Computer Science and Interdisciplinary Research Journal
Volume 1, Issue 1, ISSN: 3033-1218, DOI: 10.70862/CSIR.2024.0101-05, 2024

Using Ontologies in e-Learning in Software Engineering Course

Asya Stoyanova-Doycheva[1], Georgi Cholakov[2], Todorka Glushkova[3]

University of Plovdiv Paisii Hilendarski, 236, Bulgaria Blvd, 4027 Plovdiv, Bulgaria

[1] astoyanova@uni-plovdiv.bg, [2] gcholakov@uni-plovdiv.bg, [3] glushkova@uni-plovdiv.bg

Abstract Ontologies enhance educational processes by structuring knowledge and supporting personalized learning. This study investigates their use in e-learning for generating educational content and test questions within software engineering. We discuss two ontologies: one for educational content creation and the other for test question generation. By integrating these ontologies, we improve the adaptability and effectiveness of both educational resources and assessments, offering a more comprehensive learning experience.

Keywords – ontology, software engineering, integration ontologies.

1. Introduction

Ontologies are formal, explicit specifications of a shared conceptualization [1]. They provide a structured framework for representing knowledge about a particular domain by defining a set of concepts and the relationships between them. Ontologies play a pivotal role in enhancing learning processes by offering structured, semantic frameworks for organizing and integrating knowledge. This approach enables systems to understand and interpret information in a consistent and meaningful way, allowing for better data integration, sharing, and reasoning. They are crucial for semantic web technologies, where they help in structuring and connecting data to improve its usability and accessibility.

In educational contexts, ontologies enable precise management of educational content, improving the accessibility and relevance of resources through well-defined relationships between concepts. This structured approach supports personalized learning by tailoring educational materials to individual needs and preferences. Ontologies facilitate semantic search, which enhances the accuracy of content retrieval and recommendations. Furthermore, they underpin adaptive learning systems by allowing real-time adjustments based on learner progress and feedback. Ontologies can also enhance test systems by enabling the automatic generation of test questions and assessments, leveraging structured knowledge to create contextually relevant and personalized evaluations.

The integration of ontologies into educational systems has become increasingly significant, particularly in enhancing the adaptability and personalization of learning processes. Stancin et al. [2] provide a comprehensive overview of how ontologies have been applied in education, emphasizing their role in structuring educational content and supporting personalized learning paths. Similarly, Tapia-Leon et al. [3] explore the systematic application of ontologies in higher education, demonstrating their impact on curriculum development and content delivery. In the context of software engineering education, Wille et al. [4] discuss the practical application of ontologies in creating e-learning infrastructures, which have been pivotal in the development of adaptive and context-aware educational resources.

The emergence of Large Language Models (LLMs) has introduced new dimensions to ontology research and application. Mukanova et al. [5] and Joachimiak et al. [6] highlight how LLMs can be leveraged to enrich and refine ontologies, enhancing

DOI: https://doi.org/10.70862/CSIR.2024.0101-05

Corresponding author: Asya Stoyanova-Doycheva, University of Plovdiv Paisii Hilendarski
Email: astoyanova@uni-plovdiv.bg

Received: 14 August 2024.
Revised: 29 August 2024.
Accepted: 30 August 2024.
Published: 1 September 2024.

© 2024 Asya Stoyanova-Doycheva, Georgi Cholakov & Todorka Glushkova. This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

The article is published with Open Access at http://csirjournal.com/

their semantic depth and applicability in complex educational settings. These advancements underscore the evolving landscape of ontology-driven education, where LLMs play an increasingly critical role in the development and deployment of intelligent educational systems.

In education, ontologies support personalized learning by modeling educational content and learner profiles, enabling adaptive learning paths and automated content generation tailored to individual needs and preferences [7]. This personalization addresses diverse learning styles and paces, ultimately improving student engagement and outcomes [7]. Ontologies also streamline the creation of educational materials and assessments by providing a structured representation of knowledge, ensuring content relevance and alignment with learning objectives. This automation reduces the workload for educators and allows for consistent and scalable educational resources. Sapna and Balakrishnan [8] discuss the use of ontologies to support the teaching of software engineering, highlighting their role in improving educational outcomes in this field. Altan [9] further emphasizes the importance of ontologies in software engineering education programs, noting their contribution to structuring complex educational content and enhancing the learning experience.

This paper contributes to this growing body of research by focusing on the development and integration of ontologies within a Software Engineering course, specifically within the Virtual Education Space [10]. These ontologies are concentrated on two main areas: the automatic generation of test questions and the presentation of educational content within the "Software Engineering" (SE) discipline. In the final part of the article, we will demonstrate how the two ontologies, structured differently to meet the needs of two distinct environments – one for content generation in the SE discipline and the other developed for test question generation – can be integrated and used effectively in both contexts.

2. Ontology Development in Software Engineering Course.

This ontology describes the key concepts within in the lecture course on "Software Engineering" at the Faculty of Mathematics and Informatics of the University of Plovdiv [11]. The ontology was developed in Protégé-OWL. Concepts and notions are represented by means of classes. Also described are the relationships between the classes and their properties. Figure 1 shows a graph of some of the classes implemented in the ontology.

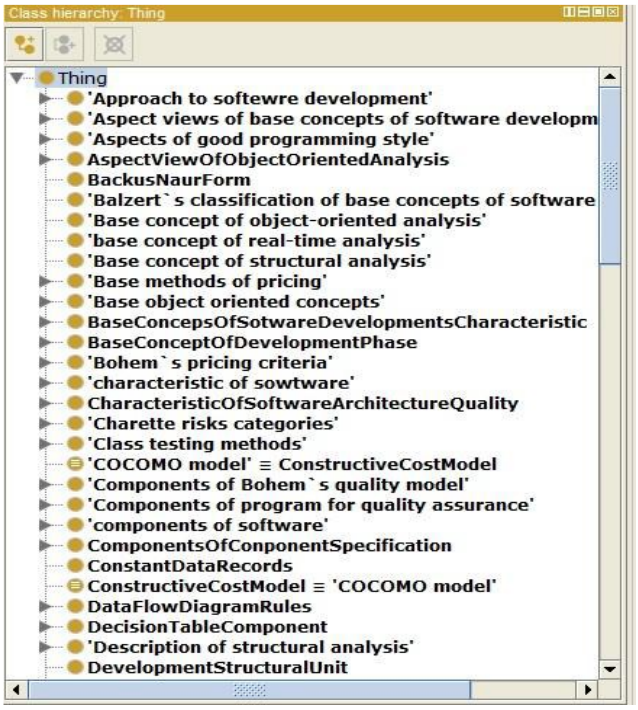


Figure 1. Hierarchy of classes in SE ontology

To be able to filter the data by certain criteria, additional information (meta-knowledge) have been added to the classes, entities, as well as constraints and properties. This have been implemented by creating two new annotations in Protégé-OWL: *importanceLevel* and *lectureNumber*. Figure 2 shows an example of the added annotation.

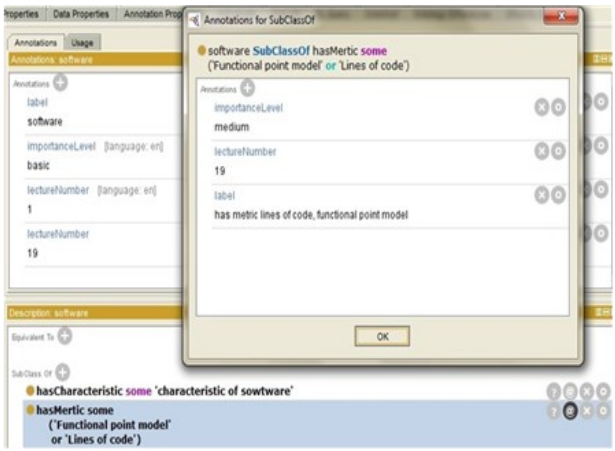


Figure 2. Annotation properties in SE ontology

The *importanceLevel* annotation takes values basic, medium and high. It predefines the importance level for each class, individuality and constraint to be displayed by the application. The second annotation added is *lectureNumber*. It represents information about the lecture number from which the concept is.

An intelligent agent processes the ontology, which aims to generate the content of the course material according to the level of the student and the relevant topics that the respective teacher would like to include in the course of study. Frequently the same subject is taught to students with different profiles. The environment for generating learning content from ontologies enables to select appropriate course topics for the corresponding level of students. Figure 3 shows a snapshot of the interface of the developed environment.

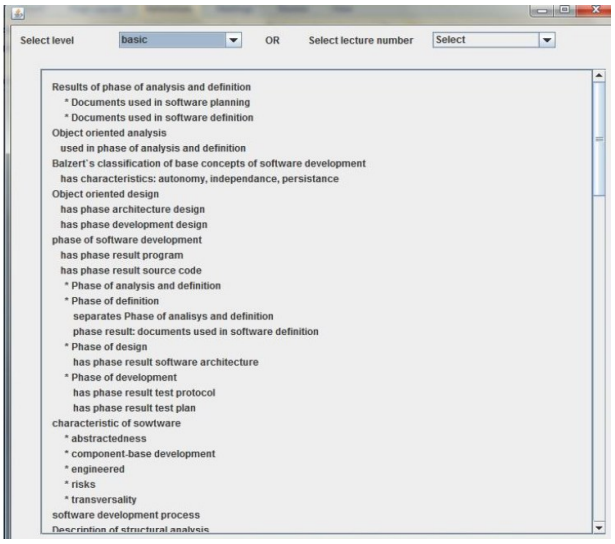


Figure 3. Generated educational content from SE ontology

The ontology's content can be updated - new classes and axioms can be added as the course is extended, or the weight of each of the topics and concepts in the ontology can be changed as decided by the instructor.

3. UML Ontology for Test Question Generation

This part of the article illustrates the structure of an ontology for test question generation. UML Ontology was developed for the syntax of Unified Modelling Language 2.4.1 using Protégé-OWL v4.3. The UML Ontology comprises basic classes that represent the fundamental concepts in UML (Fig. 4). These classes were selected to encompass the language specification and the topics covered in the Software Engineering course at the Faculty of Mathematics and Informatics. The concepts are organized in a class hierarchy, linked through object properties or data properties, forming axioms that represent the UML language syntax. This method stores information both symbolically and semantically, allowing it to be used not just as text for processing and rendering, but as interconnected concepts. This meaningful relationship between concepts enables the derivation of conclusions about the truth of statements in the field.

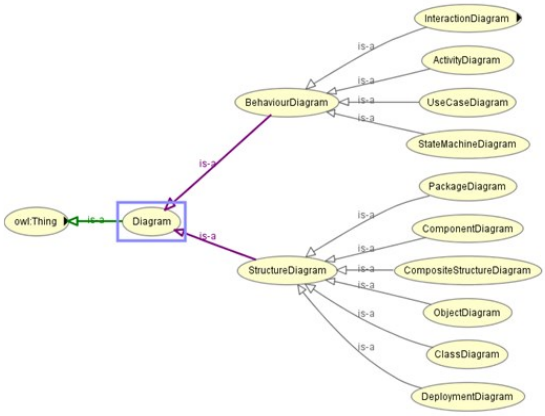


Figure 4. Hierarchy of classes in UML ontology

The class names match the construct names specified in UML 2.4.1. Some constructs are subclasses of multiple parent classes, as per the UML specification, certain elements belong to several packages, and this is accurately reflected in the ontology. The UML ontology includes two main types of properties to represent relationships between different classes: "object properties," which connect classes with other classes, and "data properties," which link classes with specific values. In Protégé-OWL, an "object property" is a property of one class that connects it to another class or links an instance of a class (individual) with another instance. Similar to classes, properties can be arranged in a hierarchy. All object properties in the Protégé-OWL hierarchy fall under topObjectProperty. Figure 5 illustrates a Protégé view of the object properties in the UML ontology.

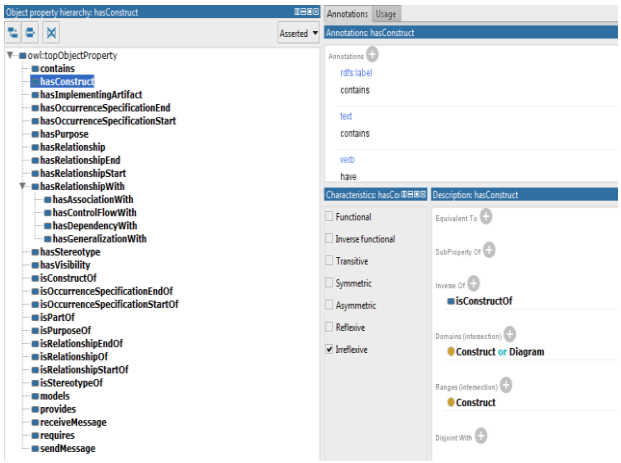


Figure 5. Object properties in UML ontology

In the ontology, classes are categorized into two types based on their conditions: primitive and defined.

Most classes in the ontology are primitive, meaning they only have necessary conditions. An individual of such a class must meet these conditions, but possessing these properties alone does not classify it as an individual of that class. Defined classes, on the other hand, have both necessary and sufficient conditions. If an individual meets these conditions, it can be conclusively classified as an individual of that specific type. In Protégé and OWL, necessary and sufficient conditions are referred to as equivalent. The defined classes in the UML ontology include Device, ExclusionEnvironment, BidirectionalAssociation, UnidirectionalAssociation, Deploy, Import, Extend, Include, Merge, Realization, Usage, CompleteMessage, FoundMessage, and LostMessage. Figure 6 illustrates the definition of the Deploy class. All other classes are primitive. To the right, the Usage view shows the constraints involving the class, with the class description below it. The first section, EquivalentTo, lists the necessary and sufficient conditions. The SubClassOf section contains the remaining necessary restrictions, and below that, SubClassOf (Anonymous Ancestor) displays the inherited restrictions from parent classes.

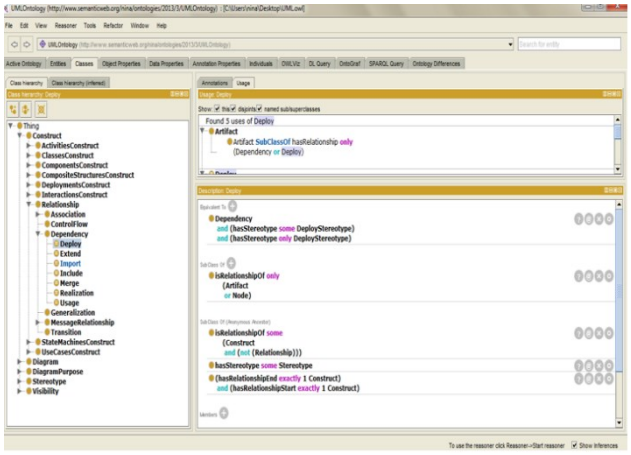


Figure 6. Constraints for class *Deploy*

Meta knowledge have been integrated into the UML ontology, significantly aiding in the generation of tests and test questions through the ontology. This type of knowledge, which provides additional information about individual elements, is known as knowledge about knowledge. In OWL, meta-knowledge is represented by annotations, which can be of various types as defined by the AnnotationProperty. The language includes several predefined annotation properties, but also allows for user-defined ones. Each ontology element can be annotated and further explained with meta-knowledge.

For the UML ontology, labels and comments are used extensively. An annotation (rdfs:comment) is added to each class, providing a textual description of the concept's essence according to the UML specification. The label (rdfs:label) provides information on how to present the ontology element to the user, eliminating the need to visualize class names directly or process them through the system. Instead, labels are used for this purpose. Each class and property in the ontology is assigned a label with a string value that meaningfully names the element in English. Since classes represent concepts, their labels are typically nouns, while properties represent relationships between classes and are best labeled with verbs (Fig. 7).

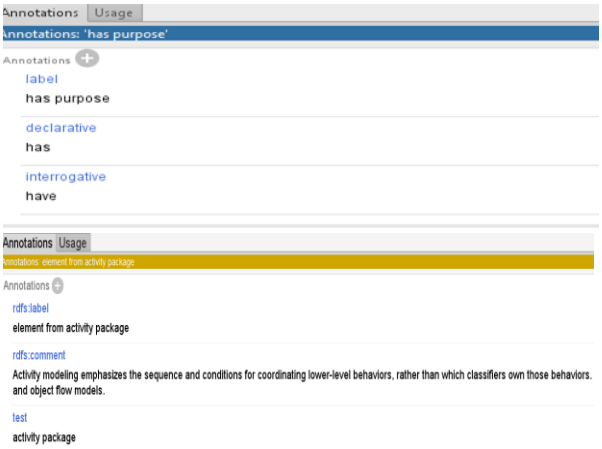


Figure 7. Annotation properties

For the current approach to automatically generate test questions, two additional annotation properties have been defined in the UML ontology: *declarative* and *interrogative* annotations. These annotations are related to the property and are intended to extend and complement their labels based on the use of the property. The values of these annotations provide the appropriate words needed for constructing different types of sentences. Essentially, these two annotations are specialized types of labels. In English, verbs change their form depending on the type of sentence being formed. Therefore, the declarative and interrogative annotations provide the label of a property for use in corresponding declarative and interrogative sentences.

The automatic question generation environment is a multi-agent system with two operative assistants and a graphical interface [12]. These assistants process the ontology to generate and validate questions. Each question is derived from an axiom, with extracted elements including a base class and its restrictions. If an axiom has multiple restrictions, one is chosen for simplicity. Properties and defining classes are also extracted. If a property is absent, the base class is a subclass of the defining class, using "is" in sentences. Otherwise, annotations are used to form declarative or

interrogative sentences. The process includes extracting annotated labels and placing them in templates for sentences. Different question types, such as open questions, follow this algorithm. The system generates questions by selecting axioms and using annotations of the base class, properties, and defining classes [13].

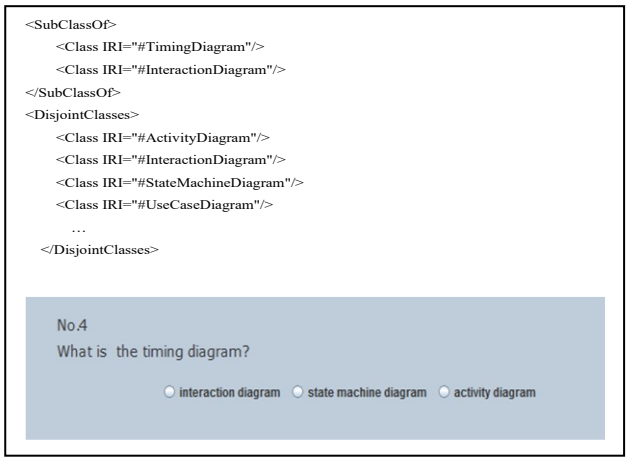


Figure 8. UML ontology axiom and generated test question from it

Example about generated test question and axiom from ontology can be seen on Figure 8.

4. Integration of Ontologies in Software Engineering.

One of the topics our team is working on is the integration of the ontologies we have already created to enrich the knowledge in software engineering and the possibility to have more capable environments for test question generation and learning content generation. To offer richer learning materials and a greater number of questions.

As presented earlier in the paper for the needs of the two environments (to generate learning content and to generate test questions), ontologies have specific elements. These specific elements are the specific annotation properties that are needed when generating e-learning content and test questions. When generating e-content from the Software Engineering Ontology, the corresponding annotation properties importanceLevel and lectureNumber need to be added to each class, and on the other hand, for the needs of the test question generation application, the corresponding annotation properties label, interrogative and declarative need to be added to the classes and properties.

Beyond altering the ontology structures to make them usable by both environments, their integration is important. For this purpose, a mapping was made between the corresponding classes and properties from the two ontologies. This may involve creating new classes and properties if necessary to reflect the combination of the two ontologies. The second important activity is the alignment of terminology. Merge differences in terminology by defining standardized terms and concepts. This is extremely important to avoid ambiguities and double interpretations. When examining the concepts in the ontologies in detail, it was concluded that the most appropriate concepts by which the two ontologies can be integrated are the different views by which an application can be represented - for this purpose, different basic concepts are used, where UML diagrams are also included. The concepts for the different core concepts exist in both ontologies (Fig. 9).

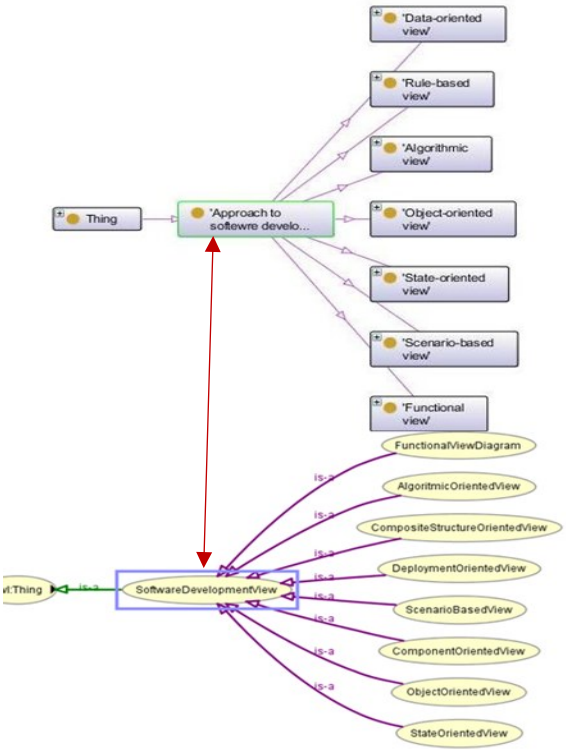


Figure 9. Common concepts in both ontologies

Each SoftwareDevelopmentView in turn contains the corresponding diagrams. For example, a class of diagrams, object diagrams, is used to model the object-oriented view in the UML ontology (Fig. 10). This knowledge is also represented in the Software Engineering Ontology. In the UML ontology these diagrams are described with the full syntax, which complements the knowledge of these diagrams with the basic notational elements in the UML syntax. On the other hand, to these two views in the Software Engineering ontology, there is knowledge about other basic concepts, but they are not part of the UML language.

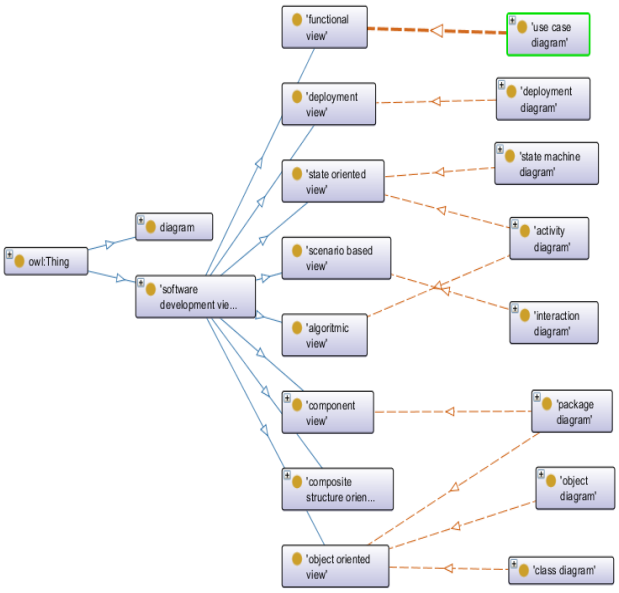


Figure 10. UML diagrams in different software views

As a final step, the classes had to be renamed to make them unique for the two ontologies. The newly created ontology basically adopted the class names from the UMLOntology because we decided to stick with the UML standard.

5. Conclusion

Integrating ontologies into an interconnected network can greatly enhance educational assessments. By combining ontologies from different disciplines such as history, science, literature, and mathematics, educators can create interdisciplinary test questions that encourage critical thinking and deeper understanding. Including ontologies related to real-world contexts, like cultural or historical events, enables the development of situational questions that challenge learners to apply their knowledge practically. Incorporating linguistic ontologies allows for assessments that test comprehension and translation across multiple languages, while integrating emotional intelligence and social behavior ontologies helps in evaluating learners' understanding of emotional responses and ethical decision-making. Expanding ontologies to cover STEM fields and technical skills can lead to complex problem-solving questions that simulate real-world challenges. Adding career-oriented ontologies can generate assessments that test both theoretical knowledge and practical skills relevant to specific professions. Incorporating environmental science and sustainability ontologies fosters questions that promote eco-conscious learning. Including cultural heritage and anthropology, ontologies enable assessments that explore cultural diversity and historical developments. Finally, integrating health, nutrition, and wellness ontologies supports the development of questions that enhance health literacy and understanding of wellness practices.

Acknowledgements

This study is financed by the project FP23-FMI-002 “Intelligent software tools and applications in research in mathematics, informatics, and teaching pedagogy” at the Plovdiv University “Paisii Hilendarski”.

References

[1] T. R. Gruber, "Toward Principles for the Design of Ontologies Used for Knowledge Sharing", International Journal of Human-Computer Studies, vol. 43, pp. 907-928, 1995. https://doi.org/10.1006/ijhc.1995.1081

[2] K. Stancin, P. Poscic and D. Jaksic, "Ontologies in education-state of the art", Education and Information Technologies, vol. 25(6), pp. 5301-5320, 2020. https://doi.org/10.1007/s10639-020-10226-z

[3] M. Tapia-Leon, A. C. Rivera, J. Chicaiza and S. Luján-Mora, "Application of ontologies in higher education: A systematic mapping study", In 2018 IEEE Global Engineering Education Conference (EDUCON), pp. 1344-1353, 2018. https://doi.org/10.1109/EDUCON.2018.8363385

[4] C. Wille, R. R. Dumke, A. Abran and J. M. Desharnais, "E-learning infrastructure for software engineering education: Steps in ontology modeling for SWEBOK", In IASTED Conf. on Software Engineering, pp. 520-525, 2004.

[5] A. Mukanova, M. Milosz, A. Dauletkaliyeva, A. Nazyrova, G. Yelibayeva, D. Kuzin and L. Kussepova, "LLM-Powered Natural Language Text Processing for Ontology Enrichment". Applied Sciences, vol. 14(13), 2024. https://doi.org/10.3390/app14135860

[6] M. P. Joachimiak, M. A. Miller, J. H. Caufield, R. Ly, N. L. Harris et al. "The Artificial Intelligence Ontology: LLM-assisted construction of AI concept hierarchies". arXiv preprint arXiv:2404.03044, 2024.

[7] Y. Wang and Y. Wang, "A Survey of Ontologies and Their Applications in e-Learning Environments". Journal of Web Engineering, vol. 20(6), pp. 1675-1720, 2021. https://doi.org/10.13052/jwe1540-9589.2061

[8] P. G. Sapna and A. Balakrishnan, "Using ontologies to aid the teaching of software engineering". In 2009 22nd Conference on Software Engineering Education and Training, pp. 224-231, 2009. https://doi.org/10.1109/CSEET.2009.32

[9] Z. Altan, "The importance of ontology in software engineering education programs". The Online Journal of Science and Technology, vol. 7(1), 119, 2017.

[10] D. Orozova, S. Stoyanov and I. Popchev, "Virtual Education Space", in International Conference, Free University of Burgas, Burgas, Bulgaria, 2013.

[11] S. Stoqyanov, A. Stoyanova - Doycheva, M. Trendafilova and E. Doychev, Software Engineering, University publishing house "Paisii Hilendarski" 2006.

[12] N. Stancheva, A. Stoyanova-Doycheva, S. Stoyanov and I. Popchev, "Automatic generation of test questions by software agents using ontologies". In: Proceedings of the IEEE 8th International Conference on Intelligent Systems, Sofia, Bulgaria, 4-6 September 2016, pp. 741-746. https://doi.org/10.1109/IS.2016.7737395

[13] N. Stancheva, A. Stoyanova-Doycheva, S. Stoyanov, I. Popchev and V. Ivanova, "A Model for Generation of Test Questions", Comptes Rendus de L'Academie Bulgare des Sciences, vol. 70(5), pp. 619-630, 2017.