Towards Legal Knowledge Representation System Leveraging RDF

Raoul Schönhof, Axel Tenschert, 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*—This paper presents a model usable for a legal system knowledge representation and an implementation of the German Civil Law System as RDF ontology. In this work, different laws are determined in an interconnected structure in order to bridge the gap between computer and social sciences. This model will be created out of natural text, for instance law texts or court decisions, by using a parsing algorithm to build the model, information retrieval tools to extract information and a reasoning algorithm to search and create connections between the particular rules. The focus of this work is to develop the design of the presented model, for an automated reusable entity generation extended by third party knowledgebases.**

Keywords-Knowledge Representation; Law Texts; Ontology; RDF; Big Data; JUNIPER Project.

I. INTRODUCTION

In computer sciences, working with highly unstructured and ambiguous data is a challenge needing to be solved in various research, industrial and social areas. Nonetheless, knowledge is mostly stored implicitly in various formats, e.g., books, articles, websites, data files and so forth. Without an overriding context, these formats contain information. This circumstance and the high complexity leads to the need of improving computer science approaches for enabling social sciences, industries and research to deal with those data. The Resource Description Framework (RDF) [1] syntax allows us to generate relations between instances, consisting of three items: object, subject and predicate. The RDF-Schema (RDFS) enables a mapping of unstructured ambiguous data in a structured manner. Developers are enabled to use RDFS triple stores or ontologies containing logically structured data leading to clearly defined information usable for reasoning tasks. Within the social sciences, there are diverse disciplines like philosophy or political science. The discipline law was chosen because of a well-defined terminology and a clear systematic structure. The thought to exploit legal systems by computer science is old; the first papers about a legal machine were published in the late fifties [2]. Since then, countless approaches have been made. In recent times, there have been several attempts to describe legal knowledge by semantic web languages [3]. Lots of approaches in this area are abstract models. Just a few models were actually generated manually, for example, with the ontology editor Protégé [4]. An automated and realized legal knowledge model for law texts does not exist yet. However, this is necessary; just between 2009 and 2013, Germany resolved 553 federal laws [5] and much more federal state laws.

This work aims to realize a knowledge ontology for the German law system by means of RDF. Center of the law system is the German Civil Code (BGB). It manages and defines fundamental and general issues. The paragraphs are numbered ongoing through the entire BGB. Moreover, most of the single paragraphs are successive subdivided to articles, sub articles and half sentences or numbers. In the scope of this work, German law texts will be explored and structured using RDFS in order to extract information out of this model, being used for automated reasoning processes. By querying the generated RDFS relations, it is possible to comprehend how rules interact and which requirements have to exist to get a legal effect. By matching these requirements with a given case ontology, it could be possible to picture the legal situation of any case. Therefore, this system assists with legal issues by providing legal advice in a fast, user friendly and affordable way.

The paper is structured as follows. Section II gives an introduction into the German legal system and explains briefly, by reference to an example, how different rules can interact together. Section III depicts the system design and shows how legal knowledge ontologies could be generated out of natural texts found in a law book by the use of computer linguistic tools. Conclusively, Section IV deals with the future tasks, as well as the assets and drawbacks.

II. EXEMPLARY SCENARIO

Law texts are not a cluster of isolated rules, but form a complicated network of provision mechanisms and relations. When thinking of relations in law texts, one of the main causes of the complexity of law systems is the aspiration to reduce repetitions as well as the use of an abstract wording. Moreover, the BGB is divided in five chapters. Each chapter manages a special part of possible law issues. The first chapter is called General Part, which is the result of the repetition reduction. It contains mostly definitions and general rules; these are used in the chapters two to five. The second chapter is called Law of

 ISBN: 978-1-61208-355-1

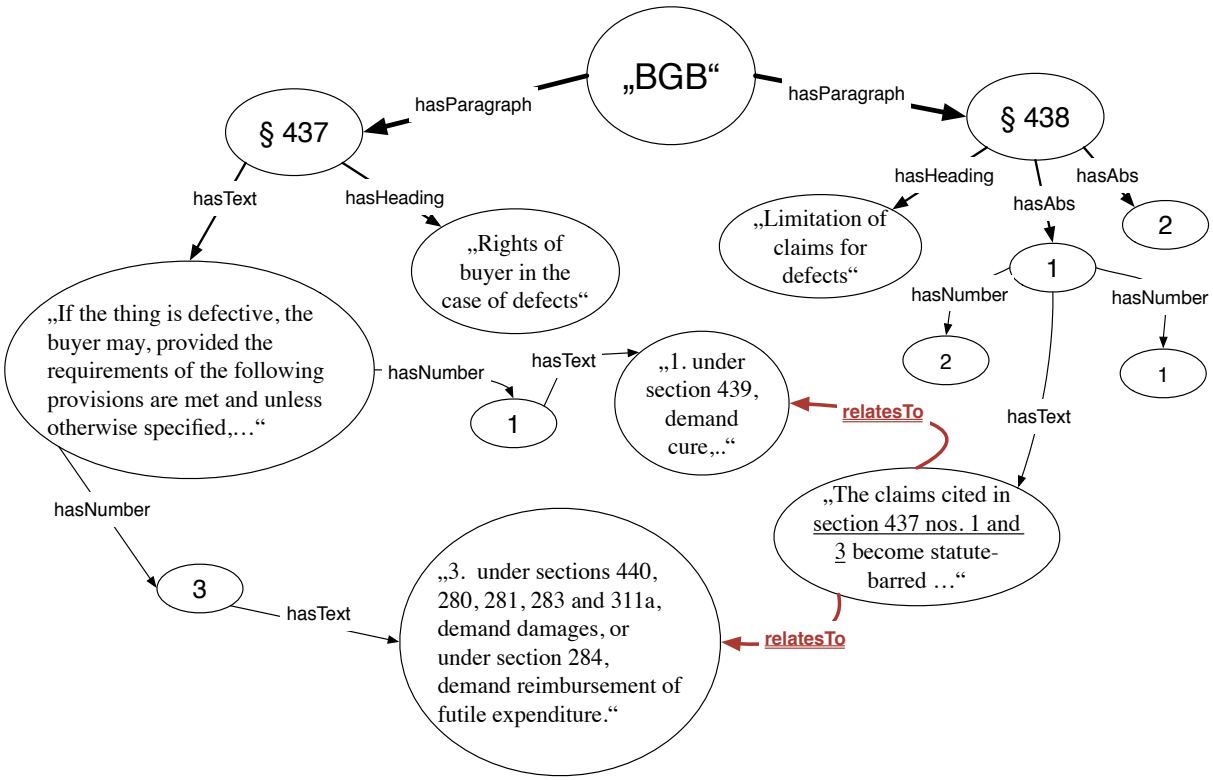

Figure 1: Example of connections in legal text

Obligations. It contains rules 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 general rules and specialized rules makes it possible that two rules regulate one situation in different ways. In such cases, the more general rule is displaced by a more specialized one or a younger rule displaces the older rule. Therefore, rules interact constantly with each other. These mechanisms shall be illustrated based on *§ 437 BGB* and *§ 438 BGB* of the Sales Convention [6]:

§ 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. ” [6].

§ 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.“ [6].

While on the one side, *§ 437 BGB* defines the rights of a buyer in case the purchased object is faulty, *§ 438 BGB* declares on the other side that some of these rights (*§ 437 nr. 1* and *3*) become statue-barred after a certain time [6]. In this example, the rules are connected through named references (see also Figure 1), but it is also common to connect rules 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 vast, therefore a system is necessary supporting non-jurists by estimating legal issues.

III. System Design

The RDF framework is generated in three consecutive steps, which is shown in Figure 2. In the first step, a parsing algorithm creates an initial RDF ontology out of Extensible Markup Language (XML) files. At this point, the model simply pictures the structure of the law texts. In the second step, additional information are extracted out of the law text by using various computer linguistic tools. This information is added to the RDF model as separated entities. Finally, a reasoning method generates the framework by connecting the extracted concepts and references.

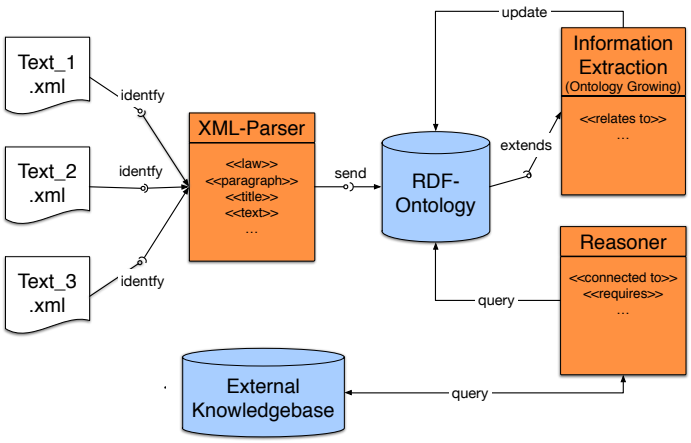

Figure 2: Architecture of the proposed system

A. *Initial RDF Ontology*

The initial model is built by a simple XML-parsing algorithm and creates the hierarchical structure of the law texts in the RDF model. The required XML files with the law texts are open source [6]. The manually provision of XML law files was replaced by an automated crawling algorithm. First, the model contains basic entities, e.g. the law names, the rule numbers and their headings, the particular paragraphs and finally the actual law text. The entities are connected by their own RDF vocabulary called legVoc, which helps to depict the structure of the law texts. Properties of legVoc are for example “hasLaw” to summarize all paragraphs in a law book or “hasSection” in order to connect a paragraph to a superior topic. The structure of the RDF model is illustrated by an extract of *§ 438 BGB* (an example can be found in Figure 3).

```
<rdf:Description rdf:about=”http://gesetzteontologie/BGB/438”>
    <legVoc:hasAbs
   rdf:resource=”http://gesetzteontologie/BGB/438/2”/>
   <legVoc:hasHeading>Limitation of claims for defects
   </legVoc:hasHeading>
   <legVoc:hasAbs rdf:resource=”http://gesetzteontologie/BGB/438/1”/>
</rdf:Description>
<rdf:Description rdf:about=”http://gesetzteontologie/BGB/438/1”>
   <legVoc:hasNumber
      rdf:resource=”http://gesetzteontologie/BGB/438/1/2”/>
   <legVoc:hasNumber
      rdf:resource=”http://gesetzteontologie/BGB/438/1/1”/>
   <legVoc:hasText>The claims cited in section 437 nos. 1 and 3 become
   statute−barred</legVoc:hasText>
```

Figure 3: Listing of RDF extraction

B. *Information Extraction*

After the initial model is generated, 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 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 RDFS 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 [7] and Gate [8] with the OWLExporter plug-in [9] as well as Protégé [10] with its plug-in OntoLT [11].

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 [12]. SNLPG also provides a special Named Entity Recognizer to find and classify salient nouns, e.g., the noun “London” as a location [13]. Furthermore, a sentence parser, e.g., Stanford Parser [14], 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 which will split 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 *§ 1 BGB* [6].

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 [15]. Thus, an alternative is necessary which could be the Zurich Dependency Parser for the German language (ParZu) [16].

 ISBN: 978-1-61208-355-1

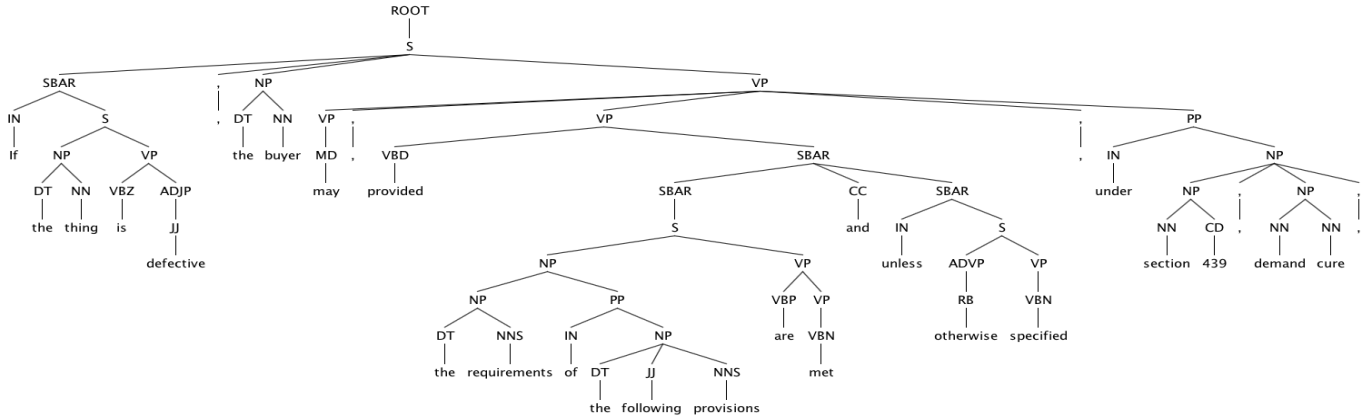

Figure 4: Output of the Stanford Parser

C. Reasoner

The reasoner is in an early stage; its purpose is to identify all rules which could be relevant for a given case. It queries the RDF ontology as well as the case ontology to identify connections between their concepts. Hereby, the reasoner is connected to several already existing knowledge bases which provide additional information like lexical-semantic information from GermaNet [17]. This information is necessary which is illustrated in the following example: A given case mentions the noun "bicycle", which cannot be found in the RDF ontology. For this noun, GermaNet returns a set of synonyms like "bike", "cycle" or "boneshaker" as well as its hyponym "two-wheeler". By following the resulting hyponym-chain, it leads to the words "vehicle" and "thing", which can be found in the RDF ontology. Therefore, each rule, which mentions a "thing", can also be applied for the concept "bicycle".

IV. CONCLUSION

To summarize, the extraction of the initial model out of XML-files was performed with German law texts. Furthermore, the development of an information extraction algorithm is advanced and common concepts were identified. There are also attempts to add information from the universal knowledgebase such as OpenCyc [18]. The advantages of this system design are obvious; it benefits from its high automation-degree enabling the fast adaption to a constantly changing law system. In addition, the RDF ontology is reusable, once generated. Also the system can be modified adapting to different countries and law systems. However, there are several unsolved problems. Firstly, there is no algorithm to transform a rule pattern-based into a logical statement. Secondly, the RDF ontology has to be validated by a test set, which does not exist yet. The third problem is, how inevitably emerging logical inconsistencies shall be handled by the reasoner. It has to be shown if RDFS is complex enough for this purpose, otherwise OWL could be an alternative. It will be the following task to answer this question and to develop the algorithms to create a logical net of statements, definitions and connections in order to solve a simple case automatically. The proposed system implementation will be done leveraging a software technology for Big Data application development, such as JUNIPER Project [19].

ACKNOWLEDGMENT

Authors would like to thank the consortium of the EU-ICT research project JUNIPER (Java platform for hIgh PErformance and Real-time large scale data management) for the support with the Java platform and first prototype of our software system development.

REFERENCES

[1] The RDF Specification: http://www.w3.org/TR/REC-rdf-syntax/ (retrieved: 07, 2014).

[2] L. Mehl: Automation in the Legal World from the Machine Processing of Legal Information to the “Law Machine”, Teddington Conference 1958.

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

[4] P. Mazzega, D. Bourcier, P. Bourgine, N. Nadah, R. Boulet: Chapter 7, A Complex-System Approach: Legal Knowledge, Ontology, Information and Networks, published in: Approches to Legal Ontologies, Springer Press, ISBN 978-94-007-0119-9.

[5] 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).

[6] 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)

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

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

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

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

[11] The OntoLT Project: http://olp.dfki.de/OntoLT/OntoLT.htm

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

[13] The Stanford Named Entity Recognizer: http://nlp.stanford.edu/software/CRF-NER.shtml (retrieved: 06, 2014).

[14] The Stanford Parser: http://nlp.stanford.edu/software/lexparser.shtml (retrieved: 06, 2014).

[15] The Stanford Dependency Parser: http://nlp.stanford.edu/software/stanford-dependencies.shtml (retrieved: 06, 2014).

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

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

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

[19] JUNIPER Project: www.juniper-project.org/ (retrieved: 06, 2014).

 ISBN: 978-1-61208-355-1