STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LV**, Number 4, 2010

# ONTOLOGY ASSISTED FORMAL SPECIFICATION EXTRACTION FROM TEXT

ANDREEA-DIANA MIHIŞ

Abstract. In the field of knowledge processing, the ontologies are the most important mean. They make possible for the computer to understand better the natural language and to make judgments. In this paper, a method which use ontologies in the semi-automatic extraction of formal specifications from a natural language text is proposed.

Between ontologies and natural language exists an old connection. The first ontologies created in the computer science were written in natural language. But in the natural language form they were not very useful. So, a formal way of representing ontologies was preferred. Some early formal forms were Hyper Text Markup Language (HTML) and Unified Modeling Language (UML). Now, the ontologies are represented in Resource Description Framework (RDF) or Web Ontology Language (OWL)[16], and in these forms they have many applications in the Semantic Web. Also, some of the existing ontologies were obtained in an automatic or an semi-automatic way from natural language texts.

In the Semantic Web, the ontologies are used for the semantic information coding behind a Web page or in a Web page, and in this way they support information retrieval. But the majority of the existing ontologies are designed for a specific domain, and they are used in that specific domain. And, as they have proved their usefulness in those domains, whay not for the specification extraction from natural language texts?

When the clients and the developers meet, the client's requirements are written in natural language, since the natural language is commonly understood. But, unfortunately, the natural language is ambiguous. In the requirements phase, the developers and the clients understand the requirements in the same way. But this does not guaranty that they will recall the same things

Received by the editors: December 3, 2010.

2010 *Mathematics Subject Classification.* 68T30, 68T50.

1998 *CR Categories and Descriptors.* I.2.7 [**Computing Methodologies**]: Artificial Intelligence – *Natural Language Processing* D.2.1 [**Software**]: Software Engineering – *Requirements/Specifications*;

*Key words and phrases.* Requirements, Ontology, Fformal Specification.

later. Maybe they are not the only persons which will develop the software. Usually, especially for large scale or average scale projects, the development team is split in smaller teams, specialized in a particular phase from the development process: specification, coding, testing, and so on. Sometimes the team responsible with the requirement elicitation is so specialized, that it can even belong to a different firm. And the requirements elicitation techniques gathers a lot of natural language requirements, which must be analyzed and transformed in specifications, from which the most useful are formal specifications. The purpose of the requirement analysis phase is the development of the Specification Document, which is in fact the contract between client and developers, and in order to be unambiguous, formal specifications must be used[17].

A method of semi-automatic formal specification extraction from requirements written in natural language is proposed in this paper.

## 1. Ontology

An ontology represents a rigorous and exhaustive organization of some knowledge domain that is usually hierarchical and contains all the relevant entities and their relations [28]. It's an old field of study in philosophy, with greek origins. "Ontology (from the Greek $\ddot{o}\nu$, genitive $\ddot{o}\nu\tau o\varsigma$: "of being" (neuter participle of $\epsilon\tilde{\imath}\nu\alpha\iota$: "to be") and $\lambda o\gamma\acute{\imath}\alpha$, -logia: science, study, theory) is the philosophical study of the nature of being, existence or reality in general, as well as the basic categories of being and their relations. Traditionally listed as a part of the major branch of philosophy known as metaphysics, ontology deals with questions concerning what entities exist or can be said to exist, and how such entities can be grouped, related within a hierarchy, and subdivided according to similarities and differences." [29].

In the field of computer science, the ontology is a relatively new field of study, but promising. As in philosophy, it intends to organize the knowledge from the web in a semantic form. And because the web is used as a support to publish information from different domains, also, the ontologies differ, some being domain specific, some being upper level ontologies [15]. The first ontologies used in the computer science domain were defined using natural language. These kinds of ontologies were called informal ontologies. But the most formal ones have the most applicability [5, 2]. Today, the backbone of the Semantic Web is consisted by the OWL (Web Ontology Language) and RDF (Resource Description Framework) [16].

A lot of ontologies were developed, by humans, as the work of different specialists or volunteers [20], to semi-automatic [13] and automatic ways [4]. Ontologies also differ in respect to the scope and purpose of their content.

The most prominent distinction is between the domain ontologies describing specific fields of endeavour, like economics or biology, and upper level ontologies describing the basic concepts and relationships invoked when information about any domain is expressed in natural language [22]. Usually the domain ontology is subordinated to the upper level ontology [15] (see Figure 1).

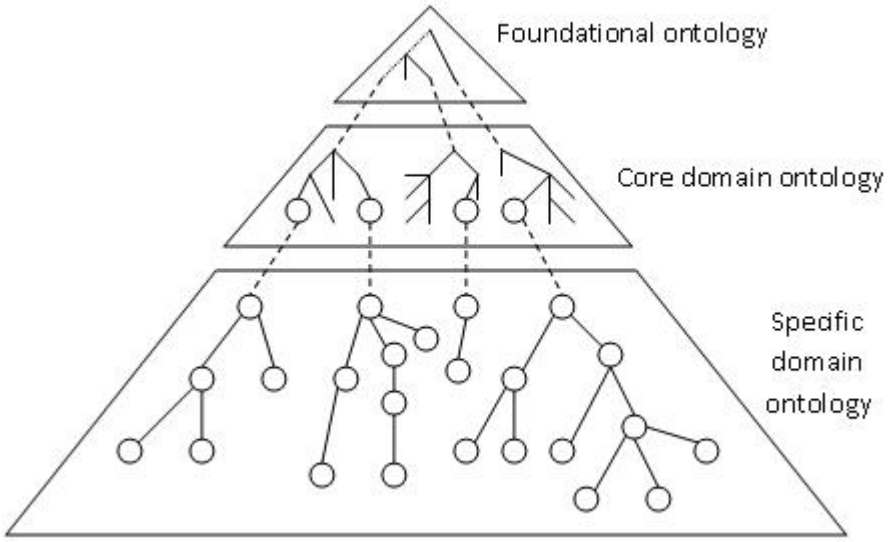


FIGURE 1. The tree levels of generality in a domain ontology [15]

From the representation point of view, one type of ontology is the taxonomy. A taxonomy is a hierarchical classification of terms. The relation between two terms, a parent and a child, is usually an "is a" relation. The terms can also be characterized by a set of properties.

But the most general way to define an ontology is by using triplets [3]. A triplet is composed from a subject or the concept, the predicate, which is a directional relation and the object, which is a characteristic. The relation and the characteristic can also be concepts. Any type of ontology can be represented in the triplet form, even if the ontology is a taxonomy or it has an integrate graph type structure.

## 2. Related work

Since the ontologies make the computer capable to better understand the natural language, they have a great applicability in the field of information

extraction. Since 2003, their applicability for information extraction was emphasized in [14], where the authors propose a step by step method for enriching an existing ontology in order to make it more appropriate for information extraction from a new text source.

In a newer paper [21], an ontology unsupervised method for information extraction is presented, without using any other knowledge sources. However, the precision of the results will depend on the quality of the input ontology. The method also identifies unused elements from the ontology, and in this way the quality of the ontology can be improved, and also the results of the extraction.

## 3. The semi-automatic formal specification extraction

In a previous paper [10], was proposed an application which assist the Stepwise Refinement process. In that application, the Stepwise Refinement process started with an abstract program [6], and the goal of the refinement was to obtain code. In another paper [11], another application which assist the Z schema usage, and also transform a Z schema into an abstract program was presented. The goal of this paper is to extract an abstract program from a natural language text which represent a program requirement. In other words, the goal is to obtain the specifications, i.e. the precondition, the postcondition and the variable list from natural language requirements. In natural language, the sentences which correspond to the precondition respective postcondition can be selected by the tense of their predicates. And the variables involved are usually the subjects of those sentences.

There are two types of requirements: requirements which are expressed by many sentences, and many requirements expressed in a single sentence. In the first case, the identification of the preconditions and postconditions seems to be much easier to be resolved, since it can be reduced to a sentence selection problem. In the second case, a single sentence must be split into one or more preconditions and one or more postconditions. In the second case, the sentence itself must be analyzed.

In the field of natural language processing, the accuracy of some natural language processing methods has improved. Some still have an accuracy less then 70%, such as the text entailment relationship [1]. But there are a lot of tools capable of identifying the correct part of speech of words from sentences, with an accuracy more than 95% [18]. In the field of grammatical analysis of a sentence also a lot of work was done, and there are available free tools capable to analyse from the syntactic point of view a sentence, such as the online tool developed by the Stanford Natural Language Processing Group [26], with an

accuracy of more than 87% [8], tool which is updated continuously [27]. The result of the grammatical analysis of a sentence is usually represented as a tree. Or, between the words of a sentence, dependence relations can be identified [9], relations which are astonishing similar to the RDF triplets, the simplest way to represent an ontology.

3.1. **Sentence Analysis.** In a previous paper [12], the output of the Stanford Parser was used to extract ontology triples, in order to identify the best ontology which matches a natural language text. In this paper, Stanford Parser is used again, and the dependence relations will be used as ontology triples concept-predicate-object, where the concept is the first word from the dependency, the predicate is the dependency, and the object is the second word from the dependency.

So, for instance for the following requirement: *"Generate the first prime number larger than a given natural number n."*[23], the syntactic analysis tree can be seen in the figure 2, the dependencies list in Figure 3 and the associated graph for the dependencies in Figure 4.

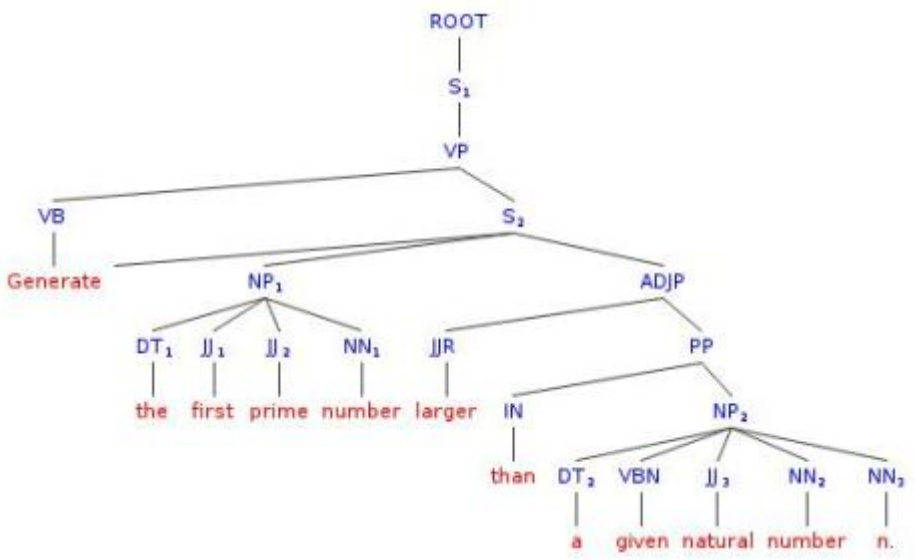


FIGURE 2. The syntactic analysis tree

From the dependencies graph, it can be observed that the sentence has two main parts: Generate + larger + number (the subject) + the + first + prime, and larger + n. + a + given + number + natural. In the first list, Generate is a VB (verb, base form: imperative, infinitive or subjunctive [25]),

*det*(number-5, the-2)
*amod*(number-5, first-3)
*amod*(number-5, prime-4)
*nsubj*(larger-6, number-5)
*xcomp*(Generate-1, larger-6)
*det*(n.-12, a-8)
*amod*(n.-12, given-9)
*amod*(n.-12, natural-10)
*nn*(n.-12, number-11)
*prep_than*(larger-6, n.-12)

FIGURE 3. The dependencies list

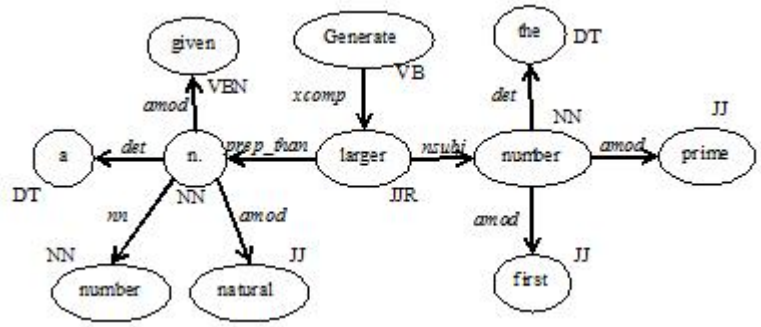


FIGURE 4. The dependencies graph

and is an imperative verb, so this list will represent the postcondition. In the second list, given is a VBN (verb, past participle [25]) and is in the past tense, so the second list will be the precondition. The two nouns, "n" and "number" will represent the variables.

The same conclusion can be reached by using the syntactic analysis tree. "the first prime number" is a NP (noun phrase [24]), and "a given number n" is another NP, subordinate to the first one. Because the verb is an imperative one, the first NP will represent the requirement, and with the first NP, will compose the postcondition.

In the case in which more sentences represent the requirements, such as in the case of: "The sequence a1, ..., an with distinct integer numbers is given. Determine all subsets of elements with sum divisible to n."[23], is easy to select the sentences by their tense. In the first sentence "given" is a VBN, and in the second "Determine" is a VB. SO, the first sentence will represent the

precondition, and the last the postcondition. Between the two sentences is a connection, but it is not an evident one. The subject of the first sentence, "the sequence" appears in the second, but not directly. There is a connection between the object from the second sentence, "subsets" and "the sequence", but the connection is a logical one, and in order to be recognized by the computer, new knowledge must be available. The most profitable is the existence of an ontology, where sequence is similar with set, and subset is a subordinate of the set, or one in which subset and sequence are related directly. If no such information is available, then the similarity of the two words will provide a hint for their relationship. The similarity may be computed between the two sentences or text to which the two words belong, or, between their glosses from a dictionary, if the words were disambiguated. If this kind of connections can be made, then the sentences graphs can be merged, and considered as a big graph as input for the following subalgorithm:

```
Subalgorithm PrePostSelection(g, pre_list, post_list)
    DATA: g - a dependencies directional graph
    @Identify the VB words list: VB_list
    @Initialize a list of lists of words, Word_ll and place in
     every list on the first position the VB word
    @Initialize an empty list of distinct words, Var_l
    For @every list L from Word_ll do
        @Identify the graph path which ends with a dependency
         "sub" for a VB or a VBD, and if none exist, the path
         which ends with a dependency "obj", respective the path
         which ends with an "amod" dependency for a VBN
        @Add all the words from the path to L
        @The NN word which is in relation "sub", "obj" or "amod"
         must be added to the variable list Var_l
        @Add all to L all the words connected to the last word
         in a recursive way (they and all the words connected to
         them)
    endfor
    RESULTS: pre_list - the lists L from Word_ll which starts
                        with a VB
             post_list - the other lists from Word_ll
             Var_l - the list of variables
endSub
```

In the case in which the requirements text contains many sentences, the proposed algorithm can be applied after a ontology alignment operation. From every sentence, the dependency graph can be constructed, and this graph is a

mini-ontology (it contains entities and relations). This is the way an ontology alignment process can be used to merge all the sentences graph into a larger graph.

For an even larger requirement text, the text must be segmented first, then for every segment, the sentences dependency graph merged, and then the proposed algorithm can be applied.

3.2. **Ontology assisted.** For the further refinement from natural language preconditions/postconditions into abstract programs, or Z schemas, is necessary the use of an ontology, which, for instance for the case of Z schemas, has as base-nodes the base types and the base operations for these types. For instance, from the first example, *"Generate the first prime number larger than a given natural number n."*[23], it can be noticed that, the basic type natural number will be used, with prime and larger than as operations, and that only first is not a simple predicate involving a natural number.

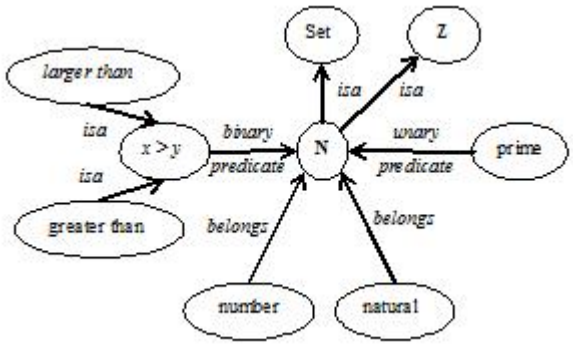


FIGURE 5. A part of Natural numbers ontology

In this case, first words which represent the variables will be searched in the ontology (see Figure 5), and replaced with the closest base type (after the node "natural" is identified, the closest base type is **N**). Then, from the natural language preconditions and postconditions all the words are searched between the sub-ontology with the base root the base type identified (If the variable list is formed from many variables, all with different types, a union of sub-ontology will be used). If they are found, then the base predicate directly connected to the base type will replace the current word.

In the discussed example, *"Generate the first prime number larger than a given natural number n."*[23], the identified sentence which express the post-condition is Generate + larger + number (the subject) + the + first + prime.

Number can be replaced by $x \in \mathbf{N}$, and also larger $x > n$. The computer can provide these base types/predicates, and in this way to assist the translation process from natural language into formal specifications.

## 4. Experiments

The proposed algorithm was implemented in C++ and tested on the laboratory requirements from Fundamentals of Programming [23]. It worked well with natural language requirements. Because these formulas are not natural language sentences, the Stanford parser cannot identify any connection between the elements of the formula. So, these elements will be disconnected from the other words from the dependencies graph. As the Stanford parser output is the current algorithm input, they were ignored further, and the quality of the results suffered. Unfortunately a more appropriate set of test was not found. A solution to overcome this problem is to replace these formulas with different symbols, and only then to apply the algorithm.

## 5. Conclusions and Future work

In this paper a method for semi-automatic formal specification extraction from a natural language text was presented. It uses the Stanford Parser to obtain the dependency graph. The dependency graph is seen as a mini-ontology of the sentence. Then ontology alignment process is used to merge the mini-ontologies. In the end, using semantic principles, natural language sentences which represents the preconditions and postconditions are extracted.

In the future, I wish to construct manually an ontology for the Z language base types, and using it, to further refine the identified natural language preconditions and post-conditions into formal specifications.

## 6. Acknowledgements

This work was supported by ANCS-CNMP, project number PNII - 91037 /2007.

## References

[1] Bar-Haim, R., Dagan, I., Dolan, B., Ferro, L., Giampiccolo, D., Magnini, B. and Szpektor, I., *The Second PASCAL Recognising Textual Entailment Challenge*, Venice, 2006.

[2] Benett, B. and Fellbaum, C., *Formal Ontology in Information Systems*, Formal ontology in information systems: proceedings of the fourth international conference, IOS Press, Amsterdam, 2006.

[3] Brachman, R. and Schmolze, J., *An Overview of the KL-ONE Knowledge Representation System*, Cognitive Science, vol. 9, no. 2, 1985, pp. 171-216, http://nlp.shef.ac.uk/kr/papers/klone.ps

[4] Dahab, M., Hassan, H. and Rafea, A., *TextOntoEx: Automatic Ontology Construction from Natural English Text*, International Conference on Artificial Intelligence and Machine Learning (AIML-06), Sharm El Sheikh, Egypt, June 13-15 2006, http://www.icgst.com/con06/aiml06/Final_Articles/P1120615104.pdf

[5] Davies, J., Fensel, D. and van Harmelen, F., *Towards the Semantic Web: Ontology-driven Knowledge Management*, John Wiley & Sons Ltd, 2002, New York.

[6] Dijkstra, E.W., *Guarded commands, nondeterminacy and formal derivation of programs*, Comm. ACM, Vol. 18(1975), Nu. 8, pp. 453-457.

[7] Euzenat, J. and Shvaiko, P., *Ontology Matching*, Springer, 2007.

[8] Klein, D. and Manning, C. D., *Fast Exact Inference with a Factored Model for Natural Language Parsing*, in Advances in Neural Information Processing Systems 15 (NIPS 2002), Cambridge, MA: MIT Press, pp. 3-10.

[9] de Marneffe, M.-C. and Manning, C. D.,*The Stanford typed dependencies representation*, in COLING Workshop on Cross-framework and Cross-domain Parser Evaluation, Manchester, United Kingdom, 2008, http://nlp.stanford.edu/pubs/dependencies-coling08.pdf.

[10] Mihiş, A.D., *An Application That Assist StepWise Refinement*, Proceedings of Symposium "Zilele Academice Clujene", 2006, pp. 143-147.

[11] Mihiş, A.D., *A Tool for Refinenent of Z Schemas*, Proceedings of Symposium "Zilele Academice Clujene", 2010, pp. 64-67.

[12] Mihiş, A.D., *Ontology Learning from Text Based on the Syntactic Analysis Tree of a Sentence*, The 5th International Conference on Virtual Learning (ICVL - 2010), Târgu Mureş, Edituara Universităţii Bucureşti, Editor: Vlada Marian, Albeanu Grigore, Popovici Dorin Mircea, ISSN: 1844-8933, 1842-4708, http://c3.icvl.eu/2010, 2010, pp. 128-134, Intel®Special Award winning article.

[13] Morita, T., Fukuta, N., Izumi, N. and Yamaguchi, T., *DODDLE-OWL: A Domain Ontology Construction Tool with OWL*, Proceedings of the 1st Asian Semantic Web Conference, Lecture Notes in Computer Science, vol. 4185, Beijing, China, 2006, pp. 537-551, http://iws.seu.edu.cn/resource/Proceedings/ASWC/2006/papers/4185/41850537.pdf

[14] Maedche, A., Neumann1, G. and Staab, S., *Bootstrapping an Ontology-based information extraction system*, in Intelligent exploration of the web, Editors: Piotr S. Szczepaniak, Javier Segovia, Janusz Kacprzyk, Lotfi A. Zadeh, Physica-Verlag GmbH Heidelberg, Germany, 2003, pp. 345 - 359.

[15] Navigli, R., Velardi, P. and Gangemi, A., *Ontology Learning and Its Application to Automated Terminology Translation*, IEEE Intelligent Systems, vol. 18, no. 1, January 2003, pp. 22-31, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.79.1758&rep=rep1&type=pdf

[16] Pollock, J. T. *Semantic Web for Dummies*, Wiley Publishing, 2009, Indianapolis, Indiana.

[17] Pressman, R. S., *Software Engineering - A Practitioners Approach*, McGraw Hill, third edition, 1992.

[18] Toutanova, K., Klein, D., Manning, C. D. and Singer, Y., *Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network*, in Proceedings of HLT-NAACL 2003, Edmonton, Canada, pp. 252-259.

[19] Walther, M., Hhne, L., Schuster, D. and Schill, A., *Locating and Extracting Product Specifications from Producer Websites*, 12th International Conference on Enterprise Information Systems (ICEIS), Funchal, Madeira, Portugal, 2010.

[20] Wilkinson, M. and Good, B., *Construction and Evaluation of OWL-DL Ontologies*, Microsoft Research Faculty Summit, Redmond, USA, July 17-18 2006, http://research.microsoft.com/en-us/um/redmond/events/fs2006/agenda_mon.aspx
[21] Yildiz, B. and Miksch, S., *ontoX - a method for Ontology-driven information extraction*, Proceedings of the 2007 international conference on Computational science and its applications - Volume Part III, Workshop on CAD/CAM and web based collaboration (CADCAM 07), Springer-Verlag Berlin, Heidelberg 2007, pp. 660-673.
[22] Zervas D., *WS-TALK - Automatic Ontology Construction*, http://thames.cs.rhul.ac.uk/ wstalk/papers.html, 2005.
[23] http://www.cs.ubbcluj.ro/ adriana/Teaching.html
[24] http://grammar.about.com/od/mo/g/nounphraseterm.htm
[25] http://bioie.ldc.upenn.edu/wiki/index.php/POS_tags
[26] The Stanford Natural Language Processing Group, *Stanford Parser*, http://nlp.stanford.edu:8080/parser/
[27] The Stanford Natural Language Processing Group, *The Stanford Parser: A statistical parser*, http://nlp.stanford.edu/software/lex-parser.shtml
[28] http://wordnetweb.princeton.edu/perl/webwn?s=ontology
[29] http://en.wikipedia.org/wiki/Ontology
[30] http://www.wordnet-online.com/business.shtml

Babeş-Bolyai University, Faculty of Mathematics and Computer Science, Department of Computer Science, 1 M. Kogălniceanu St., 400084, Cluj-Napoca, Romania

*E-mail address*: mihis@cs.ubbcluj.ro