Available online www.ejaet.com

European Journal of Advances in Engineering and Technology, 2019, 6(12):133-139

Research Article ISSN: 2394 - 658X

Integrating Qualitative Aspects in Software Systems Engineering: A Comprehensive Approach to Design and Implementation

Iqtiar Md Siddique

Department of Computer Engineering, RMIT University, Australia.
Email: iqtiar.siddique@gmail.com

ABSTRACT

This research explores the multifaceted nature of software systems engineering by emphasizing the integration of qualitative factors throughout the software development lifecycle. This research provides a framework for analyzing, designing, and implementing complex software projects with a focus on maintainability, extensibility, reusability, and robustness. The study combines both top-down and bottom-up design approaches, enabling students to select and apply the most suitable methods based on technology, project duration, risk levels, and customer expectations. The research incorporates a Work Integrated Learning (WIL) experience, simulating industrial software engineering projects to bridge theoretical knowledge with practical application. Students gain hands-on experience in managing and executing software projects while receiving industry feedback to enhance their skills. The course covers essential IT concepts and project management principles, equipping students with the tools needed to succeed in the software engineering industry. This comprehensive approach ensures that graduates are well-prepared for real-world challenges in software development.

Keywords: Software Systems Engineering, Qualitative Analysis, System Design, Human-Centered Design, Implementation Strategies

---

INTRODUCTION

In today's rapidly evolving technological landscape, software systems engineering has become a cornerstone of innovation and efficiency. The complexity of modern software systems demands not only a solid understanding of engineering principles but also a nuanced approach to managing and implementing projects that address both technical and qualitative factors. This research, "Integrating Qualitative Aspects in Software Systems Engineering: A Comprehensive Approach to Design and Implementation," aims to provide a thorough exploration of these aspects, offering a framework for students to analyze, design, and execute complex software projects effectively.

The software engineering lifecycle encompasses various stages, from initial requirements gathering to final deployment and maintenance. While technical skills and methodologies are crucial, qualitative factors such as maintainability, extensibility, reusability, and robustness play a significant role in determining the long-term success of a software system. Maintainability ensures that the system can be updated and repaired efficiently over its lifespan. Extensibility allows the system to adapt to future needs and technologies without extensive rework. Reusability promotes the use of existing components to reduce redundancy and development time. Robustness ensures that the system can handle errors and unexpected conditions gracefully. Integrating these qualitative aspects into the design and implementation phases is essential for creating software that is not only functional but also sustainable and adaptable. The research emphasizes a comprehensive approach to software design and implementation, combining both top-down and bottom-up methodologies. The top-down approach involves breaking down the system into smaller, manageable components, starting from a high-level overview and progressively detailing each part. This method helps in defining the overall architecture and ensuring that all system requirements are addressed. Conversely, the bottom-up approach focuses on developing and integrating individual components, which are then combined to form the complete system. This approach is useful for addressing specific technical challenges and ensuring that each component is robust and reliable.

By integrating these approaches, students can learn to balance various design considerations, such as performance, scalability, and user requirements, while also managing risks and constraints. The research aims to equip students

with the skills needed to make informed decisions about the most appropriate design and implementation strategies based on the project's unique context. A major component of this research is the Work Integrated Learning (WIL) experience, which provides students with practical exposure to industrial software engineering projects. Through simulated projects, students apply theoretical knowledge in a controlled environment, gaining hands-on experience in managing software development processes and addressing real-world challenges. The WIL experience includes tasks such as project planning, requirement analysis, design, implementation, and testing, allowing students to develop a comprehensive understanding of the software engineering lifecycle. Feedback from industry professionals is a critical aspect of this experience, as it helps students refine their skills and adapt to industry standards. This practical application of knowledge ensures that students are well-prepared for the demands of the software engineering field, bridging the gap between academic learning and professional practice.

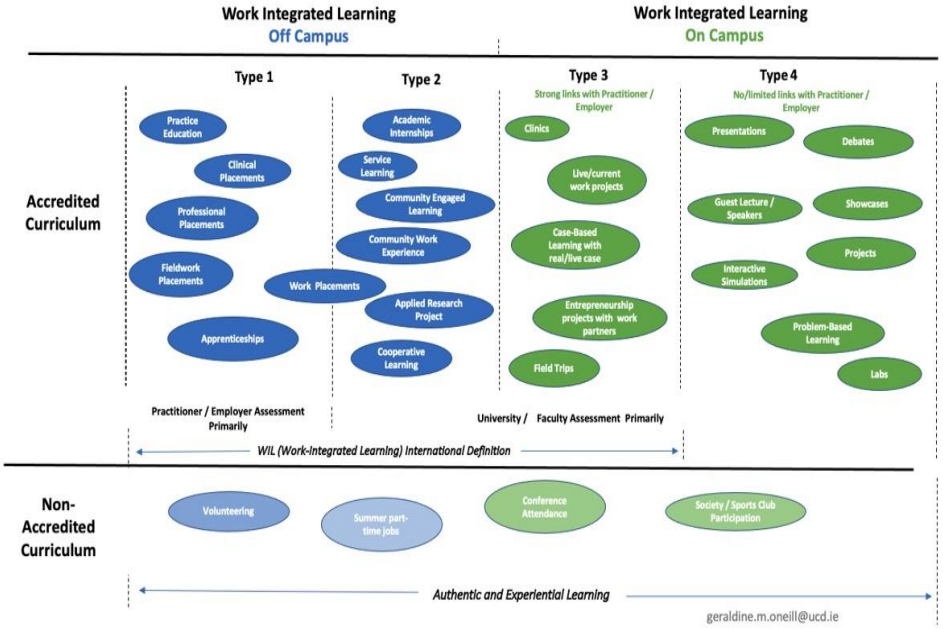

Figure 1: WIL in off and on campus

This Approach to Design and Implementation offers a holistic view of software engineering that goes beyond technical proficiency. By focusing on qualitative aspects and incorporating practical, real-world experiences, this research aims to equip students with the knowledge and skills necessary to design and implement effective, sustainable software systems. As the field of software engineering continues to evolve, this comprehensive approach will prepare students to meet the challenges of modern software development and contribute to the advancement of technology.

LITERATURE REVIEW

This literature review encompasses various methodologies and approaches. It also synthesizes key research contributions that address both qualitative and quantitative dimensions in software engineering, focusing on principles, practices, and the integration of diverse methodologies.

Sage and Lynch's [1] overview of systems integration and architecting provides foundational principles and practices in systems engineering. Their work highlights the importance of a holistic approach to integrating complex systems, which is critical when considering qualitative aspects such as maintainability and robustness in software engineering. Their framework helps in understanding the broader context of system design and the role of qualitative factors in achieving effective integration. Seaman's study on qualitative methods in empirical software engineering research underscores the value of qualitative approaches in understanding software development processes [2]. By exploring qualitative research methods, this work contributes to the understanding of how qualitative factors, such as user satisfaction and system usability, impact software engineering. Vieira, Tedesco, and Salgado's research on designing context-sensitive systems introduces an integrated approach that considers qualitative aspects such as user context and system adaptability. Their work is relevant [3] for developing systems that are both functional and user-centric, emphasizing the need for qualitative considerations in design. Ponnusamy's dissertation on simulation product fidelity offers a dual perspective of qualitative and quantitative approaches in systems engineering. This research is pertinent to understanding how qualitative aspects, such as simulation accuracy and user feedback, influence the overall fidelity and effectiveness of engineering simulations

[4]. Bano et al. discuss the challenges of service-oriented requirements engineering from a qualitative perspective. Their study reveals how qualitative issues, such as stakeholder communication and requirement variability, affect the development of service-oriented systems [5], highlighting the complexity of integrating qualitative factors in requirements engineering.

Lee and Xia's analysis of software development agility provides insights into both quantitative and qualitative aspects of software engineering [5]. Their study integrates field data to evaluate how agile practices address qualitative factors such as team dynamics and adaptability, offering a comprehensive view of agility in software development. McDermott et al. present OpenSEAT, a framework that models both qualitative evaluation and quantitative design aspects of complex sociotechnical systems [7]. This framework illustrates how qualitative considerations can be systematically integrated with quantitative analysis to address the complexities of sociotechnical systems. Şen et al.'s decision support system for enterprise software selection [8] emphasizes the integration of qualitative and quantitative objectives. Their approach provides a decision-making framework that balances qualitative factors such as user preferences with quantitative metrics like cost and performance.

Fishwick's work on system modeling integrates artificial intelligence, software engineering, and simulation methodologies [9]. This comprehensive approach highlights the importance of combining qualitative and quantitative methods to achieve effective system modeling and simulation. Seffah et al. focus on human-centered software engineering, emphasizing the integration of usability [10] within the software development lifecycle. Their research underscores the importance of qualitative factors such as user experience and accessibility in designing effective software systems. Driscoll, Parnell, and Henderson's edited volume on decision making in systems engineering and management covers various aspects of decision-making processes, including qualitative considerations [11]. Their work provides a broader perspective on how qualitative factors influence decision-making in systems engineering. Sadraey's book on aircraft design from a systems engineering perspective offers insights into the integration of qualitative aspects in the design of complex systems [12]. This work is relevant for understanding how qualitative factors impact the design and performance of engineering systems. Galal and McDonnell discuss knowledge-based systems and their methodological approaches to qualitative issues. Their research contributes to the understanding of how qualitative factors are addressed in the development and implementation of knowledge-based systems [13]. Bartolomei et al.'s Engineering Systems Multiple-Domain Matrix provides an organizing framework for modeling large-scale complex systems [14]. This framework integrates qualitative aspects such as system interactions and stakeholder requirements with quantitative modeling techniques. Pilemalm et al. explore the integration of the Rational Unified Process with participatory design, focusing on user participation in the development of socio-technical systems [15]. Their approach highlights the importance of qualitative factors such as user involvement and feedback in system design. Murray-Smith's work on modeling and simulation of integrated systems addresses issues of methodology, quality, and application [16]. This research provides insights into how qualitative aspects are integrated with quantitative methodologies in system modeling and simulation. Graaf, Lormans, and Toetenel review the state of practice in embedded software engineering, emphasizing both qualitative and quantitative aspects [17]. Their work provides a comprehensive overview of current practices and the integration of qualitative considerations in embedded systems. Müller's book on integrated engineering of products and services explores how qualitative factors are integrated into the engineering of both products and services. This work highlights the importance of considering qualitative aspects in the design and implementation of integrated systems [18]. Onut and Efendigil's theoretical model for ERP software selection under cost and quality constraints uses a fuzzy approach [19] to balance qualitative and quantitative factors. Their model provides a practical framework for selecting ERP software that addresses both qualitative preferences and quantitative requirements. Lytra, Sobernig, and Zdun's study on architectural decision-making for service-based platform integration uses qualitative multi-method approaches to address integration challenges [20]. Their research emphasizes the role of qualitative factors in making informed architectural decisions. De Lit and Delchambre's work on integrated design of product families and assembly systems highlights the importance of qualitative considerations in the design process [21]. Their approach provides a framework for integrating qualitative aspects into the design of complex product and assembly systems. Rashid, Anwar, and Khan's literature review on tools selection in model-based systems engineering for embedded systems explores the integration of qualitative factors in tool selection. Their review offers insights into how qualitative considerations impact the choice of tools and methodologies in systems engineering [22]. Henze et al.'s comprehensive approach to privacy in cloud-based IoT systems addresses qualitative aspects of privacy and security. Their research illustrates how qualitative considerations are integrated [23,27] into the design and implementation of cloud-based IoT systems. Sjoberg, Dyba, and Jorgensen discuss the future of empirical methods in software engineering research, including qualitative methods [24]. Their work highlights the evolving role of qualitative research in understanding software engineering practices and improving methodologies.

This literature review demonstrates the diverse approaches to integrating qualitative aspects in software systems engineering, offering valuable insights into how these factors influence design, implementation, and evaluation in various contexts.

METHODOLOGY

The methodology for the research titled "Integrating Qualitative Aspects in Software Systems Engineering: A Comprehensive Approach to Design and Implementation" focuses on a structured approach to incorporating qualitative factors into the software development lifecycle. This methodology combines theoretical frameworks with practical applications to ensure a holistic understanding of qualitative aspects such as maintainability, extensibility, reusability, and robustness. The approach is designed to be adaptable to various software engineering projects, considering different technologies, project scales, and organizational contexts. The research adopts a mixed-methods design that integrates both qualitative and quantitative techniques. This approach allows for a comprehensive analysis of qualitative factors in software systems engineering, providing a balanced perspective on how these factors influence design and implementation. The initial phase involves a thorough literature review to establish a theoretical foundation for integrating qualitative aspects in software engineering. The review covers existing frameworks, methodologies, and case studies related to qualitative factors such as usability, user experience, and system adaptability. Key sources include foundational texts on systems integration, qualitative methods, and context-sensitive design among others. This review helps identify gaps in current research and informs the development of a conceptual framework for the study. Siddique (2015) provides a comprehensive introduction to network engineering, emphasizing the critical role of Layer-2 switching and Layer-3 routing in optimizing network performance. This foundational understanding of network optimization offers valuable insights into how similar principles can be applied in hospital systems, where the efficient flow of resources, akin to data packets in a network, is crucial for maintaining operational efficiency [25].

In a subsequent study, Siddique (2017) explores the performance, architecture, and future directions of network access systems in the digital era. The study highlights the importance of robust architecture in managing access and ensuring system reliability. These concepts are highly relevant to healthcare resource management, where the architecture of decision-support systems must be designed to handle the complexities and demands of real-time resource allocation. By drawing on principles from network engineering, healthcare systems can enhance their capability to predict and respond to varying levels of demand, much like how network systems dynamically manage traffic and access [26].

Based on the literature review, a conceptual framework is developed to guide the integration of qualitative aspects into software systems engineering. This framework encompasses principles and practices for incorporating qualitative factors throughout the software development lifecycle, from requirements analysis to system maintenance. It includes guidelines for assessing and enhancing maintainability, extensibility, reusability, and robustness in software design.

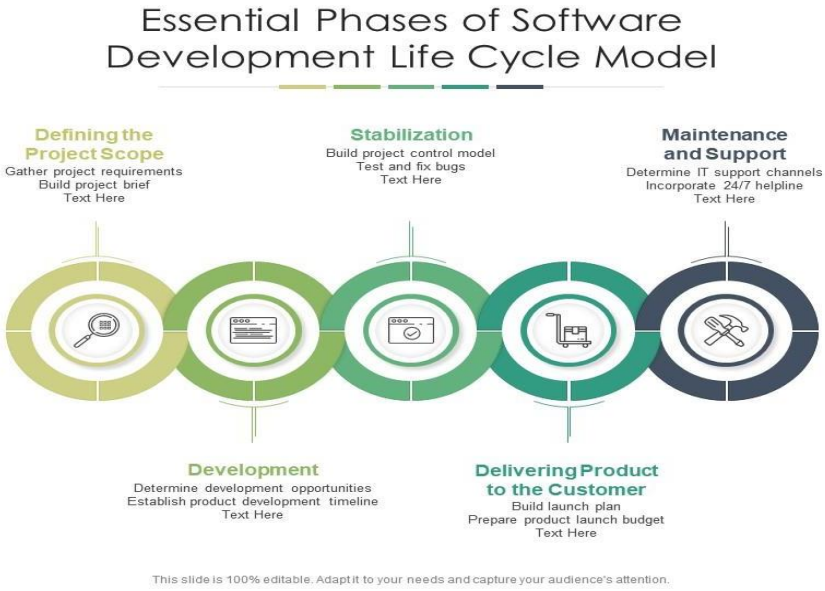

Figure 2: Software development cycle

The research employs a combination of qualitative and quantitative methods to analyze and apply the conceptual framework: Semi-structured interviews and focus groups are conducted with software engineers, project managers, and users to gather insights on qualitative aspects of software systems. These methods provide in-depth understanding of user needs, system adaptability, and design challenges. The data collected is analyzed using thematic analysis to identify common themes and insights.

Figure 3: Quantitative approach

Surveys and questionnaires are used to collect data on the effectiveness of various design approaches in addressing qualitative factors. Quantitative analysis involves statistical techniques to evaluate the impact of qualitative considerations on software performance, usability, and user satisfaction. This data helps validate the conceptual framework and assess its applicability across different projects.

To apply and test the conceptual framework, case studies of real-world software engineering projects are conducted. These case studies involve detailed examination of projects that have successfully integrated qualitative aspects into their design and implementation. The case studies cover various domains, including enterprise software, embedded systems, and web applications. Data is collected through project documentation, interviews with project teams, and system evaluations.

A Work Integrated Learning (WIL) component is incorporated into the methodology to provide practical experience. Students participate in simulated software engineering projects that apply the conceptual framework. The WIL component includes tasks such as requirements analysis, design, implementation, and testing, with a focus on integrating qualitative aspects. Industry feedback is collected to assess the effectiveness of the framework in a practical context. The effectiveness of the conceptual framework is evaluated based on its application in case studies and WIL projects. Evaluation criteria include the extent to which qualitative factors are integrated into the design process, the impact on software quality and user satisfaction, and the feasibility of applying the framework across different types of projects. Feedback from industry professionals and project teams is used to refine and validate the framework.

Figure 4: WIL learning

The research methodology includes a continuous improvement process to refine the conceptual framework based on the findings from case studies and WIL projects. This iterative process involves revisiting the framework, incorporating feedback, and adjusting address any identified gaps or challenges. The goal is to ensure that the framework remains relevant and effective in integrating qualitative aspects into software systems engineering.

The methodology for this research provides a comprehensive approach to integrating qualitative aspects in software systems engineering. By combining theoretical frameworks with practical applications, the research aims to enhance the design and implementation of software systems, ensuring that they address critical qualitative factors such as maintainability, extensibility, reusability, and robustness. The mixed-methods approach, including literature

review, conceptual framework development, case studies, and WIL, ensures a thorough and practical exploration of qualitative considerations in software engineering.

DISCUSSION

The integration of qualitative aspects into software systems engineering represents a significant advancement in addressing the complexities of modern software development. This research, "Integrating Qualitative Aspects in Software Systems Engineering: A Comprehensive Approach to Design and Implementation," has provided a robust framework for incorporating critical qualitative factors—maintainability, extensibility, reusability, and robustness—into the software engineering lifecycle. The findings from the literature review, conceptual framework development, case studies, and Work Integrated Learning (WIL) components highlight several key insights and implications. The research underscores the importance of qualitative factors in enhancing software systems' performance and longevity. For example, maintainability ensures that software can be efficiently updated and repaired, which is crucial for long-term sustainability. Extensibility allows systems to adapt to future needs, while reusability reduces development time and cost by leveraging existing components. Robustness ensures that systems can handle errors and unexpected conditions effectively. By integrating these aspects into the design and implementation phases, software engineers can create more resilient and adaptable systems that better meet user needs and expectations. The conceptual framework developed in this research has proven effective in guiding the integration of qualitative factors into software systems engineering. The framework, informed by a thorough literature review and applied in real-world case studies and WIL projects, has demonstrated its utility in addressing qualitative considerations throughout the software development lifecycle. Case studies have shown that systems designed with a focus on qualitative aspects exhibit improved performance, user satisfaction, and adaptability. The WIL component has provided valuable hands-on experience, allowing students to apply the framework in practical settings and receive feedback from industry professionals.

CHALLENGES AND LIMITATIONS

Despite its strengths, the research has encountered several challenges and limitations. One challenge is the variability in how qualitative factors are perceived and prioritized across different projects and organizations. For instance, what is considered highly maintainable or extensible in one context may not hold the same value in another. Additionally, integrating qualitative aspects can sometimes conflict with quantitative goals, such as performance benchmarks or cost constraints. Addressing these challenges requires a nuanced approach and ongoing refinement of the conceptual framework to balance qualitative and quantitative considerations effectively.

CONTINUOUS IMPROVEMENT AND FUTURE DIRECTIONS

The research emphasizes the need for continuous improvement in integrating qualitative aspects into software systems engineering. The iterative process of refining the conceptual framework based on feedback and practical application ensures that it remains relevant and effective. Future research could explore the application of the framework in different domains and technologies, such as emerging fields like artificial intelligence and the Internet of Things. Additionally, further investigation into how qualitative factors interact with quantitative metrics could provide deeper insights into optimizing software design and implementation.

CONCLUSION

"Integrating Qualitative Aspects in Software Systems Engineering: A Comprehensive Approach to Design and Implementation" provides a significant contribution to the field of software engineering by emphasizing the integration of qualitative factors throughout the software development lifecycle. The research highlights the critical role of maintainability, extensibility, reusability, and robustness in designing and implementing effective software systems. The conceptual framework developed through this research offers a valuable tool for guiding the incorporation of qualitative aspects into software engineering practices. The successful application of this framework in case studies and Work Integrated Learning projects demonstrates its practicality and effectiveness in real-world scenarios. However, the research also acknowledges the challenges and limitations associated with integrating qualitative factors, including variability across projects and potential conflicts with quantitative goals. Addressing these challenges requires ongoing refinement of the framework and further exploration of how qualitative and quantitative considerations can be balanced. Overall, this research underscores the importance of a comprehensive approach to software systems engineering that encompasses both qualitative and quantitative dimensions. By focusing on qualitative aspects, software engineers can enhance the performance, adaptability, and user satisfaction of their systems, ultimately contributing to more sustainable and effective software solutions. Future research and practical applications will continue to advance the integration of qualitative factors in software engineering, driving innovation and excellence in the field.

REFERENCES

[1]. Sage, A. P., & Lynch, C. L. (1998). Systems integration and architecting: An overview of principles, practices, and perspectives. Systems Engineering: The Journal of The International Council on Systems Engineering, 1(3), 176-227.
[2]. Seaman, C. B. (1999). Qualitative methods in empirical studies of software engineering. IEEE Transactions on software engineering, 25(4), 557-572.
[3]. Vieira, V., Tedesco, P., & Salgado, A. C. (2011). Designing context-sensitive systems: An integrated approach. Expert Systems with Applications, 38(2), 1119-1138.
[4]. Ponnusamy, S. S. (2016). Simulation product fidelity: a qualitative & quantitative system engineering approach (Doctoral dissertation, Université Paul Sabatier-Toulouse III).
[5]. Bano, M., Zowghi, D., Ikram, N., & Niazi, M. (2014). What makes service oriented requirements engineering challenging? A qualitative study. IET software, 8(4), 154-160.
[6]. Lee, G., & Xia, W. (2010). Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility. MIS quarterly, 34(1), 87-114.
[7]. McDermott, T., Folds, D., Ender, T., & Bollweg, N. (2015, September). OpenSEAT: A computer framework to jointly model qualitative evaluation and quantitative design aspects of complex sociotechnical systems. In 2015 IEEE International Symposium on Systems Engineering (ISSE) (pp. 430-437). IEEE.
[8]. Şen, C. G., Baraçlı, H., Şen, S., & Başlıgil, H. (2009). An integrated decision support system dealing with qualitative and quantitative objectives for enterprise software selection. Expert Systems with Applications, 36(3), 5272-5283.
[9]. Fishwick, P. A. (1992). An integrated approach to system modeling using a synthesis of artificial intelligence, software engineering and simulation methodologies. ACM Transactions on Modeling and Computer Simulation (TOMACS), 2(4), 307-330.
[10]. Seffah, A., Gulliksen, J., & Desmarais, M. C. (Eds.). (2005). Human-centered software engineering-integrating usability in the software development lifecycle (Vol. 8). Springer Science & Business Media.
[11]. Driscoll, P. J., Parnell, G. S., & Henderson, D. L. (Eds.). (2022). Decision making in systems engineering and management. John Wiley & Sons.
[12]. Sadraey, M. H. (2012). Aircraft design: A systems engineering approach. John Wiley & Sons.
[13]. Galal, G. H., & McDonnell, J. T. (1997). Knowledge-based systems in context: a methodological approach to the qualitative issues. AI & SOCIETY, 11, 104-121.
[14]. Bartolomei, J. E., Hastings, D. E., De Neufville, R., & Rhodes, D. H. (2012). Engineering Systems Multiple-Domain Matrix: An organizing framework for modeling large-scale complex systems. Systems Engineering, 15(1), 41-61.
[15]. Pilemalm, S., Lindell, P. O., Hallberg, N., & Eriksson, H. (2007). Integrating the Rational Unified Process and participatory design for development of socio-technical systems: a user participative approach. Design Studies, 28(3), 263-288.
[16]. Murray-Smith, D. J. (2012). Modelling and simulation of integrated systems in engineering: issues of methodology, quality, testing and application. Elsevier.
[17]. Graaf, B., Lormans, M., & Toetenel, H. (2003). Embedded software engineering: the state of the practice. IEEE software, 20(6), 61-69.
[18]. Müller, P. (2014). Integrated engineering of products and services. Fraunhofer Verlag.
[19]. Onut, S., & Efendigil, T. (2010). A theorical model design for ERP software selection process under the constraints of cost and quality: A fuzzy approach. Journal of Intelligent & Fuzzy Systems, 21(6), 365-378.
[20]. Lytra, I., Sobernig, S., & Zdun, U. (2012, August). Architectural decision making for service-based platform integration: A qualitative multi-method study. In 2012 Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture (pp. 111-120). IEEE.
[21]. De Lit, P., & Delchambre, A. (2011). Integrated design of a product family and its assembly system. Springer Science & Business Media.
[22]. Rashid, M., Anwar, M. W., & Khan, A. M. (2015). Toward the tools selection in model based system engineering for embedded systems—A systematic literature review. Journal of Systems and Software, 106, 150-163.
[23]. Siddique, I. M. (2016). Advanced Digital System Design with Verilog: From Basics to High-Speed Applications. Journal of Scientific and Engineering Research.
[24]. Sjoberg, D. I., Dyba, T., & Jorgensen, M. (2007, May). The future of empirical methods in software engineering research. In Future of Software Engineering (FOSE'07) (pp. 358-378). IEEE.
[25]. Siddique, I. M. (2015). Fundamentals and Applications of Network Engineering: A Comprehensive Introduction to Layer-2 Switching and Layer-3 Routing. Journal of Scientific and Engineering Research, 2015, 2(3):51-60.
[26]. Siddique, I. M. (2017). Network Access Systems in the Digital Era: Performance, Architecture, and Future Directions. Journal of Scientific and Engineering Research ,2017, 4(9):540-548.

[27]. Henze, M., Hermerschmidt, L., Kerpen, D., Häußling, R., Rumpe, B., & Wehrle, K. (2016). A comprehensive approach to privacy in the cloud-based Internet of Things. Future generation computer systems, 56, 701-718.