ONTOLOGY-BASED INTELLIGENT AGENT FOR SEMANTIC PARSING THE NATURAL LANGUAGE SPECIFICATIONS OF SOFTWARE REQUIREMENTS

Tetiana Hovorushchenko[1], Artem Boyarchuk[2], Olga Pavlova[1]

[1] Khmelnytskyi National University,
[2] National Aerospace University "Kharkiv Aviation Institute"
e-mails: tat_yana@ukr.net, a.boyarchuk@csn.khai.edu,
olya1607pavlova@gmail.com
Ukraine

Abstract: The paper is devoted to the development of an ontology-based intelligent agent (OBIA) for semantic parsing the natural language specifications of software requirements (SRS), which performs the parsing of the specification, determines the number and percentage of missing attributes. This allows to determine the non-functional characteristics-components of the software quality, reflect which attributes are missing for one or another sub-characteristics of the non-functional characteristic, and form a real ontology for the nonfunctional characteristics.

Key words: Software requirements specification (SRS), Non-functional software characteristics, Natural language processing, Ontology.

1. INTRODUCTION

Currently more than 250 billion USD is spent for the realization of approximately 175 thousand software projects annually [1]. At the same time, a significant number (on average up to 70%) of the software projects are unsuccessful (including 50% – challenged projects, 20% – failed projects) [1]. Moreover, the statistics of success varies greatly between small and large software projects (among successful software projects – 62% small projects, 6% large projects and 2% grand projects; among failed software projects – 11% small projects, 24% large projects and 17% grand projects) [1, 2].

A significant amount (10-23% [3], up to 56% [4]) of bugs is made at the stage of the formulation of software requirements specifications (SRS) based on business requirements, and the number of bugs increases with an increase in the size of the software project. About 50% of defects of requirements resulted by unclear or ambiguous requirements; the other 50% is the result of the incompleteness or insufficiency of specification (incomplete, insufficient or omitted requirements) [4]. Papers [5, 6] confirm that the causes of many software incidents and catastrophes are contained in the SRS. Levenson [6] describes the results of an experiment, which was conducted to confirm the hypothesis that crashes and bugs of software, which is written by different developers on one specification, are statistically independent. It was found that software versions, which were written by different developers on the same requirements, contained a number of common errors due to requirements errors. Such experimental evidence directly leads to the need to deepen the analysis of the SRS.

2. STATE-OF-THE-ART

Given that 50% of defects in requirements are the result of incompleteness or insufficiency of SRS [4], the important task is precisely the definition of the sufficiency of information in the SRS. In particular, how complete is the information on the restrictions on future software. Particular attention, in this case, should be paid to the non-functional characteristics. In [7], the theoretical principles of information technology of evaluating the sufficiency of information on quality in the SRS are developed, but the limitation of such a solution is its theoretical orientation – the search of information in the specification and the assessment of its sufficiency are manually performed. In [8], the ontology-based intelligent agent (OBIA) for evaluating the initial stages of the software lifecycle was developed, which is designed to analyze the SRS on the subject of the sufficiency of their information on the non-functional characteristics-components of software quality (Functional Suitability, Performance Efficiency, Reliability, Compatibility, Usability, Portability, Maintainability, Security – according to ISO 25010: 2011). But the disadvantage of this solution is the manual approach to the choice of information on the non-functional characteristics from the SRS. It is necessary to automate the semantic analysis of the natural language specification on the subject of the search of attributes, which are needed to determine the non-functional characteristics-components of software quality.

Consequently, today the analysis of the SRS for the compliance of the non-functional requirements with the needs of the customer (in particular, the sufficiency of their information) with the purpose of improving the software projects success, is the *unresolved and actual problem*. To solve such a problem, the *actual task* of automating the semantic parsing of the natural language specification should be solved, in particular, the automated search for attributes, which are necessary to determine the non-functional characteristics-components of the software quality. This study is aimed to solve this task.

There are different approaches to the semantic analysis of specifications of software requirements. In [9], a method is developed to provide automatic assistance to the developers by transforming the natural language requirements using UML diagrams. The authors [9] focus on creating the activity diagrams and sequence diagrams based on natural language SRS. Standard part-of-speech-tagger and a parser analyze requirements in English, provided by users, and "extract", from them, phrases, actions, and so on. The method, proposed in [9], reduces the gap between the informal natural language and the formal modelling language.

Selway et al. [10] is devoted to solve the problem of transforming the natural language business specifications into formal models, which are suitable for use in the development of information systems. The authors propose a method for transforming the natural language specifications based on the Semantics of Business Vocabulary and Business Rules (SBVR) standard [11]. This approach provides reliable semantic processing of specifications and reduces their manual formalization.

The methodology, proposed in [12], consists of four processes: parsing the requirements; reflecting the mapping of requirements using matrix; adding the requirements in the specification template; third-party inspection. The proposed methodology is effective to minimize the ambiguity of the requirements.

Diamantopoulos et al. [13] propose a mechanism for automating the reflection of requirements in formal representations by means of marking the semantic role. This mechanism can be used for reflection of the natural language functional requirements originally in the specifications, and then in the source code, but it is completely unsuitable for the non-functional requirements.

In [14], the approach for automatically extracting the semantic information from the SRS by combining methods of marking the semantic role and modelling the knowledge of the subject domain was proposed. Wang [14] chooses numerous verbs from the available specifications in the field of e-commerce and built a semantic framework for these verbs. Then the selected sentences were marked manually, and the result was used as a training sample for machine learning. The proposed approach is effective and reliable for modelling the variability and generality of functional requirements, but is not suitable for working with the non-functional requirements.

Riaz et al. [15], using methods of machine learning, have developed a technological process that takes natural language artefacts (for example, SRS) as input. This process automatically identifies relevant safety sentences in the artefacts and categorizes them according to safety objectives – either explicitly stated or predicted by sentences. The authors have developed context-sensitive templates that can be used to create a set of functional requirements of safety by filling out key information from relevant safety proposals.

A method for setting up and creating a combined parser for processing and analysis of the natural language specifications was developed [16]. This method combines the benefits of formal parsers (which are used to handle descriptions with a rigidly defined syntax – for example, the source code – and are very accurate and effective in processing the formal part) and of analysts (who are developed to handle natural language and who well understood the free text). The obtained text parser is based on Parsing Expression Grammars. The proposed approach [16] allows constructing an effective syntactic analyzer for the analysis of the natural language specifications of the requirements for industry software.

Gnesi et al. [17] developed the methodology and tool QuARS (Quality Analyzer for Requirements Specifications) for systematic and automatic analysis of the natural language requirements. QuARS provides the initial analysis of the requirements with the purpose of automatically identifying the potential linguistic defects that may cause interpretation problems at the next stages of software development. This tool can partly support the analysis of coherence and completeness by clustering the requirements according to specific topics.

Siegemung [18] presents an ontological approach to automated verification and measurement of software requirements, which is used for identification of inconsistencies, inconsequence and disadvantages of software requirements. For this, a meta-model of requirements – the ontology of requirements was developed, which provides a basis for validation and support tool, and also provides a conceptualization of knowledge about requirements. So, the ontology of requirements allows documenting the structured, multiple, unambiguous, traceable, complete, and consistent requirements in accordance with the standard IEEE 29148:2011 for the SRS. The developed ontology of requirements can be used by the requirements engineer as a tool for illustration of inconsistencies, incompleteness and qualitative disadvantages of requirements.

Farfeleder et al. [19] have represented the prototype of a semantic system, which is used for the provision of the help to requirements engineers regarding extracting the requirements with the use of a semi-formal representation. The semantic system uses the concepts, relations and axioms of the ontology of the subject domain for providing the list of proposals, on the basis of which the engineer can develop requirements. The developed semantic system effectively supports requirements engineers in determining well-structured requirements.

Murtazina and Avdeenko [20] propose an approach to support the process of requirements engineering in the process of developing the software by using ontology. In this paper, the ontology with taking into account the features of the Scrum methodology was presented. In analyzing the peculiarities of requirements engineering in the framework of the Scrum methodology, it has been established that the main efforts are focused on the analysis of the functional components, so the non-functional requirements are often not documented. Considering the high frequency of updating the requirements the proposed approach provides quick monitoring of traceability and completeness of requirements.

Mustafa et al. [21] proposes a structure for the integration of mixed requirements using ontologies. This approach is based on the general model of user requirements, which used the three existing semi-formal languages (UML, MCT and MERISE) and target-oriented languages. The method, developed in [21], focuses on minimizing the impact of the heterogeneity of requirements on the software quality.

In [4], Natural Language Processing Requirements Analysis Tools (for example, QVscribe) were developed – for automating and significantly speeding the task of searching for possible errors in natural language requirements, and for the provision of the visual scoring of each assessed requirement. But such a tool does not reveal information losses in the formation of requirements.

As a result, all considered methods of semantic analysis of the SRS do not provide validation of the accordance of the non-functional requirements to the needs of the customer. Therefore they do not aim at increasing the success of the software project. The lack of solving this task causes the *need for the development of the ontology-based intelligent agent for parsing the natural language SRS on the subject of the search of attributes for determining the non-functional characteristics-components of software quality*.

3. ONTOLOGY-BASED INTELLIGENT AGENT FOR PARSING THE SPECIFICATIONS OF SOFTWARE REQUIREMENTS

For automation of the semantic analysis of a natural language specification, the SRS must be formalized. Formalization of the SRS must be performed, taking into account the requirements, on the extraction of which the semantic parsing the SRS will be directed. For parsing the specifications for the search of attributes of the software quality components of non-functional characteristics, such formalization will be performed using ontologies. Ontologies provide identification of duplications and gaps in knowledge based on the visualization of missing logical connections, and the ability to analyze information by intelligent agents. For such formalization of the SRS, the development in [2, 7] as a base ontology of the software requirements specification (in terms of the presence of attributes) can be used. In this ontology, the attributes, which are necessary for determining the non-functional characteristics, are presented taking into account the distribution by sections of the specification. The developed ontology serves as a template of the SRS in terms of the presence of attributes and provides visual guidance to the user about the location of attributes in the specifications of software requirements.

The OBIA for semantic parsing the natural language SRS accepts the SRS as input data and automatically parses the SRS for the search of attributes, which are necessary for determining the non-functional characteristics of the software.

The method of activity of the OBIA for semantic parsing the natural language SRS consists of the following stages:

1) the search for each attribute from the base ontology of the SRS (this ontology is contained in the knowledge base of OBIA) in the specification for the actual software;

2) if <attribute$_i$> is found in the SRS, then <attribute$_i$> is the element of the set of available attributes, i=1..138 (because, according to ISO 25023:2016, there are 138 different attributes, on which the non-functional characteristics depend);

3) if <attribute$_i$> is not found in the SRS, then <attribute$_i$> is an element of the set of missing attributes, i=1..138;

4) all attributes, which are elements of the set of missing attributes, are removed from the base ontology for the non-functional characteristics-components of software quality, as developed in [2, 7];

5) the check whether all attributes from the set of available attributes are stored in the base ontology after modifying it in the previous step;

6) the save of the changes is made – the creation of the real ontology for the non-functional characteristics-components of software quality.

Consequently, the result of the work of such OBIA is the real ontology for the non-functional characteristics-components of software quality, which is the input data for the OBIA for evaluating the initial stages of the software lifecycle, as developed in [8]. As additional results of the OBIA's operation, the sets of available and missing attributes in the real SRS can also be used for further work.

The structure of the OBIA for semantic parsing the natural language SRS is presented in Fig. 1.

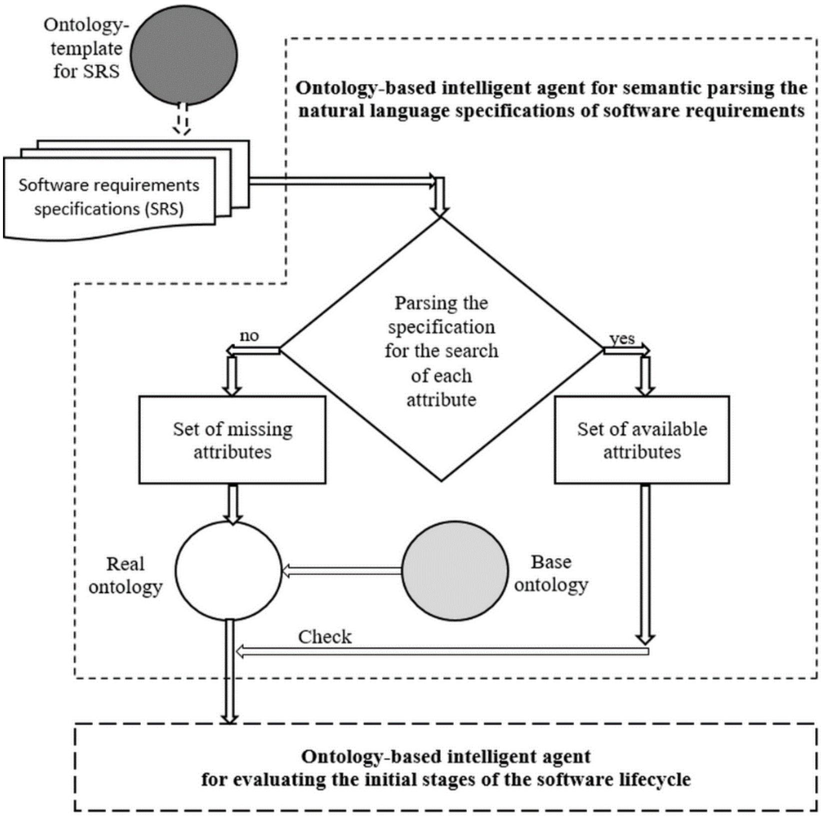

Fig. 1. Structure of OBIA for semantic parsing the natural language SRS

The developed OBIA provides the analysis of the natural language specifications on the subject of identification of the availability or absence (miss) of attributes, which are necessary for determining the non-functional characteristics. These results are then used to assess the sufficiency of information (attributes) for determining the non-functional characteristics. Since for determining the sufficiency, it is only necessary to know whether the attribute is present in the SRS or it is missing in the SRS, so the rules for parsing the SRS, on the basis of which the developed agent works, are simple. The simplicity of these rules provides a high speed of parsing the natural language SRS.

4. FUNCTIONING THE DEVELOPED ONTOLOGY-BASED INTELLIGENT AGENT

The OBIA for the semantic parsing the natural language SRS is implemented in the PHP language as the part of the free software, which is available on link – https://olp-project.herokuapp.com.

The user must upload the natural language SRS in PDF-format at the input of the developed OBIA. The OBIA parses this uploaded specification. The results of the OBIA's functioning are: the number of missing attributes (Fig. 2), the visualization of the missing attributes with the distribution by the sub-characteristics of the non-functional characteristics-components of quality, for assessment of which they are required (Fig. 3), and the real ontology for the non-functional characteristics-components of quality in OWL-format. Fig. 2 and Fig. 3 show the results of the functioning the OBIA for semantic parsing the natural language SRS for managerial software, which was developed for Ltd "Pobutradiotechnika" (Khmelnitsky).

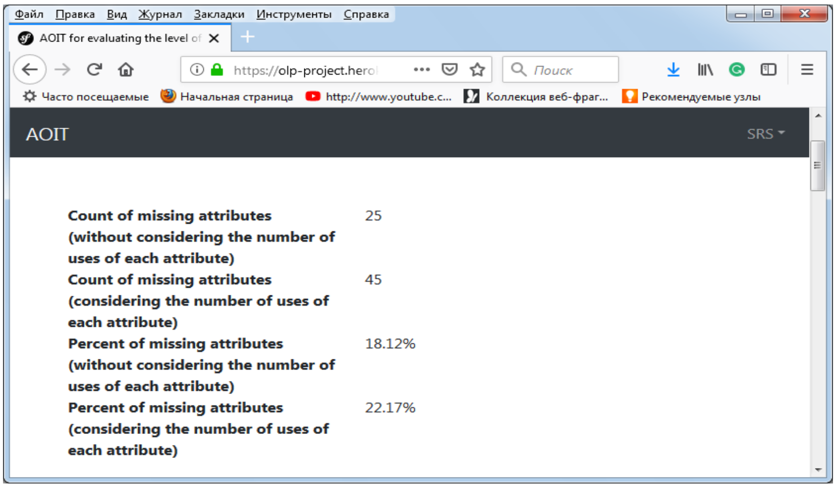

Fig. 2. Results of OBIA for semantic parsing the natural language SRS: quantity and percent of missing attributes

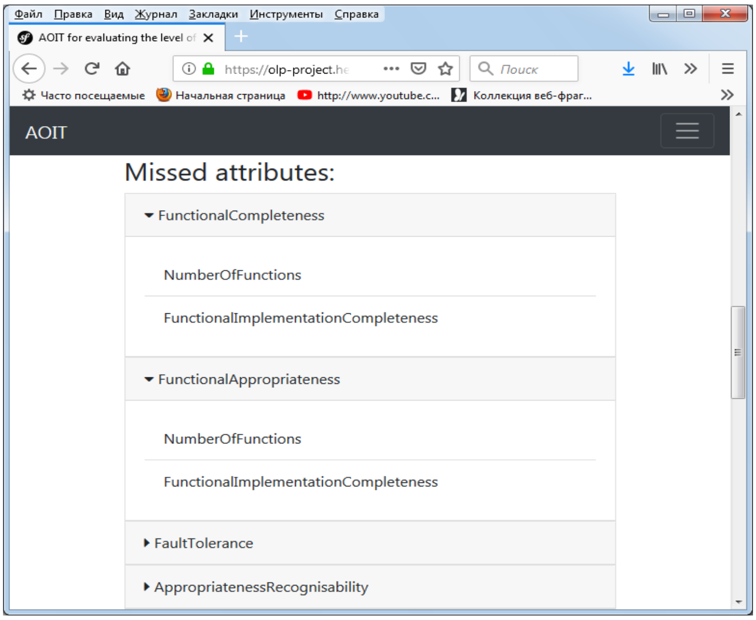

Fig. 3. Results of OBIA for semantic parsing the natural language SRS: missing attributes for Functional Completeness and Functional Appropriateness (sub-characteristics of Functional Suitability)

Fig. 2 shows, that the developed OBIA provides twice the number of missing attributes – Count of missing attributes (without considering the number of uses of each attribute) and Count of missing attributes (considering the number of uses of each attribute). Accordingly, there are two percentages of missing attributes too – Percent of missing attributes (without considering the number of uses of each attribute) and Percent of missing attributes (considering the number of uses of each attribute).

Such a double Counts and Percentages are due to the presence of a correlation of sub-characteristics and non-functional characteristics-components of software quality by attributes. There are attributes that depend on more than one sub-characteristic and one non-functional characteristic. So, according to ISO 25023:2016, the sub-characteristics of the non-functional characteristics depend on 203 attributes, but in [2] it was proved, that there are only 138 different attributes. The existence of such a correlation between non-functional characteristics and their sub-characteristics by attributes affects the significance and weight of the attributes. The more sub-characteristics and characteristics depend on a particular missing attribute, this attribute becomes more important, since more sub-characteristics and characteristics cannot be calculated reliably without it. In the future, it is recommended to the SRS developer to add attributes in the SRS, if it is necessary – with the purpose of ensuring, first, the presence of such attributes, from which more than one sub-characteristic and/or non-functional characteristic depend.

The OBIA result "Count of missing attributes (without considering the number of uses of each attribute)" represents the number of missing different attributes. The result "Percent of missing attributes (without considering the number of uses of each attribute)" is the percentage ratio of count of missing attributes (without considering the number of uses of each attribute) to 138 (number of different attributes). The OBIA result "Count of missing attributes (considering the number of uses of each attribute)" represents the number of missing attributes with repetitions of their uses, i.e. how many real gaps in knowledge exist in the calculation of non-functional characteristics-components of quality. The result "Percent of missing attributes (considering the number of uses of each attribute)" is the percentage ratio of count of missing attributes (considering the number of uses of each attribute) to 203 (the total number of attributes).

The presented results of OBIA for semantic parsing the natural language SRS can also be used to further their processing by the OBIA for evaluating the initial stages of the software lifecycle with the purpose of the calculation of the sufficiency of information about non-functional characteristics in the SRS. The OBIA results can be used as recommendations to the SRS developer (the requirements engineer) with the purpose of ensuring the availability in the SRS as many attributes as possible, which are necessary for determining the non-functional characteristics-components of the software quality.

5. CONCLUSIONS

Since the success of a software project implementation is significantly dependent on the SRS, the problem of the analysis of the SRS (especially its requirements, which are defining the non-functional characteristics-components of software quality) is actual with the purpose of increasing the success of the software, which will developed on basis of the SRS. The analysis of known methods and tools of semantic parsing the SRS has shown, that they do not provide validation of compliance of non-functional requirements of the specification to the needs of the customer, do not aim at improving the quality of the software, which is developed on basis of the SRS. Therefore, today the unsolved task is automation of semantic parsing of the natural language SRS for establishing the compliance of the non-functional requirements to the needs of the customer.

In this paper the OBIA for semantic parsing the natural language SRS is developed. It allows to provide parsing the specification; determine the count and percentage of missing attributes; visualize attributes, which are missing for one or another sub-characteristic of the non-functional characteristics; and establish the real ontology for non-functional characteristics that can be used by previously developed OBIA. This helps us to evaluate the initial stages of the software lifecycle with the purpose of assessing the sufficiency of information for determining the non-functional characteristics (for validation of compliance of the SRS non-functional requirements to the customer’s needs).

In addition, the developed approach provides automating and speeding up of parsing the SRS; indicating the "bottlenecks" places (knowledge gaps) of SRS and demonstrating which requirements need re-work for providing the further assessment of the non-functional characteristics-components of software quality; training for SRS developers and requirement engineers (they can see their mistakes and gaps in requirements); and free online access, at any time, without any registration. The limitation of the developed OBIA is the search of attributes only, which are defined by ISO 25023:2016 as necessary for the assessment of non-functional characteristics-components of software quality.

REFERENCES

[1] Standish Group 2015 Chaos Report – Q&A with Jennifer Lynch, http://www.infoq.com/articles/standish-chaos-2015 (current April 2019).

[2] Hovorushchenko, T., Pomorova, O. Ontological Approach to the Assessment of Information Sufficiency for Software Quality Determination. *CEUR-WS*, 1614, 2016, pp. 332-348. Available at: http://ceur-ws.org/Vol-1614/paper_94.pdf.

[3] McConnell, S. *Code Complete*, Microsoft Press, Redmond, USA, 2004.

[4] Leveraging Natural Language Processing in Requirements Analysis: How to eliminate over half of all design errors before they occur, http://qracorp.com/wp-content/uploads/2017/03/Leveraging-NLP-in-Requirements-Analysis.pdf (current April 2019).

[5] Ishimatsu, T., Levenson, N., Thomas, J., Fleming, C., Katahira, M., Miyamoto, Y., Ujiie, R. Hazard analysis of complex spacecraft using systems-theoretic process analysis. *Journal of Spacecraft and Rockets*, 51, 2014, pp. 509-522.

[6] Levenson, N.G. Software challenges in achieving space safety. *Journal of the British Interplanetary Society*, 62, 2009, pp. 265-272.

[7] Hovorushchenko, T. Methodology of Evaluating the Sufficiency of Information for Software Quality Assessment According to ISO 25010. *Journal of Information and Organizational Sciences*, 42 (1), 2018, pp. 63-85. Available at: https://jios.foi.hr/index.php/jios/article/view/1113/794.

[8] Hovorushchenko, T., Pavlova, O. Evaluating the Software Requirements Specifications Using Ontology-Based Intelligent Agent. *Proceedings of the 2018 IEEE International Scientific and Technical Conference on Computer Science and Information Technologies*, Ukraine, September 2018, vol. 1, pp. 215-218.

[9] Gulia, S., Choudhury, T. An Efficient Automated Design to Generate UML Diagram From Natural Language Specifications. *Proceedings of the 6th International Conference on Cloud System and Big Data Engineering*, India, January 2016, pp. 641-648.

[10] Selway, M., Grossman, G., Mayer, W., Stumptner, M. Formalising natural language specifications using a cognitive linguistic/configuration based approach. *Information Systems*, 54, 2015, pp. 191-208.

[11] About the Semantics Of Business Vocabulary and Rules Specification Version 1.4, https://www.omg.org/spec/SBVR/About-SBVR/ (current April 2019).

[12] Ali, S. W., Ahmed, Q. A., Shafi, I. Process to Enhance the Quality of Software Requirement Specification Document. *Proceedings of the International Conference on Engineering and Emerging Technologies,* Pakistan, February 2018, pp. 1-7.

[13] Diamantopoulos, T., Roth, M., Symeonidis, A., Klein, E. Software requirements as an application domain for natural language processing. *Language Resources and Evaluation*, 51 (2), 2017, pp. 495-524.

[14] Wang, Y. Semantic Information Extraction for Software Requirements using Semantic Role Labeling. *Proceedings of the IEEE International Conference on Progress in Informatics and Computing*, China, December 2015, pp. 332-337.

[15] Riaz, M., King, J., Slankas, J., Williams, L. Hidden in Plain Sight: Automatically Identifying Security Requirements from Natural Language Artifacts. *Proceedings of the 2nd IEEE International Requirements Engineering Conference*, Sweden, August 2014, pp. 183-192.

[16] Iwama, F., Nakamura, T., Takeuchi, H. Constructing Parser for Industrial Software Specifications Containing Formal and Natural Language Description. *Proceedings of the 34th International Conference on Software Engineering*, Switzerland, June 2012, pp. 1012-1021.

[17] Gnesi, S., Lami, G., Trentanni, G., Fabbrini, F., Fusani, M. An automatic tool for the analysis of natural language requirements. *Computer Systems Science and Engineering*, 20 (1), 2005, pp. 53-62.

[18] Siegemund, K. *Contributions to Ontology-Driven Requirements Engineering*, Doctoral dissertation, Technische Universität Dresden, Dresden, 2014.

[19] Farfeleder, S., Moser, T., Krall, A., Stalhane, T., Omoroniya, I., Zojer, H. Ontology-Driven Guidance for Requirements Elicitation. *Lecture Notes in Computer Science*, 6644, 2011, pp. 212-226.

[20] Murtazina, M. Sh., Avdeenko, T. V. The ontology-driven approach to support the requirements engineering process in Scrum framework. *CEUR-WS*, 2212, 2018, pp. 287-295. Available at: http://ceur-ws.org/Vol-2212/paper39.pdf.

[21] Mustafa, A., Wan-Kadir, W. M. N., Ibrahim, N., Shah, A., Younas, M. Integration of Heterogeneous Requirements using Ontologies. *International Journal of Advanced Computer Science and Applications*, 9 (5), 2018, pp. 213-218.

Information about the authors:

Tetiana Hovorushchenko – Full Doctor, Full Professor, Head of Computer Engineering and System Programming Department, Khmelnitsky National University. The scientific activities currently focus on the development of intelligent agent-based information technology for evaluating the sufficiency of information in the software requirements specifications.

Artem Boyarchuk – associate prof. at the computer systems, networks and cybersecurity department of the National aerospace university. His expertise is in models and methods for availability assessment of service-oriented infrastructures.

Olga Pavlova – PhD student of Computer Engineering and System Programming Department, Khmelnitsky National University. The research interests include intelligent agent-based information technology, ontology-based intelligent agents.

Manuscript received on 12 April 2019