Domain ontologies and the conversion of tacit knowledge in software development

Euler EVANGELISTA
PPGSIGC/Fumec University
Belo Horizonte, Minas Gerais 30.310-190, Brazil

Cristiana DE MUŸLDER
PDMA/Fumec University
Belo Horizonte, Minas Gerais 30.310-190, Brazil

ABSTRACT[1]

This study presents a proposal to build and analyze a domain ontology as a tool to support the knowledge transfer process in the context of software requirements analysis in the medical/pharmaceutical industry. The proposal is to use ontologies as an engineering artifact with the objective of representing knowledge in a specific domain, which, in the context of this research, is software modeling. A domain ontology is built to represent the requirements of a data warehouse/business intelligence software in the medical/pharmaceutical industry. The ontology-building process is supported by a specific methodology, defined with the purpose of building such artifacts, named "Methondology," and selected based on the research requirements. A prototype is created in the implementation phase of the ontology-building process. The results demonstrate that ontology domains can contribute to the process of analyzing and representing software requirements, as well as serving as a tool for organizational knowledge transfer through continuous knowledge conversion, which is critical for business sustainability. This study is an attempt to understand the knowledge conversion process in software development projects. Tacit knowledge is complex to articulate through formal language once it has been embedded with individual experience.

Keywords: Ontology, Information Science, Computer Science, Information Systems, Knowledge Management

1. INTRODUCTION

"Software pervades our world, and we sometimes take for granted its role in making our lives more comfortable, efficient, and effective." [1] This statement can be exemplified by the increase in recent decades of technological dependence on information treatment by organizations, and also by people, as software applications are increasingly seen as essential for survival in the information age and the growth of the Internet of Things. However, it is important to highlight the number of problems related to the use of knowledge-based software due to the diverse levels of complexity and the size of implementation projects. These issues highlight that there is still a long way to go to optimize the quality of services and products, despite advances in problem-solving capacity through computer science.

Regardless of the development process model chosen by software engineers, the analysis phase begins by gathering details of requirements. Requirements deal with entities, attributes, and relationships with other entities, and are represented by artifacts or models. According to Milton [2], data modeling methods entail the establishment of real-world data abstractions and information that are acceptable and relevant to the organization and its purposes.

After delineating the requirements, they must be validated in order to move on to the next phase of software development. There are several forms of validation, including requirements analysis. The requirements analysis process goes beyond describing customer-desired behavior and the problem-solving capability of solution developers. It is necessary for the defined requirements to be understood and agreed upon by parties involved in the process (engineers and customers) so that the requirements can be established and tested throughout the various phases until implementation.

Brooks [3], Standish Group [4], Boehm and Papaccio [5] indicated that a significant percentage of problems encountered during the implementation and use of software relate to a lack of understanding and/or poor definition of requirements. The challenge of understanding problems faced by customers goes beyond extraction of the declared problem – that is, the problem as described by the client verbally or in writing. Pfleeger and Atlee [1] stated that it is necessary to examine the customer's needs in depth, because sometimes verbalized requests do not reflect actual needs. In this context, where we seek to understand the problems of organizations and transform them into procedures or software, the issue of knowledge transfer emerges.

According to Nonaka and Takeuchi [6], knowledge management creates ways to reflect on information transfer and knowledge conversion processes. Examples of problems solved via known methods of knowledge conversion can be applied in the context of software development, thus describing a form of conversion of tacit into explicit knowledge. Recent developments in computer technology have enabled the sharing of knowledge regardless of time and location. Rus and Lindvall [7], in their study of knowledge management in software engineering, emphasized the importance of organizational knowledge and its role as companies' intellectual capital, but stated that "the big

[1] *The paper "Domain ontologies and the conversion of tacit knowledge in software development" was edited by Stickler Editing Editing Services. We would like to express our gratefulness to the reviewers Professor Olaf Reinhold and Professor Fabrício Ziviani who took some part of their time to share their comprehensive and detailed thoughts on this works and carefully alert its main issues.*

problem with intellectual capital is that it has legs and walks home every day. Lack of experience comes through the door as soon as the experience comes out the same door." Chugh, Chugh and Punia [8] also contributed to studies on knowledge management applied to software development, analyzing multiple cases to explore the significance of knowledge management for software construction.

According to Nonaka and Takeuchi [6], tacit knowledge is difficult to articulate from formal language, because it is embedded in individual experience and intangible factors such as perspectives, values, and personal beliefs. The real need of the software client comprises the tacit knowledge that one wants to capture, understand, and document – that is, to convert into an explicit format so as to then continue in the software design and have good chances of succeeding in its implementation. Nonaka and Takeuchi [6] stated that explicit knowledge can be transmitted through people more formally and more easily compared to tacit knowledge, since it can be articulated in language that is standardized in terms of grammatical expressions, mathematical expressions, specifications, manuals, etc.

Representation of explicit organizational knowledge requires the use of artifacts such as models, databases, knowledge management systems, and ontologies. The current study approaches ontologies as a way of representing explicit knowledge in the context of software requirements analysis. For Anumba *et al.* [9], ontology is the basis for formalizing knowledge of a domain that contains several concepts described in words that can be formalized. In a broad, or philosophical, sense, Ontology (with capital "O") is the science of what is, of the kinds and structures of objects, properties, events, processes and relations in every area of reality. [10]

Information science approaches ontologies (with lowercase "o") in a less abstract way, seeking practical application of the concept in information systems and computer science. Green and Rosemann [11] stated that "Ontologies are no longer just a philosophical concept. For more than 10 years, researchers from different areas related to information technology have been interested in applying strong ontological foundations in their work. A growing number of publications in newspapers, conferences, and workshops have been dedicated to the application of ontologies in information systems and computer science."

Given the above arguments and assumptions, the research question of this study can be delineated as follows: How can software ontology be used to facilitate knowledge transfer?

In alignment with authors such as Green and Rosemann [11], this paper proposes that ontologies have the potential to be used as artifacts, which can represent, or explain, the requirements of a software. López [12], in his study of methods of ontology construction, stated that ontologies are part of a software. This idea is supported by a definition extracted from the glossary of software engineering terminologies of the Institute of Electrical and Electronic Engineers [13] Software is "computer programs, procedures, and possibly the associated documentation and data pertaining to the operation of a computer system." Ontologies are part (sometimes only potentially) of software products; consequently, ontologies should be developed according to the proposed standards for software in general, albeit adapted to the special characteristics of ontologies [12], In alignment with López [12], the present research compares steps proposed for the construction of an ontology and the stages of analysis of software requirements, according to traditional methodologies of management of the software development cycle. The objective is to verify whether this methodology can be used in place of a typical software requirements analysis process.

The study aims to elucidate the relationship between knowledge management and domain ontology applied to software requirements. The research is based on the analysis of a case of software implementation for the management of production indicators in the medical/pharmaceutical industry. The software in question, a business intelligence platform based on a data warehouse system, was chosen because it required considerable conversion of organizational knowledge during the process of specifying and analyzing the requirements that modeled its development and subsequent implementation.

To achieve the objectives of the research, an ontology was built to represent the knowledge acquired from specialists within the domain of manufacturing control metrics in the medical/pharmaceutical industry. Ontology was used in analyzing data warehouse (DW)/business intelligence (BI) software requirements, and its use was evaluated as an organizational knowledge transfer tool.

2. DEVELOPMENT

2.1. Knowledge management

Knowledge management has gradually come to be recognized as an important process within organizations. This can be explained and supported by the change in the profile of companies' market value in the information age. In the past, the physical assets, or structural capital, of companies were responsible for most of their market value. [6]

But how do organizations use knowledge to develop new products and to continuously improve their internal processes in order to sustain their existence in the market and remain competitive? This issue has been the focus of a significant number of recent studies. Models of good practice have been used to bring the theory of knowledge management into organizations. The coordinated implementation of knowledge management creates sustainable competitive advantage that is difficult to imitate, as the knowledge is rooted in company employees, and not in physical resources, which are easily imitated by competitors and less able to react to environmental uncertainties. [14]

2.1.1. Types of knowledge: tacit and explicit

To better implement knowledge management practices in organizations, it is necessary to understand the two forms of knowledge – tacit and explicit. Distinguishing between the two types and applying the appropriate techniques to extract and transfer them is critical to successful knowledge management. Nonaka and Takeuchi [6] described the two types as follows: (a) tacit knowledge: knowledge that is subjective; skills inherent in people; systems of ideas, perceptions, and experience; and knowledge that is difficult to formalize, transfer, or explain to others; and (b) explicit knowledge: knowledge that is relatively easy to encode, transfer, and reuse; knowledge that is formalized in texts, graphs, tables, figures, drawings, diagrams, and is easily

organized in databases and publications in general, both on paper and electronically.

2.1.2. Conversion of knowledge

It is important to highlight the need for two-way conversion of both types of knowledge in order to better manage organizational knowledge as a whole. Without this continuous conversion, especially in the tacit–explicit direction, organizations run the risk of continually losing their most valuable asset – intellectual capital – when the inevitable loss of talent occurs. How, then, can internal practices, procedures, and processes that ensure continuous conversion be implemented?

The endless cycle of tacit and explicit knowledge conversion in an organization, when institutionalized and encouraged by senior management in the form of internal processes and strategic business objectives, is called a procedural approach to knowledge management. Researchers such as Nonaka and Takeuchi [6] have proven the importance of this organizational competence in the age of knowledge. The most successful organizations currently adopt, and are referenced, in knowledge management processes.

2.2. Software engineering

Software engineers use their knowledge to solve problems through computing. However, not all problems can be solved with the help of computers, and this is an important point in delimiting the boundaries of software engineering. It is necessary to solve the problem first and then, if necessary, use technology as a tool to implement the solution. [1] Thus, the problem analysis process determines whether the technology can be applied to achieving the solution. As many problems are complex, they must first be broken into smaller parts in order to understand them. In this way, they can be described through a set of smaller problems that are easier to solve. After analyzing this set of minor problems, the solution design goes through a reverse process, called synthesis, which entails assembling a large structure from small building blocks. [1]

2.2.1. Software requirements

According to IEEE [13], software requirements comprise conditions or capabilities required for users to resolve a problem or achieve a goal. The notion of "requirement" has more than one meaning, and a variety of taxonomies can be found for requirements; however, the most important is that requirements lead the development of the product [15], and reads objects or entities, the states they can assume, and the functions they perform to change their states or characteristics. [1] During the requirements analysis step, it is not necessary to focus on solving the problem explained by the requirement; rather, the problem and the behavior expected by the software client must be understood.

Requirements can be represented through models or notations. Modeling requirements helps in understanding requirements in a comprehensive way by stimulating questions that should have been asked during the survey phase, and holes in models reveal unknown or ambiguous behaviors. [1]

2.3 Ontology

The origin of the concept of ontology is generally attributed to the ancient Greeks, specifically Plato and Aristotle. Aristotle's students used the word "metaphysics" to refer to the work their teacher described as "the science of being 'in the capacity' of being." [10]. More precisely, ontology in the philosophical sense refers to the determination of fundamental categories of beings and the questioning of when and in what sense it can be affirmed that individual examples within these categories exist [16] In philosophy, ontology is the basic description of things in the world. [17]

Uschold et al. [18] stated that ontologies are often conceived as a set of concepts containing entities, attributes, and processes and their interrelationships. Ontology can take many forms, but it should always include a vocabulary of terms and some form of specification of meanings or definitions. Ontology can be described in the form of a natural, highly informal language, or it can be semiformal, expressed in restricted and structured natural language, or through a formally defined artificial language. Ontology can also assume a strict formal character, with terms meticulously defined via formal semantics and theorems.

Creation tools are currently available to represent ontologies in the form of natural language, encoded language, or even graphically. These tools can be classified into ontology editors (e.g., the Protegé application), ontology-based annotation tools, and ontology-based reasoning tools.

2.3.1. Domain ontologies

Domain ontologies represent concepts and their relationships within a specific scope of the study in question. They generally use high-level ontologies as a reference, deriving from them concepts that share the same basic attributes. Other attributes are aggregated to such entities in order to contextualize and use them as artifacts representing the knowledge contained in a given domain. In information science, an ontology refers to an engineering artifact, consisting of a specific vocabulary, used to describe a certain reality. [17]. This artifact is typically used to validate conceptual models, as described by Sadowska and Huzar [19], which propose the use of a domain ontology to validate a diagram of UML classes. Ontologies have also been proposed to validate conceptual models and schemes. [17]

3 METHOD

To achieve the objectives of this research, the paper proposes to build a domain ontology to represent the requirements of a DW/BI software. In order to seek theoretical support for this proposal, the process of ontology construction is compared to that of the modes of knowledge conversion of the knowledge spiral outlined by Nonaka and Takeuchi [6], known as outsourcing. Outsourcing suggests that knowledge conversion occurs from symbolic representation through models, concepts, and hypotheses. Domain ontology is an artifact, or model, that represents acquired knowledge from a specific domain. The process of creating an ontology requires the extraction of tacit knowledge from specialists from a given domain, and the subsequent conversion of this knowledge into an explicit nature, via the formalization of ontology in any of the formats proposed by existing methodologies.

Given the need to choose a method for the construction of ontology, a search for articles related to the theme was conducted, revealing publications by López [12] and Bautista-Zambranaa. [20] These authors provided an overview of existing methodologies for the construction of ontologies; their criteria served as the basis for choosing the methodology used in this research for building ontologies, deemed "Methondology."

To verify whether the methodology chosen for the construction of ontology would support the analysis of software requirements, the proposed steps for constructing the ontology and the steps proposed by Pfleeger and Atlee [1] were compared in the context of analyzing software requirements. Similarities were thus noted between the two processes, and a relationship established between their stages.

4. MODEL APPLIED

4.1. Data warehouse step

The problem that the DW system implementation project needed to solve pertained to the process of storing and extracting data for analysis and decision making, which was previously done manually from the manipulation of raw data in an Excel spreadsheet, resulting in nonstandardized, time-consuming, and inaccurate information. In order to resolve these issues, the project proposed the implementation of a DW and a BI tool. A DW is a copy of the operational data of a manufacturing process. It is used by the BI tool to generate indicators to users through charts and tables. This structure automates the generation of information and ensures the uniformity, standardization, and veracity of data. In addition to these benefits, implementation of the system reduced the number of hours needed to generate reports, since manual work on data collection and processing in spreadsheets was reduced and, in some cases, eliminated in reports that were implemented using the new tool. Production supervisors benefited the most, as they had previously devoted hours of work to generating the reports manually.

4.2. Data collection step

A prototype in XML file format, supported by a Web browser, was generated with the aim of enabling navigation between concepts in the domain ontology, as well as their relationships and instances. The prototype was made available to key project users, who were experts in the field, so that they could become familiar with the concepts described in the ontology and evaluate them from the perspective of the quality of its content and potential use. The prototype allowed the graphical visualization of concepts, relationships, and instances in the form of a hyperbolic tree. The ontology evaluation process included the application of two questionnaires.

The first questionnaire addressed questions related to the content of the ontology regarding the quality of requirements of the DW/BI software represented by it. The criteria for evaluating the quality of information were inspired by those for validating software requirements proposed by Pfleeger and Atlee [1], as detailed in the theoretical framework of this study. Since the first questionnaire addressed issues related to the validation of software requirements, technical discipline, it was answered by the systems analysts and members of the DW/BI project team involved in capturing and analyzing these requirements.

The second questionnaire addressed specific questions about the knowledge domain of ontology and questions related to knowledge transfer and learning. In the field of education, it is important to evaluate whether the specific content was learned by a person during the teaching process. [21] This questionnaire was applied to key users of the DW/BI project who were responsible for defining and reporting the manufacturing metrics contemplated in the system and were, therefore, experts in the field of ontology.

In conclusion, use of the prototype, together with analysis of the results of the two questionnaires, enabled evaluation of the content and use of ontology to transfer knowledge and analyze software requirements.

4.3. Ontology created

The domain ontology built as part of this research represents the taxonomy of business performance metrics, including the relationship between these metrics and their visualization dimensions, as is typical of a DW project.

The ontology prototype allowed its evaluators, system analysts and users to research concepts, their attributes, and relationships with other concepts within the domain of production metrics management. The prototype remains available and is considered an important contribution to the company, as it will be used in future for the purposes suggested in this study.

The results of the evaluation showed that most respondents agreed that the derived requirements were in accordance with the quality criteria presented in the questionnaire. These results, together with the aforementioned similarity between the processes of analysis of software requirements and the construction of domain ontologies, reinforce the conclusion that the research proposal is valid.

The results also show that all quality criteria proposed by Pfleeger and Atlee [1] could be evaluated by questionnaire respondents, and that they agreed, in most cases, that the requirements represented in the ontology were in accordance with these criteria.

The respondents' comments to the questionnaire also demonstrate that they agreed that the requirements were documented correctly, and had sufficient, coherent, and consistent information. As stated by one respondent, "ambiguity was not observed, because the information is well segregated between the requirements." The comments also referred to the prototype made available; respondents stated that the tool is interesting, especially the functionality called OntoGraf, which allows users to focus on several different objects by viewing where the objects are used and who uses them.

The comments draw attention to a possible learning curve necessary before users can begin applying the tool, which may require time and training. However, respondents stated that the method would be extremely useful in projects high in technical complexity.

Another point observed from the comments is that the tool enables the structuring and organization of ideas, which helps make requirements clearer and more organized. In addition, it was stated that the tool could be used for knowledge transfer.

4.4. Evaluation of the content of the ontology, knowledge transfer, and learning

The second questionnaire was divided into three sessions. The first consisted of nine questions that aimed to evaluate the content of the ontology. The second, composed of nine additional questions, aimed to evaluate the quality of the ontology information. The third, composed of 10 questions, evaluated the ability of the ontology to assist in the processes of knowledge transfer and learning in the context of managing production metrics. The questionnaire was answered by key users of the DW/BI project who were responsible for defining and reporting the manufacturing metrics contemplated in the system and were therefore experts in the field of ontology.

5. DISCUSSION AND IMPLICATIONS

An important issue pertains to the possibility of using the process of building a domain ontology as a method of conversion and sharing of organizational knowledge. Modern organizations have long recognized the need to retain critical knowledge for sustainability and business improvement. Organizational changes, which have become quite frequent and affect the structure of functional areas, such as the transfer of employees to other units of the organization, or even the loss of employees to other companies, usually result in a loss of critical knowledge regarding the operation and evolution of internal business processes. This loss of knowledge can cause process interruptions and financial losses, and possibly give rise to difficulties in promoting the continuous improvement of those processes. People carry with them the knowledge and skills acquired over years of personal experience with the organization. In general, these skills and knowledge are subjective, and are difficult to formalize, transfer, and explain to others. They therefore represent tacit knowledge.

The characteristics of tacit knowledge can be observed in relation to the requirements for describing calculation formulas, data sources, and the purpose of using business metrics represented in the DW/BI system studied in this research. Prior to construction of the domain ontology, this knowledge, which is critical in supporting the decision-making process within the case company, was restricted to a small group of people. Structural changes and/or loss of talent eventually result in unexpected and time-consuming training efforts of new hires, which must be conducted in a timely fashion in order to avoid business process disruptions and financial losses. The problem lies in the time required, or the learning curve, for the new hire to acquire the knowledge required to build and maintain business metrics that support the company's decision-making process.

One of the objectives of this research is to assess the use of the domain ontology built to support the software requirements of the DW/BI system as teaching/learning tool for new employees who need to understand and maintain business metrics. A questionnaire was applied during the ontology evaluation phase to verify whether the ontology is able to help a person without previous knowledge to understand how the business metrics are calculated and later used.

6. CONCLUSIONS AND NEW PERSPECTIVES

In line with a study by Fonseca [17], this work proposes the use of ontologies as engineering artifacts. The term artifact is understood as per Simon's [22] definition in the context of design science, as "artificial objects that can be characterized in terms of objectives, functions and adaptations." The objective, or the function, of the artifact proposed by the research is to represent knowledge of a specific domain. The specific domain addressed here comprises a set of requirements of DW/BI software, in accordance with Dubielewicz et al. [15], Anderlik, Neumayr and Schrefl [23], Pardillo and Mazón [24] and Green and Rosemann [11] – indeed, the use of domain ontologies as an artifact of representation of knowledge necessary for software modeling is not new in academia.

In addition to evaluating the ontology's use in the analysis of software requirements, the research sought to understand whether the ontology could later be used in the process of knowledge transfer and learning, specifically regarding the management of production metrics – the scope of the software chosen for the study. The potential use of the ontology goes beyond the representation of software requirements. Its role for the organization may thus be greater, contributing to the formation of an environment that favors the creation and sharing of knowledge critical to the business.

In order to verify whether the methodology described herein could be used in place of a typical software requirements analysis process, a comparison was conducted between the proposed steps for the ontology construction and the stages of software requirements analysis according to traditional methodologies of software development cycle management. Similarities were noted between the two processes, and relationships were established between their stages.

The construction and evaluation of the ontology artifact proposed in this study was limited to the specific application in the Data Warehouse software project described in the research. Future studies in the same direction are recommended to explore the potential benefits of using domain ontologies as a representation of knowledge to

7. ACKNOWLEDGMENT

We would like to express our gratefulness to Professor Olaf Reinhold and Professor Fabrício Ziviani for reviewing this paper.

8. REFERENCES

[1] S. L. Pfleeger and J. M. Atlee. **Software engineering, theory and practice.** Waterloo: Prentice Hall, 2006.

[2] S. Milton. **An ontological comparison and evaluation of data modelling frameworks**. Doctoral Thesis, University of Tasmania, 2000.

[3] F. P. Brooks Jr. **No Silver Bullet: essence and accidents of software engineering**. Chapel Hill, 1987. Retrieved April 28, 2021, from http://worrydream. com/refs/Brooks-NoSilverBullet. pdf.

[4] Standish Group. **The CHAOS report**. Dennis, MA: The Standish Group, 1994.

[5] B. W Boehm and P. N. Papaccio. Understanding and controlling software costs. **IEEE Transactions on Software Engineering,** Vol. 14, No. 10, 1988, pp.1462-1477. https://doi.org/10.1109/32.6191.

[6] I. Nonaka and H. Takeuchi. **The knowledge-creating company: how Japanese companies create the dynamics of innovation.** Oxford: eBook Kindle, 1995.

[7] I. Rus and M Lindvall. **Knowledge management in software engineering.** Maryland: Fraunhofer Center for Experimental Software Engineering, 2002.

[8] M. Chugh, N. Chugh and D. K. Punia. **Evaluation and analysis of knowledge management best practices in software process improvement: a multicase experience.** Dehradun, India: IEEE Conference, 2015.

[9] C. J. Anumba, R. R. A. Issa, J. Pan and I. Mutis. Ontology-based information and knowledge management in construction. **Construction Innovation**, Vol. 8 No. 3, 2008, pp.218-239. https://doi.org/10.1108/147141708 10888976.

[10] B. Smith. Ontology. In: L. Floridi (ed.). **Blackwell guide to the philosophy of computing and information**. Oxford: Blackwell, 2003. pp.155-166.

[11] P. Green and M. Rosemann. **Business systems analysis with ontologies.** Queensland: Idea Group Publishing, 2005.

[12] F. M. López. Overview of methodologies for building ontologies, **Proceedings of the IJCAI-99 workshop on Ontologies and Problem-Solving Methods.** Stockholm, 1999. Retrieved April 28, 2021, from http://oa.upm.es/5480/1/Overview_Of_Methodologies.pdf.

[13] Institute of Electrical and Electronic Engineers. **IEEE standard glossary of software engineering terminology.** New York: IEEE, 1990.

[14] S. L. Silva. Gestão do conhecimento: uma revisão crítica orientada pela abordagem da criação do conhecimento. **Ciência da Informação**, Vol. 33, No. 2, 2004, pp.143-151. https://doi.org/10.1590/S0100-19652004000200015. Retrieved April 28, 2021, from https://www.scielo.br/pdf/ci/v33n2/a15v33n2.pdf.

[15] I. Dubielewicz, B. Hnatkowska, Z. Huzar and L. Tuzinkiewicz. **Domain modeling based on requirements specification and ontology.** Wroclaw: Springer International Publishing, 2017.

[16] M. Storga, M. M. Andreasenb and D. Marjanoviça. **The design ontology: foundation for the design knowledge exchange and management.** Zagreb: Lyngby, 2008.

[17] F. Fonseca. The double role of ontologies in information science research. **Journal of the American Society for Information Science and Technology**, Vol. 58 No. 6, 2007, pp.786-793. https://doi.org/10.1002/asi.20565. Retrieved April 28, 2021, from http://mba.eci.ufmg.br/downloads/pos/OntologyDoubleRole-Fonseca.pdf.

[18] M. Uschold, M. King, S. Moralee and Y. Zorgios. The enterprise ontology: artificial intelligence. **The Knowledge Engineering Review**, Vol. 13, No. 1, 1998, pp. 31-89. https://doi.org/10.1017/S0269888998001088.

[19] M. Sadowska and Z. Huzar. **Semantic validation of UML class diagrams with the use of domain ontologies expressed in OWL 2**. Wroclaw: Springer International Publishing, 2017.

[20] M. Bautista-Zambranaa. Methodologies to build ontologies for terminological purposes. **Procedia - Social and Behavioral Science**s, Vol. 173, 2015, pp. 264-269. https://doi.org/10.1016/j.sbspro.2015.02.063.

[21] M. B. Almeida. A proposal to evaluate ontology content. **Applied Ontology**, Vol. 4, 2009, pp. 245-265. https://doi.org/10.3233/AO-2009-0070. Retrieved April 28, 2021, from http://mba.eci.ufmg.br/downloads/300697.pdf.

[22] H. A. Simon. **The sciences of the artificial.** (3rd ed.). Cambridge, MA: MIT Press, 1996.

[23] S. Anderlik, B. Neumayr and M. Schrefl. Using domain ontologies as semantic dimensions in data warehouses. In: P. Atzeni, D. Cheung and S. Ram (eds.). **Conceptual modeling**. Heidelberg: Springer, 2012. pp. 88-101. https://doi.org/10.1007/978-3-642-34002-4_7.

[24] J. Pardillo and J. N. Mazón. Using ontologies for the design of data warehouses. **International Journal of Database Management Systems**, Vol. 3, No. 2, 2011, pp. 1-115. https://doi.org/10.5121/ijdms.2011.3205. Retrieved April 28, 2021, from https://arxiv.org/ftp/arxiv/papers/1106/1106.0304.pdf.