TAXONOMY EXTRACTION FROM AUTOMOTIVE NATURAL LANGUAGE REQUIREMENTS USING UNSUPERVISED LEARNING

Martin Ringsquandl[1] and Mathias Schraps[2]

[1]University of Augsburg, Universitätsstraße 2, Augsburg, Germany
[2]Audi Electronics Venture GmbH, Sachsstraße 20, Gaimersheim, Germany

ABSTRACT

In this paper we present a novel approach to semi-automatically learn concept hierarchies from natural language requirements of the automotive industry. The approach is based on the distributional hypothesis and the special characteristics of domain-specific German compounds. We extract taxonomies by using clustering techniques in combination with general thesauri. Such a taxonomy can be used to support requirements engineering in early stages by providing a common system understanding and an agreed-upon terminology. This work is part of an ontology-driven requirements engineering process, which builds on top of the taxonomy. Evaluation shows that this taxonomy extraction approach outperforms common hierarchical clustering techniques.

KEYWORDS

Ontology Learning, Requirements Analysis, NLP, Clustering , Taxonomy Extraction

1. INTRODUCTION

Nowadays automotive system development processes have to deal with increasing complexity and distributed functionalities depending on new features, novel technologies, customer needs and development conditions (e.g., functional safety). Therefore, the impacts to requirements engineering (RE) activities are becoming more extensive. Especially the formulation of textual requirements is a challenging task, because the communication between different stakeholders (e.g., software developers, architects and testers as well as domain experts like electronic engineers) is still based on natural language. Therefore, these textual requirements are inherently fuzzy and ambiguous, but natural language is suitable to formulate their common understanding. The formulation/documentation, validation and management of such requirements is time-consuming and error-prone.

Primarily, terminologies are crucial in order to agree on a common nomenclature and to write requirements specifications, which are understandable and leave no room for interpretation. This results in less effort to modify or validate requirements.

Taxonomies can be used to build up such terminologies to enable knowledge search and reusability of terms in order to write unambiguous textual requirements. A well-defined taxonomy can support the formulation of requirements by providing a common system understanding and ensuring consistent term usage. Furthermore, it is possible to add relations and manage these dependencies between concepts in the taxonomy and their corresponding usage in

DOI : 10.5121/ijnlc.2014.3403

requirement statements in order to build up a requirements ontology to support RE tasks in early phases of automotive software development [1]. Once this knowledge is specified, it can be used to accompany the whole RE process of a project and possibly also serve as basis for future projects.

Manual construction of ontologies is an elaborate task, which should be guided by corresponding systems. We present a novel approach to learn concept hierarchies semi-automatically from text corpora consisting of representative textual requirement specifications of the automotive industry.

The structure of this paper is as follows. Section 2 surveys related work on ontology extraction and ontology-driven requirements engineering. In section 3, the NLP-pipeline and taxonomy clustering methods are presented. Section 4 gives details about the structure and the population of the requirements ontology. Section 5 focusses on the evaluation and discussion of our results. Section 6 provides a conclusion and outlines possible future work with this approach.

2. RELATED WORK

Basically, there are two areas of related work:

- Ontology extraction from text documents
- Ontology-driven requirements engineering

Due to the amount of research in both fields, this section summarizes only related work which is sufficiently close to the presented paper.

2.1. Related Work on Ontology Extraction

Cimiano and Völker [2] presented the Text2Onto framework for ontology extraction from text. This framework contains a collection of algorithms to extract concepts, concept instances and relations. These algorithms mostly rely on basic patterns and occurrence probabilities. Concept extraction is done by weighting terms according to their frequencies, whereas taxonomy extraction relies on the hypernym structure of WordNet.

In another work, Cimiano et al. [3] introduced formal concept analysis to learn taxonomies and compared its performance to hierarchical clustering.

More recently, Yang and Callan [4] investigated the usefulness of various features, e.g., contextual information and lexical-syntactic patterns, to induce taxonomic relationships between pairs of terms.

The OntoGain system for unsupervised ontology extraction from text extended the approach of Text2Onto [5]. Concepts are clustered by a hierarchical agglomerative algorithm, which relies on a lexical similarity measure. In their evaluation, they found that OntoGain showed better performance compared to Text2Onto in terms of taxonomy clustering.

All of the approaches mentioned above do not take account of the special nature of natural language requirements. Their design rather focuses on very large text corpora (e.g., collections of documents from the Web). The approach in this work directly uses characteristics of German requirements documents to cluster concepts.

2.2. Related Work on Ontology-Driven RE

Ontology-driven RE tries to establish a synergy between ontological domain modeling and requirements specification. There has been an increasing interest in research devoted to utilizing ontologies in RE [6–8].

The Ontology-driven Requirements Methodology (OntoREM) system, developed by Kossmann et al. [9], presents a metamodel to define knowledge about RE specifications. It models information about stakeholders, knowledge about the problem domain and general RE knowledge. However, the domain ontologies still need to be hand-crafted.

Mutual approaches try to extract ontologies from natural language requirements with the goal to further support RE. In his dissertation, Leonid Kof uses an approach similar to the ASIUM system to extract ontologies from system specifications written in natural language [10, 11].
Ilieva and Ormandjieva [12] employ NLP techniques to build semantic networks out of requirement statements.

3. TEXT

The taxonomy extraction approach in this paper tries to harness the special nature of natural language requirements by clustering features of a semantic vector space model. Supplementary to word-context features, we introduce a new group of features to capture semantics of compounds. A taxonomy enables so-called concept traceability between concepts (domain-specific terms) in high-level requirements and the corresponding low-level requirement statements, in which they appear. The approach is considered semi-automatically, because a user should be able to correct and manage the resulting taxonomy, since it could still contain errors due to the usage of natural language.

3.1. NLP-Pipeline

A first necessary step in any Natural Language Processing task is to preprocess text documents before further analysis can be carried out. Data cleansing and transformation are classic subtasks of preprocessing. NLP tools have difficulties handling noise, i.e., misspellings, unknown symbols and malformed sentence structures. Data cleansing can help to remove such noise and therefore to obtain more correct parsing results. The following items were removed in the preprocessing stage of this work:

- Mathematical symbols
- Domain-specific identifiers, e.g., variable names
- Tokes between brackets

The resulting text is handed to the German sentence detector and tokenizer of Apache OpenNLP (http://opennlp.apache.org/). Subsequently, every sentence is parsed by the Lexicalized Stanford Parser using the German probabilistic context-free grammar (PCFG) model. This model was trained on the Negra treebank corpus [13]. The parser returns the most probable parse tree for every requirement statement.

A first scan through these parse trees detects every noun phrase and declares their head word to be relevant concept. To make sure that every concept is only extracted once, it is necessary to match different inflections of nouns to a single lexeme. The lemmatizer of the TreeTagger (http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger) is used to lemmatize every noun. German compounds are often difficult to be reduced to a lemmatized form. Therefore, fuzzy

matching is applied using edit distance. For instance, the genitive of the compound *Zündschlüssel* ("ignition key") is *Zündschlüssels*. Their edit distance is one, because of the appended *s*. The distance of two nouns w_1 and w_2 with respective lengths $|w_1|$ and $|w_2|$ can be transformed into a matching percentage p by calculating:

$$p = 1 - \frac{dist(w_1, w_2)}{max(|w_1|, |w_2|)} \quad (1)$$

The threshold of p was set empirically to 85 percent. Whenever two nouns are matched above the threshold ($p \geq 0.85$), only the shorter form is kept. Note that fuzzy matching does not only handle inflections, it can also be used to correct misspellings.

Besides spelling mistakes and grammatical errors, nominalization poses a challenge to NLP in this case.

Nominalized verbs are tagged as nouns, but they represent no relevant concepts. This shortcoming is solved by considering the knowledge of a general thesaurus. If a head of a noun phrase has at least one entry as a verb in the thesaurus, it is not considered as concept.

In section 3.2, the thesaurus also functions as source for hypernyms. Both, GermaNet [14] and the German Wiktionary dump (http://dumps.wikimedia.org) can be used as general thesaurus in our approach. Their influence on performance is evaluated in section 5.

3.2. Taxonomy Clustering

3.2.1. Context Features

Context features capture semantics of nouns according to the distributional hypothesis [15]. In the previous stage, head words of the noun phrases have been identified as candidate concepts. In order to extract context features for every candidate concept, we define context of a concept as their verb and prepositional dependencies. These pseudo-syntactic dependencies are extracted using Tregex expressions of the Stanford NLP software (http://nlp.stanford.edu/software/tregex.shtml). Again, both, heads of prepositional phrases and verbs are lemmatized, because morphological variants still have the same meaning and should be reduced to the same lexeme. Additionally, nominalization of verbs has to be resolved, because a nominalized verb must be treated like a verb dependency. This also conforms to the SOPHIST REgelwerk, which state that the passive nature of nominalized process words in requirement statements should be resolved into an active sentence structure [16].

3.2.2. Compound Features

Nouns themselves hold implicit semantics. Especially compounds, which are used very frequently in German language, possess relational semantics within their individual components. Compound words merge the meaning of two or more nouns into a new noun and are therefore candidates for concepts. Mostly, the meaning of the whole compound can be assembled from the meaning of its parts. Thus, there exists a semantic relation between a compound and its parts. However, it is not clear what kind of relation is present at first. Both, part-of and subconcept relations are possible. A simplifying suggestions is made by Vela and Declerk [17]. Their intuitive idea states that the two aforementioned relations can be captured by the two syntactic rules presented below:

(i). compound[prefix + concept] → subconceptOf(compound, concept)

(ii). compound[concept + suffix] → meronymy(concept, suffix)

As these rules are inherently fuzzy, ontology extraction might not be very accurate, if it solely relies on them. However, they present hints about possible semantics of compounds and those hints can be combined with their contextual information to facilitate taxonomy clustering. We present a feature engineering approach, which takes advantage of this compound information.
If rule (i) applies to a compound a and concept b, then the context feature vector $\overrightarrow{v_a}$ of the compound inherits all context features $\overrightarrow{v_b}$ of the concept. Consequently, $\overrightarrow{v_a}'$ becomes the new feature vector of the compound. Formally, this yields:

$$\overrightarrow{v_a}' = \overrightarrow{v_a} + \overrightarrow{v_b} \tag{2}$$

For instance, let the context feature vectors $\overrightarrow{v_s}$, $\overrightarrow{v_z}$ of the noun *Schlüssel* ("key") and the compound *Zündschlüssel* hold the following values in a 3-dimensional feature space:

$$\overrightarrow{v_s} = \begin{pmatrix} f_1 & f_2 & f_3 \\ 1 & 0 & 0 \end{pmatrix}, \qquad \overrightarrow{v_z} = \begin{pmatrix} f_1 & f_2 & f_3 \\ 0 & 0 & 1 \end{pmatrix}$$

Rule (i) matches the compound *Zündschlüssel* with prefix *Zünd* and concept *Schlüssel*. Applying equation (2), gives the new feature vector $\overrightarrow{v_z}'$:

$$\overrightarrow{v_z}' = \begin{pmatrix} f_1 & f_2 & f_3 \\ 1 & 0 & 1 \end{pmatrix}$$

Additionally, rule (ii) can also be used to augment context feature vectors. If compound a has concept b as suffix, then the context feature vector $\overrightarrow{v_a}$ of the compound is enhanced with a new dimension d_b. This dimension models the meronymy relationship. It is equal to 1 for every $\overrightarrow{v_a}$ matching rule (ii) with b, 0 otherwise.

$$d_b = \begin{cases} 1, & \forall a : meronymy(a, b) \\ 0, & otherwise \end{cases} \tag{3}$$

$$\overrightarrow{v_a}' = \overrightarrow{v_a} \parallel d_b \tag{4}$$

Equation (4) formulates the concatenation of vector $\overrightarrow{v_a}$ and scalar d_b. As a result, the distance between two feature vectors having the same meronymy with a certain concept will be closer with respect to some distance measure in feature vector space. For example, consider the compound *Blinkerhebel* ("indicator switch"), which matches rule (ii). It appears with frequency x_i in context f_i. Here, *Blinker* and *Hebel* are both concepts with corresponding context feature vectors. Using equation (3) and (4), the feature space is enhanced with a new dimension, namely *hasHebel*. Consequently, the new feature vector $\overrightarrow{v_{bh}}'$ of *Blinkerhebel* is given as follows:

$$\overrightarrow{v_{bh}}' = \begin{pmatrix} f_1 & f_2 & f_3 & hasHebel \\ x_1 & x_2 & x_3 & 1 \end{pmatrix}$$

Every noun not being a holonym of *Hebel* holds the value zero for the dimension *hasHebel*.

3.2.3. Feature Transformation

Having word-context counts as feature representation can lead to a noisy feature space, because some words might appear very frequently, while others might appear only once. Since *term frequency-inverse document frequency* (tf-idf) weighting does not apply when dealing with word-context features, *Positive Pointwise Mutual Information* (PPMI) weighting is employed to the

data set [18]. Another issue is the sensitivity of clustering to the curse of dimensionality. In order to overcome this problem, the dimensionality reduction technique *Principal Component Analysis* (PCA) is applied to the data. PCA reduces dimensions of a dataset by using the eigenvectors of the covariance matrix so that a specified variance threshold is retained. The percentage of retained variance was set to 90 percent for case study evaluation, see section 5.

3.2.4. Clustering Algorithms

Common taxonomy clustering approaches make use of hierarchical clustering algorithms. These algorithms tend to produce deep concept hierarchies, i.e., many levels of concept abstractions. In this work, we introduce a method to create shallow concept hierarchies, which are assumed to give more concise concept descriptions in the domain of automotive RE.

As a first step, both, context and compound features are fed into a partitioning clustering algorithm. Since the number of clusters of semantically related concepts is unknown, the silhouette coefficient serves as evaluation measure [19]. Consequently, the set with the best performing number of clusters is taken. The next step is to determine a single superconcept for every cluster. This is done by querying a general thesaurus for every concept in a particular cluster.

The general thesaurus is used to derive concept hierarchies, because natural language requirements contain domain-specific nouns, e.g., *Türsteuergerät* (“door control unit”). More general nouns usually suffer from the lack of a referential index and should therefore not be used in requirements statements, cf. “generalization” in [16]. Hence, a general thesaurus delivers hypernyms that can summarize domain-specific nouns.

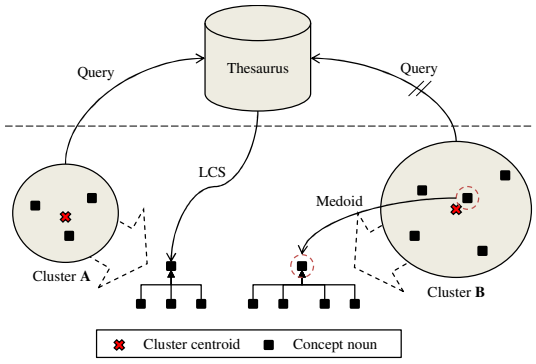

Figure 1. Derivation of concept hierarchies

Figure 1 illustrates two possible mechanisms to find the superconcept of a cluster. If more than one concept of a cluster is found by querying the thesaurus, their least common subsumer (LCS) from the thesaurus’ database is set as superconcept of the other cluster members (shown in fig. 1 on the left hand side). Note that the LCS of a single noun is the noun itself, hence when only one cluster member is found in the thesaurus, it will be chosen as superconcept. On the other hand, if the query responds no results, the cluster’s medoid (shortest distance to centroid) is set as the superconcept of the other cluster members, since it captures the “mean” context of every cluster member. It can be seen that the introduction of new hypernyms strongly depends on the used thesaurus.

4. REQUIREMENTS ONTOLOGY

In context of requirements engineering it is necessary to use well defined domain-specific terms consequently through the development process to avoid misunderstandings.

Textual requirement statements from a high-level specification can be refined into several more detailed low-level requirements which mostly describe the systems behavior and execution in detail. The taxonomy of high-level concepts supports elicitation of low-level requirements by providing a consistent terminology. However, the taxonomy also functions as common system understanding and therefore there should be a traceable relationship between high-level concepts and their corresponding low-level requirements statements. We refer to this relationship by the term *concept traceability*.

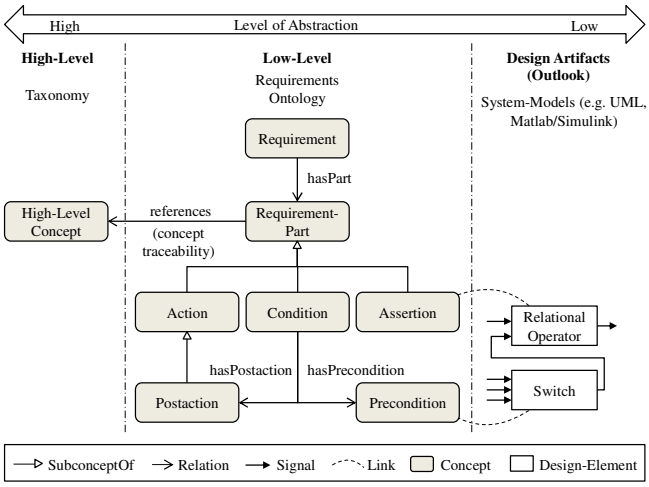

Figure 2. Structure of the requirements ontology

Figure 2 shows the structure of the concepts in the requirements ontology alongside with its dependencies to high-level concepts and design artifacts. Every requirement statement is represented as individual of the requirement concept, which has a relation to the requirement-part concept. As illustrated in figure 2, each individual of the concept requirement-part holds a reference to the high-level concepts that appear in their respective statement. These parts are either actions, conditions or assertions, which reference a high-level concept in the taxonomy. A condition represents a conditional requirement statement, which consists of one or more preconditions and a postaction.

The population of the requirements ontology is done by a simple procedure. Figure 3 shows this procedure as state diagram.

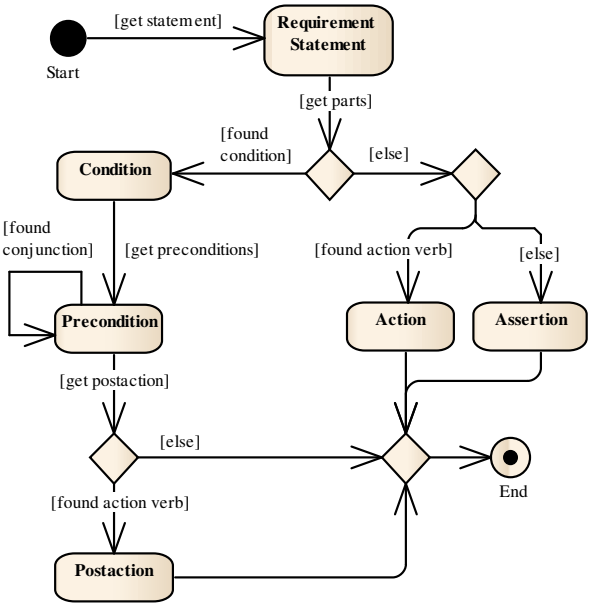

Figure 3. Populating the requirements ontology

If a requirement statement holds a conditional phrase, the protasis represents a precondition individual. Also, if there exists an action verb, i.e., a verb that is not modal or auxiliary, in the apodosis, then an individual of the concept postaction is created.

If no conditional phrase is present, but an action verb can be found, a single action individual is created. Otherwise, the statement is considered to be an assertion.

5. EVALUATION

Automated taxonomy extraction methods can only be evaluated empirically. For this reason, we compare a manually created taxonomy to the extracted one. In order to measure their degree of similarity, we used the evaluation metrics of Maedche and Staab with the adjustments of Cimiano et al. [3, 20]. More precisely, the F_1-measure, calculated as the harmonic mean of *lexical recall* and *taxonomic overlap*, serves as evaluation criterion.

Due to performance reasons, the K-Means algorithm with Euclidian distance measure was employed on the use case data set. Table 1 shows the results of taxonomy clustering carried out on 300 functional requirement statements of the case study specification [21]. Overall, our system detected 157 distinct nouns, however, the manually created taxonomy comprises 189 distinct classes. Performance results were averaged over ten runs of K-Means with random initial centroids for every clustering setting.

Table 1. F_1-measures of taxonomy clustering

Feature / Thesauri	c	c + i	c + ii	c + i + ii
GermaNet	0.557	**0.565**	0.52	0.544
Wiktionary	0.528	0.539	0.521	0.545
None	0.529	0.535	0.506	0.524

The columns in table 1 present the different features that were used to cluster the data. Context features are denoted by *c*, the roman numbers *i* and *ii* refer to the compound feature rules (i) and (ii). The rows depict a comparison between the two different thesauri and using no thesaurus at all (None), i.e., only medoids serve as superconcepts.

It can be seen that the addition of compound feature *i* increases performance compared to only context features. Also, compound feature *ii* decreases performance in all settings. The combination of both compound features shows best results in the Wiktionary setting, but for the other thesauri settings it performs worse than context features only. On average, GermaNet outperforms both other thesaurus settings. The best result (highlighted in table 1) is obtained by the setting of context features plus compound feature *i* accompanied by GermaNet having lexical recall of 0.891 and taxonomic overlap of 0.414.

In order to validate the assumption that shallow concept hierarchies are more concise in the domain of requirement statements, we compared the presented approach to common hierarchical agglomerative clustering applied to context features only. The advantage of hierarchical clustering algorithms is that there is no need to provide the number of clusters in advance. Their resulting hierarchical order of concepts can directly be transformed into a taxonomy. Both, the same distance measure and the same strategy for superconcept search were used, i.e., whenever two clusters are merged, their superconcept is either the LCS of its members or the medoid.

Corresponding results are shown in table 2.

Table 2. F_1-measures of hierarchical clustering

Merging / **Thesauri**	**single**	**avg**	**complete**
GermaNet	0.037	0.16	0.227
Wiktionary	0.038	0.168	**0.236**
None	0.038	0.167	**0.236**

We compared three different merging methods: single-linkage (single), average-linkage (avg), and complete-linkage (complete). Single-linkage and complete-linkage are both sensitive to outliers. However, single-linkage also suffers from the *chaining effect*, where multiple successive merges add one element to a cluster. This leads to heterogeneous clustering results, which explains its poor performance in this case study. Complete-linkage merging assumes that all members of a cluster are similar to each other. It tends to produce more compact clusters with equal diameters [22]. Average-linkage is more robust to outliers and can be seen as the intermediate between single- and complete-linkage.

In our case, complete-linkage clearly outperforms the other methods, similar results were reported by Cimiano et al. [3], but because of its high-order hierarchies, it can not match up to the performance of our shallow taxonomy clustering algorithm. This big difference in the F_1-measure is mostly due to taxonomic overlap, as lexical recall was comparatively similar for hierarchical clustering and shallow taxonomy clustering. Another apparent insight is that thesauri settings have no influence on hierarchical clustering, since clusters are generally too heterogeneous to make use of the LCS superconcept method.

For the assessment of taxonomy extraction, a comparison to the approach of the OntoGain system is made. The OntoGain system has previously been shown to outperform Text2Onto in terms of taxonomy extraction [5]. Using agglomerative, OntoGain achieves precision of 0.713 and recall of 0.627 on a comparable computer science corpus. Therefore, by interpreting taxonomic overlap

as precision and lexical recall as recall, it can be seen that the OntoGain system showed better F_1-measure performance (0.667 compared to 0.565), however, our approach achieves higher recall with the trade-off of lower precision. When documenting RE process knowledge, a high recall is more desirable, because it ensures that no relevant concepts are missed.

6. CONCLUSION AND OUTLOOK

In this paper, we presented a new approach to extract taxonomies from automotive requirements statements. These taxonomies will be the basis for the proposed ontology-driven RE process.
Results of taxonomy clustering showed that, in the domain of automotive requirements, there are domain specific terms, which can be organized as shallow concept hierarchy according to their context usage. The addition of a compound feature that captures subconcept hints, can help to obtain more precise results. When considering hypernyms of general thesauri, GermaNet was shown to be slightly more reliable compared to Wiktionary. Generally, the approach presented in this work outperforms common hierarchical clustering techniques.

Our future work includes the build-up of a requirements ontology based on low-level textual requirement statements on the proposed structure as well as the derivation of design-models out of this requirements ontology in order to establish concept traceability throughout the entire RE process.

REFERENCES

[1] M. Schraps and C. Allmann, "Ontologiebasierte Entwicklung von Anforderungsspezifikationen im Automotive-Umfeld," *Softwaretechnik-Trends*, vol. 32, no. 4, 2012.

[2] P. Cimiano and J. Völker, "Text2Onto - A Framework for Ontology Learning and data-driven Change Discovery," in *Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems*, 2005.

[3] P. Cimiano, A. Hotho, and S. Staab, "Clustering Concept Hierarchies from Text," in *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC 2004)*: European Language Resources Association, 2004.

[4] H. Yang and J. Callan, "Feature Selection for Automatic Taxonomy Induction," in *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York: ACM, 2009, pp. 684–685.

[5] E. Drymonas, K. Zervanou, and E. G. M. Petrakis, "Unsupervised Ontology Acquisition from Plain Texts: The OntoGain System," in *Proceedings of the Natural Language Processing and Information Systems, and 15th International Conference on Applications of Natural Language to Information Systems*, Berlin, Heidelberg: Springer-Verlag, 2010, pp. 277–287.

[6] M. Kitamura, R. Hasegawa, H. Kaiya, and M. Saeki, "A Supporting Tool for Requirements Elicitation Using a Domain Ontology," in *Software and Data Technologies: Second International Conference, ICSOFT/ENASE 2007*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 128–140.

[7] K. Siegemund, E. Thomas, Y. Zhao, J. Pan, and U. Assmann, "Towards Ontology-driven Requirements Engineering," in *The 7th International Workshop on Semantic Web Enabled Software Engineering Co-located with ISWC2011*, Bonn, 2011.

[8] S. Farfeleder, T. Moser, A. Krall, T. Stålhane, I. Omoronyia, and H. Zojer, "Ontology-Driven Guidance for Requirements Elicitation," in *Lecture Notes in Computer Science, The Semanic Web: Research and Applications*: Springer Berlin / Heidelberg, 2011, pp. 212–226.

[9] M. Kossmann, R. Wong, M. Odeh, and A. Gillies, "Ontology-driven Requirements Engineering: Building the OntoREM Meta Model," in *3rd International Conference on Information and Communication Technologies: From Theory to Applications (ICTTA)*, 2008, pp. 1–6.

[10] D. Faure and C. Nédellec, "ASIUM Learning subcategorization frames and restrictions of selection," in *10th European Conference on Machine Learning (ECML 98) - Workshop on Text Mining*, Chemnitz, 1998.

[11] L. Kof, "Text analysis for requirements engineering," Technische Universität, München, 2007.
[12] M. G. Ilieva and O. Ormandjieva, "Automatic Transition of Natural Language Software Requirements Specification into Formal Presentation," in *Lecture Notes in Computer Science, Natural Language Processing and Information Systems*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 392–397.
[13] A. Rafferty and C. D. Manning, "Parsing Three German Treebanks: Lexicalized and Unlexicalized Baselines," in *ACL WorkShop on Parsing German*, 2008.
[14] B. Hamp and H. Feldweg, "GermaNet - a Lexical-Semantic Net for German," in *Proceedings of ACL workshop 1997*, 1997, pp. 9–15.
[15] Z. Harris, *Mathematical structures of language*. New York: Interscience Publication, 1968.
[16] C. Rupp and SOPHIST GROUP, *Requirements-Engineering und -Management: Professionelle, iterative Anforderungsanalyse für die Praxis,* 5th ed. München, Wien: Hanser, 2009.
[17] M. Vela and T. Declerck, "A Methodology for Ontology Learning: Deriving Ontology Schema Components from Unstructured Text," in *Proceedings of the Workshop on Semantic Authoring, Annotation and Knowledge Markup*, 2009.
[18] P. D. Turney and P. Pantel, "From Frequency to Meaning: Vector Space Models of Semantics," *Journal of Artificial Intelligence research*, vol. 37, no. 1, pp. 141–188, 2010.
[19] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.
[20] A. Maedche and S. Staab, "Measuring Similarity between Ontologies," in *Proceedings of the European Conference on Knowledge Acquisition and Management (EKAW)*: Springer, 2002, pp. 251–263.
[21] F. Houdek and B. Paech, *Das Türsteuergerät - eine Beispielspezifikation*. IESE Report Nr. 002.02/D. Available: http://publica.fraunhofer.de/documents/N-10473.html.
[22] B. S. Everitt, *Cluster analysis,* 5th ed. Chichester: Wiley, 2011.

Authors

Martin Ringsquand is a computer science graduate student at University of Augsburg, focusing on knowledge representation and machine learning.

Mathias Schraps is a PhD student at Audi Electronics Venture GmbH and Leibniz University of Hanover. His research area is ontology-based Requirements Engineering in software development of vehicle functions.