Information Extraction from Unstructured Texts by means of Syntactic Dependencies and Constituent Trees

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 technology of automated knowledge extraction from unstructured text corpora by leveraging computer linguistic tools and cross-fertilizing them with the semantic ontologies techniques. In our approach, the quality of information (e.g., in form of OWL ontologies) that is derived by semantic analysis techniques from large domain-specific text corpora can be considerably improved by incorporating linguistic analysis tools that help gain a deeper insight into the grammatical structure of the analysed texts and thus allow the reasoning engines to cover a much wider set of rules and patterns, also positively impacting the performance. The novelty of our approach lies in a possibility of its application to the domains that require a very high quality of the knowledge extraction and analysis, such as reasoning for legacy data collections. We propose a system architecture for the implementation of our approach and illustrate its use on a practical use case for legislative and regulatory information analysis.**

Keywords-***Knowledge Representation; Legal Systems; Ontology; OWL; Big Data; Reasoning; DreamCloud Project.***

I. Introduction

Many domains use texts collected and stored in the natural language as a primary (or, in some cases, the only) trustful source of the domain-specific knowledge. In some cases, this is caused by historical reasons, when the knowledge collection had started long time before the computer standards that allows for a certain level of automation were introduced. In the other cases, the automated storage and processing was impossible by commodity analysis tools due to the complex grammatical constructions used in the texts, as well as their sizes. Whereas the newly-emerged standards like Resource Description Framework (RDF) [1] have enabled tackling with the complex issues of textual representation in the ontologically-understandable format of a logical triple “subject→predicate→object”, the information extraction from grammatically-complex texts remains a major challenge, especially for the domains that require a high precision of the information representation and formalization like law system, patent management, etc. The Semantic Web approach has shown how the information from unstructured sources on the web can be extracted and then used in a wide range of applications, from search and filtering engines to complex reasoning systems that aim to derive new knowledge that is not explicitly provided in the initial variant of texts. A lot of satellite techniques have also appeared around this topic, including the ontology construction tools like Protégé [2], semantic databases like Jena [3], reasoners like Pellet [4], etc. However, the issue of dealing with complex grammatical constructions remains being an essential drawback to promoting those technologies into a wider range of application domains that deal with complex texts analysis.

Most of the information retrieval methods and techniques, such as the language modeling [5], do not consider the grammatical structure of the sentence. However, the latest advances of natural language processing (NLP) technologies, e.g., from the Stanford NLP Group [6], allow those techniques to take some advantages of the grammar-based analysis. In particular, the analysis technologies can be leveraged in the following ways:

- generative grammar tools [7] can be used for extracting functional terms used in the text,
- dependency grammar tools [8] can be used to derive complex connections in form of relations between two words within a sentence.

The remainder of the paper is structured as follows. Section II introduces the state of the art, with focus on computer linguistic tools and related semantic web technologies (such as OWL and SWRL). Section III discusses our analysis approach and presents the design of our system’s prototype. Section IV describes an exemplary scenario based on legislative and regulatory information [9][10] analysis domain, demonstrating the usage of the system’s prototype. Section V presents conclusions.

II. State of the Art

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 [11], OntoLearn/OntoLearn Reloaded [12] OntoMiner [13] and OntoLT [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, particularly tokenization, lemmatizing and shallow parsing [11], 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 [15]. Hereby, OntoLT was using predefined mapping rules for every desired annotation tag. OntoLT then constructs an OWL ontology according to the given mapping rules [15]. 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 [16]. 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 [16] 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 [6]. 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 [7]. The resulting tree of superior and subordinated constituents generates a tree-like structure, which is mapped as an Augmented Transition Network (ATN) [17]. ATN offers a flexible and scalable technology to represent the grammatical structure of sentences. It disassembles a sentence into constituents (see Figure 1) and tags them. A very common constituency parser is the Stanford Parser [18]. 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 1, by using the constituent tags from the Penn Treebank Project [19][20]. 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.

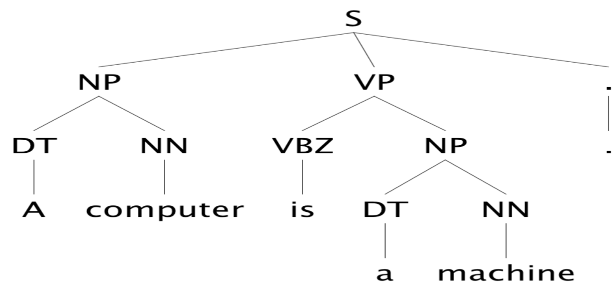

Figure 1. ATN Example based on Stanford Parser GUI

Dependency Parser. Dependency parsers are based on the dependency grammar [8], which focuses on relationships between words and their functional role within a sentence [21]. Relations can be represented in a form of a directed graph, which makes it possible to derivate a hierarchy [21]. Because the structure of the hierarchy is only depending on the grammatical syntax, it is also possible to conclude to the semantic [16], e.g., Figure 2 shows an example sentence with its dependencies and constituents.

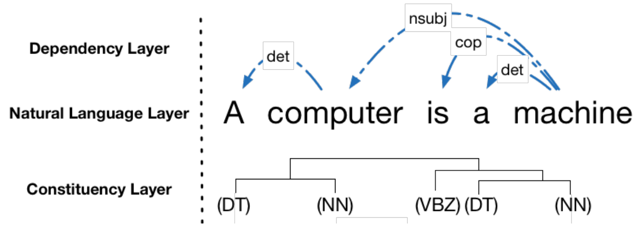

Figure 2. Pattern of a sentence

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* [22] and provide additional information about the type of grammatical relations. Basically, this pattern is representative for a sentence of the type "*Object*

→ *Subject*". Figure 2 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" [8]. Origin of this theory is the empirical knowledge of the structure-determining characteristic of verbs as presented by L. Tesnière. According to this and exposed by H. M. Mueller et al. [21] and V. Ágel [24], 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 [25]. 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) [26] - 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 [27]. OWL is based on the RDF [1] and equipped with an additional vocabulary [28]. 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. [27]

2) *Semantic Web Rule Language*

As a special sublanguage 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 [29]. 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).

$$device(?x) \wedge hasContain(?x,?y) \wedge CPU(?y) \Longrightarrow computer(?x) \quad (1)$$

III. System Arcitecture and Design

A. *General Overview*

The system concept aims to identify applicable laws for a given use case by extracting information fully automated from natural texts. The whole system contains three components shown in Figure 3: the Sentence Processing Unit, the Pattern Interpreter and the Reasoner, which is currently in progress.

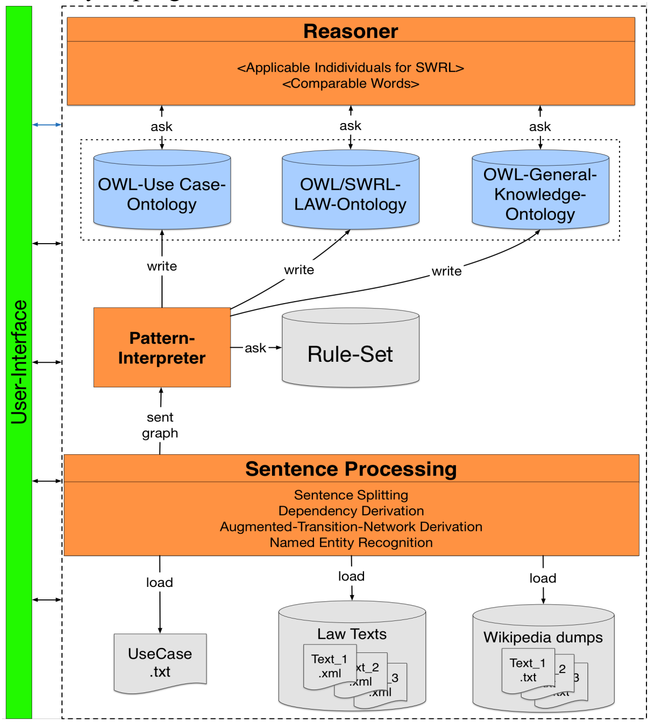

Figure 3. System Architecture

The first component represents the Sentence Processing Unit. It is basically a conglomeration of different language processing tools containing the sentence splitter from ANNIE/GATE [26], the dependency and constituency parser from the Stanford NLP Group [23], as well as a Named Entity Recognizer. The second component is the Pattern Interpreter (see Figure 3). 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.

B. *Sentence Processing*

Starting point is the raw data, which contains texts with one or many sentences. The source of the texts might be Wikipedia [30], law texts [10], 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 2. Therefore, each text passes through the ANNIE sentence splitter of the text-engineering tool GATE [26]. 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.

C. *Pattern Interpreter & Rule-Set*

The Pattern Interpreter translates a given sentence to a machine-recognizable OWL ontology, based on its pattern. The resulting OWL ontology is representing the base for any reasoning attempts. Hereby, the Pattern Interpreter 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 3. If a pattern could be identified, the Pattern Interpreter 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 p 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's apply Rule-Set that to the text mentioned in Figure 2 ("a computer is a machine"). The first rule $rs_1(p_1, i_1)$ contains pattern p_1 that describes a noun (*computer*) referencing to another noun (*machine*) using the dependency *nominal subject*. The corresponding instruction i_1 defines the first noun as a subclass of the second one:

$$i_1 := SubClassOf(Computer, Machine). \quad (1)$$

Now, let's add to our Rule Set another rule rs_2 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*".
The instruction for this rule will be the following:

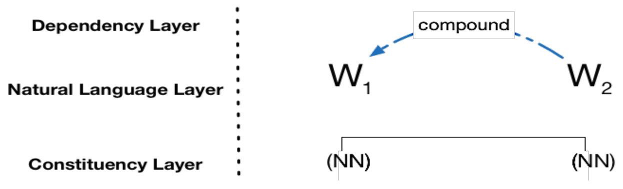

Figure 4. Pattern of rs_2

$$i_2 := Class(ComputerSystem). \quad (2)$$

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", see Figure 5, we'll see that none of them is able to cope with the more complex grammatical structure of the new text. Therefore, the initial rule set should extended by more complex rules, based on the simple patterns discussed above.

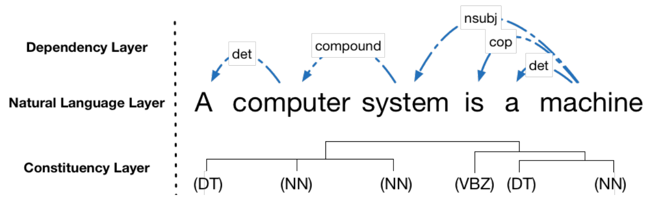

Figure 5. Joint pattern of rs_1 and rs_2

Hereby, the selection of several applicable rules follows the principle of speciality, according to which a more complex rule can be created based on the more simple one. The described patterns exist currently just in hard-coded form to proof the concept. Later, it has to be derivated by automated or semi-automated machine learning algorithms.

D. *Reasoner with OWL Ontologies*

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 tree. The results of the pattern interpreter are, depending of the input source, three OWL ontologies.

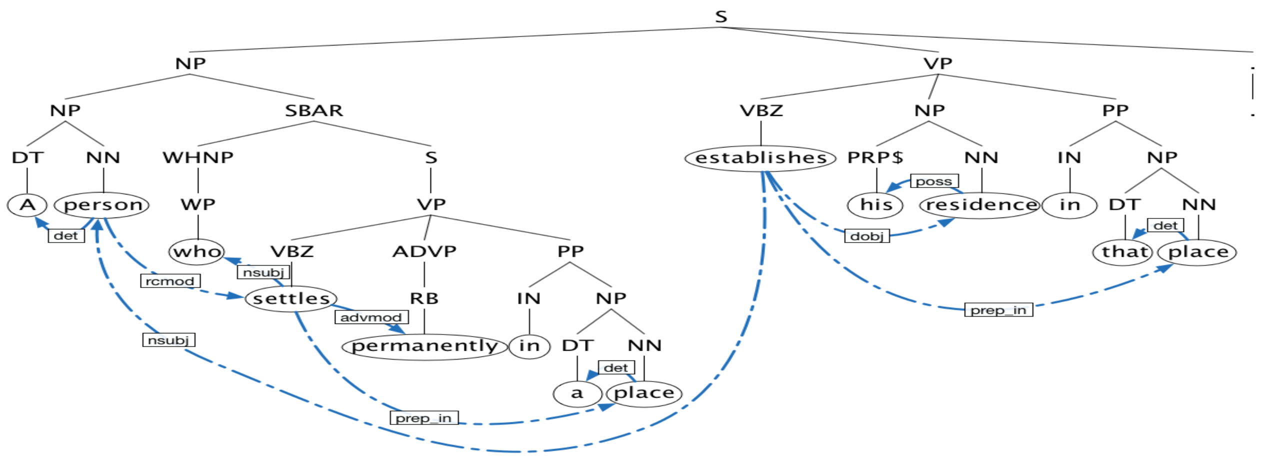

Figure 6. § 7 I BGB, parsed by the Stanford Parser GUI complemented with the dependency relations based on the Stanford Dependency Parser

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. In this state, it would be impossible to find a connection between the given case and the abstract rule. Therefore, it is necessary to bridge the missing links through additional information about the given case. Classes must be linked to hyper- and subclasses, properties like verbs must be associated with other properties. This information shall be extracted by analyzing wikidump files and stored to the General Knowledge Ontology [30]. 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 to 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 problematic becomes obvious with regard to paragraph 90a of the German Civil Code [9]. 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.

IV. Exemplary Use Case Scenario

We would like to illustrate the application of our system for the analysis of the following text from a paragraph (§7) of the German Civil Code: "*A person who settles permanently in a place establishes his residence in that place*." [9]. At first the sentence passes through the sentence processing unit, which derives an ATN and the dependencies, shown in Figure 6. In addition, the words (tree leaves) are exchanged to their lemma in order to reduce the complexity for reasoning tasks. Root point of the ATN is the constituent sentence (S). It consists of a noun phrase (NP), as well as a verbal phrase (VP). Here, the ATN depicts the difference between the legal prerequisite, the noun phrase (NP), and the legal consequence, the verbal phrase (VP). The dependency tree shows the relation between words. Root point of this dependency tree is the verb "*establishes"*. The root point is outstanding, because it has no dependency pointing at it, but one or more, which point away from it. This verb declares the action "*establishes*" for the nominal subject (nsubj) "*person*". But this noun is restricted by a sub-ordered conjunction (SBAR) [19]. Here, the noun "*person*" is getting conditioned by the clause "*settle permanently in a place*". Hereby, "*settle*" itself refers firstly to "*place*" via the preposition "*in*" (prep) and secondly to its modifier "*permanent*". The legal consequence of this sentence is contained in the verbal phrase. The verb "*establish*" in connection with the direct object (dobj) "*residence*". The main task of the pattern interpreter is to look if patterns, given from the pattern set, could match in this constituents and dependency tree. At this point of time, the actual words, respectively the content of this sentence, does not matter anymore. Depending on the pattern, nouns are converted to OWL classes or individuals. Here, Figure 6 shows three types of nouns: "*person*", "*place*" and "*residence*". By treating these nouns as OWL classes, it is possible to associate individuals to them. Beside nouns, verbs are converted to OWL object properties. The given sentence contains just the two verbs "*establish*" and "*settle*", which is restricted by the adverbial modifier (advmod) "*permanent*". In SWRL, the first noun phrase is true if there are two individuals, one of the class "*person*" and one of the class "*place*", which are connected by an object property "has*SettlePermanent*", see equation 8. The antecedent of the SWRL rule contains classes "*person*"

$$person(?x) \land hasSettlePermanent(?x,?y) \land place(?y) \Longrightarrow hasEstablishResidence(?x,?y) \qquad (3)$$

and "*place*", which are connected by the object property "has*SettlePermanent*", as consequence the individual of the class "*place*" is also declared as individual of the class "residence". Also the new object property "*EstablishResidence*" will be inserted and connects then "*person*" and "*place*".

V. CONCLUSION

In the paper, we showed how the ontology-based reasoning techniques can be improved by leveraging the syntactical analysis tools. A system architecture, as well as a use case scenario from the law domain were presented. As a proof-of-concept, a prototype implementing the system architecture was implemented based on the Java toolset from the DreamCloud project [31] was 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 and further, the use of the presented approach for handling simple unstructured texts was performed successfully. The described work serves mainly as a foundation for further research and development activities.

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. 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. After the rule set is more flexible, a detailed evaluation will be done. Besides the full text analysis and the enhanced rule set generation the presented approach will be extended by taking into account a thesaurus for improving the general knowledge ontology and thus providing the reasoner with additional information regarding language and meaning of terms.

The work done in the scope of this paper and the future developments will conclude in a flexible, syntactic dependencies and constituent tree handling, as well as meaning aware reasoning system being able to handle laws and further being applicable to other unstructured text types.

ACKNOWLEDGMENT

Authors would like to thank the consortium of the EU-ICT research project DreamCloud (Dynamic Resource Allocation in Embedded and High-Performance Computing) for the support with the Java platform and first prototype of our software system development.

REFERENCES

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

[2] About Protégé: www.protege.stanford.edu/ (retrieved: 02, 2015).

[3] About Jena: jena.apache.org/ (retrieved: 03, 2015).

[4] About Pellet: clarkparsia.com/pellet/ (retrieved: 03, 2015).

[5] J. M. Ponte and W. B. Croft, "A Language Modeling Approach to Information Retrieval", University of Massachusetts, http://ciir.cs.umass.edu/pubfiles/ir-120.pdf (retrieved: 03, 2015).

[6] Stanford NLP Tools: http://nlp.stanford.edu/software/index.shtml.

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

[8] 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.

[9] N. Mussett, Federal Ministry of Justice and consumer protection Germany: German Civil Code BGB, Date: 01.10.2013, published in: http://gesetze-im-internet.de/englisch_bgb (retrieved: 02, 2015).

[10] Law texts: http://gesetze-im-internet.de/Teilliste_translations.html.

[11] P. Cimiano and J. Völker, "Text2Onto A Framework for Ontology Learning and Data-driven Change Discovery", 2005.

[12] 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.

[13] 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.

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

[15] P. Buitelaar, D. Olejnik, 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.

[16] P. G. Otero, "The Meaning of Syntactic Dependencies", Linguistik online, 2008, Vol. 35, Issue 3, pp. 33-53, ISSN 1615-3014.

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

[18] Stanford Parser: http://nlp.stanford.edu/software/lex-parser.shtml.

[19] 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.

[20] 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.

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

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

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

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

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

[26] About GATE/ANNIE: https://gate.ac.uk/sale/tao/splitch6.html.

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

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

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

[30] M. Pataki, M. Vajna, and A. Marosi, "Wikipedia as Text", Ercim News - Special theme: Big Data, Vol. 89, 2012, pp. 48-49, About Wiki Dumps: https://dumps.wikimedia.org/backup-index.html.

[31] About DreamCloud: http://www.dreamcloud-project.org.