# Practical Aspects of Ontology-Based Analysis and Reasoning for Law Information Represented in Textual Form

Raoul Schönhof, Axel Tenschert and Alexey Cheptsov
High Performance Computing Center Stuttgart,
University of Stuttgart
Stuttgart, Germany
e-mail: raoul.schoenhof@b-f-u.de, tenschert@hlrs.de, cheptsov@hlrs.de

*Abstract*—**Legal domain is an important source of information and includes diverse law texts, court decisions, etc., which is dominated by documents collected in natural language. There is a great demand for the automatic analysis of the legal information in everyday work of lawyers and other people dealing with laws, for which the ontology-based knowledge representation would be of a great advantage. Unfortunately, the current semantic and ontology based technologies cannot be easily applied for the analysis of legal texts due to a certain complexity of those. We present a strategy that allows different categories of tools, such as those for ontology creation, syntactical analysis of texts collected in natural language, and others to interoperate in order to achieve a common goal – creation of domain-specific ontologies and performing complex reasoning over them. A model that can be applied for the legal system knowledge representation is proposed and its implementation for the German Civil Law System in form of an ontology is discussed. The model allows the creation of a common ontology spanning over the knowledge contained in diverse laws, thus paving the way towards a wide adoption of semantic technologies by the experts in the law domain.**

***Keywords-Knowledge Representation; Law Texts; Ontology; RDF; Big Data; Rule-Set.***

## I. INTRODUCTION

This paper deals with the technologies presented in our previous work [1] for legal knowledge representation as Resource Description Framework (RDF) ontologies and discusses practical aspects of their application by means of the existing analysis tools. Working with highly unstructured and ambiguous data is a major challenge when solving many research, industrial, and societal issues. The knowledge required for tackling those challenges is contained in most cases in the documents that are represented as natural texts, e.g., in books, journal articles, websites, files on the disc, etc. The information extraction from those sources requires a deep knowledge of the content as well as the understanding of the domain-specific terminology. However, the major challenge is the automation of the knowledge discovery process.

A law system can be considered as an important source of information that has to be retrieved in form of a digital knowledge base with the goal to perform reasoning over a number of interrelated information sources of the distributed nature. The knowledge base has to be built and treated dynamically from the changing data sources. For example, in Germany, 553 federal laws and many more federal state ones have appeared in the time between 2009 and 2013 [2]. Everyday, hundreds of new court decisions are made, which have a potential to influence the interpretation of the statements contained in laws.

Semantic technologies offer well-established tools for treating the legal knowledge in unstructured and dynamic data sets by retrieving the content as a structured (and thus analyzable) ontological representation. As compared with the other domains that rely on plain texts as the major source of information, the law data offer a well-defined terminology and a clear, systematic structure. Center of the law system is the German Civil Code (in German terminology "Bürgerliches Gesetzbuch" or shortly BGB). It manages and defines fundamental and general issues. The paragraphs are numbered ongoing through the entire BGB. Most of the single paragraphs are broken down into articles, sub articles and half sentences or numbers (see Figure 1). Despite the semi-structured nature of the law texts, several previous attempts to treat law information (cf. [3], [4]) by means of ontologies were not very successful as they were based on abstract model with a static structure, which made their use in practice very limited.

Our approach is different – we aim to develop a system that is able to automatically and on-the-fly build the ontological representation of the information contained in dynamic data sources and enabling reasoning over it. The goal of reasoning is to understand and analyse the information based on multiple sources, such as laws and previous court verdicts. As a use case, the German law system is considered. The law texts are explored and structured using the techniques such as Resource Description Framework Syntax (RDFS) [5] and Web Ontology Language (OWL) [6] for the ontology extraction and further use in an automated reasoning process. Using the proposed tool set, the experts of the problem domain will be able to consider a much wider spectrum of documents than available to them currently.

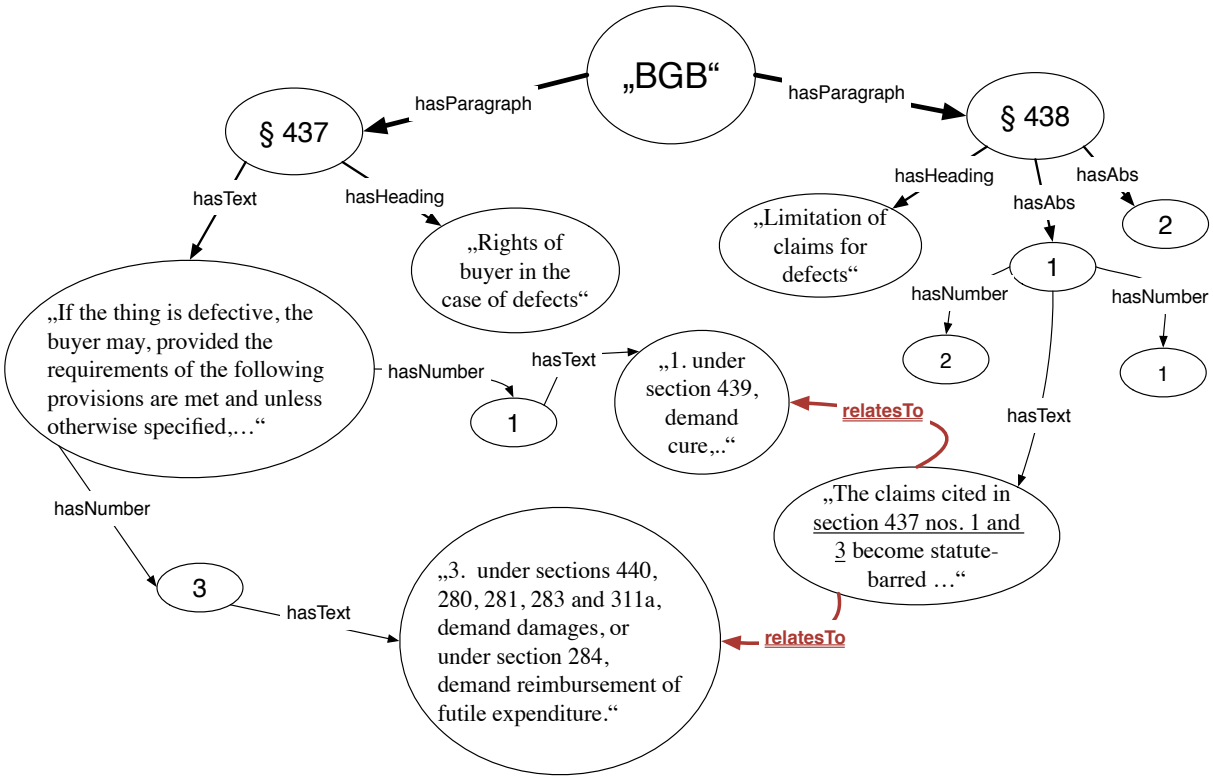


Figure 1. Example of connections in legal text

The remainder of the paper is organized as follows: Section II gives an overview of the German legal system and explains briefly, supplied by an example, how different laws can interact with one another. Section III demonstrates the capability of semantic web technologies in the field of law by means of a little use case. Section IV introduces the system design and shows how legal knowledge ontologies could be generated from natural texts found in law collections by using computer linguistic tools. Finally, Section V deals with the future tasks as well as the assets and drawbacks.

## II. EXEMPLARY SCENARIO

The information contained in individual laws only make value when considered across all other regulations of a law system related to the given case. However, the analysis of cross-references between multiple laws is often complicated by several factors, such as complex structure, use of abstract wording and specialized terminology, number of related laws, etc. For example, the BGB is divided into five chapters; each chapter manages a special legal domain. The first chapter is called General Part, which is the result of the repetition reduction. It contains mostly definitions and general rules of law; these are used in the chapters two to five. The second chapter is called Law of Obligations. It contains rules of law to any kind of contract and defines the most common contracts, for example the purchase agreement. This chapter is followed by the Law of Property, the Family Law and the Law of Succession. Especially the separation between more general rules of law and specialized rules of law makes it possible that two rules of law regulate one situation in different ways. In such cases, the more general rule of law is displaced by a more specialized one or a younger rule of law displaces the older rule of law. Therefore, rules of law interact constantly with each other.

Let us illustrate the decencies between laws based on the following example (*§ 437 BGB* and *§ 438 BGB* of the Sales Convention [7]):

*§ 437 BGB* : “If the thing is defective, the buyer may, provided the requirements of the following provisions are met and unless otherwise specified, 1. under section 439, demand cure, 2. revoke the agreement under sections 440, 323 and 326 (5) or reduce the purchase price under section 441, and 3. under sections 440, 280, 281, 283 and 311a, demand damages, or under section 284, demand reimbursement of futile expenditure. ” [7].

*§ 438 I BGB*: “The claims cited in section 437 nos. 1 an 3 become statute-barred 1. in thirty years, if the defect consists a) a real right of a third party on the basis of which return of the purchased thing may be demanded, or

b) some other right registered in the Land Register, 2. in five years a) in relation to a building, and b) in relation to a thing that has been used for a building in accordance with the normal way it is used and has resulted in the defectiveness of the building, and 3. otherwise in two years." [7].

While on the one side, *§ 437 BGB* defines the rights of a buyer in case the purchased object is faulty, *§ 438 BGB* on the other side declares that some of these rights (*§ 437 no. 1* and *3*) become statue-barred after a certain time [7]. In this example, the rules of law are connected through named references (see also Figure 1), but it is also common to connect rules of law through abstract concepts, here for example the word statute-barred, which is again defined in *§ 194 BGB*.

The total amount of relations in a legal system is usually very large. Therefore, the main idea we are proposing is to develop a tool set that supports users, e.g., jurists but also persons without any deep legal knowledge, during dealing with legal issues.

## III. The Use Case

Semantic web technology have proven efficient when dealing with large interlinked relational data sources such as can be found in the targeted law domain. Let us consider a very simple legal issue represented with the sentence, as an example: "Henry buys a bicycle from Peter". The user might be interested to find regulations, which are applicable to this issue. Depending on the law system, our system has to draw attention to the purchase agreement, e.g., § 433 I BGB [7] or to the Article 2 of the Uniform Commercial Code (UCC) [8].

Solving this task requires additional information given from three different resources; law books, dictionaries or encyclopedias like Wikipedia [9] and the legal issue from the user itself. Therefore, the particular legal issue of the user is abstracted by information from dictionaries or encyclopedias and matched to the abstract act given from law books. The system retrieves the information from natural language text. Figure 2 depicts this information within the blue, yellow and green ovals. Hereby, the respective entities are connected by black arrows representing the retrieved information from texts. The

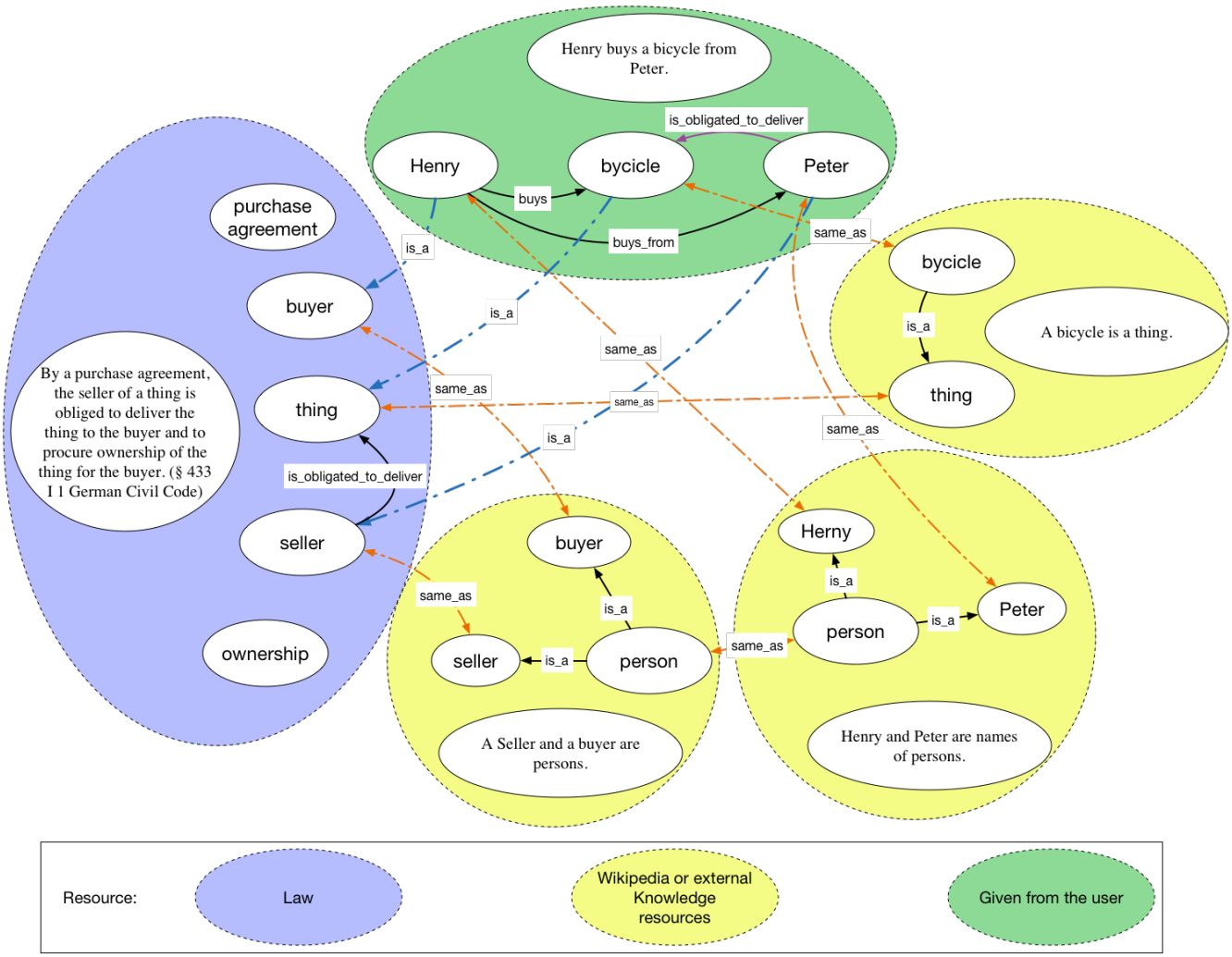


Figure 2. Use Case

purchase agreement states, among others, the seller is obligated to deliver the *thing*. Obviously, a reasoning algorithm can not match the entity *bicycle* to the entity *thing,* required for § 433 I [7]. By adding additional information from dictionaries or encyclopedias, it is possible to bridge this gap between *bicycle* and *thing*. These connections are represented through orange arrows. In a similar way, the entities *Henry* and *Peter* can be abstracted to the entities *buyer* and *seller* via the entity *person* and their names. As result (blue arrows), the entities *Henry*, *bicycle* and *Peter* can now be matched to the entities *buyer*, *thing* and *seller*. Furthermore, the purple relation *is_obligted_to_deliver* is applicable between the entity *Peter* and *bicycle*.

In this simple use case, the system reports back, that Peter is obligated to deliver the bicycle. The user benefits from a fast and simple way to get informative hints, in the field of law.

## IV. State of the Art

The information extraction from natural texts, in the context of our research goals, is based on three pillars: Linguistic, Computer Science, and Law. The first two pillars offer technologies while the third one represents an use case. In the following subsections, we concentrate on technological challenges of the analysis technologies.

### A. Information Retrieval Systems for Ontology Generation

The amount of information is constantly increasing but only available in an unstructured format. Mostly, the information is hiding in natural language texts. There have been numerous approaches to retrieve information from documents and texts, deriving an RDFS/OWL graph. To the most popular approaches belong Text2Onto [10], OntoLearn/OntoLearn Reloaded [11], OntoMiner [12] and OntoLT [13][14]. The OntoMiner approach analyzes regularities from HTML Web documents. A substantial disadvantage is the requirement of a handpicked set of web sites within the admired field of interest. The output taxonomy is strictly hierarchical, which is appropriate to classify entities, but it cannot find a considerable amount of relations between entities inside a level in the hierarchy. Interconnections are necessary performing complex reasoning tasks. The same situation looms with regards to the OntoLearn Reloaded approach. Much more promising is the approach of Text2Onto and OntoLT. Text2Onto combines machine learning strategies with basic NLP methods [10], particularly tokenization, lemmatizing and shallow parsing, allowing the application to analyze a natural language text more detailed. Testing Text2Onto has demonstrated, that the retrieved amount of information was not enough, with regards to the field of interest. Beside Text2Onto, even OntoLT was using NLP technology, above the task of named-entity-recognition, to generate semantic networks. Hereby, OntoLT was using predefined mapping rules for every desired annotation tag. OntoLT then constructs an OWL ontology according to the given mapping rules. According to our knowledge, OntoLT has not been extended since 2007.

The presented approach affiliates this concept of grammatical-driven information retrieval, implements state of the art NLP tools and expands it by considering grammatical dependencies for information retrieval to achieve a higher precision and applying it to the field of law. Using dependencies for information retrieval, the approach benefits by additional information about the semantic content of text [15]. Our approach is based on three pillars: Linguistic, Computer Science, and Law. The first two pillars offer technologies while the third one a use case. In the following subsections, we concentrate on technological challenges of the analysis technologies.

### B. Linguistic Tools and Syntax Theories

P. G. Otero [15] presents an approach for exploiting human-written text by computers, according to which it is necessary to examine the structure of each sentence considering the dependency syntax. In the last decade, the linguistic tool development has been established very well, especially with regard to grammatical parsers. For example, the Stanford NLP Group offers a comprehensive toolset for various aspects of grammatical sentence parsing [16]. For our goals, we took a closer look at four types of computer linguistic tools: i) constituency parsers, based on the generative grammar, ii) dependency parsers, based on the dependency grammar, iii) named entity recognizers, used for locating and classifying entities in text, and iv) sentence splitters.

**Constituency Parser.** Constituency parsers are based on the idea of splitting a sentence in functional units called constituents [17]. The resulting tree of superior and subordinated constituents generates a tree-like structure, which is mapped as an Augmented Transition Network (ATN) [18]. ATN offers a flexible and scalable technology to represent the grammatical structure of sentences. It disassembles a sentence into constituents (see Figure 3) and tags them. A very common constituency parser is the Stanford Parser [19]. It supports various languages, including English, German, Chinese, and Arabic. An example of ATN for the sentence "*A computer is a machine.*" is shown in Figure 3, by using the constituent tags from the Penn Treebank Project [20][21]. The sentence (S) is divided in two "sub-constituents", here the noun phrase (NP) and the verbal phrase (VP). These contain either atomic words or other constituents. Here, the determiner (DT) "*A"*, the noun (NN) "*computer"*, the verb (VBZ) "*is",* the noun "*machine*" represent atomic words, whereas "*A computer*" or "*a machine*" form a noun phrase. In combination with a verb, the constituent VBZ and NP, here "*is a machine*", form a verbal phrase.

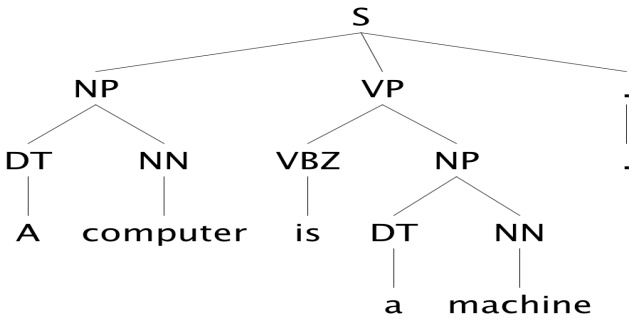


Figure 3. ATN Example based on Stanford Parser GUI

**Dependency Parser.** Dependency parsers [22] are based on the dependency grammar [23], which focuses on relationships between words and their functional role within a sentence [23]. Relations can be represented in a form of a directed graph, which makes it possible to derivate a hierarchy [23]. Because the structure of the hierarchy is only depending on the grammatical syntax, it is also possible to conclude to the semantic [15], e.g., Figure 3 shows an example sentence with its dependencies and constituents.

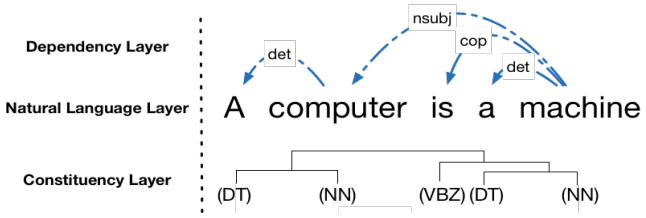


Figure 4. Grammatical structure of a sentence

The dependencies in a sentence are presented as a tree of connected word tags being knots. Hereby, the dependency tags *det*, *nsubj* and *cop* stand for *determiner*, *nominal subject* and *copula* [24] and provide additional information about the type of grammatical relations. Basically, this pattern is representative for a sentence of the type "*Object → Subject*". Figure 4 shows the resulting dependency pattern. The abstract pattern helps finding sentences with the known information structure. The identified words then need to be transferred into a more machine-recognizable format. This is not only useful for identification of classes and their subclasses but also with regard to the "valence theory" [22]. Origin of this theory is the empirical knowledge of the structure-determining characteristic of verbs as presented by L. Tesnière [25]. According to this and exposed by H. M. Mueller et al. [23] and V. Ágel [26][25], each word or word group is typically associated with a verb in the sentence. Therefore, dependencies could also help identifying the actions (= verbs) of individuals in the sentence.

**Named Entity Recognition.** Named Entity Recognizers (NER) are tools to identify typical non-context related individuals, e.g., locations ("Berlin", "Hong Kong"), organizations ("UNICEF", "NASA") or person names ("Lisa", "Rouven"). Therefore, NERs, like the Stanford NER, are using Conditional Random Fields (CRFs) to identify entities [27]. With regards to our approach, NERs are not essential but an improvement area to gather additional information helping to find some individuals, which could not be found by only focusing on ATN-trees or dependency structures.

**Sentence Splitting.** Typically, a text contains many sentences; in order to analyze them, they have to be separated. This task is performed by sentence splitters. One of the most popular sentence splitters is provided by A Nearly-New Information Extraction System (ANNIE) [28] - a software package of the GATE project. This splitter can distinguish between a full stop and any other point.

### *C. Working with Information*

While ATN, NER, and other dependency parsers can derive some useful information about the texts' structure, the ontology languages facilitate information representation. Ontologies can be leveraged to text to identify classes, individuals, or even properties in them. Alongside with that, ontology-based analysis frameworks provide tools that allow for querying the retrieved information.

#### *1) Web Ontology Language*

OWL provides a framework to store and handle information by ontologies [6]. OWL is based on the RDF [29] and equipped with an additional vocabulary [30]. Each OWL ontology can represent different kinds of information, e.g., classes, individuals or properties. While classes express abstract concepts, individuals are existing members of one or more classes. The relations between other individuals are defined by their properties. Therefore, OWL is predestinated to use ontologies with reasoning algorithms. [6]

#### *2) Semantic Web Rule Language*

As a special sub language of OWL, the SWRL represents abstract rules associating OWL individuals to any desired OWL class. Special forms of these rules are built-in relations. These rules consist of an antecedent, called "*body*", and a consequence, called "*head*". Several OWL individuals of an ontology can hereby be associated with another class [31]. This enables the use of very complex rules. A little example to illustrate: "*If a device contains a CPU, then it is a computer*". Therefore, an individual of the class "*device*" is defined as "*computer*" if this individual is connected to another individual of the class "*CPU*" by the object property "has*Contain*". The resulting SWRL Rule would be (1).

$$\text{device}(?x) \wedge \text{hasContain}(?x, ?y) \wedge \text{CPU}(?y) \Rightarrow \text{computer}(?y) \quad (1)$$

## V. System Architecture

As we proposed in [1], the current framework is defined by three components, the Sentence Processing Unit, the Ontology Generator and the Reasoner, shown in Figure 5. In this respect, we have improved our primary system architecture mentioned in 2014 [1][32]. In the first step, natural language texts form different sources like Wikipedia, law texts or from a given use case are loaded to the Sentence Processing Unit.

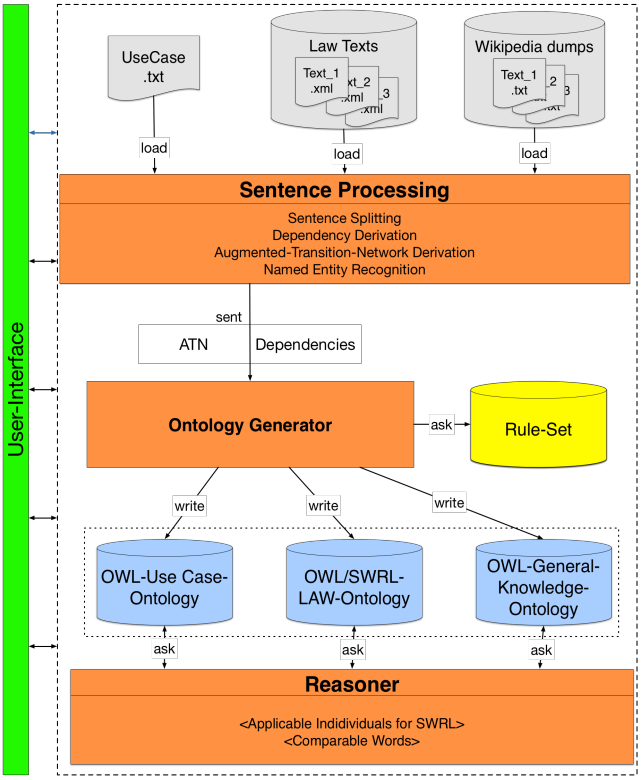


Figure 5. System Architecture

The first component represents the Sentence Processing Unit. The Sentence Processing Unit splits up the text to its sentences and divaricates the grammatical structure as an ATN and a dependency network. It is basically a conglomeration of different language processing tools containing the sentence splitter from ANNIE/GATE [28], the dependency and constituency parser from the Stanford NLP Group [19][22], as well as a Named Entity Recognizer [27]. The second component is the Ontology Generator with its Rule-Set (see Figure 5). It builds three OWL models out of natural texts. The first ontology contains the information about the questionable use case (OWL - Use Case Ontology). The second one contains the laws, respectively the legal prerequisites, represented as SWRL Rules (OWL/SWRL - LAW Ontology). The third ontology contains general knowledge, mainly about classes and subclasses (OWL - General - Knowledge). Finally, the third component is the reasoning process, respectively the reasoner. Hereby, the reasoner tries to match the given information based on the Use Case Ontology with the rules from the LAW Ontology. Because laws are written in a notional way, it is necessary to establish a connection between the individual of the use case and the SWRL rule. The General Knowledge Ontology provides this connection. The strict separation between the Use Case Ontology and the General Knowledge Ontology is necessary because the correctness of the given information in a random use case cannot be assumed.

### A. Sentence Processing

Starting point is the raw data, which contains texts with one or many sentences. The source of the texts might be Wikipedia [33], law texts [34], or any other texts related to the topic of our use case. These texts have to be processed, so the sentence structure, defined as pattern p, can be mapped. Each pattern $p \in P = \{d, A\}$ is represented by a subset of dependencies $d \in D = \{s, p, o\}$ and an ATN A. Hereby, *d* is described by triples, consisting of a subject *s*, a predicate *p* and an object *o*. While *s* and *o* are words, *p* belongs to a dependency tag, also shown in Figure 4. Therefore, each text passes through the ANNIE sentence splitter of the text-engineering tool GATE [28]. The constituent and dependency parsers then analyze the isolated sentences. Afterwards, the atomic words will be exchanged against their lemma projecting the numerous variants of a concept to a single lemma. Therefore, the complexity of the dictionary is reduced.

### B. Information Retriereval

Information about the content of the given law texts have to be extracted and added to the model, which is one of the most challenging tasks. Of an extraordinary interest is the identification of concepts in the encyclopedia text files, the law books or the particular rule as well as its heading. For instance, one of these concepts is ”statute-barred” in *§ 438 I BGB*, shown in Figure 1. The concept identification uses statistical extraction methods as well as pattern-based methods. Especially latter methods are predestinated to identify cross references, which are common in law texts. Because of the circumstance that some rules refer to another rule and some rules prohibit the applicability of another rule, the pattern based method has to distinguish between these two cases. Subsequent to the information extraction, the identified concepts are added as OWL triples to the initial model.

Naturally, these methods will just help to identify entities but they will not be able to extract a very large amount of information, e.g., the relation between a number of entities. Therefore, additional tools have to be used. Meanwhile, there are various text engineering tools, which are capable to extract information out of natural text; for instance, Text2Onto [10] and Gate [35] with the OWLExporter plug-in [36] as well as Protégé [37] with its plug-in OntoLT [38].

Beside these tools, the Stanford Natural Language Processing Group (SNLPG) at the University of Stanford developed a broad range of computer linguistic tools including a part-of-speech (POS) tagger to break sentences down into their lemma and mark them with their part of speech [39]. SNLPG also provides a special Named Entity Recognizer to find and classify salient nouns, e.g., the noun "London" as a location [27]. Furthermore, a sentence parser, e.g., Stanford Parser [19], is provided, which can be used to identify dependencies between words in a sentence.

The information extraction will be done as follows. Firstly, each sentence of the initial RDF ontology will be passed to the POS-tagger. It splits each sentence into single words and figures out, which part of speech may be present, e.g., whether it is a noun, a verb or an adjective. Also, the POS-tagger references from the words in a sentence to their lemmas. The lemma of nouns are added as isolated entities to the RDF model. After the sentence is tagged by the POS-tagger, the information about the part of speech is used by the Stanford Parser to generate a parsing tree. Dependency parsing is based on a parsing tree that represents a grammatical structure of a sentence, e.g., such as shown in Figure 4 for a part of *§ 437 BGB* as ATN including the dependencies [7].

This parser allows it to detect references between verb and noun phrases. These references will be used as properties in the RDF model. Unfortunately, there is no German language support for the Stanford Dependency Parser [22]. Thus, an alternative is necessary, which could be the Zurich Dependency Parser for the German language (ParZu) [40]. Output of the dependency parsing process may be considered as directed graph, consisting of the words as node and the dependencies as edges. These edges are tagged with the dependency type, e.g., nominal subject (nsubj). providing additional information about the specific type of relation.

### C. Ontology Generator & Rule-Set

The Ontology Generator translates a given sentence into a machine-recognizable OWL ontology, based on its pattern. The resulting OWL ontology is representing the base for any reasoning attempts. Hereby, the Ontology Generator compares the grammatical structures of a given sentence from a set of predefined grammatical patterns, called Rule-Set, to derive the OWL ontologies mentioned in Figure 5. If a pattern could be identified, the Ontology Generator converts the words as OWL Classes, OWL Individuals or SWRL Rules and interconnects them. The axioms are stored in different OWL ontologies. This is essential because the given information from a use case do not have to be true. One of the most difficult tasks is the development of the Rule-Set. This set contains patterns of typical sentence structures, as well as corresponding instructions. They can be described as follows:

Let $rs \in RS = \{p, i\}$ be the Rule-Set, which contains pattern and instruction i. The instruction describes the connection between the words as and OWL model, by generating OWL's Classes, Individuals, and Predicates. Because of the complexity of natural language, the patterns cannot exist statically (therefore, one pattern for each type of sentence) but must be composed from different rules. This process could be demonstrated at the following example.

Let us apply that Rule-Set to the text mentioned in Figure 3 ("a computer is a machine"). The first rule $rs_1(p_1,i_1)$ contains pattern $p_1$ (see Figure 6) that describes a noun (e.g., *computer*) referencing to another noun (e.g., *machine*) using the dependency *nominal subject*. In addition to the dependency structure, $p_1$ also contains the constituencies.

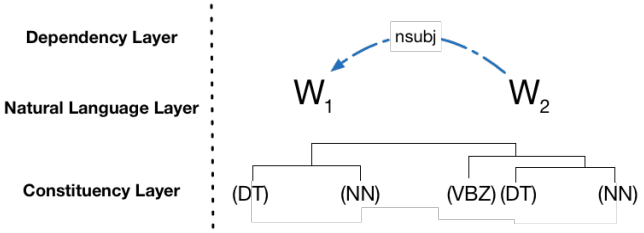


Figure 6. Pattern of $rs_1$

The corresponding instruction $i_1$ defines the first noun as a subclass of the second one:

$$i_1 := \text{SubClassOf(Computer, Machine)} \quad (2)$$

Now, let us add another rule $rs_2(p_2,i_2)$ to our Rule-Set specifying the connection between two nouns by means of the dependency "*compound*", like shown in Figure 4. The pattern is typical for compound nouns like "*computer system*" or "*street light*".

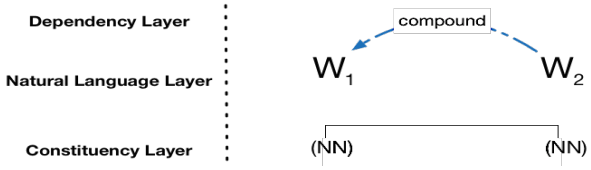


Figure 7. Pattern of $rs_2$

The instruction for this rule will be the following:

$$i_2 := \text{Class(Computer-System)} \quad (3)$$

Now, when trying to apply these both rules $rs_1$ and $rs_2$ to a more complex sentence like "A computer system is a machine", we will see that none of them alone is able to cope with the more complex grammatical structure of the new text. Therefore, the Ontology Generator must be able to combine existing rules to more complex rules, based on the simple patterns discussed above. To handle the mentioned sentence, it is necessary to combine $rs_1$ and $rs_2$, shown in Figure 8.

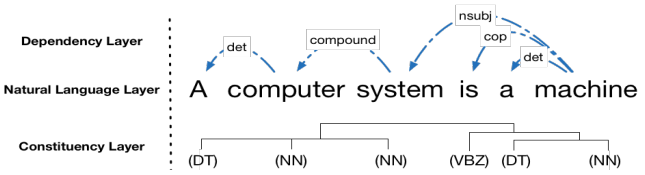


Figure 8. Joint pattern of $rs_1$ and $rs_2$

Now, the corresponding instruction $i_3$ defines the first two nouns as a subclass of the second one:

$$i_3 := \text{SubClassOf(Computer-System, Machine)} \quad (4)$$

Hereby, the selection of several applicable rules follows the principle of speciality, according to which a more complex rule takes precedence over a combination of more simple ones. The described patterns exist currently just in hard-coded form, containing a handful of rules, to proof the concept. Later, numerous rules will be derivated by automated or semi-automated machine learning algorithms.

Our approach was implemented in an early stage system to prove the proposed concept (see Figure 2). Hereby, the system contains the sentence processing unit as well as the Ontology Generator with a hardcoded Rule-Set. The given Rule-Set is capable to find and connect entities consisting of two nouns as well as simple sentences containing a simple subject-verb-object combination (see Figure 6). We defined a simple test set containing 41 sentences, as shown in Figure 9, and tested it for a small hardcoded Rule-Set (about five Rules). A more complex scenario, such as mentioned by Li et al. [41], will be a future work. Here, the given sentences are typically stating one entity as a subclass of another entity or connecting two entities by a verb. Beside grammatically correct sentences, the set contains also a grammatically malformed sentence (see Figure 9, no. 41). The resulting OWL ontology graph is shown in Figure 10. It contains gathered classes, individuals as well as object properties. So, for example, the individual "Henry" belongs

1. Ferraris are red cars.
2. A car is a vehicle.
3. A truck is a vehicle.
4. If a person buys a thing, than this person becomes the owner.
5. Karl sells his washing machine.
6. A machine is a device.
7. A device is a thing.
8. A washing machine is a machine.
9. Fritz buys a washing machine.
10. A purchase agreement is a contract.
11. A student is a person.
12. A computer is a device.
13. A coffee is a drink.
14. A thing is a thing.
15. A person is a human being.
16. A teacher is a person who teaches.
17. Peter Miller is a teacher.
18. A bicycle is a vehicle.
19. Henry is a professor.
20. A professor is a kind of teacher.
21. A vehicle is a thing.
22. A Ferrari is a car and a vehicle.
23. Stuttgart is located in Germany.
24. Stuttgart is a city.
25. A cabrio and a van are special types of cars.
26. Germany is a country in Europe.
27. Europe is a continent.
28. A person is a human.
29. Tool development is a challenge.
30. A cow is an animal.
31. A Ferrari is a fast and expensive sports car.
32. A coca-cola is a drink.
33. A vehicle is a transportation machine.
34. A Porsche is a sports car from Germany.
35. A document is a thing.
36. A contract is a document.
37. A dog is an animal.
38. Henry sells the bicycle.
39. Peter Miller buys a bicycle.
40. Peter Miller drives a truck.
41. A continent is a big. (malformed)

Figure 9. Example sentences

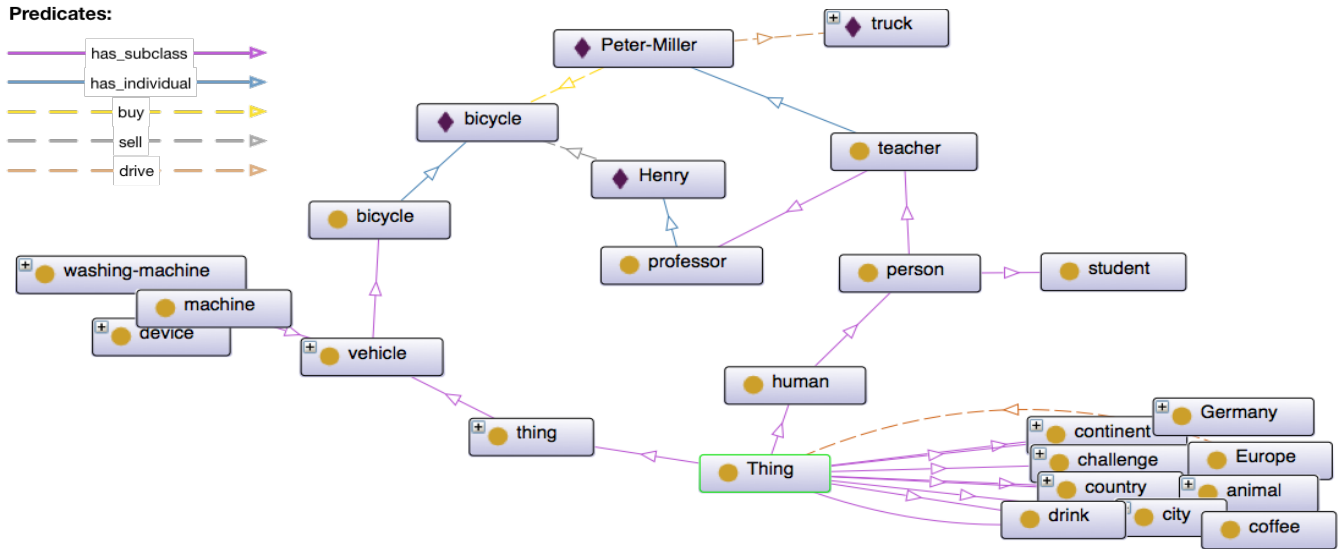


Figure 10. Resulting Ontology from the example sentences shown in Protégé [37]

to the Class "Professor". It has to be mentioned, that it is not mandatory to connect an individual just to one class. Basically an allocation to several other classes would be valid too, as long as there are no disjunctions between the chosen classes. The individual "Henry" could also be a "professor" and a "student" the same time. The given ontology shows every type of object predicate by the arrow's colour. Noticeable is the ability to distinguish between the ill-formed sentence no. 41, which is not represented in the ontology, and the well-formed ones, e.g., sentence no. 28 (see Figure 9).

### D. Reasoner

The main task of the reasoner is the identification of connections between the given case and the law ontology. Therefore, the reasoner has to find a conclusive path through the OWL networks. The results of the Ontology Generator are, depending of the input source, three OWL ontologies (see Figure 5). The record information, like individuals and their actions, is represented in the use case ontology. Information about the laws is given in the law ontology, mainly as classes and SWRL Built-in Rules. The last ontology is the General Knowledge Ontology, which is extracted from wikidump files [33]. It bridges the missing links between the particular use case and the abstract rules fom the law books through additional information. Beside the General Knowledge Ontology, the reasoner will also be connected to several already existing knowledge bases, which provide more additional information like lexical-semantic information from GermaNet [42] or OpenCyc [43]. With regards to the example shown in Section III, the following example shall illustrate the interaction.

If an individual named "bicycle" is given in the Use Case Ontology as well as a SWRL rule requiring an individual of the class "thing"; the General Knowledge Ontology contains necessary information about the hyperclasses of "bicycle". One of them is the hyperclass "thing". Therefore, the individual of the class "bicycle" can be used for a SWRL rule, which requires an individual of the class "thing".

When working with large amount of information by converting texts from natural language into an OWL model, it is likely to find an inconsistency. This circumstance is not only the result of potential mistakes in the information extraction process, but also inducted by contradictory statements in a text. The difficulty becomes obvious with regards to paragraph 90a of the German Civil Code [7]. It declares that animals are not things even though laws for things shall be applicable for animals as well. Therefore, the reasoning process will have to work with such types of inconsistencies. This problem could be solved by creating and solving two ontologies in parallel, where just one critical statement at a time is given. The result of this type of reasoning would not be a logical but a conclusive solution.

## VI. CONCLUSION

In this paper, we proposed a system for knowledge extraction and analysis from text presented in the domain of law. The system was demonstrated on a typical use case scenario from this domain. Furthermore, we showed how semantic ontologies can improve the information extraction and enable reasoning over the complex knowledge contained in the texts. As a proof-of-concept, a prototype of the analysis system was implemented, equipped with a hard coded rule set. The prototype was used to identify i) abstract concepts as OWL classes, ii) persons and specific entities as OWL individuals, and iii) verbs as OWL object properties correctly. The resulting ontology was tested with the Pellet reasoner [44] and further, the use of the presented approach for handling simple unstructured texts was performed successfully. The advantages of this system design are obvious; it benefits from its high degree of automation enabling the fast adaption to a constantly changing law system. In addition, the OWL ontology is reusable, once generated. Also, the system can be modified adapting to different countries and law systems. Future tasks will focus on several issues like implementing the reasoner and enhancing the presented approach by not only considering isolated sentences but extending the sentence analysis by broadening its scope and applying it on paragraphs as a whole and full texts. Hereby, interferences between sentences will be considered by implementing the Stanford Deterministic Coreference Resolution System [45]. In addition, the currently hard coded rule set will become a flexible more complex one containing a wide range of rules customized to the given context through adapting automated methods composed by making use of machine learning concepts and algorithms for generating tailor made rules. The current Rule-Set represents a small prototype, containing a small set of rules, to proof the concept. It will be the future task, to enlarge the Rule-Set, allowing the ontology generator to handle sentence complexities which is are usual in natural language text. After the rule set is more complex, approximately several hundreds of rules, and flexible, a detailed evaluation, especially a precision and recall test like a F-score, will be done. Besides the full text analysis and the enhanced rule set generation the presented approach will be extended by taking into account external knowledge bases like OpenCyc [43] or GermaNet [42] for improving the general knowledge ontology and thus providing the reasoner with additional information regarding different languages and meaning of terms. Furthermore, the ontology has to be validated by a test set, which does not exist yet. The third challenge is how inevitably emerging logical inconsistencies shall be handled by the reasoner. It will be future work to address this challenge and to develop the algorithms to create a logical net of statements, definitions and connections in order to solve a simple case automatically.

## ACKNOWLEDGMENT

Authors would like to thank the consortium of the EU-ICT research project DreamCloud (Dynamic Resource Allocation in Embedded and High-Performance Computing) [46] as well as the EU-ICT research project JUNIPER (Java platform for hIgh PErformance and Real-time large scale data management) [47] for the support with the Java platform and first prototype of our software system development.

## REFERENCES

[1] R. Schönhof, A. Tenschert, and A. Cheptsov, "Towards Legal Knowledge Representation System Leveraging RDF," The Eighth International Conference on Advances in Semantic Processing, pp. 13-16, Italy 2014, ISBN: 978-1-61208-355-1.

[2] Statistic of the Bundestag, 09.04.2014, p. 5: http://www.bundestag.de/blob/196202/860ee459a5e1d085fd796e376ef3bdd3/kapitel_10_01_statistik_zur_gesetzgebung-data.pdf (retrieved: 06, 2014).

[3] L. Mehl, "Automation in the Legal World from the Machine Processing of Legal Information to the "Law Machine"," Teddington Conference 1958.

[4] G. Sartor, P.Casanovas, M. A. Biasiotti, and M. Fernández-Berrera, "Approches to Legal Ontologies," Springer Press, ISBN 978-94-007-0119-9.

[5] About RDFS: w3.org/TR/rdf-schema/ (retrieved: 08, 2015).

[6] About OWL: http://w3.org/TR/owl2-overview (retrieved: 02, 2015).

[7] N. Mussett, Federal Ministry of Justice and consumer protection Germany, "German Civil Code BGB," Date: 27.07.2011, published in: http://www.gesetze-im-internet.de/englisch_bgb/; Federal Ministry of Justice and consumer protection, Germany: http://www.gesetze-im-internet.de/bgb/xml.zip (retrieved: 06, 2014).

[8] About UCC: law.cornell.edu/ucc/2/ (retrieved: 08, 2015).

[9] About Wikipedia: wikipedia.de (retrieved: 08, 2015).

[10] P. Cimiano and J. Völker, "Text2Onto A Framework for Ontology Learning and Data-driven Change Discovery," Change Discovery, Institute AIFB, University of Karlsruhe 2005.

[11] P. Velardi, S. Faralli, and R. Navigli, "OntoLearn Reloaded: A Graph-based Algorithm for Taxonomy Induction," Computer Linguistics, Vol. 39, No. 3, August 2013, pp. 665-707.

[12] H. Davulcu, S. Vadrevu, and S. Nagarajan, "OntoMiner: Bootstrapping and Populating Ontologies From Domain Specific Web Sites," The First International Workshop on Semantic Web and Databases, 2003, pp. 24-33.

[13] P. Buitelaar, D. Olejnik, and M. Sintek, "OntoLT: A Protégé Plug-In for Ontology Extraction from Text," International Semantic Web Conference (ISWC), 2003, pp. 31-44.

[14] P. Buitelaar, D. Olejnik, and M. Sintek, "A Protégé Plug-In for Ontology Extraction from Text Based on Linguistic Analysis," First European Semantic Web Symposium (ESWS), Heraklion, Greece, May 2004.

[15] P. G. Otero, "The Meaning of Syntactic Dependencies," Linguistik online, Vol. 35, Issue 3, 2008, ISSN 1615-3014.

[16] Stanford NLP Tools: http://nlp.stanford.edu/software/index.shtml (retrieved: 02, 2015).

[17] A. Carnie, "Synatx - A Generative Introduction," Third Edition, chapter 1 - 3, Published by Wiley-Blackwell 2013, ISBN 978-0-470-65531-3.

[18] W. A. Woods, "Transition network grammars for natural language analysis," Communications of the ACM, Vol. 13, Issue 10, p. 591-602, Oct. 1970.

[19] The Stanford Parser: http://nlp.stanford.edu/software/lex-parser.shtml (retrieved: 02, 2015).

[20] M. P. Marcus, B. Santorini, and M. A. Marcinkiewict, "Building a Large Annotated Corpus of English: The Penn Treebank," Computational Linguistics - Special issue on using large corpora, Vol. 19, pp. 313-330, June 1993.

[21] A. Bies, M. Ferguson, K. Katz, and R. MacIntyre, http://www.sfs.uni-tuebingen.de/~dm/07/autumn/795.10/ptb-annotation-guide/root.html (retrieved: 03, 2015).

[22] The Stanford Dependency Parser: http://nlp.stanford.edu/software/stanford-dependencies.shtml (retrieved: 02, 2015).

[23] H. M. Mueller, "Arbeitsbuch Linguistik," Second Edition, Published by Ferdinant Schoeningh 2009, ISBN 978-8252-21969-0.

[24] About Dependencies: http://nlp.stanford.edu/software/dependencies_manual.pdf (retrieved: 02, 2015).

[25] L. Tesnière, "Eléments de syntaxe structurale," Librairie C. Klincksieck, Paris, 1959, Translation: U. Engel: L. Tesnière - "Grundzüge der strukturalen Syntax," Published by Klett-Cotta Stuttgart, 1980, ISBN 3-12-911790-3.

[26] V. Ágel, "Valenztheorie," Published by Narr Tuebingen, 2009, ISBN 3-8233-4978-3.

[27] About Stanford NER: nlp.stanford.edu/software/CRF-NER.shtml.

[28] About GATE/ANNIE: https://gate.ac.uk/sale/tao/splitch6.html (retrieved: 02, 2015).

[29] About RDF: www.w3.org/RDF/ (retrieved: 03, 2015).

[30] About OWL: http://w3.org/TR/owl-features (retrieved: 02, 2015).

[31] About SWRL: http://w3.org/Submission/SWRL (retrieved: 02, 2015).

[32] R. Schönhof, A. Tenschert, and A. Cheptsov, "Information Extraction from Unstructured Texts by means of Syntactic Dependencies and Constituent Trees," The Ninth International Conference on Advances in Semantic Processing, France 2015, ISBN: 978-1-61208-420-6.

[33] About Wiki Dumps: https://dumps.wikimedia.org/backup-index.html (retrieved: 02, 2015).

[34] Law texts: http://gesetze-im-internet.de/Teilliste_translations.html (retrieved: 02, 2015).

[35] The Gate Project: https://gate.ac.uk/ (retrieved: 06, 2014).

[36] The OWLExporter Project: http://www.semanticsoftware.info/owlexporter (retrieved: 06, 2014).

[37] The Protégé Project: http://protege.stanford.edu/ (retrieved: 06, 2014).

[38] The OntoLT Project: http://olp.dfki.de/OntoLT/OntoLT.htm (retrieved: 12, 2015).

[39] The Stanford POS Tagger: http://nlp.stanford.edu/software/tagger.shtml (retrieved: 06, 2014).

[40] The Zurich Dependency Parser for German: http://kitt.cl.uzh.ch/kitt/parzu/ (retrieved: 06, 2014).

[41] T. Li, T. Balke, M. De Vos, K. Satoh, and J. A. Padget, "Detecting Conflicts in Legal Systems," Y. Motomura, A. Bulter, D. Bekki (Eds.), New Frontiers in Artificial Intelligence: JSAI-isAI 2012 Workshops, Revised Selected Papers, LNAI 7856, pp. 174 -- 189 (2013).

[42] GermaNet Homepage: http://www.sfs.uni-tuebingen.de/GermaNet/ (retrieved: 06, 2014).

[43] Cycorp Homepage: http://www.cyc.com/ (retrieved: 06, 2014).

[44] About the Pellet Reasoner: https://www.w3.org/2001/sw/wiki/Pellet (retrieved: 12, 2015).

[45] K. Raghunathan, H. Lee, S. Rangarajan, N. Chambers, M. Surdeanu, D. Jurafsky, and C. Manning, "A Multi-Pass Sieve for Coreference Resolution," EMNLP-2010, Boston, USA 2010.

[46] About DreamCloud: http://www.dreamcloud-project.org (retrieved: 12, 2015).

[47] About JUNIPER: www.juniper-project.org (retrieved: 12, 2015).