at examining OKSSRA's ability to conduct basic expansions on users' queries according to semantic meanings, such as expanding the queries using synonyms. This ability enables OKSSRA to retrieve more desired processes for users compared with only using linguistic based match. In this scenario, for example, if the subject uses a phase of *sell advertisements using the Internet* as the query, the online MITPH will return

no results. However, OKSSRA will first trim the query down to three primary key words *sell, advertisements,* and *Internet*. After measuring the semantic distances with the words in the repository, the system will expand the key words with a set of synonyms of the primary key words, such as *ads*, and *web*. Therefore, the process with a name of *sell ads on the web* can be included in the search results of OKSSRA.

The third scenario is targeted at investigating the OKSSRA's ability to conduct sophisticated language processing tasks on queries, such as expanding the queries using semantic similarity measures. In this scenario, it is quite possible that the subjects will not get any desired process using the online MITPH, unless they are reasonably familiar with MITPH, or have extensive knowledge and experiences in the business area. The obvious key words for this scenario are as follows: *manage, business*, and *relationship*. The online MITPH will not return any desired processes if the subjects use any combination of these three key words as their queries. However, after expanding the key word of *business relationship,* OKSSRA will retrieve not only the commonly known relevant processes such as *manage customer relationship,* and *manage financial relationship,* but also the uncommonly known relevant processes such as *manage regulatory relationship,* and *manage virtual relationship*. All these search results returned by OKSSRA can provide the subjects a shortcut to get familiar with the scope of managing business relationships.

6.1.2 Questionnaires

All subjects were required to fill a questionnaire sheet right after the test use of the online MITPH and OKSSRA. There were two parts in the questionnaire sheet, and each part includes six questions.

The first part of the questionnaire aims at investigating the effectiveness and usefulness of OKSSRA. The first three questions of this part of the question-

naire collect information about if the system itself and the query expansion features can help subjects gain the preliminary business knowledge. The rest three questions of this part of the questionnaire ask the subjects to compare OKSSRA with the online MITPH. The subjects were asked to point out which of the two systems is easier to use and more effective in retrieving desired results.

The second part of the questionnaire aims at examining the user experiences of OKSSRA. The first five questions of this part are chosen and modified from Software Usability Measurement Inventory (SUMI, Colbert 2005). SUMI is a de facto industry standard for evaluating the usability of software based on end users' points of view (Colbert 2005). These five questions are targeted at investigating users' opinions on the following five aspects of OKSSRA, respectively: (i) learnability, which measures whether it is easy for users to master the system; (ii) helpfulness, which investigates whether the system is self-explanatory; (iii) control, which measures whether the users feel having the control of the system; (iv) efficiency, which investigates whether the system can assist users on their tasks; and (v) affect, which measures how the users emotionally think about the system. The sixth question of this part is designed to collect users' suggestions on improving OKSSRA in the future.

The details of the questionnaire are presented in Appendix B.

6.1.3 Participants

30 senior university students voluntarily participated in this evaluation. 25 of them were majoring in Computer Science, and the rest of the subjects were majoring in Software Engineering.

22 of the participants are second or third year undergraduate students. Five of the participants are studying towards a postgraduate diploma or a master's degree. Three of the participants are PhD candidates or have a Doctoral

degree. All of the 22 undergraduate subjects have taken at least one Software Engineering paper by the time of participating in this evaluation. All of the postgraduate participants have Software Engineering background or experiences in programming or software developing.

6.1.4 Procedures

The subjects were invited to conduct the test use of OKSSRA in a computer lab and complete the questionnaire afterwards. Before commencing the test use of OKSSRA, the purposes of the evaluation were briefly explained to the subjects. A brief demonstration of using both OKSSRA and the online MITPH was presented to the subjects.

A description of the three test use scenarios was first given to each of the subjects. The background setup of the given scenarios was explained to the subjects. The subjects were assumed to be a junior requirements analyst working for a software company, which develop bespoke tools/systems for its clients. The details of the background setups have been presented in Section 6.1.1.

The subjects were asked to use the online MITPH and OKSSRA, respectively, to retrieve relevant processes for the given scenarios. For each scenario, the subjects were required to conduct no more than three searches first on the online MITPH and then on OKSSRA, respectively, using the same queries.

After completing the task in each given scenario, the subjects were asked to answer the following two questions: (i) whether they have retrieved relevant processes using one or both of two systems, and (ii) which of the two systems is more helpful with regard to the current scenario.

The subjects were required to fill in the questionnaire after the test use of OKSSRA and the online MITPH. A short interview with each subject was

conducted after them completing the questionnaire to ensure all the stated opinions have been clearly understood.

6.2 Experimental results and discussion

This section presents the experimental results of the OKSSRA evaluation. Each subset of the results is followed by a brief discussion.

Figure 6.1 illustrates the statistics of how the subjects retrieved relevant processes from MITPH for each scenario. All subjects retrieved the relevant processes using OKSSRA. For Scenario 1, 40% of the subjects cannot find the relevant processes using the online MITPH. This percentage increases to 60% and 73.3% for Scenario 2 and 3, respectively. In the meantime, the subjects found that it is more difficult to retrieve the relevant processes for Scenario 3 than for Scenario 1 and 2. All the subjects believed that OKSSRA, compared with the online MITPH, is more effective for accomplishing the tasks, when the familiarity of the repository and relevant business knowledge is required.

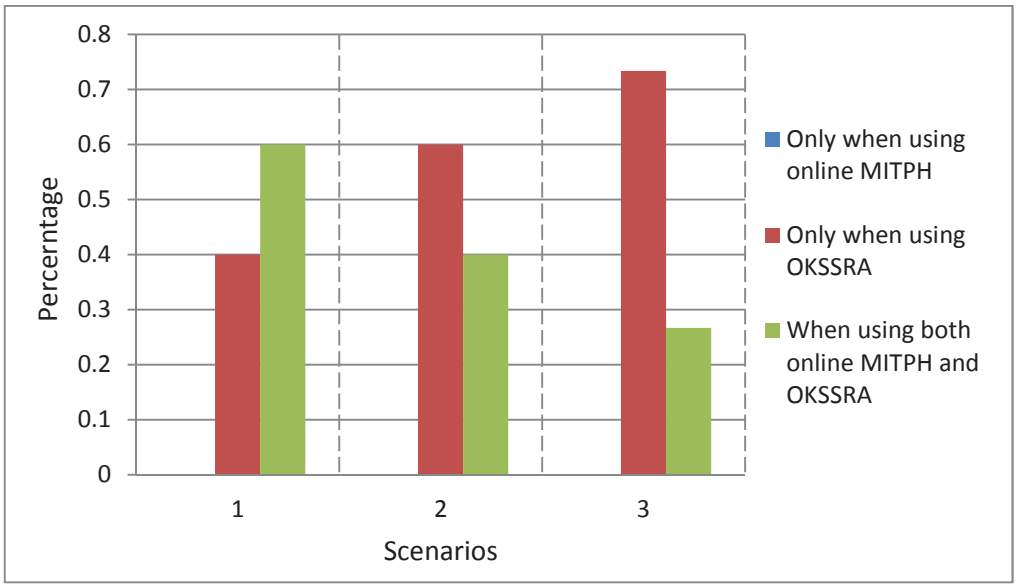

Figure 6. 1 The statistics of how subjects retrieved relevant processes

The distributions of the subjects' answers for the first part of the questionnaire are presented in Figure 6.2. For Question 1, 3, and 5, if a subject chooses *strongly agree* or *agree* as her or his answer, the subject is in favour of OKSSRA. For Question 2, 4, and 6, if a subject chooses *strongly disagree* or *disagree* as her or his answer, the subject is in favour of OKSSRA. According to the statistics shown in Figure 6.2, there was no subject holds a negative opinion against OKSSRA.

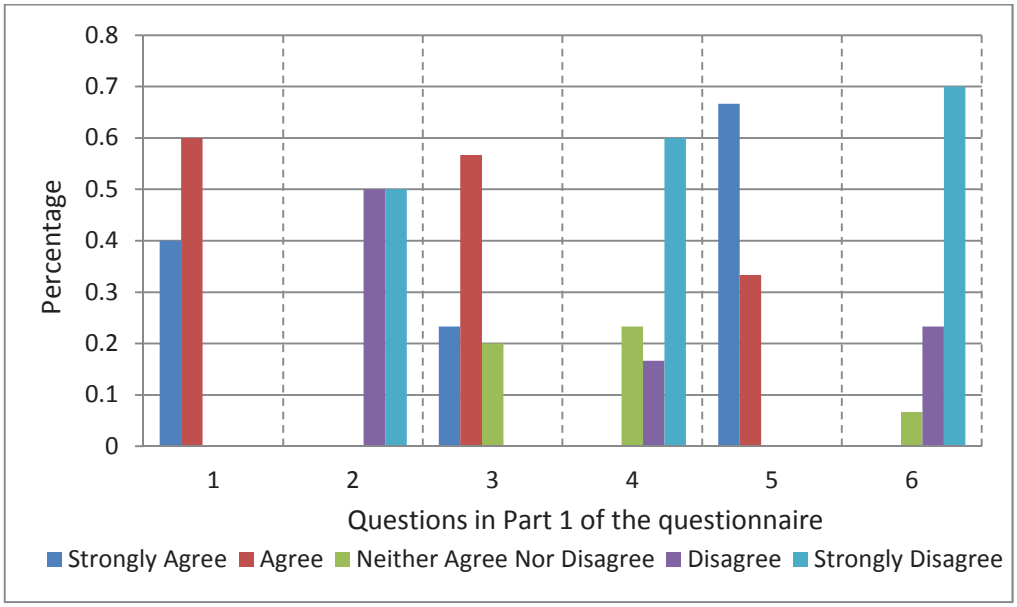

Figure 6. 2 Distributions of subjects' answers for questions in Part 1 of the questionnaire

According to the answers of the first two questions in this part, all the subjects found that: (i) OKSSRA is useful for gaining preliminary business knowledge, and (ii) the expanded key words provided by OKSSRA are helpful for retrieving the relevant processes. 80% of the subjects found that they can always found useful information using OKSSRA. According to the answers of the fifth questions of this part, all the subjects believed that it is more convenient to use OKSSRA to retrieve the relevant processes compared with using the online MITPH. A majority, 60% and 70%, respectively, of the subjects strongly believe that OKSSRA is easier to use, and more useful for retrieving the relevant processes compared with the online MITPH. Based on

above observations, we can conclude that all subjects believe that OKSSRA is effective and useful for them to gain business knowledge.

From the experimental results of the first part of the questionnaire, we have the following observations: (i) OKSSRA is commonly believed to be useful for gaining preliminary knowledge, and (ii) OKSSRA is more user-friendly than the online MITPH.

The distributions of the subjects' answers for the first five questions in Part 2 of the questionnaire are illustrated in Figure 6.3. For Question 3 and 5 in this part, *strongly disagree* and *disagree* are the answers in favour of OKSSRA. For the other three questions in this part, *strongly agree* and *agree* are the answers in favour of OKSSRA. From the subjects' answers, we have the following observations: (i) 90% of the subjects found that OKSSRA is easy to use; (ii) all the subjects believed that OKSSRA is self-explanatory, and are in favour of using OKSSRA; (iii) 93.3% of the subjects believed that they have a good control of OKSSRA; (iv) OKSSRA is able to effectively help 83.3% of the subjects to accomplish their tasks. Based on the above observations, we

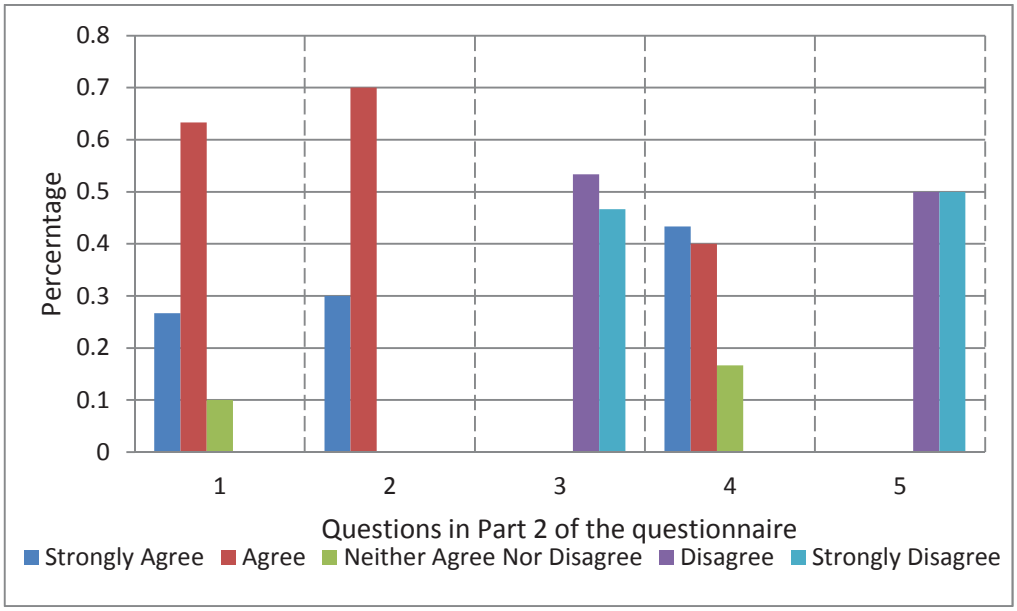

Figure 6. 3 Distributions of subjects' answers for questions in Part 2 of the questionnaire

can conclude that all users are pleased with the functions of OKSSRA and enjoyed using the system.

In order to analyse each of the subjects as an individual, the Mean Absolute Deviation (MAD) of their opinions from the neutral opinion are calculated and presented in Table 6.1. MAD can be calculated using the following equation (Walker 1931):

$$MAD = \frac{1}{n}\sum_{i=1}^{n} x_i - \bar{x}$$

where *n* is the number of the questions; x_i is the score of subject's answer to the *ith* question; $\bar{x}$ is the average score of all *n* questions. In our case, *n* has a value of 6 and 5, respectively, for Part 1 and Part 2 of the questionnaire. We define the answers that are in strong favours of OKSSRA have a score of 1, and the answers that are strongly against OKSSRA have a score of 5. Thus, the average score of all questions, $\bar{x}$, is equal to 3, which is also the score of the neutral answers, i.e., *Neither agree nor disagree*. For example, if a subject has (i) *Strongly agree* as her or his answers to Question 1 and 3 in Part 1, (ii) *Strongly disagree* as the answers to Question 2 and 4 in this part, and (iii) *Agree* and *Disagree*, respectively, as the answers to Question 5 and 6 in Part 1, the MAD value of this subject's opinions is 1.67, i.e., the score of her or his answers to Question 1 to 6 in Part 1 will be 1, 1, 1, 1, 2, and 2, respectively. The greater mean value the MAD of a subject's answers has, the stronger opinion the subject has.

From Table 6.1 we can observe that 20 subjects (66.7%) have strong opinions when completing this questionnaire, i.e., have a mean MAD value greater than or equal to 1.37. Only two subjects (6.7%), whose answers have a mean MAD value equal to 0.67, do not show much preference between OKSSRA and the online MITPH. Based on the statistics presented in Figure 6.2 and 6.3, we have known that there are no subjects holding a negative opinion against OKSSRA. Therefore, the 66.7% of subjects having strong opinions are in strong favours of OKSSRA.

Table 6. 1 The Mean Absolute Deviation of subjects' opinions on OKSSRA

Subject number	MAD values			Subject number	MAD values		
	Part 1	Part 2	Mean		Part 1	Part 2	Mean
1	1.83	1.8	1.82	16	1.33	1.6	1.47
2	1.83	1.8	1.82	17	1.33	1.6	1.47
3	1.83	1.6	1.72	18	1.67	1.2	1.43
4	1.83	1.6	1.72	19	1.33	1.4	1.37
5	1.83	1.6	1.72	20	1.33	1.4	1.37
6	1.83	1.6	1.72	21	1.67	1	1.33
7	1.67	1.6	1.63	22	1.5	1	1.25
8	1.67	1.6	1.63	23	1.5	1	1.25
9	1.67	1.6	1.63	24	1.17	1.2	1.18
10	1.33	1.6	1.47	25	1.17	1	1.08
11	1.33	1.6	1.47	26	1.33	0.6	0.97
12	1.33	1.6	1.47	27	1.17	0.6	0.88
13	1.33	1.6	1.47	28	1.17	0.6	0.88
14	1.33	1.6	1.47	29	0.67	0.6	0.63
15	1.33	1.6	1.47	30	0.67	0.6	0.63

According to the answers of the sixth question of Part 2, subjects' suggestions on the potential improvements for OKSSRA are mostly focused on the possible improvements on the user interface designing. For example, six users suggested providing a trace back function to help users check previous unsaved search results.

6.3 Summary

This chapter reported the evaluation of OKSSRA. The experimental results reported in this chapter show that it is easier for users to retrieve desired processes using OKSSRA than using the online MITPH. Most of the subjects are in strong favour of OKSSRA compared with the online MITPH for the purpose of retrieving business processes.

OKSSRA provides an easier way to retrieve the desired business processes from MITPH than that of the online MITPH. In the test use part of this experiment, we observed that both OKSSRA and the online MITPH can retrieve the desired business processes for straightforward tasks. When the test scenarios get more complicated and require more familiar with the repository, all the subjects can still retrieve the desired business processes using OKSSRA, while only few of them can accomplish the task using the online MITPH.

According to subjects' opinions, OKSSRA is more effective and more useful than the online MITPH. All the subjects think that (i) OKSSRA is useful for gaining preliminary business knowledge, (ii) the OKSSRA's function of expanding queries based on the semantic meanings is useful, and (iii) OKSSRA provides an easier way to retrieve the relevant business processes than the online MITPH does. Most of subjects prefer to use OKSSRA and believe OKSSRA is more efficient and effective compared with the online MITPH.

The subjects' feedbacks also show that OKSSRA can provide pleasant user experiences. All the subjects believe that OKSSRA is self-explanatory and enjoy using the system. More than 90% of the subjects think OKSSRA is easy to learn and to use. More than 80% of the subjects are able to efficiently accomplish their tasks when using OKSSRA.

The subjects' suggestions toward OKSSRA are focused on the possible improvements of the user interface. Based on their suggestions, a potential direction of improving OKSSRA is to enhance the designs of the user interfaces following the Nielsen's heuristics (Nielsen and Molich 1990), which is out of the scope of this research.

Chapter 7.

Conclusions and Future Work

In this chapter, we summarise our finding in this research, highlight knowledge contributions of our research, and discuss the opportunities of further research.

7.1 Summary of main findings and contributions

This thesis has proposed and presented the development of an Ontology-based Knowledge Support System for Requirements Analysis (OKSSRA).

In Software Engineering and Requirements Engineering, business knowledge plays an important role in the process of eliciting software requirements from business holders, i.e., the sponsors/customers and users. The knowledge gap between requirements analysts and business stakeholders can seriously hinder the performances of requirements analysts, and bring down the quality of the resultant software specification documentation.

However, all the existing requirements engineering tools that we reviewed have overlooked the importance of assisting requirements analysts in gaining preliminary business knowledge. Thus, we proposed and developed OKSSRA to fill this gap with the intention of helping requirements analysts gain the preliminary business knowledge related to their project. OKSSRA can enhance requirements analysts' performances on acquiring business requirements, and thus improves the quality of the resultant software specification documentation.

Through investigating the needs of requirements analysts and the common obstacles faced by knowledge support systems, we proposed a framework and identified three key modules for OKSSRA. The three key modules are: (i) a semantic similarity measure module for expanding the users' queries and maximising the use of the knowledge repository (presented in Chapter 3), (ii) an ontology mapping module for the future extension of the business knowledge repository (presented in Chapter 4), and (iii) a module for generating use case automatically on the basis of the retrieved MITPH business processes (presented in Chapter 5).

7.1.1 Semantic similarity measure combining knowledge from the Internet and WordNet

The semantic similarity measure module enables OKSSRA to (i) extend users' queries based on the semantic meaning of the key words, and (ii) effectively retrieve the relevant business processes from the business repositories. The aims of this module are as follows: (i) providing users a more effective way to gain the desired knowledge, and (ii) allowing the system making the most use of the business knowledge repository.

Semantic networks, e.g., WordNet, are considered as better choices for estimating semantic similarity than other lexical resources (Budanitsky and Hirst 2006). The three categories of WordNet-based semantic similarity measures are as follows: (i) node-based methods, (ii) edge-based methods, and (iii) hybrid methods, all of which have their own drawbacks.

The performances of node-based methods can be seriously hindered by the two following weaknesses of the employed corpora: (i) the unbalanced content of the corpora, and (ii) the data sparseness of the corpora.

The accuracy of edge-based methods is commonly affected by the lack of consideration on the variety of the semantic distances between adjacent words in WordNet.

Hybrid methods usually have better performances than the edge-based and the node-based methods. However, their performances can still be seriously affected by the two aforementioned weaknesses of the node-based methods brought by the employed corpora.

Thus, we proposed to a novel WordNet-based hybrid method for measuring semantic similarities. Our method combines the structure information of WordNet and the knowledge retrieved from the Internet to estimate the semantic similarities between words. The structure information of WordNet is first used to calculate the shortest path between the words of interest. The Internet is then employed as a corpus for estimating the semantic distances between adjacent words on the shortest path.

Using the Internet as a corpus allows our method overcoming the common weaknesses of existing WordNet-based hybrid methods that is caused by the employed corpora. Using the Internet as a corpus has three advantages compared with using the traditional corpora: (i) the Internet is currently the largest electronic source of text; (ii) the contents of the Internet are well balanced; and (iii) most of knowledge on the Internet is up-to-date. In order to be able to use the Internet as a corpus, we use Normalised Google Distance to acquire the statistics of the adjacent words on the shortest path in WordNet for calculating their semantic distances. In order to enhance the accuracy of the calculations, we use the hypernyms from WordNet as the context words to tag word senses.

Two methods were used to evaluate our semantic similarity measure: (i) using human benchmark data sets, and (ii) applying to NLP tasks. According to the

experimental results, our newly proposed semantic similarity measure can outperform previous measures.

7.1.2 Ontology mapping based on advanced semantic similarity measure and Internet knowledge

The ontology mapping module enables OKSSRA to (i) extend its ontology-based repositories, and (ii) integrate its repositories with other ontology-based repositories.

The performances of previous ontology mapping methods have been seriously hindered by the employed linguistic based matching. Some of the previous methods, e.g., Chimaera and iPROMPT, give suggestions on ontology mapping only based on simple lexical matching. Other methods, e.g., LOM, MARFA, and MIMapper, employed light weight semantic similarity measures to match ontology entities. According to the previous evaluation by Kaza and Chen (2008), the ontology mapping methods using semantic similarity measures outperform those methods using lexical matching. However, we believe that a more sophisticated semantic similarity measure can benefit ontology mapping methods on giving more accurate mapping suggestions.

Thus, we proposed a new two-step ontology mapping method to provide more accurate suggestions on ontology mapping. In the first step, a more accurate semantic similarity measure (Liu et. al. 2011), i.e., our method proposed in Chapter 3, is employed to conduct the linguistic based matching between class names. An instance based matching is conducted as the second step. In the second step, Normalised Google Distance (NGD) is employed to utilise the Most Informative Instances (MII) for providing suggestions on mapping class entities.

According to the experimental results, our newly proposed method outperforms MIMapper. MIMapper is one of the existing ontology methods

that provide the most accurate mapping suggestions (Kaza and Chen 2008). Therefore, we believe our newly proposed ontology mapping methods can outperform most existing ontology mapping methods.

7.1.3 Automatically generating Use Cases using retrieved business processes

The automatic Use Cases generating module enables OKSSRA to derive essential Use Cases automatically on the basis of the MITPH business processes. The derived Use Cases (i) can present the business knowledge in a form that requirements analysts are more familiar with, and (ii) can be used as the cornerstone for composing conventional Use Cases in the later phases of requirements elicitation.

Essentially, the main difference between Use Cases and business processes is the way of organising the business activities. Business activities, i.e., actions, are lined up sequentially in a use case, while a business process organises the business activities in a structured and measured way. Therefore, generating Use Cases based on business processes is a matter of reorganising the business knowledge from a structured style into a sequential style.

We proposed a set of explicit rules for automatically deriving essential Use Cases based on the basis of MITPH business processes are also presented. We first identified that the MITPH business processes are in a proper size for transforming into Use Cases. A thorough analysis is then provided for the purpose of explaining why it is rational to use the structure information of MITPH, i.e., the relationship among the entities, to map the business process components into corresponding use case elements.

According to the evaluation conducted by ten intermediate to senior requirements analysts or software developers, the automatically derived

essential Use Cases are believed to be helpful for gaining preliminary business knowledge, and are in a reasonably good quality.

7.1.4 An efficient and effective knowledge support system for requirements analysts

We have proposed, designed, developed and evaluated an ontology-based knowledge support system for requirements analysis (OKSSRA).

An evaluation is designed to verify the efficiency and effectiveness of OKSSRA. The evaluation consists of the following two parts: (i) a test use to compare OKSSRA with the online MITPH, and (ii) collecting feedbacks using a questionnaire.

Three test use scenarios are designed to help the subjects realise the advantages of using OKSSRA compared with using the online MITPH. According to subjects' feedbacks, the design of the scenarios is successful. After the test use, subjects were aware of that OKSSRA's abilities of extending users' queries can improve the efficiency and effectiveness of retrieving the desired business processes from MITPH.

According to the feedbacks collected by the questionnaire, the subjects believe that (i) OKSSRA is useful for gaining preliminary business knowledge; (ii) OKSSRA is more efficient and effective than the existing MITPH based knowledge support system, i.e., the online MITPH; and (iii) OKSSRA provides pleasant user experiences.

7.2 Opportunities for further research

This section presents the possible directions and opportunities of further research on the basis of this work.

According to the feedbacks gathered from the evaluation, OKSSRA can be further improved. Firstly, the user interface of OKSSRA can be more user-friendly. The possible improvements of the user interface are as follows: (i) adding more useful operation hints, (ii) providing a help document at uses' conveniences, and (iii) providing a more heuristic user interface layouts, e.g., a multi-tab layout. Secondly, extra functionalities can be added to improve the user experiences. The candidate functionalities are as follows: (i) a trace back function, which allows users to revisit the unsaved search results, and (ii) a status bar showing the searching progress, which gives users better ideas how much more time is needed by the system to complete the process of retrieving relevant business processes from MITPH.

Another challenging research opportunity is to automatically derive conventional Use Cases. In our research, we have proposed a method and a set of rules for automatically deriving essential Use Cases from business processes. To the best of our knowledge, this method is the first one that derives Use Cases automatically from business processes. The derived Use Cases are useful to help requirements analysts to gain preliminary business knowledge, and can be used as a basis for deriving conventional Use Cases. However, the resultant Use Cases of our method is essential Use Cases. It will be a research with a bright future if a framework can be developed to automatically derive conventional Use Cases on the basis of exiting business processes.

Bibliography

Achananuparp, P., Hu, X. and Xiajiong, X. (2008). The evaluation of sentence similarity measures. In: Song, I.-Y., Eder, J. and Nguyen, T.M. (Eds.), *Data Warehousing and Knowledge Discovery*, LNCS, 5182, 305-316.

Achour C. B., Rolland C., Maiden N.A.M. and Souveyet C. (1999). Guiding use case authoring: Results of an empirical study. In *Proc. IEEE International Symposium on Requirements Engineering (RE '99)*, 36-43.

Adolph, S., Bramble, P., Cockburn, A. and Pols, A. (2003). *Patterns for Effective Use Cases*. Addison-Wesley.

Al-Karaghouli, W., AlShawi, S. and Fitzgerald, G. (2000). Negotiating and Understanding Information Systems Requirements: The Use of Set Diagrams. *Requirements Engineering*, 5(2), 93-102.

Anda B., Sjoberg D. and Jorgensen M. (2001). Quality and understandability of use case models. In *Proc. European Conference on Object-Oriented Programming (ECOOP 2001)*, 402-428.

Aoyama, M. (2005). Persona-and-scenario based requirements engineering for software embedded in digital consumer products. In *Proc. IEEE International Conference on Requirements Engineering (RE'05)*, 85-94.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. NY: ACM Press, Addison-Wesley.

Banerjee, S. and Pedersen, T. (2003). Extended gloss overlaps as a measure of semantic relatedness. In *Proc. International Joint Conference on Artificial Intelligence*, 805-810.

Bhatt, M., Rahayu, W., Soni, S.P. and Wouters, C. (2007). OntoMove: A knowledge based framework for semantic requirement profiling and resource acquisition. In *Proc. Australian Software Engineering Conference (ASWEC'07)*, 137-146.

Blaha, M., LaPlant, D. and Marvak, E. (1998). Requirements for repository software, In *Proc. Working Conference on Reverse Engineering*, 164-173.

Berry, D. and Kamsties, E. (2004). *Ambiguity in requirements specification. perspectives on software requirements*, Chapter 2. Kluwer Academic Publishers.

Bouquet, P., Stoermer, H., Niederee, C. and Mana, A. (2008). Entity name system: The backbone of an open and scalable web of data. In *Proc. IEEE International Conference on Semantic Computing (ICSC'08)*, 554-561.

Budanitsky, A. and Hirst, G. (2006). Evaluating WordNet-based measures of semantic distance. *Computational Linguistics*, 32(1), 13-47.

Caralt, J.C. and Kim, J.W. (2007). Ontology driven requirements query, In *Proc. Annual Hawaii International Conference on System Sciences (HICSS'07)*, 197-206.

Choi, N., Song, I. and Han, H. (2006). A survey on ontology mapping, *ACM SIGMOD Record*, 35(3), 34-41.

Cilibrasi, R. and Vitányi P. (2001). The Google similarity distance. *IEEE Transactions on Knowledge and Data Engineering*, 19(3), 370-383.

Cockburn, A. (2001). *Writing Effective Use Cases*. Addison Wesley.

Colbert, M. (2005). User experience of communication before and during rendezvous: Interim results. *Personal and Ubiquitous Computing*, 9(3), 134-141.

Constantine L.L. (1997). The case for essential Use Cases. *Objective Magazine*, May 1997.

Constantine, L.L. and Lockwood, L.A.D. (1999). *Software for use: A practical guide to the models and methods of usage-centered design*. Addison-Wesley.

Cox, K. and Phalp, K. (2000). Replicating the CREWS use case authoring guidelines. *Empirical Software Engineering Journal*, 5(3), 245-268.

Cox, K., Aurum, A. and Jeffrey, R. (2004). An experiment in inspecting the auality of use case descriptions. *Journal of Research and Practice in Information Technology*, 36(4), 211-229.

Davenport. T. (1993). *Process innovation*. Harvard Business School Press: Boston.

Dietrich, J., Hosking, J. and Giles, J. (2007). A formal contract language for plugin-based software engineering. In *Proc. IEEE International Conference on Engineering Complex Computer Systems (ICECCS'07)*, 175-184.

Dietz, J.L.G. (2003). Deriving Use Cases from business process models. In: Song, L. and Ling, S. (Eds.), *Conceptual Modelling - ER2003*, LNCS, 2813, 131-143.

Dijkman, R. and Joosten, S. (2002). Deriving use case diagrams from business process models. *CTIT Technical Report 02-08*, CTIT, Enschede, The Netherlands.

Dobing, B. and Parsons J. (2000). Understanding the role of Use Cases in UML: A review and research agenda. *Journal of Database Management*, 11(4), 28-36.

Euzenat, J., Ferrara, A., Hollink, L., Joslyn, C., Malaisé, V., Meilicke, C., Nikolov, A., Pane, J., Scharffe, F., Shvaiko, P., Spiliopoulos, V., Stuckenschmidt, H., Šváb-Zamazal, O., Svátek, V., dos Santos, C.T. and Vouros, G. (2009). Preliminary results of the Ontology Alignment Evaluation Initiative 2009. In *Proc. ISWC International Workshop on Ontology Matching*, 304, 96-132.

Fantechi, A., Gnesi, S., Lami, G. and Maccari, A. (2002). Application of linguistic techniques for use case analysis. In *Proc. IEEE International Conference on Requirements Engineering (RE'07)*, 157-164.

Farquhar, A., Fikes, R. and Rice, J. (1997). The Ontolingua server: A tool for collaborative ontology construction. *International Journal of Human-Computer Studies*, 46, 707-728.

Fridman Noy, N. and Musen, M.A. (2000). Prompt: Algorithm and tool for automated ontology merging and alignment. In *Proc. National Conference on Artificial Intelligence (AAAI'00)*, 450-455.

Fridman Noy, N., Fergerson, R.W. and Musen, M.A. (2000). The knowledge model of Protégé-2000: Combining interoperability and flexibility. In *Proc. International Conference on Knowledge Engineering and Knowledge Management (EKAW'00)*, LNAI, 1937, 17-32.

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

Fellbaum, C. (1998). *WordNet: An electronic lexical database*, MIT Press.

Gabrilovich, E. and Markovitch, S. (2007). Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proc. International Joint Conference on Artificial Intelligence*, 1606-1611.

Gruber, T.R. (1993). A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2), 199-220.

Hirst, G. and St-Onge, D. (1998). Lexical chains as representations of context for the detection and correction of malapropisms. In: Fellbaum C. (Eds.), *WordNet: An Electronic Lexical Database*, 305-332.

Huhns, M.N. and Stephens, L.M. (2002). Semantic bridging of independent enterprise ontologies. In *Proc. International Conference on Enterprise Integration and Modelling Technique (ICEIMT'02)*, 83-90.

Jacobson, I., Booch, G. and Rumbaugh, J. (1999). *The unified software development process*. MA: Addison-Wesley.

Jacobson, I., Christerson M., Jonsson P. and Övergaard G. (1992). *Object-oriented software engineering - A use case driven approach*. MA: Addison-Wesley.

Jacobson, I., Ericsson, M. and Jakobson, A. (1994). *The object advantage: Business process reengineering with object oriented technology*. MA: Addison-Wesley.

Jiang, J.J. and Conrath, D.W. (1998). Semantic similarity based on corpus statistics and lexical taxonomy. In *Proc. International Conference on Research in Computational Linguistics*, 19-33.

Kaindl H., Brinkkemper S., Bubenko Jr. J., Farbey B., Greenspan S. J., Heitmeyer C. L., do Prado Leite J.C.S., Mead N. R., Mylopoulos J. and Siddiqi, J. (2000). Requirements engineering and technology transfer: Obstacles, incentives, and improvement agenda. *Requirements Engineering*, 7(3), 113-123.

Kaiya H. and Saeki, M. (2006). Using domain ontology as domain knowledge for requirements elicitation. In *Proc. IEEE International Conference on Requirements Engineering (RE'06)*, 186-195.

Kalfoglou, Y. and Schorlemmer, M. (2003). Ontology mapping: The state of the art. *The Knowledge Engineering Review*, 18(1), 1-31.

Kaza, S., Wang, Y. and Chen, H. (2006). Enhancing border security: Mutual information analysis to identify suspect vehicles. Decision Support Systems, 43(1), 199-210.

Kaza, S. and Chen, H. (2008). Evaluating ontology mapping techniques: An experiment in public safety information sharing. *Decision Support Systems*, 45(4), 714-728.

Kruse, P.M., Naujoks, A., Roesner, D. and Kunze, M. (2005). Clever search: A WordNet based wrapper for Internet search engines. In *Proc. GermaNet Workshop*, arXiv:cs/0501086v1.

Lambrix, P. and Tan, H. (2007). A tool for evaluating ontology alignment strategies. *Journal of Data Semantics*, 8, 182-202.

Leacock, C. and Chodorow, M. (1998). Combining local context and WordNet similarity for word sense identification. In: Fellbaum, C. (Eds.), *WordNet: An Electronic Lexical Database*, 265-283.

Leacock, C., Miller, G. and Chodorow, M. (1998). Using corpus statistics and WordNet relations for sense identification. *Computational Linguistics*, 24, 147-165.

Lee, S.W., Gandhi, R., Muthurajan, D., Yavagal, D. and Ahn, G.J. (2006). Building problem domain ontology from security requirements in regulatory documents. In *Proc. Software Engineering for Secure Systems workshop (SESS'06) in conjunction with International Conference on Software Engineering (ICSE'06)*, 20-28.

Lenat, D. and Guha, R.V. (1990). *Building large knowledge-based systems: Representation and inference in the Cyc project*. Addison-Wesley.

Li, J. (2004). LOM: A lexicon-based ontology mapping tool, In *Proc. Performance Metrics for Intelligent Systems* (*PerMIS'04)*, 24-26.

Li, Y., McLean, D., Bandar, Z.A., O'Shea, J.D. and Crockett, K. (2006). Sentence similarity based on semantic nets and corpus statistics. *IEEE Transactions on Knowledge and Data Engineering*, 18(8), 1138-1150.

Likert, R. (1932). A technique for the measurement of attitudes. *Archives of Psychology*, 140, 1-55.

Lin, D. (1998). An information-theoretic definition of similarity. In *Proc. International Conference on Machine Learning*, 296-304.

Lindsey, R., Veksler, V.D., Grintsvayg, A. and Gray, W.D. (2007). Be wary of what your computer reads: the effects of corpus selection on measuring semantic relatedness. In *Proc. International Conference on Cognitive Modeling*, 279-284.

Liu, G., Wang, R., Buckley, J., and Zhou, H.M. (2011). A WordNet-based semantic similarity measure enhanced by Internet-based knowledge. In *Proc. The 23rd International Conference on Software Engineering and Knowledge Engineering (SEKE 2011)*, 175-178.

Maedche, A., Motik, B., no Silva, N. and Volz, R. (2002). MAFRA - A MApping FRAmework for distributed ontologies. In *Proc. European Conference on Knowledge Engineering and Knowledge Management (EKAW'02)*, LNCS, 2473, 235-250.

McConnell, S. (1996). *Rapid development: Taming wild software schedules*, 1st Ed., WA: Microsoft Press.

McGuinness, D.L., Fikes, R., Rice, J. and Wilder, S. (2000). An environment for merging and testing large ontologies. In *Proc. International Conference on Principles of Knowledge Representation and Reasoning (KR'00)*, 483-493.

Merunka, V. (2006). Knowledge modelling using CraftCASE tool. *Agricultural Economics (Zemědělská ekonomika)*, 4, 165-172.

Miller, G.A. and Charles, W.G. (1991). Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1), 1-28.

Mitra, P., Wiederhold, G. and Jannink, J. (1999). Semi-automatic integration of knowledge sources. In *Proc. International Conference on Information Fusion (FUSION'99)*, 823-830.

Mitra, P., Wiederhold, G. and Kersten., M.L. (2000). A graph-oriented model for articulation of ontology interdependencies. In *Proc. Conference on Extending Database Technology (EDBT'00)*, LNCS, 1777, 86-100.

Navigli, R. (2009). Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2), 1-69.

Nielsen, J. and Molich, R. (1990). Heuristic evaluation of user interfaces. In *Proc. ACM CHI'90 Conference on Human Factors in Computing Systems*, 249-256.

Niles, I., and Pease, A. (2001). Toward a standard upper ontology, In *Proc. the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001)*, 17-19.

Niles, I. and Terry, A. (2004). The MILO: A general-purpose, mid-level ontology. In *Proc. International Conference on Information and Knowledge Engineering (IKE'04)*, 15-19.

Nurcan, S., Grosz, G. and Souveyet, C. (1998). Describing business processes with a guided use case approach. In *Proc. Conference on Advanced Information Systems Engineering*, LNCS, 1413, 339-362.

Nuseibeh, B. and Easterbrook, S. (2000). Requirements engineering: A roadmap. In. *Proc. International Conference on Software Engineering (ICSE'00)*, 35-46.

Object Management Group (2009). OMG unified modelling language specification version 2.2. Retrieved May 1, 2009, from http://www.omg.org.

Phalp K. T., Vincent J. and Cox K. (2007). Assessing the quality of use case descriptions. *Software Quality Journal*, 15 (1), 69-97.

Pirrò, G. (2009). A semantic similarity metric combining features and intrinsic information content. *Data and Knowledge Engineering*, 1289-1308.

Porter, M.F. (1980). An algorithm for suffix stripping. *Program*, 14(3), 130-137.

Pouloudi, A. (1999). Aspects of the stakeholder concept and their implications for information systems development. In *Proc. Hawaii International Conference on Systems Sciences (HICSS-32)*, 7, 7030-7046.

Rubenstein H. and Goodenough, J.B. (1965). Contextual correlates of synonymy. *Communications of the ACM*, 8(10), 627-633.

Richardson, R. and Smeaton, A.F. (1995). Using WordNet in a knowledge-based approach to information retrieval. *Working Paper CA-0395*, School of Computer Applications, Dublin City University, Ireland.

Resnik, P. (1995). Using information content to evaluate semantic similarity. In *Proc. International Joint Conference on Artificial Intelligence*, 448-453.

Robertson, J. and Robertson, S. (2006). *Mastering the Requirements Process (2nd edition).* Addison-Wesley.

Rodgers, J.L. and Nicewander, W.A. (1988). Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1), 59-66.

Salem, A.M. and Darter, M.O. (2006). Requirements analysis: A practical object oriented approach. J*ournal of Computational Methods in Sciences and Engineering*. 6, 191-204.

Santander, V.F.A. and Castro, J.F.B. (2002). Deriving Use Cases from organizational modelling. *In Proc. IEEE Joint International Conference on Requirements Engineering (RE'02)*, 32-42.

Sawyer, P., Rayson, P. and Cosh, K. (2005). Shallow knowledge as an aid to deep understanding in early phase requirements engineering. *IEEE Transactions on Software Engineering*, 31(11), 969-981.

Sharp, H., Finkelstein, A. and Galal, G. (1999). Stakeholder identification in the requirements engineering process. In *Proc. International Workshop on Database and Expert Systems Applications*, 387-391.

Sinir, S.S, (2007). Ontology Mapping Survey. Retrieved August, 2007, www.srdc.metu.edu.tr/webpage/seminars/Ontology/Ontology%20Mapping%20Survey.ppt.

Sussna, M. (1993). Word sense disambiguation for free-text indexing using a massive semantic network. In *Proc. International Conference on Information and Knowledge Management*, 67-74.

Turney, P.D. (2006). Similarity of semantic relations. *Computational Linguistics*, 32(3), 379-416.

Unhelkar, B.U. (2005). *Practical Object Oriented Analysis*. Thomson Learning Nelson.

van Dommelen, W., Joosten, S. and de Mol., M. (1999). Comparative study to aids for managing business processes (in dutch: Vergelijkend warenonderzoek hulpmiddelen beheersing bedrijfsprocessen). *Technical Report*, Department of Finance, The Hague.

van Rijsbergen, C.J. (1979). *Information Retrieval*. MA: Butterworth.

Walker, H. (1931). *Studies in the History of the Statistical Method*. MD: Williams & Wilkins.

Wasson, K.S. (2006). A case study in systematic improvement of language for requirements. In *Proc. IEEE International Conference on Requirements Engineering (RE'06)*, 6-15.

Wiegers, K.E. (2003). *Software Requirements 2: Practical techniques for gathering and managing requirements throughout the product development cycle*. Redmond: Microsoft Press.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B. and Wesslén, A. (2000). *Experimentation in software engineering: An introduction*. MA: Kluwer Academic Publishers.

Yang, D. and Powers, D.M.W. (2005). Measuring semantic similarity in the taxonomy of WordNet. In *Proc. Australasian Computer Science Conference*, 315-322.

Yu, E. (1995). *Modelling Strategic Relationships for Process Reengineering*. Phd Thesis, University of Toronto.

Appendix A. Scenarios for Test Use of OKSSRA

The three given scenarios are created for the test use of Ontology-based Knowledge Support System for Requirements Analysis (***OKSSRA***).

In these given scenarios, we assume that you are a junior requirements analyst working for a software company. Your company develops bespoke software tools/systems for clients. Your key task at work is to elicit and collect business requirements from clients for the software systems to be developed. In order to make your meetings with clients more effective, you normally prepare yourself with preliminary business knowledge related to the software systems to be developed, so that you may always ask the right questions for fetching requirements from the clients.

Please try to use online MIT Process Handbook (***MITPH***) and OKSSRA, ***respectively***, to retrieve relevant business processes for the following scenarios.

For each scenario, please ***firstly*** conduct **no *more than THREE*** searches on the online MITPH, and then conduct ***no more than THREE*** searches on OKSSRA using the ***same queries*** as those you just used on the online MITPH.

Scenario 1

There is a client of your software company, who is planning to customise a software system for hiring new employees. The software to be developed is supposed to help the Human resource department to recruit new employees at every step of the hiring process. ***Please identify how many steps that the process of hiring new employee normally includes.***

Your queries are:

---------- ---------- ----------

Did you find relevant processes for this scenario?
□ Yes, only when I was using online MITPH.
□ Yes, only when I was using OKSSRA.
□ Yes, when I was using both online MITPH and OKSSRA.
□ No, I did not.

Which software system is more helpful for this scenario?
□ Online MITPH □ OKSSRA

Scenario 2

There is an advertising agency, which is looking into extending their business by selling products through the Internet. This agency comes to your company and wants to purchase a software system customised for helping them promote products in the cyber space. In order to design this software system correctly, ***please investigate how many possible ways that this agency could sell their advertisements through the Internet.***

Your queries are:

---------- ---------- ----------

Did you find relevant processes for this scenario?
□ Yes, only when I was using online MITPH.
□ Yes, only when I was using OKSSRA.
□ Yes, when I was using both online MITPH and OKSSRA.
□ No, I did not.

Which software system is more helpful for this scenario?
□ Online MITPH □ OKSSRA

Scenario 3

There is a client of your software company, who wants to purchase a bespoke software system for managing business relationships with different parties. You are assigned to meet with the client for eliciting their requirements on this software system. As a part of preparation for the meeting, ***please investigate how a company normally manages their business relationships and what those business relationships are.***

Your queries are:

---------- ---------- ----------

Did you find relevant processes for this scenario?
□ Yes, only when I was using online MITPH.
□ Yes, only when I was using OKSSRA.
□ Yes, when I was using both online MITPH and OKSSRA.
□ No, I did not.

Which software system is more helpful for this scenario?
□ Online MITPH □ OKSSRA

Appendix B. Questionnaire for OKSSRA Evaluation

These questionnaires are anonymous. Should you require more information, please do not hesitate to contact me. We thank you for your time and support.

Part I: Effectiveness and Usefulness

1. The business processes returned by OKSSRA are helpful for gaining preliminary business knowledge.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

2. The key words provided by OKSSRA are useless for retrieving the business processes of interest.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

3. The bundle of relevant business processes returned by OKSSRA always includes something useful.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

4. The online MITPH is easier to use compared with OKSSRA
.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

5. It is easier to retrieve relevant business processes by using OKSSRA compared with the online MITPH.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

6. I prefer to use the online MITPH for retrieving relevant business processes instead of OKSSRA.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

Part II: User Experience

1. Learning how to use OKSSRA is difficult.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

2. I can understand and act on the system information provided by OKSSRA.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

3. Getting data files in and out of OKSSRA is difficult.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

4. Tasks can be performed in a straightforward manner using OKSSRA.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

5. I do *not* enjoy my sessions with OKSSRA.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

6. Do you have any suggestion on OKSSRA for future improvements?

_____

_____

_____

Appendix C. Questionnaire for Investigating the Quality of Use Cases Generated by AUCG Module

These questionnaires are anonymous. Should you require more information, please do not hesitate to contact me. We thank you for your time and support.

For the following two questions, please compare the generated Use Cases and their corresponding business processes.

1. The generated Use Cases are easier to be understood than the business processes.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

2. It is easier to compose conventional Use Cases on the basis of the Use Cases than the business processes.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

For the following five questions, please think about the quality of all of three generated Use Cases.

3. The Use Cases are easy to read and understand.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

4. The Use Cases have *not* included sufficient information to describe the situations:

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

5. The Use Cases are at a proper level of details.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

6. The events in the alternative flow have been separated from the main flow and correctly numbered:

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

7. Alternative or exceptional events are neither viable nor reasonable.

□ Strongly agree □ Agree □ Neither agree nor disagree □ Disagree □ Strongly disagree

Appendix D. Review of Current Requirements Engineering Tools

In this Section, we investigate eight existing requirements engineering tools. These tools are developed by various vendors or educational organisations. We try to review every well-known requirements engineering tool that we can access to. However, some of well-known tools, such as Accept 360°, C.A.R.E and Blueprint, have not been included in this section, since these tools do not have free trial version available.

I. Caliber DefineIT

Caliber DefineIT is a software that helps analysts communicate with business stakeholders and obtain definitions of the system to be developed. It provides a visual storyboard to improve efficiency of the interaction between the analysts and customers. With the visual storyboard, analysts can draw customers a flow chart to show their understanding of the system process and, at the same time, get feedback from the customers. During the validation phases, snap pictures of a prototype can also be attached to show customers the appearance of the system at each scenario. After analysts and customers reach an agreement on the system process, DefineIT will record the flow chart and elicit initial requirements from the chart.

The advantage for analysts to use DefineIT is that the software gives analysts a “digital white board” to communicate with customers and, also, seamlessly elicit requirements from the discussion results. However, the conveniences provided by DefineIT are built upon the assumption that the analysts have enough knowledge of the system to draw an initial flow chart describing the processes of the system. One of our goals is to help analysts to obtain sufficient of business knowledge related to the system to be developed, so that they can discuss with customers with better efficiency and effectiveness. Also, the discussion can be more thoughtful and related to the project.

II.CaliberRM

CaliberRM is designed as a requirements management system for enterprises. It has a central repository of requirements to enable sharing updated requirements to stakeholders. Requirements can be stored in the repository with text descriptions and graphic presentations, i.e., images or tables. With internet connection, CaliberRM can provide a long distance communication environment for developing team members to discuss requirements. Any behaviour of changing existing requirements will be recorded and alert all relevant parties who have accesses to the repository. In addition to the revision history, various information of each requirements can also be found in the repository, such as discussions regarding to the requirements, and related documentation. The system can also trace all requirements to see how a change of single requirements will influence other existing requirements.

In additional to providing various operations on requirements, several modules for assisting project management are also embedded into CaliberRM: (i) Datamart: it offers a dashboard-style interface to help management groups to assess the efficiency and effectiveness of the software developing process and, also allow them to adjust the developing strategies based on current progress; (ii) ESTIMATE Professional: it helps managers to estimate the scope and the cost of the project.

What CaliberRM focuses on is the operations on existing requirements, e.g., tracing requirements, deducing requirements from existing ones. We found no function provided by CaliberRM which is helpful to improve the efficiency and effectiveness of the communications between business stakeholders and analysts.

III. IBM Rational RequisitePro

RequisitePro is a requirements management software developed by IBM. It has the some similar functions as CaliberRM: (i) storing requirements into a repository, (ii) tracing the changes of requirements, (iii) recording the revision history of requirements, (iv) enabling setting different attributes to each requirement, (v) keeping the related discussions of every requirement, (vi) analysing impacts of requirements, etc.

RequisitePro and CaliberRM do have some differences from each other, though they have quite similar goals. RequisitePro has a tighter integration with MSWord than CaliberRM. RequisitePro provides a tool bar, which can be integrated into MSWord workplace. With the help of the toolbar, users are able to simply export requirements from documents into RequisitePro repository at MSWord workplace. It is only a two-step process: selecting a sentence from specification documentation, and click the “import” button of the toolbar. However, it cannot import requirements from tables in MSWord documents into the database.

The other differences are: (i) RequisitePro can import CSV documents into the database, which cannot be deal with CaliberRM; (ii) RequisitePro give a list view to users to show the components of database, which is not as friendly as the tree structure used by CaliberRM; (iii) RequisitePro can easily link requirements with their related documents stored in the database, which cannot be done by CaliberRM; (iv) because of the sophisticated integration with MSWord, RequisitePro can synchronise the requirements in database with their related documents, which is not a function of CaliberRM.

One minor imperfection of RequisitePro is that, when users are trying to export requirements right from MSWord workplace: all sentences that have been exported to the database as requirements are normally be highlighted by using dark red as font colour. If you type in some new words right after those

highlighted sentences, those new words will turn dark red as well. However, the new words have not been put into the database yet, which may mislead users.

As we mentioned before, the goal of RequisitePro, as a requirements management tool, is similar to Borland CaliberRM. The repository maintained by RequisitePro is to provide a better robust environment for stakeholders to communicate and collaborate more efficiently. Providing assistance to analysts to elicit requirements efficiently is not the goal of building the repository.

IV. DOORS

DOORS (Dynamic Object Oriented Requirements System) is a requirements management tool promoted by Telelogic. It allows users: (i) defining attributes for different types of requirements, (ii) checking the linkages between requirements, (iii) viewing the revision history of requirements, iv) importing requirements from documents into database, v) querying database and viewing discussions related to requirements, etc.

DOORS provides a C-like description language for extension developing, which is not very common in requirements management tools. The language is called DXL (DOORS eXtension Language). It provides interfaces to various tools, such as, Microsoft Word, Microsoft Excel and Rational Rose. Utilising DXL, extensions developed by users can access to most DOORS functions.

Although DOORS can import requirements from documents into database, it cannot synchronise the documents with related requirements in database.

Similar to other requirements management tools, DOORS aims at providing a good managing support to requirements operations. Using DOORS can

improve communication and collaborative between analysts and customers only in terms of organizing requirements documentation.

V. GatherSpace

GatherSpace is a pure web-based requirements management tool, which, as claimed by the vendor, is developed to suit Agile software developing process.

All operations with GatherSpace are conducted at a website (www.GatherSpace.com) through an internet browser. If a user logs into the website as an administrant of a project, s/he can grant other users various privileges of operating project related items. GatherSpace helps users organise requirements through packages, each one of which represents a logical grouping of related requirements. At the website, users can also define requirements, Use Cases, etc.

Various reports can be automatically generated by the website based on the project information stored in the website, such as (i) a vision and requirements report which gives an overview of the project, (ii) a Use Cases report which presents all Use Cases which have be defined, (iii) a feature report which shows the requirements hierarchy structure of the project associating with related requirements and Use Cases, (iv) the traceability reports which indicates the developing progress of each requirement, etc. One interesting report can also be created automatically by GatherSpace. It is Agile parking lot diagram report, which shows the developing status of each package of requirements.

However, GatherSpace does no support users uploading software requirements specifications or other requirements documentation. Generally speaking, GatherSpace is an online simplified requirements management tool, which supports basic requirements operations and managements. Similar to

other requirements management tools, the ability to give knowledge support to analysts is not considered by GatherSpace.

VI. Accompa

Similar to GatherSpace, Accompa is a web-based requirements management tool. All operations with Accompa have to be conducted by using an internet browser.

Accompa supports users creating new requirements by typing in through its workplace. Users can also import requirements from existing CSV documents, which cannot be done with GatherSpace. Additionally, users can attach files to each requirement as well. After having a set of requirements, Accompa allow users generating views of requirements that meet designated features. It also provides environment for team stakeholders to discuss on each requirement. All discussions can be recorded, archived and linked to related requirements. Accompa enable users to create requirements documentation in various file formats with selected requirements.

However, compared to GatherSpace, Accompa cannot generate reports based on stored information and can only process textual information.

Similar to GatherSpace, Accompa is only a light weight requirements management tool and does not offer knowledge support in any way.

VII. Statestep

Statestep is a tool similar to decision table tool. It supports deductions of rules presented by formal notations. Users can define their own constraints, and rules in Statestep. Utilising Statestep, users can also model system as a finite state machine to discover inconsistencies of requirements, which are written in formal notations.

Our project is not related to formal methods and does not range over requirements written in formal notations.

VIII. Tiger Pro

Tiger Pro (Tool to InGest and Elucidate Requirements PROfessional) is developed by University of South Australia. It can collect requirements from two ways: (i) importing from text files, (ii) inputting from keyboard. To input from keyboard, users can simply input requirements at a workplace, or utilise BADGER (Built-in Agent using Deterministic Grammar for the Engineering of Requirements). After filling in the some key elements of requirements, such as objective, action, and action target, BADGER can create a requirement with a pre-determined structure to avoid ambiguities which may occur in natural language.

Tiger Pro is able to perform a light weight natural language processing to give a plain evaluation. Five types of defects can be pointed out based on whether candidate requirements have employed “poor words”, which are listed in a “poor word file”. The five types of defects are: (i) multiple requirements in a paragraph, (ii) possible multiple requirements, (iii) unverifiable requirements, (iv) use of “will” or “must” instead of “shall”, and (v) the existence of “poor words”.

Tiger Pro allows users setting attributes for each requirement. After summarising attributes of requirements, the software can provide graphical reports to indicate project’s statistics, such as cost, risk, priority distribution, etc.

Similar to other requirements management tool, Tiger Pro does not provide knowledge supports to analysts.

IX. WIBNI

WIBNI (Wouldn't It Be Nice If ...?) is a light weight requirements management tool. It is built upon Microsoft Access. All wizards of WIBNI have to be run in the environment of Microsoft Access.

WINBI allow users importing requirements from Microsoft Excel files (*.xcl) and Lotus files (*.wk*). Users can define attributes for each requirement, such as priority, type, status, and dependences. Business evens and Use Cases can be attached to each requirement. WIBNI is able to capture and archive revision histories.

WINBI provides three sorts of reports based on stored requirements: (i) listing reports, (ii) validation and verification reports, and (iii) summary reports. List reports show users what contents are stored in database, such as use case report, requirements report, and business event report. WIBNI can perform a low level checking and examining on stored requirements, the examining results are given as validation and verification reports. Those reports list sets of requirements which have undefined attributes, or have not been approved, etc.

WIBNI is simply a requirements organising tool. It does not support many high level operations on requirements, such as proving traceability matrix, and impact analysis. Knowledge support has not been considered as a function of WIBNI.

In order to show the difference between them, we will compare features of above nine tools in the following table.

From the above review of nine requirements tools, most of these tools have a database as requirements repository. However, all these repositories are maintained for the purpose of restoring requirements and their related data,

such as, attributes, change history, related Use Cases, etc. Some companies have realised the importance of smoothing the communication between analysts and business stakeholders, and have already provided tools to help eliciting requirements, such Boland DefineIT. However, none of these tools has used a repository to give knowledge support to analysts for gaining business knowledge.

Appendix E. Previous Research on Ontology Mapping

Chimaera

McGuinness et al. (2000) proposed Chimaera as an ontology merging environment for the ontology editor Ontolingua (Farquhar et al. 1997). Chimaera suggests mapping ontology entities based on their names. If the names of two entities from different ontologies have simple lexical relations, e.g., (i) the names are identical, and (ii) one name is an acronym or expanded form of the others. These two entities are then believed to be a matching pair by Chinaera.

In addition to the simple lexical relations, Chimaera also takes a taxonomic relation (i.e., the subclass–superclass relation among class names) into account for providing mapping suggestions. For example, if there are two classes named “car” and “car door” in two source ontologies, respectively, Chimaera will suggest users that these two classes may be related.

Since Chimaera gives the mapping suggestions based on the light weight lexical matching between entity names, it may neglect a notable number of match entities, such as those entities whose names are synonyms or hyponyms to each other.

Fridman Noy and Musen (2000) also suggest that Chimaera only shows which pair of entities are potentially match, but failed to provide suggestions to users on how to act on the matched entity pair. For example, Chimaera does not tell users whether they should merge the two matched classes as identical classes, or put one class as a subclass of the another one.

PROMPT

PROMPT is proposed by Fridman Noy and Musen (2000) as an extension of Protégé 2000 (Fridman Noy et al. 2000). It offers a semi-automatic mapping algorithm for the ontology aligning and merging.

Fridman Noy and Musen (2003) extend the original PROMPT into a PROMPT suite, which is composed of four components: (i) iPROMPT, which is the original PROMPT; (ii) AnchorPROMPT, which gives suggestions on mapping based on a set of anchor entities (i.e., related terms in source ontologies given by users) and ontologies structures; (iii) PROMPTDiff, which provides suggestions on organising different versions of one source ontology; and (iv) PROMPTFactor, which provides suggestions on retrieving sub-ontologies from a source ontology, e.g., the size of the sub-ontologies. We review the following two components that are related to ontology mapping: (i) iPROMPT, and (ii) AnchorPROMPT.

iPROMPT

The merging procedure of iPROMPT is an interactive procedure (as shown in Fig. E.1). During the merging procedure, iPROMPT will first give an initial list of potentially matching entity pairs based on the similarities between their names. Each of these elements pairs is combined with a suggested operation for the merging procedure. At the second step, users select an operation from the suggested operations by iPROMPT. iPROMPT will then give a new list of potential matching pairs along with operation suggestions for the next loop after executing the current operation chosen by users.

iPROMPT gives the initial mapping suggestions based on a linguistic matching on class names. Since Fridman Noy et al. (2000) believed that a concrete mapping can only be achieved through interacting with users. Users are encouraged to choose their own matching module, which is supported by Protégé's component-based architecture.

Figure E. 1 The flow of PROMPT merging algorithm (Fridman Noy and Musen 2000)

Kaza and Chen (2008) believe that iPROMPT will provide a set of better suggestions on potential matching entities, if it employed more sophisticated matching methods, such as the semantic relatedness among the class names. Further, Kaza and Chen (2008) argued that merely class name matching cannot give sufficient mapping suggestions. Nevertheless, the idea of performing merging using an interactive procedure may be one of the potential ways to further improve our mapping algorithm.

AnchorPROMPT

AnchorPROMPT can be seen as a tool for augmenting the mapping ability of iPROMPT based on the structure information of ontologies. In order to initiate AnchorPROMPT, users need to provide two pairs of anchor entities in across ontologies of interest. Each entity of the anchor pair in one ontology is related to one of the anchor entities in another ontology, respectively. AnchorPROMPT will then use the equal length paths between the anchor pairs related across the two ontologies to locate more matching entities. For example, as shown in Fig. E.2, A and B, and G and H are the two pairs of given anchors, C and E, and D and F are the entities on the path between A and G, and B and H, respectively. AnchorPROMPT considers C and D, and E and F as two pairs of potential matches because of that they are at the corresponding positions of the path between A and G, and the path between B and H.

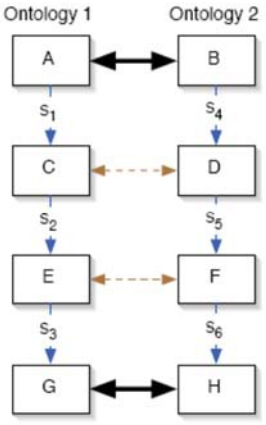

Figure E.2 the way of AnchorPROMPT predicting mapping (Fridman Noy and Musen 2003)

We argue that the positions on the paths between the anchor pairs are not sufficient to provide evidence to determine potential matching entities. We believe that the outcomes of AnchorPROMPT can be enhanced if semantic similarities are also considered for estimating the similarity between entities across ontologies.

ONION

ONION (ONtology compositION system) is a framework for integrating ontologies proposed by Mitra et al. (2000). It does not store source ontologies as parts of the system, nor actually integrates ontologies. ONION keeps a set of articulation ontologies that are composed of a set of articulation rules. The articulation rules formally describe the mapping relations among entities from different ontologies. The mapping relations are obtained through inferring ontology entities and their formalised relations.

ONION uses a tool called SKAT (Semantic Knowledge Articulation Tool, Mitra et al. 1999) to calculate the articulation rules in a semi-automatic way. The inference procedure is triggered by a set of initial mapping relations among entities across ontologies, such as, the *SubClassOf* and *AttributeOf* relations. The initial relations are provided by ontology engineers. The

outcomes of SKAT will be stored in the articulation ontologies after being validated by ontology engineers.

The ontology mapping results of ONION are claimed to be quite precise. However, we can see that the mapping process largely relies on the interference of human being. It is safe to say that the accuracy of the mapping results of ONION is a trade off with the efficiency. This trade off brings ONION some inherent problems, such as the difficulty of specifying formal relations within a large size ontologies. The articulation rules used by ONION are quite similar to the semantic bridges used by Maedche et al. (2002) in MAFRA.

MAFRA

MAFRA (MApping FRAmework for distributed ontologies) is a framework proposed by Maedche et al. (2002) for merging ontologies. It utilises a semantic bridge ontology to assist the semantic alignment during the merging process.

MAFRA utilises Resnik (1995)'s semantic similarity measure to estimate the lexical similarities among entities' names. However, we believe that a more sophisticated semantic similarity measure than Resnik's measure can be used to enhance MAFRA's ability of estimating the lexical similarities. As reviewed in Section 3.1.2, the Resnik's measure has some inherent issues, i.e., the ignorance of WordNet structure information can cause the coarse result to its estimations on semantic similarities among concepts.

Therefore, we believe that MARFA could improve the accuracy of matching entity names by employing a better semantic similarity. The candidate measures can be Jiang and Conrath (1998)'s measure or our semantic similarity measure (proposed in Section 3.2).

LOM

LOM (Lexicon-based Ontology Mapping) is an ontology mapping tool proposed by Li (2004). It offers mapping suggestions based on the lexical similarities of class names. LOM calculates the lexical similarities using the following four steps: (i) whole term matching, (ii) word constituent matching, (iii) synset matching, and (iv) type matching.

The whole term matching performs a string match over class names to identify exact matches among the class names. In this step, each class name is considered as a single term to match against each other.

In the word constituent matching step, LOM conducts a string match over words of the class names. In this step, stop-words in the class names, such as, “the”, “a”, and “and”, will be dropped first. The remaining words in the class names will then be matched against each other at the string level to compute the lexical similarities.

The synset matching step employs WordNet to identify whether non-stop-words in the class names are synonyms. For a pair of multi-term class names, the more synonyms they share, the more similar they are.

The type matching step utilises SUMO (the Suggested Upper Merged Ontology, Niles and Pease 2001) and MILO (the Mid-level Ontology, Niles and Terry 2004) to detect whether the words in class names describe the same type of classes or properties. For example, words “stapler” and “pencil” will be considered as related term in this step, as they are both stationeries.

Kaza and Chen (2008) believed that LOM does a good job on matching class names at the lexical level. However, Kaza and Chen (2008) also suggested that the mapping suggestions provided by LOM may not be sufficient if they are only based on the class name matching. The structure information of

source ontologies, such as information from instances of classes, can be used to improve LOM's mapping suggestions.

MIMapper

MIMapper is proposed by Kaza and Chen (2008) using WordNet and the information theory to provide suggestions on ontology mapping. According to their evaluation, MIMapper has a better performance than other existing methods, such as aforementioned PROMPT, Chimaera, and LOM.

MIMapper conducts the ontology mapping through a two-step process: (i) a linguistic based matching on class names, and (ii) an instance based mapping. In the first step, a light weight semantic similarity measure utilising WordNet is employed to match the class names. In the second step, the Most Informative Instances (MII) of each class is used to help match classes. Point-wise Mutual Information (PMI) is used to locate the most informative instance of each class.

Appendix F. Data Obtained from Evaluations

The similarity ratings on RG-Set computed by Jiang and Conrath's method and our method are presented in the following table.

Word Pairs		Human Ratings	Jiang and Conrath's Method	Our Method
chord	smile	0.02	2.414625	1.835365
rooster	voyage	0.04	0	0
noon	string	0.04	0	0
fruit	furnace	0.05	2.574328	2.076186
autograph	shore	0.06	0	0
automobile	wizard	0.11	0.713508	1.098267
mound	stove	0.14	2.413301	2.211298
grin	implement	0.18	0	0
asylum	fruit	0.19	2.413611	2.152028
asylum	monk	0.39	0.019635	1.483011
graveyard	madhouse	0.42	0.292105	0.130999
glass	magician	0.44	1.15024	1.684264
boy	rooster	0.44	0.771198	1.379568
cushion	jewel	0.45	2.578424	2.377549
monk	slave	0.57	1.639478	2.430525
asylum	cemetery	0.79	0.68933	1.14918
coast	forest	0.85	2.628563	1.843388
grin	lad	0.88	0	0
shore	woodland	0.9	2.630823	2.023787
monk	oracle	0.91	1.031132	1.918955
boy	sage	0.96	1.788535	1.880743
automobile	cushion	0.97	2.669067	2.184129
mound	shore	0.97	2.828229	2.869941
lad	wizard	0.99	1.996438	1.871363
forest	graveyard	1	1.247843	1.330568
food	rooster	1.09	1.236685	1.424441
cemetery	woodland	1.18	1.232562	1.322924
shore	voyage	1.22	0	0
bird	woodland	1.24	2.452944	1.383504
coast	hill	1.26	3.044067	2.948658

furnace	implement	1.37	2.964249	2.464604
crane	rooster	1.41	3.032218	2.566619
hill	woodland	1.48	2.603338	2.015757
car	journey	1.55	0	0
cemetery	mound	1.69	0.831427	1.300032
glass	jewel	1.78	1.731487	1.845733
magician	oracle	1.82	1.197395	1.907735
crane	implement	2.37	3.113713	2.862369
brother	lad	2.41	2.070014	2.164296
sage	wizard	2.46	2.034043	1.811726
oracle	sage	2.61	2.865016	2.631831
bird	crane	2.63	3.419187	3.156978
bird	cock	2.63	3.419187	3.363976
food	fruit	2.69	3.042652	2.102851
brother	monk	2.74	1.663791	2.207742
asylum	madhouse	3.04	3.289469	3.260546
furnace	stove	3.11	2.43192	1.952124
magician	wizard	3.21	3.425318	3.4588
hill	mound	3.29	3.425318	3.477683
cord	string	3.41	3.425318	3.391375
glass	tumbler	3.45	3.177189	3.31324
grin	smile	3.46	3.406484	3.469338
serf	slave	3.46	3.399896	2.919858
journey	voyage	3.58	3.169801	3.228969
autograph	signature	3.59	3.115567	3.13256
coast	shore	3.6	3.405023	3.264181
forest	woodland	3.65	3.401753	3.466286
implement	tool	3.66	3.231748	3.180132
cock	rooster	3.68	3.425318	3.470059
boy	lad	3.82	3.061208	3.311518
cushion	pillow	3.84	3.329702	3.341182
cemetery	graveyard	3.88	3.417033	3.472967
automobile	car	3.92	3.408999	3.473
midday	noon	3.94	3.40406	3.466496
gem	jewel	3.94	3.40154	3.485753

The similarity ratings on MC-Set computed by Jiang and Conrath's method and our method are presented in the following table.

Word Pairs	Human Ratings	Jiang and Conrath's Method	Our Method
car - automobile	3.92	3.409	3.473
gem - jewel	3.84	3.403	3.496
journey - voyage	3.84	3.062	3.303
boy - lad	3.76	2.899	3.256
coast - shore	3.7	3.402	3.172
asylum - madhouse	3.61	3.234	3.18
magician - wizard	3.5	3.433	3.445
midday - noon	3.42	3.402	3.472
furnace - stove	3.11	2.055	1.337
food - fruit	3.08	2.851	1.542
bird - cock	3.05	3.365	3.187
bird - crane	2.97	3.365	3.028
tool - implement	2.95	3.153	3.05
brother - monk	2.82	0.943	2.17
crane - implement	1.68	2.921	2.643
lad - brother	1.66	1.409	1.927
journey - car	1.16	0	0
monk - oracle	1.1	0.207	1.459
food - rooster	0.89	0.185	0.6
coast - hill	0.87	2.937	2.686
forest - graveyard	0.84	0.195	0.197
monk - slave	0.55	0.968	2.089
coast - forest	0.42	2.311	1.166
lad - wizard	0.42	1.363	1.466
chord - smile	0.13	2.193	1.117
glass - magician	0.11	0.032	1.019
noon - string	0.08	0	0
rooster - voyage	0.08	0	0
cemetery - woodland	0.95	0.178	0.185
shore - woodland	0.63	2.262	1.472