Universiteit Utrecht

MASTER'S THESIS

Natural Language Processing in Requirements Elicitation and Analysis: A Systematic Literature Review

Author:
Duc Janssens

Supervisor:
Dr. Fabiano Dalpiaz

Second supervisor:
Dr. Marcela Ruiz Carmona

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science

in

Business Informatics
Department of Information and Computing Sciences

UTRECHT UNIVERSITY

Abstract

Faculty of Science
Department of Information and Computing Sciences

Master of Science

Natural Language Processing in Requirements Elicitation and Analysis: A Systematic Literature Review

by Duc Janssens

Natural language processing (NLP), the area of study concerned with how computers can interpret and handle natural language, has gained interest in the last few years. Based on a preliminary study, we identified that within the field of requirements engineering (RE), NLP is widely applied and has the potential for much benefit in the industry. However, no clear overview is available on what tools in the field use NLP. Therefore, this research aims to establish the current state of research on how NLP is currently being used in RE, specifically within the phases of requirements elicitation (collecting requirements from stakeholders and other sources) and requirements analysis (further processing these requirements for understanding them better).

The main research method used in the thesis is the Systematic Literature Review (SLR). An SLR involves collecting a large amount of literature according to a pre-established protocol, selecting relevant papers from this collection, assessing their quality, and finally extracting data from this final list of literature. From this data, we established purposes of the tools that are available in the field, as well as categories of tasks that employ NLP techniques. We also performed statistical analysis using Jaccard Indices to see if any patterns of co-occurrence between RE (sub-)phases, tool purposes and NLP tasks could be discovered.

We conclude by explaining that though NLP is in the lift within the field, it is far from mature. NLP tasks are predominantly used to analyze the requirements; however, requirements documentation has the potential for more use of NLP techniques. Moreover, we ask researchers in the field to establish a common terminology, and to explicitly state used NLP techniques in their papers.

Contents

List of Figures

List of Tables

Chapter 1

Introduction and background

1.1 Introduction

Natural language is defined by the Cambridge Dictionary as "language that has developed in the usual way as a method of communicating between people, rather than language that has been created, for example for computers." [1] It is the language that we learn in order to interpret the spoken and written word, and to speak and write ourselves. For us, this usually comes natural as we learn to speak and write in our youth. However, for a computer, that is a different issue:

When a person sees or hears a sentence, he makes full use of his knowledge and intelligence to understand it. This includes not only grammar, but also his knowledge about words, the context of the sentence, and most important, his understanding of the subject matter. To model this language understanding process in a computer, we need a program which combines grammar, semantics, and reasoning in an intimate way, concentrating on their interaction.
- Terry Winograd, 1972

Researchers argue that computers are far away still from being able to fully understand natural language (Cambria and White, 2014), but there are some tasks involving natural language that computers can already successfully carry out, like part-of-speech tagging and rudimentary machine translation (Petrov, Das, and McDonald, 2011; Nadeau and Sekine, 2007). The generation and interpretation of natural language by computers is also known as Natural Language Processing, or NLP in short (Chowdhury, 2003).

The beginning of NLP can be traced back to the end of the 1940's (Hutchins, 2005). Ever since, it is a subject with interest from multiple fields, including medicine (Spyns, 1996; Nadkarni, Ohno-Machado, and Chapman, 2011; Friedman and Hripcsak, 1999), biology (Ananiadou and Mcnaught, 2006; Cohen and Hunter, 2004; Yandell and Majoros, 2002), and jurisprudence (Aletras et al., 2016; Lame, 2005; Talib et al., 2017). Over the past few years, the interest in NLP is picking up after a few years of stability (see Figure 1.1).
In this thesis, we study Natural Language Processing of a software engineering discipline in which natural language is predominant: Requirements Engineering (Pohl, 2010). Requirements Engineering, or RE in short, is the discipline that encompasses the techniques and tools used to gather, organize and use requirements in software systems (Nuseibeh and Easterbrook, 2000). There are multiple models that describe the RE process, all with their own terms for the different phases (Aurum and Wohlin,

[1] https://dictionary.cambridge.org/dictionary/english/natural-language

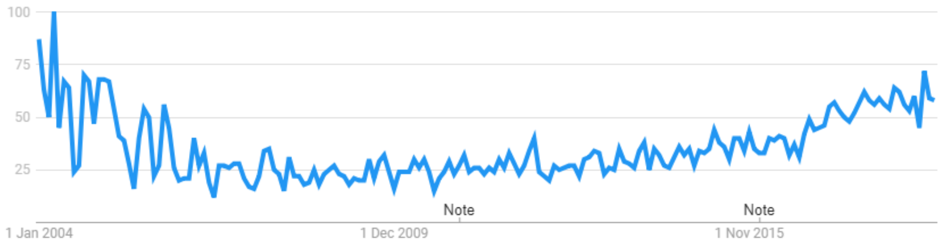

FIGURE 1.1: Interest in Natural Language Processing since 2004. Via Google Trends (December 20th, 2018).

2005); for this research, the model by Kotonya and Sommerville (1998) will be used. The scope of this research will be limited to requirements elicitation (Zowghi and Coulin, 2005) and requirements analysis (Maciaszek, 2007), which are both phases in which requirements are prepared for use in software systems. While elicitation is concerned with extracting requirements from a multitude of sources, including stakeholders, analysis encompasses further processing of these extracted requirements.

The current manuscript presents a research project to investigate the current role of NLP in requirements engineering. We focus on the requirements elicitation and analysis phases, and establish the following main research question:

What is the current state of research on Natural Language Processing within Requirements Elicitation and Requirements Analysis?

This thesis takes a systematic approach to answering the questions posed above. By conducting a Systematic Literature Review (Kitchenham, 2004), we aim to give an overview of the domain and patterns in this domain.

The next two sections of this chapter provide a background on the concepts NLP and RE, as well as a justification for this research. Furthermore, this thesis contains 8 more chapters. Chapter 2 provides the research method and used research subquestions. Chapter 3 contains a definition of NLP, as well as a classification and description of tasks contained within NLP. Chapters 4 and 5 define requirements elicitation and requirements analysis, respectively, and connect common activities to these two domains. A description of how the research was performed is featured in Chapter 6. Chapter 7 features a list of tools and tool-supported approaches that employ NLP within the domain. A categorization of NLP tasks and tool purposes can be found in Chapter 8. Chapter 9 explores patterns of co-occurrence of the NLP tasks. Finally, Chapter 10 features a discussion of the research challenges and threats to validity encountered, and conclusions are drawn in Chapter 11.

1.2 Background

1.2.1 NLP

One of the first allusions to NLP is the Turing Test (Turing, 1950), an experiment wherein a human has to interact with another, unknown party. Unbeknownst beforehand to the human, however, this unknown party is actually a computer system. If the human does not understand by the end of the interaction that the other party is a computer, the computer passes the test. However, for such a level of interaction, the computer has to correctly interpret spoken or written word by the human party, and generate correct responses. These interpretation and generation tasks are both tasks that fall under NLP.
The first successful example of machine translation, and thus of NLP, is also known as the Georgetown-IBM experiment (Dostert, 1955). In this experiment, more than 60 Russian sentences (transposed from the Cyrilic script into the Roman script) were input into a system co-developed by IBM and the University of Georgetown. Following 6 translation rules, these sentences were then translated to English sentences, with very few errors. According to Hutchins (2004), this would spur the interest and funding of machine translation by governments.
One result of the boost in interest in NLP was the founding of the Automatic Language Processing Advisory Committee, or ALPAC, in 1964 (Hutchins, 2003). In 1966, the ALPAC released a report on the field of machine translation, which was at that time the predominant field in NLP research (Pierce et al., 1966). In this report, they stated that at that time, human translation was of higher quality and speed and at a lower cost than machine translation. Therefore, there was no direct need for more research in this field. The direct influence of this is that funding, as well as interest in the scientific field of machine translation, was drastically decreased in the years after 1966. Of course, there was still some research going on, but not at the same level as pre-1966.
At the same time, new linguistic models were introduced, which lead to developments within other fields of NLP. One example of this is syntactic labeling (Chomsky, 1967), which bears strong similarities to part-of-speech tagging, discussed in Chapter 3. There was also some success with the use of Controlled Natural Languages: languages with restricted vocabulary and grammar. A renowned application of Controlled Natural Language is SHIRDLU (Winograd, 1971), an AI system that moved different shaped objects in a virtual world based on instructions provided by humans.
However, it wasn't until 1980 that NLP research would pick up. A known division in NLP is the division between symbolic and statistic (Manning and Schütze, 1999). Symbolic approaches are approaches were words are tokenized, e.g. labeled by their function in the sentence. As such, a well-known implementation of this is Part-of-Speech tagging. Statistic approaches, on the other hand, rely on probability theories applied to the co-occurrence of certain text fragments. An example is is the use of Bayesian Classifiers for word sense disambiguation.

Nowadays, NLP is a well-researched subject, with numerous applications. One of the manifest examples is Siri, an automated assistant incorporated in Apple's iPhone, that can perform mobile tasks (such as calling contacts or sending text messages) and give responses based on the user's text or speech input. This involves a plethora of NLP techniques, among which are speech recognition, word sense disambiguation, natural language generation and text-to-speech. A trend within NLP

is deep learning (Socher, Bengio, and Manning, 2012): the use of neural networks in NLP, which have as an advantage that little human intervention is needed in the training of models that NLP techniques use.
Tasks involving NLP are not as simple as would seem on first hand, as proven by the ALPAC report. This is explained by Cambria and White (2014):

"Today, most of the existing approaches are still based on the syntactic representation of text, a method that relies mainly on word co-occurrence frequencies. Such algorithms are limited by the fact that they can process only the information that they can 'see'. As human text processors, we do not have such limitations as every word we see activates a cascade of semantically related concepts, relevant episodes, and sensory experiences, all of which enable the completion of complex NLP tasks such as word sense disambiguation, textual entailment, and semantic role labeling — in a quick and effortless way."

What this quote illustrates is the importance of context in natural language. As persons speak to each other, they have prior knowledge of the subject of conversation, the meaning of the words in the conversation and their conversation partner; moreover, contextual information during the conversation can be provided, like the tone on which a statement is given or special emphasis on some words. As an exemplary sentence, take "My mom will kill me if she found out". In a literal interpretation, someone's mother will kill that person it the mother finds out something. It is at that point unclear what would drive the mother to kill her son/daughter. However, more often than not, "kill" is in this context an exaggeration for "give a stern lecture", probably for something that the son/daughter did. So long as a computer is unable to interpret sentences in a different context depending on the subject, it is clear that NLP has a long way to go.
Cambria and White (2014) also discuss how they envision NLP to become more

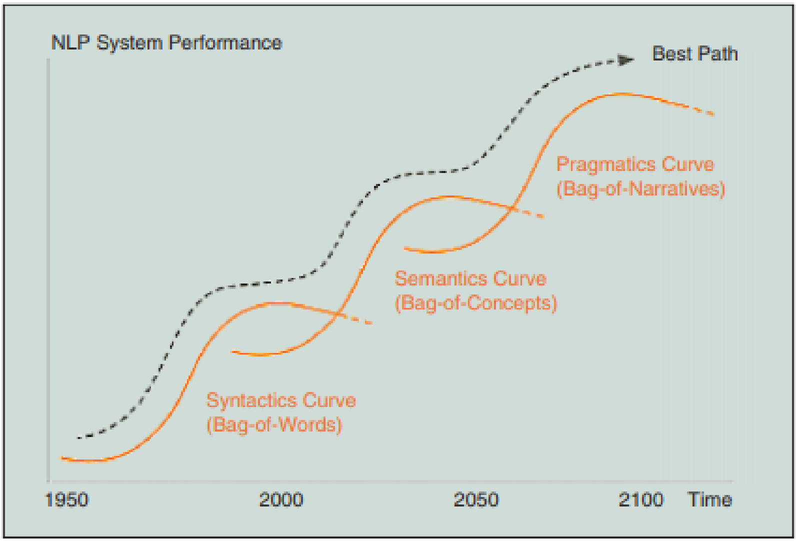

FIGURE 1.2: Model of NLP curves, as proposed by Cambria and White (2014).

advanced over the years. This is shown in Figure 1.2. Currently, NLP has just progressed from the Syntactics Curve to the Semantics Curve. What is meant by that is that while NLP used to rely on word-based techniques (Part-of-Speech tagging and stemming), now more semantic information is used (lemmatization, Semantic Role

Labeling). In 50 years, NLP research has progressed to the level of the Pragmatics Curve, *"which will enable NLP to be more adaptive and, hence, open-domain, context-aware, and intent-driven. Intent, in particular, will be key for tasks such as sentiment analysis - a concept that generally has a negative connotation, e.g., small seat, might turn out to be positive, e.g., if the intent is for an infant to be safely seated in it."* At the end of the Pragmatics Curve, when computers can fully understand and reply to human speech, NLP will have made the transition to Natural Language Understanding.

1.2.2 Requirements Engineering

A clear definition of RE is proposed by Zave (1997):

"Requirements engineering is the branch of software engineering concerned with the real-world goals for functions of and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families."
- Zave (1997)

The essence of this definition is the mapping of functions and constraints to the system itself. Commonly, requirements are split into two categories: functional requirements and non-functional requirements, also known as quality requirements (Chung and Prado Leite, 2009). Functional requirements regard the software system's functionality; an example would be for an e-mail system to save draft e-mails, to be completed and sent at a later point in time. A quality requirement is used to note qualitative properties of a system, like flexibility, speed, throughput, user experience or even use of color. An example in the e-mail system would be for an e-mail sent by the system to arrive at the destination a maximum of 2 seconds after clicking the s̀end' button.

A figure of the phases of Requirements Engineering, made by Kotonya and Sommerville (1998), is shown in Figure 1.3. As one can see, there are the following phases:

1. **Requirements elicitation:** the process of extracting requirements for the system from stakeholders.
2. **Requirements analysis:** further processing the requirements extracted in the elicitation phase. Modelling, e.g. creating models of the requirements, is usually a sub-task of this phase.
3. **Requirements negotiation:** Deciding which requirements will be implemented in the system, and which will not.
4. **Requirements documentation:** Giving an overview of which requirements are to be implemented in the software system, and which have already been implemented.
5. **Requirements validation:** Ensuring that the documented requirements are complete, consistent, and satisfy the stakeholders' needs (Bilal et al., 2016).

As noted, this thesis will solely cover the phases of requirements elicitation and requirements analysis. However, some of the phases above occur as sub-phases of elicitation or analysis.

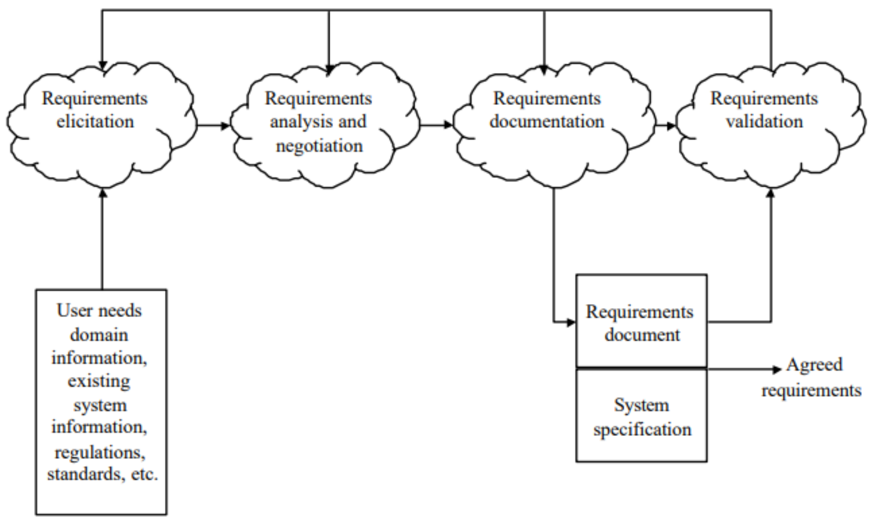

FIGURE 1.3: The Linear Requirements Engineering Process Model (Kotonya and Sommerville, 1998).

1.3 Justification of need and scientific challenge

A wide array of NLP tools and techniques for RE is already available and have scientific articles written on them. Examples are:

- Li, Dewar, and Pooley (2005): Using Part-of-Speech tagging to link natural language to object-oriented concepts;
- Casagrande et al. (2014a): Using a parse tree to extract goals from scientific publications;
- Mich and Garigliano (2002): Using semantic analysis for requirements document analysis;
- Ibrahim and Ahmad (2010a): Using rule-based analysis to extract class diagrams from requirements.

These tools can prove very beneficial for any business that wants to support their elicitation and analysis processes with some automation. However, though there are conferences and journals for NLP, like NLPIC [2], IJNLC [3] and EMNLP [4], knowledge of NLP techniques applied to practice is spread throughout many fields, including medicine, computer science, and (computational) linguistics. Therefore, getting a holistic overview of the entire NLP field for one specific domain, especially when taking developments of other domains into account, is a challenge. By systematically gathering relevant papers and classifying them, an overview of these tools and techniques can be given in a comprehensible and reproducible way, allowing businesses to decide for implementing techniques that are most useful to them.

After having conducted a preliminary study, searches were done for other systematic literature reviews of NLP within requirements engineering. One was found:

[2] https://10times.com/nlpic
[3] http://airccse.org/journal/ijnlc/
[4] http://emnlp2018.org/

Nazir et al. (2017). This systematic literature review also gathers several papers about NLP in RE and categorizes them according to a classification that they established themselves. However, this paper has several issues:

- It does not go in-depth into neither requirements analysis nor requirements elicitation;
- The paper does not provide research perspectives within the field, which could be incorporated into future research within the field of RE;
- The protocol is not given, making it difficult to reproduce the results that they gathered.

Therefore, we conclude that the study by Nazir et al. (2017) does not sufficiently represent a comprehensive literature review of the field.
Another article relevant to this SLR is Meth, Brhel, and Maedche (2013), an SLR that covers automated requirements elicitation. However, not all requirements elicitation tools within the domain of this thesis are intended for automation of requirements elicitation. Moreover, though most tools in this article involve the analysis of requirements documents, not all of them involve NLP. Lastly, the intent of this thesis is to also incorporate requirements analysis.
Though there are articles available that review NLP research and provide future research directions (Cambria and White, 2014; Demner-Fushman, Chapman, and McDonald, 2009; Jones, 1994), these articles research the theory of NLP in general. Their results can spur NLP technique development in general. However, by researching the use of NLP within the current field of RE, research directions can be provided that would not be uncovered should one look at RE from an NLP perspective.

Chapter 2

Research method

2.1 Research questions

In Chapter 1, the main research question for this thesis has been presented:

What is the current state of research on Natural Language Processing within Requirements Elicitation and Requirements Analysis?

This research question cannot be answered directly, as it comprises several factors that have to be taken account. That is why the main research question is supported by several research sub-questions, which will cover those factors. The research sub-questions are as follows:

RQ1: *What is Natural Language Processing, and which tasks does NLP encompass?*

The purpose of RQ1 is to determine what exactly is understood as NLP for this thesis by using a definition from established literature. However, solely a definition does not suffice; by linking tasks to NLP, it becomes clear for the reader what it is exactly that NLP comprises.

RQ2: *What is the definition of requirements elicitation, and what activities does it encompass?*

RQ3: *What is the definition of requirements analysis, and what activities does it encompass?*

Research questions 2 and 3 serve to establish the exact scope of requirements elicitation and requirements analysis, respectively. They start by defining their respective concepts and placing them in the broader spectrum of requirements engineering in general. Next, they link activities to the concepts, where it is made sure that there is a clear distinction between the two.

RQ4: *What tools and tool-supported approaches within requirements elicitation and requirements analysis employ NLP?*

This question serves to give the reader a clear picture of which tools and tool-supported approaches that employ NLP are available to use for elicitation and analysis. Note: this only discusses tools and tool-supported that have been published in a peer-reviewed scientific article that is gathered from a magazine, journal, workshop or conference; commercial tools not covered by any scientific medium will not

TABLE 2.1: Research tasks that make up the research method of this thesis.

Task	Research question	Method	Input	Output
Preliminary study	All	Literature study	Proceedings of RE conferences in the years 2010-2017	Preliminary list of literature
NLP research	RQ1	Literature study	Unstructured literature about NLP	Definition + Tasks of NLP
Elicitation research	RQ2	Literature study	Unstructured literature about requirements elicitation	Definition and activities of requirements elicitation
Analysis research	RQ3	Literature study	Unstructured literature about requirements analysis	Definition and activities of requirements analysis
Listing of tools and techniques	RQ4	Systematic literature review	Preliminary study, libraries, queries	List of tools within requirements elicitation and analysis that employ NLP
Classification	RQ5	Iterative coding in NVivo	List of NLP tools and techniques	Classification of RE phases, tool purposes and NLP tasks
Data synthesis	RQ6	Statistical analysis	List and classification of RE phases, tool purposes and NLP tasks	Patterns of co-occurrence within the domain

be featured.

RQ5: *How can Natural Language Processing tasks within requirements elicitation and requirements analysis be categorized?*

The purpose of this question is to introduce a categorization of the employed NLP tasks in the list of results. This categorization is done on three levels: the RE phase that the article has a tool in, the main purpose of the tool presented in the article, and the NLP tasks that the tool employs. How the categorization is constructed is discussed in the next section.

RQ6: *What patterns of co-occurrence for NLP within requirements elicitation and requirements analysis can be uncovered?*

The final research question serves to uncover patterns in the dataset of the categorized articles. This is done by looking at co-occurring instances of NLP tasks that occur in the articles, the RE phase that the article is positioned in, and the main purpose of the tool presented in the article.

2.2 Research method

To answer the research sub-questions, research must be conducted. There are several research tasks to be conducted, of which the most research tasks are literature research. For an overview of which research task answers which research question, see Table 2.1. All tasks will be discussed in the subsections below.

2.2.1 Preliminary study

The first research task to be conducted was a preliminary study to the literature surrounding NLP in requirements elicitation and requirements analysis. This preliminary study was originally intended as a structured literature search for Automated Reasoning techniques and tools within the domain of requirements engineering. However, based on the result of this preliminary study, the scope was adjusted to NLP within requirements analysis and elicitation.
The aim for this study was to get a relevant first list of literature, from which a start could be made on the list of techniques and tools that would eventually be gathered.

The sources for this literature were the Proceedings of the IEEE International Requirements Engineering Conference [1]. Of these conferences, the proceedings of the 18th up to and including the 25th edition were used (2010-2017). These conferences discuss tools, techniques, frameworks and methods for requirements engineering, and as such, these conferences were a good starting point literature-wise.
Each of these articles was listed into a Google Drive sheet with the year of the conference, the title of the article, the author(s), index terms and abstract. If index terms were not available in the article itself, index terms would be used from the abstract page of the article on the website of the conference; these index terms would also be marked in bold, to draw attention that they were not from the article itself.
Consequently, each article was tagged to four categories: AR / NLP / AI / etc., Tool / Technique / Framework / Approach etc.; Relevant to elicitation; and Relevant to analysis. The tagging was based off the title, abstract and keywords; as the tagging was done manually, reading each article in detail would be too much work for the 500+ articles in the proceedings. We decided to exclude those articles that could not be categorized using title, abstract, or index terms; the article had to mainly discuss the category, and such information would always be featured in these elements.
In the first category, the article was tagged if it involved Automated Reasoning, NLP, Artificial Intelligence or similar techniques, like mining and machine learning. The second category of tagging was meant for if the article proposed or presented a technique, tool, approach, method, or something similar like a framework or an extension of any of these; this was almost always clearly stated in the abstract of the article, and as such could be tagged with ease. The third and fourth categories marked if the article was significantly relevant to requirements elicitation and/or requirements analysis, respectively. For good measure, an extra field was also included which marked the main relevant phases of Requirements Engineering.
As a final step for the eventual initial scope, selection was carried out. Every paper had to at least include NLP, and be relevant to requirements elicitation and/or analysis. If either one missed, the article was ruled out. The category for tools etc. was more for overview than for actual selection.

The selection process resulted in 38 papers: these can be seen in Appendix A. Moreover, a full list of all articles researched is featured in Appendix B.

2.2.2 Literature studies of NLP, elicitation and analysis

The next task for the research method was the research into the concepts of NLP, requirements elicitation and requirements, congruent with RQ1, RQ2 and RQ3. The main issue here is that no predefined source of literature was available like in the preliminary study. As such, the search for literature regarding these concepts was less structured.
As such, an alternative approach was used. For all three concepts, a primary paper was looked up. Important to this paper was that it covered the concept in particular, which means that no other concepts in the paper were discussed in as much detail as the concept of interest. This was possible for two of the three concepts:

- **Natural language processing** (Chowdhury, 2003) for NLP;
- **Requirements elicitation: A survey of techniques, approaches, and tools** (Zowghi and Coulin, 2005) for requirements elicitation.

[1]https://www.computer.org/csdl/proceedings/re/index.html

For requirements analysis, a definition was made by taking four phases of requirements engineering and combining them: requirements modelling, requirements classification, requirements verification, and requirements prioritization. Why these phases were chosen was justified using relevant literature. The sources that cited any of the above stated literature could be used for further literature search, as well as the sources of the literature. Moreover, Google Scholar was used to look up any extra needed literature.

2.2.3 Systematic Literature Review

The answers to RQ1, RQ2 and RQ3 were found in an unstructured way. However, this was not an option for RQ4, as RQ4 involves getting a comprehensive list of NLP techniques and tools. This means that the search for literature needed to be conducted in a structured way, lest articles were accidentally left out giving a distorted image of the current field. Therefore, the research method for RQ4 is a systematic literature review (Kitchenham, 2004).
A Systematic Literature Review, or SLR in short, is an approach for finding all relevant literature to a particular area of study. Using it, one can (1) summarize all relevant data, (2) identify research gaps and (3) provide a framework wherein new studies can be placed. Of these three results, result 1 corresponds with RQ4 and (partially) RQ5, while result 2 was used for RQ6. Moreover, the output of RQ4 serves as an input for RQ5, and together they serve as an input for RQ6.
An SLR has to have a protocol in order to ensure that it is carried out correctly. This protocol is further discussed in Chapter 6. In this same chapter, the execution of the SLR is also further elaborated upon.

2.2.4 Iterative coding

Having acquired a full list of relevant literature through the SLR, the next step was to create a categorization. This categorization was done on three levels: the requirements engineering phase(s) that the tool, technique or tool-supported approach was relevant to; the main purpose(s) of the tool, technique or tool-supported approach; and lastly, the NLP tasks that were employed within the tool. This categorization is the answer to RQ5.
This categorization was established with an approach that we will refer to from now on as 'iterative coding.' The tool used for this step is NVivo, which is qualitative data analysis software. In this software, it is possible to import .pdf files, select text within that file and drag it to a 'code', e.g. a category that the selected text fits in. Thus, text was selected and dragged to multiple sub-categories within the categories RE Phase, Tool Purpose and NLP Task.
However, as the categorization was not always correct, and some categories were later aggregated and further improved, there were multiple rounds of coding, with in between a round of validation. This validation was done by the student in cooperation with the first supervisor, dr. Fabiano Dalpiaz, because of his more extensive domain knowledge. Therefore, we coin this research method as iterative coding, with 'iterative' referring to the cycle of coding and validation. Further details are given in Chapter 6.

2.2.5 Data synthesis

This research task is part of the SLR as well. However, a distinction must be made between this research task and the iterative coding. Where iterative coding does not involve statistical analysis of any kind, the data synthesis does, as it is aimed at finding patterns of co-occurrence, as stated in RQ6. Further detail on the used methods will be given in Chapter 6.

Chapter 3

Natural Language Processing

RQ1: *What is Natural Language Processing, and which tasks encompass this concept?*

3.1 A definition of NLP

Natural Language Processing is defined as follows by Chowdhury (2003):

"Natural Language Processing (NLP) is an area of research and application that explores how computers can be used to understand and manipulate natural language text or speech to do useful things. NLP researchers aim to gather knowledge on how human beings understand and use language so that appropriate tools and techniques can be developed to make computer systems understand and manipulate natural languages to perform the desired tasks."
- Chowdhury (2003)

Before being able to write any NLP program, there needs to be understanding as to how we as humans write, talk, listen and read in order to program this on a computer. The language that we use for these tasks is named natural language, hence Natural Language Processing. NLP lies at the intersection of computer science and linguistics, the study of natural language.
A related concept to Natural Language Processing is Natural Language Understanding (NLU): getting a computer to understand a text, e.g. the meaning and sentiment of the text. Liddy (2001) notes the difference between NLP and NLU: though their tasks overlap significantly and the goal of NLP is eventually NLU, the techniques have not progressed so far yet that computers can make inferences about the texts, which is key to NLU. This is also illustrated by the quote from Winograd (1972) in Chapter 1.

Three related concepts to NLP are computational linguistics, information retrieval and text mining. Computational linguistics lies at the intersection of computer science and linguistics, the study of language; in essence, it is the study of language using techniques coming from computer science (Hausser, 1999). NLP is a subset of computational linguistics, as computational linguistics isn't solely focused on processing natural language.
Information retrieval is the extraction of information related to the question that the extractor wants to address, from a source of data (Baeza-Yates and Ribeiro, 2011). This source can take different forms: visual, audio, but also textual data. Through NLP, the extractor has the means to extract the information that s/he wants from textual data. Text mining is very similar to textual information retrieval: text mining is the activity of extracting information from text corpora (Berry and Castellanos, 2004). Again, this can be achieved using NLP techniques.

These related concepts have many intersections with the field of NLP. As such, during the conduction of the SLR, the decision was made to include literature from these fields as well. This decision is reflected in the search strategy, seen in Chapter 6.

3.2 NLP tasks

A multitude of tasks fall under the domain of NLP. Though no complete overview of these tasks currently exists in a single article, using a multitude of sources can give an overview of the most notable tasks. In the subsections below, the most important NLP tasks will be listed. The tasks have been categorized according to three categories: Syntactic NLP, Semantic NLP and Pragmatic NLP. This categorization is based on the NLP curves that Cambria and White (2014) describe. Further explanation per category is also included.

3.2.1 Syntactic NLP

Syntactic NLP concerns the tasks that are syntax-based: as such, these tasks do not concern themselves with the context that the sentence is in, but rather looks at the individual words and their functions, and performs manipulations on these. Tasks of this category are currently the most established of the three categories, as these tasks are also the simplest.

Part-of-Speech tagging

Part-of-Speech tagging (PoS tagging) is the activity wherein each word in a sentence is labeled ("tagged") with its syntactic role (Collobert et al., 2011). Syntactic roles include nouns, verbs, adjectives etc.; see the footnote [1] for an exemplary list for the Penn Treebank project, a PoS project (Marcus, Marcinkiewicz, and Santorini, 1993). A small list of these tags has also been featured in Table 3.1. PoS tagging is a precursor activity to other NLP activities (Brill, 2000); it does not truly extract meaning from the words themselves. That does not, however, indicate that it is not an essential task; due to it being a preprocessing task, other NLP tasks depend on a correct execution of PoS tagging (Martinez, 2012).

Morphological segmentation

Morphological segmentation is the activity of segmenting words into morphemes (Demberg, 2007). Morphemes are atomic parts of words with their own meaning, that cannot be split up further. Example: "breaking" can be split up into "break" and "ing". In this case, the word "break" is a free morpheme, as can be used as a word on its own. "ing", however, is a bound morpheme, and needs to be combined with other morphemes to gain meaning. The study of morphemes is called morphology, hence the name morphological segmentation.

Word segmentation

Word segmentation is similar to morphological segmentation; however, the purpose of word segmentation is to split compound words: words made up of multiple words (Sproat et al., 1996). While this task is little used in the English language, it is much more relevant in languages such as Chinese and Dutch (Quené, 1992), as

[1] https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

TABLE 3.1: An exemplary list of possible tags in Part-of-Speech tagging, used by the Penn Treebank project (Marcus, Marcinkiewicz, and Santorini, 1993).

Tag	Description
IN	Preposition or subordinating conjunction
DT	Determiner
JJ	Adjective
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
PRP	Personal pronoun
PRP$	Possessive pronoun
RB	Adverb
VB	Verb, base form
VBD	Verb, past tense

words in these languages regularly consist of multiple words. An example: "garbage truck" is segmented by a whitespace, and thus word segmentation is not needed. However, in Dutch, this word translates to "vuilniswagen", which can be separated into "vuilnis" (garbage) and "wagen" (truck). Thus, NLP applications for the Dutch language would do good to implement word segmentation.

3.2.2 Semantic NLP

Whereas syntactic NLP looks at the functions of words and their place in the sentence, semantic NLP is about the meaning of the words and the sentence in general. As such, the tasks listed are more specialized than syntactic tasks. Syntactic tasks are often used as input for these tasks.

Named Entity Recognition

Named Entity Recognition (NER) is tasked with extracting named entities from sentences (Mohit, 2014). A named entity is a specific instance of a person, location, place in time, object, company etc. For example, in the sentence "Jan bought a Toyota Corolla in Amsterdam back in 2006", named entities are "Jan" (a person), "Toyota Corolla" (an object), "Amsterdam" (a location) and "2006" (a place in time). NER is tasked with finding and categorizing these named entities, as these usually hold the most important information in the sentence.

Semantic Role Labeling

Semantic Role Labeling (SRL) is a way of representing the different roles in a sentence, so that these roles are clear to a computer even when the sentence is represented in a different way (Jurafsky and Martin, 2014). An example would be the following two sentences: "Jan slaps Piet" and "Piet is slapped by Jan". Humans recognize that these sentences have the same meaning; however, computers interpret the second sentence differently than the first. SRL is used to remedy this phenomenon by assigning roles to the subjects, objects and verbs, so that these roles are interpreted consistently even in other forms of writing the sentence.

Word sense disambiguation

Ambiguity is a natural phenomenon for words in sentences. Words in a sentence can have multiple meanings, depending on the context. An examples is "leaves": is it the plural of "leaf", or the third person singular form of "to leave"? Though most humans can derive the meaning of the word from the context, this is another matter for computers. Word sense disambiguation is the task that concerns itself with exactly this issue and tries to solve it for automated systems (Mihalcea, 2011).

Sentiment analysis

From a text containing an opinion about a subject, one can often interpret that opinion as being positive or negative. The automatized approach of this is known as sentiment analysis (Nasukawa and Yi, 2003). A computer will read a text and analyze all words in it. If it detects that there are more words with a negative connotation than with a positive one, the text will be marked `negative'. A common application area for sentiment analysis is Twitter; Pak and Paroubek (2010), for example, used the platform for building a sentiment classifier, that can automatically detect if a sentence in a document is positive, negative or neutral.

Machine translation

Though the oldest implemented form of NLP (see also the section 'Background' in Chapter 1), fully automated correct machine translation is, as of 2018, still a well-researched subject. Machine translation is the automated translation of one language to another. This isn't as simple as translating the separate words from the one to the other language and putting them in the same order, as different languages have different grammar structures, and computers cannot perform common sense reasoning (Arnold, 2003). Current research within machine translation often revolves around neural machine translation (Bahdanau, Cho, and Bengio, 2014): the use of neural networks (Schalkoff, 1997) to translate text corpora, rather than do it per sentence.

3.2.3 Pragmatic NLP

The final NLP category discussed by Cambria and White (2014) is pragmatic NLP. The issue with pragmatic NLP is that for the previous two categories, the computer did not necessarily need to understand what the text was about. However, for pragmatic NLP tasks, this is imperative. This is best explained by Cambria and White (2014):

"Semantics, however, is just one layer up in the scale that separates NLP from natural language understanding. In order to achieve the ability to accurately and sensibly process information, computational models will also need to be able to project semantics and sentics in time, compare them in a parallel and dynamic way, according to different contexts and with respect to different actors and their intentions (Howard and Cambria, 2013). This will mean jumping from the Semantics Curve to the Pragmatics Curve, which will enable NLP to be more adaptive and, hence, open-domain, context-aware, and intent-driven."

Question answering

Question answering is the interpretation of a natural language question and answering it correctly (Hirschman and Gaizauskas, 2001). Now, for simple questions, this would probably fall under semantic NLP, like when asking: "what is the product of 2 and 4?" However, other, more difficult questions demand a correct interpretation, and establishing as well as correctly formulating the answer will be much more difficult, which would be the case with a question like "What was Napoleon's rationale behind invading Russia?" Not to mention that such questions can only be answered with a knowledge base like the internet or an encyclopedia. As such, difficult questions would fall under pragmatic NLP.

Automatic summarization

Automatic summarization involves automatically extracting the key information from a text (Hahn and Mani, 2000). A distinction can be made between extraction and abstraction methods: extraction methods weight sentences based on relevance and put them together as the summary, while abstraction methods try to extract the meaning of the text, and enrich it with other material. Abstraction methods make more use of other NLP techniques than extraction methods to achieve their goal of a richer summarization.

3.3 NLP categorizations

In the previous subsection, a categorization was used based on syntactic, semantic and pragmatic NLP, as proposed by Cambria and White (2014). However, other categorizations can also be identified for NLP tools and techniques.
A variation of the categorization posed above is a model by Bates (1995). Here, it is suggested that a generic NLP system consists of five components:

1. A *parser* that conducts syntactic analysis;
2. A *semantic processor* for representing the "meaning" of the text itself;
3. A *pragmatic & discourse processor* that also represents meaning, however in this case the context is also used;
4. A *reasoner*, that reasons how the response should take form;
5. An *action and response generator*, creating the output of the NLP system.

In this case, the first three components are more or less congruent with the three categories of the NLP curves in the previous section. The reasoner uses the input of the previous components and creates a response from it. This response is context-specific and adapts to the original input. The reasoner might not always be needed, but is useful with ambiguous requests. Finally, the response planned by the reasoner is generated by the action and response generator and propagated to the user of the system. A figure of this model can be seen in Figure 3.1.

These two categorizations are alike because they are both based on subfields of linguistics (Briscoe, 2011):

- **Phonetics:** The study of sounds that humans can make;
- **Phonology:** The study of how human sounds can be used in language;
- **Morphology:** The study of how words are organized, and what they mean;
- **Lexicon:** The vocabulary of a person or language;

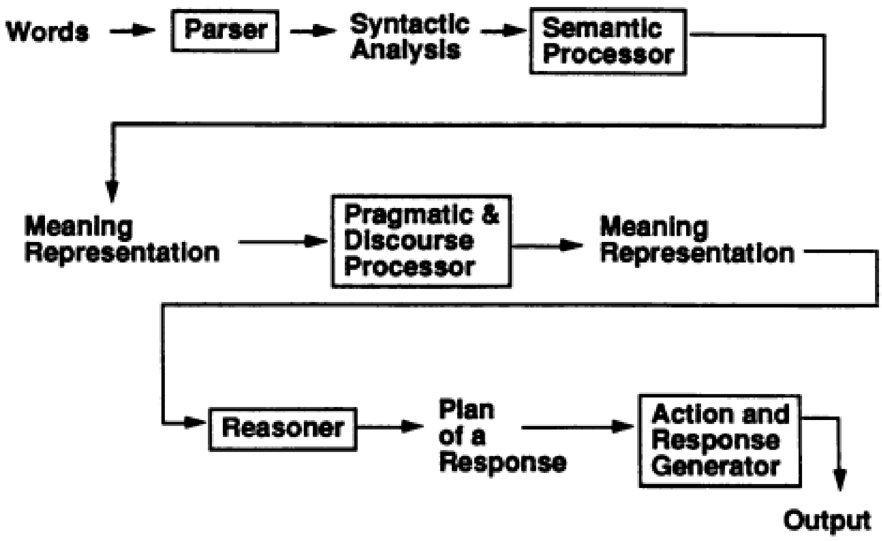

FIGURE 3.1: A pipeline model for a generic NLP system, illustrated by Bates (1995).

- **Syntax:** The study of how words can/should be put together to form sentences;
- **Semantics:** The study of the meaning of words and sentences;
- **Pragmatics:** The study of how language is used in a particular context.

NLP tools are often not categorized, and when they are, it is usually either according to the structure of the toolkit that implements them (Bird and Loper, 2004; Manning et al., 2014) or according to their linguistic task.

3.3.1 Categorizations of NLP within requirements engineering

For the field of RE, two categorizations have also been found. The first type of categorization, proposed by Berry et al. (2012), has 4 categories for tools:

1. Detection of defects and deviations from good practices for requirements documents;
2. Generation of models from descriptions in natural language;
3. Creation of traceability links between requirement descriptions, or between requirements and artifacts;
4. Identification of key abstractions, in order to get a quick understanding of the field (similar to automatic summarization tools).

The second categorization is proposed by Nazir et al. (2017), and uses phases of RE wherein the tools are used instead:

- **Classification:** Classification of requirements into categories, for example into functional and quality requirements;
- **Prioritization:** Giving a priority to certain requirements over others;

- **Ambiguity removal:** Removing ambiguity from formulated requirements;
- **Requirements assessment:** Analyzing the impact of a requirement to the system;
- **Requirements elicitation:** Gathering requirements from users and stakeholders;
- **Requirements analysis:** Performing analysis on the gathered requirements.

Chapter 4

Requirements elicitation

RQ2: *What is the definition of requirements elicitation, and what activities does it encompass?*

4.1 A definition of requirements elicitation

Several definitions of requirements elicitation have been proposed by multiple authors:

- Zowghi and Coulin (2005): *"The process of seeking, uncovering, acquiring, and elaborating requirements for computer based systems."*
- Hickey and Davis (2004): *"Learning, uncovering, extracting, surfacing, or discovering needs of customers, users, and other potential stakeholders."*
- Paetsch, Eberlein, and Maurer (2003): *"Requirements elicitation tries to discover requirements and identify system boundaries by consulting stakeholders (e.g., clients, developers, users). System boundaries define the context of the system. Understanding the application domain, business needs, system constraints, stakeholders and the problem itself is essential to gain an understanding of the system to be developed."*
- Holtzblatt and Beyer (1995): *"Requirements definition / requirements gathering / requirements elicitation / requirements engineering—all phrases for "figuring out what to build." "*

Though all these definitions describe requirements elicitation in a different way, the essence is that requirements elicitation the phase is wherein the needs of the stakeholders for the software system are gathered and the scope of the software system is defined. These needs, the requirements, can then be further processed in other steps. In the model by Kotonya and Sommerville (1998) (see Figure 1.3), requirements elicitation is portrayed as the first phase of the process. It requires an understanding of the stakeholders and the domain in order to effectively extract requirements from both. Requirements elicitation is also known as requirements definition, requirements acquisition or requirements gathering, but elicitation is the most commonly used term.

Requirements elicitation is important. In a case study conducted by Martin et al. (2002) at two software companies, requirements elicitation was the sole RE phase that was always explicitly executed. Kujala et al. (2005) emphasizes that user involvement in system development, which is carried out primarily during the requirements elicitation phase, is essential to project success. Bano and Zowghi (2013) also noted this phenomenon. Another testimony of the importance of requirements elicitation is stated by Burnay, Jureta, and Faulkner (2014):

"Elicitation is important, because misunderstanding stakeholders, or in some other way missing important information, can result in the specification of the wrong system - one that fails to satisfy requirements, and/or is inconsistent with the conditions in its operating environment (e.g., it does not comply with applicable legislation)."

4.2 Requirements elicitation process

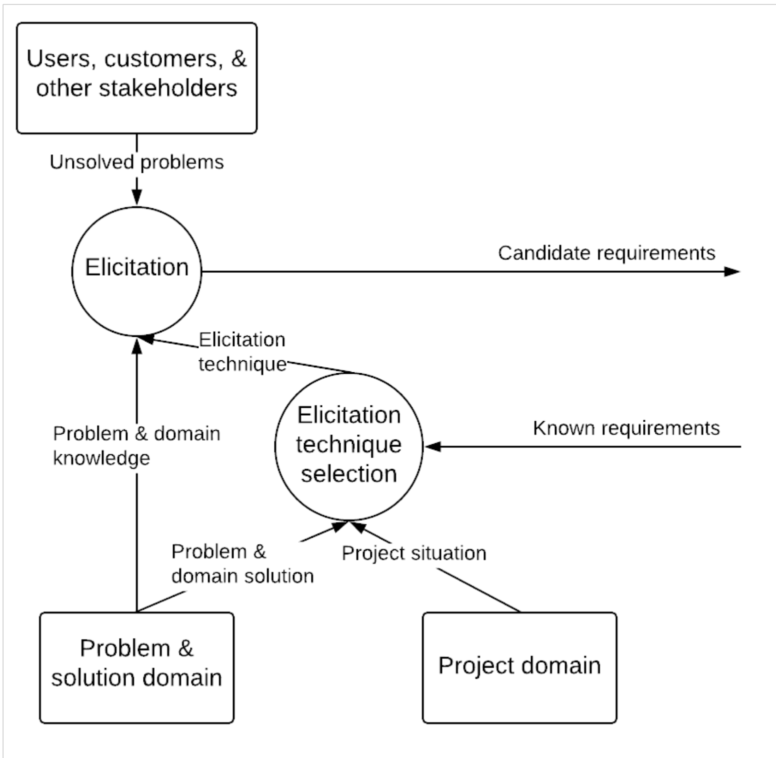

FIGURE 4.1: The Unified Requirements Elicitation Model, proposed by Hickey and Davis (2004).

A model for the process of requirements elicitation has been created by Hickey and Davis (2004); See Figure 4.1. In the model, there are two activities that are executed: elicitation and elicitation technique selection. The arrows represent flows of knowledge, and the annotations of these flows is the specific knowledge that flows. The rectangles represent the sources of the knowledge. A formal description is given in the article, but this thesis will describe informally how elicitation is conducted using this model.

1. The first step is to understand the known requirements, and what requirements are still missing. These known requirement can be based off earlier elicitation sessions, or implemented in a previous release, should the system be extended instead of created.

2. Following this, a requirements analyst must understand the characteristics of the project domain, so the domain in which the software system is built/extended. Examples are inherent characteristics of the company or project, characteristics

of the users/customers, developer characteristics (in relation to their experience with the project domain and the problem/solution domain), resources and time available, and characteristics of the elicitation analysts themselves.

3. Next, the problem domain's characteristics need to be understood. These characteristics are the understanding of the problem (for which a software system is built or extended) and its complexity, and if there are any existing systems.

4. In the same step, the solution domain's characteristics are analyzed: what type of software system is expected to be created or what kind of extension should be built, and what the development method is (in-house, Commercial Off-The-Shelf, outsourcing).

5. Based off the understanding of the project domain and the problem & solution domain, as well as known requirements, the elicitation technique can be chosen. Available elicitation techniques will be discussed in the next section.

6. Meanwhile, users, customers and other stakeholders need to be gathered for the elicitation session(s), so that they can express their unsolved problems.

7. When the stakeholders are gathered and the elicitation technique is chosen, this elicitation technique can be performed with the stakeholders to gather candidate requirements. Knowledge of the project domain and the problem domain can be used by analysts in the elicitation sessions. The candidate requirements can then be used in the next step within the RE process: requirements analysis.

An addition to this model would be the activity of stakeholder identification: identifying all parties that have a stake in the development of the software system (Sharp, Finkelstein, and Galal, 1999). Consequently, analysis (Bryson, 2004) can be carried out so that the understanding of their needs is improved, which also fortifies the elicitation. Stakeholders can also be prioritized (Parent and Deephouse, 2007); however, this is more relevant for the phase of requirements prioritization than for requirements elicitation.
The process of requirements engineering is also broken down by Zowghi and Coulin (2005). This process is mostly the same as the model from Hickey and Davis (2004), though with an extra activity for identifying the sources of the requirements. Moreover, they have an explicit step for analyzing the stakeholders, whereas in the model of Hickey and Davis (2004), this is a suggestion given implicitly.

4.3 Elicitation techniques

There are many techniques available for requirement analysts to elicit requirements from stakeholders. Zhang (2007) gives the most complete overview of these techniques, including a categorization. Thus, this article will be used as source for this section. The four categories listed will each feature a description, as well as the techniques that belong to each category.

4.3.1 Conversational methods

Techniques of this category focus on analysts and stakeholders in direct conversation with each other to determine what requirements exist for the software system in development. How this conversation takes form differs per technique. These techniques are carried out to determine key product features. The following techniques are listed:

- **Interview:** The analyst asks the stakeholders questions about desired requirements, and the stakeholder answers them. Interviews can be structured (predetermined set of questions), unstructured (no questions available beforehand, these are shaped during the interview) and semi-structured (some questions available beforehand, but elaboration and asking different questions is possible). Surveys and questionnaires can also be seen as a form of interview.
- **Focus group:** A session focusing on determining the key features of the software system.
- **Brainstorming:** A session focused on generating as much ideas for the system as possible. Innovation is key in this sessions, and that is why the focus is on ideas and not necessarily on features.

4.3.2 Observational methods

In this category, the stakeholders are observed by the analyst while hey are doing their work or a specific task. An advantage of this is that communicating requirements that are hard to express in speech, also known as tacit requirements (Stone and Sawyer, 2006), are easier to observe in a natural setting than elicited during an interview. The following techniques belong to this category:

- **Etnographic study:** An analyst spends a period of time at a company, and analyzes the culture and practices of that company in order to elicit requirements.
- **Protocol analysis:** A stakeholder performs a task, and thinks out loud about issues that s/he encounters during the execution. This task is good for modelling workflows.

4.3.3 Analytic methods

In an analytic technique, elicitation is conducted directly from documentation that the company has on hand (for example, about legacy systems or legislation), or extracting the knowledge from experts within the company about the system or the purpose it fulfills. These methods are often complementary to conversational and/or observational methods.

- **Requirements reuse:** Extracting requirements from the documentation of older / legacy systems and listing these for reuse in the new system.
- **Content analysis:** Extracting useful and relevant information from documentation that the company has on hand. This is more focused on the company itself and the market than specifically on the systems, which are covered by the requirements reuse technique.
- **Laddering:** An expert is asked by an analyst about the values to be seen in other systems. By follow-up questions, the analyst tries to determine consequences and attributes of these values (Reynolds and Gutman, 1988).

- **Card sorting:** The experts is given cards that represent functions or attributes of the system, and is asked to sort them into groups. These groups can represent functions in the system, but also priority categories.
- **Repertory grid:** A grid is given within a certain topic, with elements (representations of the topic) on one axis and attributes on the other axis. The expert is asked to value each attribute for each element (Niu and Easterbrook, 2007).

4.3.4 Synthetic methods

It is often not enough to use just one technique for the elicitation phase, because of different perspectives of different stakeholders and the form that the requirements take. As such, there are also techniques that combine elements of the previous categories, as well as other techniques not discussed in this chapter. The following techniques are provided:

- **Scenarios:** Describing a sequence of actions that must be executed to attain a goal within the system.
- **Prototyping:** Building a prototype (early first version) of the system, and performing an evaluation on it. This is often done to elicit extra requirements, after the first set have already been implemented; this first set is used to build the prototype.
- **JAD/RAD sessions:** Abbreviations for Joint/Rapid Application Design. In these sessions, all stakeholders are involved to discuss the main problems of the software system and their solutions. The difference with brainstorming is that these are better structured, and the main goals of the system are often already established (Zowghi and Coulin, 2005).
- **Contextual inquiry:** Observing the stakeholder in their work setting and asking questions about a focused subject, leading to requirements for that subject (Raven and Flanders, 1996).

All these techniques have their advantages and disadvantages. For example, contextual methods, contextual inquiry and an ethnographic study are very labor-intensive for the analyst, while creating a prototype is expensive and takes a long time. Research into the effectiveness of some of these methods has also been performed (Davis et al., 2006; Dieste and Juristo, 2011). Luckily, there has also been plenty of research to elicitation technique selection (Hickey and Davis, 2003; Carrizo, Dieste, and Juristo, 2014; Kausar et al., 2010; Hickey and Davis, 2007); following the guidelines in these articles should prove to help selecting the best elicitation technique for each situation.

4.4 Requirements documentation

After the requirements have been elicited from stakeholders or other sources, they need to be written down in order to be further processed. The phase that encompasses this task is defined as requirements documentation (Engel et al., 1993), also known as requirements authoring. The artifact that is produced by this phase is most often known as the Software Requirements Specification (SRS): a list of all current requirements elicited. This artifact, consequently, is then used as the input for the requirements analysis phase.

Requirements in the SRS can be formatted in numerous ways. One of the most well-known requirement formats within agile software development is the user story (Cohn, 2004). The format is as follows:

as a <role>, I want to <goal>, [so that <reason>].

User stories split requirements into three parts: the role, the goal and the reason. The role describes for which role in the system the user story applies. That is because different stakeholders and users have different needs and permissions, and sometimes user stories are also assigned to roles that are not necessarily acted out by human users of the system. For example, in a Computer-Aided Design (CAD) system, it does not make sense for executive management to edit some of the product models; however, they might be interested in generating a report that includes images of these models.
The goal answers what it exactly is that the role is trying to achieve. The aforementioned difference between functional and non-functional requirements can also be implemented here: the goal might describe a function that the system needs to be able to execute, but also how the function must be executed. Taking the previous example from the CAD system, a functional goal might be linking specific designs to specific manufacturing belts, and a non-functional requirement might be that the manufacturing belts must start manufacturing within 5 seconds of receiving the design for a new product.
The last part of a user story is the reason. This is an optional clause that describes why the role wants to achieve the goal. As it is optional, not every user story has it added nor needs it; however, it is good form to add when the goal itself can lead to ambiguity or unclarity. Exemplifying this with the CAD system, a designer might want to import images to the model. Now, without context, this might not make sense. However, the reason is that the product designers deliver the textures for the product via PNG-format, and when imported, it can be wrapped around the product design.
Applied to the CAD system, a user story might look like this:

As a QA manager, I want to edit materials in the manufacturing process, So that I can make prototypes of multiple versions with multiple materials to select the most robust version.

Tools available for requirements documentation are often tools that support the writing of the requirements in a specific format. Sateli, Angius, and Witte (2013b) is a good example of a tool for requirements documentation. This tool supports collaborative requirements documentation, including lightweight text analysis to support the authoring itself. Another example is Silveira et al. (2005); however, this tool does not feature integrated text analysis. What is often integrated in requirements documentation tools is requirements traceability: linkage between a requirement and the part of the system that it is implemented on. However, traceability will not be further discussed as it is not in the scope of the SLR.

Chapter 5

Requirements analysis

RQ3: *What is the definition of requirements analysis, and what activities does it encompass?*

5.1 A definition of requirements analysis

Defining requirements analysis is not an easy task, because of the lack of agreement in literature. In Figure 1.3, requirements analysis and negotiation is clearly portrayed as the second phase of the RE process; however, no clear definition could be extracted from the model. A second view is given by Maciaszek (2007): *"Business analysis (or requirements analysis) is the activity of determining and specifying customers' requirements."* This definition is ambiguous, as determining requirements is also a goal of the requirements elicitation phase, discussed in Chapter 4. Elicitation, however, is not noted in the book; thus, it makes sense that the author combines the two. Unfortunately, this does not support our view of requirements analysis.
Another definition is given by Byrd, Cossick, and Zmud (1992): *"Requirements analysis (RA) involves end users and systems analysts interacting in an effort to recognize and specify the data and information needed to develop an information system."* Again, this definition seems to incorporate requirements elicitation into the process, by mentioning the interaction between end users and systems analysts. In our opinion, the analysis phase does not involve further interaction with the stakeholders, as this has already been done during the elicitation phase.
Hay (2003) notes requirements analysis as seven processes:

1. Define scope;
2. Plan the analysis;
3. Gather information;
4. Describe the enterprise;
5. Take inventory of current systems;
6. Define what is required of a new system;
7. Plan the transition.

Again, this is a process for the entire RE process rather than analysis in specific. Elicitation is captured in the 'Gather information' process, and the requirements are defined in the sixth process.
A final definition we would like to discuss is from Grady (2010): *"System requirements analysis is a structured, or organized, methodology for identifying an appropriate set of resources to satisfy a system need and the requirements for those resources that provide a*

sound basis for the design or selection of those resources. It acts as a transformation between the customer's system need and the design concept energized by the organized application of engineering talent. The basic process decomposes a statement of customer need through a systematic exposition of what the system must do to satisfy that need. The need is the ultimate system requirement from which all other requirements and the designs flow." The transformation of the customer's need is, in our opinion, central to requirements analysis; the input from the elicitation phase must be transformed to a set of requirements that the software developers can incorporate. Also, a structured methodology is in our belief important to a correct execution of the RE process. However, this definition does not entail what exactly that methodology is. A second issue is with the "ultimate-system requirement from which all other requirements and the designs flow": this is, in our opinion an unclear statement.

Looking through the four definitions, there is little consensus over the scope of requirements analysis: where exactly does it begin, what is the input, and what output does it generate? Moreover, the exact activities during this phase are mostly vague or are focused on too broad a scope. One could even argue that multiple of these definitions mostly encompass RE in general, instead of being specific on how elicited requirements can be further processed. For the purpose of this thesis, elicitation has already been defined as a separate phase; in our opinion, this made more sense than to combine the two phases. However, the confusion is understandable; in job offers, the title 'requirements analyst' and 'requirements engineer' are often interchangeable, with the former having slightly more hits on Indeed, a job platform (244 vs. 196 as of December 20, 2018).
Thus, we propose to use our own definition of requirements analysis, based off four sub-phases in requirements engineering: requirements modeling, requirements classification, requirements verification and requirements prioritization. These four phases all analyze the output of the elicitation phase, and further manipulate it in order to get a final list of requirements that can be used for the system design. Thus, we believe that these four phases are appropriate for a conjoining into requirements analysis. A description of each phase will be featured in the next section of this chapter.
Therefore, for the purpose of this thesis, we propose the following definition for requirements analysis:

Requirements analysis is the analysis of stakeholders' needs for a software system, and further processing these needs so that they can be used for a final list of requirements for the software system. This phase involves but is not limited to classifying the requirements, modeling the requirements, performing verification of the requirements, and prioritizing the requirements.

5.2 Activities during the analysis phase

5.2.1 Requirements classification

A well-known activity performed on requirements after they have been elicited is to categorize them according to pre-determined categories. This activity is known as requirements classification (Ko et al., 2007a). A well-known scheme that has already been discussed in Chapter 1 is making a distinction between functional and non-functional/quality requirements. This specific categorization is often done early on

in the RE process; however, there are also other types of classification that are performed as an analysis task on the Software Requirements Specification.
A distinction is between manual classification and automated classification. Though both serve the same purpose, manual classification is done by requirement engineers whereas automated classification is performed by an algorithm. In both cases, prior knowledge is employed to correctly classify the requirements. In the case of manual classification, this is the expert knowledge of the requirements engineers. Automated requirements classification makes use of training data (pre-classified requirements that are similar to the requirements that are about to be classified) in order to classify the test data (the requirements to be classified) as correct as possible. Both have their fallacies: manual classification depends on the expert's knowledge, and automated classification depends on the training data and the algorithm to be sound. However, a trend is visible in the RE industry towards automated requirements classification, as it is far less time-consuming.

Aside from the classification functional / non-functional, there are several other schemes that can be discussed. The first is discussed by Hughes, Rankin, and Sennett (1994). In this taxonomy, a distinction is made between concerns and frames. Concerns describe what concerns the users have regarding the functionality or properties of the system. An example would be a functional concern, usability concerns, cost concerns and security concerns. Frames are specific technical views that abstract the system to make a system model. Examples would be an organization frame that models the different departments of the organization where the system is being made for, or an information structure frame that describes how the information used in the system is structured and related to other information.
A second taxonomy is the BABOK taxonomy (Brennan, 2009). This taxonomy makes use of functional and non-functional requirements, as well as transitional requirements: how the system should go from one state to the other. Moreover, it makes use of several other requirement types:

- Solution requirements: supercategory of the functional, non-functional and transitional requirements. Describes the requirements that the "solution", e.g. the system in development, should incorporate.
- Stakeholder requirements: supercategory of the solution requirements. Expresses the needs of the stakeholders and users of the system.
- Business requirements: the goals of the business itself. Supercategory of the stakeholder requirements.

As implied by the use of supercategories, this taxonomy is hierarchical: the higher up, the more abstract the requirements become.
A last example taxonomy to be discussed in this subcategory is a taxonomy of non-functional requirements, discussed by Cleland-Huang et al. (2007a). This article describes an NFR classifier that further decomposes the non-functional requirements. The categories are: Availability, Legal, Look-and-Feel, Maintainability, Operational, Performance, Scalability, Security and Usability. These categories were determined by mining non-functional requirements beforehand and using these as a training set.

Requirements clustering

Automated classification is also known as supervised learning. The algorithm "learns" where to classify data based on the "supervision" of the training data. However,

there is also unsupervised learning, also known as clustering (Duan, 2008). The difference is that clustering algorithms do not make use of a training set; instead, they cluster the items in the data set based on similarity to each other. This helps in identifying similar data items, but post-processing will have to be performed in order to see why those clusters are created by the algorithm.
Both requirements classification and requirements clustering are often based on word similarity between the requirements. However, Abad et al. (2017) suggests some preprocessing of the requirements beforehand for both classification and clustering, as the data might have words that are not relevant or words that are of the same verb, but are not alike (like forms of *to be*: am, are, is, was, ...). Thus, NLP techniques like tokenization, stemming, lemmatization and/or stop word removal are very applicable in this case.

5.2.2 Requirements modelling

Requirements modeling is the activity wherein the stakeholders' needs are represented into a model (Santos Soares, Vrancken, and Verbraeck, 2011). This model is an abstraction of the requirement or set of requirements. Modeling is a way of documenting the requirements; however, it is not the same as the requirements documentation phase, as that phase is conducted before the analysis phase.
Modeling requirements is a way of bridging the gap between the user's needs and the final requirements (Kujala, Kauppinen, and Rekola, 2001). The user's needs that are extracted during the elicitation phase are not yet ready to be used as requirements for the documentation. Modeling is useful for a correct format and a structuring of the requirements. It can be used to represent the goals of the system and what requirements contribute to this; this is also known as goal modeling.
An advantage of modeling requirements is to give several perspectives on the requirements. Depending on the modeling techniques used, one can, for example, get a data-centric model, an information-centric model, a goal-centric model, or a user-centric model. Creating models with these viewpoints and checking them against one another can help in the uncovering of errors, for example an incorrect data flow from the system to the user.

Modeling approaches can be categorized in several ways, but the categorization that we will use is based on the Handbook for Requirements Modeling, published by the International Requirements Engineering Board (*Handbook of Requirements Modeling IREB Standard*). In this handbook, five views on requirements and requirements engineering are proposed:

- **Context view:** Modeling the context of the system, so other systems that it interacts with, the company for which it is developed and users and stakeholders of the eventual system. Context diagrams are used for this purpose.
- **Information-structure view:** Modeling the static and structural aspects of the functionality of the system. This could be information needed in the system, or data processed by it. Diagrams often used in this view are class-responsibility-collaborator cards (Wilkinson, 1998) or entity-relationship diagrams (Chen, 1988).
- **Dynamic view:** Modeling the dynamic aspects of the functionality, for example how a user executes an action or how data flows. There are five sub-views for the dynamic view:

  - *Use case view:* Models what actions users must carry out to reach a goal in the system; see Figure 5.1 for an example of a use case diagram.
  - *Data-Flow-oriented view:* Models the functions of the system, and data dependencies between these functions and actors in the system. Data flow diagrams are often used.
  - *Control-Flow-oriented view:* Models the processes/activities within the system, and the flow between them. Activity diagrams are typically used, as well as Business Process Modeling Notation models (Chinosi and Trombetta, 2012).
  - *Scenario view:* Models the interaction between the actors and the system; often used to make use cases more specific. Scenarios can be used, as well as message sequence charts and UML sequence diagrams.
  - *State-oriented view:* Models the states that the system is in, depending on the events happening within the system. Good models for this purpose are finite automata and UML state machine diagrams.

- **Quality view:** modeling the non-functional requirements/quality requirements. Quality requirements are often annotated in other models instead of having their own models.

- **Constraints view:** Models the constraints. Constraints are boundaries of the system: they are things that the system should not do, or limits on requirements that can be expressed numerically (for example, a max speed of 128mB/s for a router). Constraints are often textualized, but class and component diagrams can be used for organizational and technical constraints as well.

The views arguably most relevant to NLP are the constraints view, the quality view, the context view, the use case view and the scenario view; these views depict the requirements mostly in a textual way.

A distinction can also be made between informal, semi-formal and formal models. Informal models are often graphic models, and use natural language in it. They are good for a visual description of the system and its components and requirements. However, no statistical analysis can be performed. Formal models are often in text, and are made as such that statistical analysis is possible and encouraged. However, they are hard to understand for those unfamiliar with the syntax of the model. Semi-formal is a hybrid of these two: it is understandable, but limited statistical analysis can be performed as well.

5.2.3 Requirements verification

The next sub-phase is one of checking if the requirements are actually correct and qualitatively sound. Checking if the requirements are correct is a sub-phase is known as requirements verification (Fanmuy, Fraga, and Llorens, 2012). In this phase, the requirements listed are checked manually or mechanically to see if they are clear, correct and unambiguous, so that they can be further used for the design of the system. Requirements are also further refined where needed.

Important to note is the difference between requirements verification and requirements validation. Simply put, as by Boehm (1984), validation is "doing the right thing" and verification is "doing the thing right". Validation is done during the phase that the requirements are implemented, as at that point the requirements are checked to see if they are implemented. Verification, on the other hand, is centered around

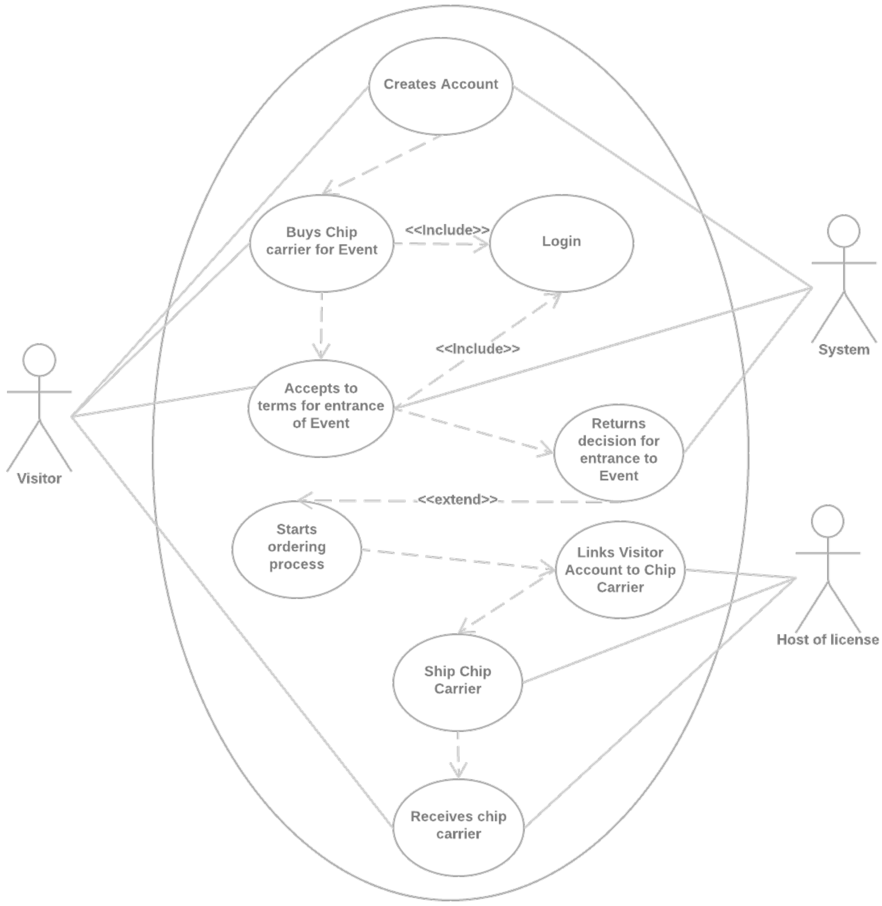

FIGURE 5.1: A use case model, in this case of a user of a system obtaining entrance to an event.

the requirements themselves and if they are comprehensible for the analysts. They are checked for spelling and grammatical errors, ambiguity and the like. Completeness is also an important factor in the verification process.
Sakthivel (1991) lists the most common requirement errors. These are:

- **Incomplete:** When there are parts missing in the specification of the requirement;
- **Inconsistent:** When specifications of requirements are not aligned/are conflicting;
- **Infeasible:** When the system is not capable of implementing the requirement;
- **Untestable:** When specifications are ambiguous or imprecise;
- **Redundant:** When multiple requirements are to implement the exact same functionality or quality;
- **Incorrect:** When wrong, non-performable or inefficient processes are being used, when two processes generate the same output, when precedence relationships in processes are wrong, when system properties are violated or processes are timed incorrectly.

The purpose of requirements verification is to check for these issues, and for refinement to remedy them.
Sakthivel (1991) also gives an overview of which techniques were in use at that time for requirements verification. However, almost 30 years later, it is unclear if some of these techniques are still applicable and recent. Fanmuy, Fraga, and Llorens (2012) notes that limited use of NLP within the verification phase is taking place, something that this research explores further.

5.2.4 Requirements prioritization

The last sub-phase of requirements analysis to be discussed is requirements prioritization. Prioritization involves ranking the requirements in order of preference (Berander and Andrews, 2005). This ranking can be done on the basis of multiple variables; examples are time, budget, ease of implementation and preference by stakeholders. Requirements prioritization is sometimes referred to as requirements negotiation; however, nowadays prioritization is used more commonly.
Prioritization is very important (Lehtola, Kauppinen, and Kujala, 2004). Most notably, it guides the release planning, as each release has a set of requirements to be implemented. Developers will have to choose between several options due to time constraints; this is most notable in agile scrum programming, where the developers work with short development cycles of a few weeks. It is the task of the developers to balance time, budget, critical stakeholders and such in order to get the best set of requirements per release. This is what requirements prioritization strives for.
Requirements prioritization is a well-researched subject, with a plethora of techniques available; an overview is given by Achimugu et al. (2014). Three notable techniques will be discussed here. The first is the Analytic Hierarchy Process (AHP) (Saaty, 2008). This method is illustrated with a fictitious example in Figure 5.2.
In AHP, the decision makers need to agree for the best option for a goal. In the

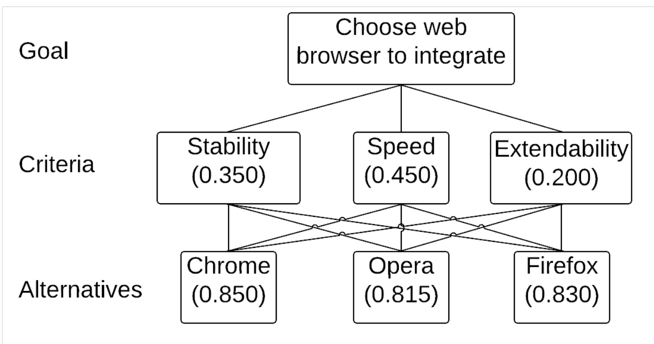

FIGURE 5.2: A fictitious example of Analytic Hierarchic Processing.

case of the example, the requirement is that a web browser needs to be integrated. But which web browser? Three are proposed: Google Chrome, Opera and Mozilla Firefox. These are evaluated on three criteria: stability, speed and extendability with plug-ins. Each of these criteria are weighted, with speed being the most important and extendability the least important. The decision makers then score the alternatives based on the criteria. An average is taken for each criterium for each alternative, which gives an overall score if each is again averaged per alternative in total. The alternative that scores best is then proposed as the best implementation.

The second technique is the MoSCoW technique (Hatton, 2008). MoSCoW is an informal method that prioritizes requirements into four categories:

- ***M**ust have*: requirements that must definitively be included;
- ***S**hould have*: requirements that have high priority, but can be excluded if time / resources are lacking;
- ***C**ould have*: requirements that have a lower priority, but should be included if there is enough time and resources after the previous categories have already been included;
- ***W**on't have*: requirements that should definitively be excluded.

Similar to MoSCoW is the requirements triage technique (Laurent, Cleland-Huang, and Duan, 2007); this technique has the categories `should definitely be included' and `should definitely be excluded'; all other requirements are put in the category `would be nice to have'.
The final technique to be discussed is the Binary Search Tree; see Fig 5.3 for an example. Back to the web browsers example. Listed are requirements that a web browser

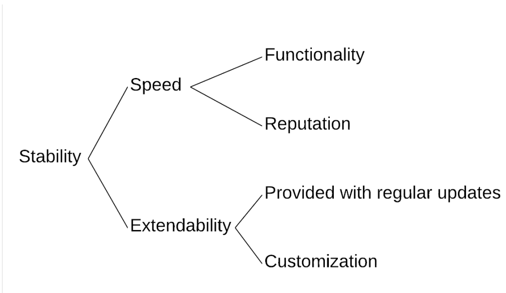

FIGURE 5.3: A fictitious example of a Binary Search Tree.

should have. One starts with the first criteria: this is the root level. Each requirement is first compared to the root. If the requirement is more important than the root, it is placed above it; if it is less important, it is placed below it. The same process is repeated for each direct child of the root, until it cannot be further compared, at which point it gets its own place. As such, the order of priority in the example is:

1. Functionality;
2. Speed;
3. Reputation;
4. Stability;
5. Provided with regular updates;
6. Extendability;
7. Customization.

Chapter 6

Research execution

This chapter gives an overview of how the Systematic Literature Review was conducted. The protocol is provided, and described per phase and sub-phase. Moreover, issues encountered are also discussed along with their solutions, as well as adjustments and deviations from the template protocol.

6.1 Protocol for the SLR

As noted in Chapter 2, a Systematic Literature Review needs to have a pre-established protocol in order to ensure a correct execution of the research. This protocol, along with the responsibilities assigned to numerous parties within the research itself, are shown in Table 6.1. The protocol is further discussed in the next sections, where deviations from the standard protocol from Kitchenham (2004) are also justified.

6.2 Planning phase

This phase was concerned with planning the SLR. This includes identifying the need for an SLR, developing the protocol and reviewing it.

6.2.1 Identify need

The need for the SLR was identified through the feasibility study discussed in Chapter 2, as well as unstructured literature search afterwards. First, the feasibility study showed a plethora of NLP being used within the domain of RE. Next, through the literature search, it was revealed that though some literature was available for an overview of the domain, its quality was questionable or the overview wasn't fully applicable to the domain of this research. For more details, please refer to Chapter 1.

6.2.2 Develop protocol

Since there was a clear need for the SLR, the protocol was established next. As noted previously, this protocol has been based on guidelines from Kitchenham (2004); however, some modifications were made to better suit our needs and the structure of the data. Some of these modifications were made beforehand, however some were made during the research, when encountering issues not anticipated beforehand. Though this is not standard procedure for a protocol for an SLR, we believe that the validity of the research has not been compromised as the general protocol of this SLR is more or less congruent with the standard protocol. We will discuss these changes in their respective sections within this chapter.

TABLE 6.1: Procotol for the SLR, including responsibilities. DJ stands for Duc Janssens, the student; FD stands for dr. Fabiano Dalpiaz, the first supervisor; MR stands for Dr. Marcela Ruiz Carmona, the second supervisor; and AV stands for dr. Andreas Vogelsang, of the Technische Universität Berlin.

Phase	Phase description		Responsible			
			DJ	FD	MR	AV
1	**Planning phase**					
1.1	Identify need		x			
1.2	Develop Protocol					
1.2.1		Establish research questions	x	x		
1.2.2		Review protocol and research proposal	x	x	x	x
2	**Conducting phase**					
2.1	Research identification					
2.1.1		Generate search strategy	x	x		
2.1.2		Document search	x			
2.2	Study selection					
2.2.1		Establish selection criteria	x	x		
		Selection process	x	x		
2.3	Quality assessment and data extraction					
2.3.1		Establish quality criteria	x	x		
2.3.2		Establish categories for extraction	x	x		
2.3.3		Extract data	x			
2.3.4		Validate and adjust (iterative coding)	x	x		
2.4	Data synthesis					
2.4.1		Descriptive analysis	x			
2.4.2		Quantitative analysis	x	x		
2.5	Review		x	x		
3	**Reporting phase**					
3.1	Report		x			
3.2	Review and adjust		x	x	x	

Establish research questions

Before beginning the research itself, the research questions were formulated in order to give a scope and direction to the SLR. These research questions are presented in Chapter 2. One thing to note is that research questions 1 through 5 have remained the same over the research, with only slight adjustments in the wording; however, RQ6 had at one point been a research question about NLP in other domains outside of IT (biology, chemistry, medicine etc.). During the study selection, it had been decided to scrap this research question in favor of other research questions. However, during the data synthesis, a different RQ6 was introduced as a research question that concerns patterns of which NLP tasks co-occur with other NLP tasks, RE phases and main tool purposes.

Review protocol and research proposal

After establishing the research questions and basing the rest of the protocol on them, the protocol was turned into a research proposal. This proposal also included the answers to RQ1, RQ2 and RQ3, so that the main concepts in the research were clear. This research proposal and protocol were then reviewed by both Dr. Fabiano Dalpiaz and dr. Marcela Ruiz, and readjusted by the student where needed. This research proposal was then greenlit. A shortened version of the proposal was also sent to dr. Andreas Vogelsang of the Technische Universität Berlin, whom provided some very useful advice for the SLR, such as scrapping a research question that demanded for a second SLR, and better formulating the research problem in order to convey that an SLR was needed as opposed to a Systematic Mapping Study (Petersen et al., 2008).

6.3 Conducting phase

This section discusses the execution of the conducting phase of the SLR, as well as some issues that have been encountered, how they have been resolved, and what deviations from the standard protocol have been made.

6.3.1 Research identification

This sub-phase concerns the initial search of literature: how it was conducted and what issues have been encountered. Two activities concern this phase: generating the search strategy, and performing the actual search. These are discussed below.

Generating the search strategy

Two inputs were needed before the literature search could be conducted: the libraries to be searched, and the search query to be used. Four digital libraries were used for the literature search:

- IEEE Xplore;
- SpringerLink;
- Elsevier, via ScienceDirect.com;
- Wiley Online Library.

These libraries were chosen as they are the most prevalent in the field of RE and NLP, with the most relevant journals, magazines and conference proceedings.
Next, the query was formulated. The following query to collect relevant literature within the selected libraries:

("requirements engineering" OR "requirements analysis" OR "requirements elicitation" OR "requirements gathering" OR "requirements modelling" OR "requirements verification" OR "requirements prioritization")

AND

(NLP OR "natural language processing" OR "computational linguistics" OR "text mining" OR "information retrieval")

The OR denotes that one of the arguments between the same parentheses must occur, but multiple may also occur. The AND denotes that a combination of a top argument and a bottom argument must occur in order for it to be included in the preliminary list. Lastly, all arguments (except for NLP, which is one word) are between quotes, as exactly those word combinations had to occur in order for it to be included. Without quotes, the combination of those words in any order and any place within the article could be included, which would greatly increase the number of irrelevant results leading to a much higher workload.
The arguments `computational linguistics', `text mining' and `information retrieval' have been added due to their intersection with NLP; this has been discussed in Chapter 3. One missing argument is `requirements classification'; that is because the category had been added to the scope after executing the search, when several results showed that requirements classification was also important to the phase of requirements analysis. The original scope, and thus the original query, did not include this category.

Document search

When the search strategy had been established, it was time to conduct the actual search. The queries were entered into the libraries and lists with articles and metadata were gained. However, some adjustments to the queries had to be made:

- In the SpringerLink library, the AND had to be replaced by an ampersand (&) and the OR by a verical bar (|);
- In the ScienceDirect library, a maximum of 10 arguments was allowed; therefore, the query had to be split into 4 subqueries separating NLP & Natural language processing, text mining, computational linguistics and information retrieval. The lists of results were later merged.

Moreover, depending on the library, some filters were used in order to reduce some irrelevant results:

- In the SpringerLink library, the *Content type* was set to Articles and Conference Papers, the *Discipline* was set to Computer Science and the *Language* was set to English;
- in the ScienceDirect library, only research articles were included in the search.
- in the Wiley Online library, the *Subject* was filtered to Computer Science.

TABLE 6.2: Number of results per library from the queries.

Library	# of results
IEEE Xplore	275
SpringerLink	1,139
Wiley Online Library	178
ScienceDirect	647
Total	2,239

The number of results per library can be found in Table 6.2. This list is excluding duplicates within libraries and overlaps between them.

6.3.2 Study selection

Having obtained the lists of articles from each library, the most time-consuming sub-phase was next: the study selection. In this sub-phase, articles were selected based on their relevance. Two activities encompass this sub-phase: Establishing the selection criteria, and the selection process itself.

Establish selection criteria

The selection criteria were established for a quick check per article, and not necessarily to review the entire article for data extraction. The selection criteria were based on the predetermined scope, as well as a criterion that the article had to present a novel tool, technique, approach, framework, method, process, model or something similar to this. An overview of the selection criteria is shown in Table 6.3.
Important to note with selection criterion Cr1 is that a model of the phases and their sub-phases was established based on the results of RQ2 and RQ3, to which each article was tested to see if it would fit in the scope. This model, including a legend, can be seen in Figure 6.1.

TABLE 6.3: Selection criteria for the study selection.

Identifier	Description
Cr1	The scope of the article must fall into the pre-established scope of requirements elicitation, requirements analysis or one of the sub-phases: requirements documentation, classification, modelling / formalization, prioritization or verification.
Cr2	The article must include at least 1 NLP task.
Cr3	The article must present a new tool, technique, approach, method, framework, process, model or something similar.

Selection process

The selection process was done by using a spreadsheet of the article list per library, making four spreadsheets in total, in conjunction with the articles themselves. Four columns were added to each spreadsheet: one to denote if a new tool, technique etc. was presented, one to denote which main RE phases the article fell in, and one to denote if any NLP was involved. If the article failed any of the three criteria, the article was out of scope.
The spreadsheets contained the article name, authors, year of publication, abstract, keywords, and source (both the library and a link to the article page). An assessment

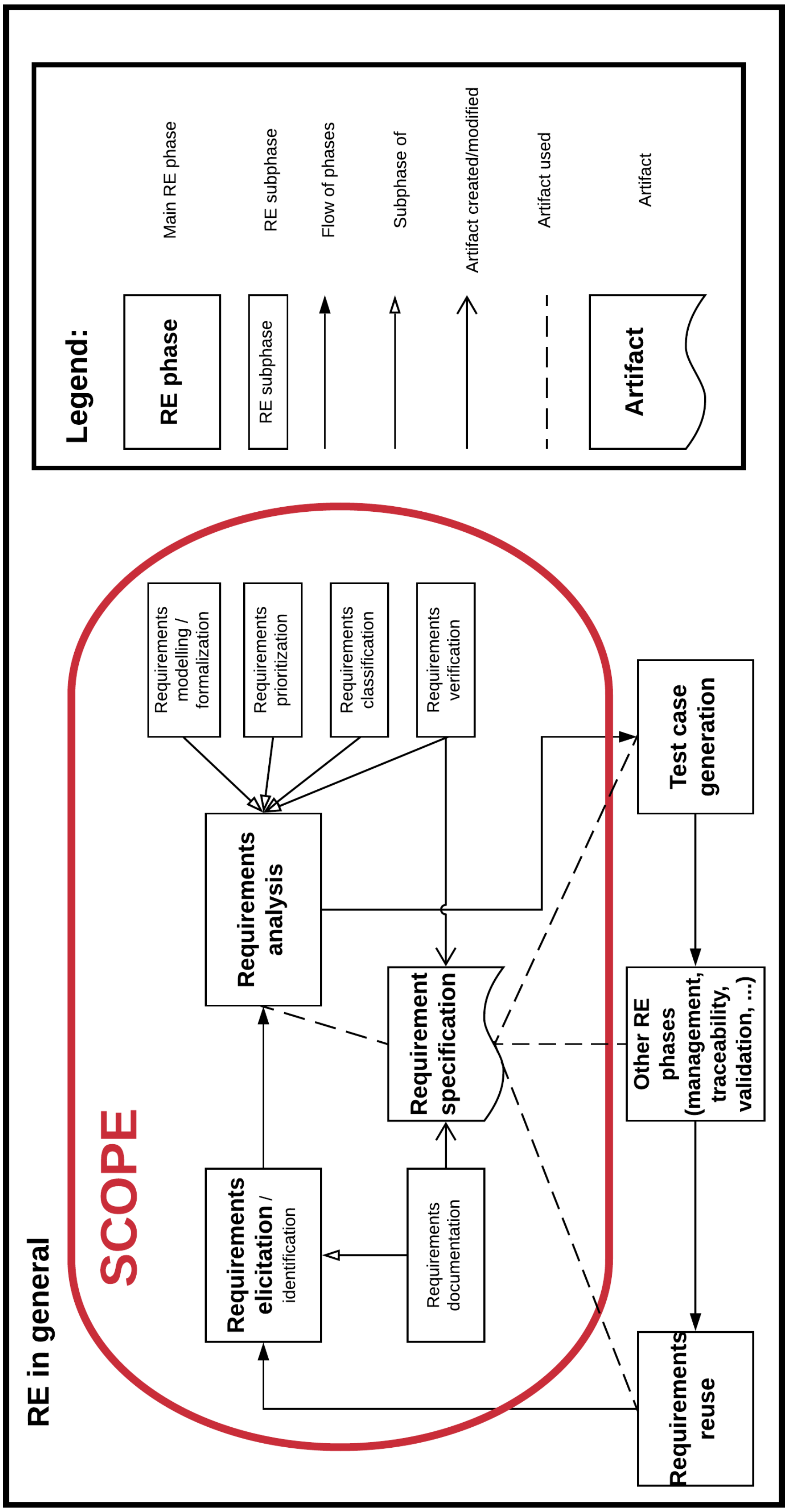

FIGURE 6.1: Taxonomy of the RE (sub-)phases that fall into the scope of this SLR. Please consult the legend for an explanation of the different shapes and arrows.

of the article could often be made based on the title, abstract and keywords: most often, these gave away if the article fell in the scope of the predetermined RE phases or if anything new was presented. However, if this could not be determined from the title, keywords or abstract, the article itself was scanned through to determine this. The article was also scanned to see whether any NLP tasks were employed.
In most cases, it was easy to determine whether an article was in scope or not. The criteria were lenient on purpose, so that the risk of not including a relevant article was small. Included articles that were not relevant after all could be removed later on. However, quite a number of articles were not relevant at all, but were included because one of the search arguments was part of the references or related works. This caused the workload to be higher than anticipated.
Should the student have issues with judging an article's relevance, the article was discussed together with dr. Fabiano Dalpiaz, and a consensus was reached. Dr. Fabiano Dalpiaz also validated a subset of the results, so that the student could readjust some of the judgments. This eventually lead to a cut-down of 2239 articles to 275 articles. These articles were downloaded and further processed using NVivo, which will be described in the next subsection.

6.3.3 Quality assessment and data extraction

This sub-phase concerned checking the articles for quality, and extracting data from the articles. Normally, these two steps would be separated. However, as we used the same tool for both the quality assessment and the data extraction, these two phases have been aggregated into one.
The tool used for this sub-phase is NVivo. NVivo is a software package for qualitative data analysis; it can be used both for analysis of text and multimedia. We used it to analyze the textual data in the articles. The articles were imported into the tool, and relevant text in the articles was selected and "coded": dragged to a predetermined category. When selecting that category, one can view all the selected text. Of this category, one can then create word clouds, cluster the text or extract it for further processing, for example as a training set for classification. A screenshot of NVivo can be seen in Figure 6.2.

Establish quality criteria

Quality criteria are criteria that examine and evaluate the overall quality of the research in the article. These do not concern establishing if the article fits in the scope, but if the research in the article is apt enough to be included in the final data set. An overview of the criteria used in this research can be seen in Table 6.4.
The quality criteria presented were created in consensus with dr. Fabiano Dalpiaz. Notable is some of the phrasing, for what makes something "clearly identified or derived"? With this is meant that the article needs to state somewhere the (sub-)phase, research gap, tool purpose or NLP task. The reason is that it can then be extracted using NVivo. Though preferred, it is not necessary that the article outright states this: for example, an article does not have to state it's phase as `requirements modelling'; it can also be coded if it states `extracting UML sequence diagrams from requirements'. An important note here is that though derivation could be done for the research gap, RE phase and tool purpose, this has not been done for NLP tasks because of the avoidance of implicit tasks. This will be further explained in Chapter 10.

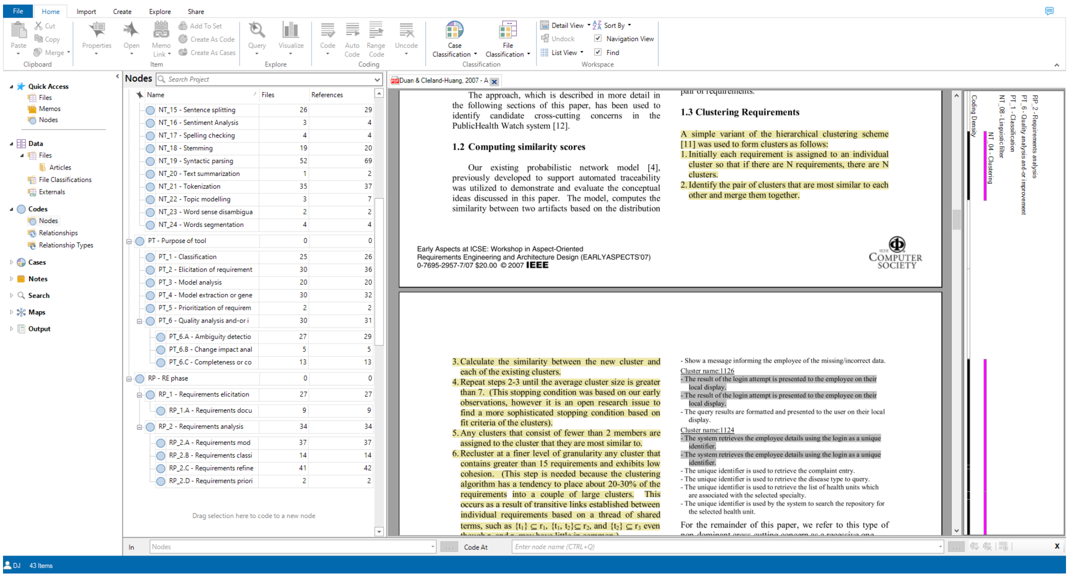

FIGURE 6.2: A screenshot of NVivo. The article can be seen as the largest part of the interface; yellow highlighted text is text that has been coded to a certain category. The categories are displayed left of the article, while "coding stripes", stripes that indicate where text has been coded to a certain category, are featured on the right of the article.

TABLE 6.4: Quality criteria used during the quality assessment sub-phase of the SLR.

Identifier	Description
QC1	The RE (sub-)phase that the research fits in can be clearly identified or derived.
QC2	The research introduces a tool, or the technique, approach, method, process, framework, model or similar is tool-supported.
QC3	The research gap that the tool or tool-supported technique/approach/etc. addresses can be clearly identified or derived.
QC4	The main purpose of the presented tool or tool-supported technique/approach/etc. can be clearly identified or derived.
QC5	The NLP tasks used by the tool or tool-supported technique/approach/etc. can be clearly identified.
QC6	The research features an evaluation of the tool or tool-supported technique/approach/etc. in some way, for example a case study or empirical evaluation.

TABLE 6.5: Coding categories established in NVivo.

Name	Description	Quality criterion
Abstract	Marks the abstract of the article, if present.	
Keywords	Marks the keywords of the article, if present.	
RE phase	Marks the RE phase(s) or sub-phase(s).	QC1
Research gap	Marks the research gap that the article aims to solve.	QC3
Purpose of tool	Marks the main purpose(s) of the tool.	QC4
NLP task	Marks the NLP task(s).	QC5
Actionability	Marks if the technique, approach, algorithm, framework, process, model, method or similar is actionable, e.g. tool-supported. Tools are automatically actionable.	QC2
Evaluation	Marks the evaluation of the tool or tool-supported technique/approach/etc.	QC6
Tool/technique/etc.	Marks that what is presented is a tool, technique, algorithm, approach, framework, process, model, method or similar.	

Establish categories for extraction

All coding categories were established beforehand; sub-categories will be discussed in the subsection 'Validate and adjust'. Some of these were established for the quality assessment, and several were established for the data extraction. The abstract and keywords were also extracted. An overview of the categories can be seen in Table 6.5.

The category 'NLP task' wasn't originally included; when the categories were established, it started out as 'NLP purpose', marking what purpose the NLP serves. Examples of categories were 'Ambiguity detection', 'Classification', 'Requirements extraction' and 'Semantic preprocessing'. However, during the data extraction, it became clear that subcategories for this category were ambiguous, and differences in coding were hard to resolve. For example, if Named Entity Recognition is used to extract a company's name, is NER in that case semantic analysis or semantic preprocessing? Thus, it was chosen to solely code the NLP tasks referred to in the articles.

Extract data

When the categories were established, the coding could begin. This meant marking text and coding, e.g. dragging, it to the predetermined categories. Though also a long process, it didn't take as long as the study selection. Meanwhile, articles that failed to meet any of the quality criteria were removed from the list in NVivo. This further cut down the number of articles from 275 to 144. These 144 articles are what the final article list consists of, answering RQ4. The full list is available in Appendix C; please refer to Chapter 7 for more details on the list. Moreover, the list of articles removed from scope during the quality assessment is featured in Appendix D.

Validate and adjust (iterative coding)

This sub-phase has already been discussed in Chapter 2, but as a reminder, it involved validating the categories, creating sub-categories, coding the text in the super-categories to the different sub-categories, validating the sub-categories and adjusting them where needed. This process was done for two categories: the purpose of the tool and the NLP task. For the RE phase, no iterative coding took place as the phases and sub-phases were determined beforehand. Therefore, no articles fell outside of this scope, as these had already been removed during the study selection.

This phase was mainly relevant to the NLP tasks. That is because there are quite numerous different NLP tasks used; therefore, some aggregation was involved to reduce this number. Moreover, some tasks that seemed different, turned out to be the same. An example is that one article used the term `shallow semantic parsing'. After looking up what it is, it turned out that it's a synonym for Semantic Role Labeling, and the article was coded to that category. Instances like this led to a reduction of the NLP tasks from 64 to 24, which is the final list of NLP tasks in this research.
Less cycles were involved in determining the sub-categories for the Tool Purpose. Most of the tool purpose categories are more or less congruent with the RE phase categories; however, a distinction is still made between the RE phase and the tool purpose as they are not always the same. An explanation for this phenomenon is provided in Chapter 8.
When a consensus was eventually reached, the final (sub-)categories were established from the data extraction, as well as a matrix of which article was coded to which categories. This answered RQ5 and served as an input to RQ6. Though the textual data is still available, it is unfortunately not further used in this research other than creating word clouds per (sub-)category. We looked into clustering the text, but it made too little sense for the results and proved too difficult. However, the data is still available and could also be used as a training set for automated classification.

6.3.4 Data synthesis

While RQ4 and RQ5 were answered at this point, they also served as the input for the final research question. In this phase, some simple analysis was used to show some interesting facts about the dataset. Moreover, crosstabs and Jaccard Indices were calculated to see any patterns of co-occurrence (RQ6).

Descriptive analysis

Descriptive analysis involves getting descriptive facts about the data set and presenting them in some way. Some of these statistics have been included in Chapter 7 and Chapter 8, where they enrich some of the information shown. These statistics were calculated using Google Sheets. Google Sheets was also used to create some of the charts in those chapters.

Quantitative analysis

Quantitative analysis means using statistical tests to see if one can uncover information in the data set. That means that one has to be familiar with the data set, the type of data and how it is distributed, and what exactly one wants to uncover. In this case, it was binary data: an article was either in a category (1) or not (0). Thus, a normal distribution is not the case and the data category is nominal. As the intent was to reveal correlations (thus, statistically significant combinations of categories), an applicable statistic had to be calculated. At first, the Chi2 value of several category combination matrices was calculated, but these results did not make sense as 0/0 was also included in the calculation, something that was to be avoided; thus, we had to look for an appropriate metric that involves 0/1, 1/0 and 1/1. This metric was found in the Jaccard Index (Jaccard, 1901).
The Jaccard index can be calculated of any crosstab with 4 cells. A fictitious example is shown in Figure 6.3. Two questions are posed to 50 people: 'Do you anticipate

rain today?' and 'Are you carrying an umbrella?' The respondents can only answer `Yes' or `No'. We want to see if there is any correlation between the responses to the two questions. The Jaccard Index is thus calculated by taking into account people that answered `Yes' to both questions and people that only answered `Yes' to one question or the other. The calculation is shown in the figure; J denotes the Jaccard Index. The result is a similarity coefficient: how similar the answers to the two questions are.

However, one might see that the score is lower because of the high number of peo-

	A	B	C	D	E
1			Are you carrying an umbrella?		
2			No	Yes	
3	Do you anticipate rain today?	No	6	2	
4		Yes	20	22	
5					
6			J = D4 / (D4 + D3 + C4) = 22/44 = 0.5		
7			R1 = D4 / (D4 + C4) = 22/42 = 0.5238...		
8			R2 = D4 / (D4 + D3) = 22/24 = 0.9166...		

FIGURE 6.3: A fictitious example of the Jaccard Index (J) and ratios (R1 and R2) described in this chapter.

ple not carrying an umbrella. Therefore, ratios have also been calculated for the NLP tasks. In this particular case, it is calculated by dividing 22 by 22+2, which gives a score of .9166, much higher than the 0.5 from the Jaccard Index. Calculation of the ratios is also shown in Figure 6.3; the example just discussed is R2, while the other ratio, so the ratio between the people that answered `Yes' to both questions and the ones that only answered yes to 'Do you anticipate rain today?' would be R1.

Applied to the research, first crosstab matrices were created to determine how frequently an NLP task occurred with another NLP task, a certain RE phase or a certain tool purpose. One was also created for how often a Tool Purpose occurred with a certain RE phase. This was done by taking the binary list of articles and what category they appear in, creating a pivot table (in Microsoft Excel) and copying these individual pivot tables into a larger matrix. Next, the Jaccard Indices were calculated, but the ratios were only calculated for the NLP tasks, as they would make no sense for Tool Purposes or RE Phases. Per Jaccard matrix, the distribution of both these scores was checked (all were normally distributed), and then the mean (μ) was calculated over the entire table, over every row and over every column. Moreover, the standard deviation (σ) was calculated for these as well. Lastly, the scores were checked if they were statistically significant for their row, column or overall. This was done by checking if the score was greater than or the same as $\mu + 2 * \sigma$. The values were formatted differently based on their significance: **bold** if it was significant for the row, underlined if it was significant for the column, and the cell was marked **green** if it was significant overall. Of course, combinations of these could occur. The results of this sub-phase are presented in Chapter 9.

6.3.5 Review

Reviewing the conducting phase was done usually during and after each sub-phase was conducted. Therefore, the review of the entire conducting phase was not necessary, and therefore not explicitly performed. However, we would like to note that because of weekly meetings and overall good communication between the student and the first supervisor, few large adjustments had to be made. For discussion of potential limitations of the study and threats to validity, please refer to Chapter 10.

6.4 Reporting phase

6.4.1 Report

This sub-phase involves reporting on the entire process of the SLR. The output of this sub-phase is the thesis that you're currently reading. Notable is that a first version of Chapters 1 through 5, albeit with some differences in scope and wording, were written before the conducting phase was started as a research proposal.

6.4.2 Review and adjust

After the first version of the thesis was handed in and a presentation was given on the research to other students and staff of the Master Business Informatics (the study of the student), the thesis was reviewed and adjusted. The review was done by the student, dr. Fabiano Dalpiaz and dr. Marcela Ruiz Carmona, while the adjustments were made by the student. A final presentation on the entire research project was given as a thesis defense.

Chapter 7

List of Articles using NLP

RQ4: *What tools and tool-supported approaches within requirements elicitation and requirements analysis employ NLP?*

After performing the study selection and quality assessment sub-phases of the SLR, 144 papers were still included. The full list can be seen in Appendix C. From all these articles, further data was extracted to answer RQ5 and RQ6. The rest of this chapter will feature examples of articles in the list per phase, as well as some descriptive statistics of the data set in general.

7.1 Examples of articles per phase

Elicitation

An example of the intersection of NLP and requirements elicitation is the work by Peng et al. (2017). In this study, the researchers collected reviews of mobile applications, and used classification techniques in combination with NLP pre-processing to identify reviews with feature requests. The reviews are then clustered, and syntactic parsing is used to extract feature requests from these clustered reviews. These feature requests can then be fed back to the developers, whom can decide to act upon them or not.

Documentation

An example of requirements documentation is the work by Daramola, Sindre, and Moser (2012), which supports this process via ontologies and boilerplates. An ontology can best be described as a knowledge base; it gives a set of concepts relevant to the domain, and terms and knowledge related to these concepts. A boilerplate is a template: for example, boilerplate code is code in programming that is used throughout the software with few alterations. One could see the user story template (discussed in Chapter 4) as a boilerplate. The article combines these to support the user of the tool with writing requirements according to a predefined format that the user can choose. This also uses rule-based analysis, on which we will go into further detail in the next chapter.

Analysis

The work of Jurkiewicz and Nawrocki (2015) reports on the analysis of use cases. As a reminder, use cases are often made based on the elicited requirements. These use cases encompass events that happen in the interaction between user and system. The automated identification of these events is what the researchers try to achieve.

They do this by using a combination of sentence splitting, lemmatization, syntactic parsing, Part-of-Speech tagging, and rule-based analysis.

Modelling

The work by Lucassen et al. (2017) is an example of requirements modelling, as a model is derived from the requirements. The researchers visualize requirements' relations to each other. This is, again, done by using clustering techniques. For this, the semantic similarity score between words in the set of requirements is calculated, and the requirements are clustered based on this score. These clusters are then shown as a visualization of the inter-relatedness of the requirements.

Classification

Ott (2013) presents a classification approach for Software Requirements Specifications at Mercedes-Benz. This classification also helps in extracting requirements from relevant documents. As Mercedes-Benz is a German car manifacturer, the documents are first pre-processed using 2 different approaches: one is k-gram indexing (splitting the word in different groups of k consecutive letters), while the other is word segmentation (splitting compound words in separate words). Classification based on k-gram indexing worked best.

Verification

One of the facets of requirements is checking for requirements completeness. One of the approaches supporting this is the work by Ferrari et al. (2014b). First, they identify 2 metrics that measure the completeness of an SRS. They then implement these metrics in a tool that can improve it. The NLP tasks used are Part-of-Speech tagging and stop word removal; the text is then classified and the completeness is improved based on the classified requirements.

Prioritization

One of only 2 articles that use NLP in requirements prioritization is Duan et al. (2009). This article attempts to attain automated requirements prioritization using clustering techniques. As discussed in Chapter 5, clustering is grouping articles without any prior data, usually by measuring word similarity. The NLP techniques involved in this article are stop word removal and stemming, which will both be discussed in Chapter 8.

7.2 Descriptive statistics

A distribution of articles over the years can be seen in Figure 7.1. This chart shows that there is a positive trend in usage of NLP within requirements elicitation and analysis, especially after 2012. Important to note is that the article search was conducted about halfway 2018, and therefore, the number of articles in 2018 is slightly lower [1]. However, since there are also years that did not have any articles on NLP in requirements elicitation or analysis (1994, 1998, 2001 and 2002), the upwards trend

[1] At the time of the study search, the 26th edition of the IEEE International Requirements Engineering Conference (2018), one of the major sources for papers in the scope, had not yet taken place

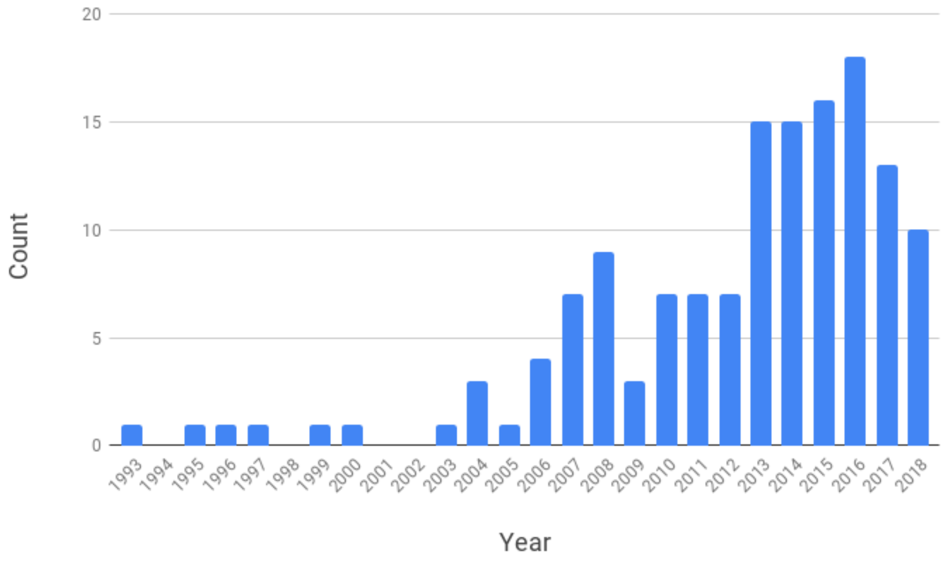

FIGURE 7.1: Number of articles published per year.

is not to be denied.

Figure 7.2 shows the distribution of the phases wherein the articles are applicable,

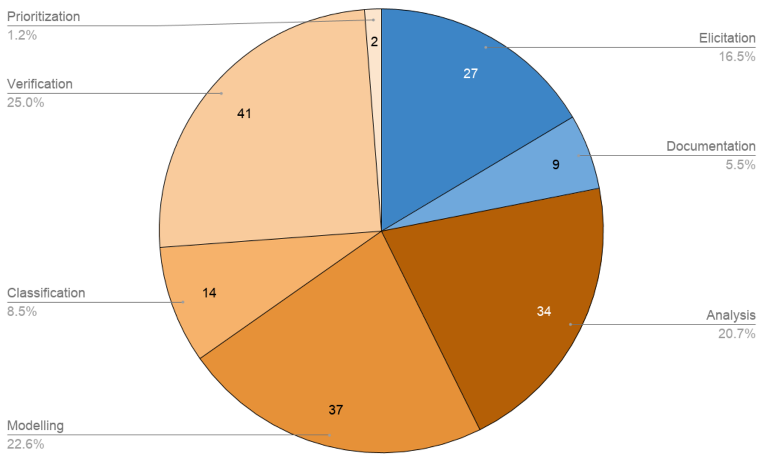

FIGURE 7.2: Distribution of the RE phases of the articles.

as based on Figure 6.1. The phases have been color-coded, to show the phases and their sub-phases' distribution. Taking into account the percentages also shown, it can be seen that far more articles fall into the scope of requirements analysis than requirements elicitation: 78 percent compared to 22 percent. From this, we can conclude that NLP within our scope is mostly centered around requirements analysis tasks. The largest sub-phase is requirements verification with exactly 25% of the assigned categories belonging to it. During the research, this became clear early on; there is a plethora of tools available for refining the requirements further, something that will also be illustrated in the next chapter. Another large sub-phase is modelling, with 22.6%.

Important to note is that if all scores of the phases (within the pie chart) are added up, one might notice that this adds up to 164, which is larger than the total number of articles at 144. That is because numerous articles are coded to multiple phases or sub-phases. This is done because the tool presented is used for overlapping phases. This was a deliberate choice: If one article could only be coded to one phase, one would have to assign priority to what phase is more important. This would only introduce more subjectivity into the study. Moreover, information could be lost if such a policy were to be enforced. Therefore, all three categories can overlap (though admittedly, this makes the most sense in the 'NLP task' category).

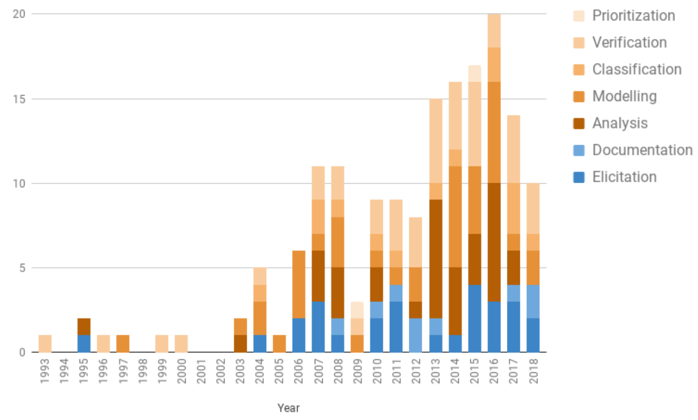

FIGURE 7.3: RE phases, stacked per year to absolute numbers.

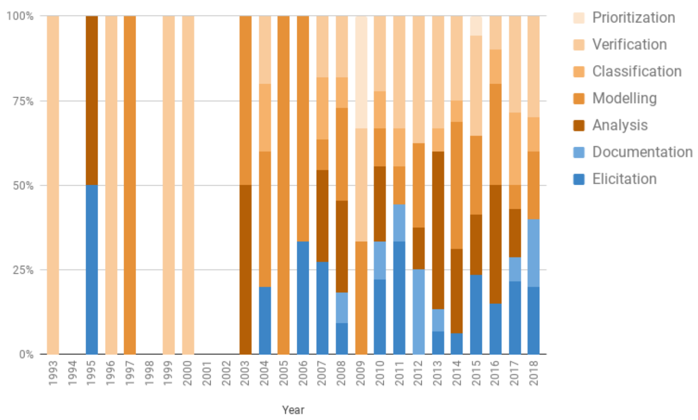

FIGURE 7.4: RE phases, stacked per year to percentages of the total of that year.

The results of combining the previous two graphs can be seen in 7.3 for absolute numbers, and to 7.4 for percentages of the totals of that year. In figure 7.3, one can see that the absolute numbers of requirements elicitation tools have remained around the same number for the last 12 years, while the number of requirements analysis tools vary. This is also illustrated in 7.4, where there is no stable trend visible in the ratio requirements elicitation to requirements analysis over the last years. Figure 7.5 shows the distribution of libraries that the articles in the SLR are sourced

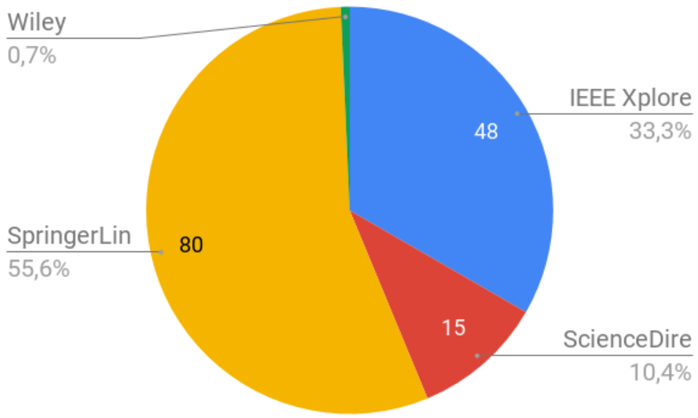

FIGURE 7.5: Distribution of libraries that the articles in the SLR are sourced from.

from. The most prevalent is SpringerLink, with over half of the articles; however, the search of Wiley resulted in only one paper: Ben Abdessalem Karaa et al. (2015).

In Figure 7.6, one can see what type of venue the papers are sourced from. By far

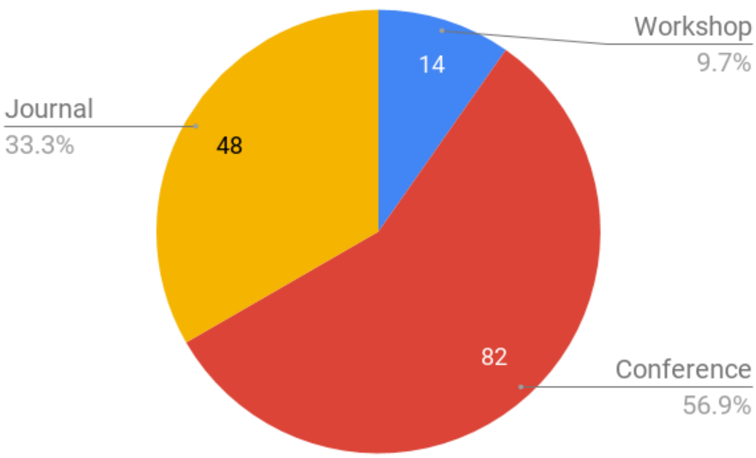

FIGURE 7.6: Distributon of paper type: conference paper, journal article or workshop paper.

the most prevalent type is the paper that comes from proceedings of conferences; the ‘International Requirements Engineering Conference’ (RE) [2] and the ‘International Working Conference on Requirements Engineering: Foundations for Software Quality’ (REFSQ) [3] were the most frequently occurring conferences, with scores of 13 and 10 respectively. If the distribution is stacked per year, like in Figure 7.7, one can see a trend: the papers are less published in conference proceedings, and more in journals. This is an indication that NLP within requirements elicitation and analysis is maturing. We hope to see the trend of increasing publishing of NLP-related articles

[2]http://requirements-engineering.org/
[3]https://refsq.org/2019/welcome/

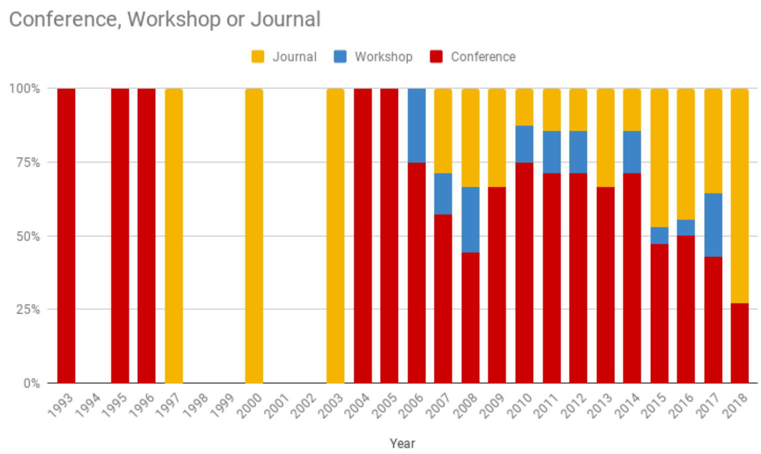

FIGURE 7.7: Distribution of type of papers, stacked per year to ratios.

in RE-related journals continuing over the coming years.

Chapter 8

A categorization of NLP within requirements elicitation and analysis

RQ5: *How can Natural Language Processing tasks within requirements elicitation and requirements analysis be categorized?*

Through the process of iterative coding, numerous categories for NLP within the domain of this research have been established. These are defined and exemplified in this chapter. Moreover, the same is done for the RE phases that the tools employing NLP can be categorized in, as well as the main purposes of those tools. Full lists of which article features which RE (sub-)phase(s), tool purpose(s) and NLP task(s) can be found in Appendices F, G and H respectively.
Important to note is the discrepancy between the main tool purpose and the RE phase. Though it mostly overlaps, there are some cases where the phase of the tool is not the same as the purpose of the tool. This can be because of multiple main tool purposes or phases, or that the tool purpose is better applied in another phase. An example is Duan et al. (2015): though the purpose of the tool is clustering, which falls under the category PT_01 - Classification, the end goal of this clustering is elicitation, and such the RE phase that the article is coded under is requirements elicitation.

8.1 RE Phase

This section features a formal definition of the RE phases discussed in Chapters 2 and 3. These categories were not established by iterative coding, but were established by the literature study beforehand. To see how these phases are connected, please refer to Figure 6.1.

8.1.1 RP_1 - Requirements elicitation

Definition: The requirements engineering phase wherein requirements for the software system are elicited from stakeholders and other sources.
Number of articles: 27
Examples: The classification of non-functional requirements by Cleland-Huang et al. (2007b); the approach for eliciting missing requirements by Gorse et al. (2004); the ontology-supported method for elicitation byShibaoka, Kaiya, and Saeki (2007).

8.1.2 RP_1.A - Requirements documentation

Definition: A sub-phase of requirements elicitation wherein the Software Requirements Specification (SRS) is written.
Number of articles: 9
Examples: The use of ontologies and boilerplates for writing an SRS by Daramola, Sindre, and Moser (2012); the approach for rapid quality checks of an SRS ("smells") by Femmer et al. (2017); the approach for collaborative requirements documentation Sateli, Angius, and Witte (2013a).

8.1.3 RP_2 - Requirements analysis

Definition: The analysis of stakeholders' needs for a software system, and further processing these needs so that they can be used for a final list of requirements for the software system.
Number of articles: 34
Examples: The use of machine learning to analyze requirements by Garzoli et al. (2013); an approach for the estimation of user stories complexity using Bayesian Networks by López-Martínez et al. (2017); an approach for quality assurance during software development by Ninaus et al. (2014).

8.1.4 RP_2.A - Requirements modelling

Definition: A Sub-phase of requirements analysis wherein models are created from the requirements in the SRS. This sub-phase also includes formalization of the requirements.
Number of articles: 37
Examples: The generation of UML class diagrams from requirements by Alkhader, Hudaib, and Hammo (2006); the extraction of conceptual models from user stories by Lucassen et al. (2017); the generation of activity and sequence diagrams from requirements by Sharma, Gulia, and Biswas (2014).

8.1.5 RP_2.B - Requirements classification

Definition: A sub-phase of requirements analysis wherein requirements are classified according to a predefined scheme using automated techniques. This sub-phase also includes requirements clustering and requirements mining.
Number of articles: 14
Examples: The classification of requirements in a Controlled Natural Language (CNL) by Minhas et al. (2011); the mining of incoherent requirements in an SRS by Saint-Dizier (2018); the classification of requirements based on neural networks by Winkler and Vogelsang (2017).

8.1.6 RP_2.C - Requirements verification

Definition: A sub-phase of requirements analysis wherein requirements are validated and improved based on their completeness, correctness, ambiguity and overall quality.
Number of articles: 41
Examples: Interaction of tool and user to refine requirements by Boddu et al. (2004); a tool that gives sources of ambiguity in an SRS by Gleich, Creighton, and Kof (2010);

a method for the verification of non-functional requirements based on the Requirements Frame model by Matsumoto, Shirai, and Ohnishi (2017).

8.1.7 RP_2.D - Requirements prioritization

Definition: A sub-phase of requirements analysis wherein requirements are ranked based on their priority.
Number of articles: 2
Examples: An approach for the automation of requirements prioritization and triage by Duan et al. (2009); a tool-supported method using NLP and satisfiability modulo theories solvers for prioritization by McZara et al. (2015).

8.2 Purpose of tool

This section defines the purposes of the tools presented in the articles. These were formed during the iterative coding process. A tool can have multiple purposes, and as such, there is some overlap. Moreover, this category largely overlaps with the RE phase; however, as discussed in the introduction of this chapter, there are some differences.

8.2.1 PT_1 - Classification

Definition: Tools that use automated techniques to classify input according to a predefined scheme. Also includes clustering: classifying input without using a predefined scheme.
Number of articles: 25
Examples: The use of clustering to increase automation of identifying early aspects by Duan et al. (2015); an approach for classifying user reviews in mobile app stores by Jha and Mahmoud (2018); the extraction of feature requests from app reviews using a classification algorithm by Peng et al. (2017).

8.2.2 PT_2 - Elicitation of requirements

Definition: Tools that support elicitation or documentation of requirements.
Number of articles: 30
Examples: The use of topic modelling for generating creative requirements by Bhowmik et al. (2015); a tool-supported method for structuring an SRS by Kiyavitskaya and Zannone (2008); a method for identifying requirements in an SRS with the quality characteristics of the ISE/IEC 9126-1 guideline by Terawaki (2011).

8.2.3 PT_3 - Model analysis

Definition: Tools that support the analysis of models, so that requirements can be extracted or improved.
Number of articles: 20
Examples: An approach for identifying ambiguity in user stories using semantic similarity by Dalpiaz, van der Schalk, and Lucassen (2018); the automated identification of events in use cases by Jurkiewicz and Nawrocki (2015); the automated identification of transactions in use cases by Ochodek and Nawrocki (2008).

8.2.4 PT_4 - Model extraction or generation

Definition: Tools that support the extraction of models from other models, or the generation of models from an SRS.
Number of articles: 30
Examples: The generation of UML models from an SRS by Deeptimahanti and Sanyal (2009); a tool that supports drawing UML diagrams corresponding with an SRS by Gulia and Choudhury (2016); an approach that identifies instances of commonality and variability in product lines in order to create feature models by Loughran, Sampaio, and Rashid (2006).

8.2.5 PT_5 - Prioritization of requirements

Definition: Tools that support prioritizing requirements.
Number of articles: 2
Examples: An approach for the automation of requirements prioritization and triage by Duan et al. (2009); a tool-supported method using NLP and satisfiability modulo theories solvers for prioritization by McZara et al. (2015).

8.2.6 PT_6 - Quality analysis and/or improvement

Definition: Tools that support analyzing and/or improving various quality aspects of requirements.
Number of articles: 30
Examples: An approach for quality analysis of requirements based on a reusable domain model Annervaz et al. (2013); a framework that extracts low-level quality indicators (e.g., number of ambiguous terms, overlapping requirements) from an SRS by Génova et al. (2013); the usage of ontologies to discover ambiguous, faulty or inaccurate parts of an SRS by Körner and Brumm (2009).

8.2.7 PT_6.A - Ambiguity detection and/or resolution

Definition: Tools that support detecting and/or resolving ambiguity in textual requirements.
Number of articles: 27
Examples: A framework that identifies and extracts uncertainty cues by Al-Sabbagh, Girju, and Diesner (2015); an approach for detecting vague words in a translated SRS by Cruz et al. (2017a); the identification of term-aliasing (multiple terms referring to one same entity) in an SRS by Misra and Das (2013).

8.2.8 PT_6.B - Change impact or dependency analysis

Definition: Tools that support change impact analysis or dependency analysis between requirements.
Number of articles: 5
Examples: Analysis of change impact between requirements by Arora et al. (2015a); a tool that analyzes requirements and recommends pairs as being interdependent by Ninaus et al. (2014); a method for identifying a type of dependency between agents (goal dependency, task dependency, soft-goal dependency or resource dependency) from requirements by Soni and Gaur (2016).

8.2.9 PT_6.C - Completeness or conformance checking

Definition: Tools that support requirements completeness checking, or their conformance to a certain template.
Number of articles: 13
Examples: A tool for checking if requirements are conformant to boilerplates by Arora et al. (2013); an approach for measuring and improving the completeness on an SRS by Ferrari et al. (2014b); the use of ontologies for checking requirements on completeness, consistency and ambiguity by Stålhane and Wien (2014).

8.3 NLP tasks

This section describes the different NLP tasks that are executed by the tools presented by the articles. Classification and clustering, though not NLP tasks in the strictest sense, have also been included because of the high number of articles that use these techniques. However, they have only been included if they were used in conjunction with other NLP tasks; often, this was in the form of pre-processing tasks like a linguistic filter or lemmatization. Some tasks aggregate multiple tasks; this will be explicitly stated in the definition.

8.3.1 NT_01 - Anaphora detection and/or resolution

Definition: The detection and/or resolution of anaphora: uses of pronouns to refer to an antecedent. For example: in the user story 'as a customer, I want to chat with customer service so that I can ask them questions with fast feedback', `them' is the pronoun and `customer service' the antecedent. The pronoun is replaced by the antecedent if the anaphora is resolved.
Number of articles: 9
Examples: The merging of references in new sentences with previous sentences by Harmain and Gaizauskas (2003); the resolution of anaphora in an SRS by Park et al. (2000); the identification of pronouns and replacement with noun phrases by Sinha et al. (2010a).

8.3.2 NT_02 - Chunking

Definition: Separating a sentence into parts that have a discrete grammatical meaning. Examples of these meanings could be the noun phrase ('the developers') or verb group ('have to develop').
Number of articles: 14
Examples: The usage of chunking to identfy noun and verb phrases by Mala and Uma (2006); identifying noun phrases and verb phrases using chunking by Arora et al. (2015a); chunking as a pre-processing step for classification by Mezghani, Kang, and Sèdes (2018).

8.3.3 NT_03 - Classification

Definition: Automated categorization of entities based on a predefined scheme.
Number of articles: 29
Examples: Classifying usability and user experience issues by Bakiu and Guzman

(2017); automated classification of requirements in app stores by Deocadez, Harrison, and Rodriguez (2017); classification of keywords based on semantic similarity by Ko et al. (2007b).

8.3.4 NT_04 - Clustering

Definition: Automated creation of similar categories for entities (clusters), without using a predefined scheme.
Number of articles: 16
Examples: Clustering requirements based on patterns for the identification of user preferences by Belsis, Koutoumanos, and Sgouropoulou (2014); clustering main verbs of use case scenarios by Ko, Kim, and Park (2018); clustering requirements around themes in order to improve user cohesion of the requirements by Misra, Sengupta, and Podder (2016).

8.3.5 NT_05 - Discourse analysis

Definition: Umbrella term for NLP-assisted analysis of discourse between two parties.
Number of articles: 1
Examples: The interpretation of separate sentences to represent their meaning by Harmain and Gaizauskas (2003).

8.3.6 NT_06 - Entity recognition and/or extraction

Definition: Recognition and/or recognition of specific linguistic entities, for example subjects, objects or verbs.
Number of articles: 21
Examples: The extraction of potential glossary terms from requirements by Dwarakanath, Ramnani, and Sengupta (2013a); the extraction from nouns, noun phrases and verbs by Ibrahim and Ahmad (2010b); the identification of nouns, noun phrases, verbs, verb phrases, adverbs etc. by Liu, Li, and Kou (2014).

8.3.7 NT_07 - Lemmatization

Definition: Reducing a word to its basic form: the lemma. For example, `gets', `got' and `getting' are all forms of *get*.
Number of articles: 14
Examples: The usage for lemmatization as word form reduction for a Requirements Description Language by Asad Naqvi et al. (2010); usage of lemmatization to improve a classification algorithm's recall score by Pinquié et al. (2016); lemmatization as pre-processing for identifying duplicate functionality in use cases by Rago, Marcos, and Diaz-Pace (2016b).

8.3.8 NT_08 - Linguistic filter

Definition: Removing specific categories of words from a text; a common form of a linguistic filter is stop word removal.
Number of articles: 36
Examples: Removal of common words as pre-processing for prioritization by Duan et al. (2015); stop word removal as pre-processing for classification by Cleland-Huang

et al. (2007b); stop word removal as a pre-processing for automated conceptual model extraction by Robeer et al. (2016a).

8.3.9 NT_09 - Morphological analysis

Definition: Analyzing words based on their morphemes: atomic parts of words that cannot be split further. Also includes morphological parsing.
Number of articles: 15
Examples: Analysis of morphemes as a part of a tool for conceptual modelling by Kiyavitskaya et al. (2004); morphological analysis as part of semantic analysis of an SRS by Rashwan (2012); usage of morphological analysis to identify the singular form of nouns and verbs by Umber, Bajwa, and Naeem (2011).

8.3.10 NT_10 - Named Entity Recognition (NER)

Definition: Recognition of named entities to belong to a certain category. For example, a Named Entity Recognition module would recognize ‘Google’ as a brand or company, depending on the categorization.
Number of articles: 10
Examples: NRE as pre-processing for machine learning-based requirements analysis by Garzoli et al. (2013); NRE in a chain of NLP tasks that serve as AI-based requirements improvement by Körner, Landhäußer, and Tichy (2014); NER as pre-processing for automated summarization in Subha and Palaniswami (2013).

8.3.11 NT_11 - Part-of-Speech (PoS) tagging

Definition: The tagging of words with their respective syntactic roles in a sentence.
Number of articles: 63
Examples: PoS tagging as pre-processing for transforming requirements to sequence diagrams by Díaz et al. (2005a); PoS tagging as pre-processing for the creation of conceptual models by Lucassen et al. (2017); PoS tagging as pre-processing for consistency analysis by Misra (2016).

8.3.12 NT_12 - Rule-based analysis

Definition: Analyzing a requirement based on a predefined set of rules or a pattern. Also includes rule-based mapping, rule-based extraction and pattern matching. Example: we have a regex analyzer <JJ><NN> | <NN><NN>. This will identify 2-word compounds with the first word being either an adjective (JJ) or a singular noun (NN) , and the second word being a singular noun. So from the requirement ‘the spreadsheet should have a green interface’, this regex analyer would return `green (JJ) interface (NN)’.
Number of articles: 49
Examples: Usage of rules to bind words to linguistic concepts by Biébow and Szulman (1993); extraction of requirements that match a certain pattern by Li et al. (2015); using patterns to detect incoherence in requirements by Saint-Dizier (2018).

8.3.13 NT_13 - Semantic Role Labeling (SRL)

Definition: Labeling words or parts of phrases to indicate their semantic role in a sentence. Also known as shallow semantic parsing. For example, in ‘The UI should have a color palette ranging from blue tot purple’, `a color palette ranging from blue

to purple' could in this case be labeled as a usability specification.
Number of articles: 11
Examples: Using the FrameNet project to semantically annotate words by Jha and Mahmoud (2017); Use of SRL to identify predicate-argument structures by Narouei and Takabi (2015); identifying predicates and arguments using SRL by Rago, Marcos, and Diaz-Pace (2016b).

8.3.14 NT_14 - Semantic similarity

Definition: Calculating how similar two words or (parts of) phrases are in a semantic way. For example, ǹon-functional requirement' and 'quality requirement' are expected to have a high semantic similarity score.
Number of articles: 22
Examples: Calculating semantic similarities using a dictionary by Asano, Hayashi, and Saeki (2017); using semantic similarity to link ontologies to each other by Bhat, Ye, and Jacobsen (2014); using semantic similarity scores to improve SRS completeness by Zachos et al. (2007).

8.3.15 NT_15 - Sentence splitting

Definition: Splitting a text into sentences.
Number of articles: 26
Examples: Splitting requirements into multiple sentences as pre-processing for ambiguity resolution by Cruz et al. (2017a); sentence splitting as pre-processing for requirements annotation by Hussain, Ormandjieva, and Kosseim (2012); sentence splitting as pre-processing for morphological analysis by Terawaki (2011).

8.3.16 NT_16 - Sentiment analysis

Definition: Analysis of the sentiment that a sentence or text conveys. For example, "I dislike the new look of the interface" would be a negative sentiment.
Number of articles: 3
Examples: Analyzing and scoring sentiment of usability and user experience features by Bakiu and Guzman (2017); sentiment analysis of app reviews by Guzman and Maalej (2014a); sentiment analysis as pre-processing for automated app review classification by Maalej et al. (2016).

8.3.17 NT_17 - Spelling checking

Definition: Checking if the spelling of words in a certain text is correct.
Number of articles: 4
Examples: Checking loaded requirements on spelling errors by Alkhader, Hudaib, and Hammo (2006); assessing writing quality based on (among others) spelling checking bySateli, Angius, and Witte (2013a); replacing misspelled words with correct forms by Sillaber and Breu (2014).

8.3.18 NT_18 - Stemming

Definition: Reducing a word to a base form: the stem. However, as opposed to lemmatization, stemming operates without context; so if, for example, 'better' would be both stemmed and lemmatized, stemming would yield no output while lemmatization would return `good'.

Number of articles: 19
Examples: Stemming as pre-processing for analyzing use cases by Bolloju, Schneider, and Sugumaran (2012); stemming words to create keywords of a requirement by Ninaus et al. (2014); stemming as pre-processing for predicate generation by Veerappa and Harrison (2013).

8.3.19 NT_19 - Syntactic parsing

Definition: Recognizing a sentence or text, and assigning a syntactic structure to it. This task includes both dependency-based and constituency-based parse trees.
Number of articles: 52
Examples: Constituent parse tree as pre-processing for the identification of candidate services by Bhat, Ye, and Jacobsen (2014); creating a dependency-based parse tree to uncover dependencies between words in sentences by Biébow and Szulman (1993); using syntactic parsing as a way to evaluate well-formedness of a user story by Lucassen et al. (2015a).

8.3.20 NT_20 - Automated summarization

Definition: Automatically creating a summary from a text.
Number of articles: 1
Examples: Automatic summarization of the requirements in an SRS by Subha and Palaniswami (2013).

8.3.21 NT_21 - Tokenization

Definition: Splitting a text into tokens: individual units that are usually separate words.
Number of articles: 35
Examples: Tokenization of sentences as pre-processing for the automatic recommendation of study nodes for HAZOP (Hazard and Operability) analysis by Daramola et al. (2011); tokenization as pre-processing for the tagging of speech acts by Morales-Ramirez, Perini, and Ceccato (2015); tokenization of the words in a use case as pre-processing for the identification of transactions by Ochodek and Nawrocki (2008).

8.3.22 NT_22 - Topic modelling

Definition: Creating a model of related words or concepts about a certain word, sentence or text.
Number of articles: 3
Examples: Creating a topic model in support of creative requirements authoring by Bhowmik et al. (2015); topic modelling as an improvement of sentiment analysis by Guzman and Maalej (2014a); topic modelling to facilitate grouping of requirements candidates by Li et al. (2015).

8.3.23 NT_23 - Word sense disambiguation

Definition: Linking a word to its context in order to eliminate ambiguity. For example, ‘diagram’ has multiple possible implementations, but adding `sequence’ or `activity’ makes the meaning clear.
Number of articles: 2
Examples: A Word sense disambiguation module in order to support requirements

identification by Garzoli et al. (2013); word sense disambiguation supporting service discovery by Zachos et al. (2007).

8.3.24 NT_24 - Word segmentation

Definition: Splitting compound words into multiple words.
Number of articles: 4
Examples: Word segmentation of Chinese words by Lili et al. (2010); word segmentation of German compound words in order to improve classification by Ott (2013); Word segmentation of Chinese words by Sun and Peng (2015).

8.4 Descriptive statistics

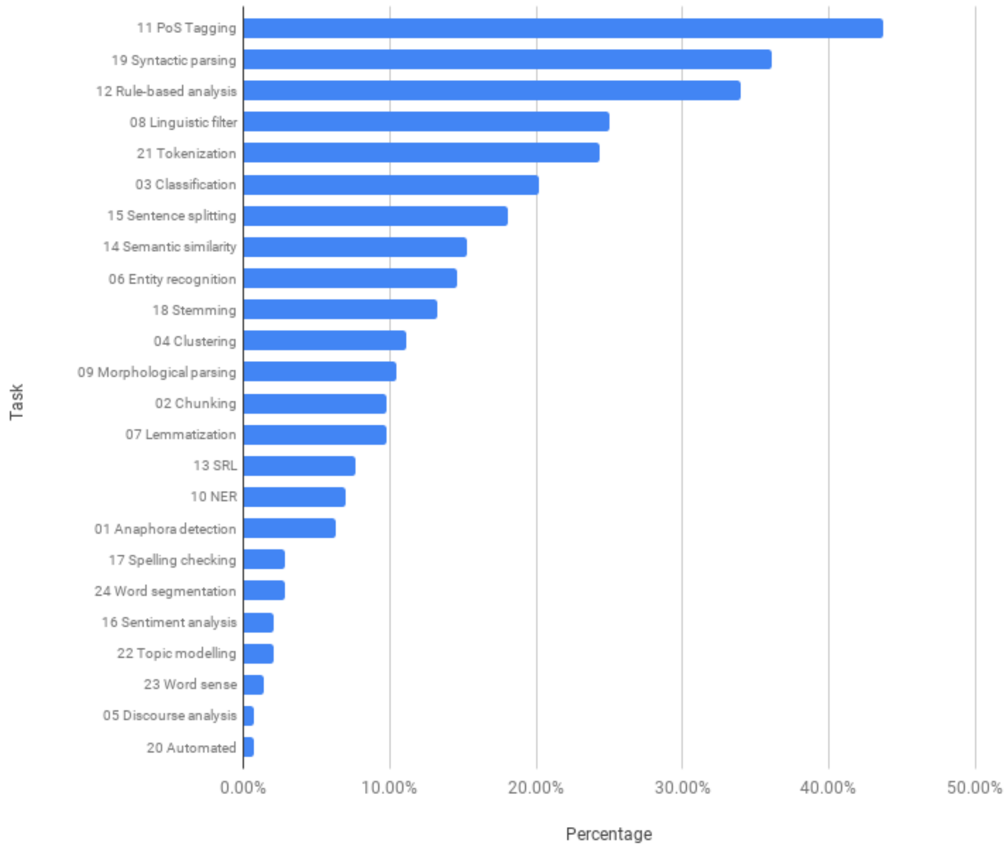

FIGURE 8.1: Likelihood per NLP task to occur.

In Figure 8.1, one can see how likely a task is to occur, in percentages. This is deliberately shown as a stacked chart instead of a pie chart, as it often occurs that multiple tasks are used within the same article. The tasks with the highest likelihood are:

1. NT_11 - Part-of-Speech tagging, occurring in 63 or 43.75% of the articles;
2. NT_19 - Syntactic parsing, occurring in 52 or 36.11% of the articles;
3. NT_12 - Rule-based analysis, occurring in 49 or 34.03% of the articles;

4. NT_08 - Linguistic filter, occurring in 36 or 25% of the articles;

5. NT_21 - Tokenization, occurring in 35 or 24.31% of the articles.

The least-used tasks are NT_05 - discourse analysis and NT_20 - automated summarization, both in 1 article (or 0.69%).
The distribution of the tool purposes can be seen in Figure 8.2. Moreover, the distri-

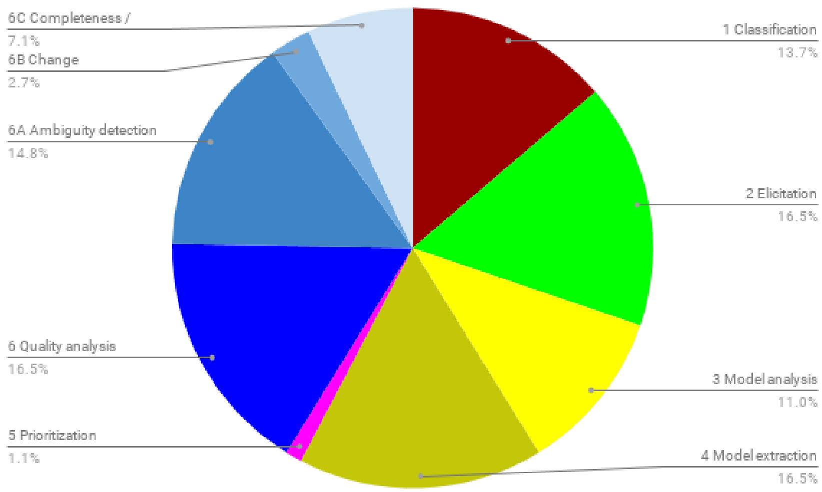

FIGURE 8.2: Distribution of tool purposes.

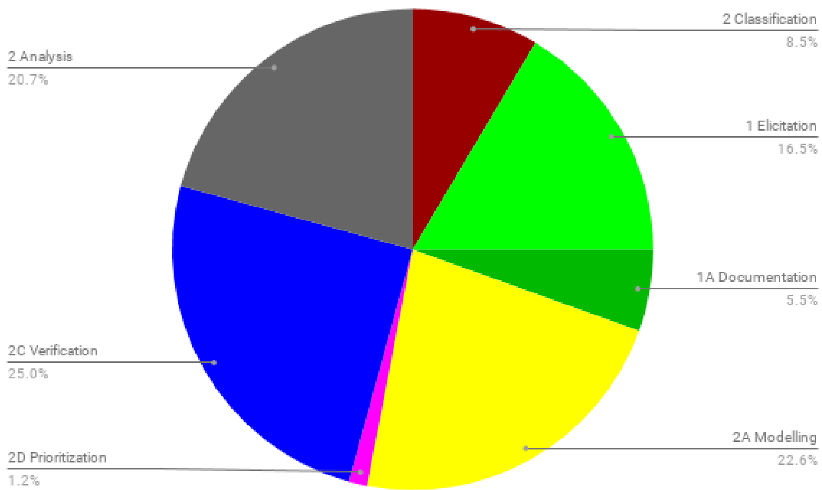

FIGURE 8.3: Distribution of RE phases, edited to better reflect the discrepancy with the tool purposes.

bution of the RE phases shown in Chapter 7 was edited to better reflect the overlap and discrepancy with similar tool purposes; see Figure 8.3. The discrepancy discussed in the intro of this Chapter is exemplified in the difference between these models. Not only are more tool purposes tagged than RE phases (182 vs. 164), categories like RP_1.A - requirements documentation, RP_2 - requirements analysis and PT_3 - model analysis are broad and subject to interpretation. This will also be discussed in Chapter 10.

What also illustrates this discrepancy is comparing the word clouds per coding category. A word cloud is a figure of most frequently used words: in this case, the most frequently used words in a coding category. The sizes of the words indicates their frequency: the larger the word, the more often it is used.

FIGURE 8.4: Word Cloud of Classification as an RE phase.

FIGURE 8.5: Word Cloud of Classification as a tool purpose.

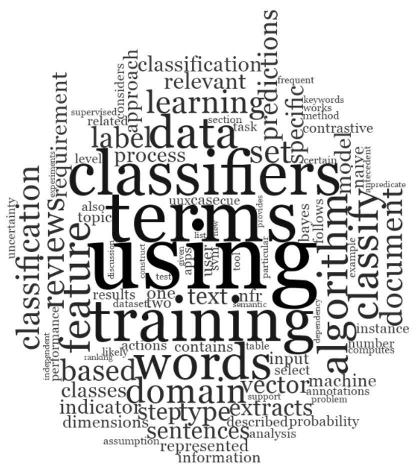

FIGURE 8.6: Word Cloud of Classification as an NLP task.

As an example, the word clouds of classification as a sub-phase of RE, as a tool purpose and as an NLP task have been featured in Figures 8.4, 8.5 and 8.6 respectively. These word clouds were created using the coding in NVivo; text in the coding category was stemmed, and the omni-present word 'requirements' was removed to better reflect other important words.
Figure 8.6 also more words that better describe how classification works, e.g. 'algorithm', 'terms', 'words' and 'learning'. Aside from the difference in number of words between Figure 8.4 and Figure 8.5, as the sub-phase is often described in less words, there are also differences in wording. Notably, the tool purpose word cloud uses words that would discuss what a tool would do, like 'propose', 'approach', 'paper' and 'extract'. The RE sub-phase word cloud is focused on more general words like 'detecting', 'clustering', 'mining' and 'classification'.

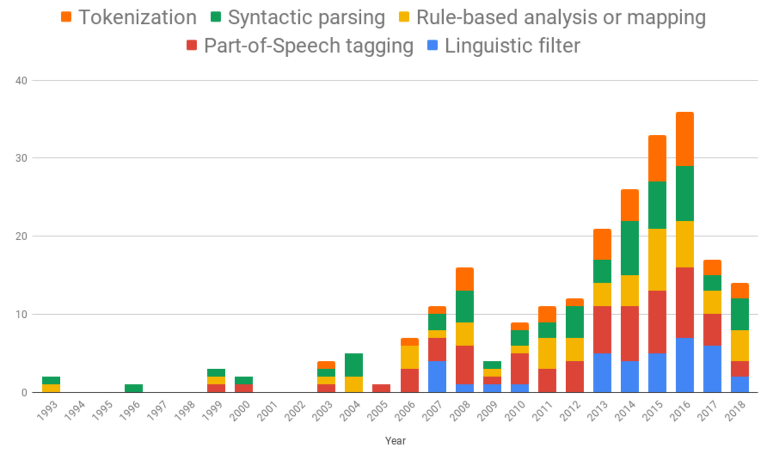

FIGURE 8.7: Distribution of the top 5 NLP tasks per year.

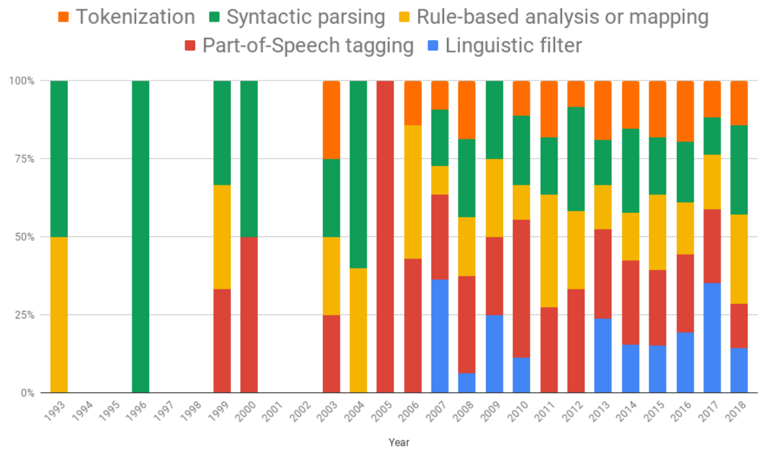

FIGURE 8.8: Distribution of the top 5 NLP tasks per year, stacked to percentages of the overall.

Figures 8.7 and 8.8 show the distribution of the aforementioned top 5 tasks per year, respectively in absolute numbers and as percentage of the total. What this especially shows is that the linguistic filter is the task most recently introduced, from 2007 onwards. There seems to be no line in the ratios between them, though two

major drop-offs of all tasks are visible in 2008 to 2009 and 2016 to 2017. The drop-off of 2008 to 2009 is consistent with the drop-off of the total number of articles as seen in Figure 7.1 shown in the previous chapter, the drop-off of 2017 is much larger relatively. This could be a coincidence, but it could also signal an increase in more complex NLP tasks being used, as these top 5 tasks are all relatively simple and established tasks.
To see if the increase of more complex tasks is indeed true, we want to discuss the NLP tasks on a more abstract scale. To do this, we have categorized the 24 tasks into 4 groups: Syntactic, Semantic and Pragmatic for true NLP tasks, as suggested by Cambria and White (2014), as well as a category 'other' for tasks that are related to NLP but not NLP tasks in the strictest sense. In Table 8.1, it is shown which tasks belong to which category.

TABLE 8.1: A categorization of the NLP tasks.

Syntactic	**Semantic**	**Pragmatic**	**Other**
Chunking	Entity recognition	Anaphora resolution	Classification
Linguistic Filter	Lemmatization	Automated summarization	Clustering
Morphological parsing	NER	Discourse analysis	Topic Modelling
PoS tagging	Semantic similarity		
Rule-based analysis	SRL		
Sentence splitting	Sentiment analysis		
Stemming	Spelling checking		
Syntactig parsing	Word sense disambiguation		
Tokenization			
Word segmentation			

Figure 8.9 shows the total distribution of the tasks within the categorization of Table 8.1. The Syntactic tasks category is by far the largest, with over 2 out of 3 occurring tasks belonging to this category. Moreover, the ratios per year can be seen in Figure 8.10. This figure shows that almost every year, syntactic tasks are, ratio-wise the largest. Though a slight decrease of its share can be observed in the figure, this is not only thanks to the semantic category, as that share remained relatively stable over the past years. There is no clear pattern as to which of the categories 'other' and 'semantic' are increasing; they mostly balance out each other, while pragmatic tasks are clearly staying behind. One would expect, based on Cambria and White (2014), that semantic tasks would be on the rise, but this figure contradicts this prediction. Perhaps in the coming years, semantic tasks will gain a larger share.

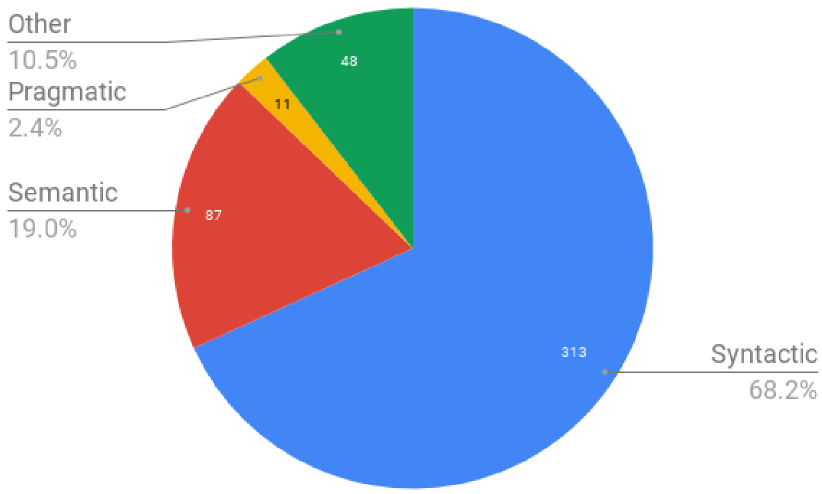

FIGURE 8.9: Distribution of the categorization of NLP tasks.

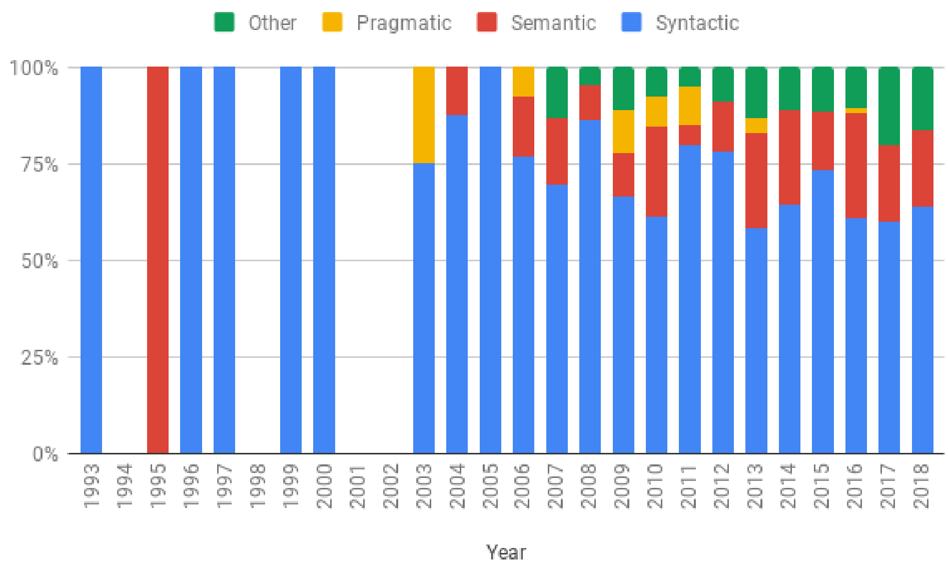

FIGURE 8.10: Distribution of the categorization of NLP tasks, stacked per year to percentages.

Chapter 9

Patterns of co-occurrence for NLP within requirements elicitation and analysis

RQ6: *What patterns of co-occurrence for NLP within requirements elicitation and requirements analysis can be uncovered?*

This chapter discusses patterns of co-occurrence that have been uncovered in the final data set. These patterns were calculated using Jaccard Indices and ratios, as discussed in Chapter 6. We will shortly discuss the patterns between the tool purposes and the (sub-)phases before discussing the patterns between NLP tasks themselves, the tool purposes and the RE (sub-)phases. Codes in the tables are abbreviated due to space and readability constraints; Table 9.1 features a quick reference.

TABLE 9.1: A quick reference for codes used in this chapter.

Code	Description	#	Code	Description	#
NT01	Anaphora detection / resolution	9	NT22	Topic modelling	3
NT02	Chunking	14	NT23	Word sense disambiguation	2
NT03	Classification	29	NT24	Word segmentation	4
NT04	Clustering	16			
NT05	Discourse analysis	1	RP1	Requirements elicitation	27
NT06	Entity recognition / extraction	21	RP1A	Requirements documentation	9
NT07	Lemmatization	14	RP2	Requirements analysis	34
NT08	Linguistic filter	36	RP2A	Requirements modelling	37
NT09	Morphological parsing	15	RP2B	Requirements classification	14
NT10	Named Entity Recognition	10	RP2C	Requirements verification	41
NT11	Part-of-Speech Tagging	63	RP2D	Requirements prioritization	2
NT12	Rule-based analysis	49			
NT13	Semantic Role Labeling	11	PT1	Classification	25
NT14	Semantic Similarity	22	PT2	Elicitation of requirements	30
NT15	Sentence splitting	26	PT3	Model analysis	20
NT16	Sentiment analysis	3	PT4	Model extraction / generation	30
NT17	Spelling checking	4	PT5	Prioritization of requirements	2
NT18	Stemming	19	PT6	Quality analysis / improvement	30
NT19	Syntactic parsing	52	PT6A	Ambiguity detection / resolution	27
NT20	Automated summarization	1	PT6B	Change impact / dependency analysis	5
NT21	Tokenization	35	PT6C	Completeness / conformance checking	13

TABLE 9.2: Jaccard Indices of the tool purposes intersected with the RE (sub-)phases.

	RP1	RP1A	RP2	RP2A	RP2B	RP2C	RP2D	μ	σ	Sig. $\geq$
PT1	0.1556	0.0303	0.1346	0.0164	**0.3929**	0.0476	0.0385	0.1165	0.1332	0.3830
PT2	**0.5000**	0.0833	0.1228	0.0635	0.1000	0.0290	0.0000	0.1284	0.1691	0.4665
PT3	0.0000	0.0357	0.1489	0.1875	0.0000	0.1296	0.0000	0.0717	0.0811	0.2339
PT4	0.0755	0.0263	0.0492	**0.5227**	0.0000	0.0597	0.0000	0.1048	0.1865	0.4778
PT5	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	**1.0000**	0.1429	0.3780	0.8988
PT6	0.0000	0.0541	0.1429	0.0152	0.0476	**0.3922**	0.0000	0.0931	0.1408	0.3747
PT6A	0.0000	0.0909	0.0893	0.0492	0.0250	**0.3878**	0.0000	0.0917	0.1358	0.3634
PT6B	0.0000	0.0000	**0.1143**	0.0244	0.0000	0.0222	0.0000	0.0230	0.0417	0.1065
PT6C	0.0256	0.1000	0.0444	0.0204	0.0000	0.2000	0.0000	0.0558	0.0722	0.2002
μ	0.0841	0.0467	0.0940	0.0999	0.0628	0.1409	0.1154	**μ_tot**		0.0920
σ	0.1647	0.0377	0.0520	0.1683	0.1283	0.1539	0.3320	**σ_tot**		0.1672
Sig. $\geq$	0.4134	0.1221	0.1980	0.4365	0.3194	0.4487	0.7793	**Sig._tot $\geq$**		0.4264

9.1 Tool purpose & RE (sub-)phase

Table 9.2 features an overview of the Jaccard Indices of the combination Tool Purpose (PT) and RE (sub-)phase (RP). As a reminder, **bold** values are significant in their respective row, underlined values are significant in their column, and cells marked **green** are significant overall ($p = 0.05$). Appendix E features an overview used for the codes that mark the tool purposes and RE (sub-)phases. Mean (μ), standard deviation (σ) and minimum significant score (Sig. $\geq$) are shown in the bottom rows for each row, in the rightmost columns for each column, and at the bottom right for the overall scores (also denoted by *_tot*).

The first observation that one can make is that only 3 combinations are significant overall: the combination PT_2 - Elicitation of requirements & RP_1 - Requirements elicitation, PT_4 - Model extraction/generation & RP_2.A - Requirements modelling, and PT_5 - Prioritization of requirements & RP_2.D - Requirements prioritization. One could argue that these three combinations are trivial, as they would make sense beforehand. However, in that case, one would also expect the combinations PT_1 - Classification & RP_2.B - Classification to be significant, as well as the combination PT_3 - Model analysis & RP_2.B - Requirements modelling. However, both these tool purposes are often used in other RE (sub-)phases, which can be seen if one looks at the rows of these tool purposes.

However, the data is also slightly distorted by the Jaccard Index of 1 between PT_5 - Prioritization of requirements & RP_2.D - Requirements prioritization. This `1' is caused by only 2 articles being about requirements prioritization as a phase, and both of these having the same tool purpose. If this `1' would not have been included, it is very likely that more values in the matrix would have been significant overall. This is also a reason that not only overall significance has been included, but significance of rows ans columns as well; it counteracts the distortion of the overall tables by extreme outliers. Other than the combinations that are significant overall, the only combination that is significant in both its row and its column is PT_1 - Classification & RP_2.B - Classification. Thus, all values that are significant in their rows are also significant in their columns.

There are also 3 values significant only in their row: PT_6.B - Change impact/ dependency analysis & RP_2 - Requirements analysis, PT_6 - Quality analysis and/or improvement & RP_2.C- Requirements verification, and PT_6.A - Ambiguity detection and/or resolution & RP_2.C - Requirements verification. This leaves two tool

purposes and one RE (sub-)phase that have no significant co-occurrences overall: PT_3 - Model analysis, PT_6.C - Completeness or conformance checking and RP_1.A - Requirements documentation.
What this matrix shows is the discrepancy between the tool purpose and the RE (sub-)phase, discussed at the start of Chapter 8. The fact that a tool has a certain purpose can not be automatically linked to a distinct phase. This can be seen by the fact that apart from PT_5 - Prioritization of requirements, all tool purpose and RE (sub-)phases have values in multiple cells in their respective rows/columns. Therefore, combining these two, at least in the context of NLP for RE, is unreliable as tool purposes can belong to multiple phases.

TABLE 9.3: Jaccard Indices of the NLP tasks intersected with the RE (sub-)phases.

	RP1	**RP1A**	**RP2**	**RP2A**	**RP2B**	**RP2C**	**RP2D**	μ	σ	**Sig.** $\geq$
NT01	0.0588	0.0588	0.1026	0.0698	0.0000	0.0417	0.0000	0.0474	0.0373	0.1219
NT02	0.0250	0.0455	0.0213	0.0851	0.0000	0.1458	0.0667	0.0556	0.0491	0.1537
NT03	0.1200	0.0270	0.1667	0.0313	0.1944	0.0938	0.0000	0.0904	0.0744	0.2393
NT04	0.1316	0.0000	0.0870	0.0392	0.1538	0.0556	0.0588	0.0751	0.0534	0.1819
NT05	0.0000	0.0000	0.0294	0.0270	0.0000	0.0000	0.0000	0.0081	0.0138	0.0356
NT06	0.1163	0.0345	0.1224	0.1600	0.0938	0.0333	0.0000	0.0800	0.0582	0.1965
NT07	0.0789	0.0455	0.1163	0.0408	0.0370	0.0784	0.0000	0.0567	0.0376	0.1320
NT08	0.1887	0.0000	0.1864	0.0735	0.1111	0.1000	0.0270	0.0981	0.0724	0.2429
NT09	0.0769	0.0909	0.0889	0.0833	0.0357	0.0980	0.0000	0.0677	0.0362	0.1400
NT10	0.0000	0.0556	0.0732	0.0682	0.0000	0.0625	0.0000	0.0371	0.0351	0.1072
NT11	0.1250	0.0435	0.1829	0.2048	0.0405	0.2235	0.0156	0.1194	0.0866	0.2925
NT12	0.1176	0.1154	0.0247	0.2464	0.0500	0.2857	0.0000	0.1200	0.1094	0.3388
NT13	0.0556	0.0000	0.0465	0.0909	0.0417	0.0400	0.0000	0.0392	0.0318	0.1029
NT14	0.0889	0.0333	0.0980	0.0926	0.1613	0.1250	0.0435	0.0918	0.0442	0.1802
NT15	0.1042	0.0606	0.1765	0.1250	0.0256	0.1167	0.0000	0.0869	0.0614	0.2098
NT16	**0.0714**	0.0000	0.0278	0.0000	0.0000	0.0000	0.0000	0.0142	0.0273	0.0687
NT17	0.0000	**0.0833**	0.0270	0.0250	0.0000	0.0227	0.0000	0.0226	0.0296	0.0817
NT18	0.1220	0.0000	0.1778	0.0566	0.1000	0.0714	0.0500	0.0825	0.0571	0.1968
NT19	0.0822	0.0000	0.1467	0.3692	0.0154	0.2400	0.0000	0.1219	0.1405	0.4028
NT20	0.0000	0.0000	**0.0294**	0.0000	0.0000	0.0000	0.0000	0.0042	0.0111	0.0264
NT21	0.1071	0.0233	0.2545	0.1613	0.0000	0.0857	0.0278	0.0942	0.0902	0.2746
NT22	**0.0714**	0.0000	0.0278	0.0000	0.0000	0.0000	0.0000	0.0142	0.0273	0.0687
NT23	0.0357	0.0000	0.0588	0.0000	0.0000	0.0000	0.0000	0.0135	0.0240	0.0615
NT24	0.0333	0.0000	0.0556	0.0250	0.0588	0.0000	0.0000	0.0247	0.0259	0.0765
μ	0.0754	0.0299	0.0970	0.0865	0.0466	0.0800	0.0121	μ**_tot**		0.0611
σ	0.0500	0.0342	0.0667	0.0890	0.0587	0.0794	0.0214	σ**_tot**		0.0670
Sig. $\geq$	0.1754	0.0983	0.2305	0.2644	0.1641	0.2388	0.0549	**Sig._tot** $\geq$		0.1951

9.2 NLP task & RE (sub-)phase

Table 9.3 features an overview of all Jaccard Indices between the NLP tasks and the RE (sub-)phases. Looking at the matrix, all overall significant values are for tasks that are in the top 5 most frequently used (NT_11 - Part-of-Speech tagging, NT_12 - Rule-based analysis, NT_19 - Syntactic parsing and NT_21 - Tokenization). 6 of the 7 tasks are in 2 sub-phases of requirements analysis; moreover, these are the same 3 tasks. The only other overall significant value is the combination of NT_21 - Tokenization and RP_2 - Requirements analysis.
There are no values that are both significant for their row and for their column. Moreover, 4 out of the 7 overall significant values are for neither their row nor their

column, and the other three are significant solely for their column. Overall, there are only 4 values significant for their row: NT_16 - Sentiment analysis & RP_1 - Requirements elicitation, NT_22 - Topic modelling & RP_1 - Requirements elicitation, N_17 - Spelling checking & RP_1.A - Requirements documentation, and NT_20 - Text summarization & RP_2 - Requirements analysis. These are all tasks with low occurrences. More values are significant for their respective columns; a possible explanation is that these significance values are calculated over more values, decreasing the influence of outliers and therefore the σ.

TABLE 9.4: Jaccard Indices of the NLP tasks intersected with the tool purposes.

	PT1	**PT2**	**PT3**	**PT4**	**PT5**	**PT6**	**PT6A**	**PT6B**	**PT6C**	μ	σ	**Sig.** $\geq$
NT01	0.0625	0.0263	0.0357	0.0833	0.0000	0.0263	0.1250	0.0000	0.0000	0.0399	0.0430	0.1259
NT02	0.0263	0.0000	0.0303	0.1000	0.0625	0.1282	0.0250	0.0556	0.0800	0.0564	0.0409	0.1383
NT03	0.3500	0.1132	0.0652	0.0172	0.0000	0.1569	0.0980	0.0000	0.0500	0.0945	0.1097	0.3138
NT04	0.2813	0.0952	0.0588	0.0222	0.0556	0.0455	0.0238	0.0000	0.0000	0.0647	0.0867	0.2381
NT05	0.0000	0.0000	0.0000	**0.0333**	0.0000	0.0000	0.0000	0.0000	0.0000	0.0037	0.0111	0.0259
NT06	0.1220	0.1860	0.0789	0.0851	0.0000	0.0200	0.0213	0.0400	0.0303	0.0648	0.0597	0.1842
NT07	0.0263	0.1000	0.1333	0.0476	0.0000	0.0732	0.1081	0.0000	0.0385	0.0586	0.0480	0.1545
NT08	0.2200	0.1579	0.0769	0.1000	0.0263	0.0820	0.0678	0.0789	0.0426	0.0947	0.0598	0.2142
NT09	0.0256	0.0465	0.0606	0.1250	0.0000	0.1250	0.1053	0.0000	0.0370	0.0583	0.0495	0.1572
NT10	0.0000	0.0811	0.0000	0.0256	0.0000	0.0526	0.0278	0.1538	0.0952	0.0485	0.0529	0.1542
NT11	0.0602	0.1341	0.1370	0.1772	0.0154	0.1481	0.1842	0.0149	0.1176	0.1099	0.0645	0.2389
NT12	0.0278	0.0972	0.1129	0.2344	0.0000	0.2344	0.2063	0.0000	0.0877	0.1112	0.0948	0.3007
NT13	0.0588	0.0250	0.1071	0.0789	0.0000	0.0789	0.0556	0.0000	0.0000	0.0449	0.0402	0.1254
NT14	0.0930	0.0833	0.1053	0.0833	0.0417	0.0833	0.0889	0.0800	0.0938	0.0836	0.0175	0.1187
NT15	0.0408	0.0769	0.1500	0.0980	0.0000	0.0980	0.0600	0.0333	0.0833	0.0712	0.0438	0.1589
NT16	0.0370	**0.1000**	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0152	0.0341	0.0834
NT17	0.0357	0.0303	0.0000	0.0303	0.0000	0.0000	0.0690	0.0000	0.0000	0.0184	0.0246	0.0675
NT18	0.1579	0.0889	0.1471	0.0426	0.0476	0.0889	0.0455	0.0909	0.0667	0.0862	0.0423	0.1708
NT19	0.0548	0.0789	0.1803	0.3016	0.0000	0.1714	0.1286	0.0364	0.0484	0.1112	0.0943	0.2997
NT20	0.0000	0.0000	0.0000	0.0000	0.0000	0.0333	0.0370	0.0000	0.0000	0.0078	0.0155	0.0389
NT21	0.0169	0.0833	0.1702	0.1607	0.0270	0.0833	0.0690	0.1111	0.0667	0.0876	0.0527	0.1930
NT22	0.0000	**0.0645**	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0072	0.0215	0.0502
NT23	0.0000	0.0323	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	**0.0714**	0.0115	0.0249	0.0613
NT24	0.0357	0.0625	0.0435	0.0303	0.0000	0.0000	0.0000	0.0000	0.0000	0.0191	0.0242	0.0676
μ	0.0722	0.0735	0.0706	0.0782	0.0115	0.0721	0.0644	0.0290	0.0420	μ**_tot**		0.0570
σ	0.0923	0.0481	0.0610	0.0773	0.0203	0.0643	0.0575	0.0438	0.0389	σ**_tot**		0.0625
Sig. $\geq$	0.2568	0.1698	0.1926	0.2328	0.0521	0.2006	0.1794	0.1166	0.1198	**Sig._tot** $\geq$		0.1820

9.3 NLP task & Tool purpose

Table 9.4 features an overview of all Jaccard Indices between the NLP tasks and the tool purposes. This matrix features more overall significant values than the Jaccard matrix of NLP task & RE (sub-)phase. This makes sense as more values are featured in this matrix. Whereas the previous matrix' highest index score was 0.3692, the highest score in this matrix is 0.3500; this occurs even though the significance score (the individual score must be greater than or equal to this score in order to be significant) for this matrix is lower.
The task with the most instances of significance is NT_12 - Rule-based analysis. This is interesting, as it had no significant scores in the previous Jaccard matrix. Again, this is a testimony to the discrepancy between tool purpose and RE (sub-)phase. Moreover, even though PT_2 - elicitation of requirements and RP_1 - requirements elicitation have a significant Jaccard index as their combination in Table 9.2, they share no task that is significant in both matrices.

Other significant values from the Jaccard matrix are the combinations NT_06 - Entity recognition / extraction & PT_2 - Elicitation of requirements, and NT_08 - Linguistic filter & PT_1 - Classification. Both make sense: during elicitation, it is useful to extract certain linguistic entities that identify, for example, what the noun of a customer wish is so that it can be more easily be formed into a user story; and removing stop words from a text will reduce the noise in that text, so that the classification algorithm performs more accurately.

9.4 NLP task & NLP task

Table 9.5 features an overview of all Jaccard Indices between NLP tasks and other NLP tasks. Indices of two same tasks (for example NT01 & NT01) have been ignored as they are trivial; moreover, scores for NT in row & NT in column and NT in column & NT in row are the same. The μ_tot and the σ_tot have been calculated for half of the values only, as the other half is exactly the same; though this makes no difference for the μ_tot, the σ_tot becomes slightly larger.
Table 9.6 features an overview of all ratios between NLP tasks and other NLP tasks. Ratios for both possibilities have been calculated; that is, under the black line, the calculation was performed as follows:
((Occurrences of NLP task in row with NLP task in column) / ((Occurrences of NLP task in row with NLP task in column) + (Occurrences of NLP task in row without NLP task in column))).
Above the black line, the calculation was altered to:
((Occurrences of NLP task in row with NLP task in column) / (Occurrences of NLP task in column with NLP task in row) + (Occurrences of NLP task in row without NLP task in column))).
This alteration makes that in the columns, one can see the ratio task X to other tasks, while in the rows, one can see the ratio other tasks to task X. The averages are scores of the entire row or column, while the ratio is the average of that row/column divided the corresponding column/row of the same task.
A fair multitude of task combinations are significant, as can be seen in the Jaccard matrix. These are:

- NT_04 - Clustering & NT_14 - Semantic similarity
- NT_07 - Lemmatization & NT_11 - Part-of-Speech tagging
- NT_07 - Lemmatization & NT_13 - Semantic Role Labeling
- NT_08 - Linguistic filter & NT_11 - Part-of-Speech tagging
- NT_08 - Linguistic filter & NT_18 - Stemming
- NT_08 - Linguistic filter & NT_21 - Tokenization
- NT_11 - Part-of-Speech tagging & NT_12 - Rule-based analysis
- NT_11 - Part-of-Speech tagging & NT_15 - Sentence splitting
- NT_11 - Part-of-Speech tagging & NT_19 - Syntactic parsing
- NT_11 - Part-of-Speech tagging & NT_21 - Tokenization
- NT_12 - Rule-based analysis & NT_19 - Syntactic parsing

TABLE 9.5: Jaccard Indices of the NLP tasks intersecting with other NLP tasks.

	NT01	**NT02**	**NT03**	**NT04**	**NT05**	**NT06**	**NT07**	**NT08**	**NT09**	**NT10**	**NT11**	**NT12**	**NT13**	**NT14**	**NT15**	**NT16**	**NT17**	**NT18**	**NT19**	**NT20**	**NT21**	**NT22**	**NT23**	**NT24**	μ	σ	**Sig. ≥**
NT01		0.0455	0.1176	0.1176	0.1111	0	0	0	**0.2**	0.1176	0.0909	0.0357	0	0	0.129	0	0	0.037	0.0893	0.1111	0.1	0	0	0.0833	0.0603	0.0593	0.1788
NT02	0.0455		0.075	0.0345	0	0.0294	0.037	0.0204	0	0.0909	**0.1324**	0.0678	0	0.125	0.1111	0	0	0.0313	0.082	0	0.0889	0	0	0	0.0422	0.0450	0.1321
NT03	0.1176	0.0750		0.0714	0	0.1111	0.0488	0.1404	0.0476	0.0263	0.1646	0.0263	0.0811	0.02	0.1224	0.0667	0	0.1429	0.1096	0.0345	0.0847	0	0	0.1	0.0692	0.0508	0.1709
NT04	0.1176	0.0345	0.0714		0	0.1212	0.0345	0.1304	0	0	0.0533	0.0156	0	**0.2667**	0	0	0.1111	0.129	0.0794	0	0.02	0.0556	0	0.0526	0.0562	0.0661	0.1884
NT05	**0.1111**	0.0000	0	0		0	0	0	0.0667	0	0.0159	0.0204	0	0	0.0385	0	0	0	0.0192	0	0.0286	0	0	0	0.0131	0.0271	0.0673
NT06	0.0000	0.0294	0.1111	0.1212	0		0.0606	0.1875	0.0286	0.0333	0.1667	0.0769	0.0323	0.1622	0.0682	0.1429	0.0417	0.1429	0.1061	0	0.0769	0.0435	0	0	0.0709	0.0612	0.1934
NT07	0.0000	0.0370	0.0488	0.0345	0	0.0606		0.1905	0.0357	0	0.2031	0.1053	**0.25**	0.0286	0.1429	0.0625	0	0.1	0.082	0	0.1667	0.1333	0	0	0.0731	0.0755	0.2242
NT08	0.0000	0.0204	0.1404	0.1304	0	0.1875	0.1905		0.0408	0	0.2222	0.1039	0.0682	0.0943	0.1071	0.0541	0.0526	**0.3095**	0.1139	0	0.2679	0.0541	0	0.0256	0.0949	0.0899	0.2748
NT09	**0.2000**	0.0000	0.0476	0	0.0667	0.0286	0.0357	0.0408		0.0417	0.0986	0.1228	0.04	0.0882	0.1081	0	0	0.0303	0.1356	0	0.1364	0	0	0.0556	0.0555	0.0557	0.1669
NT10	0.1176	0.0909	0.0263	0	0	0.0333	0	0	0.0417		0.0896	0.0351	0.05	0	**0.2**	0	0.0769	0	0.069	0.1	0.0976	0	0.0909	0	0.0486	0.0528	0.1543
NT11	0.0909	0.1324	0.1646	0.0533	0.0159	0.1667	0.2031	0.2222	0.0986	0.0896		0.2874	0.0882	0.0897	0.3284	0.0313	0.0308	0.1389	0.2637	0.0159	0.3611	0.0476	0.0317	0.0152	0.1290	0.1046	0.3381
NT12	0.0357	0.0678	0.0263	0.0156	0.0204	0.0769	0.1053	0.1039	0.1228	0.0351	**0.2874**		0.0714	0.0597	0.1194	0	0.0392	0.0462	**0.2469**	0	0.1831	0.04	0	0	0.0740	0.0770	0.2281
NT13	0.0000	0.0000	0.0811	0	0	0.0323	**0.25**	0.0682	0.04	0.05	0.0882	0.0714		0.0313	0.1563	0	0	0.0714	0.0678	0	0.0952	0	0	0	0.0480	0.0614	0.1708
NT14	0.0000	0.1250	0.02	**0.2667**	0	0.1622	0.0286	0.0943	0.0882	0	0.0897	0.0597	0.0313		0.0213	0	0.04	0.1389	0.0882	0	0.1176	0	0.0435	0	0.0615	0.0679	0.1973
NT15	0.1290	0.1111	0.1224	0	0.0385	0.0682	0.1429	0.1071	0.1081	0.2	**0.3284**	0.1194	0.1563	0.0213		0	0	0.0714	0.1471	0.0385	0.3556	0	0.0769	0	0.1018	0.0960	0.2938
NT16	0.0000	0.0000	0.0667	0	0	**0.1429**	0.0625	0.0541	0	0	0.0313	0	0	0	0		0	0.0476	0	0	0	**0.2**	0	0	0.0263	0.0518	0.1299
NT17	0.0000	0.0000	0	**0.1111**	0	0.0417	0	0.0526	0	0.0769	0.0308	0.0392	0	0.04	0	0		**0.0952**	0.0182	0	0.0541	0	0	0	0.0243	0.0341	0.0925
NT18	0.0370	0.0313	0.1429	0.129	0	0.1429	0.1	**0.3095**	0.0303	0	0.1389	0.0462	0.0714	0.1389	0.0714	0.0476	0.0952		0.1094	0	0.2	0	0.05	0.0455	0.0842	0.0745	0.2332
NT19	0.0893	0.0820	0.1096	0.0794	0.0192	0.1061	0.082	0.1139	0.1356	0.069	**0.2637**	0.2469	0.0678	0.0882	0.1471	0	0.0182	0.1094		0	0.2429	0	0.0189	0.037	0.0924	0.0764	0.2453
NT20	**0.1111**	0.0000	0.0345	0	0	0	0	0	0	**0.1**	0.0159	0	0	0	0.0385	0	0	0	0		0.0286	0	0	0	0.0143	0.0312	0.0766
NT21	0.1000	0.0889	0.0847	0.02	0.0286	0.0769	0.1667	0.2679	0.1364	0.0976	**0.3611**	0.1831	0.0952	0.1176	**0.3556**	0	0.0541	0.2	0.2429	0.0286		0.027	0.0278	0.0263	0.1212	0.1045	0.3302
NT22	0.0000	0.0000	0	0.0556	0	0.0435	**0.1333**	0.0541	0	0	0.0476	0.04	0	0	0	**0.2**	0	0	0	0	0.027		0	0	0.0261	0.0498	0.1257
NT23	0.0000	0.0000	0	0	0	0	0	0	0	**0.0909**	0.0317	0	0	0.0435	**0.0769**	0	0	0.05	0.0189	0	0.0278	0		0	0.0148	0.0267	0.0682
NT24	**0.0833**	0.0000	**0.1**	0.0526	0	0	0	0.0256	0.0556	0	0.0152	0	0	0	0	0	0	0.0455	0.037	0	0.0263	0	0		0.0192	0.0297	0.0785
μ	0.0603	0.0422	0.0692	0.0562	0.0131	0.0709	0.0731	0.0949	0.0555	0.0486	0.129	0.074	0.048	0.0615	0.1018	0.0263	0.0243	0.0842	0.0924	0.0143	0.1212	0.0261	0.0148	0.0192	**μ_tot**		0.0592
σ	0.0593	0.045	0.0508	0.0661	0.0271	0.0612	0.0755	0.0899	0.0557	0.0528	0.1046	0.077	0.0614	0.0679	0.096	0.0518	0.0341	0.0745	0.0764	0.0312	0.1045	0.0498	0.0267	0.0297	**σ_tot**		0.0719
Sig. ≥	0.1788	0.1321	0.1709	0.1884	0.0673	0.1934	0.2242	0.2748	0.1669	0.1543	0.3381	0.2281	0.1708	0.1973	0.2938	0.1299	0.0925	0.2332	0.2453	0.0766	0.3302	0.1257	0.0682	0.0785	**Sig._tot ≥**		0.2030

TABLE 9.6: Ratios of the NLP tasks intersecting with other NLP tasks. Scores in the column of a task indicate how much it co-occurs with other tasks, and scores in the row of that same task indicate how often other tasks co-occur with it.

	NT01	NT02	NT03	NT04	NT05	NT06	NT07	NT08	NT09	NT10	NT11	NT12	NT13	NT14	NT15	NT16	NT17	NT18	NT19	NT20	NT21	NT22	NT23	NT24	Average	Ratio
NT01		0.071429	0.137931	0.137931	1	0	0	0	0.266667	0.2	0.095238	0.040816	0	0	0.153846	0	0	0.052632	0.096154	1	0.114286	0	0	0.25	0.157258	0.813809
NT02	0.111111		0.103448	0.0625	0	0.047619	0.071429	0.027778	0	0.2	0.142857	0.081633	0	0.181818	0.153846	0	0	0.052632	0.096154	0	0.114286	0	0	0	0.062918	0.494135
NT03	0.444444	0.214286		0.1875	0	0.238095	0.142857	0.222222	0.133333	0.1	0.206349	0.040816	0.272727	0.045455	0.230769	0.666667	0	0.315789	0.153846	1	0.142857	0	0	0.75	0.239479	2.047851
NT04	0.444444	0.071429	0.103448		0	0.190476	0.071429	0.166667	0	0	0.063492	0.020408	0	0.363636	0	0	0.5	0.210526	0.096154	0	0.028571	0.333333	0	0.25	0.126696	1.054682
NT05	0.111111	0	0	0		0	0	0	0.066667	0	0.015873	0.020408	0	0	0.038462	0	0	0	0.019231	0	0.028571	0	0	0	0.013058	0.042903
NT06	0	0.071429	0.172414	0.25	0		0.142857	0.25	0.066667	0.1	0.190476	0.102041	0.090909	0.272727	0.115385	1	0.25	0.263158	0.134615	0	0.114286	0.333333	0	0	0.170448	1.159524
NT07	0	0.071429	0.068966	0.0625	0	0.095238		0.222222	0.066667	0	0.206349	0.122449	0.454545	0.045455	0.192308	0.333333	0	0.157895	0.096154	0	0.2	0.666667	0	0	0.133138	0.680483
NT08	0	0.071429	0.275862	0.375	0	0.428571	0.571429		0.133333	0	0.285714	0.163265	0.272727	0.227273	0.230769	0.666667	0.5	0.684211	0.173077	0	0.428571	0.666667	0	0.25	0.278459	1.953935
NT09	0.444444	0	0.068966	0	1	0.047619	0.071429	0.055556		0.1	0.111111	0.142857	0.090909	0.136364	0.153846	0	0	0.052632	0.153846	0	0.171429	0	0	0.25	0.132652	0.915302
NT10	0.222222	0.142857	0.034483	0	0	0.047619	0	0	0.066667		0.095238	0.040816	0.090909	0	0.230769	0	0.25	0	0.076923	1	0.114286	0	0.5	0	0.126643	0.882663
NT11	0.666667	0.642857	0.448276	0.25	1	0.571429	0.928571	0.5	0.466667	0.6		0.510204	0.545455	0.318182	0.846154	0.666667	0.5	0.526316	0.461538	1	0.742857	1	1	0.25	0.627906	4.135617
NT12	0.222222	0.285714	0.068966	0.0625	1	0.238095	0.428571	0.222222	0.466667	0.2	0.396825		0.363636	0.181818	0.307692	0	0.5	0.157895	0.384615	0	0.371429	0.666667	0	0	0.283719	2.686985
NT13	0	0	0.103448	0	0	0.047619	0.357143	0.083333	0.066667	0.1	0.095238	0.081633		0.045455	0.192308	0	0	0.105263	0.076923	0	0.114286	0	0	0	0.063883	0.404062
NT14	0	0.285714	0.034483	0.5	0	0.285714	0.071429	0.138889	0.2	0	0.111111	0.081633	0.090909		0.038462	0	0.25	0.263158	0.115385	0	0.171429	0	0.5	0	0.136448	1.150715
NT15	0.444444	0.285714	0.206897	0	1	0.142857	0.357143	0.166667	0.266667	0.6	0.349206	0.163265	0.454545	0.045455		0	0	0.157895	0.192308	1	0.457143	0	1	0	0.032892	0.183827
NT16	0	0	0.068966	0	0	0.142857	0.071429	0.055556	0	0	0.031746	0	0	0	0		0	0.052632	0	0	0	0.333333	0	0	0.027297	0.156957
NT17	0	0	0	0.125	0	0.047619	0	0.055556	0	0.1	0.031746	0.040816	0	0.045455	0	0		0.105263	0.019231	0	0.057143	0	0	0	0.023753	0.13658
NT18	0.111111	0.071429	0.206897	0.25	0	0.238095	0.214286	0.361111	0.066667	0	0.15873	0.061224	0.181818	0.227273	0.115385	0.333333	0.5		0.134615	0	0.257143	0	0.5	0.25	0.184309	1.032605
NT19	0.555556	0.357143	0.275862	0.3125	1	0.333333	0.357143	0.25	0.533333	0.4	0.380952	0.408163	0.363636	0.272727	0.384615	0	0.25	0.368421		0	0.485714	0	0.5	0.5	0.360396	2.89284
NT20	0.111111	0	0.034483	0	0	0	0	0	0	0.1	0.015873	0	0	0	0.038462	0	0	0	0		0.028571	0	0	0	0.014283	0.05475
NT21	0.444444	0.285714	0.172414	0.0625	1	0.190476	0.5	0.416667	0.4	0.4	0.412698	0.265306	0.363636	0.272727	0.615385	0	0.5	0.473684	0.326923	1		0.333333	0.5	0.25	0.399387	2.172343
NT22	0	0	0	0.0625	0	0.047619	0.142857	0.055556	0	0	0.047619	0.040816	0	0	0	0.333333	0	0	0	0	0.028571		0	0	0.032994	0.175124
NT23	0	0	0	0	0	0	0	0	0	0.1	0.031746	0	0	0.045455	0.076923	0	0	0.052632	0.019231	0	0.028571	0		0	0.015416	0.078791
NT24	0.111111	0	0.103448	0.0625	0	0	0	0.027778	0.066667	0	0.015873	0	0	0	0	0	0	0.052632	0.038462	0	0.028571	0	0		0.022045	0.169014
Average	0.1932	0.1273	0.1169	0.1201	0.3043	0.1470	0.1957	0.1425	0.1449	0.1435	0.1518	0.1056	0.1581	0.1186	0.1789	0.1739	0.1739	0.1785	0.1246	0.2609	0.1839	0.1884	0.1957	0.1304		
Ratio	1.2288	2.0237	0.4883	0.9482	23.3083	0.8624	1.4695	0.5118	1.0925	1.1329	0.2418	0.3722	2.4749	0.8690	5.4399	6.3712	7.3217	0.9684	0.3457	18.2648	0.4603	5.7102	12.6919	5.9167		

- NT_15 - Sentence splitting & NT_21 - Tokenization
- NT_19 - Syntactic parsing & NT_21 - Tokenization

One significant combination that we did not expect was the combination Clustering & Semantic similarity. Other significant combinations have at least one task that is frequently occurring; however, both clustering and semantic similarity, though not uncommon tasks, are not that frequent either. The combination does make sense; clustering requirements based on their semantic similarity scores has occurred multiple times in the data set.

Other significant combinations are less surprising. An interesting parallel to draw is between NT_07 - Lemmatization and NT_18 - stemming. As discussed in Chapter 8, both are tasks that have a similar goal, but the output can differ widely; moreover, lemmatization is the more advanced of the two, as it takes context into account. Though stemming is used slightly more often than lemmatization (19 occurrences vs. 14), lemmatization has a significant combination with both NT_11 - Part-of-Speech tagging and NT_13 - Semantic Role Labeling, while stemming only has a significant combination with NT_11 - Part-of-Speech tagging. Thus, one could argue that the use of stemming is more spread out, while the use of lemmatization is more focused within the field.

Discussing the Ratios matrix in Table 9.6, the columns with either white or dark green values are columns with tasks that do not occur very often (ex. NT_05 - Discourse analysis, NT_20 - Automated summarization, NT_16 - Sentiment analysis and NT_22 - Topic modelling). These have very high column ratios, while the score is preferably somewhere around one. One should compare the average of a column with the average of the corresponding row. For example, NT_05 - Discourse analysis has a column average of 0.3043. This means that, on average, in 30.43% of the cases, the task is used with another task. However, when looking at the row average, it gives a score of 0.0429. This means that other tasks work in conjunction with discourse analysis only 4.29% of the time. This gives a use:being used ratio of over 23:1. Preferably, a task has a *use : used* (average in column / average in row) ratio around 1. Examples of this are NT_01 - Anaphora detection resolution, NT_04 - Clustering and NT_09 - Morphological parsing.

Chapter 10

Discussion

This chapter serves as a reflection on the research conducted; mainly, what main challenges and issues we encountered in the research and the data, and wht threats to the validity of the study are relevant.

10.1 Research challenges

During the research, we encountered two main research challenges, along with several smaller research challenges. The first main research challenge was the lack of term uniformity in the field. This was less of a problem for identifying the RE (sub-)phase or the tool purpose, but more for the identification of NLP tasks.
Different authors use different terms to describe the same general concept. A good example is the earlier discussed example of shallow semantic parsing being Semantic Role Labeling. This is an instance that is solved with a Google search; however, a less clear example is from Tjong and Berry (2013):

SREE's lexical analyzer scans a RS, RStat by RStat, and scans each RStat, token by token, for any occurrence of any indicator in the AIC. During the scan, the lexical analyzer of SREE reads tokens from its input RS and compares each token with each indicator in the AIC. If SREE finds a match, it reports the token and its containing RStat as a potentially ambiguous RStat.

This is a description of a task that is encompassed by pattern matching, which is a part of NT_12 - Rule-based analysis. However, reading this for the first time wouldn't make either of those observations clear. That is why most articles had to be read multiple times after the quality assessment phase before being able to correctly identify an NLP task, and later aggregating these. The iterative coding was essential in this process, as without, numerous tasks would still be non-aggregated, making the final list of NLP tasks less comprehensible.
Thus, we would like to ask authors of articles related to NLP in the scope of RE to mark their relevant phases and NLP tasks according to wording that other authors use, instead of trying to word the task or phase based on their own interpretation. Even more ideal would be a common terminology spread throughout the field. As this thesis tries to find terminology that best suits the phases, tools and tasks featured in the research, the results can serve as a starting point. From this point, subcategories can be added, modified to better fit the field, or removed where needed.

The second main research challenge is implicit tasking: tasks that are used in an article, but not explicitly mentioned or described anywhere. Though more complex task are almost always mentioned in a paper, we presume that simpler tasks like tokenization, sentence splitting or stemming are often not mentioned. We base this

presumption on the observation that the count of articles using Part-of-Speech tagging is far higher than the count of articles using sentence splitting or tokenization, two tasks that would normally be done prior to PoS tagging. Now, one might argue that these are tasks are not as important as PoS tagging; however, leaving out this data does decrease the reproducibility of the study.
Moreover, implicit tasking has influenced our data set. The choice to only include tasks explicitly mentioned in the article is based on the fact that coding an article based on a gut feeling makes validating this choice impossible. Therefore, solely explicit tasks are included. Though we presume that multiple articles implicitly use sentence splitting and tokenization, and perhaps other tasks to a lesser extent, we cannot include these implicit tasks. Thus, we believe that it has some influence to the final number of occurrences included in the data set. Therefore, we request the authors of future studies to include any and all NLP tasks into their report; this helps to improve on later iterations of similar studies to this one.

There were several minor research challenges. The first one was the number of results that the queries returned; though the relevance of most of these results could be determined quickly, the amount of work was high. Tying into the amount of work is the high number of irrelevant results: of the 2,239 articles from the queries, only 144 were eventually used. However, a high amount of articles were returned during the search that had one of the terms only in either the related works section or the references, rendering them useless. The amount of manual work was also higher than expected; especially during the study selection, study quality assessment/data extraction and creation of the crosstabs for calculating the Jaccard Indices, this amount stood out. All these challenges were eventually overcome.
A last challenge that was more difficult than expected when starting the study was getting a concise understanding of both the fields of RE and that of NLP, and this understanding is likely to still not be complete. However, this only calls for more research to be done so that an even clearer picture of NLP can be created within the entire field of NLP (as opposed to only elicitation and analysis).

10.2 Threats to validity

Zhou et al. (2016) discusses the threats to validity (TTV) possible for systematic literature reviews. In this section, we will discuss the threats relevant to our study.

10.2.1 Planning phase

A relevant TTV in this phase was the incorrect or incomplete search terms in the automatic search, because of the lack of inclusion of 'requirements classification' into the query for the study search. This was not a deliberate choice; only after the search was discussed and decided that requirements classification should be included into the scope of the study as a sub-phase of requirements analysis. Therefore, it is possible that some relevant papers have not been included because of this oversight. However, we hope that the inclusion of the argument 'requirements engineering' has alleviated this TTV somewhat, as that is a much more common argument for anything related to RE than 'requirements classification'.
Also relevant to this same TTV is the decision to not include separate NLP tasks into the query, and only including NLP and related terms. Though some results might have been left out, the reason is twofold: first, it would make finding any of those

terms in every article more difficult, and second, we believe that it would only have yielded more irrelevant results with little pay-off in terms of relevant results.
The last limitation falling under the same TTV is not having a complete definition of requirements analysis. The choice to focus on requirements elicitation and requirements analysis was made early on, in order to focus the literature search. However, though requirements elicitation is well-established within the field of RE, requirements analysis and its exact definition is more ambiguous. This has also been discussed at the start of Chapter 5, and is the reason that we established our own definition. The TTV is that there is a chance that we did not include some specific subphases that would have been within our definition's parameters. Again, we hope that including the argument 'requirements engineering' has decreased this TTV.
We believe that other TTV are not applicable to our study. The research questions are appropriate for the main research question, the search and sources have been clearly stated beforehand and are established from well-established search arguments and arguments respectively, and the SLR has been documented well in this report. There is also no cultural bias involved and though the time span had some restrictions, the time frame for from what years the articles have been collected has been clearly stated.

10.2.2 Conducting phase

A TTV relevant here is subjective interpretation about the extracted data: a difference between the researchers on how the extracted data could be interpreted. This applies to our study because of the risk of incorrect or incomplete coding. Even though the articles were coded and validated in multiple cycles during the iterative coding phase, a possibility exists that some coding might be incorrect, or that some sections of text should have been coded as well. This effect is a result of the researchers being human, and thus prone to error, as well as having their differences in interpretation of certain texts between them. Additionally, the main author was not an experienced researcher in either NLP or RE. What alleviates the effect is that the coding was done by 2 researchers instead of just one, decreasing the risk because of the validation between the researchers. The risk is not eliminated because of the aforementioned inexperience of the main author, and a double-check is also partial and error-prone. This TTV might have been eliminated altogether by using classification techniques; however, this introduces the problem of ignoring the references and the related works, and at the time of the coding, no training data was available. Perhaps future studies can adjust our coding and validate the articles' categorization using classification based on a training set as result from our data.
One could also argue that the data synthesis was unsatisfactory, and that the textual data should have been used as well instead of just the categorization and frequencies. However, the textual data extracted is not perfect; because of the limitations of NVivo, some of the data is jumbled and incomplete. Therefore, the choice was made to use the categorization, but the textual data. Also important to this TTV is that the use of the textual data would not directly make sense. The set could have been used for a Term-Frequency matrix per task/phase/purpose, or perhaps for clustering; however, especially the latter brings a lot of extra effort, which might not be paid off by the resulting clusters.
Another TTV is an identification error of primary studies in the searching process. In our research, it is possible that some results were considered out of scope even though they were not. Though we have established both our study selection criteria and quality assessment criteria in Chapter 6, it is possible that several articles could

have been wrongly interpreted. This is much more likely during the study selection, as articles were read less carefully than during the quality assessment. This limitation is a result of the large amount of literature that had to be selected in combination with a not fully developed understanding of the field. We do not fear that any article that is currently in scope should be out of scope, because of the more rigorous quality assessment.
Other TTV are not applicable to our study in our opinion. We removed all duplicate and inaccessible studies from scope, had no bias in our study selection or quality assessment as the selection and quality assessment criteria were set up beforehand, the SLR was well-documented, and we believe that no classification errors in regard to the primary studies were made.

10.2.3 Reporting phase

Two TTV exist in this phase: The generalizability of the primary study and the lack of expert evaluation. We believe of both these TTV that they are not applicable to our research. As the research was performed over a wide field of literature within RE, the generalizability should not come in danger. Moreover, one expert was directly involved with this research as a first supervisor, and one expert as a second supervisor; therefore, there was no shortage of expert evaluation.

Chapter 11

Conclusion

The main research question posed in Chapter 1 was:

What is the current state of research on Natural Language Processing within Requirements Elicitation and Requirements Analysis?

Based on the SLR that we have conducted and the data that we have presented, we believe that we can now provide an answer to this question.

11.1 Conclusion

NLP within requirements elicitation and requirements analysis is a long way from mature. Though more research is being done on the subject, and there is a trend visible that more and more research is being published in journals, the NLP tasks themselves see little increase in the ratio complex to simple. Syntactic NLP tasks are still predominant. This not an unexpected finding, as syntactic NLP tasks are often used to pre-process text so it can be used for semantic and pragmatic tasks, but pragmatic tasks have not occurred in the last few years, and semantic tasks seem to have stagnated. Of the top 5 tasks used within the field, none are semantic tasks. Thus, more research should be done towards how semantic tasks can be better applied within the phases and tool purposes.
Within the field, NLP is predominantly being used to analyze requirements and prepare them for further processing. This is mostly centered around creating models from elicited requirements, and improving the quality of the Software Requirements Specification (SRS). Both these fields also have the same three tasks that frequently occur: Part-of-Speech tagging, rule-based analysis and syntactic parsing. Two areas that are being left behind compared to other RE (sub-)phases within the scope are requirements documentation, e.g. the writing of the SRS itself, and prioritization of requirements. We believe that writing requirements supported by NLP, for example by enforcing a certain template, including spelling checking and directly resolving any potential ambiguity, can prove very beneficial to requirements engineers in the industry. However, as these tools require live interaction with requirement engineers, it is understandable that research is not as frequent as less specialized tools (solely for ambiguity checking, for example).
The tool purposes classification and model extraction / generation seem to be more mature than others, with more articles being written on them and NLP tasks that are more frequently used in conjunction than in other fields. This is also valid for ambiguity checking. Some categories that are left behind are tools that prioritize requirements or execute change impact/dependency analysis, and to a lesser extent tools that check the completeness or conformance of requirements.

An observation we made is that even though a tool for RE can have a certain purpose and use certain NLP tasks to achieve that purpose, this tool purpose can belong to multiple RE (sub-)phases, even at once. We found this to be especially valid for tools that include classification or clustering. This phenomenon does make sense, as both classification and clustering are techniques with a very broad potential for implementation.
As for the NLP tasks themselves, we have found that common tasks include Part-of-Speech tagging, syntactic parsing, rule-based analysis, tokenization and linguistic filters. Rule-based analysis is the one that stands out among these five, as the other tasks are more pre-processing tasks while rule-based analysis is a task that is often further up the NLP chain. This particular task also has numerous significant intersections with a plethora of tool purposes, illustrating its wide field of applicability. As noted, pragmatic tasks tend to stay behind on these tasks, even though they have potential in the field. Discourse analysis can be used to directly elicit requirements from transcripts of discussions between stakeholders and programmers, and automated summarization can be used to summarize any SRS to a shortened version that is ideal for management to skim through.
Few of the task combinations that frequently occurred went against our expectations. Often, these task combinations stem from tasks that occur frequently in general, like Part-of-Speech tagging or tokenization. A noteworthy pattern was semantic similarity combined with clustering, a combination we did not expect to occur as frequently as it did. Also noteworthy is that even though a task frequently occurs with another task, this might not be reciprocated. Examples of this phenomenon are discourse analysis, automated summarization and topic modelling. Thus, one should be wary of ratios that show only one side of the coin.
We encountered two phenomena that hindered our research. The first phenomenon is a lack of common terminology in the field, as different researchers use different terms to describe the same concept. Secondly is the habit of researchers not to explicitly describe some of the less-advanced NLP tasks that are nevertheless used in their research. We therefore implore researchers to be as complete as possible when it comes to describing their NLP chains, and reuse terms of other researchers instead of trying to force their own descriptions.

11.2 Research directions

This study leaves a multitude of research directions open, both as continuation of this study and as related research that this study identified as interesting. First, by suggestion from dr. Andreas Vogelsang, there is little insight in what chains of NLP tasks could be employed to facilitate more sophisticated techniques. Though this study identified techniques that frequently co-occur, combinations of tasks that frequently co-occur with other tasks (for example, stemming and a linguistic filter combined with classification) have been neglected. It can be interesting to research, in the perspective of RE, how NLP task combinations can be optimized in order to execute another task to its fullest potential.
Second, as noted, this research started with Automated Reasoning in mind. Though this field is still an interesting field to research in a requirements engineering perspective, other fields that could yield more benefit are machine learning and artificial intelligence. On that same note, requirements mining has not been researched because of its very small literature base, but the mining for requirements in large

text corpora (for example, product reviews or customer service e-mails and calls) can prove beneficial to the industry.
A frequent term that occurred during the study selection, but was considered out of scope, is requirements traceability: establishing links between requirements and the software artifacts that implement them. Based on this observation, a systematic literature review to NLP within requirements traceability and its applications is noteworthy. The same can be said for NLP in requirements management: how requirements are handled within a software system from elicitation to implementation to reuse.
We presume that the NLP tasks defined in this research are not the only NLP tasks that currently exist. A potential continuation of this study could look into NLP tasks that have not yet been employed in the field, and how they could be employed. In a broader sense, one could also look into the current state of computational linguistics, text mining and text-based information retrieval, and how techniques in these fields can benefit the field of RE.
Lastly, this research has not taken any other fields where NLP is used into account. In Chapter 1, it has been established that NLP also sees plenty of use in other fields. Though it was the intention of this study to perform a study on NLP in other fields as well, this plan was eventually scrapped in favor of more rigorous data analysis. Therefore, the potential is still open, and researching NLP in other domains can prove directions on how these tasks kan be better employed within RE.

11.3 Acknowledgements

First and foremost, I would like to thank my first supervisor, dr. Fabiano Dalpiaz. Throughout the year that I've been writing his thesis, dr. Dalpiaz has been extremely helpful in conducting the research itself, has taken admirable amounts of time to review and validate my work, and has managed to put up with me and my ignorance almost every week, something even I myself find hard to do. Fabiano, thank you for guiding me through this study.
I would also like to thank my second supervisor, dr. Marcela Ruiz, for reviewing the research proposal and the final document, and challenging me to question my own research during the thesis defense. A thanks is also in order to dr. Andreas Vogelsang of the Technische Universität Berlin, for providing advice on conducting this research.
Finally, I would like to thank my family, friends and colleagues for being understanding and supporting me through the past year. Especially my cordial, Vanguard, though I'm still not sure whether I hate you or love you all.

Appendix A

Results from preliminary study

Conference year	Title	AR/AI/NLP-related?	Tool?	Relevant to elicitation?	Relevant to analysis?	Relevant phases, as gathered from abstract and keywords
2017	Bhatia and Breaux (2017)	NLP	No	No	Yes	Requirements writing
2017	Walter et al. (2017)	NLP	Method	No	Yes	Requirements specification
2017	Guzman, Ibrahim, and Glinz (2017a)	NLP and machine learning	Yes	Yes	No	Unclear
2017	Kanchev et al. (2017a)	Automated extraction	Yes	Yes	No	Requirements elicitation
2017	Cruz et al. (2017b)	NLP	Yes	Yes	No	Requirements elicitation
2017	Sannier et al. (2017b)	NLP	Extension	Yes	No	RE in general
2017	Noaeen, Abad, and Far (2017b)	NLP, machine learning	Yes	Yes	Yes	Requirements elicitation, requirements classification
2017	Williams and Mahmoud (2017)	NLP	No	Yes	Yes	Requirements elicitation
2017	Johann, Stanik, and Maalej (2017)	NLP	Approach	Yes	No	Feature extraction
2017	Tiwari and Laddha (2017)	NLP	Yes	Yes	Yes	Requirements modelling, requirements analysis
2017	Groen et al. (2017)	Automated extraction	No	Yes	Yes	Requirements analysis
2017	Abad et al. (2017)	Machine learning, automated classification	Approach	No	Yes	Requirements classification
2016	Bhatia et al. (2016a)	NLP	Theory	No	Yes	Requirements analysis
2016	Robeer et al. (2016b)	NLP, automated extraction	Method	Yes	No	Requirements elicitation
2015	Mahmoud (2015)	Automated extraction, clustering, information theory	Approach	No	Yes	Requirements analysis, requirements traceability, requirements classification
2015	Maalej and Nabil (2015a)	Text classification, NLP, sentiment analysis	Several techniques	Yes	Yes	RE in general
2015	Arora et al. (2015b)	NLP	No	No	Yes (change impact analysis	Requirements analysis, requirements specification
2015	Mahmoud and Carver (2015)	NLP	No	Sort of	Sorf of	Requirements analysis
2015	Lucassen et al. (2015b)	NLP	Yes	Unclear	Yes	Requirements authoring, requirements analysis
2015	Alabdulkareem, Cercone, and Liaskos (2015a)	NLP	Technique	No	Yes	Requirements modelling, requirements analysis
2015	Wüest, Seyff, and Glinz (2015)	NLP	No	Yes	Yes	Requirements elicitation, requirements modelling
2014	Adedjouma, Sabetzadeh, and Briand (2014)	NLP	Yes, though unnamed	Yes	No	Requirements elicitation
2014	Riaz et al. (2014a)	NL parsing	No	Yes	No	Requirements identification
2014	Guzman and Maalej (2014b)	NLP, automated extraction	Approach	No	Yes	Requirements analysis
2014	Gärtner et al. (2014)	NLP	No	No	Yes	Requirements analysis
2014	Pruski et al. (2014)	NLP	Yes	No	Sort of	Requirements traceability
2014	Badger, Throop, and Claunch (2014)	NLP	Yes	No	Yes	Requirements verification, requirements analysis
2013	Dwarakanath, Ramnani, and Sengupta (2013b)	NLP	Method	No	Yes	Requirements analysis
2013	Sutcliffe and Sawyer (2013)	Several mentioned	Framework used for evaluation	Yes	No	Requirements elicitation
2012	Yang et al. (2012a)	NLP, machine learning, rule-based reasoning (possibly)	Approach	No	Yes	Requirements analysis
2012	Ferrari and Gnesi (2012a)	NLP (ambiguity detection	Approach	No	Yes	Requirements specification, requirements analysis
2011	Fitzgerald, Letier, and Finkelstein (2011)	Unclear	Method	No	Yes	Requirements analysis
2011	Greenwood et al. (2011)	NLP	Yes	No	Yes	Requirements modelling
2011	Boutkova and Houdek (2011a)	Some sort of NLP	Approach	Yes	No	Requirements reuse
2010	Sinha et al. (2010b)	NLP	Extension	Yes	No	Requirements validation, requirements elicitation
2010	Kof (2010)	NLP	Yes	No	Yes	Requirements analysis
2010	Kamalrudin, Hosking, and Grundy (2010)	Supposedly yes (NLP)	Yes	No	Yes	Requirements analysis
2010	Gacitua, Sawyer, and Gervasi (2010)	Unclear, presumably yes	Technique	Yes	Yes	Requirements elicitation

Appendix B

Full list of articles researched during the preliminary study

Note: the year references to the year that the respective Requirements Engineering conference took place in. All articles can be viewed at: https://www.computer.org/csdl/proceedings/r/list.html#collapse_RE .

Year	Author(s)	Title
2017	G. Williams and A. Mahmoud	Mining Twitter Feeds for Software User Requirements
2017	E. Guzman, M. Ibrahim and M. Glinz	A Little Bird Told Me: Mining Tweets for Requirements and Software Evolution
2017	T. Johann, C. Stanik, A. M. B. and W. Maalej	SAFE: A Simple Approach for Feature Extraction from App Descriptions and App Reviews
2017	G. M. Kanchev, P. K. Murukannaiah, A. K. Chopra and P. Sawyer	Canary: Extracting Requirements-Related Information from Online Discussions
2017	D. v. Linden, A. Zamansky and I. Hadar	A Framework for Improving the Verifiability of Visual Notation Design Grounded in the Physics of Notations
2017	Y. Elrakaiby, A. Ferrari, P. Spoletini, S. Gnesi and B. Nuseibeh	Using Argumentation to Explain Ambiguity in Requirements Elicitation Interviews
2017	Z. Kurtanovic and W. Maalej	Mining User Rationale from Software Reviews
2017	M. Stade, F. Fotrousi, N. Seyff and O. Albrecht	Feedback Gathering from an Industrial Point of View
2017	E. C. Groen, S. Kopczynska, M. P. Hauer, T. D. Krafft and J. Doerr	Users — The Hidden Software Product Quality Experts?: A Study on How App Users Report Quality Aspects in Online Reviews
2017	M. Goodrum, J. Cleland-Huang, R. Lutz, J. Cheng and R. Metoyer	What Requirements Knowledge Do Developers Need to Manage Change in Safety-Critical Systems?
2017	S. Malviya, M. Vierhauser, J. Cleland-Huang and S. Ghaisas	What Questions do Requirements Engineers Ask?
2017	W. Zogaan, P. Sharma, M. Mirahkorli and V. Arnaoudova	Datasets from Fifteen Years of Automated Requirements Traceability Research: Current State, Characteristics, and Quality
2017	S. Turpe	The Trouble with Security Requirements
2017	M. Lindvall, M. Diep, M. Klein, P. Jones, Y. Zhang and E. Vasserman	Safety-Focused Security Requirements Elicitation for Medical Device Software
2017	H. Hibshi and T. D. Breaux	Reinforcing Security Requirements with Multifactor Quality Measurement
2017	G. Mathew, T. Menzies, N. A. Ernst and J. Klein	"SHORT"er Reasoning About Larger Requirements Models
2017	A. M. Grubb and M. Chechik	Modeling and Reasoning with Changing Intentions: An Experiment
2017	A. Mavin, P. Wilkinson, S. Teufl, H. Femmer, J. Eckhardt and J. Mund	Does Goal-Oriented Requirements Engineering Achieve Its Goal?

2017	D. Callele, K. Wnuk and B. Penzenstadler	New Frontiers for Requirements Engineering
2017	S. Saito, Y. Iimura, A. K. Massey and A. I. Anton	How Much Undocumented Knowledge is there in Agile Software Development?: Case Study on Industrial Project Using Issue Tracking System and Version Control System
2017	A. T. Calazans et al.	Software Requirements Analyst Profile: A Descriptive Study of Brazil and Mexico
2017	A. Maier and D. M. Berry	Improving the Identification of Hedonic Quality in User Requirements — A Controlled Experiment
2017	H. Gaspard-Boulinc and S. Conversy	Usability Insights for Requirements Engineering Tools: A User Study with Practitioners in Aeronautics
2017	B. D. Cruz, B. Jayaraman, A. Dwarakanath and C. McMillan	Detecting Vague Words & Phrases in Requirements Documents in a Multilingual Environment
2017	M. Warnier and A. Condamines	A Case Study on Evaluating the Relevance of Some Rules for Writing Requirements Through an Online Survey
2017	M. Beckmann, A. Vogelsang and C. Reuter	A Case Study on a Specification Approach Using Activity Diagrams in Requirements Documents
2017	B. Walter, J. Hammes, M. Piechotta and S. Rudolph	A Formalization Method to Process Structured Natural Language to Logic Expressions to Detect Redundant Specification and Test Statements
2017	K. Lauenroth, E. Kamsties and O. Hehlert	Do Words Make a Difference? An Empirical Study on the Impact of Taxonomies on the Classification of Requirements
2017	A. Crapo, A. Moitra, C. McMillan and D. Russell	Requirements Capture and Analysis in ASSERT(TM)
2017	X. Lian, J. Cleland-Huang and L. Zhang	Mining Associations Between Quality Concerns and Functional Requirements
2017	N. Sannier et al.	Legal Markup Generation in the Large: An Experience Report
2017	M. C. Evans, J. Bhatia, S. Wadkar and T. D. Breaux	An Evaluation of Constituency-Based Hyponymy Extraction from Privacy Policies
2017	F. M. Kifetew et al.	Gamifying Collaborative Prioritization: Does Pointsification Work?
2017	M. Z. Kolpondinos and M. Glinz	Behind Points and Levels — The Influence of Gamification Algorithms on Requirements Prioritization
2017	Z. S. Abad, G. Ruhe and M. Bauer	Task Interruptions in Requirements Engineering: Reality Versus Perceptions!

2017	R. Kasauli, G. Liebel, E. Knauss, S. Gopakumar and B. Kanagwa	Requirements Engineering Challenges in Large-Scale Agile System Development
2017	L. Montgomery and D. Damian	What do Support Analysts Know About Their Customers? On the Study and Prediction of Support Ticket Escalations in Large Software Organizations
2017	M. Borg, T. Olsson and J. Svensson	Piggybacking on an Autonomous Hauler: Business Models Enabling a System-of-Systems Approach to Mapping an Underground Mine
2017	X. Franch et al.	How do Practitioners Perceive the Relevance of Requirements Engineering Research? An Ongoing Study
2017	M. Nayebi and G. Ruhe	Optimized Functionality for Super Mobile Apps
2017	J. Bhatia and T. D. Breaux	A Data Purpose Case Study of Privacy Policies
2017	A. Ferrari, P. Spoletini, B. Donati, D. Zowghi and S. Gnesi	Interview Review: Detecting Latent Ambiguities to Improve the Requirements Elicitation Process
2017	A. Zamansky, D. v. Linden and S. Baskin	Pushing Boundaries of RE: Requirement Elicitation for Non-human Users
2017	N. Mead, F. Shull, J. Spears, S. Heibl, S. Weber and J. Cleland-Huang	Crowd Sourcing the Creation of Personae Non Gratae for Requirements-Phase Threat Modeling
2017	D. Alami and F. Dalpiaz	A Gamified Tutorial for Learning About Security Requirements Engineering
2017	I. Morales-Ramirez, D. Munante, F. Kifetew, A. Perini, A. Susi and A. Siena	Exploiting User Feedback in Tool-Supported Multi-criteria Requirements Prioritization
2017	L. H. Paucar, N. Bencomo and K. K. Yuen	Juggling Preferences in a World of Uncertainty
2017	T. Viana, A. Zisman and A. K. Bandara	Identifying Conflicting Requirements in Systems of Systems
2017	Z. S. Abad, A. Shymka, J. Le, N. Hammad and G. Ruhe	A Visual Narrative Path from Switching to Resuming a Requirements Engineering Task
2017	S. Tiwari and M. Laddha	UCAnalyzer: A Tool to Analyze Use Case Textual Descriptions
2017	M. Noaeen, Z. S. Abad and B. H. Far	Let's Hear it from RETTA: A Requirements Elicitation Tool for TrAffic Management Systems
2017	L. Montgomery, E. Reading and D. Damian	ECrits — Visualizing Support Ticket Escalation Risk
2017	R. Darimont, W. Zhao, C. Ponsard and A. Michot	Deploying a Template and Pattern Library for Improved Reuse of Requirements Across Projects
2017	L. H. Paucar and N. Bencomo	ARRoW: Tool Support for Automatic Runtime Reappraisal of Weights

2017	P. Ghazi, Z. S. Abad and M. Glinz	Choosing Requirements for Experimentation with User Interfaces of Requirements Modeling Tools
2017	M. F. Granda, N. Condori-Fernandez, T. E. Vos and O. Pastor	CoSTest: A Tool for Validation of Requirements at Model Level
2017	F. Kifetew, D. Munante, A. Perini, A. Susi, A. Siena and P. Busetta	DMGame: A Gamified Collaborative Requirements Prioritisation Tool
2017	G. M. Kanchev, P. K. Murukannaiah, A. K. Chopra and P. Sawyer	Canary: An Interactive and Query-Based Approach to Extract Requirements from Online Forums
2017	M. Ledger	A Demonstration of Respecify: A Requirements Authoring Tool Harnessing CNL
2017	M. A. Jackson	The Right-Hand Side Problem: Research Topics in RE
2017	P. Spoletini and A. Ferrari	Requirements Elicitation: A Look at the Future Through the Lenses of the Past
2017	G. Ruhe, M. Nayebi and C. Ebert	The Vision: Requirements Engineering in Society
2017	R. Wieringa	Requirements Engineering Since the Year One Thousand
2017	R. R. Lutz	RE at 50, with a Focus on the Last 25 Years
2017	A. Dekhtyar and V. Fong	RE Data Challenge: Requirements Identification with Word2Vec and TensorFlow
2017	Z. Kurtanovic and W. Maalej	Automatically Classifying Functional and Non-functional Requirements Using Supervised Machine Learning
2017	Z. S. Abad, O. Karras, P. Ghazi, M. Glinz, G. Ruhe and K. Schneider	What Works Better? A Study of Classifying Requirements
2017	A. Ferrari, G. O. Spagnolo and S. Gnesi	PURE: A Dataset of Public Requirements Documents
2017	N. Munaiah, A. Meneely and P. K. Murukannaiah	A Domain-Independent Model for Identifying Security Requirements
2017	P. K. Murukannaiah, N. Ajmeri and M. P. Singh	Toward Automating Crowd RE
2017	M. Rath, P. Rempel and P. Mader	The IlmSeven Dataset
2017	O. Oni	Towards a Bayesian Decision Model for Release Planning in Incremental Development
2017	A. I. Leite	An Approach to Support the Specification of Agile Artifacts in the Development of Safety-Critical Systems
2017	M. B. Duran	Reusable Goal Models
2017	J. Dabrowski	Towards an Adaptive Framework for Goal-Oriented Strategic Decision-Making

2017	R. M. Carvalho	Dealing with Conflicts Between Non-functional Requirements of UbiComp and IoT Applications
2017	A. M. Moura	Awareness Driven Software Reengineering
2017	J. Brings	Verifying Cyber-Physical System Behavior in the Context of Cyber-Physical System-Networks
2017	R. M. Falcao	Improving the Elicitation of Delightful Context-Aware Features: A Data-Based Approach
2017	N. Niu	RE in the Age of Continuous Deployment
2017	D. M. Berry, J. Cleland-Huang, A. Ferrari, W. Maalej, J. Mylopoulos and D. Zowghi	Panel: Context-Dependent Evaluation of Tools for NL RE Tasks: Recall vs. Precision, and Beyond
2016	J. Bosch	Delivering Customer Value in the Age of Autonomous, Continuously Evolving Systems
2016	R. R. Lutz	Requirements for Molecular Programmed Nanosystems (Keynote)
2016	Y. Zheng	Urban Computing: Tackling Urban Challenges Using Big Data
2016	C. Ebert and C. H. Duarte	Requirements Engineering for the Digital Transformation: Industry Panel
2016	Z. S. Abad, M. Noaeen and G. Ruhe	Requirements Engineering Visualization: A Systematic Literature Review
2016	K. Beckers and S. Pape	A Serious Game for Eliciting Social Engineering Security Requirements
2016	J. Bhatia, T. D. Breaux, J. R. Reidenberg and T. B. Norton	A Theory of Vagueness and Privacy Risk Perception
2016	C. Burnay, J. Horkoff and N. Maiden	Stimulating Stakeholders' Imagination: New Creativity Triggers for Eliciting Novel Requirements
2016	J. Eckhardt, A. Vogelsang, H. Femmer and P. Mager	Challenging Incompleteness of Performance Requirements by Sentence Patterns
2016	A. Ferrari, P. Spoletini and S. Gnesi	Ambiguity Cues in Requirements Elicitation Interviews
2016	F. Furtado and A. Zisman	Trace++: A Traceability Approach to Support Transitioning to Agile Software Engineering
2016	P. Ghazi and M. Glinz	An Exploratory Study on User Interaction Challenges When Handling Interconnected Requirements Artifacts of Various Sizes
2016	A. M. Grubb and M. Chechik,	Looking into the Crystal Ball: Requirements Evolution over Time
2016	E. Guzman, R. Alkadhi and N. Seyff	A Needle in a Haystack: What Do Twitter Users Say about Software?

2016	J. Horkoff et al.	Goal-Oriented Requirements Engineering: A Systematic Literature Map
2016	S. Hotomski, E. B. Charrada and M. Glinz	An Exploratory Study on Handling Requirements and Acceptance Test Documentation in Industry
2016	W. Hussain, D. Zowghi, T. Clear, S. MacDonell and K. Blincoe	Managing Requirements Change the Informal Way: When Saying 'No' is Not an Option
2016	O. Kafali, M. P. Singh and L. Williams	NANE: Identifying Misuse Cases Using Temporal Norm Enactments
2016	O. Karras, S. Kiesling and K. Schneider	Supporting Requirements Elicitation by Tool-Supported Video Analysis
2016	X. Lian, M. Rahimi, J. Cleland-Huang, L. Zhang, R. Ferrai and M. Smith	Mining Requirements Knowledge from Collections of Domain Documents
2016	P. K. Murukannaiah, N. Ajmeri and M. P. Singh	Acquiring Creative Requirements from the Crowd: Understanding the Influences of Personality and Creative Potential in Crowd RE
2016	N. Niu, A. Koshoffer, L. Newman, C. Khatwani, C. Samarasinghe and J. Savolainen	Advancing Repeated Research in Requirements Engineering: A Theoretical Replication of Viewpoint Merging
2016	M. Robeer, G. Lucassen, J. M. Werf, F. Dalpiaz and S. Brinkkemper	Automated Extraction of Conceptual Models from User Stories via NLP
2016	M. Santos, C. Gralha, M. Goulao, J. Araujo, A. Moreira and J. Cambeiro	What is the Impact of Bad Layout in the Understandability of Social Goal Models?
2016	R. Wohlrab, J. Steghofer, E. Knauss, S. Maro and A. Anjorin	Collaborative Traceability Management: Challenges and Opportunities
2016	J. Baker et al.	Requirements Engineering for Retrofittable Subsea Equipment
2016	A. M. Duarte, D. Duarte and M. Thiry	TraceBoK: Toward a Software Requirements Traceability Body of Knowledge
2016	A. Furfaro, T. Gallo, A. Garro, D. Sacca and A. Tundis	ResDevOps: A Software Engineering Framework for Achieving Long-Lasting Complex Systems
2016	M. Gharib et al.	Privacy Requirements: Findings and Lessons Learned in Developing a Privacy Platform
2016	L. Liu, L. Feng, Z. Cao and J. Li	Requirements Engineering for Health Data Analytics: Challenges and Possible Directions
2016	A. Mavin, P. Wilksinson, S. Gregory and E. Uusitalo	Listens Learned (8 Lessons Learned Applying EARS)

2016	M. Rudolph, D. Feth, J. Doerr and J. Spilker	Requirements Elicitation and Derivation of Security Policy Templates—An Industrial Case Study
2016	I. Oliver	Experiences in the Development and Usage of a Privacy Requirements Framework
2016	G. Soltana, M. Sabetzadeh and L. C. Briand	Model-Based Simulation of Legal Requirements: Experience from Tax Policy Simulation
2016	A. Takoshima and M. Aoyama	A Two-Stage Inspection Method for Automotive Software Systems and Its Practical Applications
2016	J. Terzakis and S. Gregory	RAMP: Requirements Authors Mentoring Program
2016	D. Amyot, A. A. Anda, M. Baslyman, L. Lessard and J. Bruel	Towards Improved Requirements Engineering with SysML and the User Requirements Notation
2016	S. A. Fricker, E. Wallmuller and I. Paschen	Requirements Engineering as Innovation Journalism: A Research Preview
2016	J. Gillain, C. Burnay, I. Jureta and S. Faulkner	AnalyticGraph.com: Toward Next Generation Requirements Modeling and Reasoning Tools
2016	F. R. Golra, A. Beugnard, F. Dagnat, S. Guerin and C. Guychard	Continuous Requirements Engineering Using Model Federation
2016	D. v. Linden, I. Hadar and A. Zamansky	Towards a Marketplace of Visual Elements for Notation Design
2016	A. Murugesan, M. W. Whalen, E. Ghassabani and M. P. Heimdahl	Complete Traceability for Requirements in Satisfaction Arguments
2016	L. Piras, P. Giorgini and J. Mylopoulos	Acceptance Requirements and Their Gamification Solutions
2016	P. Spoletini, C. Brock, R. Shahwar and A. Ferrari	Empowering Requirements Elicitation Interviews with Vocal and Biofeedback Analysis
2016	L. Shi, C. Chen, Q. Wang and B. Boehm	Is It a New Feature or Simply "Don't Know Yet"?: On Automated Redundant OSS Feature Requests Identification
2016	A. Sutcliffe and P. Sawyer	Beyond Awareness Requirements
2016	M. Touzani and C. Ponsard	Towards Modelling and Analysis of Spatial and Temporal Requirements
2016	S. Pradhan, A. V. Patil and B. Penzenstadler	Application of Artifact-Based RE in Fortified Cards
2016	N. Deb, N. Chaki and A. Ghose	i*ToNuSMV: A Prototype for Enabling Model Checking of i* Models

2016	N. Gao and Z. Li	Generating Testing Codes for Behavior-Driven Development from Problem Diagrams: A Tool-Based Approach
2016	Y. Jiang, S. Wang, K. Fu, W. Zhang and H. Zhao	SCCMT: A Stigmergy-Based Collaborative Conceptual Modeling Tool
2016	R. Darimont, W. Zhao, C. Ponsard and A. Michot	A Modular Requirements Engineering Framework for Web-Based Toolchain Integration
2016	S. Maro and J. Steghofer	Capra: A Configurable and Extendable Traceability Management Tool
2016	X. Li, B. Upreti and Z. Zhang	Mobility Requirements Engineering Tool (MoRE)
2016	L. H. Paucar and N. Bencomo	RE-PREF: Support for REassessment of PREFerences of Non-functional Requirements for Better Decision-Making in Self-Adaptive Systems
2016	O. Akhigbe	Towards a Regulator-Oriented Regulatory Intelligence Framework
2016	P. R. Anish	Towards an Approach to Stimulate Architectural Thinking during Requirements Engineering Phase
2016	F. Fotrousi	Quality-Impact Assessment of Software Systems
2016	C. Gralha	Evaluation of Requirements Models
2016	L. Li	Exact Analysis for Next Release Problem
2016	T. R. Silva	Definition of a Behavior-Driven Model for Requirements Specification and Testing of Interactive Systems
2016	B. Wang	Requirements Traceability Technologies Selection for Industry
2016	T. Zhao	The Generation and Evolution of Adaptation Rules in Requirements Driven Self-Adaptive Systems
2015	A. Y. Mirzaee	You versus users - who owns your roadmap? (keynote)
2015	K. Holtzblatt	Can you intentionally design a product that is Cool? (keynote)
2015	M. Jackson	Requirements, behaviours, and software engineering (keynote)
2015	C. Arora, M. Sabetzadeh, A. Goknil, L. C. Briand and F. Zimmer	Change impact analysis for Natural Language requirements: An NLP approach
2015	H. Holm, T. Sommestad and J. Bengtsson	Requirements engineering: The quest for the dependent variable
2015	A. Ferrari, P. Spoletini and S. Gnesi	Ambiguity as a resource to disclose tacit knowledge
2015	A. Mahmoud	An information theoretic approach for extracting and tracing non-functional requirements

2015	S. Dey and S. Lee	From requirements elicitation to variability analysis using repertory grid: A cognitive approach
2015	F. Alabdulkareem, N. Cercone and S. Liaskos	Goal and Preference Identification through natural language
2015	R. B. Svensson and M. Taghavianfar	Selecting creativity techniques for creative requirements: An evaluation of four techniques using creativity workshops
2015	F. Sarro, A. A. Al-Subaihin, M. Harman, Y. Jia, W. Martin and Y. Zhang	Feature lifecycles as they spread, migrate, remain, and die in App Stores
2015	P. R. Anish, M. Daneva, J. Cleland-Huang, R. J. Wieringa and S. Ghaisas	What you ask is what you get: Understanding architecturally significant functional requirements
2015	M. Vierhauser, R. Rabiser, P. Grunbacher and B. Aumayr	A requirements monitoring model for systems of systems
2015	A. Cailliau and A. van Lamsweerde	Handling knowledge uncertainty in risk-based requirements engineering
2015	W. Maalej and H. Nabil	Bug report, feature request, or simply praise? On automatically classifying app reviews
2015	G. Lucassen, F. Dalpiaz, J. M. van der Werf and S. Brinkkemper	Forging high-quality User Stories: Towards a discipline for Agile Requirements
2015	D. Aceituna and H. Do	Exposing the susceptibility of off-nominal behaviors in reactive system requirements
2015	H. Hibshi, T. D. Breaux and S. B. Broomell	Assessment of risk perception in security requirements composition
2015	P. K. Murukannaiah, A. K. Kalia, P. R. Telangy and M. P. Singh	Resolving goal conflicts via argumentation-based analysis of competing hypotheses
2015	T. D. Breaux, D. Smullen and H. Hibshi	Detecting repurposing and over-collection in multi-party privacy requirements specifications
2015	P. Rempel and P. Mader	A quality model for the systematic assessment of requirements traceability
2015	D. Wuest, N. Seyff and M. Glinz	Sketching and notation creation with FlexiSketch Team: Evaluating a new means for collaborative requirements elicitation
2015	J. H. Hayes, G. Antoniol, B. Adams and Y. Gueheneuc	Inherent characteristics of traceability artifacts less is more
2015	J. Guo, N. Monaikul and J. Cleland-Huang	Trace links explained: An automated approach for generating rationales

2015	D. Ameller et al.	Handling non-functional requirements in Model-Driven Development: An ongoing industrial survey
2015	Z. J. Oster, G. R. Santhanam and S. Basu	Scalable modeling and analysis of requirements preferences: A qualitative approach using CI-Nets
2015	M. van Zee, F. Bex and S. Ghanavati	Rationalization of goal models in GRL using formal argumentation
2015	I. Hadar and A. Zamansky	Cognitive factors in inconsistency management
2015	Z. S. Abad and G. Ruhe	Using real options to manage Technical Debt in Requirements Engineering
2015	N. Bencomo	QuantUn: Quantification of uncertainty for the reassessment of requirements
2015	H. Kaindl, R. Popp and D. Raneburger	Towards reuse in safety risk analysis based on product line requirements
2015	J. Zhou, K. Hanninen, K. Lundqvist, Y. Lu, L. Provenzano and K. Forsberg	An environment-driven ontological approach to requirements elicitation for safety-critical systems
2015	P. Avesani, A. Perini, A. Siena and A. Susi	Goals at risk? Machine learning at support of early assessment
2015	T. Johann and W. Maalej	Democratic mass participation of users in Requirements Engineering?
2015	A. Mahmoud and D. Carver	Exploiting online human knowledge in Requirements Engineering
2015	M. Hosseini, A. Shahri, K. Phalp and R. Ali	Towards engineering transparency as a requirement in socio-technical systems
2015	A. Borgida, I. Jureta and A. Zamansky	Towards a general formal framework of Coherence Management in RE
2015	E. Zavala, X. Franch, J. Marco, A. Knauss and D. Damian	SACRE: A tool for dealing with uncertainty in contextual requirements at runtime
2015	M. Fockel and J. Holtmann	ReqPat: Efficient documentation of high-quality requirements using controlled natural language
2015	T. Li, E. Paja, J. Mylopoulos, J. Horkoff and K. Beckers	Holistic security requirements analysis: An attacker's perspective
2015	L. Chen, L. Huang, H. Zhong, C. Li and X. Wu	Breeze: A modeling tool for designing, analyzing, and improving software architecture
2015	I. T. Koitz and M. Glinz	StakeCloud Tool: From cloud consumers' search queries to new service requirements
2015	J. Kuriakose and J. Parsons	An enhanced requirements gathering interface for open source software development environments
2015	R. Darimont and C. Ponsard	Supporting quantitative assessment of requirements in Goal Orientation
2015	J. Pimentel, J. Vilela and J. Castro	Web tool for Goal modelling and statechart derivation

2015	M. Soltani and E. Knauss	Challenges of Requirements Engineering in AUTOSAR ecosystems
2015	M. C. Panis	Reuse of architecturally derived Standards Requirements
2015	A. Vogelsang, H. Femmer and C. Winkler	Systematic elicitation of mode models for multifunctional systems
2015	M. Holmegaard, J. B. Jorgensen, M. S. Loft and M. S. Stissing	Requirements problems in the development of a new user interface for healthcare equipment
2015	D. Callele, P. Dueck, K. Wnuk and P. Hynninen	Experience requirements in video games definition and testability
2015	M. Kapyaho and M. Kauppinen	Agile requirements engineering with prototyping: A case study
2015	J. Krisch and F. Houdek	The myth of bad passive voice and weak words an empirical investigation in the automotive industry
2015	M. R. Basirati, H. Femmer, S. Eder, M. Fritzsche and A. Widera	Understanding changes in use cases: A case study
2015	T. Morciniec and A. Podelski	Using the requirements specification to infer the implicit test status of requirements
2015	U. Eliasson, R. Heldal, E. Knauss and P. Pelliccione	The need of complementing plan-driven requirements engineering with emerging communication: Experiences from Volvo Car Group
2015	A. M. Pitangueira	Incorporating preferences from multiple stakeholders in software requirements selection an interactive search-based approach
2015	M. Warnier	How can corpus linguistics help improve requirements writing? Specifications of a space project as a case study
2015	R. H. Kegel	The Personal Information Security Assistant
2015	Y. Nugraha	Security Assurance Requirements Engineering (STARE) for trustworthy service level agreements
2015	P. Ghazi	A magnet-and-spring based visualization technique for enhancing the manipulation of requirements artifacts
2015	B. Lebeaupin	A language for writing system specifications in an aeronautical context
2015	J. Cleland-Huang, M. Rahimi and M. Mirakhorli	Ready-Set-Transfer! Technology transfer in the requirements engineering domain
2015	C. H. Duarte and T. Gorschek	Technology transfer - Requirements Engineering research to industrial practice an open (ended) debate

2014	A. I. Wasserman	Startups and requirements (keynote)
2014	A. I. Anton	Now more than ever: Privacy and security are required (keynote)
2014	M. Daneva	How practitioners approach gameplay requirements? An exploration into the context of massive multiplayer online role-playing games
2014	C. Putnam and J. Cheng	Therapist-centered requirements: A multi-method approach of requirement gathering to support rehabilitation gaming
2014	A. Alkhanifer and S. Ludi	Towards a situation awareness design to improve visually impaired orientation in unfamiliar buildings: Requirements elicitation study
2014	J. Horkoff, R. Salay, M. Chechik and A. Di Sandro	Supporting early decision-making in the presence of uncertainty
2014	A. Cailliau and A. van Lamsweerde	Integrating exception handling in goal models
2014	A. K. Chopra, F. Dalpiaz, F. B. Aydemir, P. Giorgini, J. Mylopoulos and M. P. Singh	Protos: Foundations for engineering innovative sociotechnical systems
2014	M. Adedjouma, M. Sabetzadeh and L. C. Briand	Automated detection and resolution of legal cross references: Approach and a study of Luxembourg's legislation
2014	S. Ghanavati, A. Rifaut, E. Dubois and D. Amyot	Goal-oriented compliance with multiple regulations
2014	A. K. Massey, R. L. Rutledge, A. I. Anton and P. P. Swire	Identifying and classifying ambiguity for regulatory requirements
2014	L. M. Tran and F. Massacci	An Approach for Decision Support on the Uncertainty in Feature Model Evolution
2014	S. Gartner, T. Ruhroth, J. Burger, K. Schneider and J. Jurjens	Maintaining requirements for long-living software systems by incorporating security knowledge
2014	W. Qian, X. Peng, B. Chen, J. Mylopoulos, H. Wang and W. Zhao	Rationalism with a dose of empiricism: Case-based reasoning for requirements -driven self-adaptation
2014	P. Pruski et al	TiQi: Towards natural language trace queries
2014	N. Niu, T. Bhowmik, H. Liu and Z. Niu	Traceability-enabled refactoring for managing just-in-time requirements
2014	V. Gervasi and D. Zowghi	Supporting traceability through affinity mining
2014	E. Guzman and W. Maalej	How Do Users Like This Feature? A Fine Grained Sentiment Analysis of App Reviews
2014	T. D. Breaux and F. Schaub	Scaling requirements extraction to the crowd: Experiments with privacy policies

2014	A. Sutcliffe, P. Rayson, C. N. Bull and P. Sawyer	Discovering affect-laden requirements to achieve system acceptance
2014	M. Riaz, J. King, J. Slankas and L. Williams	Hidden in plain sight: Automatically identifying security requirements from natural language artifacts
2014	R. Slavin, J. Lehker, J. Niu and T. D. Breaux	Managing security requirements patterns using feature diagram hierarchies
2014	C. Tsigkanos, L. Pasquale, C. Menghi, C. Ghezz i and B. Nuseibeh	Engineering topology aware adaptive security: Preventing requirements violations at runtime
2014	E. Knauss, D. Damian, A. Knauss and A. Borici	Openness and requirements: Opportunities and tradeoffs in software ecosystems
2014	S. Saito, M. Takeuchi, S. Yamada and M. Aoyama	RISDM: A requirements inspection systems design methodology: Perspective-based design of the pragmatic quality model and question set to SRS
2014	M. Pinto-Albuquerque and A. Rashid	Tackling the requirements jigsaw puzzle
2014	T. Bhowmik, N. Niu, A. Mahmoud and J. Savolainen	Automated support for combinational creativity in requirements engineering
2014	M. Rahimi, M. Mirakhorli and J. Cleland-Huang	Automated extraction and visualization of quality concerns from requirements specifications
2014	L. Antonelli, G. Rossi, J. C. do Prado Leite and A . Oliveros	Language Extended Lexicon points: Estimating the size of an application using its language
2014	D. G. Gordon and T. D. Breaux	The role of legal expertise in interpretation of legal requirements and definitions
2014	H. Tohonen, M. Kauppinen and T. Mannisto	Evaluating the business value of information technology: Case study on game management system
2014	F. Li et al.	Non-functional requirements as qualities, with a spice of ontology
2014	F. Fotrousi, S. A. Fricker and M. Fiedler	Quality requirements elicitation based on inquiry of quality-impact relationships
2014	S. Ingolfo, A. Siena and J. Mylopoulos	Nòmos 3: Reasoning about regulatory compliance of requirements
2014	T. H. Nguyen, J. Grundy and M. Almorsy	GUITAR: An ontology-based automated requirements analysis tool
2014	A. Gregoriades, M. Pampaka and A. Sutcliffe	Simulation-based requirements discovery for smart driver assistive technologies
2014	E. Knauss and I. Hammouda	EAM: Ecosystemability assessment method
2014	Yanji Liu, Yukun Su, Xinshang Yin and G. Mussbacher	Combined goal and feature model reasoning with the User Requirements Notation and jUCMNav
2014	S. K. Saxena and R. Chakraborty	Decisively: Application of Quantitative Analysis and Decision Science in Agile Requirements Engineering

2014	J. Badger, D. Throop and C. Claunch	VARED: Verification and analysis of requirements and early designs
2014	X. Zhu, Dongyue Mou and D. Ratiu	Structured multi-view modeling by tabular notation
2014	R. Laue, F. Hogrebe, B. Bottcher and M. Nuttgens	Efficient visual notations for efficient stakeholder communication
2014	O. Letychevskyi and T. Weigert	Symbolic verification of requirements in VRS system
2014	S. Witt, S. Feja, A. Speck and C. Hadler	Business Application Modeler: A process model Validation and Verification tool
2014	L. Bruun, M. B. Hansen, J. B. Iversen, J. B. Jorgensen and B. Knudsen	Handling design-level requirements across distributed teams: Developing a new feature for 12 Danish mobile banking apps
2014	R. Wohlrab, T. de Gooijer, A. Koziolek and S. Becker	Experience of pragmatically combining RE methods for performance requirements in industry
2014	D. Rapp, A. Hess, N. Seyff, P. Sporri, E. Fuchs and M. Glinz	Lightweight requirements engineering assessments in software projects
2014	W. Maalej and S. Ghaisas	Capturing and sharing domain knowledge with business rules lessons learned from a global software vendor
2014	C. Porter, E. Letier and M. A. Sasse	Building a National E-Service using Sentire experience report on the use of Sentire: A volere-based requirements framework driven by calibrated personas and simulated user feedback
2014	G. Valenca, C. Alves, V. Heimann, S. Jansen and S. Brinkkemper	Competition and collaboration in requirements engineering: A case study of an emerging software ecosystem
2014	A. Rubython and N. Maiden	The effect of variability modeling on requirements satisfaction for the configuration and implementation of off-the-shelf software packages
2014	C. Bomfim, W. Nunes, L. Duboc and M. Schots	Modelling sustainability in a procurement system: An experience report
2014	L. E. Martins and T. de Oliveira	A case study using a protocol to derive safety functional requirements from Fault Tree Analysis
2014	T. Stalhane and T. Wien	The DODT tool applied to sub-sea software
2014	J. Zhou, Y. Lu, K. Lundqvist, H. Lonn, D. Karlsson and B. Liwang	Towards feature-oriented requirements validation for automotive systems
2014	P. R. Anish and S. Ghaisas	Product knowledge configurator for requirements gap analysis and customizations
2014	P. Filipovikj, M. Nyberg and G. Rodriguez-Navas	Reassessing the pattern-based approach for formalizing requirements in the automotive domain

2014	F. Chen	From architecture to requirements: Relating requirements and architecture for better Requirements Engineering
2014	M. Al Hinai	Quantification of social sustainability in software
2014	M. A. Teruel	Improving collaborative and Post-WIMP systems through requirements specification
2014	T. Bhowmik	Stakeholders' social interaction in requirements engineering of open source software
2014	M. Bano	Aligning services and requirements with user feedback
2014	J. Zhou	Requirements Development and Management of Embedded Real-Time Systems
2014	D. Ionita	Context-sensitive Information security Risk identification and evaluation techniques
2014	I. Buksa	Business processes and regulations compliance management technology
2014	M. G. Ferreira	Creative Strategic Scenarios for preparation to requirements evolution
2014	J. Huffman Hayes and D. Zowghi	Ready-set-transfer! Technology transfer in the requirements engineering domain (panel)
2013	N. Maiden	Requirements engineering as information search and idea discovery (keynote)
2013	F. Cousins	Starchitects and Jack-Hammers: Requirements engineering challenges and practices in the construction industry (keynote)
2013	K. K. Breitman, R. Leite and J. Sabat	Brasilian perspectives on software production (keynote panel)
2013	A. K. Massey, J. Eisenstein, A. I. Anton and P. P. Swire	Automated text mining for requirements analysis of policy documents
2013	T. D. Breaux and A. Rao	Formal analysis of privacy requirements specifications for multi-tier applications
2013	J. C. Maxwell, A. I. Anton and J. B. Earp	An empirical investigation of software engineers' ability to classify legal cross-references
2013	A. Mahmoud and N. Niu	Supporting requirements traceability through refactoring
2013	J. Guo, J. Cleland-Huang and B. Berenbach	Foundations for an expert system in domain-specific traceability
2013	H. Sultanov and J. H. Hayes	Application of reinforcement learning to requirements engineering: requirements tracing
2013	C. Ghezzi, C. Menghi, A. M. Sharifloo and P. Spoletini	On requirements verification for model refinements

2013	I. Krka and N. Medvidovic	Distributing refinements of a system-level partial behavior model
2013	D. Dietrich and J. M. Atlee	A mode-based pattern for feature requirements, and a generic feature interface
2013	A. Sutcliffe and P. Sawyer	Requirements elicitation: Towards the unknown unknowns
2013	I. Todoran, N. Seyff and M. Glinz	How cloud providers elicit consumer requirements: An exploratory study of nineteen companies
2013	P. Caire, N. Genon, P. Heymans and D. L. Moody	Visual notation design 2.0: Towards user comprehensible requirements engineering notations
2013	D. Pagano and W. Maalej	User feedback in the appstore: An empirical study
2013	L. Shi, Q. Wang and M. Li	Learning from evolution history to predict future requirement changes
2013	D. G. Gordon and T. D. Breaux	Assessing regulatory change through legal requirements coverage modeling
2013	H. Nakagawa, A. Ohsuga and S. Honiden	A goal model elaboration for localizing changes in software evolution
2013	T. A. Alspaugh and W. Scacchi	Ongoing software development without classical requirements
2013	A. Zarghami, E. Vriezekolk, M. Z. Eslami, M. van Sinderen and R. Wieringa	Assumption-based risk identification method (ARM) in dynamic service provisioning
2013	J. Wang, J. Li, Q. Wang, D. Yang, H. Zhang and M. Li	Can requirements dependency network be used as early indicator of software integration bugs?
2013	P. Rempel, P. Mader and T. Kuschke	An empirical study on project-specific traceability strategies
2013	N. Niu, S. Reddivari and Z. Chen	Keeping requirements on track via visual analytics
2013	N. R. Mead	A history of the international requirements engineering conference (RE)RE@21
2013	S. Nair, J. L. de la Vara and S. Sen	A review of traceability research at the requirements engineering conference re@21
2013	S. J. Morris	Models in the RE seriesre@21
2013	G. Mussbacher and J. Kienzle	A vision for generic concern-oriented requirements reusere@21
2013	F. Salger	Requirements reviews revisited: Residual challenges and open research questions
2013	J. Savolainen, D. Hauksdottir and M. Mannion	Challenges in balancing the amount of solution information in requirement specifications for embedded products
2013	S. Hesari, R. Behjati and T. Yue	Towards a systematic requirement-based test generation framework: Industrial challenges and needs

2013	A. Vogelsang and S. Fuhrmann	Why feature dependencies challenge the requirements engineering of automotive systems: An empirical study
2013	N. Larburu, I. Widya, R. G. Bults, H. J. Hermens and C. Napolitano	Early phase telemedicine requirements elicitation in collaboration with medical practitioners
2013	A. Niknafs and D. M. Berry	An industrial case study of the impact of domain ignorance on the effectiveness of requirements idea generation during requirements elicitation
2013	J. Terzakis	The impact of requirements on software quality across three product generations
2013	S. Saito, M. Takeuchi, M. Hiraoka, T. Kitani and M. Aoyama	Requirements clinic: Third party inspection methodology and practice for improving the quality of software requirements specifications
2013	M. Felderer and A. Beer	Using defect taxonomies for requirements validation in industrial projects
2013	A. Muyanja, P. I. Musasizi, C. Nassimbwa, S. S. Tickodri-Togboa, E. K. Kayihura and A. Ngabirano	Requirements engineering for the uganda police force crime records management system
2013	A. Golnam, G. Regev, A. Wegmann and S. Kyriakopoulou	The integration of an RE method and AHP: A pilot study in a large Swiss bank
2013	A. Dwarakanath, R. R. Ramnani and S. Sengupta	Automatic extraction of glossary terms from natural language requirements
2013	P. Nistala and P. Kumari	An approach to carry out consistency analysis on requirements: Validating and tracking requirements through a configuration structure
2013	D. Renzel, M. Behrendt, R. Klamma and M. Jarke	Requirements Bazaar: Social requirements engineering for community-driven innovation
2013	R. Guillerm, H. Demmou and N. Sadou	A safety requirement engineering method and tool
2013	S. Teufl, D. Mou and D. Ratiu	MIRA: A tooling-framework to experiment with model-based requirements engineering
2013	C. Palomares, C. Quer and X. Franch	PABRE-Proj: Applying patterns in requirements elicitation
2013	F. Schneider, B. Bruegge and B. Berenbach	A tool implementation of the unified requirements modeling language as enterprise architect add-in
2013	L. Baresi, G. Ripa and L. Pasquale	IRET: Requirements for service platforms
2013	J. Cleland-Huang, A. Czauderna and J. H. Hayes	Using tracelab to design, execute, and baseline empirical requirements engineering experiments
2013	L. Pasquale, Y. Yu, M. Salehie, L. Cavallaro, T. T. Tun and B. Nuseibeh	Requirements-driven adaptive digital forensics

2013	C. Ncube, S. L. Lim and H. Dogan	Identifying top challenges for international research on requirements engineering for systems of systems engineering
2013	J. Cleland-Huang and S. Ghaisas	Ready-Set-Transfer: Technology transfer in the requirements engineering domain
2013	N. Maiden	Future directions of the RE conference and its community
2013	R. Wieringa, P. van Eck and J. Mylopoulos	Requirements engineering conferences: Wither industry tracks?
2013	M. Mahaux and A. Mavin	A new paradigm for applied requirements engineering research
2013	T. Gorschek	A little rebellion now and then is a good thing: Views on the requirements engineering conference
2013	M. Lencastre and J. Beatty	Top tips you can apply immediately to projects: Highlights from the RE'13 tutorials
2013	J. Beatty	Winning the hidden battle: Requirements tool selection and adoption
2013	E. Yu, D. Amyot, G. Mussbacher, X. Franch and J. Castro	Practical applications of i* in industry: The state of the art
2013	M. Glinz and R. Wieringa	RE@21 spotlight: Most influential papers from the requirements engineering conference
2013	M. Mahaux and D. Callele	Creative collisions: Meet and create: And other "RE interactive" suggestions
2013	O. Creighton and M. Borges	Workshops and doctoral symposium at RE'13: The results: Presentation of new ideas for researchers and practitioners who weren't there
2013	B. Penzenstadler, D. M. Fernandez, D. Richardson, D. Callele and K. Wnuk	The requirements engineering body of knowledge (REBoK)
2013	A. Moreira and P. Grunbacher	RE 2013 doctoral symposium
2013	D. G. Gordon	The regulatory world and the machine: Harmonizing legal requirements and the systems they affect
2013	S. Nair	Evidence management for evolutionary safety assurance and certification
2013	S. Reddivari	Visual analytics for software requirements engineering
2013	G. Valenca	Requirements negotiation model: A social oriented approach for software ecosystems evolution
2012	N. Niu and A. Mahmoud	Enhancing candidate link generation for requirements tracing: The cluster hypothesis revisited

2012	O. Gotel et al.	The quest for Ubiquity: A roadmap for software and systems traceability research
2012	E. Ben Charrada, A. Koziolek and M. Glinz	Identifying outdated requirements based on source code changes
2012	W. Bekkers, S. Brinkkemper, L. van den Bemd, F. Mijnhardt, C. Wagner and I. van de Weerd	Evaluating the Software Product Management Maturity Matrix
2012	D. Ameller, C. Ayala, J. Cabot and X. Franch	How do software architects consider non-functional requirements: An exploratory study
2012	I. Todoran	StakeCloud: Stakeholder requirements communication and resource identification in the cloud
2012	N. Riegel	Model-based prioritization in business-process-driven software development
2012	A. Knauss	On the usage of context for requirements elicitation: End-user involvement in IT ecosystems
2012	A. Hoffmann	A Pattern-based approach for analysing requirements in socio-technical systems engineering
2012	A. Alexandrova	Business requirements analysis and development for legacy system replacement projects in government organizations
2012	M. Vrbaski, D. Petriu and D. Amyot	Tool support for combined rule-based and goal-based reasoning in Context-Aware systems
2012	S. Supakkul and L. Chung	The RE-Tools: A multi-notational requirements modeling toolkit
2012	E. Paja, F. Dalpiaz, M. Poggianella, P. Roberti and P. Giorgini	STS-tool: Socio-technical Security Requirements through social commitments
2012	K. Welsh and N. Bencomo	Run-time model evaluation for requirements model-driven self-adaptation
2012	S. Reddivari, Z. Chen and N. Niu	ReCVisu: A tool for clustering-based visual exploration of requirements
2012	L. Li, H. ZHao and W. Zhang	MbFM: A matrix-based tool for modeling and configuring feature models
2012	D. Wuest, N. Seyff and M. Glinz	Flexible, lightweight requirements modeling with Flexisketch
2012	M. Smialek and T. Straszak	Facilitating transition from requirements to code with the ReDSeeDS tool
2012	W. Kong, J. H. Hayes, A. Dekhtyar and O. Dekhtyar	Process improvement for traceability: A study of human fallibility
2012	D. Amyot, S. Leblanc, J. Kealy and J. Kienzle	CoFM: An environment for collaborative feature modeling
2012	D. Ameller, O. Collell, X. Franch	ArchiTech: Tool support for NFR-guided architectural decision-making

2012	D. Hauksdottir, A. Vermehren and J. Savolainen	Requirements reuse at Danfoss
2012	O. M. Mendizabal, M. Spier and R. Saad	Log-based approach for performance requirements elicitation and prioritization
2012	D. Ott	Defects in natural language requirement specifications at Mercedes-Benz: An investigation using a combination of legacy data and expert opinion
2012	B. Berenbach, F. Schneider and H. Naughton	The use of a requirements modeling language for industrial applications
2012	X. Song and B. Hwong	Categorizing requirements for a contract-based system integration project
2012	Y. Chernak	Requirements Composition Table explained
2012	R. Tawhid et al.	Towards outcome-based regulatory compliance in aviation security
2012	A. Maglyas, U. Nikula and K. Smolander	What do practitioners mean when they talk about product management?
2012	E. Knauss, D. Damian, G. Poo-Caamano and J. Cleland-Huang	Detecting and classifying patterns of requirements clarifications
2012	A. Ghazarian	Characterization of functional software requirements space: The law of requirements taxonomic growth
2012	J. Cleland-Huang, P. Mader, M. Mirakhorli and S. Amornborvornwong	Breaking the big-bang practice of traceability: Pushing timely trace recommendations to project stakeholders
2012	S. Liaskos, R. Jalman and J. Aranda	On eliciting contribution measures in goal models
2012	S. Arora, P. Sampath and S. Ramesh	Resolving uncertainty in automotive feature interactions
2012	R. R. Lutz et al.	Requirements analysis for a product family of DNA nanodevices
2012	A. Cailliau and A. van Lamsweerde	A probabilistic framework for goal-oriented risk analysis
2012	R. Salay, M. Chechik and J. Horkoff	Managing requirements uncertainty with partial models
2012	A. Ferrari and S. Gnesi	Using collective intelligence to detect pragmatic ambiguities
2012	A. Niknafs and D. M. Berry	The impact of domain knowledge on the effectiveness of requirements idea generation during requirements elicitation
2012	A. Gross and J. Doerr	What you need is what you get!: The vision of view-based requirements specifications
2012	J. Greenyer, A. M. Sharifloo, M. Cordy and P. Heymans	Efficient consistency checking of scenario-based product-line specifications
2012	P. Shaker, J. M. Atlee and S. Wang	A feature-oriented requirements modelling language
2012	L. Yi, W. Zhang, H. Zhao, Z. Jin and H. Mei	Mining binary constraints in the construction of feature models

2012	T. T. Tun et al.	Privacy arguments: Analysing selective disclosure requirements for mobile applications
2012	L. Fu, X. Peng, Y. Yu, J. Mylopoulos and W. Zhao	Stateful requirements monitoring for self-repairing socio-technical systems
2012	H. Yang, A. De Roeck, V. Gervasi, A. Willis and B. Nuseibeh	Speculative requirements: Automatic detection of uncertainty in natural language requirements
2012	M. Salehie, L. Pasquale, I. Omoronyia, R. Ali and B. Nuseibeh	Requirements-driven adaptive security: Protecting variable assets at runtime
2012	J. C. Maxwell, A. I. Anton and P. Swire	Managing changing compliance requirements by predicting regulatory evolution
2011	M. Whalen, J. Atlee and C. Mascolo	In a ubiquitous world requirements are ubiquitous too
2011	S. Ramesh, P. Sampath and S. Arora	Evolving specifications formally
2011	I. Jureta, A. Borgida and N. A. Ernst	Finding incremental solutions for evolving requirements
2011	D. Greer and S. McGee	Software requirements change taxonomy: Evaluation by case study
2011	E. Bjarnason, K. Wnuk and B. Regnell	Requirements are slipping through the gaps – A case study on causes & effects of communication gaps in large-scale software development
2011	D. Damian and S. Marczak	How interaction between roles shapes the communication structure in requirements-driven collaboration
2011	T. Gorschek et al.	Prioritization of quality requirements: State of practice in eleven companies
2011	E. Letier and W. Heaven	Simulating and optimising design decisions in quantitative goal models
2011	E. Letier and V. Veerappa	Understanding clusters of optimal solutions in multi-objective decision problems
2011	M. Antkiewicz, K. Czarnecki and R. Rauf	Logical structure extraction from software requirements documents
2011	J. H. Hayes, Wei-Keat Kong, J. Holden, O. Dekhtyar, D. Cuddeback and A. Dekhtyar	On human analyst performance in assisted requirements tracing: Statistical analysis
2011	O. C. Gotel and S. J. Morris	Out of the labyrinth: Leveraging other disciplines for requirements traceability
2011	J. C. Trujillo, C. Cares, A. Mate and X. Franch	On the joint use of i* with other modelling frameworks: A vision paper
2011	T. Tamai, M. Hirayama, Y. Kamei and N. Ubayashi	A context analysis method for embedded systems – Exploring a requirement boundary between a system and its context
2011	A. Post, A. Podelski and J. Hoenicke	Vacuous real-time requirements
2011	K. Schneider	Focusing spontaneous feedback to support system evolution
2011	E. Knauss and O. Brill	Structured and unobtrusive observation of anonymous users and their context for requirements elicitation

2011	A. Milne and N. Maiden	Power and politics in requirements engineering: A proposed research agenda
2011	A. I. Anton, J. C. Maxwell and P. Swire	A legal cross-references taxonomy for identifying conflicting software requirements
2011	P. N. Otto, A. K. Massey, A. I. Anton and B. Smith	Assessing the accuracy of legal implementation readiness decisions
2011	Bihuan Chen, Wenyun Zhao, Xin Peng and Yijun Yu	Are your sites down? Requirements-driven self-tuning for the survivability of Web systems
2011	C. Fitzgerald, A. Finkelstein and E. Letier	Early failure prediction in feature request management systems
2011	Thein Than Tun, Yijun Yu, B. Nuseibeh, R. Wieringa and V. N. Franqueira	Risk and argument: A risk-based argumentation method for practical security
2011	J. Valkonen, T. Tommila, T. Mannisto and M. Raatikainen	Challenges of requirements engineering – A case study in nuclear energy domain
2011	P. Clements, S. Abrahao, A. Pickard and A. J. Nolan	Managing requirements uncertainty in engine control systems development
2011	B. Westphal, S. F. Arenis, D. Dietsch and A. Podelski	Disambiguation of industrial standards through formalization and graphical languages
2011	P. Grunbacher, K. Lehner, H. Vogl and A. Egyed	Reconciling requirements and architectures with the CBSP approach in an iPhone app project
2011	K. Ohashi, Y. Tananaka, R. Yamamoto and H. Kurihara	A means of establishing traceability based on a UML model in business application development
2011	X. Franch and J. P. Carvallo	Requirements negotiation for multilayer system components
2011	A. Wever and N. Maiden	What are the day-to-day factors that are preventing business analysts from effective business analysis?
2011	B. Waldmann	There's never enough time: Doing requirements under resource constraints, and what requirements engineering can learn from agile development
2011	R. Fahney and S. Puhl	How to assign cost to "avoidable requirements creep": A step towards the waterfall's agilization
2011	E. Boutkova and F. Houdek	Semi-automatic identification of features in requirement specifications
2011	A. Sutcliffe	Emotional requirements engineering
2011	B. Regnell	Large-scale feature evolution: Problems and solutions from the mobile domain

2011	J. Cleland-Huang and D. Damian	Ready-set-transfer! Technology transfer in the requirements engineering domain
2011	V. Gervasi, P. Sawyer and B. Nuseibeh	Unknown knowns: Tacit knowledge in requirements engineering
2011	M. Jarke	Design requirements: Challenges and approaches
2011	Nan Niu and A. Mahmoud	TraCter: A tool for candidate traceability link clustering
2011	Bo Wei, D. Zowghi, Zhi Jin and Bin Yin	rS: Automated reasoning tool for non-functional requirement goal models
2011	S. Faily and I. Flechais	Eliciting usable security requirements with misusability cases
2011	X. Franch, C. Palomares and C. Quer	PABRE-Man: Management of a requirement patterns catalogue
2011	J. Noppen, A. Rashid, R. Chitchyan, P. Greenwood, A. Solberg and F. Fleurey	Modelling adaptability and variability in requirements
2011	M. Khalil, S. Teufl, E. Geisberger and Dongyue Mou	Experience with content-based requirements engineering assessments
2011	N. Seyff, M. Bortenschlager and G. Ollmann	iRequire: Gathering end-user requirements for new apps
2011	H. Schmidt, M. Heisel, I. Cote and D. Hatebur	UML4PF – A tool for problem-oriented requirements analysis
2011	B. Nuseibeh, A. Tedeschi, Yijun Yu, V. N. Franqueira and Thein Than Tun	OpenArgue: Supporting argumentation to evolve secure software systems
2011	S. Liaskos and V. Rogozhkin	Applying preference-based customization
2011	N. Bencomo, K. Welsh and P. Sawyer	Run-time resolution of uncertainty
2010	V. Gervasi, R. Gacitua and P. Sawyer	On the Effectiveness of Abstraction Identification in Requirements Engineering
2010	I. Menzel, M. Mueller, J. Doerr and A. Gross	An Experimental Comparison Regarding the Completeness of Functional Requirements Specifications
2010	A. Willis, A. de Roeck, B. Nuseibeh, V. Gervasi and H. Yang	Extending Nocuous Ambiguity Analysis for Anaphora in Natural Language Requirements
2010	N. Maiden, N. Seyff and F. Graf	Using Mobile RE Tools to Give End-Users Their Own Voice
2010	A. I. Antón and J. D. Young	A Method for Identifying Software Requirements Based on Policy Commitments
2010	A. Milne, N. Maiden, S. Jones, K. Karlsen, K. Zachos and R. Neill	Requirements Engineering as Creative Problem Solving: A Research Agenda for Idea Finding
2010	L. Bass and P. Clements	Using Business Goals to Inform a Software Architecture
2010	J. Geisler, O. Sudmann, C. Henke, N. H. Madhavji, R. Ferrari and W. Schafer	Requirements Engineering Decisions in the Context of an Existing Architecture: A Case Study of a Prototypical Project

2010	D. C. Gause, S. Jantunen and R. Wessman	Making Sense of Product Requirements
2010	J. Whittle, A. Finkelstein, P. Sawyer, E. Letier and N. Bencomo	Requirements-Aware Systems: A Research Agenda for RE for Self-adaptive Systems
2010	X. Peng, B. Chen, W. Zhao and Y. Yu	Self-Tuning of Software Systems Through Goal-based Feedback Loop Control
2010	N. A. Qureshi and A. Perini	Requirements Engineering for Adaptive Service Based Applications
2010	A. Borgida, J. Mylopoulos, I. J. Jureta and N. A. Ernst	Techne: Towards a New Generation of Requirements Modeling Languages with Goals, Preferences, and Inconsistency Handling
2010	L. Pasquale, P. Spoletini and L. Baresi	Fuzzy Goals for Requirements-Driven Adaptation
2010	S. Liaskos, J. Mylopoulos, S. Sohrabi and S. A. McIlraith	Integrating Preferences into Goal Models for Requirements Engineering
2010	M. Daneva, A. Herrmann, R. Wieringa, Z. Racheva and K. Sikkel	Do We Know Enough about Requirements Prioritization in Agile Projects: Insights from a Case Study
2010	D. Pfahl, G. Ruhe and A. Al-Emran	Decision Support for Product Release Planning Based on Robustness Analysis
2010	S. Fricker and M. Glinz	Comparison of Requirements Hand-off, Analysis, and Negotiation: Case Study
2010	S. Supakkul, T. Hill, T. T. Tun, J. C. do Prado Leite and L. Chung	An NFR Pattern Approach to Dealing with NFRs
2010	A. Morali and R. Wieringa	Risk-based Confidentiality Requirements Specification for Outsourced IT Systems
2010	H. Sultanov and J. H. Hayes	Application of Swarm Techniques to Requirements Engineering: Requirements Tracing
2010	A. Egyed, F. Graf and P. Grünbacher	Effort and Quality of Recovering Requirements-to-Code Traces: Two Exploratory Experiments
2010	A. Dekhtyar, J. Hayes and D. Cuddeback	Automated Requirements Traceability: The Study of Human Analysts
2010	R. Mazo, O. Djebbi, D. Diaz and C. Salinesi	Using Integer Constraint Solving in Reuse Based Requirements Engineering
2010	A. Mashkoor and J. Jacquot	Domain Engineering with Event-B: Some Lessons We Learned
2010	E. Kang and D. Jackson	Dependability Arguments with Trusted Bases
2010	P. Wilkinson and A. Mavin	Big Ears (The Return of "Easy Approach to Requirements Engineering")
2010	J. Talvio, M. Kauppinen, M. Komssi, M. Pyhäjärvi and T. Männistö	Persuading Software Development Teams to Document Inspections: Success Factors and Challenges in Practice
2010	J. Savolainen, A. Vilavaara and J. Kuusela	Transition to Agile Development - Rediscovery of Important Requirements Engineering Practices

2010	J. Hill and S. Tilley	Creating Safety Requirements Traceability for Assuring and Recertifying Legacy Safety-Critical Systems
2010	M. C. Panis	Successful Deployment of Requirements Traceability in a Commercial Engineering Organization...Really
2010	K. Czarnecki, D. M. Berry, M. AbdElRazik and M. Antkiewicz	Requirements Determination is Unstoppable: An Experience Report
2010	F. Peng, L. Liu and T. Li	Why Requirements Engineering Fails: A Survey Report from China
2010	M. Reiser, H. Glathe, M. Bittner and M. Weber	Manufacturer-Supplier Requirements Synchronization Using Exchange Containers and Multi-Level Systems
2010	M. C. Primrose	User Experience Grading Via Kano Categories
2010	N. Y. David and D. K. Vijayamma	Enhancing Customer Partnership Through Requirements Framework
2010	S. Siegl, R. German and K. Hielscher	Model Based Requirements Analysis and Testing of Automotive Systems with Timed Usage Models
2010	D. Bush, M. Everdij, J. Lockerbie, H. Blom and N. Maiden	Using i* Modelling as a Bridge between Air Traffic Management Operational Concepts and Agent-based Simulation Analysis
2010	A. McLucas, S. El Sawah and M. Ryan	Using Cognitive Mapping to Elicit Modelling Requirements: An Overview
2010	A. Sinha, H. Takeuchi, A. Paradkar and T. Nakamura	Extending Automated Analysis of Natural Language Use Cases to Other Languages
2010	B. Berenbach	Requirements Engineering for Industrial Systems: No Easy Answers
2010	N. Maiden, M. Mahaux and P. Heymans	Making It all Up: Getting in on the Act to Improvise Creative Requirements
2010	V. Gervasi	Who Framed Roger User? Problem Frames as a User Interaction Design Tool
2010	M. Hirzalla	Realizing Business Agility Requirements through SOA and Cloud Computing
2010	S. Saito, T. Nakatani and M. Aoyama	REBOK Manifest: Towards a Requirements Engineering Body of Knowledge
2010	M. Glinz	Very Lightweight Requirements Modeling
2010	J. Beatty	My Company Can't Afford RE Training Right Now, What Do We Do?
2010	M. Glinz, K. Ryan and N. Maiden	If You Want Innovative RE, Never Ask the Users; A Formal Debate
2010	L. Kof	From Requirements Documents to System Models: A Tool for Interactive Semi-Automatic Translation
2010	J. Hosking, J. Grundy and M. Kamalrudin	MaramaAI: Automated and Visual Approach for Inconsistency Checking of Requirements

2010	K. Schneider, D. Callele and E. Neufeld	An Introduction to Experience Requirements
2010	P. Grünbacher, N. Seyff and F. Graf	Using Contextual Information to Guide on-site Analysts
2010	M. Smialek, H. Kaindl and W. Nowakowski	Case-based Reuse with Partial Requirements Specifications
2010	B. Yin, X. Chen and Z. Jin	Dptool: A Tool for Supporting the Problem Description and Projection
2010	M. Jehle, S. Fricker, M. Glinz and R. Stoiber	Feature Unweaving: Refactoring Software Requirements Specifications into Software Product Lines
2010	P. Patrício, V. Amaral, J. Araújo and R. Monteiro	Mdgore: Towards Model-Driven and Goal-Oriented Requirements Engineering
2010		Elicitation of Dependability Requirements: A HAZOP-based Approach
2010	D. Callele, K. Wnuk and B. Regnell	Guiding Requirements Scoping Using ROI: Towards Agility, Openness and Waste Reduction
2010	C. Stanbridge	Retrospective Requirement Analysis Using Code Coverage of GUI Driven System Tests
2010	J. Beatty	If You Build It, Will They Use It? Leveraging Business Objectives to Deliver Successful Projects

Appendix C

Final list of literature from the SLR

#	Reference	Title
1	Alabdulkareem, Cercone, and Liaskos (2015b)	Goal and Preference Identification through natural language
2	Al-Hroob, Imam, and Al-Heisa (2018)	The use of artificial neural networks for extracting actions and actors from requirements document
3	Alkhader, Hudaib, and Hammo (2006)	Experimenting with extracting software requirements using NLP approach
4	Al-Sabbagh, Girju, and Diesner (2015)	A Unified Framework to Identify and Extract Uncertainty Cues, Holders, and Scopes in One Fell-Swoop
5	Annervaz et al. (2013)	Natural language requirements quality analysis based on business domain models
6	Arora et al. (2013)	Automatic checking of conformance to requirement boilerplates via text chunking: An industrial case study
7	Arora et al. (2015a)	Automated checking of conformance to requirements templates using natural language processing
8	Arora et al. (2015c)	Change Impact Analysis for Natural Lauguage Requirements: An NLP Approach
9	Asad Naqvi et al. (2010)	Cross-document dependency analysis for system-of-system integration
10	Asano, Hayashi, and Saeki (2017)	Detecting bad smells of refinement in goal-oriented requirements analysis
11	Aysolmaz et al. (2018)	A semi-automated approach for generating natural language requirements documents based on business process models
12	Bajwa et al. (2012)	On a chain of transformations for generating alloy from NL constraints
13	Bajwa, Lee, and Bordbar (2012)	Resolving syntactic ambiguities in natural language specification of constraints
14	Bakiu and Guzman (2017)	Which feature is unusable? Detecting usability and user experience issues from user reviews
15	Belsis, Koutoumanos, and Sgouropoulou (2014)	PBURC: A patterns-based, unsupervised requirements clustering framework for distributed agile software development
16	Ben Abdessalem Karaa et al. (2015)	Automatic builder of class diagram (ABCD): an application of UML generation from functional requirements
17	Bhat, Ye, and Jacobsen (2014)	Orchestrating SOA using requirement specifications and domain ontologies
18	Bhowmik et al. (2015)	Leveraging topic modeling and part-of-speech tagging to support combinational creativity in requirements engineering

#	Reference	Title
19	Biébow and Szulman (1993)	Acquisition and validation: from text to semantic network
20	Boddu et al. (2004)	RETNA: From requirements to testing in a natural way
21	Bolloju and Schneider (2011)	Facilitating the Creation of Quality Object Models Using a Knowledge Based System
22	Bolloju, Schneider, and Sugumaran (2012)	A knowledge-based system for improving the consistency between object models and use case narratives
23	Ciemniewska et al. (2007)	Supporting Use-case Reviews
24	Cleland-Huang et al. (2007b)	Automated classification of non-functional requirements
25	Cruz et al. (2017a)	Detecting Vague Words & Phrases in Requirements Documents in a Multilingual Environment
26	Dalpiaz, van der Schalk, and Lucassen (2018)	Pinpointing ambiguity and incompleteness in requirements engineering via information visualization and NLP
27	Daramola et al. (2011)	Enabling hazard identification from requirements and reuse-oriented HAZOP analysis
28	Daramola, Sindre, and Moser (2012)	Ontology-based support for security requirements specification process
29	Deeptimahanti and Sanyal (2009)	An Innovative Approach for Generating Static UML Models from Natural Language Requirements
30	Deocadez, Harrison, and Rodriguez (2017)	Automatically classifying requirements from app stores: A preliminary study
31	Díaz et al. (2005a)	Integrating Natural Language Techniques in OO-Method
32	Drazan and Mencl (2007)	Improved processing of textual use cases: Deriving behavior specifications
33	Duan and Cleland-Huang (2007)	A Clustering Technique for Early Detection of Dominant and Recessive Cross-Cutting Concerns
34	Duan et al. (2009)	Towards automated requirements prioritization and triage
35	Duan et al. (2015)	User-constrained clustering in online requirements forums
36	Dwarakanath, Ramnani, and Sengupta (2013a)	Automatic extraction of glossary terms from natural language requirements
37	Elallaoui, Nafil, and Touahni (2018)	Automatic Transformation of User Stories into UML Use Case Diagrams using NLP Techniques
38	Femmer et al. (2017)	Rapid quality assurance with Requirements Smells

#	Reference	Title
39	Ferrari et al. (2014b)	Measuring and improving the completeness of natural language requirements
40	Garzoli et al. (2013)	Robust Requirements Analysis in Complex Systems through Machine Learning
41	Génova et al. (2013)	A framework to measure and improve the quality of textual requirements
42	Ghosh et al. (2016)	ARSENAL: Automatic requirements specification extraction from natural language
43	Giganto and Smith (2008)	Derivation of classes from use cases automatically generated by a three-level sentence processing algorithm
44	Gleich, Creighton, and Kof (2010)	Ambiguity detection: Towards a tool explaining ambiguity sources
45	Gokyer et al. (2008)	Non-functional requirements to architectural concerns: MML and NLP at crossroads
46	Goldin and Berry (1997)	AbStFinder, a Prototype Natural Language Text Abstraction Finder for Use in Requirements Elicitation
47	Gorse et al. (2004)	Mixing linguistic and formal techniques for high-level requirements engineering
48	Gulia and Choudhury (2016)	An efficient automated design to generate UML diagram from Natural Language Specifications
49	Guzman and Maalej (2014a)	How do users like this feature? A fine grained sentiment analysis of App reviews
50	Harksoo Kim et al. (1999)	Informal requirements analysis supporting system for human engineer
51	Harmain and Gaizauskas (2003)	CM-Builder: A natural language-based CASE tool for object-oriented analysis
52	Hussain, Kosseim, and Ormandjieva (2008)	Using linguistic knowledge to classify non-functional requirements in SRS documents
53	Hussain, Ormandjieva, and Kosseim (2012)	LASR: A tool for large scale annotation of software requirements
54	Ibrahim and Ahmad (2010b)	Class diagram extraction from textual requirements using natural language processing (NLP) techniques
55	Jha and Mahmoud (2017)	Mining user requirements from application store reviews using frame semantics
56	Jha and Mahmoud (2018)	Using frame semantics for classifying and summarizing application store reviews

#	Reference	Title
57	Jin et al. (2010)	A concern-based approach to generating formal requirements specifications
58	Jurkiewicz and Nawrocki (2015)	Automated events identification in use cases
59	Kamalrudin, Hosking, and Grundy (2011)	Improving requirements quality using essential use case interaction patterns
60	Kiyavitskaya et al. (2004)	Experimenting with linguistic tools for conceptual modelling: Quality of the models and critical features
61	Kiyavitskaya et al. (2008)	Requirements for tools for ambiguity identification and measurement in natural language requirements specifications
62	Kiyavitskaya and Zannone (2008)	Requirements model generation to support requirements elicitation: The Secure Tropos experience
63	Knauss et al. (2015)	Patterns of continuous requirements clarification
64	Ko et al. (2007b)	Using classification techniques for informal requirements in the requirements analysis-supporting system
65	Ko, Kim, and Park (2018)	Automatic recommendation to omitted steps in use case specification
66	Körner and Brumm (2009)	RESI - A natural language specification improver
67	Körner, Landhäußer, and Tichy (2014)	Transferring research into the real world: How to improve RE with AI in the automotive industry
68	Ländhäußer et al. (2015)	DeNom: A tool to find problematic nominalizations using NLP
69	Laue, Koop, and Gruhn (2016)	Indicators for open issues in business process models
70	Li et al. (2015)	Automated requirements extraction for scientific software
71	Lili et al. (2010)	Research on User Requirements Elicitation Using Text Association Rule
72	Liu, Li, and Kou (2014)	Eliciting relations from natural language requirements documents based on linguistic and statistical analysis
73	López-Martínez et al. (2017)	User stories complexity estimation using Bayesian networks for inexperienced developers
74	Loughran, Sampaio, and Rashid (2006)	From requirements documents to feature models for aspect oriented product line implementation
75	Lucassen et al. (2015a)	Forging high-quality User Stories: Towards a discipline for Agile Requirements
76	Lucassen et al. (2016a)	Improving agile requirements: the Quality User Story framework and tool

#	Reference	Title
77	Lucassen et al. (2016b)	Visualizing user story requirements at multiple granularity levels via semantic relatedness
78	Lucassen et al. (2017)	Extracting conceptual models from user stories with Visual Narrator
79	Maalej et al. (2016)	On the automatic classification of app reviews
80	Mahmoud and Williams (2016)	Detecting, classifying, and tracing non-functional software requirements
81	Mala and Uma (2006)	Automatic Construction of Object Oriented Design Models [UML Diagrams] from Natural Language Requirements Specification
82	Mala and Uma (2006)	Elicitation of Non-functional Requirement Preference for Actors of Usecase from Domain Model
83	Manrique-Losada, Zapata-Jaramillo, and Burgos (2016)	Re-expressing business processes information from corporate documents into controlled language
84	Matsumoto, Shirai, and Ohnishi (2017)	A Method for Verifying Non-Functional Requirements
85	McZara et al. (2015)	Software requirements prioritization and selection using linguistic tools and constraint solvers—a controlled experiment
86	Mefteh, Bouassida, and Ben-Abdallah (2018)	Towards naturalistic programming: Mapping language-independent requirements to constrained language specifications
87	Meth, Maedche, and Einoeder (2013)	Is knowledge power? The role of knowledge in automated requirements elicitation
88	Mezghani, Kang, and Sèdes (2018)	Using k-Means for Redundancy and Inconsistency Detection: Application to Industrial Requirements
89	Meziane, Athanasakis, and Ananiadou (2008)	Generating Natural Language specifications from UML class diagrams
90	Minhas et al. (2011)	Controlled vocabulary based software requirements classification
91	Misra and Das (2013)	Entity disambiguation in natural language text requirements
92	Misra (2016)	Terminological inconsistency analysis of natural language requirements
93	Misra, Sengupta, and Podder (2016)	Topic cohesion preserving requirements clustering
94	Mitchell, Ferguson, and Parrington (1995)	Rapid prototyping: An integrated CASE based approach
95	Morales-Ramirez, Perini, and Ceccato (2015)	Towards supporting the analysis of online discussions in OSS communities: A speech-act based approach

#	Reference	Title
96	Narouei and Takabi (2015)	Automatic top-down role engineering framework using natural language processing techniques
97	Ninaus et al. (2014)	Content-based recommendation techniques for requirements engineering
98	Ochodek and Nawrocki (2008)	Automatic transactions identification in use cases
99	Osborne and MacNish (1996)	Processing natural language software requirement specifications
100	Ott (2013)	Automatic requirement categorization of large natural language specifications at Mercedes-Benz for review improvements
101	Park et al. (2000)	Implementation of an efficient requirements-analysis supporting system using similarity measure techniques
102	Park, An, and Lee (2010)	Anaphora resolution system for natural language requirements document in Korean
103	Peng et al. (2017)	An approach of extracting feature requests from app reviews
104	Pinquié et al. (2016)	Natural Language Processing of Requirements for Model-Based Product Design with ENOVIA/CATIA V6
105	Quirchmayr et al. (2018)	Semi-automatic rule-based domain terminology and software feature-relevant information extraction from natural language user manuals: An approach and evaluation at Roche Diagnostics GmbH
106	Rago, Marcos, and Diaz-Pace (2013)	Uncovering quality-attribute concerns in use case specifications via early aspect mining
107	Rago, Marcos, and Diaz-Pace (2016a)	Assisting requirements analysts to find latent concerns with REAssistant
108	Rago, Marcos, and Diaz-Pace (2016b)	Identifying duplicate functionality in textual use cases by aligning semantic actions
109	Rashwan (2012)	Semantic Analysis of Functional and Non-functional Requirements in Software Requirements Specifications
110	Riaz et al. (2014b)	Hidden in plain sight: Automatically identifying security requirements from natural language artifacts
111	Robeer et al. (2016a)	Automated Extraction of Conceptual Models from User Stories via NLP

#	Reference	Title
112	Saint-Dizier (2018)	Mining incoherent requirements in technical specifications: Analysis and implementation
113	Salemi, Selamat, and Penhaker (2016)	A model transformation framework to increase OCL usability
114	Sannier and Baudry (2014)	INCREMENT: A Mixed MDE-IR Approach for Regulatory Requirements Modeling and Analysis
115	Sannier et al. (2016)	Automated classification of legal cross references based on semantic intent
116	Sannier et al. (2017a)	An automated framework for detection and resolution of cross references in legal texts
117	Santos et al. (2008)	Generating requirements analysis models from textual requirements
118	Sardinha et al. (2013)	EA-Analyzer: Automating conflict detection in a large set of textual aspect-oriented requirements
119	Sateli, Angius, and Witte (2013a)	The ReqWiki Approach for Collaborative Software Requirements Engineering with Integrated Text Analysis Support
120	Sawyer, Gacitua, and Stone (2008)	Profiling and tracing stakeholder needs
121	Sharma, Gulia, and Biswas (2014)	Automated Generation of Activity and Sequence Diagrams from Natural Language Requirements
122	Sharma, Srivastava, and Biswas (2015)	From natural language requirements to UML class diagrams
123	Sharma, Sharma, and Biswas (2017)	Machine Learning for Detecting Pronominal Anaphora Ambiguity in NL Requirements
124	Shibaoka, Kaiya, and Saeki (2007)	GOORE : Goal-Oriented and Ontology Driven Requirements Elicitation Method
125	Sillaber and Breu (2014)	Improving Near-Duplicate Detection in Multi-Layered Collaborative Requirements Engineering Discussions Through Discussion Clustering
126	Sinha et al. (2010a)	Extending Automated Analysis of Natural Language Use Cases to Other Languages
127	Soni and Gaur (2016)	A methodological approach to identify type of dependency from user requirements
128	Stålhane and Wien (2014)	The DODT tool applied to sub-sea software

#	Reference	Title
129	Subha and Palaniswami (2013)	Quality factor assessment and text summarization of unambiguous natural language requirements
130	Sun and Peng (2015)	A scenario model aggregation approach for mobile app requirements evolution based on user comments
131	Terawaki (2011)	Supporting of requirements elicitation for ensuring services of information systems used for education
132	Terawaki (2013)	Framework for quantitatively evaluating the quality requirements of software system
133	Thovex and Trichet (2014)	Automatic Building of Socio-semantic Networks for Requirements Analysis
134	Tjong and Berry (2013)	The design of SREE - A prototype potential ambiguity finder for requirements specifications and lessons learned
135	Umber, Bajwa, and Naeem (2011)	NL-Based Automated Software Requirements Elicitation and Specification
136	Veerappa and Harrison (2013)	Assessing the maturity of requirements through argumentation: A good enough approach
137	Vidya Sagar and Abirami (2014)	Conceptual modeling of natural language functional requirements
138	Winkler and Vogelsang (2017)	Automatic classification of requirements based on convolutional neural networks
139	Winter et al. (2017)	Characterizing regulatory documents and guidelines based on text mining
140	Yang et al. (2011)	Analysing anaphoric ambiguity in natural language requirements
141	Yang et al. (2012b)	Speculative requirements: Automatic detection of uncertainty in natural language requirements
142	Zachos et al. (2007)	Discovering web services to specify more complete system requirements
143	Zeni, Mich, and Mylopoulos (2016)	GaiusT 2.0: Evolution of a framework for annotating legal documents
144	Zhang and Lee (2010)	Complementary classification techniques based personalized software requirements retrieval with semantic ontology and user feedback

Appendix D

List of articles removed during quality assessment

#	Reference	Title
1	Aceituna et al. (2014)	Model-based requirements verification method: Conclusions from two controlled experiments
2	Ali and Kasirun (2008)	Developing tool for crosscutting concern identification using nlp
3	Ambriola and Gervasi (1997)	Processing natural language requirements
4	Ambriola and Gervasi (2006)	On the systematic analysis of natural language requirements with c irce
5	Bagheri, Ensan, and Gasevic (2012)	Decision support for the software product line domain engineering lifecycle
6	Bäumer and Geierhos (2016)	Running Out of Words: How Similar User Stories Can Help to Elaborate Individual Natural Language Requirement Descriptions
7	Bergmann and Cunningham (2002)	Acquiring Customers' Requirementsin Electronic Commerce
8	Bhatia et al. (2013)	Using grammatical knowledge patterns for structuring requirements specifications
9	Bhatia et al. (2016b)	Automated extraction of regulated information types using hyponymy relations
10	Bokaei Hosseini, Breaux, and Niu (2018)	Inferring Ontology Fragments from Semantic Role Typing of Lexical Variants
11	Böttger et al. (2003)	Towards reconciling use cases via controlled language and graphical models
12	Boutkova and Houdek (2011b)	Semi-automatic identification of features in requirement specifications
13	Bowden, Hargreaves, and Langensiepen (2000)	Estimation support by lexical analysis of requirements documents
14	Boyd, Zowghi, and Gervasi (2007)	Optimal-constraint lexicons for requirements specifications
15	Breaux, Hibshi, and Rao (2014)	Eddy, a formal language for specifying and analyzing data flow specifications for conflicting privacy requirements
16	Carlson and Laplante (2014)	The NASA automated requirements measurement tool: A reconstruction
17	Casagrande et al. (2014b)	NLP-KAOS for systems goal elicitation: Smart metering system case study
18	Casamayor, Godoy, and Campo (2010)	Identification of non-functional requirements in textual specifications: A semi-supervised learning approach
19	Casamayor, Godoy, and Campo (2012)	Functional grouping of natural language requirements for assistance in architectural software design

#	Reference	Title
20	Cascini, Fantechi, and Spinicci (2004)	Natural language processing of patents and technical documentation
21	Chen et al. (2010)	Text-based requirements preprocessing using nature language processing techniques
22	Cleland-Huang et al. (2006)	The detection and classification of non-functional requirements with application to early aspects
23	Silvat and Carvalho (2006)	A knowledge representation semantic network for a natural language syntactic analyzer based on the UML
24	del Socorro Bernardos Galindo and Cea (2001)	A New Approach in Building a Corpus for Natural Language Generation Systems
25	Díaz et al. (2005b)	Interaction transformation patterns based on semantic roles
26	Driss et al. (2010)	A requirement-centric approach to web service modeling, discovery, and selection
27	Evans et al. (2017)	An Evaluation of Constituency-Based Hyponymy Extraction from Privacy Policies
28	Farfeleder et al. (2011)	Ontology-driven guidance for requirements elicitation
29	Fatwanto (2013)	Natural language requirements specification analysis using Part-of-Speech Tagging
30	Ferrari and Gnesi (2012b)	Using collective intelligence to detect pragmatic ambiguities
31	Ferrari et al. (2014a)	From commercial documents to system requirements: an approach for the engineering of novel CBTC solutions
32	Ferrari et al. (2014c)	Pragmatic ambiguity detection in natural language requirements
33	Ferrari, Donati, and Gnesi (2017)	Detecting domain-specific ambiguities: An NLP approach based on wikipedia crawling and word embeddings
34	Ferrari et al. (2018)	Detecting requirements defects with NLP patterns: an industrial experience in the railway domain
35	Ferreira and Silva (2009)	A Controlled Natural Language Approach for Integrating Requirements and Model-Driven Engineering
36	Fliedl et al. (2000)	Linguistically based requirements engineering – The NIBA-project

#	Reference	Title
37	Fliedl et al. (2004)	Semantic Tagging and Chunk-Parsing in Dynamic Modeling
38	Fliedl, Kop, and Mayr (2005)	From textual scenarios to a conceptual schema
39	Fliedl et al. (2007)	Deriving static and dynamic concepts from software requirements using sophisticated tagging
40	Fu, Bastani, and Yen (2008)	Model-driven prototyping based requirements elicitation
41	Gärtner et al. (2014)	Maintaining requirements for long-living software systems by incorporating security knowledge
42	Geierhos and Bäumer (2016)	How to complete customer requirements: Using concept expansion for requirement refinement
43	Glava??, Fertalj, and ??najder (2012)	From requirements to code: Syntax-based requirements analysis for data-driven application development
44	Gonzalez Moreno and Vazquez Lopez (2009)	Using Techniques Based on Natural Language in the Development Process of Multiagent Systems
45	Gulla, Vos, and Thiel (1997)	An abductive, linguistic approach to model retrieval
46	Guzman, Ibrahim, and Glinz (2017b)	A Little Bird Told Me: Mining Tweets for Requirements and Software Evolution
47	Hajri et al. (2015)	Applying product line Use case modeling in an industrial automotive embedded system: Lessons learned and a refined approach
48	Halkidis, Chatzigeorgiou, and Stephanides (2009)	Moving from Requirements to Design Confronting Security Issues: A Case Study
49	Hartong, Goel, and Wijesekera (2014)	Security requirement derivation by noun-verb analysis of use-misuse case relationships: A case study using positive train control
50	He et al. (2009)	Predicting upgrade project defects based on enhancement requirements: An empirical study
51	He, Wang, and Liang (2010)	Semantic interoperability aggregation in service requirements refinement
52	Hosseini et al. (2017)	CRAFT: A crowd-annotated feedback technique
53	Hu et al. (2014)	Supporting the elicitation of requirements compliant with regulations
54	Ilyas and Küng (2009)	A similarity measurement framework for requirements engineering

#	Reference	Title
55	Ilyas and Küng (2010)	Ontology-based similarity measurement in software projects through SimReq framework
56	Jiang, Ruan, and Zhang (2014)	Analysis of Economic Impact of Online Reviews: An Approach for Market-Driven Requirements Evolution
57	Jin et al. (2015)	Translating online customer opinions into engineering characteristics in QFD: A probabilistic language analysis approach
58	Jin, Ji, and Gu (2016)	Identifying comparative customer requirements from product online reviews for competitor analysis
59	Flores (2004)	Semantic Filtering of Textual Requirements Descriptions
60	Kacfah Emani (2014)	Automatic detection and semantic formalisation of business rules
61	Kaiya and Saeki (2005)	Ontology Based Requirements Analysis: Lightweight Semantic Processing Approach
62	Kaiya and Saeki (2006)	Using Domain Ontology as Domain Knowledge for Requirements Elicitation
63	Kamalrudin, Grundy, and Hosking (2012)	Supporting requirements modelling in the Malay language using essential use cases
64	Kanchev et al. (2017b)	Canary: An Interactive and Query-Based Approach to Extract Requirements from Online Forums
65	Kang and Saint-Dizier (2014)	Requirement compound mining and analysis
66	Kim, Park, and Sugumaran (2004)	A Linguistics-Based Approach for Use Case Driven Analysis Using Goal and Scenario Authoring
67	Kim, Park, and Sugumaran (2006)	Improving use case driven analysis using goal and scenario authoring: A linguistics-based approach
68	Kirchsteiger et al. (2008)	Specification-based Verification of Embedded Systems by Automated Test Case Generation
69	Knauss et al. (2011)	Supporting requirements engineers in recognising security issues
70	Kof (2007a)	Scenarios: Identifying missing objects and actions by means of computational linguistics
71	Kof (2007b)	Treatment of Passive Voice and Conjunctions in Use Case Documents

#	Reference	Title
72	Kof (2008)	On the identification of goals in stakeholders' dialogs
73	Kof (2009)	Translation of textual specifications to automata by means of discourse context modeling
74	Kof and Penzenstadler (2011)	From requirements to models: Feedback generation as a result of formalization
75	Kop and Mayr (1998)	Conceptual predesign bridging the gap between requirements and conceptual design
76	Körner and Landhäußer (2010)	Semantic enriching of natural language texts with automatic thematic role annotation
77	Kroha (2000)	Preprocessing of requirements specification
78	Lee and Bryant (2004)	Automation of Software System Development Using Natural Language Processing and Two-Level Grammar
79	Li et al. (2011)	An engineerable ontology based approach for requirements elicitation in process centered problem domain
80	Li, Zhang, and Wang (2018)	Automatic user preferences elicitation: A data-driven approach
81	Li et al. (2018)	Automatically classifying user requests in crowdsourcing requirements engineering
82	Lian, Cleland-Huang, and Zhang (2017)	Mining Associations between Quality Concerns and Functional Requirements
83	Liang and Palmer (1994)	A pattern matching and clustering based approach for supporting requirements transformation
84	Liu et al. (2004)	Natural language requirements analysis and class model generation using UCDA
85	Maalej and Nabil (2015b)	Bug report, feature request, or simply praise? On automatically classifying app reviews
86	Madala et al. (2017)	Automated identification of component state transition model elements from requirements
87	Madhan, Kalaiselvi, and Donald J.P (2017)	Tool development for formalizing the requirement for the safety critical software engineering process

#	Reference	Title
88	Mahmud, Seceleanu, and Ljungkrantz (2017)	Specification and semantic analysis of embedded systems requirements: From description logic to temporal logic
89	Martin-Rodilla and Gonzalez-Perez (2014)	An ISO/IEC 24744-derived modelling language for discourse analysis
90	Massey et al. (2013)	Automated text mining for requirements analysis of policy documents
91	Mazón et al. (2007)	Improving the development of data warehouses by enriching dimension hierarchies with WordNet
92	Mich and Garigliano (1994)	A linguistic approach to the development of Object Oriented Systems using the NL system LOLITA
93	Montgomery et al. (2018)	Customer support ticket escalation prediction using feature engineering
94	Morales-Ramirez, Kifetew, and Perini (2017)	Analysis of online discussions in support of requirements discovery
95	Nassar and Khamayseh (2015)	Constructing activity diagrams from Arabic user requirements using Natural Language Processing tool
96	Noaeen, Abad, and Far (2017a)	Let's Hear it from RETTA: A Requirements Elicitation Tool for TrAffic Management Systems
97	Omoronyia et al. (2010)	A domain ontology building process for guiding requirements elicitation
98	Papanikolaou (2012)	Natural language processing of rules and regulations for compliance in the cloud
99	Parreira Júnior and Penteado (2018)	ObasCId(-Tool): an ontologically based approach for concern identification and classification and its computational support
100	Portugal, Do Prado Leite, and Almentero (2015)	Time-constrained requirements elicitation: Reusing GitHub content
101	Potts and Takahashi (1993)	An active hypertext model for system requirements
102	Rolland and Proix (1992)	A natural language Approach for requirements engineering
103	Rolland and Ben Achour (1998)	Guiding the construction of textual use case specifications
104	Saeki and Kaiya (2008)	Supporting the elicitation of requirements compliant with regulations
105	Saint-Dizier (2017)	Mining Incoherent Requirements in Technical Specifications
106	Sawyer, Rayson, and Cosh (2005)	Shallow knowledge as an aid to deep understanding in early phase requirements engineering
107	Sayao, Filho, and Prado (2008)	Requirements engineering for distributed development using software agents

#	Reference	Title
108	Selway, Mayer, and Stumptner (2014)	Semantic interpretation of requirements through cognitive grammar and configuration
109	Sharma, Bhatia, and Biswas (2014)	Automated identification of business rules in requirements documents
110	Sharma and Biswas (2015)	Generating Logical Representations for Natural Language Requirements Using Syntactic Dependencies and Norm Analysis Patterns
111	Shukur, Zin, and Ban (2002)	M2Z: A tool for translating a natural language software specification into Z
112	Simko et al. (2012)	Verifying temporal properties of use-cases in natural language
113	Skoutas and Simitsis (2007)	Flexible and customizable NL representation of requirements for ETL processes
114	Soares and Moura (2015)	A methodology to guide writing Software Requirements Specification document
115	Steinberger, Reinhartz-Berger, and Tomer (2016)	A tool for analyzing variability based on functional requirements and testing artifacts
116	Tang and Feng (2009)	An Expert System Based Approach to Modeling and Selecting Requirement Engineering Techniques
117	Todd and Cleland-Huang (2008)	A dynamic approach for managing stakeholder credentials for lightweight, collaborative, requirements elicitation
118	Van Rooijen et al. (2017)	From user demand to software service: Using machine learning to automate the requirements specification process
119	Wang et al. (2013a)	An algorithm for transforming design text ROM diagram into FBS model
120	Wang et al. (2013b)	Automatic detection of ambiguous terminology for software requirements
121	Wang et al. (2017)	A Regression Model Based Approach for Identifying Security Requirements in Open Source Software Development
122	Wang and Zhang (2017)	Experiment on automatic functional requirements analysis with the EFRFs semantic cases
123	Wang, Jiang, and Wang (2017)	Using workflow patterns to model and validate service requirements
124	Wasson et al. (2005)	Using occurrence properties of defect report data to improve requirements
125	Weber-Jahnke and Onabajo (2009)	Finding defects in natural language confidentiality requirements

#	Reference	Title
126	Weston, Chitchyan, and Rashid (2009)	Formal semantic conflict detection in aspect-oriented requirements
127	Wiesner et al. (2014)	Requirements Engineering for Cyber-Physical Systems
128	Wilmink and Bockisch (2017)	On the ability of lightweight checks to detect ambiguity in requirements documentation
129	Winkler and Vogelsang (2018)	Using Tools to Assist Identification of Non-requirements in Requirements Specifications – A Controlled Experiment
130	Wnuk, Höst, and Regnell (2012)	Replication of an experiment on linguistic tool support for consolidation of requirements from multiple sources
131	Young (2011)	Commitment analysis to operationalize software requirements from privacy policies

Appendix E

Quick reference list for codes

Code	Description	#	Code	Description	#
NT01	Anaphora detection / resolution	9	NT22	Topic modelling	3
NT02	Chunking	14	NT23	Word sense disambiguation	2
NT03	Classification	29	NT24	Word segmentation	4
NT04	Clustering	16			
NT05	Discourse analysis	1	RP1	Requirements elicitation	27
NT06	Entity recognition / extraction	21	RP1A	Requirements documentation	9
NT07	Lemmatization	14	RP2	Requirements analysis	34
NT08	Linguistic filter	36	RP2A	Requirements modelling	37
NT09	Morphological parsing	15	RP2B	Requirements classification	14
NT10	Named Entity Recognition	10	RP2C	Requirements verification	41
NT11	Part-of-Speech Tagging	63	RP2D	Requirements prioritization	2
NT12	Rule-based analysis	49			
NT13	Semantic Role Labeling	11	PT1	Classification	25
NT14	Semantic Similarity	22	PT2	Elicitation of requirements	30
NT15	Sentence splitting	26	PT3	Model analysis	20
NT16	Sentiment analysis	3	PT4	Model extraction / generation	30
NT17	Spelling checking	4	PT5	Prioritization of requirements	2
NT18	Stemming	19	PT6	Quality analysis / improvement	30
NT19	Syntactic parsing	52	PT6A	Ambiguity detection / resolution	27
NT20	Automated summarization	1	PT6B	Change impact / dependency analysis	5
NT21	Tokenization	35	PT6C	Completeness / conformance checking	13

Appendix F

Coding of RE (sub-)phases per article

Note: the # corresponds to the # in Appendix C.

#	RP1	RP1A	RP2	RP2A	RP2B	RP2C	RP2D
1	1	0	0	0	0	0	0
2	0	0	0	1	0	0	0
3	0	0	0	1	0	0	0
4	0	0	1	0	0	0	0
5	0	0	0	0	0	1	0
6	0	0	0	0	0	1	0
7	0	0	0	0	0	1	0
8	0	0	1	0	0	0	0
9	0	0	1	0	0	0	0
10	0	0	0	0	0	1	0
11	0	1	0	0	0	0	0
12	0	0	0	1	0	0	0
13	0	0	0	0	0	1	0
14	1	0	0	0	0	0	0
15	0	0	0	0	1	0	0
16	0	0	0	1	0	0	0
17	0	0	0	1	0	0	0
18	1	0	0	0	0	0	0
19	0	0	0	0	0	1	0
20	0	0	0	0	1	1	0
21	0	0	0	1	0	0	0
22	0	0	0	1	0	1	0
23	0	0	0	1	0	1	0
24	1	0	0	0	1	0	0
25	0	0	0	0	0	1	0
26	0	0	0	0	0	1	0
27	1	0	0	0	0	0	0
28	0	1	0	0	0	0	0
29	0	0	0	1	0	0	0
30	0	0	0	0	1	0	0
31	0	0	0	1	0	0	0
32	0	0	1	0	0	1	0
33	0	0	1	0	0	0	0
34	0	0	0	0	0	0	1
35	1	0	0	0	0	0	0

#	RP1	RP1A	RP2	RP2A	RP2B	RP2C	RP2D
36	0	0	1	0	0	0	0
37	0	0	0	1	0	0	0
38	0	1	0	0	0	1	0
39	0	0	0	0	0	1	0
40	0	0	1	0	0	0	0
41	0	0	0	0	0	1	0
42	0	0	0	1	0	0	0
43	0	0	0	1	0	1	0
44	0	0	0	0	0	1	0
45	0	0	1	0	0	0	0
46	0	0	0	1	0	0	0
47	1	0	0	1	0	0	0
48	0	0	0	1	0	0	0
49	0	0	1	0	0	0	0
50	0	0	0	0	0	1	0
51	0	0	1	1	0	0	0
52	0	0	0	0	1	0	0
53	0	1	0	0	0	0	0
54	0	0	0	1	0	0	0
55	0	0	0	0	1	0	0
56	1	0	0	0	0	0	0
57	0	1	0	0	0	0	0
58	0	0	1	0	0	0	0
59	0	0	0	0	0	1	0
60	0	0	0	1	0	0	0
61	0	0	0	0	0	1	0
62	1	0	0	1	0	0	0
63	0	0	0	0	0	1	0
64	0	0	0	0	1	0	0
65	0	0	0	0	0	1	0
66	0	0	0	0	0	1	0
67	0	0	0	0	0	1	0
68	0	0	0	0	0	1	0
69	0	0	1	0	0	0	0
70	1	0	0	0	0	0	0
71	1	0	0	0	0	0	0
72	0	0	0	1	0	0	0
73	0	0	1	0	0	0	0
74	0	0	0	1	0	0	0
75	0	0	0	0	0	1	0
76	0	0	0	0	0	1	0
77	0	0	1	1	0	0	0
78	0	0	0	1	0	0	0
79	1	0	0	0	0	0	0
80	1	0	0	0	1	0	0
81	1	0	0	1	0	0	0
82	1	0	0	1	0	0	0
83	0	0	0	1	0	0	0
84	0	0	0	0	0	1	0
85	0	0	0	0	0	0	1

#	RP1	RP1A	RP2	RP2A	RP2B	RP2C	RP2D
86	0	1	0	0	0	0	0
87	1	0	0	0	0	0	0
88	0	0	0	0	0	1	0
89	0	1	0	0	0	0	0
90	0	0	0	0	1	0	0
91	0	0	0	0	0	1	0
92	0	0	1	0	0	0	0
93	0	0	0	0	1	0	0
94	1	0	1	0	0	0	0
95	0	0	1	0	0	0	0
96	0	0	0	1	0	0	0
97	0	0	1	0	0	1	0
98	0	0	1	0	0	0	0
99	0	0	0	0	0	1	0
100	0	0	1	0	1	0	0
101	0	0	0	0	0	1	0
102	0	0	0	0	0	1	0
103	1	0	0	0	0	0	0
104	0	0	0	0	0	1	0
105	1	0	0	0	0	0	0
106	0	0	1	0	0	0	0
107	0	0	1	0	0	0	0
108	0	0	0	0	0	1	0
109	0	0	1	0	0	0	0
110	1	0	0	0	0	0	0
111	0	0	0	1	0	0	0
112	0	0	0	0	1	0	0
113	0	0	0	1	0	0	0
114	0	0	0	1	0	0	0
115	0	0	1	0	0	0	0
116	0	0	1	0	0	0	0
117	0	0	0	1	0	0	0
118	0	0	0	0	0	0	0
119	0	1	0	0	0	0	0
120	0	0	1	0	0	0	0
121	0	0	0	1	0	0	0
122	0	0	0	1	0	0	0
123	0	0	1	0	0	0	0
124	1	0	0	0	0	0	0
125	0	0	1	0	0	0	0
126	0	0	1	0	0	0	0
127	0	0	1	0	0	0	0
128	0	0	0	0	0	1	0
129	0	0	1	0	0	0	0
130	0	0	0	1	0	0	0
131	1	0	0	0	0	1	0
132	0	0	1	0	0	0	0
133	0	0	0	1	0	0	0
134	0	0	0	0	0	1	0
135	1	0	0	0	0	0	0

#	RP1	RP1A	RP2	RP2A	RP2B	RP2C	RP2D
136	0	0	1	0	0	0	0
137	0	0	0	1	0	0	0
138	0	0	0	0	1	0	0
139	1	0	0	0	0	0	0
140	0	1	0	0	0	1	0
141	0	0	0	0	0	1	0
142	1	0	1	0	0	0	0
143	1	0	0	0	0	0	0
144	1	0	0	0	1	0	0

Appendix G

Coding of tool purposes per article

Note: the # corresponds to the # in Appendix C.

#	PT1	PT2	PT3	PT4	PT5	PT6	PT6A	PT6B	PT6C
1	0	1	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0
3	0	0	0	1	0	0	1	0	0
4	0	0	0	0	0	0	1	0	0
5	0	0	0	0	0	1	0	0	0
6	0	0	0	0	0	0	0	0	1
7	0	0	0	0	0	0	0	0	1
8	0	0	0	0	0	0	0	1	0
9	0	0	0	1	0	0	1	0	0
10	0	0	0	0	0	1	0	0	0
11	0	1	0	0	0	0	0	0	0
12	0	0	0	0	0	0	1	0	0
13	0	0	0	0	0	0	1	0	0
14	0	1	0	0	0	0	0	0	0
15	1	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	1
18	0	1	0	0	0	0	0	0	0
19	0	0	0	0	0	1	0	0	0
20	1	0	0	0	0	1	1	0	0
21	0	0	1	0	0	0	0	0	0
22	0	0	1	0	0	0	0	0	0
23	0	0	1	0	0	0	0	0	0
24	1	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	1	0	0
26	0	0	1	1	0	0	1	0	1
27	0	1	0	0	0	0	0	0	0
28	0	0	0	0	0	0	1	0	1
29	0	0	0	1	0	0	0	0	0
30	1	0	0	0	0	0	0	0	0
31	0	0	1	0	0	0	0	0	0
32	0	0	0	0	0	1	0	0	0
33	1	0	0	0	0	1	0	0	0
34	1	0	0	0	1	0	0	0	0
35	1	0	0	0	0	0	0	0	0
36	0	0	0	1	0	0	0	0	0
37	0	0	0	1	0	0	0	0	0

#	PT1	PT2	PT3	PT4	PT5	PT6	PT6A	PT6B	PT6C
38	0	0	0	0	0	1	1	0	1
39	0	0	0	0	0	0	0	0	1
40	0	1	0	0	0	0	0	0	0
41	0	0	0	0	0	1	1	0	0
42	0	0	0	1	0	0	0	0	0
43	0	0	1	1	0	1	0	0	0
44	0	0	0	0	0	0	1	0	0
45	0	1	0	0	0	0	0	0	0
46	0	0	0	1	0	0	0	0	0
47	0	1	0	1	0	0	0	0	0
48	0	0	0	1	0	0	0	0	0
49	0	1	0	0	0	0	0	0	0
50	0	0	0	0	0	1	1	0	1
51	0	0	0	1	0	0	0	0	0
52	0	1	0	0	0	0	0	0	0
53	0	0	0	0	0	1	0	0	0
54	0	0	0	1	0	0	0	0	0
55	1	0	0	0	0	0	0	0	0
56	1	0	0	0	0	0	0	0	0
57	0	1	0	0	0	0	0	0	0
58	0	0	1	0	0	0	0	0	0
59	0	0	0	1	0	1	1	0	1
60	0	0	0	1	0	0	1	0	0
61	0	0	0	0	0	0	1	0	0
62	0	1	0	1	0	0	0	0	0
63	0	0	0	0	0	1	0	0	0
64	1	0	0	0	0	0	0	0	0
65	0	0	1	0	0	0	0	0	0
66	0	0	0	0	0	1	0	0	0
67	0	0	0	0	0	1	0	0	0
68	0	0	0	0	0	0	1	0	0
69	0	0	0	0	0	0	0	0	0
70	0	0	0	0	0	0	0	0	0
71	0	1	0	0	0	0	0	0	0
72	0	0	0	0	0	0	0	1	0
73	0	0	1	0	0	0	0	0	0
74	0	0	0	1	0	0	0	0	0
75	0	0	0	0	0	1	0	0	0
76	0	0	1	0	0	1	0	0	0
77	1	0	1	0	0	0	0	0	0
78	0	0	0	1	0	0	0	0	0
79	1	1	0	0	0	0	0	0	0
80	1	1	0	0	0	0	0	0	0
81	0	0	0	1	0	0	0	0	0
82	0	1	0	0	0	0	0	0	0
83	0	0	0	1	0	0	0	0	0
84	0	0	0	0	0	1	0	0	0
85	0	0	0	0	1	0	0	0	0
86	0	0	0	1	0	0	0	0	0
87	0	1	0	0	0	0	0	0	0

#	PT1	PT2	PT3	PT4	PT5	PT6	PT6A	PT6B	PT6C
88	1	0	0	0	0	1	0	0	0
89	0	0	1	0	0	0	0	0	0
90	1	0	0	0	0	0	0	0	0
91	0	0	0	0	0	0	1	0	0
92	0	0	0	0	0	0	0	0	0
93	1	0	0	0	0	0	0	0	0
94	0	1	0	0	0	0	0	0	0
95	0	1	0	0	0	0	0	0	0
96	0	1	0	0	0	0	0	0	0
97	0	1	0	0	0	0	1	1	0
98	0	0	1	0	0	0	0	0	0
99	0	0	0	0	0	1	0	0	0
100	1	1	0	0	0	0	0	0	0
101	0	0	0	0	0	1	1	0	1
102	0	0	0	0	0	0	1	0	0
103	1	1	0	0	0	0	0	0	0
104	0	0	0	1	0	0	0	0	0
105	0	1	0	0	0	0	0	0	0
106	0	0	1	0	0	1	0	0	0
107	0	0	1	0	0	1	0	0	0
108	0	0	1	0	0	1	0	0	0
109	0	0	0	0	0	1	0	0	0
110	0	1	0	0	0	0	0	0	0
111	0	0	0	1	0	0	0	0	0
112	0	0	0	0	0	1	0	0	0
113	0	0	1	0	0	0	0	0	0
114	0	0	1	0	0	0	0	0	0
115	1	0	0	0	0	0	0	0	0
116	0	0	0	0	0	0	0	1	0
117	0	0	0	1	0	0	0	0	0
118	0	0	0	0	0	1	0	0	0
119	0	1	0	0	0	0	0	0	0
120	0	0	0	0	0	0	0	0	0
121	0	0	0	1	0	0	0	0	0
122	0	0	0	1	0	0	0	0	0
123	1	0	0	0	0	0	1	0	0
124	0	1	0	0	0	0	0	0	0
125	1	0	0	0	0	0	0	0	0
126	0	0	1	0	0	0	0	0	0
127	0	0	0	0	0	0	0	1	0
128	0	0	0	0	0	0	1	0	1
129	0	0	0	0	0	1	1	0	0
130	0	0	0	1	0	0	0	0	0
131	0	1	0	0	0	0	0	0	0
132	0	0	0	0	0	1	0	0	0
133	0	0	0	1	0	0	0	0	0
134	0	0	0	0	0	0	1	0	0
135	0	0	0	1	0	0	0	0	0
136	1	0	0	0	0	1	0	0	1
137	0	0	0	1	0	0	0	0	0

#	PT1	PT2	PT3	PT4	PT5	PT6	PT6A	PT6B	PT6C
138	1	0	0	0	0	0	0	0	0
139	1	0	0	0	0	0	0	0	0
140	1	0	0	0	0	0	1	0	0
141	0	0	0	0	0	0	1	0	0
142	0	0	0	0	0	0	0	0	1
143	0	1	0	0	0	0	0	0	0
144	0	1	0	0	0	0	0	0	0

Appendix H

Coding of NLP tasks per article

Note: the # corresponds to the # in Appendix C.

#	NT01	NT02	NT03	NT04	NT05	NT06	NT07	NT08	NT09	NT10	NT11	NT12	NT13	NT14	NT15	NT16	NT17
1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
2	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0
3	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1
4	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	1	1	1	0	0	0	0	0	0	0	0	0	1	0	0	0
6	0	1	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0
7	0	1	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0
8	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0
9	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
12	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
14	0	0	1	0	0	1	0	1	0	0	1	0	0	0	0	1	0
15	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
18	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0
19	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0
20	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
24	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
27	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

#	NT01	NT02	NT03	NT04	NT05	NT06	NT07	NT08	NT09	NT10	NT11	NT12	NT13	NT14	NT15	NT16	NT17
29	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
30	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0
34	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0
37	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
38	0	0	0	0	0	0	1	0	1	0	1	1	0	0	0	0	0
39	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0
41	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
42	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
43	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
44	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
45	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0
46	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
48	0	0	0	0	0	1	0	0	0	0	1	0	0	0	1	0	0
49	0	0	0	0	0	1	1	1	0	0	1	0	0	0	0	1	0
50	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
51	1	0	0	0	1	0	0	0	1	0	1	1	0	0	1	0	0
52	0	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0
53	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
54	0	1	0	0	0	1	0	0	0	0	1	1	0	1	0	0	0
55	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
56	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0

#	NT01	NT02	NT03	NT04	NT05	NT06	NT07	NT08	NT09	NT10	NT11	NT12	NT13	NT14	NT15	NT16	NT17
57	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
58	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0
59	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
60	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0
61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
62	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
63	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0
64	0	0	1	0	0	1	0	1	0	0	1	0	0	0	1	0	0
65	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0
66	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
67	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0
68	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0
69	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
70	0	0	0	0	0	0	1	1	0	0	1	1	0	0	0	0	0
71	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
72	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0
73	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
74	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
75	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
76	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
77	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0
78	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0
79	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0
80	0	0	0	1	0	0	1	1	0	0	0	0	0	1	0	0	0
81	1	1	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0
82	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0
83	0	0	0	0	0	1	0	1	1	0	0	0	0	1	0	0	0
84	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

#	NT01	NT02	NT03	NT04	NT05	NT06	NT07	NT08	NT09	NT10	NT11	NT12	NT13	NT14	NT15	NT16	NT17
85	0	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0
86	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
87	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0
88	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0
89	0	0	0	0	0	0	0	0	1	0	1	1	0	1	0	0	0
90	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
91	0	0	0	1	0	1	0	1	0	0	1	0	0	1	0	0	1
92	0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	0	0
93	0	0	0	1	0	1	0	0	0	0	1	0	0	1	0	0	0
94	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
95	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0
96	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0
97	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0
98	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0
99	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
100	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
101	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
102	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
103	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
104	0	0	0	0	0	0	1	1	0	0	1	1	0	0	1	0	0
105	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0	0
106	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
107	0	0	1	0	0	0	1	1	0	0	1	0	1	0	1	0	0
108	0	0	0	0	0	0	1	1	0	0	1	1	1	0	1	0	0
109	0	0	1	0	0	0	0	0	1	0	1	0	0	0	1	0	0
110	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0
111	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0
112	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0

#	NT01	NT02	NT03	NT04	NT05	NT06	NT07	NT08	NT09	NT10	NT11	NT12	NT13	NT14	NT15	NT16	NT17
113	0	0	0	0	0	1	1	1	0	0	1	1	1	0	1	0	0
114	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
115	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
116	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0
117	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
118	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
119	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	1
120	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
121	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
122	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
123	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
124	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0
125	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1
126	1	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0
127	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
128	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
129	1	0	1	0	0	0	0	0	0	1	1	0	0	0	1	0	0
130	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
131	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0
132	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
133	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
134	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
135	0	0	0	0	0	0	0	0	1	0	1	1	1	0	1	0	0
136	0	0	1	0	0	1	0	1	0	0	1	0	0	0	0	0	0
137	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
138	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0
139	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0
140	1	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0

#	**NT01**	**NT02**	**NT03**	**NT04**	**NT05**	**NT06**	**NT07**	**NT08**	**NT09**	**NT10**	**NT11**	**NT12**	**NT13**	**NT14**	**NT15**	**NT16**	**NT17**
141	0	1	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0
142	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0
143	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0
144	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0

#	NT18	NT19	NT20	NT21	NT22	NT23	NT24
1	0	0	0	0	0	0	0
2	0	1	0	1	0	0	0
3	0	0	0	1	0	0	0
4	0	0	0	0	0	0	0
5	0	1	0	0	0	0	0
6	0	0	0	1	0	0	0
7	0	0	0	1	0	0	0
8	0	0	0	1	0	0	0
9	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0
12	0	1	0	0	0	0	0
13	0	1	0	0	0	0	0
14	1	0	0	0	0	0	0
15	0	0	0	0	0	0	0
16	0	1	0	1	0	0	0
17	0	1	0	0	0	0	0
18	0	0	0	0	1	0	0
19	0	1	0	0	0	0	0
20	0	1	0	0	0	0	0
21	0	0	0	0	0	0	0
22	1	1	0	0	0	0	0
23	0	1	0	0	0	0	0
24	1	0	0	0	0	0	0
25	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0
27	0	1	0	1	0	0	0
28	0	0	0	0	0	0	0
29	0	1	0	0	0	0	0
30	0	0	0	0	0	0	0
31	0	0	0	0	0	0	0
32	0	1	0	0	0	0	0
33	0	0	0	0	0	0	0
34	1	0	0	0	0	0	0
35	1	0	0	0	0	0	0
36	0	1	0	0	0	0	0
37	0	1	0	0	0	0	0
38	0	0	0	0	0	0	0
39	0	0	0	0	0	0	0
40	0	1	0	0	0	1	0
41	0	0	0	0	0	0	0
42	0	1	0	0	0	0	0
43	0	1	0	0	0	0	0
44	0	0	0	0	0	0	0
45	0	0	0	0	0	0	0
46	0	0	0	0	0	0	0
47	0	1	0	0	0	0	0
48	0	1	0	1	0	0	0
49	0	0	0	0	1	0	0
50	0	1	0	0	0	0	0

#	NT18	NT19	NT20	NT21	NT22	NT23	NT24
51	0	1	0	1	0	0	0
52	1	0	0	0	0	0	0
53	0	0	0	0	0	0	0
54	1	1	0	0	0	0	0
55	0	0	0	0	0	0	0
56	0	0	0	0	0	0	0
57	0	0	0	0	0	0	0
58	0	1	0	0	0	0	0
59	0	0	0	0	0	0	0
60	0	1	0	0	0	0	0
61	0	1	0	1	0	0	0
62	0	1	0	0	0	0	0
63	0	0	0	0	0	0	0
64	0	0	0	0	0	0	0
65	0	1	0	0	0	0	0
66	0	0	0	0	0	0	0
67	0	1	0	0	0	0	0
68	0	0	0	0	0	0	0
69	0	0	0	0	0	0	0
70	0	0	0	1	1	0	0
71	0	0	0	0	0	0	1
72	0	1	0	0	0	0	0
73	0	0	0	0	0	0	0
74	0	0	0	0	0	0	0
75	0	1	0	0	0	0	0
76	0	1	0	0	0	0	0
77	0	0	0	0	0	0	0
78	0	1	0	1	0	0	0
79	0	0	0	0	0	0	0
80	1	0	0	0	0	0	0
81	0	0	0	0	0	0	0
82	0	0	0	0	0	0	0
83	0	1	0	1	0	0	0
84	0	0	0	0	0	0	0
85	0	0	0	1	0	0	0
86	0	0	0	0	0	0	0
87	0	0	0	1	0	0	0
88	0	0	0	0	0	0	0
89	0	0	0	1	0	0	0
90	0	0	0	0	0	0	0
91	1	0	0	0	0	0	0
92	0	0	0	0	0	0	0
93	0	0	0	0	0	0	0
94	0	0	0	0	0	0	0
95	0	0	0	1	0	0	0
96	0	0	0	0	0	0	0
97	1	0	0	1	0	0	0
98	0	1	0	1	0	0	0
99	0	1	0	0	0	0	0
100	0	0	0	0	0	0	1

#	NT18	NT19	NT20	NT21	NT22	NT23	NT24
101	0	1	0	0	0	0	0
102	0	0	0	0	0	0	0
103	0	1	0	0	0	0	0
104	0	1	0	1	0	0	0
105	0	1	0	1	0	0	0
106	1	0	0	1	0	0	0
107	1	1	0	1	0	0	0
108	1	0	0	1	0	0	0
109	0	0	0	1	0	0	0
110	0	0	0	0	0	0	0
111	0	0	0	1	0	0	0
112	0	0	0	0	0	0	0
113	0	0	0	1	0	0	0
114	0	1	0	0	0	0	0
115	0	0	0	0	0	0	0
116	0	0	0	1	0	0	0
117	0	0	0	0	0	0	0
118	0	0	0	0	0	0	0
119	0	0	0	0	0	0	0
120	0	0	0	0	0	0	0
121	0	1	0	0	0	0	0
122	0	1	0	0	0	0	0
123	0	1	0	0	0	0	0
124	0	0	0	0	0	0	0
125	1	1	0	1	0	0	0
126	1	1	0	1	0	0	1
127	1	1	0	1	0	0	0
128	0	0	0	0	0	0	0
129	0	0	1	1	0	0	0
130	0	1	0	0	0	0	1
131	0	0	0	0	0	0	0
132	0	0	0	0	0	0	0
133	0	0	0	0	0	0	0
134	0	0	0	0	0	0	0
135	0	1	0	1	0	0	0
136	1	0	0	0	0	0	0
137	1	1	0	1	0	0	0
138	0	0	0	0	0	0	0
139	0	0	0	0	0	0	0
140	0	0	0	0	0	0	0
141	0	1	0	0	0	0	0
142	1	0	0	1	0	1	0
143	0	0	0	0	0	0	0
144	0	0	0	0	0	0	0

Bibliography

Abad, Zahra Shakeri Hossein et al. (2017). "What works better? a study of classifying requirements". In: *Requirements Engineering Conference (RE), 2017 IEEE 25th International*. IEEE, pp. 496–501.
Aceituna, Daniel et al. (2014). "Model-based requirements verification method: Conclusions from two controlled experiments". In: *Information and Software Technology* 56.3, pp. 321–334.
Achimugu, Philip et al. (2014). "A systematic literature review of software requirements prioritization research". In: *Information and software technology* 56.6, pp. 568–585.
Adedjouma, Morayo, Mehrdad Sabetzadeh, and Lionel C Briand (2014). "Automated detection and resolution of legal cross references: Approach and a study of Luxembourg's legislation". In: *Requirements Engineering Conference (RE), 2014 IEEE 22nd International*. IEEE, pp. 63–72.
Al-Hroob, Aysh, Ayad Tareq Imam, and Rawan Al-Heisa (2018). *The use of artificial neural networks for extracting actions and actors from requirements document*. DOI: 10.1016/j.infsof.2018.04.010.
Al-Sabbagh, Rania, Roxana Girju, and Jana Diesner (2015). "A Unified Framework to Identify and Extract Uncertainty Cues, Holders, and Scopes in One Fell-Swoop". In: *Computational Linguistics and Intelligent Text Processing*. Ed. by Alexander Gelbukh. Cham: Springer International Publishing, pp. 310–334. ISBN: 978-3-319-18111-0.
Alabdulkareem, Fatima, Nick Cercone, and Sotirios Liaskos (2015a). "Goal and Preference Identification through natural language". In: *Requirements Engineering Conference (RE), 2015 IEEE 23rd International*. IEEE, pp. 56–65.
— (2015b). "Goal and Preference Identification through natural language". In: *2015 IEEE 23rd International Requirements Engineering Conference, RE 2015 - Proceedings*. ISBN: 9781467369053. DOI: 10.1109/RE.2015.7320408.
Aletras, Nikolaos et al. (2016). "Predicting judicial decisions of the European Court of Human Rights: A natural language processing perspective". In: *PeerJ Computer Science* 2, e93.
Ali, Busyairah Syd and Zarinah Mohd Kasirun (2008). "Developing tool for crosscutting concern identification using nlp". In: *Information Technology, 2008. ITSim 2008. International Symposium on*. Vol. 3. IEEE, pp. 1–8.
Alkhader, Yara, Amjad Hudaib, and Bassam Hammo (2006). "Experimenting with extracting software requirements using NLP approach". In: *2nd International Conference on Information and Automation, ICIA 2006*. ISBN: 1424405556. DOI: 10.1109/ICINFA.2006.374136.
Ambriola, Vincenzo and Vincenzo Gervasi (1997). "Processing natural language requirements". In: *ase*. IEEE, p. 36.
— (2006). "On the systematic analysis of natural language requirements with c irce". In: *Automated Software Engineering* 13.1, pp. 107–167.
Ananiadou, Sophia and John Mcnaught (2006). *Text mining for biology and biomedicine*. Artech House London.

Annervaz, K. M. et al. (2013). "Natural language requirements quality analysis based on business domain models". In: *2013 28th IEEE/ACM International Conference on Automated Software Engineering, ASE 2013 - Proceedings*. ISBN: 9781479902156. DOI: 10.1109/ASE.2013.6693132.

Arnold, Doug (2003). "Why translation is difficult for computers". In: *Computers and Translation: A translator's guide, Amsterdam and Philadelphia: John Benjamins*, pp. 119–42.

Arora, Chetan et al. (2013). "Automatic checking of conformance to requirement boilerplates via text chunking: An industrial case study". In: *International Symposium on Empirical Software Engineering and Measurement*. ISBN: 978-0-7695-5056-5. DOI: 10.1109/ESEM.2013.13.

Arora, Chetan et al. (2015a). "Automated checking of conformance to requirements templates using natural language processing". In: *IEEE Transactions on Software Engineering*. ISSN: 00985589. DOI: 10.1109/TSE.2015.2428709.

Arora, Chetan et al. (2015b). "Change impact analysis for natural language requirements: An NLP approach". In: *Requirements Engineering Conference (RE), 2015 IEEE 23rd International*. IEEE, pp. 6–15.

Arora, Chetan et al. (2015c). "Change Impact Analysis for Natural Lauguage Requirements: An NLP Approach". In: *23rd IEEE International Requirements Engineering Conference (RE'15)*. ISBN: 978-1-4673-6905-3. DOI: 10.1109/RE.2015.7320403.

Asad Naqvi, Syed et al. (2010). "Cross-document dependency analysis for system-of-system integration". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3642125654. DOI: 10.1007/978-3-642-12566-9_11.

Asano, Keisuke, Shinpei Hayashi, and Motoshi Saeki (2017). "Detecting bad smells of refinement in goal-oriented requirements analysis". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319706245. DOI: 10.1007/978-3-319-70625-2_12.

Aurum, Aybüke and Claes Wohlin (2005). "Requirements engineering: setting the context". In: *Engineering and managing software requirements*. Springer, pp. 1–15.

Aysolmaz, Banu et al. (2018). "A semi-automated approach for generating natural language requirements documents based on business process models". In: *Information and Software Technology*. ISSN: 09505849. DOI: 10.1016/j.infsof.2017.08.009.

Badger, Julia, David Throop, and Charles Claunch (2014). "Vared: verification and analysis of requirements and early designs". In: *Requirements Engineering Conference (RE), 2014 IEEE 22nd International*. IEEE, pp. 325–326.

Baeza-Yates, Ricardo, Berthier de Araújo Neto Ribeiro, et al. (2011). *Modern information retrieval*. New York: ACM Press; Harlow, England: Addison-Wesley,

Bagheri, Ebrahim, Faezeh Ensan, and Dragan Gasevic (2012). "Decision support for the software product line domain engineering lifecycle". In: *Automated Software Engineering* 19.3, pp. 335–377.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473*.

Bajwa, Imran Sarwar, Mark Lee, and Behzad Bordbar (2012). "Resolving syntactic ambiguities in natural language specification of constraints". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642286032. DOI: 10.1007/978-3-642-28604-9_15.

Bajwa, Imran Sarwar et al. (2012). "On a chain of transformations for generating alloy from NL constraints". In: *7th International Conference on Digital Information Management, ICDIM 2012*. ISBN: 9781467324281. DOI: 10.1109/ICDIM.2012.6360153.

Bakiu, Elsa and Emitza Guzman (2017). "Which feature is unusable? Detecting usability and user experience issues from user reviews". In: *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017*. ISBN: 9781538634882. DOI: 10.1109/REW.2017.76.

Bano, Muneera and Didar Zowghi (2013). "Users' involvement in requirements engineering and system success". In: *Empirical Requirements Engineering (EmpiRE), 2013 IEEE Third International Workshop on*. IEEE, pp. 24–31.

Bates, Madeleine (1995). "Models of natural language understanding". In: *Proceedings of the National Academy of Sciences* 92.22, pp. 9977–9982.

Bäumer, Frederik S and Michaela Geierhos (2016). "Running Out of Words: How Similar User Stories Can Help to Elaborate Individual Natural Language Requirement Descriptions". In: *International Conference on Information and Software Technologies*. Springer, pp. 549–558.

Belsis, Petros, Anastasios Koutoumanos, and Cleo Sgouropoulou (2014). "PBURC: A patterns-based, unsupervised requirements clustering framework for distributed agile software development". In: *Requirements Engineering*. ISSN: 1432010X. DOI: 10.1007/s00766-013-0172-9.

Ben Abdessalem Karaa, Wahiba et al. (2015). "Automatic builder of class diagram (ABCD): an application of UML generation from functional requirements". In: *Software: Practice and Experience* 46.11, pp. 1443–1458. DOI: 10.1002/spe.2384. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.2384.

Berander, Patrik and Anneliese Andrews (2005). "Requirements prioritization". In: *Engineering and managing software requirements*. Springer, pp. 69–94.

Bergmann, Ralph and Pádraig Cunningham (2002). "Acquiring Customers' Requirementsin Electronic Commerce". In: *Artificial Intelligence Review* 18.3-4, pp. 163–193.

Berry, Daniel et al. (2012). "The case for dumb requirements engineering tools". In: *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer, pp. 211–217.

Berry, Michael W and Malu Castellanos (2004). "Survey of text mining". In: *Computing Reviews* 45.9, p. 548.

Bhat, Manoj, Chunyang Ye, and Hans Arno Jacobsen (2014). "Orchestrating SOA using requirement specifications and domain ontologies". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783662453902. DOI: 10.1007/978-3-662-45391-9_30.

Bhatia, Jaspreet and Travis D Breaux (2017). "A Data Purpose Case Study of Privacy Policies". In: *Requirements Engineering Conference (RE), 2017 IEEE 25th International*. IEEE, pp. 394–399.

Bhatia, Jaspreet et al. (2013). "Using grammatical knowledge patterns for structuring requirements specifications". In: *2013 IEEE third international workshop on requirements patterns (RePa)*. IEEE, pp. 31–34.

Bhatia, Jaspreet et al. (2016a). "A theory of vagueness and privacy risk perception". In: *Requirements Engineering Conference (RE), 2016 IEEE 24th International*. IEEE, pp. 26–35.

Bhatia, Jaspreet et al. (2016b). "Automated extraction of regulated information types using hyponymy relations". In: *2016 IEEE 24th International Requirements Engineering Conference Workshops (REW)*. IEEE, pp. 19–25.

Bhowmik, Tanmay et al. (2015). "Leveraging topic modeling and part-of-speech tagging to support combinational creativity in requirements engineering". In: *Requirements Engineering*. ISSN: 1432010X. DOI: 10.1007/s00766-015-0226-2.

Biébow, Brigitte and Sylvie Szulman (1993). "Acquisition and validation: from text to semantic network". In: *Knowledge Acquisition for Knowledge-Based Systems*. Ed. by N Aussenac et al. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 427–446. ISBN: 978-3-540-47996-3.

Bilal, Hafiz Anas et al. (2016). "Requirements Validation Techniques: An Empirical Study". In: *International Journal of Computer Applications* 148.14.

Bird, Steven and Edward Loper (2004). "NLTK: the natural language toolkit". In: *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*. Association for Computational Linguistics, p. 31.

Boddu, Ravishankar et al. (2004). "RETNA: From requirements to testing in a natural way". In: *Proceedings of the IEEE International Conference on Requirements Engineering*. ISBN: 0769521746. DOI: 10.1109/ICRE.2004.1335683.

Boehm, Barry W (1984). "Software engineering economics". In: *IEEE transactions on Software Engineering* 1, pp. 4–21.

Bokaei Hosseini, Mitra, Travis D. Breaux, and Jianwei Niu (2018). "Inferring Ontology Fragments from Semantic Role Typing of Lexical Variants". In: *Requirements Engineering: Foundation for Software Quality*. Ed. by Erik Kamsties, Jennifer Horkoff, and Fabiano Dalpiaz. Cham: Springer International Publishing, pp. 39–56. ISBN: 978-3-319-77243-1.

Bolloju, Narasimha and Christoph Schneider (2011). "Facilitating the Creation of Quality Object Models Using a Knowledge Based System". In: *Information Systems Development*. Ed. by Jaroslav Pokorny et al. New York, NY: Springer New York, pp. 137–147. ISBN: 978-1-4419-9790-6.

Bolloju, Narasimha, Christoph Schneider, and Vijayan Sugumaran (2012). "A knowledge-based system for improving the consistency between object models and use case narratives". In: *Expert Systems with Applications*. ISSN: 09574174. DOI: 10.1016/j.eswa.2012.02.126.

Böttger, K. et al. (2003). "Towards reconciling use cases via controlled language and graphical models". In: *Web Knowledge Management and Decision Support*. ISSN: 03029743. DOI: 10.1007/3-540-36524-9_10.

Boutkova, Ekaterina and Frank Houdek (2011a). "Semi-automatic identification of features in requirement specifications". In: *Requirements Engineering Conference (RE), 2011 19th IEEE International*. IEEE, pp. 313–318.

— (2011b). "Semi-automatic identification of features in requirement specifications". In: *Proceedings of the 2011 IEEE 19th International Requirements Engineering Conference, RE 2011*. ISBN: 9781457709234. DOI: 10.1109/RE.2011.6051627.

Bowden, P., M. Hargreaves, and C. S. Langensiepen (2000). "Estimation support by lexical analysis of requirements documents". In: *Journal of Systems and Software*. ISSN: 01641212. DOI: 10.1016/S0164-1212(99)00114-4.

Boyd, S., D. Zowghi, and Vincenzo Gervasi (2007). "Optimal-constraint lexicons for requirements specifications". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISSN: 03029743 16113349. DOI: 10.1007/978-3-540-73031-6_15.

Breaux, Travis D., Hanan Hibshi, and Ashwini Rao (2014). "Eddy, a formal language for specifying and analyzing data flow specifications for conflicting privacy requirements". In: *Requirements Engineering*. ISSN: 1432010X. DOI: 10.1007/s00766-013-0190-7.

Brennan, Kevin et al. (2009). *A Guide to the Business Analysis Body of Knowledger*. Iiba.

Brill, Eric (2000). "Part-of-speech tagging". In: *Handbook of natural language processing*, pp. 403–414.

Briscoe, Ted (2011). "Introduction to Linguistics for Natural Language Processing". In: *Computer Laboratory, University of Cambridge.–October* 4.

Bryson, John M (2004). "What to do when stakeholders matter: stakeholder identification and analysis techniques". In: *Public management review* 6.1, pp. 21–53.

Burnay, Corentin, Ivan J Jureta, and Stéphane Faulkner (2014). "An exploratory study of topic importance in requirements elicitation interviews". In: *International Conference on Advanced Information Systems Engineering*. Springer, pp. 180–195.

Byrd, Terry Anthony, Kathy L Cossick, and Robert W Zmud (1992). "A synthesis of research on requirements analysis and knowledge acquisition techniques". In: *MIS quarterly*, pp. 117–138.

Cambria, Erik and Bebo White (2014). "Jumping NLP curves: A review of natural language processing research". In: *IEEE Computational intelligence magazine* 9.2, pp. 48–57.

Carlson, Nathan and Phil Laplante (2014). "The NASA automated requirements measurement tool: A reconstruction". In: *Innovations in Systems and Software Engineering*. ISSN: 16145054. DOI: 10.1007/s11334-013-0225-8.

Carrizo, Dante, Oscar Dieste, and Natalia Juristo (2014). "Systematizing requirements elicitation technique selection". In: *Information and Software Technology* 56.6, pp. 644–669.

Casagrande, Erik et al. (2014b). "NLP-KAOS for systems goal elicitation: Smart metering system case study". In: *IEEE Transactions on Software Engineering*. ISSN: 00985589. DOI: 10.1109/TSE.2014.2339811.

Casagrande, Erik et al. (2014a). "NLP-KAOS for systems goal elicitation: Smart metering system case study". In: *IEEE Transactions on Software Engineering* 40.10, pp. 941–956.

Casamayor, Agustin, Daniela Godoy, and Marcelo Campo (2010). "Identification of non-functional requirements in textual specifications: A semi-supervised learning approach". In: *Information and Software Technology* 52.4, pp. 436–445. ISSN: 0950-5849. DOI: https://doi.org/10.1016/j.infsof.2009.10.010. URL: http://www.sciencedirect.com/science/article/pii/S0950584909001918.

— (2012). "Functional grouping of natural language requirements for assistance in architectural software design". In: *Knowledge-Based Systems*. ISSN: 09507051. DOI: 10.1016/j.knosys.2011.12.009.

Cascini, Gaetano, Alessandro Fantechi, and Emilio Spinicci (2004). "Natural language processing of patents and technical documentation". In: *Document analysis systems VI*. ISSN: 03029743. DOI: 10.1007/b100557. arXiv: 9780201398298.

Chen, Huafeng et al. (2010). "Text-based requirements preprocessing using nature language processing techniques". In: *2010 International Conference on Computer Design and Applications, ICCDA 2010*. ISBN: 9781424471638. DOI: 10.1109/ICCDA.2010.5540935.

Chen, Peter Pin-Shan (1988). "The entity-relationship model—toward a unified view of data". In: *Readings in artificial intelligence and databases*. Elsevier, pp. 98–111.

Chinosi, Michele and Alberto Trombetta (2012). "BPMN: An introduction to the standard". In: *Computer Standards & Interfaces* 34.1, pp. 124–134.

Chomsky, Noam (1967). "Recent contributions to the theory of innate ideas". In: *Synthese* 17.1, pp. 2–11.

Chowdhury, Gobinda G (2003). "Natural language processing". In: *Annual review of information science and technology* 37.1, pp. 51–89.

Chung, Lawrence and Julio Cesar Sampaio do Prado Leite (2009). "On non-functional requirements in software engineering". In: *Conceptual modeling: Foundations and applications*. Springer, pp. 363–379.

Ciemniewska, Alicja et al. (2007). "Supporting Use-case Reviews". In: *Proceedings of the 10th International Conference on Business Information Systems*. ISBN: 978-3-540-72034-8. DOI: 10.1007/978-3-540-72035-5_33.

Cleland-Huang, Jane et al. (2006). "The detection and classification of non-functional requirements with application to early aspects". In: *Proceedings of the IEEE International Conference on Requirements Engineering*. ISBN: 9780769525556. DOI: 10.1109/RE.2006.65.

— (2007b). "Automated classification of non-functional requirements". In: *Requirements Engineering*. ISBN: 0947-3602. DOI: 10.1007/s00766-007-0045-1. arXiv: arXiv:1011.1669v3.

— (2007a). "Automated classification of non-functional requirements". In: *Requirements Engineering* 12.2, pp. 103–120.

Cohen, K Bretonnel and Lawrence Hunter (2004). "Natural language processing and systems biology". In: *Artificial intelligence methods and tools for systems biology*. Springer, pp. 147–173.

Cohn, Mike (2004). *User stories applied: For agile software development*. Addison-Wesley Professional.

Collobert, Ronan et al. (2011). "Natural language processing (almost) from scratch". In: *Journal of Machine Learning Research* 12.Aug, pp. 2493–2537.

Cruz, Breno Dantas et al. (2017a). "Detecting Vague Words & Phrases in Requirements Documents in a Multilingual Environment". In: *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017*. ISBN: 9781538631911. DOI: 10.1109/RE.2017.24.

— (2017b). "Detecting Vague Words & Phrases in Requirements Documents in a Multilingual Environment". In: *Requirements Engineering Conference (RE), 2017 IEEE 25th International*. IEEE, pp. 233–242.

Cziharz, Torsten et al. *Handbook of Requirements Modeling IREB Standard*.

Dalpiaz, Fabiano, Ivor van der Schalk, and Garm Lucassen (2018). "Pinpointing ambiguity and incompleteness in requirements engineering via information visualization and NLP". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319772424. DOI: 10.1007/978-3-319-77243-1_8.

Daramola, Olawande, Guttorm Sindre, and Thomas Moser (2012). "Ontology-based support for security requirements specification process". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642336171. DOI: 10.1007/978-3-642-33618-8_28.

Daramola, Olawande et al. (2011). "Enabling hazard identification from requirements and reuse-oriented HAZOP analysis". In: *2011 4th International Workshop on Managing Requirements Knowledge, MaRK'11 - Part of the 19th IEEE International Requirements Engineering Conference, RE'11*. ISBN: 9781457709388. DOI: 10.1109/MARK.2011.6046555.

Davis, Alan et al. (2006). "Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review". In: *Requirements Engineering, 14th IEEE International Conference*. IEEE, pp. 179–188.

Deeptimahanti, Deva Kumar and Ratna Sanyal (2009). "An Innovative Approach for Generating Static UML Models from Natural Language Requirements". In: *Advances in Software Engineering*. Ed. by Tai-hoon Kim et al. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 147–163. ISBN: 978-3-642-10242-4.

del Socorro Bernardos Galindo, M and Guadalupe Aguado de Cea (2001). "A New Approach in Building a Corpus for Natural Language Generation Systems". In: *Computational Linguistics and Intelligent Text Processing*. Ed. by Alexander Gelbukh. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 216–225. ISBN: 978-3-540-44686-6.

Demberg, Vera (2007). "A language-independent unsupervised model for morphological segmentation". In: *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 920–927.

Demner-Fushman, Dina, Wendy W Chapman, and Clement J McDonald (2009). "What can natural language processing do for clinical decision support?" In: *Journal of biomedical informatics* 42.5, pp. 760–772.

Deocadez, Roger, Rachel Harrison, and Daniel Rodriguez (2017). "Automatically classifying requirements from app stores: A preliminary study". In: *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017*. ISBN: 9781538634882. DOI: 10.1109/REW.2017.58.

Díaz, Isabel et al. (2005a). "Integrating Natural Language Techniques in OO-Method". In: *CICLing 2005*. ISSN: 03029743. DOI: 10.1007/978-3-540-30586-6_60.

Díaz, Isabel et al. (2005b). "Interaction transformation patterns based on semantic roles". In: *10th International Conference on Applications of Natural Language to Information Systems, NLDB 2005: Natural Language Processing and Information Systems*. ISBN: 03029743 (ISSN).

Dieste, Oscar and Natalia Juristo (2011). "Systematic review and aggregation of empirical studies on elicitation techniques". In: *IEEE Transactions on Software Engineering* 37.2, pp. 283–304.

Dostert, Leon E (1955). "The georgetown-ibm experiment". In: *1955). Machine translation of languages. John Wiley & Sons, New York*, pp. 124–135.

Drazan, Jaroslav and Vladimir Mencl (2007). "Improved processing of textual use cases: Deriving behavior specifications". In: *SOFSEM 2007: Current Trends in Theory and Practice of Computer Science*. ISSN: 0302-9743. DOI: 10.1007/978-3-540-69507-3.

Driss, Maha et al. (2010). "A requirement-centric approach to web service modeling, discovery, and selection". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3642173578. DOI: 10.1007/978-3-642-17358-5_18.

Duan, Chuan (2008). "Clustering and its application in requirements engineering". In: *Technical Reports*, p. 7.

Duan, Chuan and Jane Cleland-Huang (2007). "A Clustering Technique for Early Detection of Dominant and Recessive Cross-Cutting Concerns". In: *Early Aspects at ICSE: Workshops in Aspect-Oriented Requirements Engineering and Architecture Design (EARLYASPECTS'07)*. DOI: 10.1109/EARLYASPECTS.2007.1.

Duan, Chuan et al. (2009). "Towards automated requirements prioritization and triage". In: *Requirements Engineering*. ISBN: 0769529356. DOI: 10.1007/s00766-009-0079-7.

Duan, Chuan et al. (2015). "User-constrained clustering in online requirements forums". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319161006. DOI: 10.1007/978-3-319-16101-3_21.

Dwarakanath, Anurag, Roshni R. Ramnani, and Shubhashis Sengupta (2013a). "Automatic extraction of glossary terms from natural language requirements". In: *2013 21st IEEE International Requirements Engineering Conference, RE 2013 - Proceedings*. ISBN: 9781467357654. DOI: 10.1109/RE.2013.6636736.

Dwarakanath, Anurag, Roshni R Ramnani, and Shubhashis Sengupta (2013b). "Automatic extraction of glossary terms from natural language requirements". In: *Requirements Engineering Conference (RE), 2013 21st IEEE International*. IEEE, pp. 314–319.

Elallaoui, Meryem, Khalid Nafil, and Raja Touahni (2018). "Automatic Transformation of User Stories into UML Use Case Diagrams using NLP Techniques". In: *Procedia Computer Science* 130, pp. 42–49. ISSN: 1877-0509. DOI: 10.1016/J.PROCS.2018.04.010. URL: https://www.sciencedirect.com/science/article/pii/S1877050918303600.

Engel, Marcin et al. (1993). "A formal approach to computer systems requirements documentation". In: *Hybrid Systems*. Springer, pp. 452–474.

Evans, Morgan C. et al. (2017). "An Evaluation of Constituency-Based Hyponymy Extraction from Privacy Policies". In: *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017*. ISBN: 9781538631911. DOI: 10.1109/RE.2017.87.

Fanmuy, Gauthier, Anabel Fraga, and Juan Llorens (2012). "Requirements verification in the industry". In: *Complex Systems Design & Management*. Springer, pp. 145–160.

Farfeleder, Stefan et al. (2011). "Ontology-driven guidance for requirements elicitation". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642210631. DOI: 10.1007/978-3-642-21064-8_15.

Fatwanto, Agung (2013). "Natural language requirements specification analysis using Part-of-Speech Tagging". In: *Second International Conference on Future Generation Communication Technologies (FGCT 2013)*. ISBN: 978-1-4799-2975-7. DOI: 10.1109/FGCT.2013.6767215.

Femmer, Henning et al. (2017). "Rapid quality assurance with Requirements Smells". In: *Journal of Systems and Software*. ISSN: 01641212. DOI: 10.1016/j.jss.2016.02.047. arXiv: 1611.08847.

Ferrari, Alessio, Beatrice Donati, and Stefania Gnesi (2017). "Detecting domain-specific ambiguities: An NLP approach based on wikipedia crawling and word embeddings". In: *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017*. ISBN: 9781538634882. DOI: 10.1109/REW.2017.20.

Ferrari, Alessio and Stefania Gnesi (2012a). "Using collective intelligence to detect pragmatic ambiguities". In: *Requirements Engineering Conference (RE), 2012 20th IEEE International*. IEEE, pp. 191–200.

— (2012b). "Using collective intelligence to detect pragmatic ambiguities". In: *2012 20th IEEE International Requirements Engineering Conference, RE 2012 - Proceedings*. ISBN: 9781467327855. DOI: 10.1109/RE.2012.6345803.

Ferrari, Alessio et al. (2014a). "From commercial documents to system requirements: an approach for the engineering of novel CBTC solutions". In: *International Journal on Software Tools for Technology Transfer*. ISSN: 14332787. DOI: 10.1007/s10009-013-0298-6.

Ferrari, Alessio et al. (2014b). "Measuring and improving the completeness of natural language requirements". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319058429. DOI: 10.1007/978-3-319-05843-6_3.

Ferrari, Alessio et al. (2014c). "Pragmatic ambiguity detection in natural language requirements". In: *2014 IEEE 1st International Workshop on Artificial Intelligence for Requirements Engineering, AIRE 2014 - Proceedings*. ISBN: 9781479963553. DOI: 10.1109/AIRE.2014.6894849.

Ferrari, Alessio et al. (2018). *Detecting requirements defects with NLP patterns: an industrial experience in the railway domain*. DOI: 10.1007/s10664-018-9596-7.

Ferreira, David de Almeida and Alberto Rodrigues da Silva (2009). "A Controlled Natural Language Approach for Integrating Requirements and Model-Driven Engineering". In: *2009 Fourth International Conference on Software Engineering Advances*. ISBN: 978-1-4244-4779-4. DOI: 10.1109/ICSEA.2009.81.

Fitzgerald, Camilo, Emmanuel Letier, and Anthony Finkelstein (2011). "Early failure prediction in feature request management systems". In: *Requirements Engineering Conference (RE), 2011 19th IEEE International*. IEEE, pp. 229–238.

Fliedl, Günther, Christian Kop, and Heinrich C. Mayr (2005). "From textual scenarios to a conceptual schema". In: *Data & Knowledge Engineering* 55.1, pp. 20–37. ISSN: 0169-023X. DOI: https://doi.org/10.1016/j.datak.2004.11.007. URL: http://www.sciencedirect.com/science/article/pii/S0169023X04002204.

Fliedl, Günther et al. (2000). "Linguistically based requirements engineering – The NIBA-project". In: *Data & Knowledge Engineering* 35.2, pp. 111–120. ISSN: 0169-023X. DOI: https://doi.org/10.1016/S0169-023X(00)00029-X. URL: http://www.sciencedirect.com/science/article/pii/S0169023X0000029X.

Fliedl, Günther et al. (2004). "Semantic Tagging and Chunk-Parsing in Dynamic Modeling". In: *Natural Language Processing and Information Systems*. Ed. by Farid Meziane and Elisabeth Métais. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 421–426. ISBN: 978-3-540-27779-8.

Fliedl, Günther et al. (2007). "Deriving static and dynamic concepts from software requirements using sophisticated tagging". In: *Data & Knowledge Engineering* 61.3, pp. 433–448. ISSN: 0169-023X. DOI: https://doi.org/10.1016/j.datak.2006.06.012. URL: http://www.sciencedirect.com/science/article/pii/S0169023X06001170.

Flores, Jorge J García (2004). "Semantic Filtering of Textual Requirements Descriptions". In: *Natural Language Processing and Information Systems*. Ed. by Farid Meziane and Elisabeth Métais. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 427–433. ISBN: 978-3-540-27779-8.

Friedman, Carol, George Hripcsak, et al. (1999). "Natural language processing and its future in medicine". In: *Acad Med* 74.8, pp. 890–5.

Fu, Jicheng, Farokh B. Bastani, and I. Ling Yen (2008). "Model-driven prototyping based requirements elicitation". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3540897771. DOI: 10.1007/978-3-540-89778-1_7.

Gacitua, Ricardo, Pete Sawyer, and Vincenzo Gervasi (2010). "On the effectiveness of abstraction identification in requirements engineering". In: *Requirements Engineering Conference (RE), 2010 18th IEEE International*. IEEE, pp. 5–14.

Gärtner, Stefan et al. (2014). "Maintaining requirements for long-living software systems by incorporating security knowledge". In: *Requirements Engineering Conference (RE), 2014 IEEE 22nd International*. IEEE, pp. 103–112.

Gärtner, Stefan et al. (2014). "Maintaining requirements for long-living software systems by incorporating security knowledge". In: *2014 IEEE 22nd International Requirements Engineering Conference, RE 2014 - Proceedings*. ISBN: 9781479930333. DOI: 10.1109/RE.2014.6912252.

Garzoli, F. et al. (2013). *Robust Requirements Analysis in Complex Systems through Machine Learning*. ISBN: 9783642452598. DOI: 10.1007/978-3-642-45260-4_4.

Geierhos, Michaela and Frederik Simon Bäumer (2016). "How to complete customer requirements: Using concept expansion for requirement refinement". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319417530. DOI: 10.1007/978-3-319-41754-7_4.

Génova, Gonzalo et al. (2013). "A framework to measure and improve the quality of textual requirements". In: *Requirements Engineering*. ISSN: 09473602. DOI: 10.1007/s00766-011-0134-z.

Ghosh, Shalini et al. (2016). "ARSENAL: Automatic requirements specification extraction from natural language". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319406473. DOI: 10.1007/978-3-319-40648-0_4. arXiv: 1403.3142.

Giganto, Reynaldo and Tony Smith (2008). "Derivation of classes from use cases automatically generated by a three-level sentence processing algorithm". In: *3rd International Conference on Systems, ICONS 2008*. ISBN: 9780769531052. DOI: 10.1109/ICONS.2008.50.

Glava??, Goran, Kre??imir Fertalj, and Jan ??najder (2012). "From requirements to code: Syntax-based requirements analysis for data-driven application development". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642311772. DOI: 10.1007/978-3-642-31178-9_44.

Gleich, Benedikt, Oliver Creighton, and Leonid Kof (2010). "Ambiguity detection: Towards a tool explaining ambiguity sources". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3642141919. DOI: 10.1007/978-3-642-14192-8_20. arXiv: 9780201398298.

Gokyer, Gokhan et al. (2008). "Non-functional requirements to architectural concerns: MML and NLP at crossroads". In: *Proceedings - The 3rd International Conference on Software Engineering Advances, ICSEA 2008, Includes ENTISY 2008: International Workshop on Enterprise Information Systems*. ISBN: 9780769533728. DOI: 10.1109/ICSEA.2008.28.

Goldin, Leah and Daniel M. Berry (1997). "AbStFinder, a Prototype Natural Language Text Abstraction Finder for Use in Requirements Elicitation". In: *Automated Software Engineering*. ISSN: 09288910. DOI: 10.1023/A:1008617922496.

Gonzalez Moreno, Juan Carlos and Luis Vazquez Lopez (2009). "Using Techniques Based on Natural Language in the Development Process of Multiagent Systems". In: *INTERNATIONAL SYMPOSIUM ON DISTRIBUTED COMPUTING AND ARTIFICIAL INTELLIGENCE 2008*. ISBN: 978-3-540-85862-1.

Gorse, N. et al. (2004). "Mixing linguistic and formal techniques for high-level requirements engineering". In: *Proceedings of the International Conference on Microelectronics, ICM*. ISBN: 0-7803-8656-6. DOI: 10.1109/ICM.2004.1434581.

Grady, Jeffrey O (2010). *System requirements analysis*. Elsevier.

Greenwood, Phil et al. (2011). "Modelling adaptability and variability in requirements". In: *Requirements Engineering Conference (RE), 2011 19th IEEE International*. IEEE, pp. 343–344.

Groen, Eduard C et al. (2017). "Users—The Hidden Software Product Quality Experts?: A Study on How App Users Report Quality Aspects in Online Reviews". In: *Requirements Engineering Conference (RE), 2017 IEEE 25th International*. IEEE, pp. 80–89.

Gulia, Sarita and Tanupriya Choudhury (2016). "An efficient automated design to generate UML diagram from Natural Language Specifications". In: *Proceedings of the 2016 6th International Conference - Cloud System and Big Data Engineering, Confluence 2016*. ISBN: 9781467382021. DOI: 10.1109/CONFLUENCE.2016.7508197.

Gulla, Jon Alte, Bram van der Vos, and Ulrich Thiel (1997). "An abductive, linguistic approach to model retrieval". In: *Data & Knowledge Engineering*. ISSN: 0169023X. DOI: 10.1016/S0169-023X(97)00008-6.

Guzman, Emitza, Mohamed Ibrahim, and Martin Glinz (2017a). "A Little Bird Told Me: Mining Tweets for Requirements and Software Evolution". In: *Requirements Engineering Conference (RE), 2017 IEEE 25th International*. IEEE, pp. 11–20.

— (2017b). "A Little Bird Told Me: Mining Tweets for Requirements and Software Evolution". In: *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017*. ISBN: 9781538631911. DOI: 10.1109/RE.2017.88.

Guzman, Emitza and Walid Maalej (2014a). "How do users like this feature? A fine grained sentiment analysis of App reviews". In: *2014 IEEE 22nd International Requirements Engineering Conference, RE 2014 - Proceedings*. ISBN: 9781479930333. DOI: 10.1109/RE.2014.6912257.

— (2014b). "How do users like this feature? a fine grained sentiment analysis of app reviews". In: *Requirements Engineering Conference (RE), 2014 IEEE 22nd International*. IEEE, pp. 153–162.

Hahn, Udo and Inderjeet Mani (2000). "The challenges of automatic summarization". In: *Computer* 33.11, pp. 29–36.

Hajri, Ines et al. (2015). "Applying product line Use case modeling in an industrial automotive embedded system: Lessons learned and a refined approach". In: *2015 ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems, MODELS 2015 - Proceedings*. ISBN: 9781467369084. DOI: 10.1109/MODELS.2015.7338265.

Halkidis, Spyros T., Alexander Chatzigeorgiou, and George Stephanides (2009). "Moving from Requirements to Design Confronting Security Issues: A Case Study". In: *On the Move to Meaningful Internet Systems: OTM 2009, Confederated International Conferences, CoopIS, DOA, IS, and ODBASE 2009*. ISBN: 978-3-642-05150-0. DOI: 10.1007/978-3-642-05151-7_4.

Harksoo Kim et al. (1999). "Informal requirements analysis supporting system for human engineer". In: *IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.99CH37028)*. Vol. 3. IEEE, pp. 1013–1018. ISBN: 0-7803-5731-0. DOI: 10.1109/ICSMC.1999.823367. URL: http://ieeexplore.ieee.org/document/823367/.

Harmain, H. M. and R. Gaizauskas (2003). "CM-Builder: A natural language-based CASE tool for object-oriented analysis". In: *Automated Software Engineering*. ISBN: 09288910. DOI: 10.1023/A:1022916028950.

Hartong, Mark, Rajni Goel, and Duminda Wijesekera (2014). "Security requirement derivation by noun-verb analysis of use-misuse case relationships: A case study using positive train control". In: *Innovations in Systems and Software Engineering*. ISSN: 16145054. DOI: 10.1007/s11334-013-0227-6.

Hatton, Sarah (2008). "Choosing the right prioritisation method". In: *Software Engineering, 2008. ASWEC 2008. 19th Australian Conference on*. IEEE, pp. 517–526.

Hausser, Roland (1999). *Foundations of computational linguistics*. Springer.

Hay, David C (2003). *Requirements analysis: From business views to architecture*. Prentice Hall Professional.
He, Ke Qing, Jian Wang, and Peng Liang (2010). "Semantic interoperability aggregation in service requirements refinement". In: *Journal of Computer Science and Technology*. ISSN: 10009000. DOI: 10.1007/s11390-010-9392-3.
He, Lei et al. (2009). "Predicting upgrade project defects based on enhancement requirements: An empirical study". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642016790. DOI: 10.1007/978-3-642-01680-6_25.
Hickey, Ann M and Alan M Davis (2003). "Elicitation technique selection: how do experts do it?" In: *Requirements engineering conference, 2003. proceedings. 11th ieee international*. IEEE, pp. 169–178.
— (2004). "A unified model of requirements elicitation". In: *Journal of Management Information Systems* 20.4, pp. 65–84.
— (2007). "An ontological approach to requirements elicitation technique selection". In: *Ontologies*. Springer, pp. 403–431.
Hirschman, Lynette and Robert Gaizauskas (2001). "Natural language question answering: the view from here". In: *natural language engineering* 7.4, pp. 275–300.
Holtzblatt, Karen and Hugh R Beyer (1995). "Requirements gathering: the human factor". In: *Communications of the ACM* 38.5, pp. 31–32.
Hosseini, Mahmood et al. (2017). "CRAFT: A crowd-annotated feedback technique". In: *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017*. ISBN: 9781538634882. DOI: 10.1109/REW.2017.27.
Howard, Newton and Erik Cambria (2013). "Intention awareness: improving upon situation awareness in human-centric environments". In: *Human-centric Computing and Information Sciences* 3.1, p. 9.
Hu, Wensheng et al. (2014). "Requires analysis based on software maintainability". In: *ICRMS 2014 - Proceedings of 2014 10th International Conference on Reliability, Maintainability and Safety: More Reliable Products, More Secure Life*. ISBN: 9781479919925. DOI: 10.1109/ICRMS.2014.7107202.
Hughes, KJ, RM Rankin, and CT Sennett (1994). "Taxonomy for requirements analysis". In: *Requirements Engineering, 1994., Proceedings of the First International Conference on*. IEEE, pp. 176–179.
Hussain, Ishrar, Leila Kosseim, and Olga Ormandjieva (2008). "Using linguistic knowledge to classify non-functional requirements in SRS documents". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3540698574. DOI: 10.1007/978-3-540-69858-6_28.
Hussain, Ishrar, Olga Ormandjieva, and Leila Kosseim (2012). "LASR: A tool for large scale annotation of software requirements". In: *2012 2nd IEEE International Workshop on Empirical Requirements Engineering, EmpiRE 2012 - Proceedings*. ISBN: 9781467343657. DOI: 10.1109/EmpiRE.2012.6347683.
Hutchins, John (2003). "ALPAC: the (in) famous report". In: *Readings in machine translation* 14, pp. 131–135.
— (2005). "The history of machine translation in a nutshell". In: *Retrieved December* 20, p. 2009.
Hutchins, W John (2004). "The Georgetown-IBM experiment demonstrated in January 1954". In: *Conference of the Association for Machine Translation in the Americas*. Springer, pp. 102–114.

Ibrahim, Mohd and Rodina Ahmad (2010a). "Class diagram extraction from textual requirements using Natural language processing (NLP) techniques". In: *Computer Research and Development, 2010 Second International Conference on*. IEEE, pp. 200–204.

— (2010b). "Class diagram extraction from textual requirements using natural language processing (NLP) techniques". In: *2nd International Conference on Computer Research and Development, ICCRD 2010*. ISBN: 9780769540436. DOI: 10.1109/ICCRD.2010.71.

Ilyas, Muhammad and Josef Küng (2009). "A similarity measurement framework for requirements engineering". In: *4th International Multi-Conference on Computing in the Global Information Technology, ICCGI 2009*.

— (2010). "Ontology-based similarity measurement in software projects through SimReq framework". In: *Proceedings - 5th International Conference on Software Engineering Advances, ICSEA 2010*. ISBN: 9780769541440. DOI: 10.1109/ICSEA.2010.70.

Jaccard, Paul (1901). "Étude comparative de la distribution florale dans une portion des Alpes et des Jura". In: *Bull Soc Vaudoise Sci Nat* 37, pp. 547–579.

Jha, Nishant and Anas Mahmoud (2017). "Mining user requirements from application store reviews using frame semantics". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319540443. DOI: 10.1007/978-3-319-54045-0_20.

— (2018). *Using frame semantics for classifying and summarizing application store reviews*. DOI: 10.1007/s10664-018-9605-x.

Jiang, Wei, Haibin Ruan, and Li Zhang (2014). "Analysis of Economic Impact of Online Reviews: An Approach for Market-Driven Requirements Evolution". In: *Communications in Computer and Information Science*. ISBN: 9783662436097. DOI: 10.1007/978-3-662-43610-3_4.

Jin, Jian, Ping Ji, and Rui Gu (2016). "Identifying comparative customer requirements from product online reviews for competitor analysis". In: *Engineering Applications of Artificial Intelligence*. ISSN: 09521976. DOI: 10.1016/j.engappai.2015.12.005.

Jin, Jian et al. (2015). "Translating online customer opinions into engineering characteristics in QFD: A probabilistic language analysis approach". In: *Engineering Applications of Artificial Intelligence*. ISSN: 09521976. DOI: 10.1016/j.engappai.2015.02.006.

Jin, Ying et al. (2010). "A concern-based approach to generating formal requirements specifications". In: *Frontiers of Computer Science in China*. ISSN: 16737350. DOI: 10.1007/s11704-010-0151-y.

Johann, Timo, Christoph Stanik, Walid Maalej, et al. (2017). "Safe: A simple approach for feature extraction from app descriptions and app reviews". In: *Requirements Engineering Conference (RE), 2017 IEEE 25th International*. IEEE, pp. 21–30.

Jones, Karen Sparck (1994). "Natural language processing: a historical review". In: *Current issues in computational linguistics: in honour of Don Walker*. Springer, pp. 3–16.

Jurafsky, Dan and James H Martin (2014). *Speech and language processing*. Vol. 3. Pearson London:

Jurkiewicz, J. and J. Nawrocki (2015). "Automated events identification in use cases". In: *Information and Software Technology*. ISSN: 09505849. DOI: 10.1016/j.infsof.2014.09.011.

Kacfah Emani, Cheikh (2014). "Automatic detection and semantic formalisation of business rules". In: *Lecture Notes in Computer Science (including subseries Lecture*

Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). ISBN: 9783319074429. DOI: 10.1007/978-3-319-07443-6_57.

Kaiya, Haruhiko and Motoshi Saeki (2005). "Ontology Based Requirements Analysis: Lightweight Semantic Processing Approach". In: *5th International Conference on Quality Software (QSIC'05), Proceedings*. ISSN: 1550-6002. DOI: 10.1109/QSIC.2005.46.

— (2006). "Using Domain Ontology as Domain Knowledge for Requirements Elicitation". In: *14th IEEE International Requirements Engineering Conference (RE'06)*. ISBN: 0-7695-2555-5. DOI: 10.1109/RE.2006.72.

Kamalrudin, Massila, John Grundy, and John Hosking (2012). "Supporting requirements modelling in the Malay language using essential use cases". In: *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC*. ISBN: 9781467308502. DOI: 10.1109/VLHCC.2012.6344503.

Kamalrudin, Massila, John Hosking, and John Grundy (2010). "MaramaAI: Automated and Visual Approach for Inconsistency Checking of Requirements". In: *Requirements Engineering Conference (RE), 2010 18th IEEE International*. IEEE, pp. 393–394.

— (2011). "Improving requirements quality using essential use case interaction patterns". In: *2011 33rd International Conference on Software Engineering (ICSE)*. ISSN: 0270-5257. DOI: 10.1145/1985793.1985866.

Kanchev, Georgi M et al. (2017a). "Canary: An Interactive and Query-Based Approach to Extract Requirements from Online Forums". In: *Requirements Engineering Conference (RE), 2017 IEEE 25th International*. IEEE, pp. 470–471.

Kanchev, Georgi M. et al. (2017b). "Canary: An Interactive and Query-Based Approach to Extract Requirements from Online Forums". In: *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017*. ISBN: 9781538631911. DOI: 10.1109/RE.2017.84.

Kang, Juyeon and Patrick Saint-Dizier (2014). "Requirement compound mining and analysis". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319098692. DOI: 10.1007/978-3-319-09870-8_14.

Kausar, Sumaira et al. (2010). "Guidelines for the selection of elicitation techniques". In: *Emerging Technologies (ICET), 2010 6th International Conference on*. IEEE, pp. 265–269.

Kim, Jintae, Sooyong Park, and Vijayan Sugumaran (2004). "A Linguistics-Based Approach for Use Case Driven Analysis Using Goal and Scenario Authoring". In: *Natural Language Processing and Information Systems*. Ed. by Farid Meziane and Elisabeth Métais. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 159–170. ISBN: 978-3-540-27779-8.

— (2006). "Improving use case driven analysis using goal and scenario authoring: A linguistics-based approach". In: *Data & Knowledge Engineering* 58.1, pp. 21–46. ISSN: 0169-023X. DOI: https://doi.org/10.1016/j.datak.2005.05.006. URL: http://www.sciencedirect.com/science/article/pii/S0169023X05000765.

Kirchsteiger, Christoph M. et al. (2008). "Specification-based Verification of Embedded Systems by Automated Test Case Generation". In: *Distributed Embedded Systems: Design, Middleware and Resources*. ISSN: 15715736. DOI: 10.1007/978-0-387-09661-2_4.

Kitchenham, Barbara (2004). "Procedures for performing systematic reviews". In: *Keele, UK, Keele University* 33.2004, pp. 1–26.

Kiyavitskaya, Nadzeya and Nicola Zannone (2008). "Requirements model generation to support requirements elicitation: The Secure Tropos experience". In: *Automated Software Engineering*. ISSN: 09288910. DOI: 10.1007/s10515-008-0028-6.

Kiyavitskaya, Nadzeya et al. (2004). "Experimenting with linguistic tools for conceptual modelling: Quality of the models and critical features". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISSN: 03029743.

Kiyavitskaya, Nadzeya et al. (2008). "Requirements for tools for ambiguity identification and measurement in natural language requirements specifications". In: *Requirements Engineering*. ISSN: 09473602. DOI: 10.1007/s00766-008-0063-7.

Knauss, Eric et al. (2011). "Supporting requirements engineers in recognising security issues". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642198571. DOI: 10.1007/978-3-642-19858-8_2.

Knauss, Eric et al. (2015). "Patterns of continuous requirements clarification". In: *Requirements Engineering*. ISSN: 1432010X. DOI: 10.1007/s00766-014-0205-z.

Ko, Deokyoon, Suntae Kim, and Sooyong Park (2018). "Automatic recommendation to omitted steps in use case specification". In: *Requirements Engineering*. ISSN: 1432-010X. DOI: 10.1007/s00766-018-0288-z. URL: https://doi.org/10.1007/s00766-018-0288-z.

Ko, Youngjoong et al. (2007b). "Using classification techniques for informal requirements in the requirements analysis-supporting system". In: *Information and Software Technology*. ISSN: 09505849. DOI: 10.1016/j.infsof.2006.11.007.

— (2007a). "Using classification techniques for informal requirements in the requirements analysis-supporting system". In: *Information and Software Technology* 49.11-12, pp. 1128–1140.

Kof, Leonid (2007a). "Scenarios: Identifying missing objects and actions by means of computational linguistics". In: *Proceedings - 15th IEEE International Requirements Engineering Conference, RE 2007*. ISBN: 0769529356. DOI: 10.1109/RE.2007.54.

— (2007b). "Treatment of Passive Voice and Conjunctions in Use Case Documents". In: *Natural Language Processing and Information Systems*. ISSN: 03029743. DOI: 10.1007/978-3-540-73351-5_16.

— (2008). "On the identification of goals in stakeholders' dialogs". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3540897771. DOI: 10.1007/978-3-540-89778-1_13.

— (2009). "Translation of textual specifications to automata by means of discourse context modeling". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3642020496. DOI: 10.1007/978-3-642-02050-6_17.

— (2010). "From requirements documents to system models: A tool for interactive semi-automatic translation". In: *Requirements Engineering Conference (RE), 2010 18th IEEE International*. IEEE, pp. 391–392.

Kof, Leonid and Birgit Penzenstadler (2011). "From requirements to models: Feedback generation as a result of formalization". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642216398. DOI: 10.1007/978-3-642-21640-4_9.

Kop, C. and H.C. Mayr (1998). "Conceptual predesign bridging the gap between requirements and conceptual design". In: *Proceedings of IEEE International Symposium on Requirements Engineering: RE '98*. IEEE Comput. Soc, pp. 90–98. ISBN:

0-8186-8356-2. DOI: `10.1109/ICRE.1998.667813`. URL: `http://ieeexplore.ieee.org/document/667813/`.

Körner, Sven J. and Torben Brumm (2009). "RESI - A natural language specification improver". In: *ICSC 2009 - 2009 IEEE International Conference on Semantic Computing*. ISBN: 9780769538006. DOI: `10.1109/ICSC.2009.47`.

Körner, Sven J. and Mathias Landhäußer (2010). "Semantic enriching of natural language texts with automatic thematic role annotation". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3642138802. DOI: `10.1007/978-3-642-13881-2_9`.

Körner, Sven J., Mathias Landhäußer, and Walter F. Tichy (2014). "Transferring research into the real world: How to improve RE with AI in the automotive industry". In: *2014 IEEE 1st International Workshop on Artificial Intelligence for Requirements Engineering, AIRE 2014 - Proceedings*. ISBN: 9781479963553. DOI: `10.1109/AIRE.2014.6894851`.

Kotonya, Gerald and Ian Sommerville (1998). *Requirements engineering: processes and techniques*. Wiley Publishing.

Kroha, P. (2000). "Preprocessing of requirements specification". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783540679783.

Kujala, Sari, Marjo Kauppinen, and Sanna Rekola (2001). "Bridging the gap between user needs and user requirements". In: *Advances in Human-Computer Interaction I (Proceedings of the Panhellenic Conference with International Participation in Human-Computer Interaction PC-HCI 2001), Typorama Publications*, pp. 45–50.

Kujala, Sari et al. (2005). "The role of user involvement in requirements quality and project success". In: *Requirements Engineering, 2005. Proceedings. 13th IEEE International Conference on*. IEEE, pp. 75–84.

Lame, Guiraude (2005). "Using NLP techniques to identify legal ontology components: concepts and relations". In: *Law and the Semantic Web*. Springer, pp. 169–184.

Ländhäußer, Mathias et al. (2015). "DeNom: A tool to find problematic nominalizations using NLP". In: *2nd International Workshop on Artificial Intelligence for Requirements Engineering, AIRE 2015 - Proceedings*. ISBN: 9781509001255. DOI: `10.1109/AIRE.2015.7337623`.

Laue, Ralf, Wilhelm Koop, and Volker Gruhn (2016). "Indicators for open issues in business process models". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319302812. DOI: `10.1007/978-3-319-30282-9_7`.

Laurent, Paula, Jane Cleland-Huang, and Chuan Duan (2007). "Towards automated requirements triage". In: *Requirements Engineering Conference, 2007. RE'07. 15th IEEE International*. IEEE, pp. 131–140.

Lee, Beum-Seuk and Barrett R Bryant (2004). "Automation of Software System Development Using Natural Language Processing and Two-Level Grammar". In: *Radical Innovations of Software and Systems Engineering in the Future*. Ed. by Martin Wirsing, Alexander Knapp, and Simonetta Balsamo. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 219–233. ISBN: 978-3-540-24626-8.

Lehtola, Laura, Marjo Kauppinen, and Sari Kujala (2004). "Requirements prioritization challenges in practice". In: *International Conference on Product Focused Software Process Improvement*. Springer, pp. 497–508.

Li, Chuanyi et al. (2018). "Automatically classifying user requests in crowdsourcing requirements engineering". In: *Journal of Systems and Software* 138, pp. 108–123.

ISSN: 0164-1212. DOI: https://doi.org/10.1016/j.jss.2017.12.028. URL: http://www.sciencedirect.com/science/article/pii/S0164121217303096.

Li, Ge et al. (2011). "An engineerable ontology based approach for requirements elicitation in process centered problem domain". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642259746. DOI: 10.1007/978-3-642-25975-3_19.

Li, Ke, Rick G Dewar, and Rob J Pooley (2005). "Towards Semiautomation in Requirements Elicitation: mapping natural language and object-oriented concepts". In: *RE05*, pp. 5–7.

Li, Tong, Fan Zhang, and Dan Wang (2018). "Automatic user preferences elicitation: A data-driven approach". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319772424. DOI: 10.1007/978-3-319-77243-1_21.

Li, Yang et al. (2015). "Automated requirements extraction for scientific software". In: *Procedia Computer Science*. ISBN: 1877-0509. DOI: 10.1016/j.procs.2015.05.326.

Lian, Xiaoli, Jane Cleland-Huang, and Li Zhang (2017). "Mining Associations between Quality Concerns and Functional Requirements". In: *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017*. ISBN: 9781538631911. DOI: 10.1109/RE.2017.68.

Liang, Jianhong and J D Palmer (1994). "A pattern matching and clustering based approach for supporting requirements transformation". In: *Proceedings of IEEE International Conference on Requirements Engineering*. DOI: 10.1109/ICRE.1994.292387.

Liddy, Elizabeth D (2001). "Natural language processing". In:

Lili, Dong et al. (2010). "Research on User Requirements Elicitation Using Text Association Rule". In: *Intelligence Information Processing and Trusted Computing (IPTC), 2010 International Symposium on*. DOI: 10.1109/IPTC.2010.72.

Liu, D et al. (2004). "Natural language requirements analysis and class model generation using UCDA". In: *Innovations in Applied ...* ISSN: 03029743.

Liu, Lin, Tianying Li, and Xiaoxi Kou (2014). "Eliciting relations from natural language requirements documents based on linguistic and statistical analysis". In: *Proceedings - International Computer Software and Applications Conference*. ISBN: 978-1-4799-3575-8. DOI: 10.1109/COMPSAC.2014.27.

López-Martínez, Janeth et al. (2017). *User stories complexity estimation using Bayesian networks for inexperienced developers*. DOI: 10.1007/s10586-017-0996-z.

Loughran, Neil, Américo Sampaio, and Awais Rashid (2006). "From requirements documents to feature models for aspect oriented product line implementation". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3540317805. DOI: 10.1007/11663430_27.

Lucassen, Garm et al. (2015a). "Forging high-quality User Stories: Towards a discipline for Agile Requirements". In: *2015 IEEE 23rd International Requirements Engineering Conference, RE 2015 - Proceedings*. ISBN: 9781467369053. DOI: 10.1109/RE.2015.7320415.

Lucassen, Garm et al. (2015b). "Forging high-quality user stories: towards a discipline for agile requirements". In: *Requirements Engineering Conference (RE), 2015 IEEE 23rd International*. IEEE, pp. 126–135.

Lucassen, Garm et al. (2016a). "Improving agile requirements: the Quality User Story framework and tool". In: *Requirements Engineering*. ISSN: 1432010X. DOI: 10.1007/s00766-016-0250-x.

Lucassen, Garm et al. (2016b). "Visualizing user story requirements at multiple granularity levels via semantic relatedness". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319463964. DOI: 10.1007/978-3-319-46397-1_35.

Lucassen, Garm et al. (2017). "Extracting conceptual models from user stories with Visual Narrator". In: *Requirements Engineering*. ISSN: 1432010X. DOI: 10.1007/s00766-017-0270-1.

Maalej, Walid and Hadeer Nabil (2015a). "Bug report, feature request, or simply praise? on automatically classifying app reviews". In: *Requirements Engineering Conference (RE), 2015 IEEE 23rd International*. IEEE, pp. 116–125.

— (2015b). "Bug report, feature request, or simply praise? On automatically classifying app reviews". In: *2015 IEEE 23rd International Requirements Engineering Conference, RE 2015 - Proceedings*. ISBN: 9781467369053. DOI: 10.1109/RE.2015.7320414.

Maalej, Walid et al. (2016). "On the automatic classification of app reviews". In: *Requirements Engineering*. ISSN: 1432010X. DOI: 10.1007/s00766-016-0251-9.

Maciaszek, Leszek (2007). *Requirements analysis and system design*. Pearson Education.

Madala, Kaushik et al. (2017). "Automated identification of component state transition model elements from requirements". In: *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017*. ISBN: 9781538634882. DOI: 10.1109/REW.2017.73.

Madhan, V., V.K.G. Kalaiselvi, and Donald J.P (2017). "Tool development for formalizing the requirement for the safety critical software engineering process". In: *2017 2nd International Conference on Computing and Communications Technologies (ICCCT)*. ISBN: 978-1-5090-6221-8. DOI: 10.1109/ICCCT2.2017.7972273.

Mahmoud, Anas (2015). "An information theoretic approach for extracting and tracing non-functional requirements". In: *Requirements Engineering Conference (RE), 2015 IEEE 23rd International*. IEEE, pp. 36–45.

Mahmoud, Anas and Doris Carver (2015). "Exploiting online human knowledge in Requirements Engineering". In: *Requirements Engineering Conference (RE), 2015 IEEE 23rd International*. IEEE, pp. 262–267.

Mahmoud, Anas and Grant Williams (2016). "Detecting, classifying, and tracing non-functional software requirements". In: *Requirements Engineering*. ISSN: 1432010X. DOI: 10.1007/s00766-016-0252-8.

Mahmud, N, C Seceleanu, and O Ljungkrantz (2017). "Specification and semantic analysis of embedded systems requirements: From description logic to temporal logic". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISSN: 03029743. DOI: 10.1007/978-3-319-66197-1_21.

Mala, G. S. Anandha and G. V. Uma (2006). "Automatic Construction of Object Oriented Design Models [UML Diagrams] from Natural Language Requirements Specification". In: *PRICAI 2006: Trends in Artificial Intelligence*. Ed. by Qiang Yang and Geoff Webb. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 1155–1159. ISBN: 978-3-540-36668-3.

Manning, Christopher et al. (2014). "The Stanford CoreNLP natural language processing toolkit". In: *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pp. 55–60.

Manning, Christopher D and Hinrich Schütze (1999). *Foundations of statistical natural language processing*. MIT press.

Manrique-Losada, Bell, Carlos M. Zapata-Jaramillo, and Diego A. Burgos (2016). "Re-expressing business processes information from corporate documents into controlled language". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319417530. DOI: 10.1007/978-3-319-41754-7_37.

Marcus, Mitchell P, Mary Ann Marcinkiewicz, and Beatrice Santorini (1993). "Building a large annotated corpus of English: The Penn Treebank". In: *Computational linguistics* 19.2, pp. 313–330.

Martin, Sacha et al. (2002). "Requirements engineering process models in practice". In: *7th Australian workshop on requirements engineering. Deakin University, Melbourne, Australia*, pp. 41–47.

Martin-Rodilla, Patricia and Cesar Gonzalez-Perez (2014). "An ISO/IEC 24744-derived modelling language for discourse analysis". In: *2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS)*. IEEE, pp. 1–10. ISBN: 978-1-4799-2393-9. DOI: 10.1109/RCIS.2014.6861073. URL: http://ieeexplore.ieee.org/document/6861073/.

Martinez, Angel R (2012). "Part-of-speech tagging". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 4.1, pp. 107–113.

Massey, Aaron K. et al. (2013). "Automated text mining for requirements analysis of policy documents". In: *2013 21st IEEE International Requirements Engineering Conference, RE 2013 - Proceedings*. ISBN: 9781467357654. DOI: 10.1109/RE.2013.6636700.

Matsumoto, Yuuma, Sayaka Shirai, and Atsushi Ohnishi (2017). "A Method for Verifying Non-Functional Requirements". In: *Procedia Computer Science*. DOI: 10.1016/j.procs.2017.08.006.

Mazón, Jose Norberto et al. (2007). "Improving the development of data warehouses by enriching dimension hierarchies with WordNet". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3540754733. DOI: 10.1007/978-3-540-75474-9_6.

McZara, Jason et al. (2015). "Software requirements prioritization and selection using linguistic tools and constraint solvers—a controlled experiment". In: *Empirical Software Engineering*. ISSN: 15737616. DOI: 10.1007/s10664-014-9334-8.

Mefteh, Mariem, Nadia Bouassida, and Hanêne Ben-Abdallah (2018). "Towards naturalistic programming: Mapping language-independent requirements to constrained language specifications". In: *Science of Computer Programming* 166, pp. 89–119. ISSN: 0167-6423. DOI: 10.1016/J.SCICO.2018.05.006. URL: https://www.sciencedirect.com/science/article/pii/S0167642318301941.

Meth, Hendrik, Manuel Brhel, and Alexander Maedche (2013). "The state of the art in automated requirements elicitation". In: *Information and Software Technology* 55.10, pp. 1695–1709.

Meth, Hendrik, Alexander Maedche, and Maximilian Einoeder (2013). "Is knowledge power? The role of knowledge in automated requirements elicitation". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642387081. DOI: 10.1007/978-3-642-38709-8_37.

Mezghani, Manel, Juyeon Kang, and Florence Sèdes (2018). "Using k-Means for Redundancy and Inconsistency Detection: Application to Industrial Requirements". In: *Natural Language Processing and Information Systems*. Ed. by Max Silberztein et al. Cham: Springer International Publishing, pp. 501–508. ISBN: 978-3-319-91947-8.

Meziane, Farid, Nikos Athanasakis, and Sophia Ananiadou (2008). "Generating Natural Language specifications from UML class diagrams". In: *Requirements Engineering*. ISSN: 09473602. DOI: 10.1007/s00766-007-0054-0.

Mich, Luisa and Roberto Garigliano (1994). "A linguistic approach to the development of Object Oriented Systems using the NL system LOLITA". In: *Object-Oriented Methodologies and Systems*. Ed. by Elisa Bertino and Susan Urban. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 371–386. ISBN: 978-3-540-48804-0.

— (2002). "NL-OOPS: A requirements analysis tool based on natural language processing". In: *WIT Transactions on Information and Communication Technologies* 28.

Mihalcea, Rada (2011). "Word sense disambiguation". In: *Encyclopedia of machine learning*. Springer, pp. 1027–1030.

Minhas, Nasir Mehmood et al. (2011). "Controlled vocabulary based software requirements classification". In: *2011 5th Malaysian Conference in Software Engineering, MySEC 2011*. ISBN: 9781457715310. DOI: 10.1109/MySEC.2011.6140639.

Misra, Janardan (2016). "Terminological inconsistency analysis of natural language requirements". In: *Information and Software Technology*. ISSN: 09505849. DOI: 10.1016/j.infsof.2015.11.006.

Misra, Janardan and S. Das (2013). "Entity disambiguation in natural language text requirements". In: *Proceedings - Asia-Pacific Software Engineering Conference, APSEC*. ISBN: 9781479921430. DOI: 10.1109/APSEC.2013.41.

Misra, Janardan, Shubhashis Sengupta, and Sanjay Podder (2016). "Topic cohesion preserving requirements clustering". In: *Proceedings of the 5th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering - RAISE '16*. ISBN: 9781450341653. DOI: 10.1145/2896995.2896998.

Mitchell, Ian, Ian Ferguson, and Norman Parrington (1995). "Rapid prototyping: An integrated CASE based approach". In: *OOER '95: Object-Oriented and Entity-Relationship Modeling*. Ed. by Michael P Papazoglou. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 308–317. ISBN: 978-3-540-48527-8.

Mohit, Behrang (2014). "Named entity recognition". In: *Natural language processing of semitic languages*. Springer, pp. 221–245.

Montgomery, Lloyd et al. (2018). *Customer support ticket escalation prediction using feature engineering*. DOI: 10.1007/s00766-018-0292-3.

Morales-Ramirez, Itzel, F M Kifetew, and Anna Perini (2017). "Analysis of online discussions in support of requirements discovery". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISSN: 03029743. DOI: 10.1007/978-3-319-59536-8_11.

Morales-Ramirez, Itzel, Anna Perini, and Mariano Ceccato (2015). "Towards supporting the analysis of online discussions in OSS communities: A speech-act based approach". In: *Lecture Notes in Business Information Processing*. ISBN: 9783319192697. DOI: 10.1007/978-3-319-19270-3_14.

Nadeau, David and Satoshi Sekine (2007). "A survey of named entity recognition and classification". In: *Lingvisticae Investigationes* 30.1, pp. 3–26.

Nadkarni, Prakash M, Lucila Ohno-Machado, and Wendy W Chapman (2011). "Natural language processing: an introduction". In: *Journal of the American Medical Informatics Association* 18.5, pp. 544–551.

Narouei, M and H Takabi (2015). "Automatic top-down role engineering framework using natural language processing techniques". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISSN: 03029743. DOI: 10.1007/978-3-319-24018-3_9.

Nassar, I N and F T Khamayseh (2015). "Constructing activity diagrams from Arabic user requirements using Natural Language Processing tool". In: *2015 6th International Conference on Information and Communication Systems (ICICS)*. DOI: 10.1109/IACS.2015.7103200.

Nasukawa, Tetsuya and Jeonghee Yi (2003). "Sentiment analysis: Capturing favorability using natural language processing". In: *Proceedings of the 2nd international conference on Knowledge capture*. ACM, pp. 70–77.

Nazir, Farhana et al. (2017). "The applications of natural language processing (NLP) for software requirement engineering-a systematic literature review". In: *International Conference on Information Science and Applications*. Springer, pp. 485–493.

Ninaus, Gerald et al. (2014). "Content-based recommendation techniques for requirements engineering". In: *2014 IEEE 1st International Workshop on Artificial Intelligence for Requirements Engineering, AIRE 2014 - Proceedings*. ISBN: 9781479963553. DOI: 10.1109/AIRE.2014.6894853.

Niu, Nan and Steve Easterbrook (2007). "So, you think you know others' goals? A repertory grid study". In: *IEEE software* 24.2.

Noaeen, Mohammad, Zahra Shakeri Hossein Abad, and Behrouz Homayoun Far (2017a). "Let's Hear it from RETTA: A Requirements Elicitation Tool for TrAffic Management Systems". In: *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference, RE 2017*. ISBN: 9781538631911. DOI: 10.1109/RE.2017.78. arXiv: 1707.01927.

— (2017b). "Let's hear it from RETTA: A Requirements Elicitation Tool for TrAffic management systems". In: *Requirements Engineering Conference (RE), 2017 IEEE 25th International*. IEEE, pp. 450–451.

Nuseibeh, Bashar and Steve Easterbrook (2000). "Requirements engineering: a roadmap". In: *Proceedings of the Conference on the Future of Software Engineering*. ACM, pp. 35–46.

Ochodek, Mirosław and Jerzy Nawrocki (2008). "Automatic transactions identification in use cases". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3540852786. DOI: 10.1007/978-3-540-85279-7_5.

Omoronyia, Inah et al. (2010). "A domain ontology building process for guiding requirements elicitation". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3642141919. DOI: 10.1007/978-3-642-14192-8_18. arXiv: 9780201398298.

Osborne, Miles and C.K. MacNish (1996). "Processing natural language software requirement specifications". In: *Proceedings of the Second International Conference on Requirements Engineering*. DOI: 10.1109/ICRE.1996.491451.

Ott, Daniel (2013). "Automatic requirement categorization of large natural language specifications at Mercedes-Benz for review improvements". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642374210. DOI: 10.1007/978-3-642-37422-7_4.

Paetsch, Frauke, Armin Eberlein, and Frank Maurer (2003). "Requirements engineering and agile software development". In: *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on*. IEEE, pp. 308–313.

Pak, Alexander and Patrick Paroubek (2010). "Twitter as a corpus for sentiment analysis and opinion mining." In: *LREc*. Vol. 10. 2010.

Papanikolaou, Nick (2012). "Natural language processing of rules and regulations for compliance in the cloud". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642336140. DOI: 10.1007/978-3-642-33615-7_11.

Parent, Milena M and David L Deephouse (2007). "A case study of stakeholder identification and prioritization by managers". In: *Journal of business ethics* 75.1, pp. 1–23.

Park, Ki Seon, Dong Un An, and Yong Seok Lee (2010). "Anaphora resolution system for natural language requirements document in Korean". In: *ICIC 2010 - 3rd International Conference on Information and Computing*. ISBN: 9780769540474. DOI: 10.1109/ICIC.2010.9.

Park, Sooyong et al. (2000). "Implementation of an efficient requirements-analysis supporting system using similarity measure techniques". In: *Information and Software Technology*. ISSN: 09505849. DOI: 10.1016/S0950-5849(99)00102-0.

Parreira Júnior, Paulo Afonso and Rosângela Aparecida Dellosso Penteado (2018). "ObasCId(-Tool): an ontologically based approach for concern identification and classification and its computational support". In: *Journal of the Brazilian Computer Society* 24.1, p. 3. ISSN: 1678-4804. DOI: 10.1186/s13173-017-0067-6. URL: https://doi.org/10.1186/s13173-017-0067-6.

Peng, Zhenlian et al. (2017). "An approach of extracting feature requests from app reviews". In: *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*. ISBN: 978-3-319-59287-9 978-3-319-59288-6. DOI: 10.1007/978-3-319-59288-6_28.

Petersen, Kai et al. (2008). "Systematic Mapping Studies in Software Engineering." In: *EASE*. Vol. 8, pp. 68–77.

Petrov, Slav, Dipanjan Das, and Ryan McDonald (2011). "A universal part-of-speech tagset". In: *arXiv preprint arXiv:1104.2086*.

Pierce, JR et al. (1966). "Computers in translation and linguistics (ALPAC report). report 1416". In: *National Academy of Sciences/National Research Council*.

Pinquié, Romain et al. (2016). "Natural Language Processing of Requirements for Model-Based Product Design with ENOVIA/CATIA V6". In: *Product Lifecycle Management in the Era of Internet of Things*. Ed. by Abdelaziz Bouras et al. Cham: Springer International Publishing, pp. 205–215. ISBN: 978-3-319-33111-9.

Pohl, Klaus (2010). *Requirements engineering: fundamentals, principles, and techniques*. Springer Publishing Company, Incorporated.

Portugal, Roxana Lisette Quintanilla, Julio Cesar Sampaio Do Prado Leite, and Eduardo Almentero (2015). "Time-constrained requirements elicitation: Reusing GitHub content". In: *1st International Workshop on Just-in-Time Requirements Engineering, JIT RE 2015 - Proceedings*. ISBN: 9781509001194. DOI: 10.1109/JITRE.2015.7330171.

Potts, C. and K. Takahashi (1993). "An active hypertext model for system requirements". In: *Proceedings of 1993 IEEE 7th International Workshop on Software Specification and Design*. DOI: 10.1109/IWSSD.1993.315512.

Pruski, Piotr et al. (2014). "Tiqi: Towards natural language trace queries". In: *Requirements Engineering Conference (RE), 2014 IEEE 22nd International*. IEEE, pp. 123–132.

Quené, Hugo (1992). "Durational cues for word segmentation in Dutch." In: *Journal of Phonetics*.

Quirchmayr, Thomas et al. (2018). *Semi-automatic rule-based domain terminology and software feature-relevant information extraction from natural language user manuals: An approach and evaluation at Roche Diagnostics GmbH*. DOI: 10.1007/s10664-018-9597-6.

Rago, Alejandro, Claudia Marcos, and J. Andrés Diaz-Pace (2013). "Uncovering quality-attribute concerns in use case specifications via early aspect mining". In: *Requirements Engineering*. ISSN: 09473602. DOI: 10.1007/s00766-011-0142-z.

Rago, Alejandro, Claudia Marcos, and J. Andres Diaz-Pace (2016a). "Assisting requirements analysts to find latent concerns with REAssistant". In: *Automated Software Engineering*. ISSN: 15737535. DOI: 10.1007/s10515-014-0156-0.

Rago, Alejandro, Claudia Marcos, and J. Andrés Diaz-Pace (2016b). "Identifying duplicate functionality in textual use cases by aligning semantic actions". In: *Software and Systems Modeling*. ISSN: 16191374. DOI: 10.1007/s10270-014-0431-3.

Rashwan, Abderahman (2012). *Semantic Analysis of Functional and Non-functional Requirements in Software Requirements Specifications*. DOI: 10.1007/978-3-642-30353-1_42.

Raven, Mary Elizabeth and Alicia Flanders (1996). "Using contextual inquiry to learn about your audiences". In: *ACM SIGDOC Asterisk Journal of Computer Documentation* 20.1, pp. 1–13.

Reynolds, Thomas J and Jonathan Gutman (1988). "Laddering theory, method, analysis, and interpretation". In: *Journal of advertising research* 28.1, pp. 11–31.

Riaz, Maria et al. (2014a). "Hidden in plain sight: Automatically identifying security requirements from natural language artifacts". In: *Requirements Engineering Conference (RE), 2014 IEEE 22nd International*. IEEE, pp. 183–192.

— (2014b). "Hidden in plain sight: Automatically identifying security requirements from natural language artifacts". In: *2014 IEEE 22nd International Requirements Engineering Conference, RE 2014 - Proceedings*. ISBN: 9781479930333. DOI: 10.1109/RE.2014.6912260.

Robeer, Marcel et al. (2016a). "Automated Extraction of Conceptual Models from User Stories via NLP". In: *Proceedings - 2016 IEEE 24th International Requirements Engineering Conference, RE 2016*. ISBN: 9781509041213. DOI: 10.1109/RE.2016.40. arXiv: 1512.05558.

Robeer, Marcel et al. (2016b). "Automated extraction of conceptual models from user stories via NLP". In: *Requirements engineering conference (RE), 2016 IEEE 24th international*. IEEE, pp. 196–205.

Rolland, Colette and Camille Ben Achour (1998). "Guiding the construction of textual use case specifications". In: *Data and Knowledge Engineering*. ISSN: 0169023X. DOI: 10.1016/S0169-023X(97)86223-4.

Rolland, Colette and C Proix (1992). "A natural language Approach for requirements engineering". In: *Lecture Notes in Computer Science, 4th International Conference CAiSE '92 Manchester, UK, May 12–15, 1992 Proceedings*. ISSN: 16113349. DOI: 10.1007/BFb0035136.

Saaty, Thomas L (2008). "Decision making with the analytic hierarchy process". In: *International journal of services sciences* 1.1, pp. 83–98.

Saeki, Motoshi and Haruhiko Kaiya (2008). "Supporting the elicitation of requirements compliant with regulations". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3540695338. DOI: 10.1007/978-3-540-69534-9_18.

Saint-Dizier, Patrick (2017). "Mining Incoherent Requirements in Technical Specifications". In: *Natural Language Processing and Information Systems*. Ed. by Flavius Frasincar et al. Cham: Springer International Publishing, pp. 71–83. ISBN: 978-3-319-59569-6.

— (2018). "Mining incoherent requirements in technical specifications: Analysis and implementation". In: *Data & Knowledge Engineering*. ISSN: 0169-023X. DOI: 10.1016/J.DATAK.2018.05.006. URL: https://www.sciencedirect.com/science/article/pii/S0169023X17305232.

Sakthivel, Sachidanandam (1991). "A survey of requirements verification techniques". In: *Journal of Information Technology* 6.2, pp. 68–79.

Salemi, Samin, Ali Selamat, and Marek Penhaker (2016). "A model transformation framework to increase OCL usability". In: *Journal of King Saud University - Computer and Information Sciences*. ISSN: 22131248. DOI: 10.1016/j.jksuci.2015.04.002.

Sannier, Nicolas and Benoit Baudry (2014). "INCREMENT: A Mixed MDE-IR Approach for Regulatory Requirements Modeling and Analysis". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319058429. DOI: 10.1007/978-3-319-05843-6_11.

Sannier, Nicolas et al. (2016). "Automated classification of legal cross references based on semantic intent". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319302812. DOI: 10.1007/978-3-319-30282-9_8.

— (2017a). "An automated framework for detection and resolution of cross references in legal texts". In: *Requirements Engineering*. ISSN: 1432010X. DOI: 10.1007/s00766-015-0241-3.

Sannier, Nicolas et al. (2017b). "Legal Markup Generation in the Large: An Experience Report". In: *Requirements Engineering Conference (RE), 2017 IEEE 25th International*. IEEE, pp. 302–311.

Santos, Joao et al. (2008). "Generating requirements analysis models from textual requirements". In: *2008 1st International Workshop on Managing Requirements Knowledge, MARK'08*. ISBN: 9780769536279. DOI: 10.1109/MARK.2008.4.

Santos Soares, Michel dos, Jos Vrancken, and Alexander Verbraeck (2011). "User requirements modeling and analysis of software-intensive systems". In: *Journal of Systems and Software* 84.2, pp. 328–339.

Sardinha, Alberto et al. (2013). "EA-Analyzer: Automating conflict detection in a large set of textual aspect-oriented requirements". In: *Automated Software Engineering*. ISBN: 9780769538914. DOI: 10.1007/s10515-012-0106-7.

Sateli, B, E Angius, and R Witte (2013a). "The ReqWiki Approach for Collaborative Software Requirements Engineering with Integrated Text Analysis Support". In: *Proceedings of the 37th IEEE Annual Computer Software and Applications Conference (COMPSAC)*. ISBN: 978-0-7695-4986-6. DOI: 10.1109/COMPSAC.2013.68.

Sateli, Bahar, Elian Angius, and René Witte (2013b). "The reqwiki approach for collaborative software requirements engineering with integrated text analysis support". In: *Computer Software and Applications Conference (COMPSAC), 2013 IEEE 37th Annual*. IEEE, pp. 405–414.

Sawyer, Pete, Ricardo Gacitua, and Andrew Stone (2008). "Profiling and tracing stakeholder needs". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 3540897771. DOI: 10.1007/978-3-540-89778-1_15.

Sawyer, Pete, Paul Rayson, and Ken Cosh (2005). "Shallow knowledge as an aid to deep understanding in early phase requirements engineering". In: *IEEE Transactions on Software Engineering*. ISSN: 00985589. DOI: 10.1109/TSE.2005.129.

Sayao, M, A H Filho, and H A do Prado (2008). "Requirements engineering for distributed development using software agents". In: *Advances in Conceptual Modeling-Challenges and Opportunities. ER 2008 Workshops CMLSA, ECDM, FP-UML,M2AS,RIGiM, SeCoGIS, WISM, 20-23 Oct. 2008*.

Schalkoff, Robert J (1997). *Artificial neural networks*. Vol. 1. McGraw-Hill New York.

Selway, Matt, Wolfgang Mayer, and Markus Stumptner (2014). "Semantic interpretation of requirements through cognitive grammar and configuration". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence* *and Lecture Notes in Bioinformatics)*. ISSN: 16113349. DOI: 10.1007/978-3-319-13560-1.

Sharma, Richa, Jaspreet Bhatia, and K. K. Biswas (2014). "Automated identification of business rules in requirements documents". In: *Souvenir of the 2014 IEEE International Advance Computing Conference, IACC 2014*. ISBN: VO -. DOI: 10.1109/IAdCC.2014.6779538.

Sharma, Richa and K. K. Biswas (2015). "Generating Logical Representations for Natural Language Requirements Using Syntactic Dependencies and Norm Analysis Patterns". In: *Natural Language Processing and Information Systems*. Ed. by Chris Biemann et al. Cham: Springer International Publishing, pp. 432–436. ISBN: 978-3-319-19581-0.

Sharma, Richa, Sarita Gulia, and K. K. Biswas (2014). "Automated Generation of Activity and Sequence Diagrams from Natural Language Requirements". In: *9th International Conference on Evaluation of Novel Approaches to Software Engineering, ENASE 2014*. DOI: 10.5220/0004893600690077.

Sharma, Richa, Nidhi Sharma, and K. K. Biswas (2017). "Machine Learning for Detecting Pronominal Anaphora Ambiguity in NL Requirements". In: *Proceedings - 4th International Conference on Applied Computing and Information Technology, 3rd International Conference on Computational Science/Intelligence and Applied Informatics, 1st International Conference on Big Data, Cloud Computing, Data Science*. ISBN: 9781509048717. DOI: 10.1109/ACIT-CSII-BCD.2016.043.

Sharma, Richa, Pratyoush K. Srivastava, and Kanad K. Biswas (2015). "From natural language requirements to UML class diagrams". In: *2015 IEEE Second International Workshop on Artificial Intelligence for Requirements Engineering (AIRE)*. ISBN: 978-1-5090-0125-5. DOI: 10.1109/AIRE.2015.7337625.

Sharp, Helen, Anthony Finkelstein, and Galal Galal (1999). "Stakeholder identification in the requirements engineering process". In: *Database and expert systems applications, 1999. proceedings. tenth international workshop on*. Ieee, pp. 387–391.

Shibaoka, Masayuki, Haruhiko Kaiya, and Motoshi Saeki (2007). "GOORE : Goal-Oriented and Ontology Driven Requirements Elicitation Method". In: *Advances in Conceptual Modeling – Foundations and Applications*. Ed. by Jean-Luc Hainaut et al. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 225–234. ISBN: 978-3-540-76292-8.

Shukur, Z, A M Zin, and A Ban (2002). "M2Z: A tool for translating a natural language software specification into Z". In: *Formal Methods and Software Engineering, Proceedings*. ISSN: 16113349.

Sillaber, Christian and Ruth Breu (2014). "Improving Near-Duplicate Detection in Multi-Layered Collaborative Requirements Engineering Discussions Through Discussion Clustering". In: *The 8th International Conference on Knowledge Management in Organizations*. Ed. by Lorna Uden et al. Dordrecht: Springer Netherlands, pp. 249–261. ISBN: 978-94-007-7287-8.

Silvat, Alberto Tavares da and Luis Alfredo V Carvalho (2006). "A knowledge representation semantic network for a natural language syntactic analyzer based on the UML". In: *Professional Practice in Artificial Intelligence*. ISBN: 0-387-34655-4.

Silveira, Clara et al. (2005). "Wiki based requirements documentation of generic software products". In: *Proceedings of the 10th Australian Workshop on Requirements Engineering (AWRE)*, pp. 42–51.

Simko, Viliam et al. (2012). "Verifying temporal properties of use-cases in natural language". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642357428. DOI: 10.1007/978-3-642-35743-5_21.

Sinha, Avik et al. (2010a). "Extending Automated Analysis of Natural Language Use Cases to Other Languages". In: *2010 18th IEEE International Requirements Engineering Conference*. ISSN: 1090-705X. DOI: `10.1109/RE.2010.52`.

— (2010b). "Extending automated analysis of natural language use cases to other languages". In: *Requirements Engineering Conference (RE), 2010 18th IEEE International*. IEEE, pp. 364–369.

Skoutas, Dimitrios and Alkis Simitsis (2007). "Flexible and customizable NL representation of requirements for ETL processes". In: *12th International Conference on Applications of Natural Language to Information Systems NLDB 2007 June 27 2007 June 29 2007*. ISSN: 03029743.

Soares, Hécio A. and Raimundo S. Moura (2015). "A methodology to guide writing Software Requirements Specification document". In: *Proceedings - 2015 41st Latin American Computing Conference, CLEI 2015*. ISBN: 9781467391436. DOI: `10.1109/CLEI.2015.7360001`.

Socher, Richard, Yoshua Bengio, and Christopher D Manning (2012). "Deep learning for NLP (without magic)". In: *Tutorial Abstracts of ACL 2012*. Association for Computational Linguistics, pp. 5–5.

Soni, Anuja and Vibha Gaur (2016). "A methodological approach to identify type of dependency from user requirements". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319420882. DOI: `10.1007/978-3-319-42089-9_27`.

Sproat, Richard et al. (1996). "A stochastic finite-state word-segmentation algorithm for Chinese". In: *Computational linguistics* 22.3, pp. 377–404.

Spyns, Peter (1996). "Natural language processing in medicine: an overview". In: *Methods of information in medicine* 35.04/05, pp. 285–301.

Stålhane, Tor and Tormod Wien (2014). "The DODT tool applied to sub-sea software". In: *2014 IEEE 22nd International Requirements Engineering Conference, RE 2014 - Proceedings*. ISBN: 9781479930333. DOI: `10.1109/RE.2014.6912293`.

Steinberger, Michal, Iris Reinhartz-Berger, and Amir Tomer (2016). "A tool for analyzing variability based on functional requirements and testing artifacts". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319477169. DOI: `10.1007/978-3-319-47717-6_21`.

Stone, Andrew and Peter Sawyer (2006). "Identifying tacit knowledge-based requirements". In: *IEE Proceedings-Software* 153.6, pp. 211–218.

Subha, R. and S. Palaniswami (2013). "Quality factor assessment and text summarization of unambiguous natural language requirements". In: *Communications in Computer and Information Science*. ISBN: 9783642363207. DOI: `10.1007/978-3-642-36321-4-12`.

Sun, Dong and Rong Peng (2015). "A scenario model aggregation approach for mobile app requirements evolution based on user comments". In: *Communications in Computer and Information Science*. ISBN: 978-3-662-48633-7. DOI: `10.1007/978-3-662-48634-4_6`.

Sutcliffe, Alistair and Pete Sawyer (2013). "Requirements elicitation: Towards the unknown unknowns". In: *Requirements Engineering Conference (RE), 2013 21st IEEE International*. IEEE, pp. 92–104.

Talib, Muhammad Ramzan et al. (2017). "TEXT MINING OF JUDICIAL SYSTEM's CORPORA VIA CLAUSE ELEMENTS". In: *International Journal on Information Technologies & Security* 9.3.

Tang, Yan and Kunwu Feng (2009). "An Expert System Based Approach to Modeling and Selecting Requirement Engineering Techniques". In: *Proceedings of the*

International Conference on Web Information Systems and Mining. ISBN: 978-3-642-05249-1. DOI: 10.1007/978-3-642-05250-7_3.

Terawaki, Yuki (2011). "Supporting of requirements elicitation for ensuring services of information systems used for education". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642217920. DOI: 10.1007/978-3-642-21793-7_7.

— (2013). "Framework for quantitatively evaluating the quality requirements of software system". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISSN: 03029743. DOI: 10.1007/978-3-642-39209-2-44.

Thovex, Christophe and Francky Trichet (2014). "Automatic Building of Socio-semantic Networks for Requirements Analysis". In: *Knowledge Science, Engineering and Management*. Ed. by Robert Buchmann, Claudiu Vasile Kifor, and Jian Yu. Cham: Springer International Publishing, pp. 37–48. ISBN: 978-3-319-12096-6.

Tiwari, Saurabh and Mayank Laddha (2017). "UCAnalyzer: A Tool to Analyze Use Case Textual Descriptions". In: *Requirements Engineering Conference (RE), 2017 IEEE 25th International*. IEEE, pp. 448–449.

Tjong, Sri Fatimah and Daniel M. Berry (2013). "The design of SREE - A prototype potential ambiguity finder for requirements specifications and lessons learned". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642374210. DOI: 10.1007/978-3-642-37422-7_6.

Todd, Kurtis D. and Jane Cleland-Huang (2008). "A dynamic approach for managing stakeholder credentials for lightweight, collaborative, requirements elicitation". In: *2008 1st International Workshop on Managing Requirements Knowledge, MARK'08*. ISBN: 9780769536279. DOI: 10.1109/MARK.2008.1.

Turing, Alan (1950). "Computing machinery and intelligence". In: *Mind* 59.236, p. 433.

Umber, Ashfa, Imran Sarwar Bajwa, and M Asif Naeem (2011). "NL-Based Automated Software Requirements Elicitation and Specification". In: *Advances in Computing and Communications: Proceedings of the First International Conference on Advances in Computing and Communications (ACC 2011)*. ISBN: 978-3-642-22713-4. DOI: 10.1007/978-3-642-22714-1_4.

Van Rooijen, Lorijn et al. (2017). "From user demand to software service: Using machine learning to automate the requirements specification process". In: *Proceedings - 2017 IEEE 25th International Requirements Engineering Conference Workshops, REW 2017*. ISBN: 9781538634882. DOI: 10.1109/REW.2017.26.

Veerappa, Varsha and Rachel Harrison (2013). "Assessing the maturity of requirements through argumentation: A good enough approach". In: *2013 28th IEEE/ACM International Conference on Automated Software Engineering, ASE 2013 - Proceedings*. ISBN: 9781479902156. DOI: 10.1109/ASE.2013.6693131.

Vidya Sagar, Vidhu Bhala R. and S. Abirami (2014). "Conceptual modeling of natural language functional requirements". In: *Journal of Systems and Software*. ISSN: 01641212. DOI: 10.1016/j.jss.2013.08.036.

Walter, Benedikt et al. (2017). "A Formalization Method to Process Structured Natural Language to Logic Expressions to Detect Redundant Specification and Test Statements". In: *Requirements Engineering Conference (RE), 2017 IEEE 25th International*. IEEE, pp. 263–272.

Wang, Min et al. (2013a). "An algorithm for transforming design text ROM diagram into FBS model". In: *Computers in Industry*. ISSN: 01663615. DOI: 10.1016/j.compind.2013.02.007.

Wang, Wentao et al. (2017). "A Regression Model Based Approach for Identifying Security Requirements in Open Source Software Development". In: *2017 IEEE 25th International Requirements Engineering Conference Workshops (REW)*. IEEE, pp. 443–446. ISBN: 978-1-5386-3488-2. DOI: 10.1109/REW.2017.56. URL: http://ieeexplore.ieee.org/document/8054894/.

Wang, Ye, Bo Jiang, and Ting Wang (2017). "Using workflow patterns to model and validate service requirements". In: *Proceedings - 2016 IEEE 24th International Requirements Engineering Conference Workshops, REW 2016*. ISBN: 9781509036943. DOI: 10.1109/REW.2016.18.

Wang, Yinglin and Jianzhang Zhang (2017). "Experiment on automatic functional requirements analysis with the EFRFs semantic cases". In: *PIC 2016 - Proceedings of the 2016 IEEE International Conference on Progress in Informatics and Computing*. ISBN: 9781509034833. DOI: 10.1109/PIC.2016.7949577.

Wang, Yue et al. (2013b). "Automatic detection of ambiguous terminology for software requirements". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783642388231. DOI: 10.1007/978-3-642-38824-8_3.

Wasson, Kimberly S. et al. (2005). "Using occurrence properties of defect report data to improve requirements". In: *Requirements Engineering, 2005. Proceedings. 13th IEEE International Conference on*. ISSN: 1090705X. DOI: 10.1109/RE.2005.77.

Weber-Jahnke, Jens H. and Adeniyi Onabajo (2009). "Finding defects in natural language confidentiality requirements". In: *Proceedings of the IEEE International Conference on Requirements Engineering*. ISBN: 9780769537610. DOI: 10.1109/RE.2009.41.

Weston, Nathan, Ruzanna Chitchyan, and Awais Rashid (2009). "Formal semantic conflict detection in aspect-oriented requirements". In: *Requirements Engineering*. ISBN: 09473602. DOI: 10.1007/s00766-009-0083-y.

Wiesner, Stefan et al. (2014). "Requirements Engineering for Cyber-Physical Systems". In: *Advances in Production Management Systems: Innovative and Knowledge-Based Production Management in a Global-Local World*. ISSN: 18684238. DOI: 10.1007/978-3-662-44739-0_35.

Wilkinson, Nancy M et al. (1998). "Using CRC Cards". In: *Cambridge Books*.

Williams, Grant and Anas Mahmoud (2017). "Mining Twitter Feeds for Software User Requirements". In: *Requirements Engineering Conference (RE), 2017 IEEE 25th International*. IEEE, pp. 1–10.

Wilmink, Martin and Christoph Bockisch (2017). "On the ability of lightweight checks to detect ambiguity in requirements documentation". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319540443. DOI: 10.1007/978-3-319-54045-0_23.

Winkler, Jonas Paul and Andreas Vogelsang (2017). "Automatic classification of requirements based on convolutional neural networks". In: *Proceedings - 2016 IEEE 24th International Requirements Engineering Conference Workshops, REW 2016*. ISBN: 9781509036943. DOI: 10.1109/REW.2016.16.

— (2018). "Using Tools to Assist Identification of Non-requirements in Requirements Specifications – A Controlled Experiment". In: *Requirements Engineering: Foundation for Software Quality*. Ed. by Erik Kamsties, Jennifer Horkoff, and Fabiano Dalpiaz. Cham: Springer International Publishing, pp. 57–71. ISBN: 978-3-319-77243-1.

Winograd, Terry (1971). *Procedures as a representation for data in a computer program for understanding natural language*. Tech. rep. MASSACHUSETTS INST OF TECH CAMBRIDGE PROJECT MAC.

— (1972). "Understanding natural language". In: *Cognitive psychology* 3.1, pp. 1–191.

Winter, Karolin et al. (2017). "Characterizing regulatory documents and guidelines based on text mining". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. ISBN: 9783319694610. DOI: 10.1007/978-3-319-69462-7_1.

Wnuk, Krzysztof, Martin Höst, and Björn Regnell (2012). "Replication of an experiment on linguistic tool support for consolidation of requirements from multiple sources". In: *Empirical Software Engineering*. ISSN: 13823256. DOI: 10.1007/s10664-011-9174-8.

Wüest, Dustin, Norbert Seyff, and Martin Glinz (2015). "Sketching and notation creation with FlexiSketch Team: Evaluating a new means for collaborative requirements elicitation". In: *Requirements Engineering Conference (RE), 2015 IEEE 23rd International*. IEEE, pp. 186–195.

Yandell, Mark D and William H Majoros (2002). "Genomics and natural language processing". In: *Nature Reviews Genetics* 3.8, p. 601.

Yang, Hui et al. (2011). "Analysing anaphoric ambiguity in natural language requirements". In: *Requirements Engineering*. ISSN: 09473602. DOI: 10.1007/s00766-011-0119-y.

Yang, Hui et al. (2012a). "Speculative requirements: Automatic detection of uncertainty in natural language requirements". In: *Requirements Engineering Conference (RE), 2012 20th IEEE International*. IEEE, pp. 11–20.

Yang, Hui et al. (2012b). "Speculative requirements: Automatic detection of uncertainty in natural language requirements". In: *2012 20th IEEE International Requirements Engineering Conference, RE 2012 - Proceedings*. ISBN: 9781467327855. DOI: 10.1109/RE.2012.6345795.

Young, Jessica D. (2011). "Commitment analysis to operationalize software requirements from privacy policies". In: *Requirements Engineering*. ISSN: 09473602. DOI: 10.1007/s00766-010-0108-6.

Zachos, Konstantinos et al. (2007). "Discovering web services to specify more complete system requirements". In: *Lecture Notes in Computer Science*. ISSN: 0302-9743; 978-3-540-72987-7. DOI: 10.1007/978-3-540-72988-4{_}11.

Zave, Pamela (1997). "Classification of research efforts in requirements engineering". In: *ACM Computing Surveys (CSUR)* 29.4, pp. 315–321.

Zeni, Nicola, Luisa Mich, and John Mylopoulos (2016). "GaiusT 2.0: Evolution of a framework for annotating legal documents". In: *Communications in Computer and Information Science*. ISBN: 9783319491561. DOI: 10.1007/978-3-319-49157-8_4.

Zhang, Tao and Byungjeong Lee (2010). "Complementary classification techniques based personalized software requirements retrieval with semantic ontology and user feedback". In: *Proceedings - 10th IEEE International Conference on Computer and Information Technology, CIT-2010, 7th IEEE International Conference on Embedded Software and Systems, ICESS-2010, ScalCom-2010*. ISBN: 9780769541082. DOI: 10.1109/CIT.2010.243.

Zhang, Zheying (2007). "Effective requirements development-A comparison of requirements elicitation techniques". In: *Software Quality Management XV: Software Quality in the Knowledge Society, E. Berki, J. Nummenmaa, I. Sunley, M. Ross and G. Staples (Ed.) British Computer Society*, pp. 225–240.

Zhou, Xin et al. (2016). "A map of threats to validity of systematic literature reviews in software engineering". In: *Software Engineering Conference (APSEC), 2016 23rd Asia-Pacific*. IEEE, pp. 153–160.

Zowghi, Didar and Chad Coulin (2005). "Requirements elicitation: A survey of techniques, approaches, and tools". In: *Engineering and managing software requirements*. Springer, pp. 19–46.