Journal of Information Technology and Cyber Security 3(1) January 2025: 14-32 EISSN 2987-386X | PISSN 2987-3878
DOI: 10.30996/jitcs.12297

Ontology in Requirements Software Development Method: A Systematic Literature Review

Reza Fauzan [1,*], Mohammad Zaenuddin Hamidi [2], Winda Ayu Safitri [3], Daniel Oranova Siahaan [4], and Muhammad Ihsan Karimi [5]

[1] Department of Informatics Engineering, Politeknik Negeri Banjarmasin, Indonesia
[2] Department of Informatics Engineering, Universitas Mataram, Indonesia
[3,4] Department of Informatics Engineering, Institut Teknologi Sepuluh Nopember, Indonesia
[5] Solution Architect Industry Services, Dassault Systemes, Berlin, Germany
* Corresponding author: reza.fauzan@poliban.ac.id

Received: 06 December 2024
Accepted: 21 January 2025
Revised: 14 January 2025
Available online: 24 January 2025

To cite this article: Fauzan, R., Hamidi, M, Z., Safitri, W. A., Siahaan, D. O., & Karimi, M. I. (2025). Ontology in Requirements Software Development Method: A Systematic Literature Review. *Journal of Information Technology and Cyber Security, 3*(1), 14-32. https://doi.org/10.30996/jitcs.12297

Abstract

The requirement process is one of the most critical factors in determining whether the software development process is successful. It is crucial to consider the function that ontology plays in the requirements of software engineering development. People and organizations can more easily utilize and share data, information, and knowledge with one another because of the implementation of ontology. During our systematic assessment of the literature published between 2011 and 2020, we came across twenty publications that discussed ontology in requirements and how it might be used in software development processes. To determine which studies were the most pertinent to our research endeavors, we developed and implemented inclusion and exclusion criteria in two separate rounds. The review identified the leading ontology in data software development challenges. We found various ways to do this in our selected papers with different systematics as well. However, our findings indicate that the ontology requirements in software development must be addressed by examining various software development methods apart from agile Scrum and Extreme Programming (XP).

Keywords: ontology, requirement, software development method.

1. Introduction

An ontology is a structured framework used to define and organize knowledge within a particular domain (Biagetti, 2021; Osman, Noah, & Saad, 2022). It provides a formal representation of concepts, their attributes, and the relationships connecting them, ensuring clarity and shared understanding among users (Pliatsios, Kotis, & Goumopoulos, 2023; Said, et al., 2023). By offering consistent terminology and structure, ontologies help reduce ambiguities and improve communication across different systems or fields (Fraga, Vegetti, & Leone, 2020; Guizzardi & Guarino, 2024). They are commonly applied in areas such as artificial intelligence, semantic web, software development, and data management to support knowledge sharing, integration, and reasoning (Chaccour, Saad, Debbah, Han, & Poor, 2024; Fu, Jiang, & Chen, 2022; Olan, et al., 2022). Additionally, ontologies enable better interoperability of data, enhance decision-making, and facilitate the development of automated systems by organizing and connecting information in a systematic way (De Nicola & Villani, 2021; Jing, et al., 2022).

The use of ontology in recent years has the biggest opportunity in the requirement (Farghaly, Soman, & Zhou, 2023; Tudorache, 2020). The ontology is used in any development technology, such as in artificial intelligence, the Internet of Things, medical informatics, and information architecture (Alrumaih, Mirza, & Alsalamah, 2020). Software development ontology can describe various formats such as semantic web, UML, and other documents (Husáková & Bureš, 2020). The requirements process is a vital element in assessing the success of the software development process. It is essential to evaluate the role of ontology in the requirements of software engineering development. The adoption of ontology facilitates the easier

utilisation and sharing of data, information, and knowledge across individuals and organisations. Ontologies serve as a conduit that links many systems, stakeholders, and domains via the standardisation of ideas and their interrelations. This standardisation mitigates uncertainty in requirement interpretation, enabling a common understanding across all stakeholders in the software development lifecycle. The application of ontology extends beyond the basic organisation of information. It offers a systematic framework for semantic integration, allowing computers to read data meaningfully and engage intelligently. In software engineering, ontology assists in correlating domain-specific terminology with technical entities, hence enhancing automated reasoning and validation procedures. This capacity is essential in dynamic settings such as Agile development, where requirements often change, and ensuring consistency poses a considerable difficulty. Furthermore, ontology may be used to represent domain knowledge, rendering it available for several applications, including test case generation, design validation, and compliance assurance with industry standards. Ontology substantially alters the definition, analysis, and implementation of software requirements by bridging the divide between human comprehension and machine interpretation.

It is crucial to consider the role that ontology plays in the requirements of software engineering development (Abioye, Arogundade, Misra, Akinwale, & Adeniran, 2020; Poveda-Villalón, Fernández-Izquierdo, Fernández-López, & García-Castro, 2022). The goal of ontology is to achieve intelligent system interoperability by making it easier for individuals and organizations to utilize and share data, information, and knowledge with one another (Shahzad, et al., 2021; Smirnov, Levashova, Ponomarev, & Shilov, 2021). This approach resolves semantic conflicts between the various data sources. Ontology also makes it possible to meet the needs of consumers and stakeholders more quickly (Helmy, Abdelgaber, Fahmy, & Montasser, 2020; Pileggi, 2021). The requirements process is a vital element in assessing the success of the software development process. It is essential to evaluate the role of ontology in the requirements of software engineering development. The adoption of ontology facilitates the easier utilisation and sharing of data, information, and knowledge across individuals and organisations. Ontologies serve as a conduit linking many systems, stakeholders, and disciplines via the standardisation of ideas and their interrelations. This standardisation mitigates uncertainty in requirement interpretation, enabling a common understanding across all stakeholders in the software development lifecycle. Ontology extends beyond the basic organisation of information. It offers a systematic framework for semantic integration, allowing computers to comprehend data meaningfully and engage intelligently. In software engineering, ontology assists in aligning domain-specific terminology with technical entities, hence enhancing automated reasoning and validation procedures. This capacity is essential in dynamic settings such as Agile development, where requirements often change, and ensuring consistency poses a considerable difficulty. Furthermore, ontology may be used to represent domain information, rendering it accessible for diverse applications like test case generation, design validation, and compliance assurance with industry standards. Ontology substantially alters the definition, analysis, and implementation of software requirements by connecting human comprehension with machine interpretation.

Numerous studies have explored the role of ontology in requirement engineering (RE). For example, Sitthithanasakul & Choosri (2016) demonstrated that the use of ontology in Agile processes significantly improves communication between software development teams and stakeholders, particularly in eliciting and documenting requirements. Additionally, Mukhopadhyay & Ameri (2016) highlighted that ontology facilitates consistency in product specifications by providing a structured framework to resolve ambiguities and conflicts during the design phase. In their study, they implemented ontology in a software development project, reducing inconsistencies by 25% compared to traditional approaches. These findings underscore the potential of ontology as a critical tool in enhancing both the efficiency and accuracy of requirement engineering across various methodologies.

An application from the avionics sector has been given its own ontology, which has been built to reflect the software needs of the application (Fauzan, Siahaan, Rochimah, & Triandini, 2018; Tan, Adlemo, Tarasov, & Johansson, 2017). Several of the systems used in the avionics sector must have high safety-criticalness. Several industry standards, such as DO-178B (Johnson, 1998), must be adhered to throughout the software development process for these systems to comply. To begin the design (Fauzan, Siahaan, Rochimah, & Triandini, 2018), during the implementation stages of the system, it is essential to conduct an analysis to specify and verify the needs of both the entire system and the individuals comprising it. As part of our work, we established an ontology representing the criteria that must be met by a software component associated with an embedded system. In addition to being supplied in papers written in normal language, avionic industry professionals who are considered experts in their field have manually checked the standards. The requirements ontology generated based on these needs will be used later in the software

development process to accommodate more sophisticated techniques. More specifically, the ontology is used in our work to automate a portion of the testing process, namely the development of test cases. This is the particular application of the method.

Over the years, several articles have been written on this field of ontology study. The assessment of an ontology is of utmost significance since it shows the model's validity and usefulness (Guizzardi, 2005; Guizzardi, Halpin, & Halpin, 2008). In this work, the assessment is centered on determining whether the requirements ontology is beneficial (Hlomani & Stacey, 2014; Wang, Mendori, & Xiong, 2014). This is because, in most instances, the individuals who utilize an ontology are not typically the same individuals responsible for developing the ontology. Similarly, the requirements ontology presented in this article may be used by quality engineers or testers employed in the avionics industry. Through our work, even a computer program may be used to automate a portion of the testing process by using ontology as an input. The assessment results are provided, including the user experiences that occurred while using the needs ontology. We take into consideration the fact that the ontology is both usable and applicable. Therefore, this study aims to analyze the application of ontology in the software requirements engineering process through various development methods, such as Agile, Scrum, and Extreme Programming (XP). In addition, this study explores the main challenges faced in ontology development and identifies potential improvements through automation and standardization.

2. Related Works

There is little research in ontology requirements on software methods such as software development in agile (Tan, Ismail, Tarasov, Adlemo, & Johansson, 2016), ontology to enhance requirement engineering in agile software process (Murtazina & Avdeenko, 2019; Murtazina & Avdeenko, 2018; Sitthithanasakul & Choosri, 2016; Triandini, et al., 2022).

An ontology is used to improve the process of requirement engineering in agile software development (Sitthithanasakul & Choosri, 2016). Utilising ontologies may expedite the elicitation of customer or stakeholder needs by offering a standardised framework for recording and understanding requirements. This enhances communication by guaranteeing that all stakeholders possess a uniform grasp of terminology and ideas. Furthermore, the ontology output can be converted into various formats, including a requirements document, a Unified Modeling Language (UML) diagram, a data model, and Web Ontology Language (OWL) applications. These outputs improve validation by establishing explicit correlations between requirements and system components, facilitating the identification and resolution of problems. Based on the previous findings (Bhatia, Kumar, Beniwal, & Malik, 2020), we identified the criteria that have been modified, removed, or introduced. Additionally, to automatically include the updated requirements, we used the Browser View (OWL Doc) plugin of the Protégé software. This plugin converts the revised Software Requirements Specification (SRS) ontology into HTML format, ensuring that the updated SRS is accessible and traceable for all project stakeholders.

The study by Peroni (2017) describes a unique agile approach to ontology production. The Test-Driven production process significantly impacted this approach in Software Engineering and contemporary agile ontology development methods like Extreme Design (XD). This methodology prioritises iterative development and validation, guaranteeing that the ontology adapts to project requirements while preserving consistency and quality throughout the development process.

A approach based on ontology is used for requirements engineering in an agile context. This method uses the organised framework of ontology to formalise and standardise requirement representation, hence minimising ambiguities and inconsistencies. The use of ontologies will enhance the quality of requirements management and development by matching stakeholder expectations with technological execution. Implementing the recommended technique will allow for the timely oversight of newly established requirements that have been enacted, hence promoting the early identification of faults and discrepancies throughout the development process. It will serve as an effective instrument for documenting the project's progress, offering a clear and methodical approach to monitor modifications and updates. This is especially advantageous in an agile environment, marked by frequent adjustments of requirements owing to its iterative nature and swift response to changing business demands. Employing ontology in these situations guarantees precise capture and efficient communication of changes among all stakeholders, hence improving cooperation and minimising the risk of misunderstanding (Murtazina & Avdeenko, 2018).

Another study by Adnan and Afzal (2017) introduced a methodology that employs ontology models tailored for Scrum and XP to address two critical challenges in software development: software effort estimation and knowledge management, particularly within e-commerce systems. These ontology models

Table 1
Search sources.

Electronic databases	Searched items Search applied on	Language	Publication period
ACM Digital library, IEEE, SpringerLink, and ScienceDirect	Journal and conference	English	2010-2020

provide a systematic framework for collecting and maintaining domain information, enhancing the precision of effort assessment and the efficacy of knowledge sharing across development teams. The suggested strategy concurrently incorporates technologies that enable the systematic management of process knowledge throughout the development lifecycle. This is particularly advantageous during the iterative and incremental development stages in Scrum and XP, when rapid response to changing requirements is essential. The technique further integrates tools for storing and organising process information into semantic repositories for both XP and Scrum processes. These repositories facilitate the maintenance of a centralised and readily available knowledge base, hence enhancing decision-making in subsequent initiatives. Moreover, the methodology actively motivates project stakeholders to record insights gained during the development process. This method facilitates ongoing development via reflection on prior experiences and guarantees the preservation of essential insights for future iterations or initiatives. This strategy utilises ontology to foster cooperation, mitigate the likelihood of redundant errors, and improve the overall efficiency and quality of e-commerce software development.

Even though many highlight the use of ontology requirements as a support for system development, few know how to apply ontology to various system development methods. Thus, we propose to make SLRs related to the application of ontology requirements in various system development methods. In addition to discussing applications in the development process, we also discuss future challenges related to ontology requirements in software development methods.

3. Methods

To conduct our study, we adhered to the principles proposed by Fauzan, et al. (2023), Kitchenham, et al. (2007), and Kitchenham, et al. (2010). Subsequently, we discussed the primary stages of our systematic review, which included planning, conducting, and presenting the findings.

3.1. Planning the Review

Our goal was to submit research questions relevant to our object study based on the work associated with it, and we presented this information in the following.

3.1.1. Review the objective and research question

Due to the creation of ontology in necessity, several breakthroughs in other techniques have been made feasible. This is because of the availability of ontology. One example is the use of ontology in agile requirements, which assists software requirements for traceability. In addition, several other things are used in the Agile software development approach to enhance the RE process. The fundamental purpose of this research is to ascertain the degree of success that may be achieved via the use of ontology regarding the fulfillment of agile requirements, as well as to identify the challenges encountered when trying to include ontology needs in agile. Taking this into consideration, we propose that the following research question shall be examined as part of this study:

R1. How do we apply ontology requirements to software development methodology?
R2. What are the challenges of applying ontology requirements to software development methodology?

3.1.2. Search strategy

In accordance with the specification of the research project, we start by conducting a search for relevant material and reading content that is associated with the topic that we have chosen to investigate. Table 1 is an overview of some of the sources that we have found. The table information includes the electronic database, the search item, the language, and the publishing period. The sorts of publications that we used throughout the publishing era from 2010 to 2020 include journals, proceedings, and conferences. Other forms of publications include conferences. The purpose of this supplementary plan was to include any and all potential works that could have been missed for inclusion in the original plan. Following that, the criteria for inclusion and exclusion were applied to the studies retrieved in two successive rounds, each of which comprised a different number of researchers. This was done to ensure that none of the studies were excludeed.

3.1.3. Search Criteria

For the search criteria, we utilized two components, C1 and C2, defined as follows: C1 is a string composed of keywords related to requirements in software development methods, including terms such as

"agility," "agile," "Scrum," and "extreme programming." C2, on the other hand, is a string consisting of keywords associated with ontology in requirements, such as "ontology requirements," "ontology," and "software requirement." To illustrate, an example of a search performed in electronic databases combines these components as follows: "Ontology in requirement (agile OR agility OR Scrum OR 'XP' OR 'extreme programming' OR 'software development') AND ('ontology in requirements' OR 'requirements')."

3.1.4. Inclusion and exclusion criteria

In this study, inclusion and exclusion criteria were employed to determine the eligibility of studies for the systematic review. The inclusion criteria were designed to ensure that only relevant, high-quality, and research-aligned studies were considered. First, the selected studies had to be peer-reviewed publications (I1), guaranteeing academic rigor and validity. Second, the studies had to be written in English (I2), as it is the most widely used language in the global research community and facilitates broader understanding. Third, the studies needed to be relevant to the predefined search keywords, such as "ontology in requirements" and related terms (I3). Fourth, the publications considered had to be empirical research papers, experience reports, or workshop papers (I4), as these types of works provide data that can be analyzed to answer the research questions. Finally, only studies published between 2010 and 2020 (I5) were included to ensure the findings were aligned with recent advancements in the field.

Conversely, the exclusion criteria were applied to filter out studies that were less relevant or unsuitable. First, studies that did not explicitly focus on ontology in software requirements but mentioned it only as a secondary or incidental topic (E1) were excluded. For instance, papers that used the term "ontology" descriptively without in-depth analysis were not considered. Second, studies that did not address the application of ontology in software development requirements (E2) were also excluded. Third, any studies that failed to meet one or more of the inclusion criteria, such as non-peer-reviewed papers or those published outside the specified time range (E3), were eliminated. Lastly, publications such as opinion pieces, keynote speeches, editorials, comments, tutorials, prefaces, anecdotal accounts, or slide-based presentations without accompanying papers (E4) were excluded due to the lack of empirical evidence that could be utilized for analysis.

This rigorous approach ensured that the selected studies were directly relevant and contributed meaningfully to the research objectives. By applying these strict criteria, the systematic review could focus on studies that provided substantial insights into the role of ontology in software requirements engineering.

3.2. Conducting the Review

In this section, we present the findings of our search and extraction of information from relevant sources and databases.

3.2.1. Study search and selection

The research search and selection process was conducted systematically to ensure the inclusion of high-quality and relevant literature for the systematic review. In accordance with the search strategy defined in Section 3.1.2, databases were meticulously chosen, and searches were conducted using predetermined keywords and search strings. The searches aimed to obtain research from diverse sources, including journals and conference proceedings. The preliminary search yielded 71 publications, constituting the pool of prospective studies for evaluation. One of the researchers read the titles and abstracts of these papers during the first screening phase, designated as Round 1. At this point, the inclusion criteria (I2, I3, and I5) were rigorously enforced. This guaranteed that all papers were written in English, relevant to the search criteria, and published between 2010 and 2020. Following the first screening phase, 24 individuals for the research were selected. The studies were chosen for their compliance with the inclusion criteria and their relevance to the study issue. Particular emphasis was placed on confirming that all chosen research were from esteemed publications or conferences, in accordance with inclusion criteria I4. In Round 2, the selected studies were subjected to further examination via the application of the exclusion criteria (E1, E2, E3, and E4). This stage sought to exclude studies that either did not explicitly address ontology in software requirements or were otherwise unrelated to the study topic. At this point, four more articles were removed for not meeting the criteria, resulting in a final total of 20 publications considered appropriate for inclusion in the review.

The research team performed the review of these studies collectively to guarantee uniformity and minimise bias. The researchers independently evaluated the chosen papers according to established grading standards. The ratings were used to assess the methodological rigour, relevance, and quality of the research. Discrepancies in the ratings awarded by the researchers were reconciled by discussion and agreement. This method guaranteed that the final selection of research was of superior quality and consistent with the review's aims. Table 2 provides a comprehensive overview of the research selection pro-

Table 2
Study Selection.

Database	Retrieved	Round 1		Round 2	
		Include	Exclude	Include	Exclude
ACM Digital library	3	2	1	0	2
IEEE	23	13	10	8	5
SpringerLink	8	7	1	1	6
ScienceDirect	7	4	3	3	1
Scopus	30	18	12	8	10
Total	71	44	27	20	24

cess, including the number of studies retrieved, included, and eliminated at each step across several databases. The stringent selection method established a solid basis for the ensuing analysis and synthesis of results in this systematic review.

3.2.2. Data extraction and synthesis

The data extraction and synthesis process was designed to systematically collect and organize information from the 20 primary studies selected during the selection phase. This stage was essential for answering the research questions and ensuring that all pertinent components of the investigations were taken into account. A structured data extraction form was created to do this. This form functioned as a standardised instrument to extract essential components from each research, ensuring uniformity and precision in data gathering. The form was used to document several facts, including the review date, which signified when the research was evaluated, together with the title and authors, which offered the fundamental identification of each investigation. The reference and database from which the research was obtained were recorded to guarantee traceability and transparency. Additional information was gathered to evaluate the pertinence of each study to the research issue, concentrating on the use of ontology in software requirements, including the difficulties, practices, and methodologies addressed. The technique used in each study, including interviews, case studies, reports, or surveys, was documented to comprehend the research methodologies implemented. Data analysis methodologies and validation procedures were used to assess the robustness and reliability of the results in each investigation. Additionally, the form included information on prospective future work proposed by the authors, establishing a foundation for identifying gaps and possibilities for additional study. The studies identified limitations to contextualise the results and comprehend the restrictions of the study done. To provide a comprehensive view, information like the country or location of the research and the year of publication was also included. This methodical technique to data extraction guaranteed that all pertinent ideas, thoughts, and contributions from the selected research were documented thoroughly. The gathered data were then synthesised to allow for advanced interpretation, enhancing comprehension of ontology's function in software requirements engineering. Through the integration and analysis of the gathered data, the review successfully derived significant findings and offered vital insights into the research enquiries.

3.2.3. Methodological quality assessment

This systematic review employed the quality criteria proposed in previous works—namely, Guyatt & Rennie (1993) and Guyatt G. , Rennie, Meade, & Cook (2014) —to assess the methodological quality of the primary studies selected for review, with a particular focus on empirical studies discussing the application of ontology in software development methodologies. These quality standards consist of questions that evaluate the degree of fulfillment and the extent to which research will expand the field of investigation. The quality assessment process followed a structured approach based on established guidelines (Fauzan, et al., 2023). Each study was independently assessed by researchers using predefined criteria, including validity, relevance, and methodological rigor. For instance, studies were scored on a scale of 1 to 5 based on their alignment with inclusion criteria, clarity in methodology, and robustness of findings. Disagreements were resolved through discussion until consensus was reached. The criteria center on the study's importance, reliability, and comprehensiveness. Our selection of these criteria was based on two factors: (i) the quality metrics linked to these criteria have been utilized in several recent systematic reviews, and (ii) they may be used to evaluate the utility of synthesis results and interpretation.

Quality scoring is crucial in ensuring the reliability and validity of included studies. Standardized scoring allows for an objective evaluation of study robustness and helps to identify gaps in methodological rigor. Fig. 1 illustrates the distribution of research papers based on their quality scores across different ranges. Most of the papers, represented by the largest orange segment, fall within the quality score range of 66% to 85%, indicating that most studies achieved moderate to high-quality standards. A smaller portion,

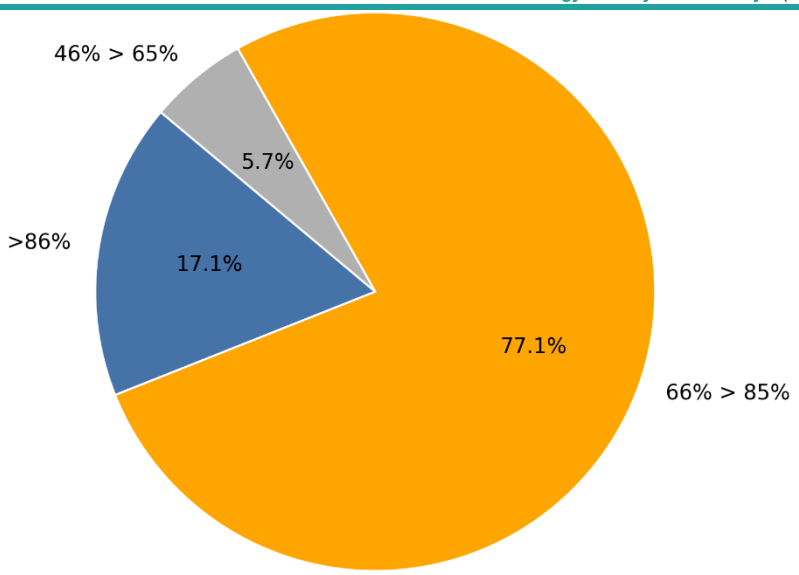

Fig. 1. Percentage scores for quality assessments of studies.

represented by the blue segment, achieved quality scores exceeding 86%, showcasing a limited number of papers with exceptionally high quality. Additionally, the grey segment represents papers with quality scores between 46% and 65%, suggesting a moderate quality level for these studies. Papers with quality scores below 45% or 20% are either negligible or absent, highlighting a general trend toward acceptable and higher-quality research within the analyzed dataset.

4. Findings

4.1. Overview of Studies

As noted before, we found 20 studies. Thirty percent (4) of the 20 research were journal papers, while around seventy percent (14) were conference publications. It should be dispersed among the publishing venues, according to our findings. One or two articles have been published in each publishing source. We found that the majority of co-authors in each of our 20 papers were from different nations. Nevertheless, authorship by regional distribution per research cannot be ascertained.

Of the 20 studies included, 70% scored between 66% and 85% on the quality assessment scale, indicating moderate to high quality. However, 30% of the studies scored below 65%, reflecting limitations in methodological rigor or reporting. These studies were included to provide comprehensive coverage of the field but were carefully interpreted to avoid biasing conclusions. For instance, one study lacked detailed validation of its proposed ontology model, reducing its reliability.

The line chart depicts the number of research papers published over several academic years, highlighting fluctuations in publication activity (see Fig. 2). Starting from 2011–2012, the number of papers is 3. Fig. 2 shows a slight decrease in 2013–2014, indicating a small dip in publication output. A significant surge was observed in 2015–2016, where the number of papers rose to 6, marking the highest point on the chart. However, this is followed by a sharp decline in 2017–2018, dropping to 2 papers. The publication count rebounded in 2019–2020, reaching 6 again. Overall, the chart reflects an inconsistent trend in the number of papers published, with notable peaks in 2015–2016 and 2019–2020, and declines in the intervening years.

4.2. RQ1: How do we apply ontology requirements to software development methodology?

The extraction of requirements into an ontology is a crucial process for converting system requirements into a structured ontology model. The process starts with the identification of the requirement domain, whereby the total system needs are comprehended in relation to its context. In an e-commerce system, a requirement such as "customers can add products to the cart" is seen as an essential feature. At this juncture, papers like the SRS or stakeholder interviews serve as the main sources of information. The subsequent phase involves requirements analysis and the identification of important concepts. This procedure entails aligning the specified criteria with the system's fundamental ideas. Text analysis methods may be used to extract keywords that delineate principal entities, properties, and connections. For instance, from the stipulation “customers can add products to cart,” the principal notions discernible are Customer, Product, and Cart, with the add-to relationship linking these entities. After

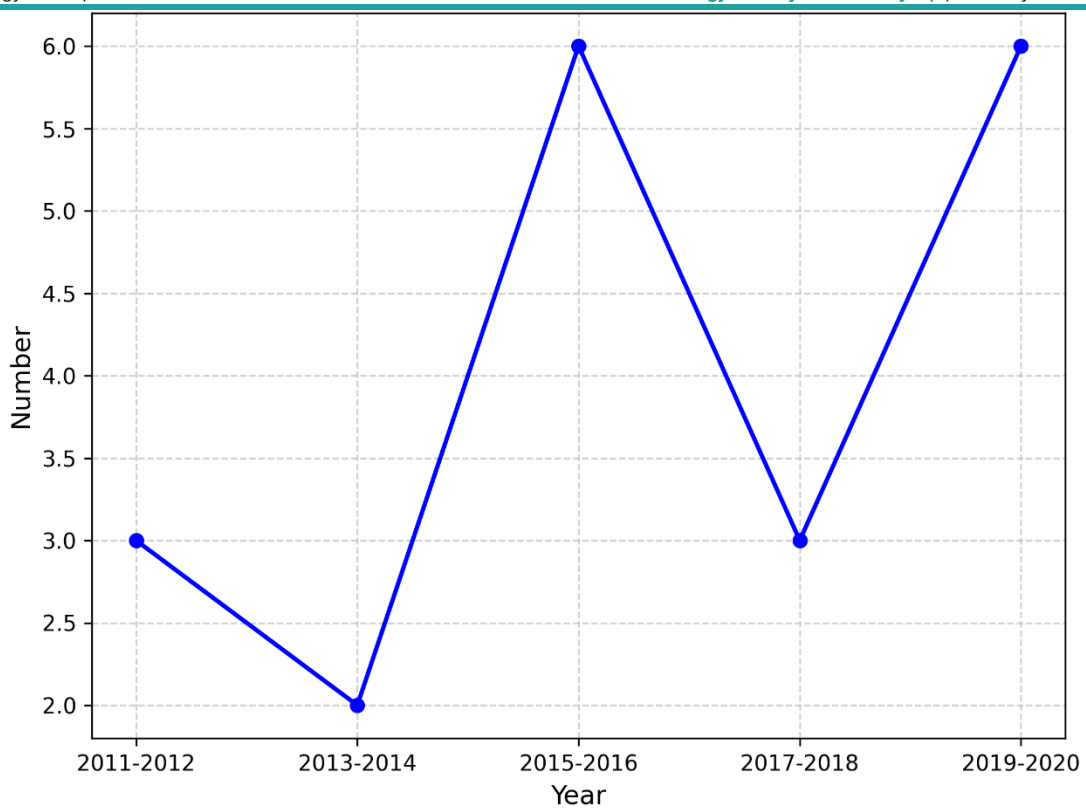

Fig. 2. Distribution of studies by year.

identifying the important ideas, the subsequent step is to align the concepts with the ontology framework. During this process, ideas are converted into ontology components, including classes, attributes, and connections. In the aforementioned need situation, Customer, Product, and Basket are delineated as classes inside the ontology. Attributes like product price are categorised as data properties of the Product class, but the connection "adds to" functions as an object property linking Customer to Basket. The subsequent phase involves the formal articulation of the ontology, where the mapped concepts are expressed in languages such as OWL or Resource Description Framework (RDF). This procedure may be executed using tools like Protégé, which offers an interface for delineating classes, attributes, and relationships inside the ontology. The concluding phase is ontology validation, in which the resultant ontology is assessed to confirm its consistency and completeness. In Protégé, tools like the HermiT Reasoner may verify the ontology's coherence, guaranteeing that all criteria are met without semantic discrepancies. Validation includes evaluation by domain experts to confirm the ontology's relevance and suitability to the original requirements.

For instance, in the creation of an airline ticket booking system, a requirement like “passengers can select seats” can be shown by establishing classes for Passenger, Seat, and Flight. The choose relationship establishes a connection between Passenger and Seat. Data characteristics, such as seat number, are included as elements of the Seat class. This procedure guarantees that user needs are well articulated, enabling the system to comprehend and process them autonomously. This method enhances communication efficiency among stakeholders while ensuring that system needs are handled with exceptional consistency and precision.

We found several studies related to using ontology in requirements for development methods. The first is During the development of the ontology. In particular, an ontology offers a means of exchanging information that ensures uniform language across the application. Regarding service-oriented computing (Bichier & Lin, 2006; Yangui, Goscinski, Drira, Tari, & Benslimane, 2021), ontologies play an important role (Tsai, Wu, Jin, Huang, & Li, 2013). For the developers worked as a pair and followed an iterative and incremental process. In each iteration, the developers followed the steps suggested in Ontology 101 (Cristani & Cuel, 2005; Kendall & McGuinness, 2019; Iqbal, Murad, Mustapha, & Sharef, 2013) and considered the activities described in the supporting process in methontology (Corcho, Fernández-López, Gómez-Pérez, & López-Cima, 2005; Fernández-López, Gómez-Pérez, & Juristo Juzgado, 1997; Zakaria, et al., 2018). In each iteration, the developers met with industry experts to discuss the issues encountered

during the acquisition and specification steps. They also received feedback from the ontology's users and modified it when needed.

The development life cycle in Ontological Engineering, as outlined by Tan, Ismail, Tarasov, Adlemo, & Johansson (2016), offers a systematic framework for ontology creation and maintenance, including five separate phases: evaluation, maintenance, formalisation, conceptualisation, and specification. The steps are structured to guarantee that the ontology accurately encapsulates domain information, fulfils stakeholder expectations, and retains adaptability to changing demands. The evaluation phase includes comprehending and delineating the domain needs, determining the ontology's objective, and evaluating the extent of its applicability. The maintenance phase emphasises the continual updating of the ontology to accommodate changes in domain knowledge or system needs. The formalisation phase transforms conceptualised ideas into a structured format via logical representation or ontology languages like OWL. The conceptualisation phase focusses on delineating the fundamental ideas, connections, and properties within the domain. The specification step delineates the ontology's structure and components for implementation in software systems. Abdelghany, Darwish, & Hefni (2019) offer a methodology that incorporates agile principles and practices into the ontological engineering process, aligning the dynamic and iterative characteristics of agile with ontology building. This technique categorises the ontology production process into three primary phases: pre-game, development, and post-game. The pre-game phase includes the delineation of the ontology's aim and scope, the selection of tools and techniques, and the identification of ontology needs and sources. This phase lays the foundation for ontology development by ensuring that project objectives and resources are clearly defined. The development phase encompasses several iterative procedures. The process starts with the accumulation of information, whereby domain expertise is obtained from several sources, such expert interviews, existing documentation, or data analysis. The conceptualisation step organises information into coherent ideas and connections. Formalisation converts these notions into a logical framework, while integration guarantees that the ontology is compatible with pre-existing ontologies or systems. Agile-inspired methodologies such as sprint preparation and sprint review are used to provide fast iterations and ongoing input from stakeholders, assuring the ontology's relevance and accuracy throughout its development. The post-game phase ultimately concentrates on finalising the ontology for deployment. This step encompasses comprehensive verification and validation, guaranteeing that the ontology is coherent, exhaustive, and fulfils its intended purposes. Validation is often conducted by automated reasoning tools or by engaging domain experts to confirm its relevance in practical situations. The post-game phase also involves documenting the ontology and establishing protocols for its use and upkeep. This systematic and iterative methodology not only improves the speed and precision of ontology building but also permits adaptability to evolving needs. Integrating agile approaches into ontological engineering guarantees that the final ontology is resilient, user-centric, and attuned to the evolving requirements of contemporary systems and applications.

Secondly, ontology's use in agile techniques, particularly Scrum and XP, has been extensively examined for its capacity to improve the efficacy and efficiency of software development. Previous research by Schwaber (2004) and Srivastava, Bhardwaj, & Saraswat (2017) highlighted the significance of ontology in Scrum for enhanced organization and administration of requirements, while other research by Beck (2000) Beck (2000) and Shrivastava, Jaggi, Katoch, Gupta, & Gupta (2021) examined its advantages in XP for optimizing processes and augmenting flexibility. Ontologies are advantageous in agile contexts since they provide a systematic framework for defining and standardising ideas, minimising ambiguities, and improving communication among stakeholders. This organised methodology is further upon in a semi-formal ontology pertaining to the agile area of software processes. The primary aim of this ontology is to provide a standardised vocabulary and structure for the discourse of agile-related terminology, applicable to diverse use cases. Three principal implementation cases are identified: the instantiation of elements (mapping specific agile components), the instantiation of concepts (defining relationships between agile principles and practices), and the development of web applications to visualise these relationships, as emphasised by Ortega-Ordoñez, Pardo-Calvache, & Pino-Correa (2019). Additionally, a prior study by Sitthithanasakul & Choosri (2016) suggested a bifurcated methodology for incorporating ontology into Agile Requirements Engineering (RE). The first phase emphasises ontology development, which entails delineating the scope and objectives of the ontology, identifying the domain, and formalising the governing rules and constraints. The second step is the incorporation of ontologies into Agile Requirements Engineering, particularly for activities like requirement elicitation, when stakeholders' demands are collected and examined. During this phase, the ontology facilitates the organisation and verification of requirements to guarantee their alignment with the project goals. This method encompasses the construction of the ontology and its validation via

iterative methods. Other research by Cao & Ramesh (2008), Inayat, Salim, Marczak, Daneva, & Shamshirband (2015), and Schön, Thomaschewski, & Escalona (2017) underscore the significance of these stages in addressing the frequent demand changes inherent in agile techniques. Ontology functions as a crucial tool for preserving consistency, enhancing communication, and ensuring that all stakeholders have a unified understanding of project needs. The use of ontology into Agile approaches like as Scrum and XP provides substantial advantages, including greater traceability, superior management of changing requirements, and enhanced collaboration. Ontology offers a formalised framework for managing and visualising ideas, aiding agile teams in navigating the fluidity of software development, hence enhancing project delivery in terms of correctness and efficiency.

The suggested management requirements framework employs an ontology-driven Knowledge Management (KM) approach to assist organisations in efficiently addressing difficulties in requirements engineering. This approach is especially beneficial for managing the intricacies of global software development projects, as teams are often dispersed across many places and must collaborate effectively despite geographical, cultural, and temporal disparities. Kumar & Kumar (2011) assert that this ontology-based methodology offers a systematic framework for segmenting requirements engineering challenges into several tiers, facilitating enhanced categorisation and prioritisation of activities. The framework methodically organises these difficulties, enhancing communication among stakeholders and ensuring that essential needs are not neglected. This approach improves traceability and consistency throughout the project lifecycle, facilitating the monitoring of changes and ensuring alignment with project goals. The framework aids organisations in enhancing requirements management methods by tackling the challenges inherent in global software development environments. A second ontology has been established expressly for modelling software needs, in addition to the management framework. This ontology is essential for comprehending and organising ideas across many areas in Information Systems & Technology. The major emphasis is on facilitating the requirement analysis phase, the first stage in the software engineering lifecycle. In this phase, essential requirements are collected, recorded, and analysed to provide the basis of the software development process. The ontology facilitates the development of precise and thorough models, including diagrams that illustrate both functional and non-functional needs. These models are crucial for guaranteeing that all stakeholders, including customers, developers, and project managers, possess a unified comprehension of the project's scope and goals. Furthermore, the ontology's applicability beyond fundamental requirement analysis. It offers a structured framework for delineating links among ideas, facilitating enhanced knowledge integration across various systems and technologies. This feature is particularly crucial in transdisciplinary projects where needs may include several fields. The ontology may reconcile disparities between technical and business viewpoints, guaranteeing that software solutions satisfy both user requirements and technological limitations. The ontology enhances the building of precise and comprehensive models, so reducing the danger of misunderstanding and aiding in the early identification of possible concerns. This proactive strategy results in superior software solutions that more effectively correspond with user expectations and organisational objectives. Both the ontology-based knowledge management approach and the software requirements modelling ontology substantially enhance the progression of requirements engineering methods. They provide methodical answers to prevalent difficulties, augment cooperation, and boost the overall efficiency and efficacy of global software development initiatives. By using these technologies, organisations may manage the intricacies of contemporary software development and provide solutions that satisfy various stakeholder requirements.

Another study by Innab, Kayed, & Sajeev (2012) established a comprehensive framework aimed at identifying the key principles commonly used in various modeling diagrams. These diagrams are crucial instruments in software development, since they provide visual representations of systems, processes, and interactions, enhancing comprehension and communication among developers and stakeholders.

The research examined several diagramming notations, as articulated by several studies by Eng, Bracewell, & Clarkson (2009), Sousa, Vanderdonckt, Henderson-Sellers, & Gonzalez-Perez (2012), Sundaramoorthy (2022), and Ye et al. (2020), including specific diagrams such as activity diagrams and modeling languages like UML. UML is esteemed in software engineering for its capacity to depict several facets of a system, encompassing its structure, behaviour, and interactions, rendering it an essential instrument for requirements analysis and system design. The use of ontology into these diagramming methods introduces a semantic dimension that improves their accuracy and interoperability. Ontology offers a systematic framework for delineating connections and qualities among entities shown in diagrams, therefore mitigating ambiguities and enhancing consistency among models. By using ontology, activity diagrams may more effectively encapsulate the semantics of activities, processes, and decision points,

whilst UML diagrams can gain from enhanced metadata that connects software components with their conceptual meanings. Moreover, the use of ontology in software development methodologies such as XP and Scrum has shown considerable potential in tackling issues like software work estimate and knowledge management. Adnan, Afzal, & Asif (2019) emphasized the capacity of the ontology model for utilizing existing knowledge bases through Hermit reasoning—a description logic reasoning engine—to provide precise and logical assessments based on defined criteria. This procedure utilises established logical principles and connections inside the ontology to assess needs and provide estimates that correspond with the project's goals. Utilising Hermit reasoning, development teams may detect discrepancies, verify requirement dependencies, and enhance estimates with more efficacy. Likewise, the Scrum methodology derives advantages from ontology-driven techniques. Adnan & Afzal (2017) shown that ontology models may substantially improve the precision of effort estimate and optimise knowledge management operations. Integrating ontologies into Scrum processes enables teams to establish semantic repositories that systematically collect and organise information acquired during development. These repositories enhance traceability, support decision-making, and guarantee the retention of lessons learnt for future iterations or projects. The organised framework of ontologies aids teams in managing the iterative and incremental aspects of Scrum, whereby needs regularly shift. The integration of diagramming notations, ontology models, and agile approaches such as XP and Scrum provides a comprehensive framework for enhancing several facets of software development. These methodologies augment communication, promote enhanced cooperation among stakeholders, and refine the precision of estimations and information management. Integrating semantic reasoning with ontology-driven insights enables developers to get enhanced consistency, flexibility, and efficiency in managing intricate software projects.

The third topic pertains to the Ontology of Software Requirements, which is essential for guaranteeing the accuracy and dependability of the requirements engineering process. In contrast to conventional techniques that often need specialised knowledge in logic or simulation, ontology-based approaches provide more intuitive use without requiring extensive technical skills. This clarity allows stakeholders, including those with non-technical backgrounds, to engage successfully in the requirements formulation process. Moreover, the use of ontology enhances the identification of structural and behavioural flaws in the design process. Errors may be rapidly discovered and rectified by reasoning procedures inside ontological frameworks, hence minimising the need for expensive rework later in the development cycle (Chen, Chen, Liu, & Ye, 2020). In agile development, ontology is especially beneficial because of its flexibility and capacity to manage evolving needs. Ontologies developed from engineering requirements improve the quality and management of those needs by offering a systematic description of ideas and their interrelations. This organised framework guarantees that requirements remain systematic, coherent, and traceable throughout the development process (Murtazina & Avdeenko, 2019). Furthermore, using ontology as a modelling instrument facilitates the verification of the completeness and consistency of requirements, ensuring that no essential components are neglected and that the needs are congruent with the overarching system goals. An important benefit of ontology-based methodologies is their capacity to automate certain elements of requirements engineering. Client-specified criteria can be automatically converted into service descriptions in Web Ontology Language for Services (OWL-S), which are then used to generate customized software systems (Yuan & Zhang, 2015). This degree of automation not only optimises the development process but also reduces mistakes caused by human interpretation, hence improving the efficiency and precision of the system design. The development of ontologies offers several advantages within the fields of software engineering and management. These include enhanced knowledge collection and dissemination, since ontologies serve as repositories for domain-specific information that may be used and modified for subsequent initiatives. The organised framework of ontologies allows teams to access, understand, and use essential information more efficiently, promoting cooperation and informed decision-making across the software development lifecycle (Alsanad, Chikh, & Mirza, 2019). Moreover, the use of ontologies facilitates the standardisation of requirements procedures, hence enhancing alignment across various teams and stakeholders, particularly in intricate or decentralised development contexts. The ontology of software requirements provides a comprehensive and adaptable framework for enhancing the quality, consistency, and automation of requirements engineering. Ontology-based approaches substantially improve the efficiency and efficacy of software development processes by tackling difficulties such as error detection, dynamic requirement changes, and knowledge sharing.

The ontology-based verification approach we have developed has a good impact on identifying incorrectness and incompleteness in SRS (Davis, 1990; Haris, Kurniawan, & Ramdani, 2020; Mustaffa, Sallim, & Mohamed, 2021). Nevertheless, the efficacy of the ontology-based verification approach is still

restricted because of mistakes that include ambiguity and inconsistency. This is for automatic detection and updating of software requirements. This ontology captures the domain-specific knowledge about the issue. For this reason, a domain ontology is developed to describe all the product's needs, goals, and objectives. The creation of the SRS ontology is the second phase in the process (Dzung & Ohnishi, 2013). This ontology is a comprehensive expansion of the domain ontology, and it is in accordance with the standard established by the IEEE. The design document, sometimes referred to as the Software Design Document (SDD), is created during the third stage. Traditional techniques, such as UML, will use standardized models that ensure consistency and clarity in the design. The actual implementation of the source code and the fulfillment of all of the criteria described in the SR's ontology are components of the fourth stage. During this stage, one is responsible for the development of both the front end and the back end of the program. Developing a cross-reference ontology and source code listings is the fifth phase in the process. According to the source code listings ontology, all of the particular entities that are a part of the source code are listed down, and the responsibilities each of these things plays are defined. The cross-reference listing ontology is responsible for defining the link between the many different software requirements that are interconnected (Bhatia, Kumar, Beniwal, & Malik, 2020).

4.3. RQ2: What are the challenges of applying ontology requirements to software development methodology?

At the beginning of the ontology assessment process, the most difficult part is figuring out which quality aspects will be assessed and which evaluation technique will be used. The quality aspects to assess depend on various elements, such as the kind of ontology, the emphasis of an evaluation, and who is doing the evaluation. The literature has examined various evaluation criteria across a wide range of topics (Tan, Ismail, Tarasov, Adlemo, & Johansson, 2016). These standards are often set down in a document based on paper. Model View Descriptor (MVD) developers engage in the process of manually translating the requirements included within the Information Data Management (IDM) document in accordance with the IFC schema. The materials that have been translated are written in documents that are either paper-based or electronic. The criteria for an IDM and an MVD are the same; however, they are stated in various forms and documents according to their respective needs. To alleviate these challenges, this paper proposes an ontology-based approach to developing an IDM (Jiang & Tiwari, 2022; Ye, 2023) and a MVD (Jiang, et al., 2021; Kop, 2012). The following results are anticipated from this approach: The following will enable software suppliers and MVD developers to determine if a defined MVD satisfies all IDM requirements: Data verification and specifications for the correctness and consistency of IDM information and model views; (2) high-level information representations that can be converted from model data to knowledge; and (3) a robust knowledge framework that aids in capturing the relevant semantics that software developers are familiar with (Lee, Eastman, & Solihin, 2016).

Numerous issues, most of which are associated with requirements change management, are demanding tasks that call for extensive communication among all parties involved. In addition, as software engineering continued to expand, the projects for which it was responsible got more complicated and demanded software of a higher quality. All of these factors promoted the development of an ontology for requirements change management in global software development. Increasing the efficiency of the Requirements Change Management process may be accomplished by enhancing the coordination, communication, and control among stakeholders. This ontology, which can serve as a foundation for other methodologies, can be utilized to accomplish this. Specifically, the ontology may be used with other ontologies as a software application to guarantee that the modification request is semantically accurate. Requirements Change Management in Global Software Development is the subject of this study, which tries to construct a domain ontology to manage requirements changes (Alsanad, Chikh, & Mirza, 2019).

The success metrics for adopting ontology in software requirements engineering concentrate on many critical areas. This include maintaining uniformity in requirement documentation, enhancing communication efficacy between development teams and stakeholders, and facilitating the early identification of mistakes or discrepancies throughout the development process. Metrics may include a reduction in discrepancies within requirement specifications, a decrease in documentation completion time, and an enhancement in stakeholder satisfaction, which may be assessed by surveys or interviews.

Formulating these measures requires a methodical methodology. The first phase involves ascertaining the organization's requirements and objectives about the ontology implementation. This involves involving stakeholders, including developers, project managers, and end-users, to identify the essential parts for assessment. Upon identification, key performance indicators (KPIs) may be chosen to provide quantifiable and actionable data. Preliminary evaluation of the suggested measures in a limited pro-

ject confirms their significance and applicability prior to broader implementation.

Nonetheless, significant obstacles exist in the formulation and execution of these measurements. The intricacy of ontology creation might hinder the establishment of complete measurements, since extremely sophisticated ontologies may not be entirely enclosed. Moreover, constraints on resources, like the time and effort necessary for data gathering and processing, might impede the process. Resistance to change is a possible impediment, since stakeholders may exhibit reluctance to embrace new procedures or approaches prompted by ontology adoption. Effectively tackling these difficulties requires meticulous preparation and communication to guarantee the proper implementation of the measures.

5. Discussion

An important aspect highlighted in the analysis of our 20 selected studies is the geographic locations of the authors. From the geographic distribution of the studies, we selected they were directly from different continents, but we did not find any studies from Australia. For those from Asia, there are several countries, such as China, Pakistan, Saudi Arabia, Singapore, Japan, and Thailand, while for America, there are Mexico, Colombia, and the USA. For Europe, we found Sweden, Russia, and Italy, and from Africa, there was Egypt. because the distribution is even. It is difficult to predict the similarity of outcomes due to differences in organizational culture, country-specific cultures, and social norms across organizations globally.

We found a lot of ontology use in requirements related to application development but did not specifically discuss the method used. of several software development methods, most of the studies discuss the application of the Agile method, while other methods are difficult to find, especially the older application development methods. Most of the 20 studies we selected discussed the agile method. however, it is very difficult to find steps to use this method in relation to the ontology requirement.

Furthermore, we have identified the current ontology requirements for software development methods in response to RQ1, which describes several steps in the development of an ontology and its application in various software development methods such as Agile, Scrum, and XP. Of the few studies, only a few found the use of ontology orientation with Scrum and XP on e-commerce systems, and the rest only talked about methodology.

In response to RQ2, we have determined the ontology-related problems. The ontology provided in this work was created to reflect the software needs of an embedded system used in the aviation sector. Software requirements papers served as the foundation for the ontology. We provide a technique that integrates elements from many ontology development approaches from the standpoint of ontology development. The combined approach is very useful and offers the direction required to create a requirements ontology from requirements papers for a specific use case.

The reviewed studies emphasize the critical role of ontology in enhancing various aspects of software development methodologies. Ontologies provide a structured approach to knowledge representation, facilitating effective communication and collaboration among stakeholders. This is particularly evident in agile methodologies, where the frequent evolution of requirements poses challenges to maintaining consistency and alignment across development cycles. The use of ontology in methods like Scrum and XP enables better traceability and integration of requirements, ensuring that changes are effectively managed and validated. Additionally, ontology-driven approaches in requirement engineering have demonstrated significant improvements in knowledge sharing and error detection during early stages, such as design and specification. Despite these benefits, challenges persist, including the high cost and complexity of ontology development and the need for standardized practices across diverse software development contexts.

The difficulties in implementing ontology requirements in software development approaches are complex and need thorough examination to provide practical insights. A significant problem is in the intricacy of ontology development and integration. Creating an ontology that precisely reflects domain-specific needs requires careful work, including domain knowledge gathering, formalisation, and validation. The complexity increases when trying to achieve semantic interoperability across diverse systems, which is a primary objective in agile and global software development environments.

A notable difficulty is the elevated expense and resource demands linked to ontology development. Developing ontologies requires cooperation among domain specialists, software developers, and knowledge engineers, which may be time-consuming and costly. Furthermore, the lack of standardised procedures for ontology building exacerbates this issue, since practitioners often depend on custom or inadequately specified frameworks, resulting in variable outcomes across projects.

Methodological deficiencies in ontology engineering must be rectified regarding current methodolo-

gies. Techniques such as METHONTOLOGY, Ontology 101, and Test-Driven Ontology Development provide basic methodologies but often exhibit limitations in scalability and flexibility within dynamic software development contexts such as Agile and Scrum. These approaches prioritise iterative and modular development; yet, their effectiveness in situations with quickly altering needs, typical of agile processes, is constrained. Integrating ontology engineering with agile methodologies necessitates more structured processes, including the incorporation of ontology validation into routine sprint reviews and the use of automated reasoning tools to facilitate real-time requirement modifications.

Moreover, prior research has emphasised the difficulty of sustaining ontology quality over time. Ontologies, especially those used in software requirements, must adapt in accordance with shifts in domain knowledge and stakeholder demands. Nonetheless, maintaining consistency and preventing semantic drift during this progression is a continual struggle. Contemporary tools and frameworks, such Protégé and OWL, provide essential assistance for ontology management but are deficient in effective procedures for automated quality assurance and change management in dynamic settings.

Ultimately, the use of ontology in particular software development methodologies remains inadequately examined. Although several studies have examined Agile, Scrum, and XP, there is a paucity of study on earlier or less prevalent techniques, such as Waterfall or Spiral, and the potential of ontology to augment these approaches. An in-depth examination of these methodologies and their incorporation with ontology may uncover novel prospects for enhancing requirement engineering in various development environments. Future research must tackle these deficiencies by emphasising the automation of ontology creation, standardising processes, and improving tool support for ontology administration. Integrating machine learning and natural language processing methods may facilitate the automation of ontology construction from textual requirements, thereby decreasing costs and enhancing scalability.

5.1. Implication

This review has several implications for researchers and practitioners. In terms of research, more empirical studies that use the ontology in requirements approach are needed in software development. Some have described some of the uses of ontology in various software developments such as Agile, scrum, and XP. However, it needs to be more in-depth and more specific about the approach and practice in large-scale software development. In addition, an evaluation of the ontology in terms of usability and application needs to be conducted, both from the user and application points of view. In order to lower the cost of software development, the ontology generated from the software requirements papers is meant to facilitate the automation of various jobs in later phases of the process. Nevertheless, creating an ontology is a costly and challenging undertaking. Therefore, automating the process of creating ontologies from software requirements papers is another avenue for our future research. Therefore, the actual cost of software development operations may be greatly decreased when the ontology-based strategy for software engineering is used in an industrial setting.

5.2. Limitation

Bias in study selection and potential imprecision in data extraction from varied sources are the fundamental constraints of every systematic review. When creating our research plan, we took the following actions to get rid of this bias and guarantee accuracy and precision in the selection of studies. The first is using keywords to find references that match the needs of the research question. Combining words from shared keywords must also be considered so that search engines precisely direct the information needed. Our assessment is not too broad; it only concerns several application development methods such as agile, scrum, and XP. Of the three methods, we get more of the agile method because most of them discuss ontology in agile. Then, the 20 studies we used were very varied and slightly biased from the topics we determined, so the results of the learning process were accurate with the research questions we submitted. Furthermore, we acknowledge that we have not provided a thorough explanation of one of the articles that explains the difficulties since the research we read show various things, and it is challenging for us to articulate these issues because the authors do not specifically disregard them in their investigations.

6. Conclusions

A thorough analysis of the literature on the difficulties and practices of ontology requirements in software development techniques is presented in this study. Through a multi-stage screening procedure with independent confirmation at each level, 20 pertinent publications were retrieved from the 71 original papers in reputable electronic research databases. Then, all of our problems are assessed and grouped into ontology in requirements for software development methods and ontology challenges in software development methods. We have summarized several application development methods that apply ontology to requirements. Our review of ontology in requirement shows that there are a variety of methods that can

be used in software development. The application development method commonly has several iterations to get the ontology, such as specification, conceptualization, formalization, evaluation, and maintenance. In terms of challenges in ontology, there are many criteria for evaluation, such as the quality of evaluation depending on various factors such as type of ontology and focus of evaluation.

This study highlights the growing importance of ontology in addressing challenges within software requirement engineering and development methodologies. Ontology provides a structured framework that enhances consistency, traceability, and knowledge sharing, making it a valuable tool in both traditional and agile methodologies. Through its application in domains such as requirement modeling, validation, and management, ontology has demonstrated its ability to improve the accuracy and completeness of requirements while reducing ambiguities. However, challenges such as the complexity of ontology development, lack of standardization, and scalability issues remain significant barriers to widespread adoption. Future research should focus on streamlining ontology creation through automation and establishing standardized guidelines to ensure interoperability and usability across diverse software engineering environments. By addressing these gaps, the potential of ontology to revolutionize software development practices can be fully realized, paving the way for more efficient and reliable systems.

7. CRediT Authorship Contribution Statement

Reza Fauzan: Conceptualization, Investigation, Project administration, Resources, Validation, Visualization, Writing – original draft, and Writing – review & editing. **Mohammad Zaenuddin Hamidi:** Data curation, Formal Analysis, and Funding acquisition. **Winda Ayu Safitri:** Writing – original draft, and Writing – review & editing. **Daniel Oranova Siahaan:** Conceptualization, Supervision, Validation, and Writing – review & editing. **Muhammad Ihsan Karimi:** Writing – review & editing.

8. Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

9. References

Abdelghany, A. S., Darwish, N. R., & Hefni, H. A. (2019). An Agile Methodology for Ontology Development. *International Journal of Intelligent Engineering and Systems, 12*(2), 170-181. doi:https://doi.org/10.22266/ijies2019.0430.17

Abioye, T. E., Arogundade, O. T., Misra, S., Akinwale, A. T., & Adeniran, O. J. (2020). Toward ontology-based risk management framework for software projects: An empirical study. *Journal of Software: Evolution and Process, 32*(12). doi:https://doi.org/10.1002/smr.2269

Adnan, M., & Afzal, M. (2017). Ontology Based Multiagent Effort Estimation System for Scrum Agile Method. *IEEE Access, 5*, 25993-26005. doi:https://doi.org/10.1109/ACCESS.2017.2771257

Adnan, M., Afzal, M., & Asif, K. H. (2019). Ontology-Oriented Software Effort Estimation System for E-commerce Applications Based on Extreme Programming and Scrum Methodologies. *The Computer Journal, 62*(11), 1605–1624. doi:https://doi.org/10.1093/comjnl/bxy141

Alrumaih, H., Mirza, A., & Alsalamah, H. (2020). Domain Ontology for Requirements Classification in Requirements Engineering Context. *IEEE Access, 8*, 89899-89908. doi:https://doi.org/10.1109/ACCESS.2020.2993838

Alsanad, A. A., Chikh, A., & Mirza, A. (2019). A Domain Ontology for Software Requirements Change Management in Global Software Development Environment. *IEEE Access, 7*, 49352-49361. doi:https://doi.org/10.1109/ACCESS.2019.2909839

Beck, K. (2000). *Extreme Programming Explained: Embrace Change.* Addison-Wesley.

Bhatia, M. P., Kumar, A., Beniwal, R., & Malik, T. (2020). Ontology driven software development for automatic detection and updation of software requirement specifications. *Journal of Discrete Mathematical Sciences and Cryptography, 23*(1), 197-208. doi:https://doi.org/10.1080/09720529.2020.1721884

Biagetti, M. T. (2021). Ontologies as knowledge organization systems. (B. H. Gnoli, Ed.) *Knowledge Organization, 48*(2), 152-176. Retrieved from Knowledge Organization: https://www.isko.org/cyclo/ontologies

Bichier, M., & Lin, K.-J. (2006). Service-oriented computing. *Computer, 39*(3), 99-101. doi:https://doi.org/10.1109/MC.2006.102

Cao, L., & Ramesh, B. (2008). Agile Requirements Engineering Practices: An Empirical Study. *IEEE Software, 25*(1), 60-67. doi:https://doi.org/10.1109/MS.2008.1

Chaccour, C., Saad, W., Debbah, M., Han, Z., & Poor, H. V. (2024). Less Data, More Knowledge: Building Next Generation Semantic Communication Networks. *IEEE Communications Surveys & Tutorials*. doi:https://doi.org/10.1109/COMST.2024.3412852

Chen, R., Chen, C.-H., Liu, Y., & Ye, X. (2020). Ontology-based requirement verification for complex systems. *Advanced Engineering Informatics, 46*. doi:https://doi.org/10.1016/j.aei.2020.101148

Corcho, O., Fernández-López, M., Gómez-Pérez, A., & López-Cima, A. (2005). Building Legal Ontologies with METHONTOLOGY and WebODE. In L. N. Science, *Law and the Semantic Web* (Vol. 3369, pp. 142–157). Berlin, Heidelberg: Springer. doi:https://doi.org/10.1007/978-3-540-32253-5_9

Cristani, M., & Cuel, R. (2005). A Survey on Ontology Creation Methodologies. *International Journal on Semantic Web and Information Systems (IJSWIS), 1*(2), 49-69. doi:https://doi.org/10.4018/jswis.2005040103

Davis, A. M. (1990). *Software requirements: analysis and specification.* United States: Prentice Hall Press. Retrieved from https://dl.acm.org/doi/abs/10.5555/78225

De Nicola, A., & Villani, M. L. (2021). Smart City Ontologies and Their Applications: A Systematic Literature Review. *Sustainability, 13*(10). doi:https://doi.org/10.3390/su13105578

Dzung, D. V., & Ohnishi, A. (2013). Evaluation of Ontology-Based Checking of Software Requirements Specification. *2013 IEEE 37th Annual Computer Software and Applications Conference. 37*, pp. 425-430. Kyoto, Japan: IEEE. doi:https://doi.org/10.1109/COMPSAC.2013.70

Eng, N. L., Bracewell, R. H., & Clarkson, P. J. (2009). Concept Diagramming Software for Engineering Design Support: A Review and Synthesis of Studies. *ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, (pp. 1221-1234). San Diego, California, USA. doi:https://doi.org/10.1115/DETC2009-86840

Farghaly, K., Soman, R. K., & Zhou, S. A. (2023). The evolution of ontology in AEC: A two-decade synthesis, application domains, and future directions. *Journal of Industrial Information Integration, 36*. doi:https://doi.org/10.1016/j.jii.2023.100519

Fauzan, R., Siahaan, D., Rochimah, S., & Triandini, E. (2018). Class Diagram Similarity Measurement: A Different Approach. *2018 3rd International Conference on Information Technology, Information System and Electrical Engineering (ICITISEE). 3*, pp. 215-219. Yogyakarta, Indonesia: IEEE. doi:https://doi.org/10.1109/ICITISEE.2018.8721021

Fauzan, R., Siahaan, D., Solekhah, M., Saputra, V. W., Bagaskara, A. E., & Karimi, M. I. (2023). A Systematic Literature Review of Student Assessment Framework in Software Engineering Courses. *Journal of Information Systems Engineering and Business Intelligence, 9*(2), 264-275. doi:https://doi.org/10.20473/jisebi.9.2.264-275

Fernández-López, M., Gómez-Pérez, A., & Juristo Juzgado, N. (1997). METHONTOLOGY: From Ontological Art Towards Ontological Engineering. *Proceedings of the Ontological Engineering AAAI-97 Spring Symposium Series* (pp. 33-40). Stanford, California, United States: AAAI. Retrieved from https://oa.upm.es/5484/?trk=public_post_main-feed-card-text

Fraga, A. L., Vegetti, M., & Leone, H. P. (2020). Ontology-based solutions for interoperability among product lifecycle management systems: A systematic literature review. *Journal of Industrial Information Integration, 20*. doi:https://doi.org/10.1016/j.jii.2020.100176

Fu, C., Jiang, H., & Chen, X. (2022). RETRACTED ARTICLE: Modeling of an Enterprise Knowledge Management System Based on Artificial Intelligence. *Knowledge Management Research & Practice*, 1–13. doi:https://doi.org/10.1080/14778238.2020.1854632

Guizzardi, G. (2005). *Ontological foundations for structural conceptual models.* Enschede, Netherlands: University of Twente. Retrieved from https://research.utwente.nl/en/publications/ontological-foundations-for-structural-conceptual-models/

Guizzardi, G., & Guarino, N. (2024). Explanation, semantics, and ontology. *Data & Knowledge Engineering, 153*. doi:https://doi.org/10.1016/j.datak.2024.102325

Guizzardi, G., Halpin, T., & Halpin, T. (2008). Ontological foundations for conceptual modelling. *Applied Ontology, 3*(1-2), 1-12. doi:https://doi.org/10.3233/AO-2008-0049

Guyatt, G. H., & Rennie, D. (1993). Users' Guides to the Medical Literature. *JAMA, 270*(17), 2096–2097. doi:https://doi.org/10.1001/jama.1993.03510170086037

Guyatt, G., Rennie, D., Meade, M., & Cook, D. (2014). *Users' Guides to the Medical Literature: Essentials of Evidence-Based Clinical Practice.* McGraw Hill Education.

Haris, M. S., Kurniawan, T. A., & Ramdani, F. (2020). Automated Features Extraction from Software Requirements Specification (SRS) Documents as The Basis of Software Product Line (SPL) Engineering. *JITeCS (Journal of Information Technology and Computer Science), 5*(3), 279-292. doi:https://doi.org/10.25126/jitecs.202053219

Helmy, Y. M., Abdelgaber, S., Fahmy, H., & Montasser, H. S. (2020). A conceptual ontological framework for managing the social business process to enhance customer experience. *Knowledge and Process Management, 27*(4), 262-271. doi:https://doi.org/10.1002/kpm.1652

Hlomani, H., & Stacey, D. (2014). Approaches, methods, metrics, measures, and subjectivity in ontology evaluation: A survey. *Undefined, 1*, 1–5. Retrieved from https://semantic-web-journal.net/system/files/swj657.pdf

Husáková, M., & Bureš, V. (2020). Formal Ontologies in Information Systems Development: A Systematic Review. *Information, 11*(2). doi:https://doi.org/10.3390/info11020066

Inayat, I., Salim, S. S., Marczak, S., Daneva, M., & Shamshirband, S. (2015). A systematic literature review on agile requirements engineering practices and challenges. *Computers in Human Behavior, 51*, 915-929. doi:https://doi.org/10.1016/j.chb.2014.10.046

Innab, N., Kayed, A., & Sajeev, A. S. (2012). An ontology for software requirements modelling. *2012 IEEE International Conference on Information Science and Technology* (pp. 485-490). Wuhan, China: IEEE. doi:https://doi.org/10.1109/ICIST.2012.6221694

Iqbal, R., Murad, M. A., Mustapha, A., & Sharef, N. M. (2013). An Analysis of Ontology Engineering Methodologies: A Literature Review. *Research Journal of Applied Sciences, Engineering and Technology, 6*(16), 2993-3000. Retrieved from https://pdfs.semanticscholar.org/c017/bfc3d6c2b7fb3f6d2042b6cd483a63efce87.pdf

Jiang, D., Wu, Z., Hsie, C.-Y., Chen, G., Liao, B., Wang, Z., . . . Hou, T. (2021). Could graph neural networks learn better molecular representation for drug discovery? A comparison study of descriptor-based and graph-based models. *Journal of Cheminformatics, 13*(12). doi:https://doi.org/10.1186/s13321-020-00479-8

Jiang, N., & Tiwari, R. S. (2022). The Design of Mental Health Information Data Management System Under the Background of Informationization. *Cyber Security Intelligence and Analytics (CSIA 2022). 123*, pp. 303–310. Cham: Springer. doi:https://doi.org/10.1007/978-3-030-96908-0_38

Jing, X., Min, H., Gong, Y., Sittig, D. F., Biondich, P., Robinson, D., . . . Gimbel, R. (2022, May 11). *medRxiv.* doi:https://doi.org/10.1101/2022.05.11.22274984

Johnson, L. A. (1998). DO-178B, "Software Considerations in Airborne. *Crosstalk*, (pp. 11-20). Retrieved from http://www.dcs.gla.ac.uk/~johnson/teaching/safety/reports/schad.html

Kendall, E. F., & McGuinness, D. L. (2019). *Ontology Engineering: Synthesis Lectures on the Semantic Web: Theory and Technology.* Morgan & Claypool.

Kitchenham, B., Charters, S., Budgen, D., Brereton, P., Turner, M., Linkman, S., . . . Visaggio, G. (2007). *Guidelines for performing Systematic Literature Reviews in Software Engineering.* Durham, UK & Staffordshire, UK: Keele University & University of Durham.

Kitchenham, B., Pretorius, R., Budgen, D., Brereton, O. P., Turner, M., Niazi, M., & Linkman, S. (2010). Systematic literature reviews in software engineering – A tertiary study. *Information and Software Technology, 52*(8), 792-805. doi:https://doi.org/10.1016/j.infsof.2010.03.006

Kop, C. (2012). Checking feasible completeness of domain models with natural language queries. *APCCM '12: Proceedings of the Eighth Asia-Pacific Conference on Conceptual Modelling. 130*, pp. 33 - 42. Melbourne, Australia: Australian Computer Society. Retrieved from https://dl.acm.org/doi/abs/10.5555/2523782.2523787

Kumar, S. A., & Kumar, T. A. (2011). Study the impact of requirements management characteristics in global software development projects: an ontology based approach. *International Journal of Software Engineering & Applications (IJSEA), 2*(4). doi:https://doi.org/10.5121/ijsea.2011.2410

Lee, Y.-C., Eastman, C. M., & Solihin, W. (2016). An ontology-based approach for developing data exchange requirements and model views of building information modeling. *Advanced Engineering Informatics, 30*(3), 354-367. doi:https://doi.org/10.1016/j.aei.2016.04.008

Mukhopadhyay, A., & Ameri, F. (2016). An ontological approach to engineering requirement representation and analysis. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 30*(4), 337-352. doi:https://doi.org/10.1017/S0890060416000330

Murtazina, M. S., & Avdeenko, T. V. (2018). Ontology-Based Approach to the Requirements Engineering in Agile Environment. *2018 XIV International Scientific-Technical Conference on Actual Problems of*

Electronics Instrument Engineering (APEIE). 14, pp. 496-501. Novosibirsk, Russia: IEEE. doi:https://doi.org/10.1109/APEIE.2018.8546144

Murtazina, M., & Avdeenko, T. (2019). An Ontology-Based Approach to the Agile Requirements Engineering. *Perspectives of System Informatics. 11964*, pp. 205–213. Cham: Springer. doi:https://doi.org/10.1007/978-3-030-37487-7_17

Mustaffa, S. N., Sallim, J. B., & Mohamed, R. B. (2021). Semi – Automated Software Requirement Specification (SRS) Document Generator: The Guideline to Novice System Analyst. *2021 International Conference on Software Engineering & Computer Systems and 4th International Conference on Computational Science and Information Management (ICSECS-ICOCSIM)* (pp. 80-85). Pekan, Malaysia: IEEE. doi:https://doi.org/10.1109/ICSECS52883.2021.00022

Olan, F., Arakpogun, E. O., Suklan, J., Nakpodia, F., Damij, N., & Jayawickrama, U. (2022). Artificial intelligence and knowledge sharing: Contributing factors to organizational performance. *Journal of Business Research, 145*, 605-615. doi:https://doi.org/10.1016/j.jbusres.2022.03.008

Ortega-Ordoñez, W. A., Pardo-Calvache, C. J., & Pino-Correa, F. J. (2019). OntoAgile: an ontology for agile software development processes. *Dyna, 86*(209), 79-90. doi:http://doi.org/10.15446/dyna.v86n209.76670

Osman, M. A., Noah, S. A., & Saad, S. (2022). Ontology-Based Knowledge Management Tools for Knowledge Sharing in Organization—A Review. *IEEE Access, 10*, 43267-43283. doi:https://doi.org/10.1109/ACCESS.2022.3163758

Peroni, S. (2017). A Simplified Agile Methodology for Ontology Development. *OWL: Experiences and Directions – Reasoner Evaluation (OWLED 2016, ORE 2016)* (pp. 55–69). Cham: Springer. doi:https://doi.org/10.1007/978-3-319-54627-8_5

Pileggi, S. F. (2021). Knowledge interoperability and re-use in Empathy Mapping: an ontological approach. *Expert Systems with Applications, 180*. doi:https://doi.org/10.1016/j.eswa.2021.115065

Pliatsios, A., Kotis, K., & Goumopoulos, C. (2023). A systematic review on semantic interoperability in the IoE-enabled smart cities. *Internet of Things, 22*. doi:https://doi.org/10.1016/j.iot.2023.100754

Poveda-Villalón, M., Fernández-Izquierdo, A., Fernández-López, M., & García-Castro, R. (2022). LOT: An industrial oriented ontology engineering framework. *Engineering Applications of Artificial Intelligence, 111*. doi:https://doi.org/10.1016/j.engappai.2022.104755

Said, A., Zhao, Y., Derr, T., Shabbir, M., Abbas, W., & Koutsoukos, X. (2023, Aug 23). *A Survey of Graph Unlearning.* doi:https://doi.org/10.48550/arXiv.2310.02164

Schön, E.-M., Thomaschewski, J., & Escalona, M. J. (2017). Agile Requirements Engineering: A systematic literature review. *Computer Standards & Interfaces, 49*, 79-91. doi:https://doi.org/10.1016/j.csi.2016.08.011

Schwaber, K. (2004). *Agile Project Management with Scrum.* Microsoft Press.

Shahzad, B., Javed, I., Shaikh, A., Sulaiman, A., Abro, A., & Memon, M. A. (2021). Reliable Requirements Engineering Practices for COVID-19 Using Blockchain. *Sustainability, 13*(12). doi:https://doi.org/10.3390/su13126748

Shrivastava, A., Jaggi, I., Katoch, N., Gupta, D., & Gupta, S. (2021). A Systematic Review on Extreme Programming. *Journal of Physics: Conference Series, 1969*. doi:https://doi.org/10.1088/1742-6596/1969/1/012046

Sitthithanasakul, S., & Choosri, N. (2016). Using ontology to enhance requirement engineering in agile software process. *10th International Conference on Software, Knowledge, Information Management & Applications (SKIMA). 10*, pp. 181-186. Chengdu, China: IEEE. doi:https://doi.org/10.1109/SKIMA.2016.7916218

Smirnov, A., Levashova, T., Ponomarev, A., & Shilov, N. (2021). Methodology for Multi-Aspect Ontology Development: Ontology for Decision Support Based on Human-Machine Collective Intelligence. *IEEE Access, 9*, 135167-135185. doi:https://doi.org/10.1109/ACCESS.2021.3116870

Sousa, K., Vanderdonckt, J., Henderson-Sellers, B., & Gonzalez-Perez, C. (2012). Evaluating a graphical notation for modelling software development methodologies. *Journal of Visual Languages & Computing, 23*(4), 195-212. doi:https://doi.org/10.1016/j.jvlc.2012.04.001

Srivastava, A., Bhardwaj, S., & Saraswat, S. (2017). SCRUM model for agile methodology. *2017 International Conference on Computing, Communication and Automation (ICCCA)* (pp. 864-869). Greater Noida, India: IEEE. doi:https://doi.org/10.1109/CCAA.2017.8229928

Sundaramoorthy, S. (2022). *UML Diagramming: A Case Study Approach* (1st ed ed.). Auerbach Publications. doi:https://doi.org/10.1201/9781003287124

Tan, H., Adlemo, A., Tarasov, V., & Johansson, M. E. (2017). Evaluation of an Application Ontology. *Proceedings of the Joint Ontology Workshops 2017 Episode 3: The Tyrolean Autumn of Ontology.* Bozen-Bolzano, Italy: DiVA. Retrieved from https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1185058&dswid=9486

Tan, H., Ismail, M., Tarasov, V., Adlemo, A., & Johansson, M. (2016). Development and Evaluation of a Software Requirements Ontology. *Proceedings of the 7th International Workshop on Software Knowledge (IC3K 2016) - SKY. 1*, pp. 11-18. Porto, Portugal: SciTePress. doi:https://doi.org/10.5220/0006079300110018

Triandini, E., Fauzan, R., Siahaan, D. O., Rochimah, S., Suardika, I. G., & Karolita, D. (2022). Software similarity measurements using UML diagrams: A systematic literature review. *Register: Jurnal Ilmiah Teknologi Sistem Informasi, 8*(1), 10-23. doi:https://doi.org/10.26594/register.v8i1.2248

Tsai, W.-T., Wu, B., Jin, Z., Huang, Y., & Li, W. (2013). Ontology patterns for service-oriented software development. *Software: Practice and Experience, 43*(7), 867-883. doi:https://doi.org/10.1002/spe.1132

Tudorache, T. (2020). Ontology engineering: Current state, challenges, and future directions. *Semantic Web, 11*(1), 125-138. doi:https://doi.org/10.3233/SW-190382

Wang, J., Mendori, T., & Xiong, J. (2014). A language learning support system using course-centered ontology and its evaluation. *Computers & Education, 78*, 278-293. doi:https://doi.org/10.1016/j.compedu.2014.06.009

Yangui, S., Goscinski, A., Drira, K., Tari, Z., & Benslimane, D. (2021). Future generation of service-oriented computing systems. *Future Generation Computer Systems, 118*, 252-256. doi:https://doi.org/10.1016/j.future.2021.01.019

Ye, K., Ni, W., Krieger, M., Ma'ayan, D., Wise, J., Aldrich, J., . . . Crane, K. (2020). Penrose: from mathematical notation to beautiful diagrams. *ACM Transactions on Graphics, 39*(4), 144:1 - 144:16. doi:https://doi.org/10.1145/3386569.3392375

Ye, X. (2023). A method of computer library information data management based on network analysis. *Journal of Computational Methods in Sciences and Engineering, 23*(2), 759-771. doi:https://doi.org/10.3233/JCM-226579

Yuan, X., & Zhang, X. (2015). An ontology-based requirement modeling for interactive software customization. *2015 IEEE International Model-Driven Requirements Engineering Workshop (MoDRE)* (pp. 1-10). Ottawa, ON, Canada: IEEE. doi:https://doi.org/10.1109/MoDRE.2015.7343872

Zakaria, Z., Kasim, S., Hasbullah, N. H., Azadin, A. A., Ahmar, A. S., & Hidayat, R. (2018). The Development of Personality Ontology Based on the Methontology Approach. *International Journal of Engineering & Technology, 7*(2.5), 73-76. doi:http://dx.doi.org/10.14419/ijet.v7i2.5.13955