Prospects of Using the Ontology Description Language ONTOL V2 in the Automation of the Educational Process

Danil Pestryakov

Abstract. This paper discusses the prospects of using text language to describe ONTOL V2 ontologies in the automation of the educational process in engineering specialties. ONTOL V2 is a modification of the ONTOL V1 language that allows you to describe UML class diagrams, as well as software implementations of concepts and theorems of specific domain areas in a single style. The paper also analyzes existing solutions for visualizing ontologies represented by UML diagrams based on a text description. Examples of approaches to automating the intermediate and final assessment procedures are also provided. A distinctive feature of the ONTOL V2 language is the transition from modeling to metamodeling.

Introduction

The forms of representation of ontologies in a computer have overcome a long evolutionary path. Semantic networks are considered one of the first such forms. However, to solve applied problems, specialized representations of ontologies were required. One of these tasks is to build a semantic web. To solve this problem, the World Wide Web Consortium develops its own standards for describing ontologies: RDF, RDFS, OWL. The semantic web was supposed to make the information on the Internet suitable for automatic analysis, which could serve as a driver to improve the validity of responses to user search queries. However, this initiative faced several obstacles and was not fully implemented.

Despite the difficulties of using ontologies for structuring information on the Web, they have found their place for structuring information in the educational process. Thus, ontologies described using the UML graphical notation are an integral part of the course 'Discrete Mathematics' at Peter the Great St. Petersburg Polytechnic University [1, 2]. In addition, to automate the implementation of intermediate assessment for this course, a tool was developed to automatically check the correctness of formula transformations of mathematical expressions [3, 4]. The formulas required for each section are recorded in a separate system. However, it is obvious that formulas are tied to specific concepts. Thus, a single format for textual description of relations between concepts is necessary, as well as related formulas and theorems.

In March 2025, a group of initiative students developed the text language to describe ontologies ONTOL V1 [5], which follows the concept of faceted ontologies presented in [1]. A translator from ONTOL V1 to PlantUML was implemented for automatic diagram visualization. However, the ONTOL V1 language has several drawbacks that the ONTOL V2 language was designed to address.

1 Description of the ONTOL V2 language structure

The transition from modeling to metamodeling [6] is a key feature of the ONTOL V2 language in comparison with ONTOL V1. The pilot operation of ONTOL V1 revealed that a language with a fixed metamodel narrows the space of subject areas that can be described using this language. The modeling process itself also becomes more complex, which reduces the number of potential users. Switching to the description of metamodels allows you to more accurately describe each subject area with the degree of detail that is necessary. It also allows students to read model descriptions in a language they understand, leaving all the difficulties at the metamodel level.

The introduction of the metamodel description level motivated us to divide the description of specific domains into three files: ontolmy for metamodels; ontol for models; sofy for describing ontology visualization styles. This approach has a significant advantage: it is assumed that metamodel descriptions will change many times less often than model descriptions. Thus, following the approach from [7], ontolmy can be loaded into the computer's memory in advance and reused whenever the model needs to be recompiled. YAML is offered as a text format for describing metamodels, models, and styles.

2 Visualization of knowledge using the graphical language UML

There are many tools for building UML diagrams: desktop and web. Desktop tools are difficult to distribute and often have a complex diagram serialization structure, so we do not consider them in this paper.

One of the most popular web tools for building UML diagrams using text descriptions is PlantUML. However, its significant drawback is the compilation of the finished image. This means that you can't manually edit chart elements.

Developers of the d2 diagram visualization language [8] provide a WEB IDE to create diagrams with bidirectional actions: changes in the text change the diagram and vice versa, manual editing of the diagram changes the text. However, this tool is also not suitable for use in an educational environment, as the WEB IDE is available for a paid subscription. Also, the language itself contains errors in describing UML diagrams: for example, you can specify multiplicity for the generalization relation.

Thus, examples of existing tools for visualizing diagrams based on a text description are provided. The existence of these tools proves that it is possible to implement the chart visualization tool described in ONTOL V2.

3 Checking the correctness of proofs of theorems

A theorem is an entity that consists of an input expression, an output expression, and initial statements (axioms) written in the same formal language. We assume that the rules of logical inference are contained on the side of the tool that checks the correctness of proofs of theorems. There are various methods of automatic proof of theorems. Let's take a look at some examples.

The paper [9] is devoted to a tool that allows generating logical formulas based on a simple semi-formal description of the problem. The solution of the original problem is reduced to solving the SAT problem. In [3, 4], a tool is described to automatically check the correctness of formula transformations in mathematical expressions. The solution is considered correct if the entire chain of transitions from the original expression to the final one is correct. A separate transition is valid if there is a transformation in the system that translates an expression on the left into an expression on the right.

Thus, examples of existing tools for automatic verification of the correctness of proofs of theorems are given. Obviously, it doesn't make sense to link to a specific one. Therefore, we need a unified way to write theorems that is built into the ontology description language.

It is worth noting that the above approaches can be applied when creating a system to check algorithmic knowledge. The algorithm here should be perceived as a sequence of steps to transform one concept of the specific domain into another. Each algorithm consists of steps. Each step can also be represented by an algorithm. Then the axioms are statements about the structure of input and output data after each step of the algorithm is executed. The system can be arranged as follows: the student is given an algorithm written in some formal language, in the form of a sequence of steps. It is suggested to describe the sub-steps of each step. Having statements about the structure of input and output data after each step in the system, you can check the correctness of the algorithm description by steps.

Conclusion

The article discusses the following areas of automation in the educational process: the automation of knowledge visualization and the procedures for intermediate and final assessment. The proof of solvability of both problems separately is given. The ONTOL V2 ontology description language has been developed, which allows you to describe ontologies in such a way that they can be automatically visualized, as well as used to describe the limitations of specific domains expressed in the form of theorems. ONTOL V2 is designed with the current context in mind and allows you to describe ontologies at the metamodel level. Thus, each researcher has the opportunity to work with more complex concepts and describe specific domains more accurately. The student can also read ontologies in a language that they understand, as the connection with UML is encapsulated at the metamodel level.

References

[1] Novikov F. A., Molotkov I. I. Ontology of Discrete Mathematics in Education // Computer Tools in Education. 2021. No. 1. P. 5–16. (in Russian)

[2] Novikov F. A. Discrete Mathematics: Textbook for Universities. 2nd ed. Third Generation Standard. — St. Petersburg: Peter. 2013. 432 p.: illus. (in Russian)

[3] Novikov, F., Katsman, V., 2018. Gamification of problem solving process based on logical rules. In Informatics in Schools. Fundamentals of Computer Science and Software Engineering, Pozdniakov, S. N. & Dagienė, V. (eds). Scopus, Springer International Publishing, 369–380.

[4] Novikov, F., Katsman, V., Mosin, V., 2020. Automated verification of expression transformation chains based on computational experiments. IOP Conference Series: Materials Science and Engineering. Scopus, 012132

[5] M. I. Aptukov, M. D. Markov, F. A. Novikov, D. D. Pestryakov, V. S. Skvortsov, I. I. Khamidullin, Ontology description language ONTOL V1 // Computer Tools in Education. 2025. No. 1. P. 91–107. (in Russian)

[6] Novikov F.A., Ivanov D.Yu. Modeling in UML. Theory, Practice, Video Course. — St. Petersburg, Professi onal Literature, Science and Technology, 2010

[7] Terekhov, A. N. "Library introductions and separate translation of procedures in a translator from Algol 68." Abstracts of reports and messages for the All-Union conference. Part. Vol. 2. 1980. (in Russian)

[8] D2 repository [Online]. Available: `https://github.com/terrastruct/d2` (accessed: 05.07.2025)

[9] O. Gasquet, D. Longin, E. Lorini, F. Maris, P. R´egnier, and S. Soloviev, TouIST, a Teacher- and Student-Friendly Language for Propositional Logic and Discrete Mathematics // Computer Tools in Education. 2021. No. 2. P. 13–25.

Danil Pestryakov
Institute of Physics and Mechanics
Peter the Great St. Petersburg Polytechnic University
Saint Petersburg, Russia
e-mail: `pestryakovdanil@mail.ru`