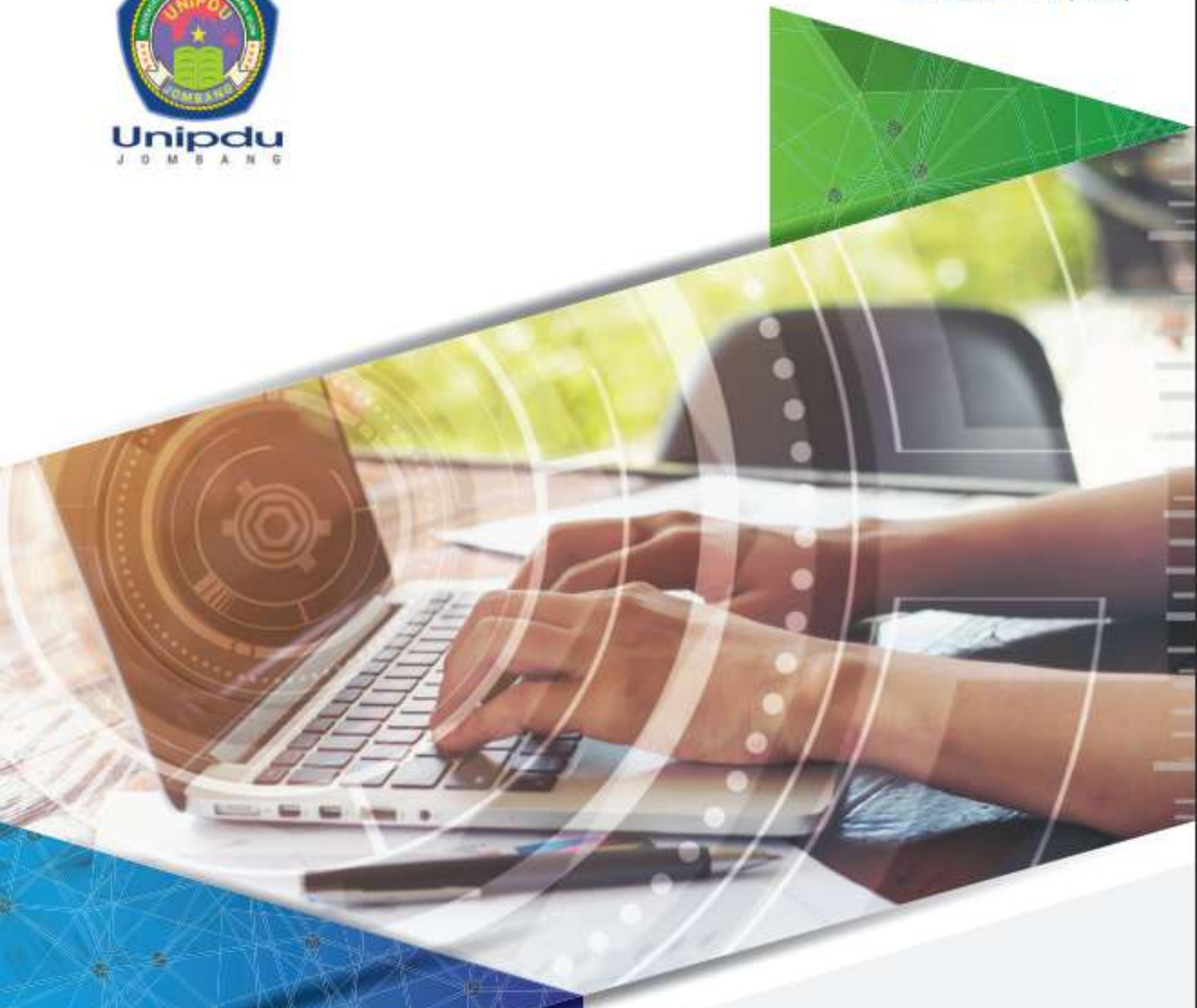
ISSN 2502-3357 (Online)
ISSN 2503-0477 (Print)

Unipdu
JOMBANG



Register

Jurnal Ilmiah Teknologi Sistem Informasi
Scientific Journals of Information System Technology

Vol. 08, No. 01, January 2022

Department of Information Systems
Faculty of Science and Technology
Universitas Pesantren Tinggi Darul 'Ulum

8 (1) January 2022 10-23 ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
Register: *Jurnal Ilmiah Teknologi Sistem Informasi*

Contents lists available at www.journal.unipdu.ac.id

Register

Journal Page is available to www.journal.unipdu.ac.id/index.php/register

Review article

Software similarity measurements using UML diagrams: A systematic literature review

Evi Triandini [a,], Reza Fauzan [b], Daniel O. Siahaan [c], Siti Rochimah [d], I Gede Suardika [e], Devi Karolita [f]*

[a,e] *Department of Information Systems, Institut Teknologi dan Bisnis STIKOM Bali, Denpasar, Indonesia*
[b] *Department of Informatics Engineering, Politeknik Negeri Banjarmasin, Banjarmasin, Indonesia*
[c,d] *Department of Informatics Engineering, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia*
[e] *Department of Software Systems and Cybersecurity, Monash University, Melbourne, Australia*
email: [a,*] evi@stikom-bali.ac.id, [b] reza.fauzan@poliban.ac.id, [c] daniel@if.its.ac.id, [d] siti@if.its.ac.id, [e] suardika@stikom-bali.ac.id, [f] devi.karolita@monash.edu
* *Correspondence*

ARTICLE INFO

Article history:
Received 20 January 2021
Revised 22 April 2021
Accepted 27 April 2021
Available online 17 May 2021

Keywords:
software similarity
similarity measurement
UML diagram similarity
semantic similarity
structural similarity

Please cite this article in IEEE style as:
E. Triandini, R. Fauzan, D. O. Siahaan, S. Rochimah, I. G. Suardika and D. Karolita, "Software similarity measurements using UML diagrams: A systematic literature review," *Register: Jurnal Ilmiah Teknologi Sistem Informasi*, vol. 8, no. 1, pp. 10-23, 2022.

ABSTRACT

Every piece of software uses a model to derive its operational, auxiliary, and functional procedures. Unified Modeling Language (UML) is a standard displaying language for determining, recording, and building a software product. Several algorithms have been used by researchers to measure similarities between UML artifacts. However, there no literature studies have considered measurements of UML diagram similarities. This paper presents the results of a systematic literature review concerning similarity measurements between the UML diagrams of different software products. The study reviews and identifies similarity measurements of UML artifacts, with class diagram, sequence diagram, statechart diagram, and use case diagram being UML diagrams that are widely used as research objects for measuring similarity. Measuring similarity enables resolution of the problem domains of software reuse, similarity measurement, and clone detection. The instruments used to measure similarity are semantic and structural similarity. The findings indicate opportunities for future research regarding calculating other UML diagrams, compiling calculation information for each diagram, adapting semantic and structural similarity calculation methods, determining the best weight for each item in the diagram, testing novel proposed methods, and building or finding good datasets for use as testing material.

1. Introduction

Every piece of software features a set of models constituting its structural, behavioral, and functional perspectives. Unified Modeling Language (UML) is a standard modeling language used to specify, document, and build software products [1]. Software development using UML demands considerable effort and substantial time be invested in producing a good model. Reusing UML diagrams could solve this problem by helping to accelerate the software development process. However, reusing UML diagrams requires calculating the similarities between UML diagram artifacts.

Researchers have used several algorithms to measure the similarities between UML artifacts. However, no literature studies have considered measurements of similarities between UML diagram artifacts. Thus, this review aims to systematically analyze existing peer-reviewed literature, aggregating the results to explore similarity measurements for UML diagrams used in multiple software products.

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

This study utilizes a systematic literature review (SLR) technique following the approaches used by Kitchenham and Charters [2] and Inayat et al., [3].This technique enables the summary of existing evidence concerning using UML diagrams for measurement and the presentation of background knowledge to properly position new research activities.

For this SLR, we have considered UML sequence diagrams, use case diagrams, statechart diagrams, activity diagrams, deployment diagrams, interaction diagrams, and object diagrams that have been investigated to measure similarities in software design. This study contributes an in-depth overview of similarity measurement UML diagrams and demonstrates the challenge of measuring UML diagrams using lexical information produced by the diagrams. Researchers have generally used UML diagrams as a research topic because UML is widely used in both software development and formal software engineering education.

The paper is structured as follows: Section 2 describes the method used for the review. Section 3 gathers and analyzes the selected studies. Section 4 discusses the results and answers the research question posed by this systematic review. Finally, Section 5 presents the SLR's conclusions, along with its benefits, drawbacks, and implications for future research.

2. Methodology

This paper's research method (see Fig. 1) was developed according to guidelines provided by Kitchenham and Charters [2]. First, we identified research questions to dictate the paper's contribution. Second, we produced a search strategy. Third, we selected the papers, which included performing quality assessment. Finally, we extracted the data.

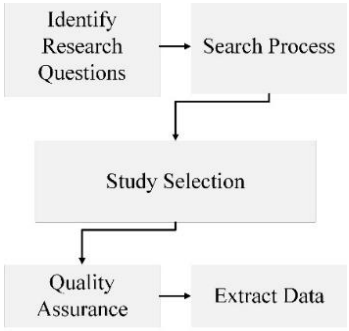


Fig. 1. Systematic literature review phases for this study

2.1. Research questions

This paper answers five main research questions, which were developed to shape the paper's contributions.

RQ 1: What Unified Modeling Language diagrams are used to measure the similarity between two software products?
This question's answer will provide a list of diagrams that have been measured and have not been measured, enabling the opportunity to measure diagrams that have never been measured.

RQ2: What are the parameters (measuring instruments) used to measure the similarity between two software products?
This second question's answer will demonstrate the measurement perspective used to measure similarity and provide the opportunity to comprehensively evaluate different parameters.

RQ3: What is the domain of the problem resolved by the study?
This question's answer will demonstrate the study's purposes for measuring software similarity using UML diagrams.

RQ4: What methods are used to calculate the similarity between the Unified Modeling Language diagrams of two software products?

RQ4.1: How does the study in question determine the weight of the calculation if it is weighted?
This fourth question's answer will help us to list the method used for similarity measurement and provide opportunities for improving techniques and the weighting of calculations.

RQ5: What dataset is used?
This final question's answer will demonstrate the difficulty of identifying a useful dataset to be used to train and test data. A good dataset will help us improve the measurement.

 http://doi.org/10.26594/register.v8i1.2248

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

2.2. Search strategy

We searched for studies using various digital libraries: Scopus, Science Direct, IEEE, ACM, and Mendeley. We searched based on title, keywords, and abstracts, and we used the same set of keywords for all libraries: "software similarity"; "software similarity measure"; "software similarity metric"; "UML similarity"; "UML similarity measure"; "UML similarity metric". The keywords used for this study were chosen to locate research articles measuring the similarity between designs; these keywords do not relate to the quality of the software. The searches were conducted for papers published between January 2013 and April 2020. This produced 833 papers from Science Direct, 1341 papers from IEEE, and 1087 papers from ACM.

2.3. Study selection

Paper selection included several stages (see Fig. 2). The initial search using different digital libraries and based on titles, keywords, and abstracts found 3261 papers. Those papers were filtered manually by excluding papers that were irrelevant with research topic or were duplicates. Following this step, 82 papers remained. This approach followed that used by Kitchenham et al., [2]. The remaining papers were subject to a more sophisticated assessment based on the full paper; additionally, papers not written in English were excluded. Following this step, 74 papers remained, including four literature review papers with different objects. Next, two researchers filtered each paper for inclusion independently; this involved a quick reading of each paper. These two researchers both had a software engineering research background. Papers were chosen if they could answer at least one of RQ1, RQ 2, or RQ 3. Following this step, only 16 papers remained. The next step involved forward and backward snowballing. This led to the identification of 14 additional papers. Then, the second step was repeated, ultimately leaving 28 papers.

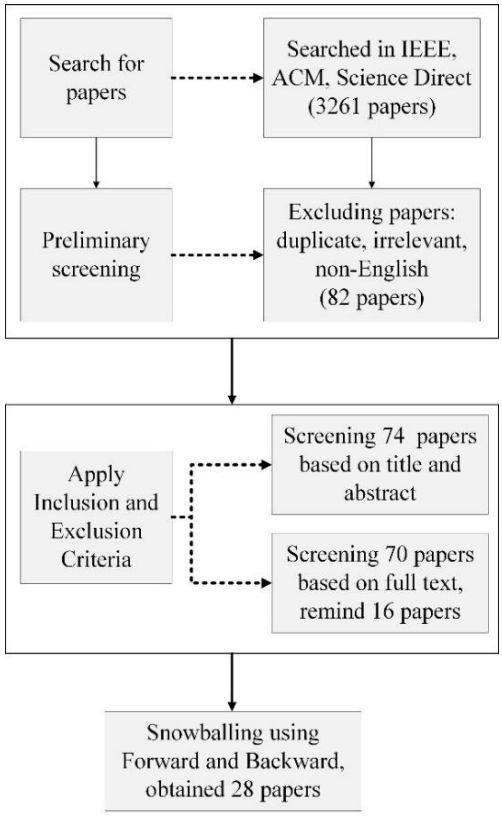


Fig. 2. Paper search and selection stage of systematic literature review

2.4. Quality assurance and data extraction

The quality assurance in this paper is followed by Tuma, Calikli, and Scandariato [4]. We performed an assessment of included/excluded papers. As we knew before, there were two researchers who assess the paper. The criterion for the researcher who had assessed was that of midwife research experience in software engineering. The papers selected had to be included by both researchers. If one or both excluded a paper, that paper would not be selected. The quality of each paper was tested using a

 http://doi.org/10.26594/register.v8i1.2248

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

modification of the Souza metric [5]. These consistency norms incorporate inquiries evaluating the quality of a report and its contribution to the SLR's range. Principles included each investigation's meticulousness, reliability, and importance. The first criterion (C1) was whether the research objectives were well explained, with answers being yes (1), nominally (0.5), and no (0). The second criterion (C2) was whether the research context was handled properly, with answers being yes (1), nominally (0.5), and no (0). The third criterion (C3) was whether the findings were clearly stated, with answers being yes (1), nominally (0.5), and no (0). The fourth criterion (C4) was how valuable the research was, based on the findings, with answers being >80% (1), <20% (0), and in-between (0.5). The quality measurement results are presented in Fig. 3. Based on Fig. 3, 70% of the papers are excellent, 21% of the papers are very good, and the rest are good.

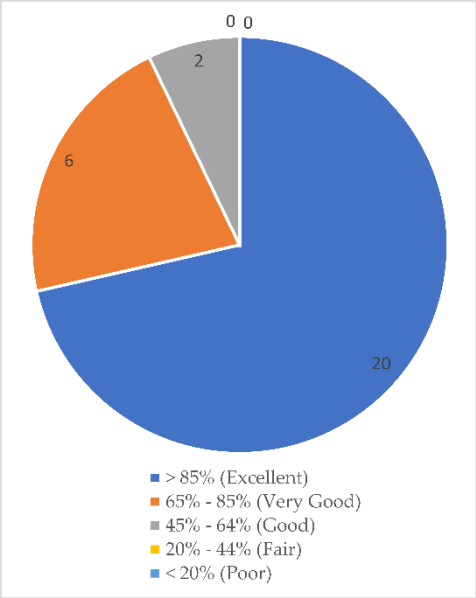


Fig. 3. Percentage ratings for study quality

3. Findings

The 28 papers selected are detailed in Table 1. For each review, we identified the problem solved, the method, the UML diagram measured, the parameters, and the source of the dataset. The 28 articles selected for this study are used to discuss and explain the answers to the research questions. The research papers considered by this SLR represent research discussing one or more UML diagrams.

Table 1. Overview of previous work

Prev. Work & Method	UML Diagram	Parameter	Problem Domain & Source of Dataset
Park and Bae [6] Match two UML to identify the candidate component set for reuse	Class diagram, sequence diagram	Lexical information, structural information	Software Reuse & Reverse from source code
Storrle [7] Match the similarity of the element in every diagram syntactically.	Activity diagram, class diagram, deployment diagram, interaction diagram, object diagram, statechart, use case diagram	Lexical information	Clone Detection & One software design (XMI-format)
Robles et al., [8] Calculate the similarity between two lexical information through the ontology as domain knowledge.	Class diagram	Lexical information	Software Reuse & n.a.
Salami and Ahmed [9] Transform the class diagram into a graph. Match them based on the structure of the graph.	Class diagram	Structural Similarity	Software Reuse & n.a.
Bonilla-Morales, Crespo, and Clunie [10] Use lexical information from the use case as an input to find the use case in the ontology.	Use case diagram	Lexical information	Software Reuse & n.a.
Salami and Ahmed [11] Optimize reuse from previous work [9]with algorithm modification.	Class Diagram	Structural Similarity	Software Reuse & Two class diagrams from opensource system

 http://doi.org/10.26594/register.v8i1.2248

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

Prev. Work & Method	UML Diagram	Parameter	Problem Domain & Source of Dataset
Assuncao and Vergilio [12] Optimize the searching process by finding an adequate model.	Class diagram	Lexical information, structural information	Software Reuse & n.a.
Qiu, Li, and Sun [13] Divide the instrument into two. They are property (attribute and operation) and relationship (kind and number of public operation).	Class diagram	Lexical information, structural information	Similarity Assessment & Reverse from source code
Salami and Ahmed [14] Perform initial screening by taking metadata, sorting, and calculating the diagram, adapting artifacts to be used, integrated into the new system.	Class diagram, sequence diagram	Lexical information	Software Reuse & n.a.
Singh and Kaur [15] Compare the number of attributes, attribute names, number of operations, name of operation, class type and visibility of two class diagrams	Class diagram	Lexical information	Clone Detection & Two hospital class diagrams
Al-Khiaty and Ahmed [16] Calculate similarity using the greedy algorithm.	Class diagram	Lexical information, structural information	Software Reuse & Two class diagrams from opensource system
Al-Khiaty and Ahmed [17] Calculate similarity using simulated annealing.	Class diagram	Lexical information, structural information	Software Reuse & Two class diagrams from opensource system
Salami and Ahmed [18] Use the Message Object Order Graph method and combine the Genetic Algorithm	Sequence diagram	Structural information	Software Reuse & Ten software engineering undergraduate lecture materials
Nikiforova et al., [19] Match every instrument in abstract level between two diagrams. But they proposed a detail information level.	Class diagram	Lexical information, structural information	Similarity Assessment & Three simple class diagrams
Elkamel, Gzara, and Ben-Abdallah [20] Recommend a new class by clustering similar classes.	Class diagram	Lexical information	Software Reuse & Twenty class diagrams from undergraduate student
Al-Khiaty and Ahmed [21] Propose two instruments; they are internal information and neighborhood information. Find the best weight combination of them.	Class diagram	Lexical information, structural information	Software Reuse & Two class diagram flight booking system
Adamu and Zainoon [22] Divide into three instruments, namely similarity of concepts (Csim); similarity of functions (Fsim); metric similarity (MBSim)	Class diagram	Lexical information, structural information	Similarity Assessment & Five simple class diagrams
Al-Khiaty and Ahmed [23] Optimize previous work [21] by combining Greedy with the Genetic Algorithm.	Class diagram	Lexical information, structural information	Software Reuse & Two class diagrams from opensource system
Adamu and Zainon [24] Combine similarities in structure, functional, and behavioral points of view using different weights.	Class diagram, sequence diagram, state machine diagram	Lexical information, structural information	Software Reuse & Six projects from undergraduate student
Adamu and Zainon [25] The mapping between diagrams then calculate similarity using edit distance	Sequence diagram	Structural information	Software Reuse & Ten projects from undergraduate student
Siahaan et al., [26] Divide similarity instruments into two, namely objects and structures with arbitrary weights	Sequence diagram	Lexical information, structural information	Similarity Assessment & Three sequence diagrams
Adamu and Zainon [27] Match the graph that builds from the transition information.	Statechart diagram	Structural information	Software Reuse & Two state diagrams
Fauzan et al., [28] Divide similarity instruments into two, namely property and relations.	Class diagram	Lexical information	Similarity Assessment & Three class diagrams

 http://doi.org/10.26594/register.v8i1.2248

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

Prev. Work & Method	UML Diagram	Parameter	Problem Domain & Source of Dataset
Fauzan et al., [29] Divide similarity instruments into two, namely property and transitions.	Activity diagram	Lexical information	Similarity Assessment & Two activity diagrams
Adamu, Wan, and Abdulrahman [30] Combine similarities in structure, functional, and behavioral points of view using different weights.	Class diagram, statechart diagram	Lexical information	Software Reuse & Four project that consist of 4 class diagrams and ten statechart diagrams
Triandini et al., [31] Divide similarity instruments into two, namely property and messages.	Sequence Diagram	Lexical Information	Similarity Assessment & Three sequence diagrams
Fauzan et al., [32] Divide similarity instruments into two, namely property and relation.	Use case diagram	Lexical information	Similarity Assessment & Two use case diagrams
Čech [33] Translate class diagram into graph.	Class diagram	Lexical information, structural information	Software Reuse & One hundred and nineteen class diagrams

4. Results and Discussion

This section provides and discusses the results in the context of the research questions. A detailed description of the findings is presented to investigate the similarity measurements of diagrams used in software design.

The correspondent/first author diversity of publications had an even distribution, spreading from Europe, Asia, America, and Africa. We considered the location of the first author affiliation country to determine the authorship per geographical distribution (Fig. 4). Spain, Panama, Denmark, Brazil, Latvia, Czechia, India, Republic of Korea, China, Nigeria, and Arab Saudi. Arab Saudi was the most productive country with nine publications. Some studies were authored/co-authored by the same person, indicating the existence of an active research group in this field.

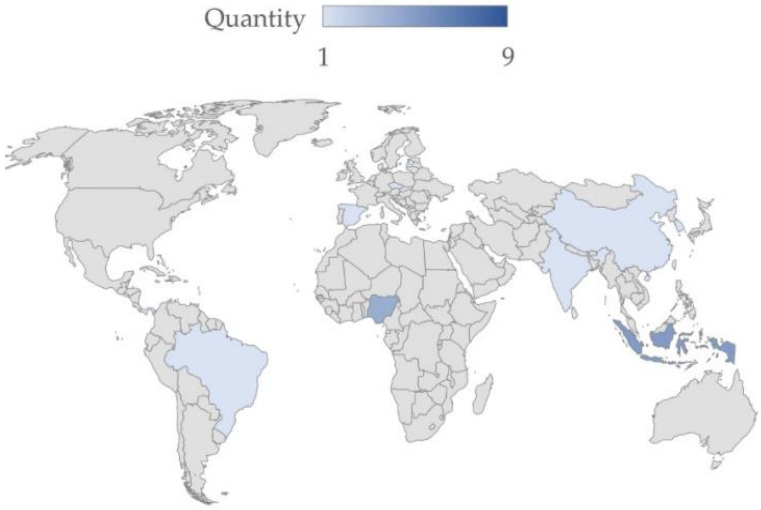


Fig. 4. Authorship distribution per country

4.1. What UML diagrams are used to measure the similarity between two software products?

As shown in Table 1, seven UML diagrams are used to measure the similarity between two software products: activity diagrams, class diagrams, deployment diagrams, interaction diagrams, object diagrams, statechart diagrams, sequence diagrams, and use case diagrams. Two papers discussed activity diagrams [7, 29], 20 papers discussed class diagrams [6, 7, 8, 9, 11, 12, 13, 14, 15, 17], [19, 20, 21, 22, 23, 24, 28, 30, 33, 34], one paper discussed deployment diagrams [7], one paper discussed interaction diagrams [7], one paper discussed object diagrams [7], four papers discussed statechart diagrams [7], [14, 24, 27], seven papers discussed sequence diagrams [6, 14, 18, 24, 25, 26, 31], and three papers discussed use case diagrams (see Fig. 5). Studies considered by this SLR describes the results for research considering one or more UML diagram types. Class diagrams are the UML diagram type most often used to measure the similarity between two software products, followed by sequence diagrams, statechart diagrams, and use case diagrams.

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

Research considering class diagrams is most common (constituting 71.4% of research articles) because class diagrams are a software development standard during the design phase [8, 35]. Class diagrams constitute a UML structural diagram; as such, there remains substantial opportunity to examine other diagrams pertaining to different categories.

In terms of frequency, research into behavior diagrams follows. This type includes sequence diagrams, statechart diagrams, and use case diagrams. As Fig. 5 indicates, class diagrams appear to be the most exciting direction for similarity measurement, with research into use case diagram similarity measurements not appearing until after 2012 and measurements utilizing statechart diagrams reappearing in 2017, indicating a different approach to measuring similarity.

Several studies [6, 7, 14, 24] do not measure only one diagram, instead utilizing several diagrams as a single software measurement. For example, Adamu and Zainon [24] measured software design using three types of diagrams: class diagrams, sequence diagrams, and statechart diagrams. Class diagrams were used as structural components, sequence diagrams as functional components, and statechart diagrams as behavioral components. That study combined the three components in one valuation with an arbitrary weight.

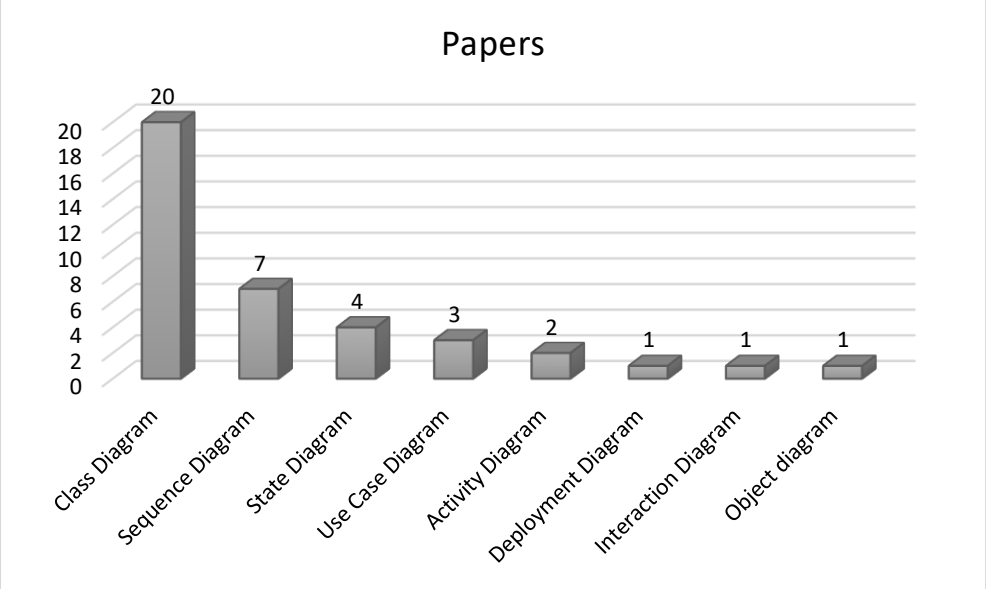


Fig. 5. The number of articles based on the type of diagram

Several studies [6, 7, 14, 24] do not measure only one diagram, instead utilizing several diagrams as a single software measurement. For example, Adamu and Zainon [24] measured software design using three types of diagrams: class diagrams, sequence diagrams, and statechart diagrams. Class diagrams were used as structural components, sequence diagrams as functional components, and statechart diagrams as behavioral components. That study combined the three components in one valuation with an arbitrary weight.

4.2. What are the parameters (measuring instruments) used to measure the similarity between two software products?

One paper [7] discussed various different diagrams and calculated document similarity in XMI [36] directly. Thus, it did not consider the structure of each calculated diagram. The rest of the papers discussed similarities in terms of diagrams, with similarities divided into semantic and structural similarity [26].

All of the papers used lexical information from class diagrams to measure semantic similarity between two classes. Such lexical information comprises the class name, the list of attributes, and the list of operations. Additionally, some papers [9, 11, 13, 16, 17, 21, 25, 28, 29, 32] used lexical information to measure the structural similarity between the diagrams, adding the relationship name and the neighbor's name. However, some papers did not use lexical information to measure structural similarity. For example, [15] used the number of neighbors from each class.

The semantic similarity between the two classes needs to be more complex, given it is known that a class does not only comprise the class name, attribute names, and operation names. For instance, an

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

attribute contains more information inside, such as modifier and datatype. Meanwhile, operations include, for example, modifier, operation type, and parameters. Additionally, structural similarity should be more complex. For example, a relationship features a type, a name, a cardinality, etc. Thus, the challenge is to be able to measure the similarity between class diagrams with complete information.

Structural similarity is measured by the message between two objects, which contains lexical information, including message name, source object or source class name, and target object or target class name. However, complete information is needed to make better measurements. For instance, we could add message type. The semantic similarity of the statechart diagram is also measured using lexical information, including state name, activity, and type. However, a state also consists of more complex information. For instance, a state features entry activity, which contains the activity value and type (e.g., signal, call).

Additionally, more transition information is needed to measure structural similarity. Although most of the papers used the source state and target state names, transition information includes more complex items such as trigger events, guard, and affect behavior. Furthermore, a trigger event features even more complex information (e.g., expression, value, kind). While all of the papers used lexical information from the use case diagram, they only measured the structural similarities. Lexical information combines the name of the actor, the use case, and the relation. However, we can also calculate the semantic similarity based on the list of actors and use cases.

Based on these findings, we can classify instruments into two categories: lexical information and structural information. Lexical information comprises complete information and partial information. Whole lexical information is a collection of detailed information (e.g., information class name, list of attributes, and list of operations in the class diagram), while partial information only uses some of the lexical information. Structural information comprises graph-form information and neighborhood information. Graph-form information prioritizes the similarity of the structure of the diagram that has been converted into a graph, thus overriding lexical information. Meanwhile, neighborhood information uses lexical information from the structure and is also divided into two categories: whole neighborhood information and partial neighborhood information. All information can be either lexical or based on the instrument number. The detailed instrument, based on previous research, appears in Fig. 6, which demonstrates the two kinds of information: lexical and structural.

Research on whole lexical information was identified for class diagrams [7, 13, 15, 16, 17, 19, 21, 23, 28], sequence diagrams [7, 26, 31], statechart diagrams [7] and use case diagrams [7, 10, 32], while research on partial lexical information was identified for class diagrams [6, 8, 14, 20, 22], sequence diagram [14, 25], and statechart diagrams [14]. Regarding structural information which includes graph-form information and neighborhood information, research on graph-form information was identified for class diagrams [33], sequence diagrams [6, 18], and statechart diagrams [27]. Finally, research on whole neighborhood information was identified for class diagrams [8, 16, 17, 19, 21, 23] and sequence diagrams [26] and research on partial neighborhood information was identified for class diagrams [6, 9, 11, 12, 13, 22] and sequence diagrams [25].

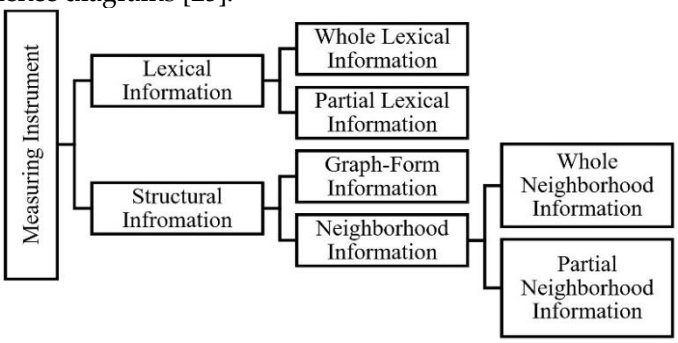


Fig. 6. Measuring instruments

Most of the research papers aimed to propose or optimize a method, with only a few aiming to test a proposed method. For software reuse, only two papers [20, 21] tested the proposed method using the class diagram. For clone detection, two papers [7, 15] used class diagrams for testing. However, no similarity measurement papers tested a proposed method. Fig. 7 presents the aims of the different studies considered.

 http://doi.org/10.26594/register.v8i1.2248

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

Software reuse papers tested their research using precision and recall, with none utilizing similarity measurements to test their research. Clone detection papers also tested their research using precision and recall. However, it would be somewhat inappropriate to test similarity measurements and clone detection using precision and recall. This is because design similarity has no standardized goals, with the average of all measurements being based on subjective expert assessments, with measurement agreement among experts used to measure a design's similarity. However, this does not necessarily obtain the correct score. Therefore, design similarity should be tested according to the method's reliability. This can involve using the kappa statistic [37, 38, 39] or Gwet's AC1 [40, 41, 42] to measure a method's results and the Alpha Cronbach [43, 44] to ensure the data's reliability. But, Gwet's AC1 could be better than the kappa statistic for assessment case [45, 46, 47, 48].

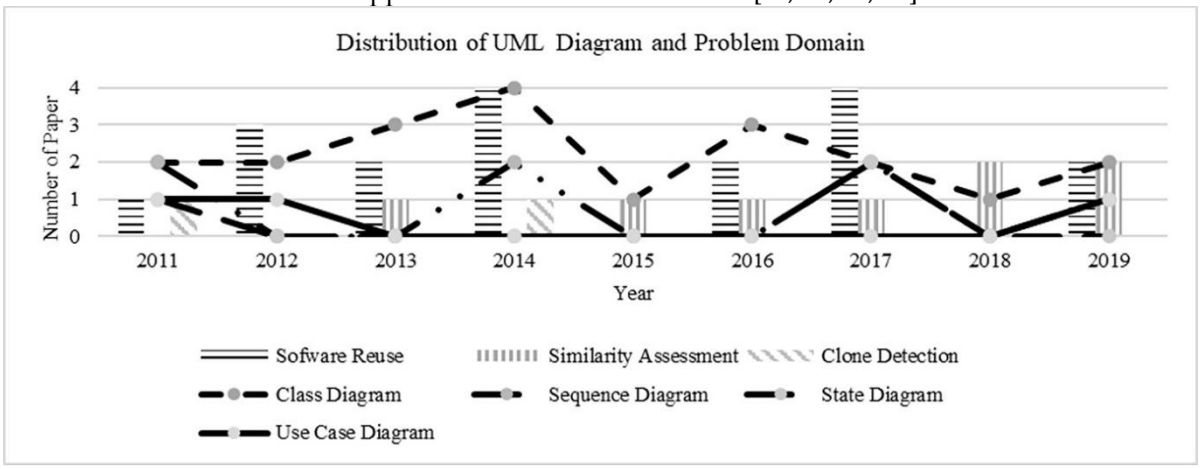


Fig. 7. Distribution of Unified Modeling Language diagrams and problem domains

Software reuse papers tested their research using precision and recall, with none utilizing similarity measurements to test their research. Clone detection papers also tested their research using precision and recall. However, it would be somewhat inappropriate to test similarity measurements and clone detection using precision and recall. This is because design similarity has no standardized goals, with the average of all measurements being based on subjective expert assessments, with measurement agreement among experts used to measure a design's similarity. However, this does not necessarily obtain the correct score. Therefore, design similarity should be tested according to the method's reliability. This can involve using the kappa statistic [36] to measure a method's results and the Alpha Cronbach [37, 38] to ensure the data's reliability.

4.3. What methods are used to calculate the similarity between the Unified Modeling Language diagrams of two software products?

The methods can be divided into those calculating lexical similarity and those calculating structural similarity. First, the method for calculating lexical similarity comprises syntactic and semantic approaches. In the early years, several papers used a syntactic approach [49, 50, 51, 52], which is generally used to detect clones so that the similarity between words becomes binary [7]. Then, such research [18, 23, 28] began to use semantic similarity [53, 54, 55]. The similarity between words was no longer binary but located on a scale between zero and one. The most widely used algorithm for semantic calculation is Wu Palmer [56, 57, 58], which combines with Wordnet [59, 60, 61] to find the semantic similarity between two words [57, 62]. The Wu Palmer similarity measurement is characterized by its straightforwardness and precise findings.

Second, most papers changed diagrams into graphs [63] or ontologies [10] to calculate structural similarities. Then, they used graph matching [64, 65, 66, 67, 68] to measure similarity. Yuan [63] used the maximum common subgraph [69, 70, 71, 72] as a similarity method. Maximum common subgraphs examine isomorphism graphs. The bigger the isomorphism graph, the higher the level of similarity. Another method for measuring graph similarity is the graph edit distance [73, 74, 75, 76, 77]. The difference between the graph edit distance and the maximum common subgraph is on the part of the graph being measured.

The other papers used lists of combinations of source and target information in the diagram. For example, some used a combination of source class, relation name, and target class to calculate the

 http://doi.org/10.26594/register.v8i1.2248

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

similarity of the combination to the combination identified for other diagrams. Some of these methods utilize a repository to perform software reuse.

Based on our findings, syntactic similarity performs better than semantic similarity in terms of word comparison for software reuse and clone detection. However, semantic similarity performs better than syntactic similarity when it comes to similarity measurements.

Some papers used the weight in their calculation to show the importance of each item. For example, weights are required for each combination of source classes, relation names, and target classes to measure the similarity of combinations between the two diagrams. However, most use an arbitrary weight and have not tested the weight to obtain the best weight for calculating similarity. Only one paper [21] tested weight combinations in the context of class diagram instruments. As such, another experiment is needed to identify the best weight for every item in the diagram.

4.4. What dataset was used?

According to Table 1, the datasets used were limited, with some comparing two simple diagrams and some using a dataset built by the researchers themselves. The problem with self-built datasets is the lack of good datasets produced. Furthermore, most papers concluded that the method needed to be tested on a better, more complex dataset. Thus, it is necessary to address the challenging problem of obtaining a dataset suited to testing such methods.

5. Conclusion

This paper's SLR considered 28 papers measuring UML similarity. The papers were derived from some digital libraries, and four diagrams were found to be commonly used to measure software similarity: class diagram, sequence diagram, statechart diagram, and use case diagram. The problem domains resolved were software reuse, similarity assessment, and clone detection. The instruments used to measure similarity were semantic and structural. Semantic similarity was identified as more suitable for comparing two words, whereas syntactic similarity was identified as more suitable for measuring similarity in the contexts of software reuse and clone detection.

As per the results discussed in Section 4, opportunities in future work have been identified. First, there are many UML diagrams that have yet to be used to calculate similarity measurements. Second, it is necessary to complete the information for each diagram calculated. Third, we can adapt both semantic and structural similarity calculation methods. Fourth, we can determine the optimum weight for each item in a given diagram. Fifth, it is necessary to test the proposed methods. Finally, it is necessary to build or identify a dataset that can be usefully applied as source material for testing the method.

Author Contributions

Evi Triandini: Conceptualization, methodology, and writing – review & editing. Reza Fauzan: Validation, formal analysis, and writing – original draft. Daniel O. Siahaan: Supervision. Siti Rochimah: Resources. I Gede Suardika: Data curation. Devi Karolita: Resources.

Declaration of Competing Interest

We declare that we have no conflict of interest.

References

[1] M. J. Chonoles, "Chapter 2 - What is UML?," in *OCUP Certification Guide*, Morgan Kaufmann, 2018, pp. 17-41.

[2] B. Kitchenham, R. Pretorius, D. Budgen, O. P. Brereton, M. Turner, M. Niazi and S. Linkman, "Systematic literature reviews in software engineering – A tertiary study," *Information and Software Technology,* vol. 52, no. 8, pp. 792-805, 2010.

[3] I. Inayat, S. S. Salim, S. Marczak, M. Daneva and S. Shamshirband, "A systematic literature review on agile requirements engineering practices and challenges," *Computers in Human Behavior,* vol. 51, pp. 915-929, 2015.

[4] K. Tuma, G. Calikli and R. Scandariato, "Threat analysis of software systems: A systematic literature review," *Journal of Systems and Software,* vol. 144, pp. 275-294, 2018.

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

[5] E. Souza, A. Moreira and M. Goulão, "Deriving architectural models from requirements specifications: A systematic mapping study," *Information and Software Technology,* vol. 109, pp. 26-39, 2019.

[6] W.-J. Park and D.-H. Bae, "A two-stage framework for UML specification matching," *Information and Software Technology,* vol. 53, no. 3, pp. 230-244, 2011.

[7] H. Störrle, "Towards clone detection in UML domain models," in *ECSA '10: Proceedings of the Fourth European Conference on Software Architecture: Companion*, 2010.

[8] K. Robles, A. Fraga, J. Morato and J. Llorens, "Towards an ontology-based retrieval of UML Class Diagrams," *Information and Software Technology,* vol. 54, no. 1, pp. 72-86, 2012.

[9] H. O. Salami and M. A. Ahmed, "A Framework for Class Diagram Retrieval Using Genetic Algorithm," in *The 24th International Conference on Software Engineering & Knowledge Engineering*, San Francisco Bay, 2012.

[10] B. Bonilla-Morales, S. Crespo and C. Clunie, "Reuse of Use Cases Diagrams: An Approach based on Ontologies and Semantic Web Technologies," *IJCSI International Journal of Computer Science Issues,* vol. 9, no. 1, pp. 24-29, 2012.

[11] H. O. Salami and M. Ahmed, "Class Diagram Retrieval Using Genetic Algorithm," in *2013 12th International Conference on Machine Learning and Applications*, 2013.

[12] W. K. G. Assuncao and S. R. Vergilio, "Class Diagram Retrieval with Particle Swarm Optimization," in *The 25th International Conference on Software Engineering and Knowledge Engineering (SEKE 2013)*, 2013.

[13] D. H. Qiu, H. Li and J. L. Sun, "Measuring software similarity based on structure and property of class diagram," in *2013 Sixth International Conference on Advanced Computational Intelligence (ICACI)*, 2013.

[14] H. O. Salami and M. Ahmed, "A framework for reuse of multi-view UML artifacts," *The International Journal of Soft Computing and Software Engineering [JSCSE],* vol. 3, no. 3, pp. 156-162, 156-162.

[15] S. Singh and R. Kaur, "Clone Detection in UML Class Models using Class Metrics," *ACM SIGSOFT Software Engineering Notes,* vol. 39, no. 3, 2014.

[16] M. A.-R. Al-Khiaty and M. Ahmed, "Similarity assessment of UML class diagrams using a greedy algorithm," in *2014 International Computer Science and Engineering Conference (ICSEC)*, 2014.

[17] M. A.-R. Al-Khiaty and M. Ahmed, "Similarity assessment of UML class diagrams using simulated annealing," in *2014 IEEE 5th International Conference on Software Engineering and Service Science*, 2014.

[18] H. O. Salami and M. Ahmed, "Retrieving sequence diagrams using genetic algorithm," in *2014 11th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 2014.

[19] O. Nikiforova, K. Gusarovs, L. Kozacenko, D. Ahilcenoka and D. Ungurs, "An Approach to Compare UML Class Diagrams Based on Semantical Features of Their Elements," in *ICSEA 2015: The Tenth International Conference on Software Engineering Advances*, 2015.

[20] A. Elkamel, M. Gzara and H. Ben-Abdallah, "An UML class recommender system for software design," in *2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)*, 2016.

[21] M. A.-R. Al-Khiaty and M. Ahmed, "UML Class Diagrams: Similarity Aspects and Matching," *Lecture Notes on Software Engineering,* vol. 4, no. 1, pp. 41-47, 2016.

[22] A. Adamu and W. M. N. W. Zainoon, "A Framework for Enhancing the Retrieval of UML Diagrams. In: Kapitsaki G., Santana de Almeida E. (eds) Software Reuse: Bridging with Social-Awareness," in *International Conference on Software Reuse*, Cham, 2016.

[23] M. A.-R. Al-Khiaty and M. Ahmed, "Matching UML class diagrams using a Hybridized Greedy-Genetic algorithm," in *2017 12th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT)*, 2017.

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

[24] A. Adamu and W. M. N. W. Zainon, "Multiview Similarity Assessment Technique of UML Diagrams," *Procedia Computer Science,* vol. 124, pp. 311-318, 2017.

[25] A. Adamu and W. M. N. W. Zainon, "Similarity Assessment of UML Sequence Diagrams Using Dynamic Programming. In: Badioze Zaman H. et al. (eds) Advances in Visual Informatics," in *International Visual Informatics Conference*, Cham, 2017.

[26] D. O. Siahaan, Y. Desnelita, Gustientiedina and S. Sunarti, "Structural and semantic similarity measurement of UML sequence diagrams," in *2017 11th International Conference on Information & Communication Technology and System (ICTS)*, 2017.

[27] A. Adamu and W. M. N. W. Zainon, "Matching and retrieval of state machine diagrams from software repositories using Cuckoo Search Algorithm," in *2017 8th International Conference on Information Technology (ICIT)*, 2017.

[28] R. Fauzan, D. O. Siahaan, S. Rochimah and E. Triandini, "Class Diagram Similarity Measurement: A Different Approach," in *2018 3rd International Conference on Information Technology, Information System and Electrical Engineering (ICITISEE)*, 215-219, 2018.

[29] R. Fauzan, D. O. Siahaan, S. Rochimah and E. Triandini, "Activity Diagram Similarity Measurement: A Different Approach," in *2018 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, 2018.

[30] A. Adamu, W. M. N. Wan and S. M. Abdulrahman, "Empirical Investigation of UML Models Matching through Different Weight Calibration," in *ICSCA '19: Proceedings of the 2019 8th International Conference on Software and Computer Applications*, 2019.

[31] P. E. Triandini, R. Fauzan, D. O. Siahaan and S. Rochimah, "Sequence Diagram Similarity Measurement: A Different Approach," in *2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, 2019.

[32] R. Fauzan, D. O. Siahaan, S. Rochimah and E. Triandini, "Use Case Diagram Similarity Measurement: A New Approach," in *2019 12th International Conference on Information & Communication Technology and System (ICTS)*, 2019.

[33] P. Čech, "Matching UML class models using graph edit distance," *Expert Systems with Applications,* vol. 130, pp. 206-224, 2019.

[34] M. Bae, S. Kang and S. Oh, "Semantic similarity method for keyword query system on RDF," *Neurocomputing,* vol. 146, pp. 264-275, 2014.

[35] M. Fowler, UML Distilled: A Brief Guide to the Standard Object Modeling Language, 3rd ed., Addison-Wesley Professional, 2003.

[36] J. Kovse and T. Härder, "Generic XMI-Based UML Model Transformations," in *International Conference on Object-Oriented Information Systems*, Berlin, Heidelberg, 2002.

[37] M. L. McHugh, "Interrater reliability: the kappa statistic," *Biochemia Medica,* vol. 22, no. 3, pp. 276-282, 2012.

[38] J. L. Fleiss, B. Levin and M. C. Paik, Statistical Methods for Rates and Proportions, John Wiley & Sons, 2013.

[39] J. R. Landis and G. G. Koch, "The Measurement of Observer Agreement for Categorical Data," *Biometrics,* vol. 33, no. 1, pp. 159-174, 1977.

[40] K. L. Gwet, "Computing inter-rater reliability and its variance in the presence of high agreement," *British Journal of Mathematical and Statistical Psychology,* vol. 61, no. 1, pp. 29-48, 2008.

[41] K. L. Gwet, "Testing the Difference of Correlated Agreement Coefficients for Statistical Significance," *Educational and Psychological Measurement,* vol. 76, no. 4, 2016.

[42] T. Ohyama, "Statistical inference of Gwet's AC1 coefficient for multiple raters and binary outcomes," *Communications in Statistics - Theory and Methods,* 2020.

[43] E. Cho, "Making Reliability Reliable: A Systematic Approach to Reliability Coefficients," *Organizational Research Methods,* pp. 1-32, 2016.

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

[44] A. Wieland, C. F. Durach, J. Kembro and H. Treiblmaier, "Statistical and judgmental criteria for scale purification," *Supply Chain Management,* vol. 22, no. 4, pp. 321-328, 2017.

[45] S. J. Zepeda and A. M. Jimenez, "Teacher Evaluation and Reliability: Additional Insights Gathered from Inter-rater Reliability Analyses," *Journal of Educational Supervision,* vol. 2, no. 2, pp. 11-26, 2019.

[46] K. Gwet, "Kappa Statistic is not Satisfactory for Assessing the Extent of Agreement Between Raters," *Statistical Methods for Inter-rater Reliability Assessment,* no. 1, 2002.

[47] N. Wongpakaran, T. Wongpakaran, D. Wedding and K. L. Gwet, "A comparison of Cohen's Kappa and Gwet's AC1 when calculating inter-rater reliability coefficients: a study conducted with personality disorder samples," *BMC Medical Research Methodology,* vol. 12, no. 61, 2013.

[48] A. M. Jimenez and S. J. Zepeda, "A Comparison of Gwet's AC1 and Kappa When Calculating Inter-Rater Reliability Coefficients in a Teacher Evaluation Context," *Journal of Education Human Resources,* vol. 38, no. 3, pp. 290-300, 2020.

[49] P. Pakray, S. Bandyopadhyay and A. Gelbukh, "Textual Entailment Using Lexical and Syntactic Similarity," *International Journal of Artificial Intelligence & Applications (IJAIA),* vol. 2, no. 1, pp. 43-58, 2011.

[50] A. Pawar and V. Mago, "Calculating the similarity between words and sentences using a lexical database and corpus statistics," 2018.

[51] D. Mazgutova and J. Kormos, "Syntactic and lexical development in an intensive English for Academic Purposes programme," *Journal of Second Language Writing,* vol. 29, pp. 3-15, 2015.

[52] B. Thompson and M. Post, "Paraphrase Generation as Zero-Shot Multilingual Translation: Disentangling Semantic Similarity from Lexical and Syntactic Diversity," in *Proceedings of the 5th Conference on Machine Translation (WMT)*, 2020.

[53] R. Fauzan, D. Siahaan, S. Rochimah and E. Triandini, "A Different Approach on Automated Use Case Diagram Semantic Assessment," *International Journal of Intelligent Engineering and Systems,* vol. 14, no. 1, pp. 496-505, 2021.

[54] R. Fauzan, D. Siahaan, S. Rochimah and E. Triandini, "Automated Class Diagram Assessment using Semantic and Structural Similarities," *International Journal of Intelligent Engineering and Systems,* vol. 14, no. 2, pp. 52-66, 2021.

[55] R. Fauzan, D. O. Siahaan, S. Rochimah and E. Triandini, "Novel Approach to Automated Behavioral Diagram Assessment Using Label Similarity and Subgraph Edit Distance," *Computer Science,* vol. 22, no. 2, pp. 191-207, 2021.

[56] X. Zhang, S. Sun and K. Zhang, "A New Hybrid Improved Method for Measuring Concept Semantic Similarity in WordNet," *The International Arab Journal of Information Technology,* vol. 17, no. 4, pp. 433-439, 2020.

[57] Z. Wu and M. Palmer, "Verb Semantics and Lexical Selection," in *ACL '94: Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, 1994.

[58] F. Husein and R. Sarno, "Developing Word Sense Disambiguation Corpuses Using Word2vec and Wu Palmer for Disambiguation," in *2018 International Seminar on Application for Technology of Information and Communication*, 2018.

[59] R. P. Honeck, "Semantic similarity between sentences," *Journal of Psycholinguistic Research,* vol. 2, p. 137–151, 1973.

[60] P. Sunilkumar and A. P. Shaji, "A Survey on Semantic Similarity," in *2019 International Conference on Advances in Computing, Communication and Control (ICAC3)*, 2019.

[61] J.-B. Gao, B.-W. Zhang and X.-H. Chen, "A WordNet-based semantic similarity measurement combining edge-counting and information content theory," *Engineering Applications of Artificial Intelligence,* vol. 39, pp. 80-88, 2015.

[62] A. M. Jacobs and A. Kinder, "Features of word similarity," arXiv, 2018.

ISSN 2502-3357 (online) | ISSN 2503-0477 (print)
8 (1) January 2022 10-23

[63] Z. Yuan, L. Yan and Z. Ma, "Structural similarity measure between UML class diagrams based on UCG," *Requirements Eng.,* vol. 25, p. 213–229, 2020.

[64] L. A. Zager and G. C. Verghese, "Graph similarity scoring and matching," *Applied Mathematics Letters,* vol. 21, no. 1, pp. 86-94, 2008.

[65] H. Bunke, "Exact (Graph) Matching," TU Wien, Szeged, 2013.

[66] M. Fey, J. E. Lenssen, C. Morris, J. Masci and N. M. Kriege, "Deep Graph Matching Consensus," arXiv, 2020.

[67] P. Swoboda, D. Kainmüller, A. Mokarian, C. Theobalt and F. Bernard, "A Convex Relaxation for Multi-Graph Matching," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[68] C. Liu, R. Wang, Z. Jiang and J. Yan, "Deep Reinforcement Learning of Graph Matching," arXiv, 2020.

[69] R. Hoffmann, C. McCreesh and C. Reilly, "Between Subgraph Isomorphism and Maximum Common Subgraph," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, 2017.

[70] C. Luo, X. Wang, C. Su and Z. Ni, "A Fixture Design Retrieving Method Based on Constrained Maximum Common Subgraph," *IEEE Transactions on Automation Science and Engineering,* vol. 15, no. 2, pp. 692-704, 2018.

[71] E. Duesbury, J. D. Holliday and P. Willett, "Maximum Common Subgraph Isomorphism Algorithms: A Review," *MATCH Communications in Mathematical and in Computer Chemistry,* vol. 77, no. 2, pp. 213-232, 2017.

[72] Y. Bai, D. Xu, A. Wang, K. Gu, X. Wu, A. Marinovic, C. Ro, Y. Sun and W. Wang, "Fast Detection of Maximum Common Subgraph via Deep Q-Learning," arXiv, 2020.

[73] H. Munawaroh, D. O. Siahaan, R. Fauzan and E. Triandini, "Structural Similarity Measurement using Graph Edit Distance-Greedy on State chart Diagrams," in *2020 2nd International Conference on Cybernetics and Intelligent System (ICORIS)*, 2020.

[74] K. Riesen, M. Ferrer and H. Bunke, "Approximate Graph Edit Distance in Quadratic Time," *IEEE/ACM Transactions on Computational Biology and Bioinformatics,* vol. 17, no. 2, pp. 483-494, 2020.

[75] F. Zulfa, D. O. Siahaan, R. Fauzan and E. Triandini, "Inter-Structure and Intra-Structure Similarity of Use Case Diagram using Greedy Graph Edit Distance," in *2020 2nd International Conference on Cybernetics and Intelligent System (ICORIS)*, 2020.

[76] K. Riesen and H. Bunke, "Improving Approximate Graph Edit Distance by Means of a Greedy Swap Strategy," in *International Conference on Image and Signal Processing*, Cham, 2014.

[77] K. Riesen and H. Bunke, "Graph Edit Distance — Novel Approximation Algorithms," in *Handbook of Pattern Recognition and Computer Vision*, 2016, pp. 275-291.