An integration of uml use case diagram and activity diagram with Z language for formalization of library management system

Zainab Hassan Muhamad[1], Dhafer Abdulameer Abdulmonim[2], Bashar Alathari[3]
[1,2] Department of Computer Science, Directorate of Education in Najaf, Iraqi Ministry of Education, Iraq
[3]ITRDC, University of Kufa, Iraq

Article Info

Article history:

Received Jun 28, 2018
Revised Mar 22, 2019
Accepted Apr 3, 2019

Keywords:

Formal method
Formal specification
Unified modeling language
Z language

ABSTRACT

Unified Modeling Language (UML) is the effective standard for modeling object-oriented software systems. However, the ambiguity of semantics and the absence of consistency among UML diagrams lead to lack of precisely defining the requirements of a system. On the other hand, formal methods are techniques and tools use the mathematical notations, and they involve the precise syntax and semantics of the unambiguous software requirements specification. It applied in early stages of Software Development Life Cycle (SDLC). Therefore, an integrated between UML specification and formal specification is required to reduce the requirements' ambiguity and error, and to improve the quality and security of software systems. This paper proposes an approach involves the combining UML use-case diagram and activity diagrams with Z language for formalization of Library Management System (LMS). The focus of this paper is on consistency between the UML diagrams to Z Schema, and then verified by using the Z / EVEs tool.

Corresponding Author:

Dhafer Abdulameer Abdulmonim,
Department of Computer Science,
Directorate of Education, Iraqi Ministry of Education,
Najaf, Iraq.
Email: Dhafer985@gmail.com

1. INTRODUCTION

Unified Modeling Language (UML) is a visual language for constructing and modeling of software systems. It combines a graphical notation with a semi-formal language for object-oriented based system, to provide requirements specifications with syntax and semantics constraints [1, 2]. UML diagrams are considered as a standard tool to compose requirements, and then transform them into structures. In order to, understand system's requirements and reduce complexity of the system. UML includes thirteen diagrams: Use Case, Class, Activity and others. However, there is the lack of precision in the definition of UML notations and semantics [3]. Consequently, UML diagrams have a different interpretation of the described model. In addition to, the consistency between models is very critical. Therefore, a consistency and a concise formalization of UML semantics is required for mapping between graphical and logical by using mathematical theories [4]. In Software Engineering, the integrated formal specifications and UML specifications are very useful to understand the requirements specifications of any system by combination of UML diagrams with formal approaches [5, 6]. With emphasis on consistency between UML diagrams. Formal methods are techniques based on mathematical notations, which play a critical role as they focus on refinement of requirements in the initial stages of the System Development Life Cycle (SDLC) like the requirements analysis and specification. It used to ensure that the requirements specifications are precisely express. In order to refinement of a system's requirements which therefore increase the system’s accuracy and consistency [7]. Formal methods specify the unambiguous system specification, reduce the error and improved effectiveness in design and development phases [8, 9]. The formal specification removes ambiguity, reduces the design time, it also addresses improve software's reliability and comprehensibility [10]. Formal specification languages are widely used for development an accurate, consistent and unambiguous systems and software. On that point are various formal languages like VDM, B-Methods, Larch and Z notation are used for formal specification. Z notation is a good example based on formal specification language which uses the set theory for determining the behavior of sequential systems [11]. The literature indicates that the most well-known issue for UML is lack of obvious semantics, various interpretation of the diagram and inconsistency among diagrams. This leads to ambiguity, errors, and reduced software quality. in addition, the most popular UML diagrams used in industry are use case, class, sequence to integrate with formal methods. But, The formal specifications have not been made from activity diagram. Hence, the combination of informal method and formal method is required to handle ambiguity, specifying consistency and improving completeness [12]. Therefore, this paper proposes an approach that aims to integrate of UML use case and activity diagrams with formal specification method, to bridge a gap between informal and formal methods. The main focus of this paper is on consistency and complement among UML use case diagram, UML activity diagram and Z schemas. Besides, there was a research gap of proper integration of activity diagrams with formal specifications. Also, Z notation used for formal analysis of LMS (LMS) which is further verified by using the Z/EVES tool.

The rest of this paper organized as follows. Section 2 introduces the related work. Section 3 fully describes the proposed approach to integrate of UML use case and activity diagrams with formal specification method. Section 4 presents the results of the case study to demonstrate the effectiveness of the proposed approach. Section 5 describes the strength and weakness of the proposed approach. and finally Section 6 remarks the conclusions.

2. RELATED WORKS

The formal approach for requirements specification using Z language that had been addressed by many researchers [4, 6–8, 11, 12], they were offered it in different aspects. The solution has produced with greater improvement than informal method. Minhas, et al. [4] proposed an integration of UML specification with formal specification. They have been used UML sequence diagram and Z language for developed. The Flight Reservation System. In addition, they concluded to use more than one UML diagram, in order to the results can be more improved. To cover ambiguity, identifying consistency and enhancing completeness, the informal and formal method combination is proper to develop the Road Traffic Management System by Singh, et al. [6]. In their study UML use case, class and sequence diagrams with Z language used. Moreover, the Z/EVEs tool used to verify Z schemas. In the research provided from Bakri et al. [7], related to our research study where they offer a formal specification for the inventory system using Z language. Their approach based on translating the UML specification into Z schemas. It also focused on the consistency between UML specification and Z schema, in order to effectively improve system reliability and reduce defect for developing the inventory system. However, in their study have been used UML class diagram. Researchers presented in [8, 11, 12] formal specifications by using UML use case diagram and Z schemas to improve software correctness, reliability, and efficiency. As it used to reduce time and cost at an initial stage. However, in [8] have been used Z/EVEs tool to verify Z schemas. While, in both of the studies [11, 12] no tool used to verify Z schemas. The comparative study of related works focused on a number of formal specification approaches to improve the systems development as reviewed in Table 1.

Table 1. Review of related works

Author name/ year	A Proposed Approach			The results obtained from the combination of informal and formal specification
	Supported UML diagrams	Formal Method	Verification	
S. H. Bakri, *et al.* 2013 [7]	Use case, Class	Z language	Z/EVEs tool	Improve the system quality and reduce cost
M. W. Azeem, *et al.* 2014 [11]	Use case	Z language	___	Improve the quality efficiently, fixing bugs and reduce time and cost
N. M. Minhas, *et al.* 2015 [4]	Sequence	Z language	Fuzz tool	The integrating of sequence and other UML diagram is more effective
S. A. Khan & H. Jamshed, 2016 [12]	Use case	Z language	___	Improve correctness, reliability, user friendliness and efficiency.
M. Singh, *et al.* 2016 [6]	Use case, Class, Sequence	Z language	Z/EVEs tool	Unambiguity, consistency and completeness
P. Saratha, *et al.* 2017 [8]	Use case	Z language	Z/EVEs tool	Unambiguous system specification, reduce the error and improved effectiveness

Based on Table 1, the literature indicates that the most popular UML diagrams used in industry are use case, class, sequence. The formal specifications did not construct from activity diagram as well. Therefore, this paper is used combination of UML use case and activity diagram with Z formal specification language to design the LMS. In addition to, we have discussed earlier that the most well-known issue for UML is lack of clear semantics, different interpretation of the described model and inconsistency among models. Hence, the consistency between UML diagrams and Z schema focused in this paper.

3. RESULTS AND ANALYSIS

UML is a common modelling method in an object-oriented based system. The consistency is a significant rule between diagrams, to ensure that implementation of the diagrams is complementing. Moreover, the consistency between use-case diagram and activity diagram confirmed by [13,14]. On the other hand, Formal methods are an essential solution to the issues related to requirement analysis and specifications. In addition, it describes of unambiguity requirements specifications, and ensures that the system indeed satisfies the requirements, reliability and corrects [10]. Consequently, this paper aims to propose an approach that integrates UML diagrams (use case diagram and activity diagram) with the formal specification language (Z language) for specifying requirements for LMS, and then verified by using Z/EVES type checker tool. This tool used for verifying the schema specification, that wrote based on Z notation language [7]. This verification includes syntax and semantics of the LMS formal specification. The proposed approach for designing LMS uses UML and the formal method as shown in Figure 1.

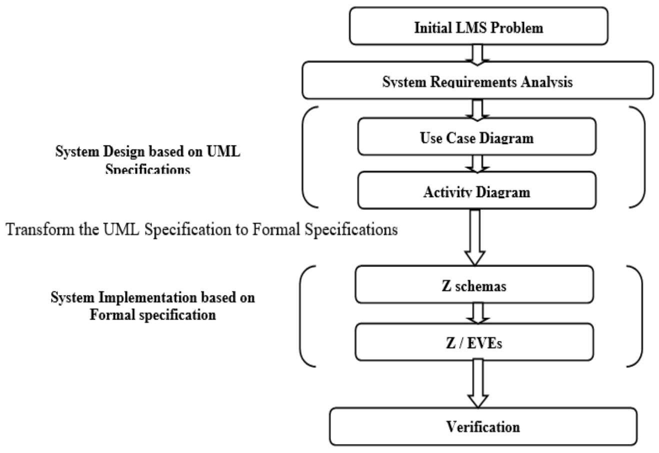

Initial LMS Problem
System Requirements Analysis
System Design based on UML Specifications
Use Case Diagram
Activity Diagram
Transform the UML Specification to Formal Specifications
System Implementation based on Formal specification
Z schemas
Z / EVEs
Verification

Figure 1. The proposed approach steps to integrate UML specification with formal specification

4. CASE STUDY AND APPROACH MODELLING

In this paper, we present a case study of A Library Management System (LMS). A developer wants to develop LMS. When a user wants to borrow a book, the system should be able to verify the membership of the user first, if the user is a member, then the system checks the availability of the book. If the book is available, then the system lets the member reserves the book and the database needs to update to indicate that the book is not available anymore. The first step is to determine the requirements analysis for the scenario as in the next subsection.

4.1. System requirements analysis

The analysis stage was the basis of the system design. At this stage, it needs to obtain the user requirements as accurate, and analysis of demand, thus to specify system requirements. The system requirements analysis is as follows:

The LMS entrusted by book borrowing and book return business. Book borrows management need to verify the membership of the user and check the availability of the book. As book return management need to verify the membership of the user and the legality of the book. Librarian task was to maintain the new book information addition. The stakeholders of the system are Librarian and Member. The system has basic functions that are Verify membership, Check a book availability, Borrow a book, Return a book and Add a book.

4.2. System design based on UML specifications

Based on the above analysis, we then draw the diagrams of UML mainly the use case diagram and activity diagram. In this paper, the Rational Rose UML tool used to create a complete UML diagrams. Figure 2 shows use case diagram for the LMS. There are two users to use the system, which are Librarian and Member. A Librarian can do activities for Borrow a book, Return a book and Add a book. As a Member can do activities for Borrow a book and Return a book. While, the functions Verify membership and Check a book's availability are included functions of the system. Based on the use case diagram in Figure 2, there are three main functions are Borrow a book, Return a book and Add a book. Thus, the activity diagram of LMS consists of these three activities. Figure 6 shows the activity diagram of Add a book function in LMS.

Figure 3 shows the activity diagram of Borrow a book function in LMS; it demonstrates when any user wants to borrow a book should be entering his/her details and the book details. LMS will verify the user membership. If it is incorrect, the system will display an error message. Otherwise, the system will check the availability of the book. If it is not available, the system will display an error message. While, if it is available, the system will record the book as lent book, delete it from shelves and display a success message.

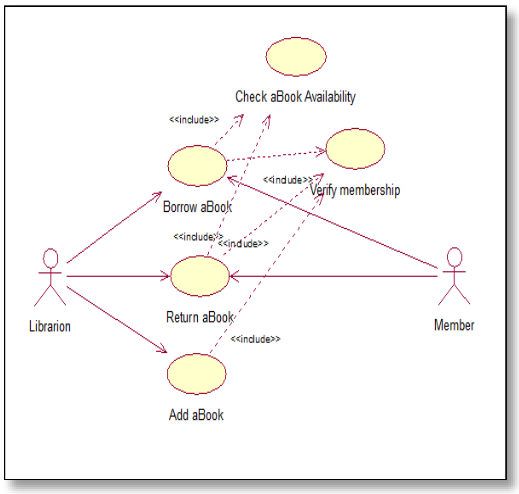


Figure 2. The use case diagram for LMS

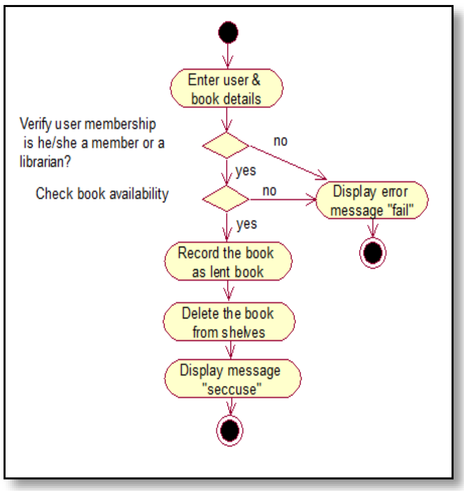


Figure 3. Activity diagram of Borrow a book function in LMS

Figure 4 shows the activity diagram of Return a book function in LMS; it explains when any user wants to return a book should be entering his/her details and the book details. LMS will verify the user membership. If it is incorrect, the system will display an error message. Otherwise, the system will verify from the book. If it is not lent, the system will display an error message. While, if it is lent, the system will delete it from being loan, record it on the shelves and display a success message.

Figure 5 shows the activity diagram of Add a book function in LMS; it shows when any user wants to add a book should be entering his/her details and the book t details. LMS will verify the user membership. If it is not a librarian, the system will display an error message. Else, the system will let the librarian to enter the book details. Then, add the book details into catalogue record it on the shelves and display a success message. Each function with its regarding activity diagram will be transform to Z schema, with consideration for consistency between UML diagrams and the Z schema as discussed in the next section.

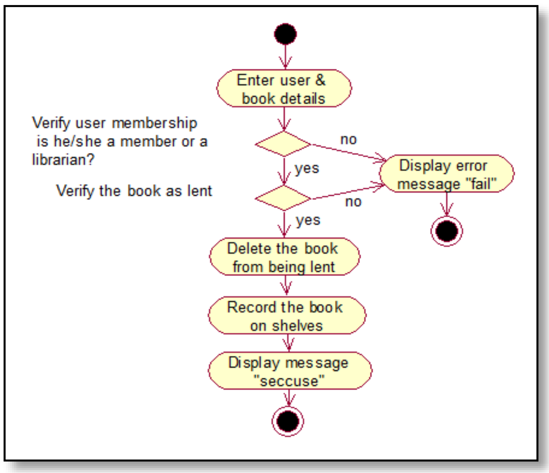

Enter user & book details
Verify user membership is he/she a member or a librarian?
no
yes
Display error message "fail"
Verify the book as lent
no
yes
Delete the book from being lent
Record the book on shelves
Display message "seccuse"

Figure 4. The activity diagram of Return a book function in LMS

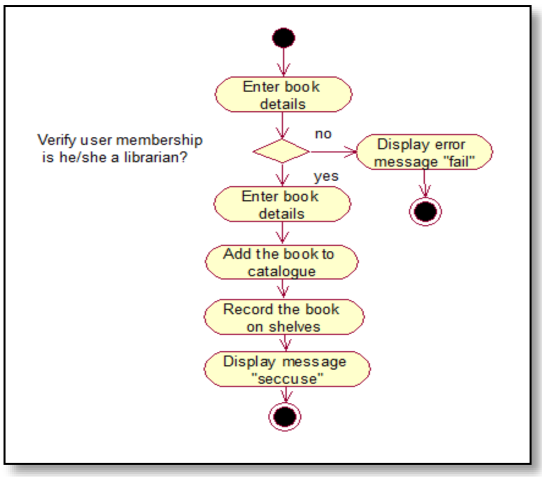
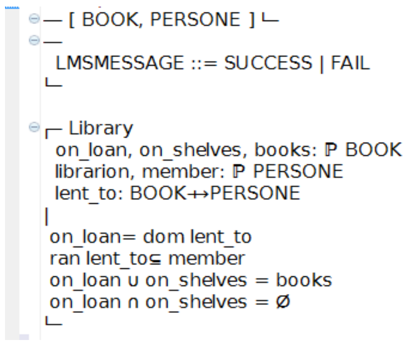

Enter book details
Verify user membership is he/she a librarian?
no
Display error message "fail"
yes
Enter book details
Add the book to catalogue
Record the book on shelves
Display message "seccuse"

Figure 5. The activity diagram of Add a book function in LMS

4.3. System implementation based on formal specifications

The Z is a formal specification language for system specification, and it based on set theory and mathematical logic. Z notations employed in industry; it used in this paper as to be the main part of the solution for this study's issue. Z schemas describe the static and dynamic characteristics of a system like state transition, operations and relationships. The Z schemas describe system specifications and its behaviour to provide sufficient confidence before coding. It written as two parts, the declaration part describes the used variables for function, and the operation part demonstrates the relationship between the values of those variables [11]. In this section, the LMS specifications are implemented based on Z schemas and used Z/EVEs tool for verification. Z/EVEs is an interactive tool which writes, verifies and analysis Z notations. This tool provides two interfaces: a graphical interface and a command line interface. Both interfaces support the analysis of specifications and manage the synchronization of the analysis with modifications to the specification [8]. The proposed approach in this paper indicates to integrate and consisting of UML use-case diagram and activity diagram with Z language. Consequently, UML activity diagrams constructed based on UML use-case diagram in the previous section. In this section, Z specification interpreted from the UML activity diagram. The activity diagram of LMS consists of three activity diagrams based on the main functions in the use-case diagram. Each activity diagram has two cases: valid activity case and invalid activity case. Therefore, Z schemas constructed for two cases of each activity diagram as following:

Figure 6 shows Z Schema declares the power sets BOOK and PERSON. In addition, we represent the response message by using variable LMSMESSAGE. There are only two types of response, which are SUCCESS or FAIL. LMS initialized and all default values have been set to initial value. Books identified as on_loan and on_shelves. Librarian and member are the system users, they declared as PERSON set. In addition to, Lent_to is the relationship that can exist between a Book and a Person as represented by lent to: BOOK↔PERSONE .



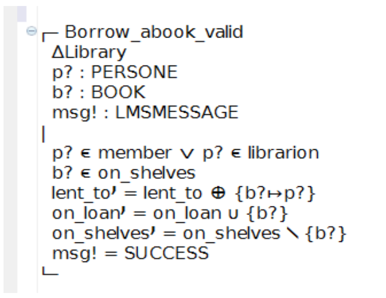
```
— [ BOOK, PERSONE ] └
—
 LMSMESSAGE ::= SUCCESS | FAIL
└

┌ Library
 on_loan, on_shelves, books: ℙ BOOK
 librarion, member: ℙ PERSONE
 lent_to: BOOK↔PERSONE
|
 on_loan= dom lent_to
 ran lent_to⊆ member
 on_loan ∪ on_shelves = books
 on_loan ∩ on_shelves = Ø
└
```

Figure 6. Z Schema for Initialization state of LMS

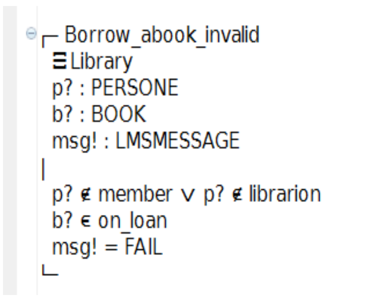
Figure 7 shows the Z schema is verified that, the person that wants to borrow a book was recorded as being a librarian or member, and the particular book is available (not already on loan). Then the book is recorded as being lent to person as represented by $lent\_to' = lent\_to \oplus \{b? \mapsto p?\}$, and the response 'SUCCESS' should be output. Figure 8 shows the Z schema is verified that, the person that wants to borrow a book was not recorded as being a librarian or member, and the particular book is not available (already on loan). Then the LMS state values will not be change as represented by Ξ and the response 'FAIL' should be as output. Figure 9 shows the Z schema verified that, the person that wants to return a book recorded as being a librarian or member, and the particular book recorded as being on loan. Then the book is recorded as being back on the shelves, the record of it being lent to a person is deleted as represented by $lent\_to' = \{b?\} \ndres lent\_to$ and the response 'SUCCESS' should be output.

Figure 10 shows the Z schema verified that, the person that wants to borrow a book was not record as being a librarian or member, and the particular book is not record as not being on loan. Then the LMS state values will not be change as represented by Ξ and the response 'FAIL' should be as output. Figure 11 shows the Z schema verified that, the person that wants to return a book recorded as being a librarian. Then the book shall be add to the library catalogue and the response 'SUCCESS' should be as output. Figure 12 shows the Z schema verified that, the person that wants to borrow a book was not record as being a librarian. Then the LMS state values will not be change as represented by Ξ and the response 'FAIL' should be as output.



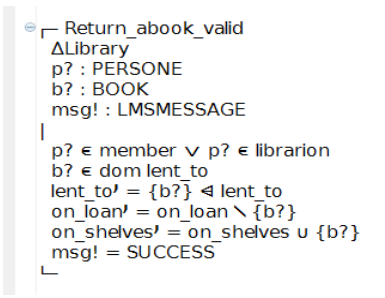
```
Borrow_abook_valid
ΔLibrary
p? : PERSONE
b? : BOOK
msg! : LMSMESSAGE
|
p? ∊ member ∨ p? ∊ librarion
b? ∊ on_shelves
lent_to′ = lent_to ⊕ {b?↦p?}
on_loan′ = on_loan ∪ {b?}
on_shelves′ = on_shelves ∖ {b?}
msg! = SUCCESS
```

Figure 7. Z Schema for Borrow a book valid case of LMS



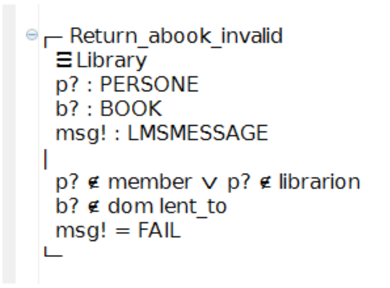
```
Borrow_abook_invalid
ΞLibrary
p? : PERSONE
b? : BOOK
msg! : LMSMESSAGE
|
p? ∉ member ∨ p? ∉ librarion
b? ∊ on_loan
msg! = FAIL
```

Figure 8. Z Schema for Borrow a book invalid case of LMS



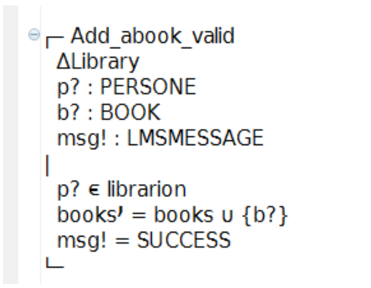
```
Return_abook_valid
ΔLibrary
p? : PERSONE
b? : BOOK
msg! : LMSMESSAGE
|
p? ∊ member ∨ p? ∊ librarion
b? ∊ dom lent_to
lent_to′ = {b?} ⩤ lent_to
on_loan′ = on_loan ∖ {b?}
on_shelves′ = on_shelves ∪ {b?}
msg! = SUCCESS
```

Figure 9. Z Schema for Return a book valid case of LMS



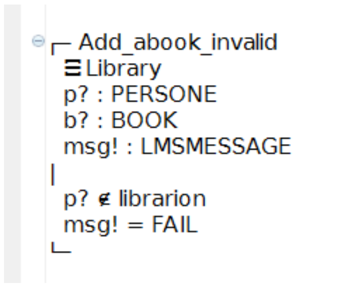
```
Return_abook_invalid
ΞLibrary
p? : PERSONE
b? : BOOK
msg! : LMSMESSAGE
|
p? ∉ member ∨ p? ∉ librarion
b? ∉ dom lent_to
msg! = FAIL
```

Figure. 10 Z Schema for Return a book invalid case of LMS



```
Add_abook_valid
ΔLibrary
p? : PERSONE
b? : BOOK
msg! : LMSMESSAGE
|
p? ∊ librarion
books′ = books ∪ {b?}
msg! = SUCCESS
```

Figure 11. Z Schema for Add a book valid case of LMS



```
Add_abook_invalid
ΞLibrary
p? : PERSONE
b? : BOOK
msg! : LMSMESSAGE
|
p? ∉ librarion
msg! = FAIL
```

Figure 12. Z Schema for Add a book invalid case of LMS

5. RESULTS AND DISCUSSION

The main contribution of this paper is to provide an approach to convert from informal specification into formal specification by focusing on integration and consistency between UML specification and formal method. The proposed approach is a formalization of Library Management System (LMS). It starts from the system's requirements, and then translated to UML use case diagram and activity diagram. Furthermore the Z schemas were adapted based on activity diagrams. The syntax and semantics of LMS specifications are verified with the tool Z/EVES. In the proposed approach, it has been seen that bridged a gap between informal and formal methods. There are certain benefits to improved system factors during analysis and design phases before implementation. Following benefits can be a result of the integration UML specification with Z language:

5.1. Reduced the ambiguity

This is one of the biggest advantages that can be achieved by integration of informal and formal methods. This approach emphasizes on identifying the system behavior by using UML behavior diagrams (use case diagram and activity diagram) based on system requirements. In addition to, the formal methods are mathematically based techniques for specifying a description of the system to be developed. Therefore, the system requirements specification will be clearly defined

5.2. Reduced the error

This is one of the most important aspects of the software design. The proposed approach is certifying a developed system, by generating models based on system's requirements by using the Rational Rose tool to create a complete UML diagrams. And then uses formal methods to generate proofs that the schema meets requirements. Moreover the schemas are verified by using Z/EVES checker tool, to overcome the errors.

5.3. Improved the quality

Formal methods use mathematical and logical formalization. Formal methods have been used in our approach, starting from the design phase to find defects. The basic purpose is to improve the quality of the resulting software. Therefore, formal methods were the best way to improve software quality can be applied in early stages of the Software Development Life Cycle (SDLC).

5.4. Confidentiality

We anticipate that software security can be improved through the application of formal methods. By implementing Z schema for each function of that software. Thus, it makes it easier to manage and implement the code and make it secure against unauthorized access. With this we can sort that which user can access what type of information. So the information is protected.

Briefly, the results of the integration UML specification with formal specification are reduced the ambiguity and Error, and Improved the quality and confidentiality of the system in the early stages. However, the limitation of the proposed approach is that it does not support code generation. Even if the proposed approach to design LMS does not provide the fine-grained steps and practices for impletation phase, the development processes in this paper allow the software developer to describe and capture the analysis and design phases for LMS. However, since the standard Z notations used to support the code generation by any programming language.

6. CONCLUSION

This paper proposed an approach to formalization of Library Management System (LMS) by focusing on integration and consistency between UML specification and formal method. The behavior of the system is specified from system's requirements, and then translated to UML use case diagram and activity diagram. Moreover the Z schemas were adapted based on activity diagrams. The syntax and semantics of LMS specifications are verified with the Z/EVES tool. The proposed approach is an integration of UML specification with Z language, as a result that has been bridged a gap between informal and formal methods. The obtained results were to improve the system's requirements during analysis and design phases in terms reduced the ambiguity and error, improved the quality and confidentiality.

ACKNOWLEDGEMENTS

The authors would like to thank the University of Kufa, Ministry of Higher Education and Scientific Research (MoHESR), Directorate of Education in Najaf and the Iraqi Ministry of Education for their support with collaboration and assistance in this research.

REFERENCES

[1] N. Ibrahim, R. Ibrahim, M. Z. Saringat, D. Mansor & T. Herawan, "Consistency rules between UML use case and activity diagrams using logical approach," *International Journal of Software Engineering and its Applications*, Vol. 5, No. 3, pp. 119-134, 2011.

[2] B. Mondal, B. Das & P. Banerjee, "Formal Specification of UML Use Case Diagram-A CASL Based Approach," *International Journal of Computer Science and Information Technologies*, Vol. 5, No. 9, pp.2113-2717, 2014.

[3] B. Falah, M. Akour, I. Arab & Y. M'hanna, "An Attempt Towards a Formalizing UML Class Diagram Semantics," *In Proceedings of the New Trends in Information Technology (NTIT-2017), The University of Jordan, Amman, Jordan*, pp. 21-27, 25-27, April 2017.

[4] N. M. Minhas, A. M. Qazi, S. Shahzadi & S. Ghafoor, "An Integration of UML Sequence Diagram with Formal Specification Methods-A Formal Solution Based on Z," *Journal of Software Engineering and Applications*, Vol. 8, No. 8, pp. 372-383, 2015.

[5] M. Singh, A. K. Sharma & R. Saxena, "Why Formal Methods Are Considered for Safety Critical Systems?," *Journal of Software Engineering and Applications*, Vol. 8, No. 10, pp. 531-538, 2015.

[6] M. Singh, A. K. Sharma & R. Saxena, "Formal Transformation of UML Diagram: Use Case, Class, Sequence Diagram with Z Notation for Representing the Static and Dynamic Perspectives of System," *In Proceedings of International Conference on ICT for Sustainable Development, Springer Singapore*, pp. 25-38, 2016.

[7] S. H. Bakri, H. Harun, A. Alzoubi, R. Ibrahim, Z. Jamaludin, N. ChePa & M. S. A. Bakar, "The formal specification for the inventory system using Z language," *In Proceedings 4th International Conference on Computing and Informatics(ICOCI)*, pp. 28-30, 2013.

[8] P. Saratha, G. V. Uma & B. Santhosh, "Formal Specification for Online Food Ordering System Using Z Language," *In Proceedings of International Conference on Recent Trends and Challenges in Computational Models (ICRTCCM), Second International, IEEE,* pp. 343-348, 2017.

[9] P. Srichetta, "Conceptual Framework of Transforming Informal Requirements Specification to Z Specification," *In Proceedings of International Conference on Computer and Software Modeling IPCSIT, Singapore*, pp. 19-24, 2011.

[10] M. Batra, "Formal methods: Benefits, challenges and future direction," *Journal of Global Research in Computer Science*, Vol. 4, No. 5, pp. 21-25, 2013.

[11] M. W. Azeem, M. Ahsan, N. M. Minhas & K. Noreen, "Specification of e-Health system using Z: A motivation to formal methods," *In Proceedings International Conference for Convergence of Technology (ICCT), IEEE*, pp. 1-6, 2014.

[12] S. A. Khan & H. Jamshed, "Analysis of formal methods for specification of e-Commerce applications," *Mehran University Research Journal Of Engineering & Technology*, Vol. 35, No. 1, pp. 19-28, 2016.

[13] D. K. Kim, "Development of Mobile Cloud Applications using UML," *International Journal of Electrical and Computer Engineering (IJECE),* Vol. 8, No. 1, pp. 596-604, February 2018.

[14] A. Lasbahani, M. Chhiba & A. Tabyaoui, "A UML Profile for Security and Code Generation," *International Journal of Electrical and Computer Engineering (IJECE),* Vol.8, No.6, pp. 5278-5291, December 2018.

BIOGRAPHIES OF AUTHORS

Zainab Hassan Muhamad received the Bachelor's degree in Computer Scinece from University of Babylon, Iraq, in 2007, and the Master's degree in Software Engineering from Universiti Tun Hussein Onn Malaysia (UTHM), Batu Pahat, Malaysia. She is currently a Lecturer of Computer Science at Directorate of Education, Iraqi Ministry of Education, Najaf, Iraq .Her research interests include software enginering, software testing and formal method.

Dhafer Abdulameer Abdulmonim received the Bachelor's degree in Computer Scinece from University of Babylon, Babil, Iraq, in 2007, and the Master's degree in Software Engineering from Universiti Tun Hussein Onn Malaysia (UTHM), Batu Pahat, Malaysia, in 2015. He is currently a Lecturer of Computer Science at Directorate of Education, and Open Educatinal College, Najaf, Iraq. His research interests include software enginering, software testing and formal method.

Bashar Alathari received a M.Sc. degree in software engineering from University of UTHM in 2015 from Malaysia. He is working lecture and Researcher in university of kufa, ITRDC. Among his research interests are research in software engineering and networking.