I.J. Modern Education and Computer Science, **2024, 1, 37-46**
Published Online on February 8, 2024 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2024.01.04

A Proposed Algorithm for Assessing and Grading Automatically Student UML Diagrams

Rhaydae Jebli*
SIGL Laboratory, ENSATE, Abdelmalek Essaadi University, Tetuan, Morocco
Email: rhaydaejeb@gmail.com
ORCID iD: https://orcid.org/0000-0001-6425-2651
*Corresponding Author

Jaber El Bouhdidi
SIGL Laboratory, ENSATE, Abdelmalek Essaadi University, Tetuan, Morocco
Email: jaber.elbouhdidi@uae.ac.ma
ORCID iD: https://orcid.org/0000-0003-3801-9316

Mohamed Yassin Chkouri
SIGL Laboratory, ENSATE, Abdelmalek Essaadi University, Tetuan, Morocco
Email: mychkouri@uae.ac.ma
ORCID iD: https://orcid.org/0000-0001-7256-329X

Received: 24 March, 2023; Revised: 25 May, 2023; Accepted: 19 June, 2023; Published: 08 February, 2024

Abstract: Digital technologies and innovative methods have shown a significant impact on educational systems, and have made work easier for both learners and teachers. Additionally, they have improved the quality and the capability to digitize the assessment of student work produced during a learning process. Assessing and scoring students' UML diagrams has become a challenging task for teachers, especially with the growing number of students, as well as the necessity to better manage their time. Consequently, there will be a necessity to automate the assessment of these learners. This paper presents an approach for assessing and grading automatically the student's UML diagrams. The approach uses an algorithm implemented in Java, which takes the tutor's and student's solution diagrams as input, then provides the student's scores and identifies differences and errors made. Our algorithm was tested and evaluated in a real case within a web platform, and the results obtained demonstrate the effectiveness of our solution.

Index Terms: Automatic assessment, UML diagrams, Algorithm, Educational Technologies, Distance education.

1. Introduction

Assessments of the learner's product are one of the most significant components of effective learning, they occupy a very important role in the education process, especially in higher education, from both the instructor's and learner's side. The common goal of assessment is to evaluate and improve student learning, analyze the learner's development in order to respond to pedagogical decisions, and achieve the learning outcomes set for each learning program.

The emergence of New Information and Communication Technologies in the field of training, aimed at enhancing students' abilities and skills [1] has led to the appearance of a new learning mode called e-learning. The latter refers to a distance learning process based on multimedia resources, which allows one or more people to be trained from their computer without geographical barriers. This learning mode is being integrated into educational processes as part of digital transformation in education, to enhance learning and teaching experiences [2] for all parties involved. They can benefit institutions and educational entities by automating internal operations and optimizing regular activities. [3] They are also evolving daily to assist teachers with the task of automating assessment, which has turned into a priority in the learning process, particularly in distance education, as it allows them to reach a larger audience than ever before. The insertion of automated assessment into an online learning environment can greatly improve the quality of our learning experience.

Unified Modeling Language (UML) is largely used for software engineering, and it is a standardized modeling language made up of an integrated set of diagrams to specify, visualize, and express the design of software and build a software product. [4] UML is supported by a variety of tools that provide powerful features for creating, editing, and

analyzing UML models. It is typically adopted in most courses of object-oriented software engineering. The assessment of student-developed UML models in universities is a real challenge for teachers and tutors, with a large number of students, and also the reason that UML models can be more complex and abstract, which requires a lot of time and resources.

Currently, automatic assessment of learner's products has become a popular topic in software engineering classes, and it has been the subject of much attention in recent years, particularly with the integration of e-learning platforms. [5] Automatic assessment has significant capabilities such as rapidity, flexibility, accuracy, and objectivity of assessment. [6] This would necessitate the use of automated tools and algorithms to analyze, assess, and grade students' assignments.

The objective of this study is to develop an effective solution for automatically assessing and grading students' UML diagrams. Many solutions exist for assessing UML diagrams, but they frequently require manual assessment by instructors or the use of limited automated tools with specific criteria. The proposed approach aims to overcome these limitations by implementing an algorithm that focuses on finding similarities and differences between the student's UML diagram and the tutor's UML diagram as a reference, then providing scores and feedback. We evaluate and test our algorithm on a real case, within a developed web platform invented by the authors. We conducted our work in this manner: students were assigned a UML class diagram to develop from textual descriptions for each exercise, these diagrams will be transformed into XMI files to be compared with a tutor model, to finally assess and provide student's scores and feedback.

The paper is organized as follows: it will begin by introducing and illustrating the context of student assessment in the distance learning environment. Next, the paper will provide an overview of the literature in the area of automatic assessment of student productions, especially the UML diagrams. The paper then will describe the research method and the different steps followed to compare and assess the student's UML diagram, the algorithm developed, and also the algorithm's processing phases. Next, the paper will present the implementation of our algorithm and discusses the result obtained. Lastly, the paper will be summarized by a conclusion and will present some perspectives and future works.

2. Student Assessment and Distance Learning

Over time, the assessment of learners has made remarkable developments, from both a theoretical and practical point of view, and now assessment is becoming an indispensable social discipline and an essential element in the educational environment, focusing on the learning progress and achievement of each student. It can be designed and implemented internally within the school or outside. It has many roles in the educational system, [7] and it must integrate grading, learning, and learner motivation.

Currently, learning platforms have been revolutionizing the way engineering and science are attempted and solved. They are web and multimedia tools intended for learning, rich in content and educational resources. They are being integrated into educational processes as part of the digital transformation of education.[8] As a result, universities have been obliged to change the way they manage educational programs in response to changing trends and technological advances. [9] Therefore, this clearly establishes the necessity to discuss the assessment of student production in these distance learning environments.

Having frequent assessments gives the students continuous feedback and enables them to take appropriate corrective measures, which is well-suited to fields that involve continuous learning and skills development. Today, digital assessment, which is the use of computer technology to support the learning assessment process, for formative and summative purposes, [10] has proven to be a reliable, globally consistent, and highly innovative tool for improving and enhancing learner outcomes. It is widely and quickly made accessible to students, and provides useful learning opportunities for students. [11] The digital assessment allows learners the possibility to practice their knowledge and skills on their own time whenever they are in a position to do this task. And also, it offers immediate feedback, which is not the case with traditional paper exams.

We aim in this work to develop an approach applicable in the higher education sector for assessing and grading automatically the student's productions, in order to digitize assessment of student work produced during a learning process, and we focus as a case study on the UML class diagram, since they are more complicated to assess, as they may have multiple solutions for the same problem [12] which is a huge pain for teachers and tutors to deal with. Our approach was evaluated and tested in our developed web platform.

3. Related Work

In this section, we present some approaches already studied by researchers in the area of automated assessment of UML diagrams. As in the study [13] the authors worked on calculating the similarity between two UML class diagrams using individual metrics that compare lexical names, and compound metrics that measure the similarity based on the combination of names and their neighborhoods. The findings, from two case studies, demonstrate the better performance of compound measures over individual measures. This research does not yet take into account all similar information, such as data types. The work [14] focuses on a developed algorithm based on the semantic aspects of the UML class diagram elements. The algorithm used consists of matching two diagrams' elements in accordance with their semantic significance, it requires human assistance at this point, then a distance is calculated between each pair, and

added to a difference vector, and its length is predicted to store the difference as results. The authors of [15] propose an automated method for grading class diagrams using metamodels, the method consists of using syntactic, semantic, and structural matching. The syntactic matching is calculated using a Levenshtein distance, the semantic matching is calculated using the WordNet database, and the structural matching covers the property similarity. The result shows that their tool was able to automatically grade 20 students giving a 14% difference compared to the grade received by the instructor. In this study, [16] the Edit Distance is applied to determine the similarity between sequence diagrams with the use of dynamic programming to increase the relevance of the matching of two or more sequence diagrams. The authors of this work, [17] measure the similarity between two use case diagrams, using the cosine similarity applied for the metadata of the diagram. This study has not measured the similarity between the types of relations. The authors of this work [18] propose a developed algorithm to automate UML class diagrams grading and provide feedback files containing scores and errors area. In this study, [19] the authors explain a research method based on a machine learning approach to automatically rate and score student's UML class diagrams, the method consists of training a regression and multiple classification models, based on two experiments that construct a prediction model from data collected from bachelor students works. The precision of their research result is 69%.

The next sections describe our research methodology and our novel developed algorithm for automatically assessing and grading students' UML diagrams.

4. Research Methodology

Our aim is to assess and grade automatically the student's UML diagram, and also produce detailed instructions based on the differences found between the student's UML diagrams and a tutor's UML diagrams. In these sub-sections, we explain our approach, and the steps followed.

4.1 Approach and Architecture

Our objective is to develop an approach for assessing the student's UML diagram in a simple and efficient way, optimize the correction time, facilitate the tasks of teachers while improving the grading process, [20], and also deliver an objective and coherent assessment of the student's contribution. As a case study, we focus this paper on the assessment of students' UML class diagrams. Figure 1 illustrates the global architecture of our approach, which is generally based on two aspects that represent our contribution: the first concerns the user interface for the teacher and the student. The second aspect concerns the development of a new algorithm that can assess and score the student's UML diagrams.

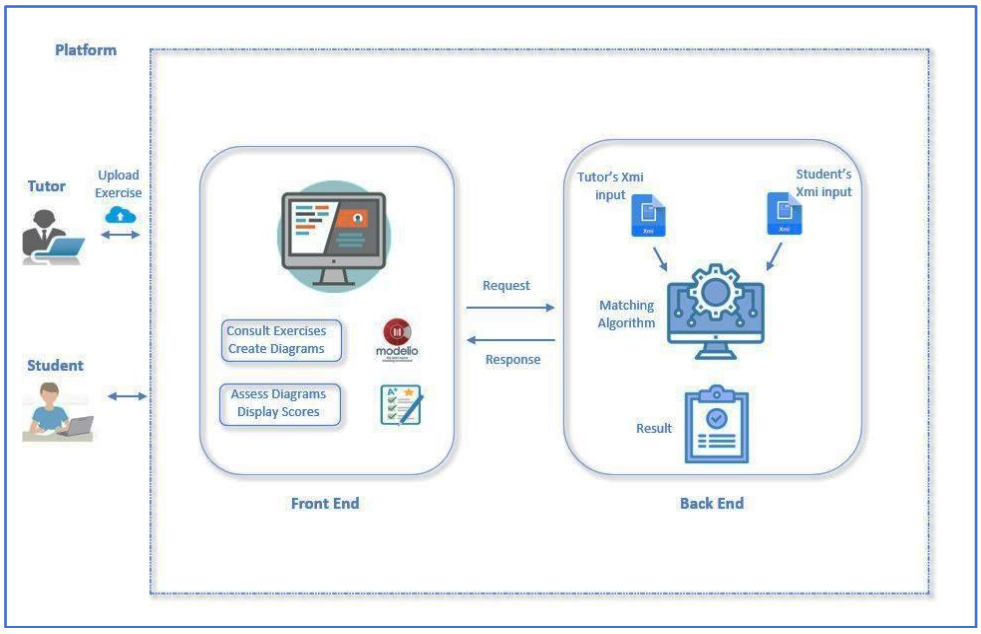


Fig.1. Architecture of our platform

The methodology followed consists of different steps. Students were given a UML class diagram to develop from textual descriptions, the modeling software chosen to design and develop the diagrams is Modelio, [21] which is an open-source drawing tool. Then this diagram is transformed into an XMI file and parsed using a DOM API to extract the required elements of a diagram. Therefore, the elements extracted are grouped in lists depending on their categories (classes, attributes, operations, relationships…). Next, the lists of each category are compared to provide the differences found. Finally, detailed differences found between the student's diagram and the tutor's diagram is generated, and also the student's mark is produced based on a special criterion assessment.

The algorithm illustrated in Figure 2 is designed to assess and grade the UML diagrams produced by students for a specific problem and a given tutor's solution. The algorithm takes as input the student's file. XMI and the tutor's file.XMI. As output, the algorithm produces the final mark of students with a detailed instruction that contains the differences found between the two diagrams. The process of the algorithm is as follows:

For each elaborated diagram:

- Extract elements of the tutor's diagram.
- Extract elements of the student's diagram.
- Create lists of elements depending on their category.
- Compare the lists of elements to identify the differences between the students and the tutor.
- Display detailed differences found between the student's diagram and the tutor's diagram.
- Adapt the result of comparison with an assessment criterion to give the students the final marks and critical feedback.

Algorithm: Automatic Assessment of UML Diagram

```
Start
Input : Student's file.xmi , Tutor's file.xmi
Outputs : detailed differences, student's grade
1: repeat for all diagram's elements
2:   for all classes in student's file.xmi   do
3:     Liste_Student_Classes ⟵ student's file.xmi.getClasses()
4:   for all classes in tutor's file.xmi   do
5:     Liste_Tutor_Classes   ⟵ tutor's file.xmi.getClasses()
6:   for all attributes in student's file.xmi   do
7:     Liste_Student_Attributes ⟵ student's file.xmi.getAttributes ()
8:   for all classes in tutor's file.xmi  do
9:     Liste_Tutor_Attributes ⟵ tutor's file.xmi.getAttributes()
…
10: repeat functions for all diagram's elements
11:          Function          CompareClasses(Liste_Student_Classes,
Liste_Tutor_Classes)
12:    if match then no difference found
13:    else then name and number of difference
14:         Function    CompareAttributes    (Liste_Student_Attributes,
Liste_Tutor_Attributes)
15:    if match then no difference found
16:    else then name and number of difference
…
17: detailed differences
18: assessment criteria
19: student final mark
End
```

Fig. 2. Algorithm of UML diagram assessment

Figure 3 presents the flowchart that describes the algorithm's processing phases.

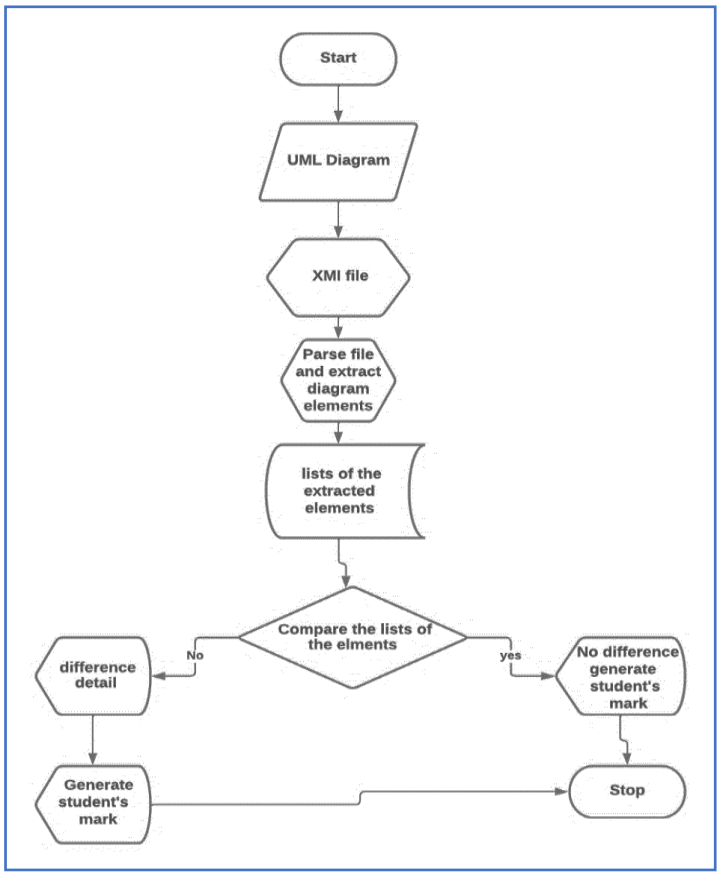


Fig.3. Flowchart pseudocode

4.2 UML Modelling Tool

Today, a variety of tools are available for modeling UML diagrams and providing detailed models with particular symbols and features that explain how any software or hardware system operates. After detailed research and a careful study of a list of tools, we choose to use a tool from this list that meets our demands and fits our criteria. The tool is Modelio which is available as an open-source UML diagram tool developed in java, it creates many types of UML diagrams, offers an XMI import/export feature, and export diagrams in SVG, Gif or JPEG format. Modelio gives a robust Java API for accessing metamodels [22].

4.3 Parsing File

XML (Extensible Markup Language) was designed to store and manage data in plain text format and in a standardized way. Analyzing and processing XML documents is a crucial step across most XML applications. This operation is called XML parser, which is a software library that offers a software developer an API for communicating with the XML document. [23] Parsers process an XML document to read and extract useful information from it. And the most XML parsers that are widely used are Simple API for XML (SAX) and Document Object Model (DOM).

The first one is a stream-based API, which implements SAX API that manages the processing of an XML document using events. SAX reads an XML document as steam and utilizes callback functions as internal procedures. [24] It consumes significantly less memory than DOM. Rather, SAX simply transmits data to the application as it is read.

The second one is a tree-based API. It defines an interface that allows applications to have random access to the information data. The entire document is loaded into memory during processing, and a tree of nodes is constructed. [25] The tree includes every element of the document. The root node is in the head of the tree, the other nodes have a parent's node, a list of child nodes if exist, and also a list of attributes of each node element. These nodes can be accessed by nodeName and nodeValue proprieties.

In our approach, the files were parsed using the DOM API. Because it's the most suitable for manipulating our XMI file and extracting all the elements required in our comparison.

4.4 Matching Elements

As described earlier, the matching process consists of comparing each list of the student's diagram elements with each list of the tutor's diagram elements and then producing the differences found between them. Sometimes, some challenges are facing us during this matching process, such as the use of abbreviations, synonyms, using special

symbols, and incorrect spelling. To reinforce our comparison process, some algorithms [26,27] of natural language processing, will be used to deal with these issues. Like using stemming and lemmatization for removing part of the words to their root, removing some special symbols, using databases of synonyms, and lowercase conversion.

For implementing the stemming algorithm, we used the library for Java Apache. Lucene, which is a powerful library for search features. The stemming algorithm transforms a word into its root form, by removing suffixes such as the "ing", "ed"," s" and "ly". Lemmatization, as distinct from stemming, uses lexical knowledge bases to obtain the correct basic forms of words. For its implementation, we used the library Stanford CoreNLP, which is a Java-based library for natural language processing that gives basic word forms, provides a set of natural language analysis tools, and generates extensive linguistic annotation. Therefore, some special symbols should be removed from the word such as the symbols: (&, _, %, -, @,", ...). And also, the Wordnet database was used for finding some synonyms.

The following table presents some examples after applying the different algorithms.

Table 1. Example of Words after Applying Algorithms

Algorithm	Example
Lemmatization	Children →child
Stemming	Programs,Programming,Programmed→program
Removing special symbols	Library_System → librarysystem
Wordnet database for synonyms	Customer and Client

5. Result and Discussion

The approach was evaluated and tested in a real case, within a developed web platform invented by authors and dedicated to the assessment of student's UML diagrams, where a tutor can upload as many as UML class diagrams exercises with textual specifications, to finally assess student's diagrams for each exercise. The result shows that the algorithm can provide efficient student grades and detailed instructions based on the differences founds between the student's diagram and the tutor's diagram solution. We plan as future work to indicate places of errors on the student's modeled diagram and also assign course guidance for each mistake made to increase students' knowledge and skills.

We carefully selected a set of representative exercises. Students connect to the platform to solve the exercises proposed by the teacher and upload their solutions. Our System automatically corrects the solutions proposed by the students, comparing them with those proposed by the teacher.

In this part, we used a UML class diagram for Library Management System. This diagram defines the structure of the system by listing the classes of the system, its attributes, the operations, the relationships, and the other elements of the system.

Figure 4 displays the screenshot of the tutor's diagram.

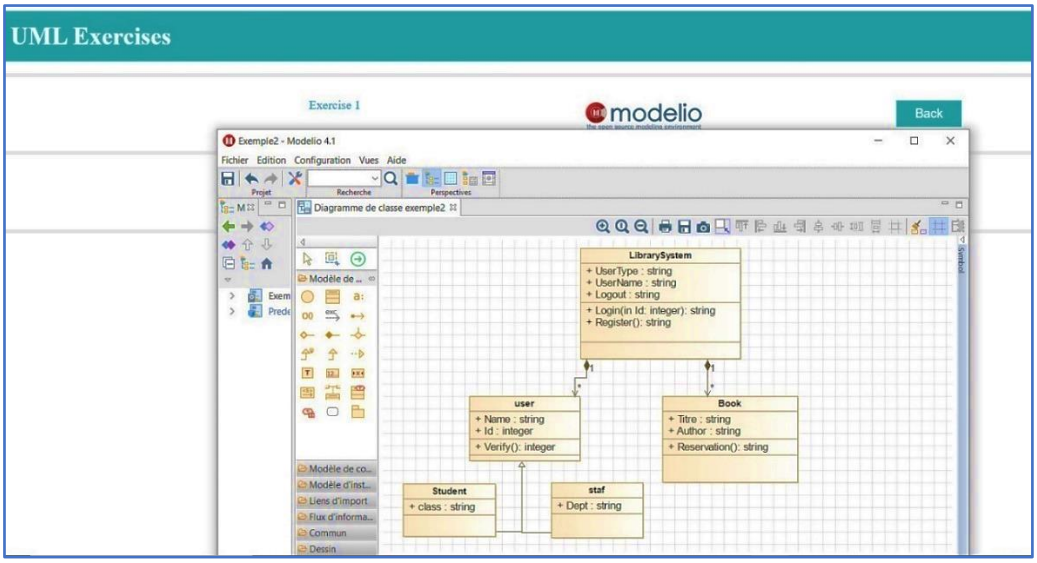


Fig.4. Screenshot of tutor's diagram

After consulting the list of exercises proposed by the tutor, the student starts by creating and modeling his diagram for each exercise. Figure 5 illustrates a screenshot of the UML class diagram created by the student for the exercise: Library Management System.

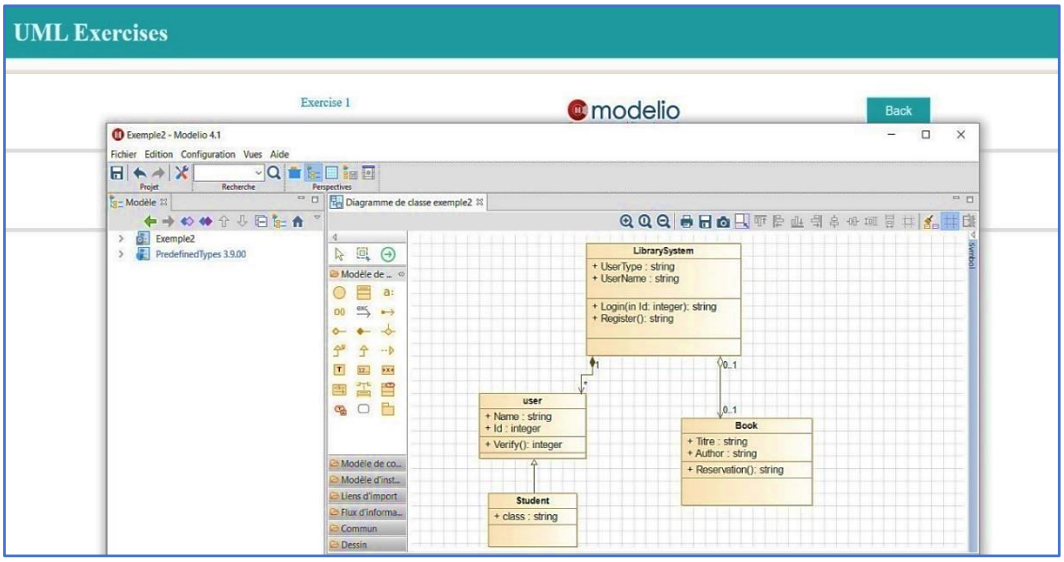


Fig.5. Screenshot of student's diagram

Once the diagrams have been created and the files have been generated, for both the student and the tutor. The tutor determines the student's name and the name of the exercise that needs to be corrected. Figure 6 presents a screenshot of the tutor's input file.

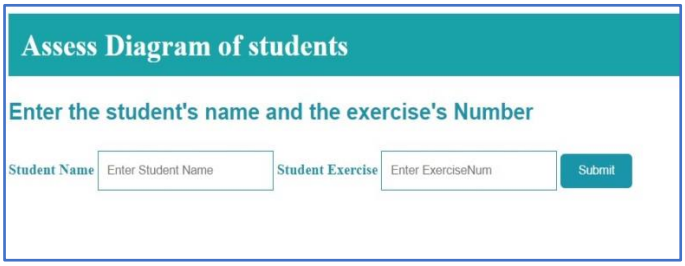


Fig.6. Screenshot of tutor's input file

In the next step, our algorithm compares the student's file and the tutor's file, depending on the exercise chosen, then produces the result of comparison with detailed constructions according to the differences found.

Figure 7 presents the student's UML class diagram, and the different lists containing the tutor's diagram elements with their total numbers.

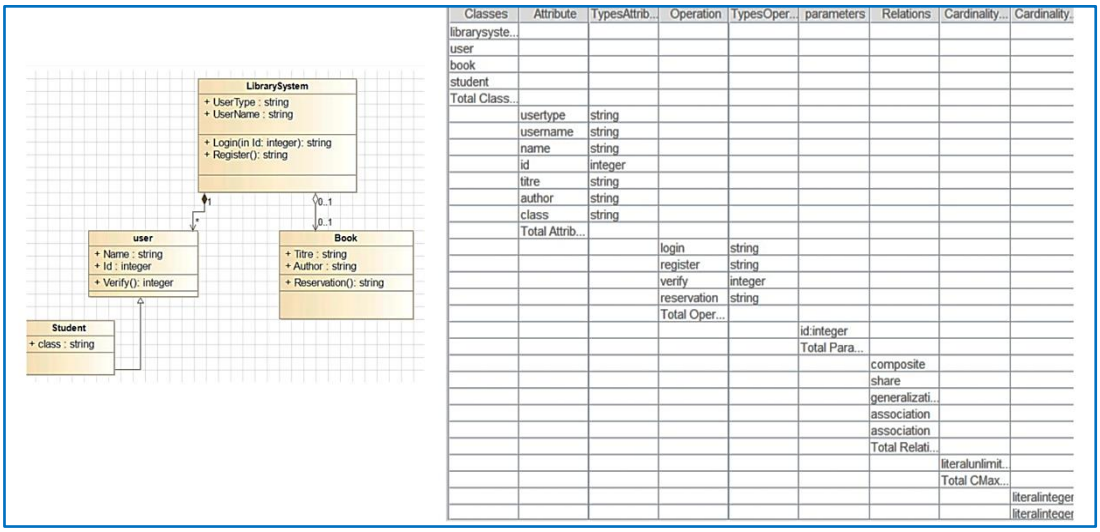
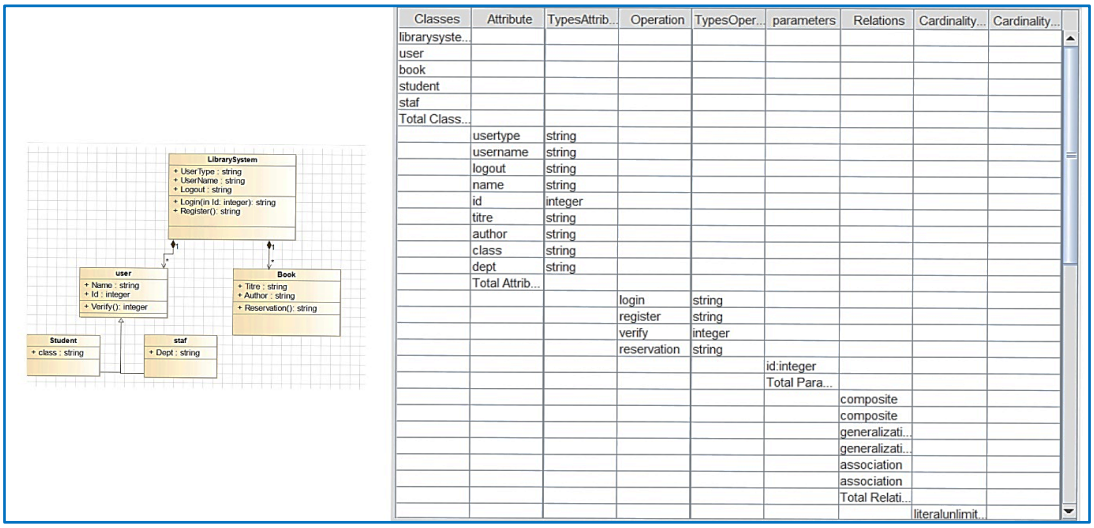


Fig.7. Screenshot of student's UML class diagram and lists of diagram elements

Figure 8 presents the tutor's UML class diagram, and the different lists containing the student's diagram elements with their total numbers.



Classes	Attribute	TypesAttrib...	Operation	TypesOper...	parameters	Relations	Cardinality...	Cardinality...
librarysyste...								
user								
book								
student								
staf								
Total Class...								
	usertype	string						
	username	string						
	logout	string						
	name	string						
	id	integer						
	titre	string						
	author	string						
	class	string						
	dept	string						
	Total Attrib...							
			login	string				
			register	string				
			verify	integer				
			reservation	string				
					id:integer			
					Total Para...			
						composite		
						composite		
						generalizati...		
						generalizati...		
						association		
						association		
						Total Relati...		
							literalunlimit...	

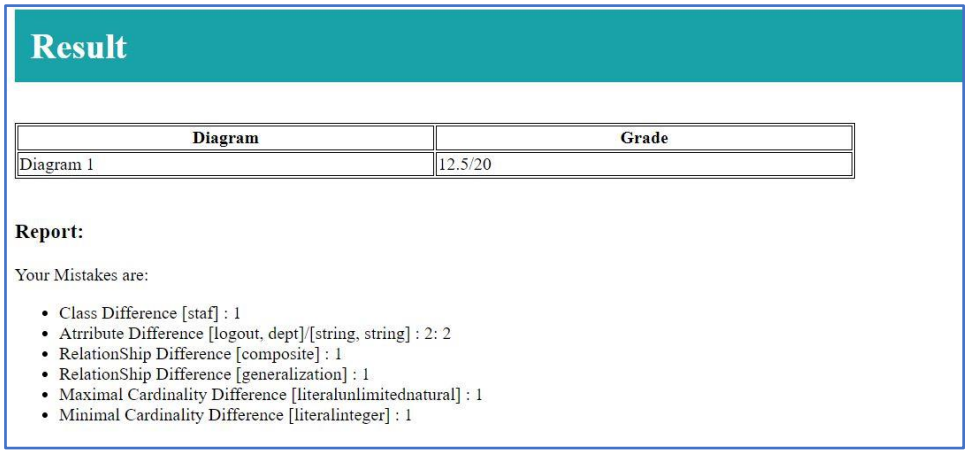
Fig.8. Screenshot of tutor's UML class diagram and lists of diagram elements

Our algorithm produces a table of comparison describing the differences found between the student's diagram and the tutor's diagram. Figure 9 displays these differences based on their categories, names, and numbers.

Difference element Type	Difference	Difference Number
Class Difference	[staf]	: 1
Atrribute Difference	[logout, dept]/[string, string]	: 2: 2
RelationShip Difference	[composite]	: 1
RelationShip Difference	[generalization]	: 1
Maximal Cardinality Differe...	[literalunlimitednatural]	: 1
Minimal Cardinality Differen...	[literalinteger]	: 1

Fig.9. Screenshot of the differences between the two UML diagrams

The student's scores will be calculated depending on the differences found and related to a grading scale proposed by the teacher which serves as an example at this stage, given that it can be modified and developed over time, and then presented as a grade /20. Figure 10 shows the student's grade and the difference in detail from the previous comparison. The score varies according to the type of element.



Result

Diagram	Grade
Diagram 1	12.5/20

Report:

Your Mistakes are:

- Class Difference [staf] : 1
- Atrribute Difference [logout, dept]/[string, string] : 2: 2
- RelationShip Difference [composite] : 1
- RelationShip Difference [generalization] : 1
- Maximal Cardinality Difference [literalunlimitednatural] : 1
- Minimal Cardinality Difference [literalinteger] : 1

Fig.10. Screenshot of the student's grade

The implementation stage was necessary to integrate our algorithm into a real application and guarantee its reliability and its usability. Experimental results demonstrated the effectiveness of the algorithm in automatically assessing student class diagrams, while reducing the workload of teachers to benefit from valuable time gained, avoiding the need to manually evaluate each student diagram.

6. Conclusion

UML diagrams are generally the most widely used in academic environments for the various programming environments and development processes. However, manual evaluation of these diagrams presents a huge pain for teachers and tutors to deal with. Automating assessment practices seems to be an effective way to enhance an educational process. This article presents an approach aimed at automatically assessing and grading students' UML diagrams. The approach implements an algorithm that inputs solution diagrams from the tutor and the student, then generates the student's grades and identifies differences and errors. We evaluate and test our approach on real case scenarios, on examples of student diagrams compared with teacher diagrams using our algorithm. The results obtained have conclusively demonstrated that our approach has achieved its objectives and demonstrated its ability to deliver consistent and accurate assessments for automatic assessment of student class diagrams. We plan in future work to provide feedback to students to indicate places of errors on the student diagram created. We intend also to extend our implementation to other types of diagrams.

Conflict of Interest

The authors declare no conflict of interest.

References

[1] M. Kamil Budiarto, T. Rejekiningsih, and Sudiyanto, "Implementation of Computer-assisted Learning in High School: Teachers and Students‟ Perspective," *Int. J. Educ. Manag. Eng.*, vol. 11, no. 4, pp. 26–34, 2021.

[2] L. Salamat, G. Ahmad, M. I. Bakht, and I. L. Saifi, "Effects of E-Learning on Students' Academic learning at university Level," *Assian Innov. J. Soc. Sci. Humanit.*, vol. 2, no. 2, pp. 1–12, 2018, [Online]. Available: https://www.researchgate.net/publication/326461349.

[3] C. G. Demartini, L. Benussi, V. Gatteschi, and F. Renga, "Education and digital transformation: The 'riconnessioni' project," *IEEE Access*, vol. 8, pp. 1–34, 2020, doi: 10.1109/ACCESS.2020.3018189.

[4] 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," *Regist. J. Ilm. Teknol. Sist. Inf.*, vol. 8, no. 1, pp. 10–23, 2022, doi: 10.26594/register.v8i1.2248.

[5] H. Simanjuntak, "Proposed framework for automatic grading system of ER diagram," *Proc. - 2015 7th Int. Conf. Inf. Technol. Electr. Eng. Envisioning Trend Comput. Inf. Eng. ICITEE 2015*, pp. 141–146, 2015, doi: 10.1109/ICITEED.2015.7408930.

[6] M. Amelung, K. Krieger, and D. Rösner, "E-assessment as a service," *IEEE Trans. Learn. Technol.*, vol. 4, no. 2, pp. 162–174, 2011, doi: 10.1109/TLT.2010.24.

[7] I. Stăncescu, "The Importance Of Assessment In The Educational Process - Science Teachers' Perspective," no. July, pp. 753–759, 2017, doi: 10.15405/epsbs.2017.07.03.89.

[8] E. O. Adebayo, "Efficacy of Assistive Technology for Improved Teaching and Learning in Computer Science," *Int. J. Educ. Manag. Eng.*, vol. 12, no. 5, pp. 9–17, 2022.

[9] P. Mashau and J. Nyawo, "The use of an online learning platform: a step towards e-learning," *South African J. High. Educ.*, vol. 35, May 2021, doi: 10.20853/35-2-3985.

[10] I. Adeshola and A. M. Abubakar, "Assessment of Higher Order Thinking Skills," pp. 153–168, 2020, doi: 10.4018/978-1-7998-2314-8.ch008.

[11] A. Barana, M. Marchisio, and M. Sacchet, "Advantages of using automatic formative assessment for learning mathematics," *Commun. Comput. Inf. Sci.*, vol. 1014, pp. 180–198, 2019, doi: 10.1007/978-3-030-25264-9_12.

[12] S. Zougari, M. Tanana, and A. Lyhyaoui, "Towards an automatic assessment system in introductory programming courses," *Proc. 2016 Int. Conf. Electr. Inf. Technol. ICEIT 2016*, pp. 496–499, 2016, doi: 10.1109/EITech.2016.7519649.

[13] M. A.-R. Al-Khiaty and M. Ahmed, "UML Class Diagrams: Similarity Aspects and Matching," *Lect. Notes Softw. Eng.*, vol. 4, no. 1, pp. 41–47, 2016, doi: 10.7763/LNSE.2016.V4.221.

[14] 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," *ICSEA 2015 Tenth Int. Conf. Softw. Eng. Adv.*, no. 342, pp. 147–152, 2015, doi: 10.13140/RG.2.1.3104.4889.

[15] W. Bian, O. Alam, and J. Kienzle, "Automated grading of class diagrams," *Proc. - 2019 ACM/IEEE 22nd Int. Conf. Model Driven Eng. Lang. Syst. Companion, Model. 2019*, pp. 700–709, 2019, doi: 10.1109/MODELS-C.2019.00106.

[16] A. Adamu and W. M. N. W. Zainon, "Similarity assessment of uml sequence diagrams using dynamic programming," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10645 LNCS, pp. 270–278, 2017, doi: 10.1007/978-3-319-70010-6_25.

[17] R. Fauzan, D. Siahaan, S. Rochimah, and E. Triandini, "Use case diagram similarity measurement: A new approach," *Proc. 2019 Int. Conf. Inf. Commun. Technol. Syst. ICTS 2019*, pp. 3–7, 2019, doi: 10.1109/ICTS.2019.8850978.

[18] S. Modi, H. A. Taher, and H. Mahmud, "A Tool to Automate Student UML diagram Evaluation," *Acad. J. Nawroz Univ.*, vol. 10, no. 2, pp. 189–198, 2021, doi: 10.25007/ajnu.v10n2a1035.

[19] D. R. Stikkolorum, P. Van Der Putten, C. Sperandio, and M. R. V. Chaudron, "Towards automated grading of UML class diagrams with machine learning," *CEUR Workshop Proc.*, vol. 2491, pp. 1–13, 2019.

[20] B. P. Cipriano, N. Fachada, and P. Alves, "Drop Project : An automatic assessment tool for programming assignments," *SoftwareX*, vol. 18, p. 101079, 2022, doi: 10.1016/j.softx.2022.101079.

[21] "Modelio." https://www.modelio.org/.
[22] R. Jebli, J. El Bouhdidi, and M. Y. Chkouri, "A Proposed Architecture of an Intelligent System for Assessing the Student's UML Class Diagram," *Int. J. Emerg. Technol. Learn.*, vol. 16, no. 21, pp. 4–12, Nov. 2021, doi: 10.3991/ijet.v16i21.25105.
[23] V. M. Deshmukh and G. R. Bamnote, "An empirical study of XML parsers across applications," *Proc. - 1st Int. Conf. Comput. Commun. Control Autom. ICCUBEA 2015*, pp. 396–401, 2015, doi: 10.1109/ICCUBEA.2015.83.
[24] M. V. M. Deshmukh and D. G. R. Bamnote, "An Empirical Study: XML Parsing using Various Data Structures," *Int. J. Comput. Sci. Appl.*, vol. 6, no. 2, pp. 400–405, 2013.
[25] J. Holm, "XML Parsers - A comparative study with respect to adaptability," 2018.
[26] V. Vachharajani, J. Pareek, and S. Gulabani, "Effective label matching for automatic evaluation of use - Case diagrams," *Proc. - 2012 IEEE 4th Int. Conf. Technol. Educ. T4E 2012*, pp. 172–175, 2012, doi: 10.1109/T4E.2012.33.
[27] R. Fauzan, D. Siahaan, S. Rochimah, and E. Triandini, "Automated Class Diagram Assessment using Semantic and Structural Similarities," *Int. J. Intell. Eng. Syst.*, vol. 14, no. 2, pp. 52–66, 2021, doi: 10.22266/ijies2021.0430.06.

Authors' Profiles

Rhaydae Jebli is a PhD student at the SIGL Laboratory, University Abdelmalek Essaadi, National School of Applied Sciences Tetouan-Morocco. She holds a master's degree (2017) from University Abdelmalek Essaadi and BSC Degree (2015) from University Mohamed First. Her research interests are educational technologies, e-learning environments, machine learning.

Jaber El Bouhdidi is an HDR Professor of Computer Science at the SIGL Laboratory, University Abdelmalek Essaadi, National School of Applied Sciences –Tetouan -Morocco. His research interest includes Web Semantic, multi-agents' systems, e-learning Adaptive Systems and Big data. He has several papers in international conferences and journals.

Mohamed Yassin Chkouri is a professor of Computer Science at the SIGL Laboratory, University Abdelmalek Essaadi, National School of Applied Sciences –Tetouan -Morocco, he is Head of Computer Engineering Department, SIGL Research Laboratory Director and Academic coordinator Erasmus + e-VAL project.

How to cite this paper: Rhaydae Jebli, Jaber El Bouhdidi, Mohamed Yassin Chkouri, "A Proposed Algorithm for Assessing and Grading Automatically Student UML Diagrams", International Journal of Modern Education and Computer Science(IJMECS), Vol.16, No.1, pp. 37-46, 2024. DOI:10.5815/ijmecs.2024.01.04