Validation Framework for Aspectual Requirements Engineering (ValFAR)

Sohil F. Alshareef
Faculty of Information Technology
Benghazi University, Libya
sohil.alshareef@gmail.com

Abdelsalam M. Maatuk
Faculty of Information Technology
Benghazi University, Libya
abdelsalam.maatuk@uob.edu.ly

Tawfig M. Abdelaziz
Faculty of Information Technology
Benghazi University, Libya
tawfig.tawill@uob.edu.ly

Mohammed Hagal
Faculty of Information Technology
Benghazi University, Libya
mohamed.hagal@uob.edu.ly

ABSTRACT
Aspect-Oriented Requirements Engineering (AORE) extends the existing requirements engineering approaches to support the identification and handling of crosscutting concerns. Crosscutting concerns are considered as potential aspects and can lead to the phenomenal "tyranny of the dominant decomposition". Requirements-level aspects are responsible for producing scattered and tangled descriptions of requirements in the requirements document. Requirements validation artifact is an essential task in software development. This task ensures that requirements are correct and valid in terms of completeness and consistency, hence, reducing the development cost, maintenance and establish an approximately correct estimate of effort and completion time of the project. In this paper, we present a validation framework for aspectual requirements that can be used with AORE approaches to facilitate the validation of the resulting crosscutting relationships and aspects. The proposed framework comprises a high-level and low-level validation. The high-level validation is to validate the concerns with stakeholders, whereas the low-level validation validates the aspectual requirement by developers using a checklist. The approach has been evaluated using a case study. The results demonstrate that the proposed framework is feasible and acceptable.

Keywords
AORE, validation and verification, requirements engineering, aspectual requirements, crosscutting concern.

1. INTRODUCTION
The Aspect-Oriented Requirements Engineering (AORE) aims to extract and construct good identification and separation of crosscutting concerns [14]. Conventional requirements engineering (RE) approaches treat the resulting intersection of concerns, i.e., aspects, as a rule, or restriction relationship [33] [6].

ICEMIS'20, September 14–16, 2020, Almaty, Kazakhstan

ACM ISBN 978-1-4503-7736-2/20/06…$15.00
https://doi.org/10.1145/3410352.3410777

Such concerns might lead to what so-called tyranny of the dominant decomposition, if not handled properly, and this might result in scattered and tangled implementation of concerns to other software artifacts [16]. In terms of aspect orientation, traditional RE approaches have been extended to support the crosscutting nature of concerns as early as possible in software development.

In this context, shortcomings, such as clean mapping of requirements artifacts to later stages, satisfactorily trade-off analysis, equality treating of concerns and validation of AORE artifacts remain a challenge and not fully studied [1][13].

Validation in requirements engineering is the process to eliminate the conflicts among requirements in software requirements specification (SRS) [1][2]. Besides, requirements validation relates to the analysis since it is involved in detecting problems associated with the requirements, i.e. inconsistencies, incompleteness and ambiguities among requirements [2]. Inconsistent requirements are the requirements that are stated and described differently at different places in requirements documents, where ambiguous requirements referred to as one requirement that has multiple perceptions, and incomplete requirements, which refers to the requirement that does not convey a complete and meaningful requirement.

Furthermore, requirements validation is a crucial activity as it helps to reduce the cost of maintenance. This is due to the reason that, the cost of fixing errors in later stages, i.e., architecture, design, and code, of software development is more expensive than to rectify them in the requirements analysis phase [2]. In addition, errors in SRS lead to huge rework cost and effort when issues are discovered in later stages. Fixing errors in later stages of software development require more time, effort and cost [33][15][37]. Moreover, making changes at this stage requires developers to properly manage the changes as these changes would affect other artifacts, which are associated with the requirements and it would to re-testing of the system when the new changes are employed.

Through our investigation described in [3], we noticed that almost all of the approaches focus on identifying crosscutting concerns as well as representing them either using graphical models, XML schema, templates or text-based representation. Therefore, the resulting artifacts from requirements phase must be validated before moving to other development stages to reduce extra work and effort caused by missing information and unintentionally overlooked inconsistency rules. In addition, the validation can be either dynamic or static. Dynamic validation is easier to perform and it allows the automatic checking and

interpretation of software artifacts, e.g., Test Driven Development. In contrast, the static validation implies understandability and completeness of requirements artifacts to be validated [39]-[43]. However, dynamic validation is often more costly than static validation as the problem domain needs to be formalized and fully covered at first.

This paper introduces a validation framework for crosscutting concerns and aspectual requirements. The framework is based on two levels of validation. The high-level validation is concerned with validating the concerns and the crosscutting relationship that form the basis for the aspect. The other level of validation, which is low-level, is concerned with validating the aspectual requirements. The proposed solution has been applied and evaluated using a case study, where the results showed that the proposed method is practical and satisfactory.

The remainder of the paper is organized as follows: Section 2 presents the related work to the validation and verification of aspects and crosscutting concerns. Section 3 introduces the ValFAR approach. Section 4 discusses some of the obtained results. Section 5 concludes the paper.

2. RELATED WORK

Different AORE approaches are dedicated to aspectual requirements focusing on different areas. We have found that only a few of AORE proposals provide validation for elicited aspects [6][11][7]. Theme approach [11], is based on lexical analysis of the requirements and its effective method to identify and extract the relationship between aspects by action verbs. AORE with ArCaDe [6] identifies aspects as viewpoints and it encapsulates them in XML schema with their requirements and sup-requirements. However, this method focuses on the composition of aspectual requirements rather than validating the concerns and the identified aspects.

The AORE model proposed in [6] applies a viewpoint-oriented approach to identify stakeholders' requirements using viewpoints and XML [38]. Araŭjo et al. [6] introduced a framework based on a viewpoint-oriented approach. It also generates templates to identify crosscutting nonfunctional requirements.

Grundy [8] proposed an aspect-oriented requirements engineering approach for component-based systems, whereas Jackson [9][13] proposed a problem frame approach, which identifies aspects from the problem domain. This method only applies when the problem being investigated has been solved in the past, and the solution can be used again by breaking down problems into subproblems in parallel with defining composition rules, which can be used to resolve the conflict between aspects.

Other models, e.g., Cosmos [13] and CORE [7] are proposed for concern-oriented or multidimensional separation of concern approach. Although Cosmos is known as the best for its ability to analyze the relationship between concerns, it lacks the systematic means to identify any type of concern. Moreira [7] proposes an approach, called CORE that introduces a projection table to reduce the concern dimension.

Theme approach [5][10] is an AORE approach that is not based on conventional requirements engineering techniques such as, viewpoint, use case scenario and goal-oriented. Theme solution extracts crosscutting concerns and identifies aspects by keyword and lexical analysis.

Araŭjo et al. [16] proposed a use-case based requirement approach that identifies concerns from different scenarios of the system. It is a template-based approach for identifying aspectual use-cases, which are modeled, providing traceability similar to GERMSoC [36]. In addition, there are numerous approaches that identify aspects using use-case requirement engineering [14][15][16].

In the context of aspectual requirements validation and to the best of our knowledge, we have found only one framework presented in [4], which argues that the validation of aspectual requirements can be accomplished with two levels validation. It amalgamates three AORE approaches: viewpoint-oriented approach, goal-oriented approach and use case-oriented approach. It converts the identified concerns from each AORE approach into their respective graphs. An aspectual graph is produced and provided as an input to the aspectual graph parser, which generates an XML file for the aspectual graph to compare it with each XML file from each approach to remove any duplicated concerns and ambiguities. However, there is no way to validate the framework in general. It was presented without actual results or case studies to prove that the method is working properly with these three approaches. The conversion process itself is not clearly defined with specific steps to follow neither the integration of the three models into one XML file.

3. THE PROPOSED SOLUTION

This section describes our proposed solution, which is called Validation Framework for Aspectual Requirements (ValFAR). The solution validates the AORE artifacts in three main phases, each of which has its own activities as shown in Figure 1. Phase 1 is for concern handling, Phase 2 is for concern validation, i.e., high-level validation, whereas Phase 3: consists of aspectual requirement specification and validation, i.e., low-level validation. The subsequent subsections describe these three phases.

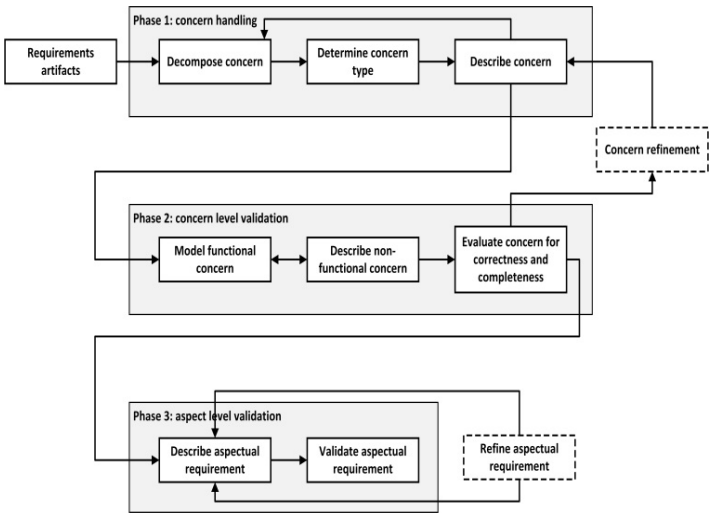

Figure 1. The ValFAR Framework

Prior to Phase 1, the correlation between concerns that form the crosscutting relationship should be addressed with a matrix. In addition, it is useful to link the source of information of each concern with the corresponding stakeholder.

Phase 1: Concern Handling

AORE approaches treat concerns based on the approach being used to identify concerns and aspects. In a viewpoint-oriented approach, concerns are treated as viewpoints. Goal-oriented approaches are based on soft-goal and sub-goal concepts and aspects are discovered and identified as operationalities that affect the sub-goal, i.e., non-functional concern. In the Theme/Doc approach, the theme and base theme are functional and non-

functional concerns respectively. Thus, the purpose of this phase is to treat each artifact, i.e., viewpoint, goal, sub-goal, base theme, and cross-cutting theme, which is used to identify aspects, as a concern in order to perform the activities to achieve high granularity. This phase involves three activates as follows:

Activity 1: Determine Type of Concern

The concern type should be determined before processing. A concern might be functional or non-functional. Reviewing the analysis documents and the requirements specification helps to understand the nature of concerns, hence determining its type. Functional concerns are treated differently from non-functional in terms of specification, modeling. Thus, determining concern type is crucial for granularity.

Activity 2: Decompose Concern

Concerns might contain additional information that should be separated or new concerns that are derived from the main concern [20]. Such concerns can be difficult to handle due to their tangled nature [19][21]. Therefore, we propose to decompose the general concerns into sub-concerns using the following decomposition rules:

- If some parts of the concern present a limitation as to how specific goal is achieved or implies design constraints on the original concern.
- If there are parts of the concern that are not used by other concerns, but they are used to describe the issue or the problem and the description of the issue is too long.
- If the concern contains a reference of description to other different concerns and requires information from different concerns.
- If there is a need to derive new concerns from the original concern or some parts of the original concern are going to be associated with other concerns.

The decomposed concern, i.e., the main concern, is documented in a separate template with a link to sub-concerns that have been extracted from the main concern [21][23]. A concern might be decomposed into one or more sub-concerns.

Activity 3: Describe Concern

A description of concerns is important to understand the structure of each concern [17][29]. Concerns should be documented in a template to help developers understand the nature and goal of concerns [18][17]. The elements of the concern description template are:

- **Concern id**: the unique identifier of the concern for traceability purposes.
- **Concern name**: a name that should not be repeated elsewhere to prevent inconsistencies and confusion.
- **Objective**: a brief description of concerns objective written in clear language.
- **Type**: type of concern, whether it is a functional or non-functional concern.
- **Successful scenarios**: a concise and complete description of the process executed by the concern.
- **Alternative scenarios**: an alternative to the main scenario when limitations constraint it.
- **Revision date**: the date that the concern was modified last.
- **Review count**: indicates the number of iterations performed to process the concern.
- **Reference**: a reference to the concern analysis process and concern representation.

Phase 2: Concern validation (high-level validation)

In Phase 1, concerns are decomposed and described structurally based on the type of concern. Moreover, in Phase 2, functional concerns are validated using the semi-formal technique. In addition, non-functional concerns those constraints the functional concerns are linked with the influenced concerns described and separated in a dedicated template. The stakeholder is part of this phase as the developer validates the functional concerns and gets immediate feedback from the stakeholder. The following three activates describe this phase.

Activity 1: Model Functional Concern

In this activity, the functional concerns are validated in a semi-formal manner [24]. The UML sequence diagram model is used to verify the interactions with corresponding artifacts of the system [27]. Besides, the specification of the concern must be validated to ensure the correctness and it satisfies its objective. Additionally, the stakeholder, i.e., end-user or customer, is involved for better results [25][26]. The number of iterations of this activity depends on the granularity of the feedback between the customer and the developer.

Activity 2: Describe the non-functional concern

In contrast to functional concerns, non-functional concerns usually are not associated with verbs that express action and cannot be modeled by UML models [29]. In addition, non-functional concerns employ how the system should respond and react to the user's interaction, i.e., response time or performance [30] [28]. Therefore, we propose a structured template to describe non-functional constraints or design limitations that limit the structure of the functional concern. The elements to be included in the non-functional template are:

- **Name**: Name of the non-functional concern as described by developers.
- **ID**: a unique identifier to avoid redundancy and increase the ability to trace and concern.
- **Related concerns**: functional concerns that are constrained by this non-functional concern.
- **Specification**: clear and concise description of the limitation of this concern on other concerns.
- **Revision date**: a date of which the concern was last reviewed or altered.
- **Review count**: number of iterations performed by developers to process the concern.

A functional concern might be constrained by more than one non-functional concern [29][30]. Using this technique to document and specify non-functional concerns benefits in several ways:

- Developers can derive architectural choices from the template and behavior of the functional concern.

- Separate functional and non-functional concern specifications to improve the understandability and readability of the artifacts.
- Separate overlapping quality attributes that might be included without detailed analysis within the functional concern.
- Improve traceability of non-functional concern, since the template can be altered any given time during the development in terms of evolution or refinement.

Activity 3: Evaluate Concerns Correctness and Completeness

Concerns that are part of the crosscutting relationship must be investigated to validate the relationship and the integrity of the aspect through inspecting the concerns using a checklist that contains quality questions aimed at evaluating the completeness and consistency, i.e., logical and grammatical consistency. This activity helps to ensure that concerns and requirements have been extracted well and ready to be moved to the next stage. In addition, all of the questions of the checklist must be answered satisfactorily to each element; otherwise, the process is repeated until all questions are answered with "Yes" [11]. The questions to be included in the template are:

- Is the concern defined clearly?
- Are there missing details or anything forgotten?
- Is there any association with other concerns or requirements?
- Is there any association with other concerns or requirements?
- Does it include all the necessary information to make the architectural decision and design it?
- Is there enough information to develop a test case for the concern?
- Is the successful scenario of each concern written in clear language?
- Is the description written so that it won't lead to misinterpretation?
- Are there any parts of the concern repeated in other concerns?
- Are there any conflicts with other concerns or requirements?
- Is the concern traceable to its origins and stakeholder?

Phase 3: Aspectual requirement specification and validation

The crosscutting relationship between concerns is validated according to the defined criteria in Phase 2. The area of intersection, i.e., common interest with other concerns or requirements is considered as a candidate aspect. Thus, aspectual requirements should be specified correctly. Although aspects are systematically extracted with the aid of AORE approaches, the lack of capturing all the necessary information in a single unit structurally, e.g., aspectual requirement document remains overlooked. To overcome this issue, we propose a structured template to document the critical details and the required information to validate the aspect statically as Phase 3 of the solution, which contains two main activities.

Activity 1: Describe Aspectual Requirement

Aspectual requirements should not be too technical and have to be as abstract as possible [33]. The aspectual requirement describes the solution concept of the crosscutting concern, which has to be carefully treated when designing and implementing the aspect in later stages. Besides, the description of the aspectual requirements has to be clear and concise [31]. The details to be included in the aspect document are:

- **Aspect ID**: a unique identifier for the aspect.
- **Aspect name**: a name to describe the aspect and it should not be repeated with other aspects.
- **Concerns**: list of concerns or concerns that are influenced by this aspect.
- **Aspect description**: a brief and concise description of the aspect of behavior.
- **Aspect priority**: the degree of importance for the aspect demanded by stakeholders.
- **Aspect post-condition**: a service that the aspect will provide.
- **Reference**: reference to the analysis material, documentation, and representation of the aspect for understandability.

After documenting the aspectual requirement, a matrix is developed if there is a dependency between aspects in terms of execution and responsibility [32]. In order to achieve a good result from this matrix, the analysis of the requirements and concerns must have addressed the aspect life cycle. It could be shown as a graphical representation of the aspect or through tables provided by some of the AORE approaches that handle this issue.

Activity 2: Validate Aspectual Requirement

To ensure the quality of the information acquired by the requirements engineers, we propose to develop a list of questions that are targeting the core concepts of the aspects in the requirements level. In other words, the other development stages architecture, design, and coding should not be included in the description of the aspectual requirements. Therefore, the requirements section of the validation form must be answered to "Yes". It means that all the necessary information for the aspectual requirements is available in the correct format. The description of the aspectual solution should not limit the ability of developers to design the aspect. If, however, constraints or design limitation that should be mentioned in the description, developers need to take that into consideration to perform trade-off analysis and negotiation with stakeholders. The questions to be included in the template are:

- Does the aspect describe the solution concept for the problem?
- Does the aspect define the process of inaccurate description and understandable language?
- Does the aspect contain all the related parts of the crosscutting concerns?
- Have the areas of intersection with other requirements or concerns have been clearly identified and specified within the scope of the aspect?

Are the requirements and concerns affected by the aspect identified and represented clearly?

4. EXPERIMENTS AND RESULTS

In this study, we have conducted two experiments on two existing AORE approaches, i.e., AORE with ArCaDe and Theme/Doc. Each solution treats concerns differently. Our method treats each artifact of each approach as a concern, which is the basic element

of the aspectual requirement. Theme/Doc is a powerful tool to extract the relationship between concerns based on the lexical analysis. ArCaDe, on the other hand, uses the viewpoints to store the aspect along with its requirements and concerns. In addition, we investigated the outcome of each approach (concerns and aspects) and concluded that aspects identified based on viewpoints approach requires more decomposition to isolate the tangled concerns as it gives the ability to validate each concern individually. In Theme/Doc, the identified cross-cutting theme was acceptable. In both approaches, the ValFAR has shown that the identified concerns needed to be modified and refined several times before they can form the basis for the aspectual identification. Further details of our experimental studies can be found in [35].

5. CONCLUSION

In this paper, we have introduced a validation framework for aspectual requirements. The framework validates AORE artifacts on three main phases. Each phase has its own activities to be performed in order to achieve the best results. The proposed framework comprises a high-level and low-level validation. The high-level validation is to validate the concerns with stakeholders, whereas the low-level validation validates the aspectual requirements by developers using a checklist. The approach has been evaluated using the analysis artifacts of two AORE approaches. The results demonstrated that the ValFAR method is feasible and acceptable. To conclude, the investigation into the aspectual requirements validation has shown that aspects in requirements analysis are not specified in requirements specification as the regular requirements and concerns. As a result, developers document the aspects on their own without any guidelines or standards to do so. In the available literature, we showed that the existing approaches specify concerns and aspects to their satisfaction. Each approach describes and specifies aspects different from the other approaches. Thus, a standard and formal specification of aspects desired to cope with this issue is required.

REFERENCES

[1] H. A. Bila, M. Ilyas, Q. Tariq, and M. Hummayun, "Requirements validation techniques: An empirical study," *Int. J. Comput. Appl.*, vol. 148, no. 14, pp. 5–10, 2016.

[2] Lana, Cristiane Aparecida, Milena Guessi, Pablo Oliveira Antonino, Dieter Rombach, and Elisa Yumi Nakagawa. "A Systematic Identification of Formal and Semi-Formal Languages and Techniques for Software-Intensive Systems-of-Systems Requirements Modeling." *IEEE Systems Journal* (2018), 2018.

[3] Alshareef, Sohil F., Abdelsalam M. Maatuk, and Tawfig M. Abdelaziz. "Aspect-oriented requirements engineering: approaches and techniques." In *Proceedings of the First International Conference on Data Science, E-learning and Information Systems*, p. 13. ACM, 2018.

[4] Khan, S. S., S. Ali, and M. Jaffar-ur-Rehman. "An enhanced framework for validation of aspectual requirements." In *Proceedings of the IEEE Symposium on Emerging Technologies, 2005.*, pp. 435-439. IEEE, 2005.

[5] Baniassad and S. Clarke, "Theme: An Approach for Aspect-Oriented Analysis and Design": *International Conference on Software Engineering, 2004.*

[6] Rashid, A. Moreira, and J. Araŭjo, "Modularisation and Composition of Aspectual Requirements". *2nd International Conference on Aspect-Oriented Software, 2003*.

[7] Moreira, J. Araŭjo, and A. Rashid, "A Concern-Oriented Requirements Engineering Model". O. Pastor and J. Falcao e Cunha (EDs): CAiSE 2005, LNCS 3520, pp. 293-308, Springer –Verlag Berlin Heidelberg 2005.

[8] J. Grundy, "Aspect-Oriented Requirements Engineering for Component-based Software Systems" *4th IEEE International symposium on RE, IEEE Computer Society Press, pp. 84-91, 1999.*

[9] M. Jackson, "Problems, Subproblems, and Concerns". Position paper submitted to Early Aspect 2004.

[10] E. Baniassad and S. Clarke, "Finding Aspects in Requirements with Theme/Doc," presented at *Workshop on Early Aspects (held with AOSD 2004), Lancaster, UK, 2004.*

[11] Ullah, S., Iqbal, M. and Khan, A.M., 2011, July. A survey on issues in non-functional requirements elicitation. In *International Conference on Computer Networks and Information Technology* (pp. 333-340). IEEE.

[12] Barroca, Leonor, José Luiz Fiadeiro, Michael Jackson, Robin Laney, and Bashar Nuseibeh. "Problem frames: A case for coordination." In *International Conference on Coordination Languages and Models*, pp. 5-19. Springer, Berlin, Heidelberg, 2004.

[13] Sutton Jr, S.M. and Rouvellou, I., 2002, April. Modeling of software concerns in cosmos. In Proceedings of the 1st international conference on Aspect-oriented software development (pp. 127-133). ACM.

[14] G. M. C. Sousa and J. B. Castro, "Separation of Crosscutting Concerns from Requirements to Design: Adapting a Use Case Driven Approach". *Early Aspect 2004 -- Aspect Oriented Requirement Engineering and Architecture Design, 2004.*

[15] Brito and A. Moreira, "Integrating the NFR framework in RE model". Early Aspect 2004 Aspect Oriented Requirement Engineering and Architecture Design, 2004.

[16] J. Araŭjo, A. Moreria, I. Brito and A. Rashid, "Aspect-Oriented Requirements with UML". *Aspect Modeling with UML workshop at the Fifth International Conference, 2002.*

[17] De Souza, Alessandro J., and Anderson Luiz O. Cavalcanti. "Visual Language for Use Case Description." *Software: Practice and Experience* 46, no. 9 (2016): 1239-1261. 2016

[18] Raengkla, Monthawan, and Taratip Suwannasart. "A Test Case Selection from Using Use Case Description Changes." In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, vol. 1, pp. 13-15. 2013.

[19] Taibi, Davide, Valentina Lenarduzzi, Andrea Janes, Kari Liukkunen, and Muhammad Ovais Ahmad. "Comparing requirements decomposition within the scrum, scrum with kanban, XP, and banana development processes." In *International Conference on Agile Software Development*, pp. 68-83. Springer, Cham, 2017.

[20] Pohl, Klaus. Requirements engineering: fundamentals, principles, and techniques. Springer Publishing Company, Incorporated, 2010.

[21] Dick, Jeremy, Elizabeth Hull, and Ken Jackson. *Requirements engineering*. Springer, 2017.

[22] Liu, Kevin, Ricardo Valerdi, and Phillip A. Laplante. "Better requirements decomposition guidelines can improve cost estimation of systems engineering and human systems integration." *Hoboken, NJ*, 2010.

[23] Cruz, Estrela F., Ricardo J. Machado, and Maribel Y. Santos. "On the decomposition of use cases for the

refinement of software requirements." In *2014 14th International Conference on Computational Science and Its Applications*, pp. 237-240. IEEE, 2014.

[24] Broy, Manfred. "Multifunctional software systems: Structured modeling and specification of functional requirements." *Science of Computer Programming* 75, no. 12 (2010): 1193-1214.

[25] Helming, Jonas, Maximilian Koegel, Florian Schneider, Michael Haeger, Christine Kaminski, Bernd Bruegge, and Brian Berenbach. "Towards a unified requirement modeling language." In *2010 Fifth International Workshop on Requirements Engineering Visualization*, pp. 53-57. IEEE, 2010.

[26] Saadatmand, Mehrdad, Antonio Cicchetti, and Mikael Sjödin. "UML-based modeling of non-functional requirements in telecommunication systems." In *The Sixth International Conference on Software Engineering Advances (ICSEA 2011)*, pp. 213-220. Barcelona, Spain: The Institute of Electrical and Electronics Engineers, Inc., 2011.

[27] dos Santos Soares, Michel, Jos Vrancken, and Alexander Verbraeck. "User requirements modeling and analysis of software-intensive systems." *Journal of Systems and Software*84, no. 2 (2011): 328-339.

[28] Chung, Lawrence, Brian A. Nixon, Eric Yu, and John Mylopoulos. *Non-functional requirements in software engineering*. Vol. 5. Springer Science & Business Media, 2012.

[29] Robertson, Suzanne, and James Robertson. *Mastering the requirements process: Getting requirements right*. Addison-Wesley, 2012.

[30] Leffingwell, Dean. Agile software requirements: lean requirements practices for teams, programs, and the enterprise. Addison-Wesley Professional, 2010.

[31] Zambrano, Arturo, Johan Fabry, Guillermo Jacobson, and Silvia Gordillo. "Expressing aspectual interactions in requirements engineering: experiences in the slot machine domain." In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pp. 2161-2168. ACM, 2010.

[32] Sardinha, Alberto, João Araújo, Ana Moreira, and Awais Rashid. "Conflict Management in Aspect-Oriented Requirements Engineering." *Information Sciences and Technologies Bulletin of the ACM Slovakia* 2, no. 1 (2010): 56-59, 2010.

[33] Singh, Narender, and Nasib Singh Gill. "Aspect-oriented requirements engineering for advanced separation of concerns: A review." *International Journal of Computer Science Issues (IJCSI)* 8, no. 5 (2011): 288, 2011.

[34] Zheng, Xiaojuan, Xiaomei Liu, and Shulin Liu. "Use case and non-functional scenario template-based approach to identify aspects." In *2010 Second International Conference on Computer Engineering and Applications*, vol. 2, pp. 89-93. IEEE, 2010.

[35] Sohil F. Alshareef. Validation Framework for Aspectual Requirements (ValFAR). Benghazi University, Libya, 2019

[36] S. C. Bertagnolli, M. L. B. Lisbôa, "The FRIDA Model". *Workshop on: Analysis of Aspect-Oriented Software (AAOS '03)* held in conjunction with the European Conference on Object-Oriented Programming (ECOOP) July 21, 2003.

[37] Mohammed, A. O., Abdelnabi, Z. A., Maatuk, A. M. and Abdalla, A. S. An Experimental Study on Detecting Semantic Defects in Object-Oriented Programs using Software Reading Techniques. In *Proc. of ACM Int. Conf. on Engineering & MIS* (ICEMIS '15), ACM, New York, NY, USA, Article 24, 6 pages. DOI=http://dx.doi.org/10.1145/2832987.2833025

[38] Maatuk, A. M., Akhtar, M. A. and Aljawarneh, S. An algorithm for constructing XML Schema documents from relational databases. In *Proc. of ACM Int. Conf. on Engineering & MIS* (ICEMIS '15). ACM, New York, NY, USA, Article 12, 6 pages. DOI=http://dx.doi.org/10.1145/2832987.2833007

[39] Shadi Aljawarneh, Shadi Aljawarneh, and Manisha Malhotra. 2017. Critical Research on Scalability and Security Issues in Virtual Cloud Environments (1st. ed.). IGI Global, USA.

[40] Shadi Aljawarneh, Muneer Bani Yassein, and We'am Adel Talafha. 2017. A resource-efficient encryption algorithm for multimedia big data. Multimedia Tools Appl. 76, 21 (November 2017), 22703–22724. DOI:https://doi.org/10.1007/s11042-016-4333-y

[41] Muneer Bani Yassein, Shadi Aljawarneh, and Esraa Masadeh. 2017. A new elastic trickle timer algorithm for Internet of Things. J. Netw. Comput. Appl. 89, C (July 2017), 38–47. DOI:https://doi.org/10.1016/j.jnca.2017.01.024

[42] Shadi A. Aljawarneh, Raja A. Moftah, and Abdelsalam M. Maatuk. 2016. Investigations of automatic methods for detecting the polymorphic worms signatures. Future Gener. Comput. Syst. 60, C (July 2016), 67–77. DOI:https://doi.org/10.1016/j.future.2016.01.020

[43] Raja A. Moftah, Abdelsalam M. Maatuk, Peter Plasmann, and Shadi Aljawarneh. 2015. An Overview about the Polymorphic Worms Signatures. In Proceedings of the The International Conference on Engineering & MIS 2015 (ICEMIS '15). Association for Computing Machinery, New York, NY, USA, Article 29, 1–4. DOI:https://doi.org/10.1145/2832987.2833031

Columns on Last Page Should Be Made As Close As Possible to Equal Length

Authors' background

Your Name	Title*	Research Field	Personal website
Sohil F. Alshareef	master student	Software Engineering	sohil.alshareef@gmail.com
Abdelsalam M. Maatuk	Associate professor	Database Systems, Software Engineering	abdelsalam.maatuk@uob.edu.ly
Tawfig M. Abdelaziz	Associate professor	Software Engineering, Multi-Agent Systems	tawfig.tawill@uob.edu.ly
Mohammed Hagal	Assistant professor	Software Engineering	mohamed.hagal@uob.edu.ly

***This form helps us to understand your paper better,the form itself will not be published.**

***Title can be chosen from: master student, Phd candidate, assistant professor, lecture, senior lecture, associate professor, full professor**