2023 IEEE 9th Information Technology International Seminar (ITIS)
Batu Malang, Indonesia, October 18 – 20, 2023

Fast Collaboration Competencies Model for Software Development Life Cycle (SDLC)

Muhammad Yusuf
Department of Information System
Universitas Trunojoyo Madura
Bangkalan, Indonesia
muhammadyusuf@trunojoyo.ac.id

M Kautsar Sophan
Department of Information System
Universitas Trunojoyo Madura
Bangkalan, Indonesia
kautsar@trunojoyo.ac.id

Aang Kisnu Darmawan
Department of Information System
Universitas Islam Madura
Pamekasan, Indonesia
ak.darmawan@gmail.com

Budi Dwi Satoto
Department of Information System
Universitas Trunojoyo Madura
Bangkalan, Indonesia
budids@trunojoyo.ac.id

Devie Rosa Anamisa
Department of Information System
Universitas Trunojoyo Madura
Bangkalan, Indonesia
devros_gress@trunojoyo.ac.id

Wahyudi Agustiono
Department of Information System
Universitas Trunojoyo Madura
Bangkalan, Indonesia
wahyudi.agustiono@trunojoyo.ac.id

***Abstract*— Some existing Software Development Life Cycle (SDLC) models exist, such as Waterfall, Spiral, V-Model, Iterative, Big Bang, Agile, and others. SDLC as a framework based on the literature review has six (6) critical problems and weaknesses: poor documentation structures, lack of flexibility, inadequate analysis, confusion in selecting the right SDLC, unsuitable architecture, and lack of adaptability. To overcome the weaknesses of the SDLC, researchers have conducted various studies but have not found satisfactory results. This research aims to propose a novel SDLC model called the Fast Collaboration Competencies (FCC) model to make a better SDLC process. The novelties of this model contain three essential things: fast, collaboration, and competencies. The research produced a Novel FCC for the SDLC process and a list of advantages and disadvantages of the proposed Novel FCC. This research significantly contributes to software development by providing theoretical understanding, practical guidance, and enhanced team dynamics. The FCC model can reduce project delays, improve productivity, and increase customer satisfaction. By sharing findings, industry professionals can prioritize collaboration and teamwork, contributing to ongoing improvement and evolution of software development practices.**

Keywords—Software Development Life Cycle (SDLC), Fast Collaboration Competencies (FCC), Project Management Model, Software Development

I. INTRODUCTION

Software Development Life Cycle (SDLC) is a systematic process of developing and maintaining software. SDLC can be used in different stages of software development, including traditional desktop application development, trending development, and much more [1]. The process of SDLC follows a well-defined set of steps, which include requirements gathering, design, development, testing, and deployment [2]. The models used in SDLC are designed to represent knowledge about the various phases of the life cycle inherent in different models and the possibility of describing the recurrence of phases [3]. Meanwhile, a focused SDLC for specific complex design issues can be useful in understanding diverse user needs [4]. Researchers can also study top management involvement in the various SDLC phases to give guidance on its importance and support for the success of information system projects [5].

However, the literature review shows that SDLC has many crucial problems and weaknesses that must be improved. SDLC has been found to have poor documentation structures [6], lack of flexibility [7], inadequate analysis [8], confusion in selecting the right SDLC [7], [9], unsuitable architecture [10], and lack of adaptability [11]. Researching and selecting the SDLC model that best fits the project's requirements is essential. It is necessary to comprehend the benefits and drawbacks of each SDLC model to determine the optimal model for a given project. Several research studies have tried to solve this, but they have not yielded satisfactory results. Several previous studies were A Multi-Sprint Model for Fault Detection in Agile Software Development [12], Improved Student Collaboration and Communication Through the Software System Development Life Cycle [13], Predicting software problems phase by phase with fuzzy logic and metrics [14] and The Waterfall Software Development Life Cycle Model as a Simulation [15], Secure SDLC framework for adding security into the SDLC process that the CIA drives [16], Assessing and measuring security threats and vulnerabilities in secure software development[17], Formalization of how software development life cycle models are ranked and how they are predicted[18] and Z-SDLC Model: A New Model For Software Development Life Cycle (SDLC)[19].

This research aims to propose a novel SDLC model called the Fast Collaboration Competencies (FCC) model to make a better SDLC process. Developing a model for fast collaboration competencies in the software development life cycle (SDLC) could greatly assist software development project teams and organizations. Collaboration is essential to the success of any SDLC, and possessing skills that facilitate quick and effective teamwork can result in better project outcomes.

The Fast Collaboration Competencies (FCC) model for the Software Development Life Cycle (SDLC) research is essential for various reasons: (1)Improved Collaboration: Software initiatives succeed through collaboration. Fast collaboration helps teams respond to changing requirements, minimize development cycles, and deliver software products faster to satisfy customer objectives. This can boost project productivity and quality, producing better software. (2)Teamwork and Happiness: Collaboration improves teamwork and job happiness. Your research can help create a collaborative workplace where team members feel empowered and supported by stressing fast collaboration competencies. (3)The Fast Cooperation Competencies (FCC) paradigm for SDLC research improves cooperation, project efficiency, team performance, and software development company competitiveness.

This research makes several valuable contributions to the field: (1)Theoretical Contribution: Your research can contribute to the theoretical understanding of collaboration in software development. (2)Practical Guidance: The FCC

2023 IEEE 9th Information Technology International Seminar (ITIS) | 979-8-3503-0683-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/ITIS59651.2023.10420226

model can offer practical guidance to software development teams and organizations. The model can reduce project delays, enhance productivity, and increase customer satisfaction by ensuring smooth and efficient collaboration among team members. (3)Enhanced Team Dynamics: Collaboration is closely tied to team dynamics and cohesion. Focusing on fast collaboration competencies, your findings, and recommendations can inspire industry professionals to prioritize collaboration and teamwork in their organizations. By sharing your research findings with the industry, you can contribute to the ongoing improvement and evolution of software development practices.

II. Literature Review

Gupta et al. (2022) have researched An in-depth look at how software development life cycle models work [11]; the given information needs to provide specific results of the paper as it is a general overview of software development life cycle models. It includes information about the different types of SDLC models and their advantages and disadvantages. Gupta et al. (2021) on a study on how other SDLC models work [2]. The paper is a comparative study of different Software Development Life Cycle (SDLC) models. It explains the advantages and limitations of six SDLC models - Waterfall, Spiral, V, Agile, Iterative, and Rapid Application Development (RAD). The paper primarily aims to explain these models and know their differences. The report needs to provide specific results as it studies and analyzes different SDLC models. Mistarihi et al. (2018) on Business Process Re-Engineering Through the Application of Structural System Analysis and Information Technology [8] present a case study of the implementation of Structural System Analysis (SSA) and Information Technology (IT) in business process re-engineering at Mix Grill restaurant in Irbid, Jordan. The proposed methodology was effective and efficient for the re-engineered To-Be process, significantly enhancing the business process throughput rate, latency, and customer satisfaction. The study recommends using SSA and IT to build data and process models, which can help understand and predict customer needs to gain and sustain a competitive advantage.

Agarwal et al. (2017) on Risk Analysis and Model Selection for the SDLC [20] discuss the comparison of different SDLC models (waterfall, V-shaped, prototype, and RAD) used for software development. It proposes a tool to identify the best-suited model for a given project. The device also incorporates risk management activities within the models, making the product more resource-efficient. However, the paper needs to provide specific results or findings related to the effectiveness of the proposed tool. Karim et al. (2016) on SDLC safe software development: a model and case study: SDLC-secure software development [21], The paper investigates the methodologies being used in software development in Saudi Arabia and describes a model for integrating security into the software development life cycle (SDLC). The aim is to identify the appropriate means of introducing security measures earlier in the SDLC. The research identified various essential elements, such as security standards, policies, processes being practiced, and tools used within SDLC projects. The non-functional security requirements were also found for using FORTIFY and HP ALM for source code review and web application testing. The paper provides recommendations and verification to elicit the appropriate activities for each SDLC phase. MacTavish et al. (2015), on the Systematic Approach to Sustainability through Convincing Design [4], introduce a System Development Life Cycle (SDLC) for persuasive design for sustainability based on cognitive dissonance and SDLC research frameworks. It identifies sensitive issues and factors in design, aiming to subside ethical aspects in persuasion for sustainability. The proposed SDLC is promising for supporting organizations and designers in addressing sustainability issues. However, empirical verification is needed for further exciting results.

Misra et al. (2015) on Modelling an OASDLC (Open Agile Software Development Life Cycle) [22] presents the results of experiments conducted to test the Open Agile Software Development Life Cycle (OASDLC) model. The results show that the overall cost for OASDLC is the lowest (62,500) compared to other methods, as it involves volunteer developers who work with the Cdvp, thereby reducing the cost. The overall cost for Agile SDLC is the highest (10,000) compared to any other method as it involves only skilled Cdvp. Thitisathienkul et al . (2014) on Software document characteristics metric-based software development process document quality evaluation [6] introduces a method for assessing the quality of Software Development Life Cycle (SDLC) documents based on content and structure. It uses measurement processes and information models to define metrics to evaluate SDLC document characteristics. The results can indicate document quality and identify flaws, leading to improved communication and support for software product development. The method's results can be validated by comparing them with experts' expectations and incorporating stakeholder feedback for future measurement processes.

Öztürk et al. (2013) on Fuzzy logic software development lifecycle selection suggest using Fuzzy Logic (FL) to choose the most suitable Software Development Life Cycle (SDLC) for a project based on criteria like requirements, development time, size, complexity, experience, and risks. The FL system was developed and tested, reducing development time, cost, overhead, risk exposure, uncertainty management, quality improvement, client relations promotion, and better project tracking and control. Kumar et al . (2013), A method for making suggestions based on rules for choosing software development life cycle models, introduces a rule-based recommendation system for selecting the most suitable software development life cycle (SDLC) model based on a software product's characteristics. The authors classified software products based on their characteristics and surveyed literature to elicit recommendations. The system provides valuable hints for selecting an SDLC and validates and refines SDLC recommendation rules. The paper also presents a taxonomy for software product classification and compares it with existing works—Ruparelia et al. (2010) on software development process models [23]. The given text information does not provide any specific results of the paper as it is a general overview of software development lifecycle models. It provides a tour of the main SDLC models, discusses their relative merits, and discusses the future of SDLC models. Therefore, there are no specific results to report. Majid et al . (2010), in A survey of practitioners' experiences with user participation in the software development life cycle [24], A study on user involvement in the software development life cycle (SDLC) found that the focus is mainly on functional requirement gathering, with practitioners not involving users in non-functional requirements gathering. The study used the

Human Centered System Development Life Cycle (HCSDLC) model and SPSS version 13.0 for data analysis.

III. Methodology

The determination of this research method was carried out by first studying the relevant literature for framework development steps [25],[20], [24], [22], and [2], and how to evaluate it [17], [21], then about the history of SDLC models and how to analyze them [23].

The research methods on the Fast Collaboration Competencies (FCC) model for the Software Development Life Cycle (SDLC) are described in Fig. 1:

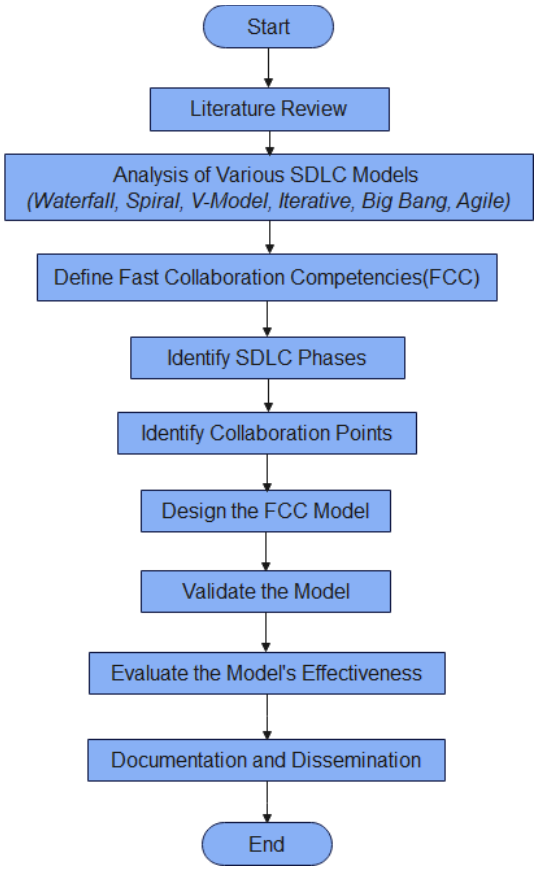

Fig. 1. Research Steps

A. Literature Review

This stage began by conducting a thorough literature review to comprehend the extant theories, models, and frameworks about collaboration competencies within the context of SDLC. This will assist you in identifying any research gaps and determining how your model can add to the existing corpus of knowledge.

B. Analysis of Various SDLC Models

In this stage, the various existing SDLC models are analyzed, namely Waterfall, Spiral, V-Model, Iterative, Big Bang, and Agile, and then comprehensively compare the current methods.

C. Define Fast Collaboration Competencies(FCC)

This stage was defined precisely by what you mean by "fast collaboration competencies" in the context of the SDLC. This could include abilities, behaviors, and dispositions that facilitate effective and efficient teamwork.

D. Identify SDLC Phases

This stage was to understand the relevant phases and locations of the software development life cycle (SDLC). Typical steps include requirements gathering, design, development, testing, deployment, and maintenance.

E. Identify Collaboration Points

This stage determines the critical components of collaboration within each SDLC phase. These are the areas where cooperation is crucial for the successful execution of a project.

F. Design the FCC Model

This stage was to develop a conceptual model that outlines the critical competencies required for rapid collaboration at each collaboration point within the SDLC based on your literature review and understanding of fast collaboration competencies. Consider communication, collaboration, conflict management, decision-making, and problem-solving.

G. Validate the Model

This stage was considering conducting interviews, surveys, or case studies with software development experts to ensure the dependability and validity of your FCC model. Collect their comments and observations to refine and validate the model.

H. Evaluate the Model's Effectiveness

This stage was after establishing the FCC model and evaluating its applicability to real-world scenarios. Contrast the project outcomes and team performance with and without the model's application. This evaluation will assist in demonstrating the worth and influence of your model.

I. Documentation and Dissemination

This stage includes the FCC model, methodology, results, and conclusions in your documentation of research findings. Contribute to information systems and SDLC by sharing your research via academic publications, conferences, or industry forums.

IV. Result and Discussion

The following presents the analysis results obtained after going through a series of steps described in the previous research methodology section.

A. Fast Collaboration Competencies (FCC) Model for SDLC

This research produces a novel model for SDLC, which is called the FCC model, as presented in Figure 2 below :

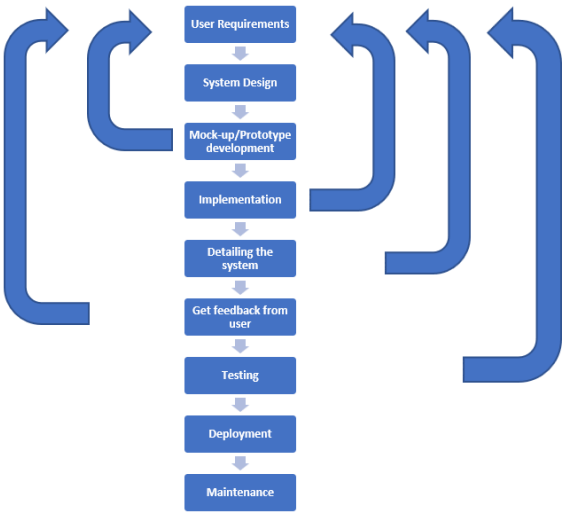

Fig. 2. The proposed FCC Model

The proposed FCC model is presented in Fig 2 that consists of 9 stages below:

1. User requirements: this stage is conducted by a system analyst through the interview, communication, and discussion with the user. Therefore, the system analyst needs some competencies, such as communication, critical thinking, design thinking, and analytical thinking.
2. System analysts also conduct system design using UML and design tools like Power Designer or other frameworks and tools. This stage contains flow design, database design, activity diagram, deployment diagram, and others. In this stage, the system analyst needs competencies such as database designer, business process modeler, and UI/UX designer.
3. Programmers use PHP, Java, Python, and others to create mock-ups or prototypes. The software is a prototype. The development team needs internal and user collaboration at this level. Programmers need programming languages, and this level is redundant. Thus, the programmer needs user-internal developer team communication and collaboration. User requirements will reset the process.
4. Implementation: this stage has the same explanation and collaboration as above. However, implementation in this stage aims to produce the complete product. The programmer in this stage needs some competencies, such as DevOps, managing server machines, programming, and API or platform integration.
5. Detailing system: the programmer conducts this stage to develop more detailed software features. The programmer in this step needs the same competencies and collaborations as in the fourth stage above.
6. Getting feedback from the user: the developer team discusses with the user to get input to improve the product in this stage. The developer team in this stage needs the same competencies and collaborations as in the fourth step above.
7. Testing: the software tester will test, evaluate, and validate the product to ensure that the developed software works correctly. Moreover, the tester needs some competencies, such as testing skills and operating the testing tools. Also, the testing process requires collaboration between the internal development team and the user.
8. Deployment is conducted by the developer team installing the software on the user's computer, laptop, or server. The deployment process needs collaboration between the internal development team and the user.
9. The developer team conducts maintenance to maintain the product, fix errors and bugs, and ensure the software runs well.

This novel model is based on the development of the Madura Herb Database System, a Village Open Data Application, a Digital Village Index (DVI) application, a Geographic Information System (GIS) for Salt Management, an E-Commerce application which is called e-Juwelen.com and other projects in the last two years ago.

B. Comparison of Some previous SDLC Models and the Novel FCC Model

Furthermore, Table 1 compares some previous SDLC models and the novel FCC model, such as Waterfall, Spiral, V-Model, Iterative, Big Bang, Agile, and FCC models.

TABLE I. COMPARISON OF SOME PREVIOUS SDLC MODELS AND THE NOVEL FCC MODEL

Stages in Waterfall Model[26], [27], [28], [29]	**Stages in Spiral Model** [30], [31], [32], [33]	**Stages in V-Model** [34], [35], [36]	**Stages in the Iterative Model** [37], [38], [39], [40], [41]	**Stages in Big Bang Model** [42], [43]	**Stages in Agile Model** [44],[45], [46],[47],[12]	**Stages in the Fast Collaboration Competencies (FCC) Model**
1. Requirements specification 2. Software design 3. Implementation 4. Testing 5. Maintenance	1. Planning 2. Risk analysis 3. Engineering 4. Evaluation 5. Repeat	1. Requirements specification 2. Design 3. Coding 4. Testing 5. Maintenance	1. Planning 2. Design 3. Implementation 4. Testing 5. Feedback 6. Iteration	1. Minimal formal development process and planning 2. Implementing requirements as they come without requiring complete software revamping 3. Suitable for smaller projects with only one or two software engineers required	1. Pre-planning 2. Planning and design 3. Execution or development 4. Review and feedback 5. Retrospective	1. User Requirement 2. System design 3. Mock Up or Prototype development 4. Implementation 5. Detailing system 6. Getting feedback from the user 7. Testing 8. Deployment 9. Maintenance

Based on the comparison above, the novel FCC model complements the previous SDLC process model with its advantages and disadvantages, as shown in Table II. It's important to note that this research contributes three novelties factors in the model: speed, collaboration, and competencies. Therefore, the stages of the Fast Collaboration Competencies (FCC) Model stress the value of fast collaboration skills throughout the process. This means that the FCC Model considers the need for effective teamwork at each stage to ensure that software is produced quickly and well. The FCC Model aims to improve team members' teamwork, communication, and coordination throughout the SDLC by adding fast collaboration competencies and skills. This can improve project results, shorten development processes, and make software development projects more efficient. The Fast Collaboration Competencies (FCC) Model emphasizes the value of fast collaboration skills throughout the process. In this model, the stages of defining requirements, designing software, putting it into action, testing it, and keeping it up to date are crucial times when people must work well together. The FCC Model understands that effective collaboration between team members is critical to finishing projects faster, making the whole team more efficient, and ensuring that software products are of high quality. The FCC Model aims to improve teamwork, communication, and planning by putting fast collaboration skills into each stage. This will speed up the development process and lead to better project results. The FCC Model recognizes collaboration as a critical part of successful software development, which differs from standard SDLC models that may overlook or undervalue it. It acknowledges that fast-paced and effective teamwork is needed to keep up with the needs and requirements of software projects, which change quickly. The FCC Model helps the software development team communicate, make decisions, solve problems, and deal with conflicts by emphasizing collaboration. This collaborative method makes projects run more smoothly and helps team members be happier and do better overall. With its focus on fast collaboration skills, the FCC Model aims to improve how well software development teams work together and, in turn, how well software development projects turn out.

TABLE II. ADVANTAGES AND DISADVANTAGES OF THE NOVEL FCC MODEL

No	Advantages	Disadvantages
1	You can get more detailed features without waiting for the complete product.	It is unsuitable for a small project, such as using less than three features.
2	A competent person conducts each stage and produces a good product.	We need more than two people for each team.
3	Reduce the risk of implementation errors	Need collaboration tools, such as GitHub, Trello, Jira, and others
4	More efficient	Need user involvement in each process
5	Saving time and cost	Development time will be longer if the user requirements constantly change.
6	Can give value to the user in advance	It is not suitable for software development with a transparent process
7	Features are suitable for the user's requirements.	
8	Minimizing unpacking ready-made features	

The Fast Collaboration Competencies (FCC) model allows software developers to obtain more detailed features without waiting for the complete product, competent execution at each stage, reduced risk of implementation errors, increased efficiency, time and cost savings, value delivery to users in advance, alignment with user requirements, and minimizing unused features. However, it is best for larger projects, requires more team members, depends on collaboration tools, requires user involvement, may delay if user requirements change, and is unsuitable for software development with a transparent and rigid process. These pros and cons show the FCC model's pros and negatives, stressing the need to examine the project's context and requirements while applying it to software development. Evaluation of the FCC model is needed based on real projects in future research.

V. CONCLUSION

This study proposes a new SDLC model, the Fast Collaboration Competencies (FCC) model, to improve the SDLC process. The research produced Novel components for the SDLC process, such as fast, collaboration, and competencies. Moreover, the FCC model consists of stages: User Requirement, System design, mock-up or Prototype development, Implementation, Detailing system, Getting feedback from the user, Testing, Deployment, and Maintenance. This research needs further steps with model validation, evaluating the model's effectiveness, and the final stage with model documentation and dissemination based on real software development projects.

ACKNOWLEDGMENT

Thank you to the Information Systems Study Program, Faculty of Engineering, and LPPM Universitas Trunojoyo Madura for the support so that this research can be carried out correctly in 2022.

REFERENCES

[1] Mudita and D. Gupta, "The Aspects of Artificial Intelligence in Software Engineering," *j comput theor nanosci*, vol. 17, no. 9, pp. 4635–4642, Jul. 2020, doi: 10.1166/jctn.2020.9291.

[2] A. Gupta, "Comparative Study of Different SDLC Models," *IJRASET*, vol. 9, no. 11, pp. 73–80, Nov. 2021, doi: 10.22214/ijraset.2021.38736.

[3] T. E. Shulga and D. E. Khramov, "Life cycle ontology of software engineering," *Vestnik of Astrakhan State Technical University. Series: Management, computer science, and informatics*, vol. 2023, no. 2, pp. 66–74, Apr. 2023, doi: 10.24143/2072-9502-2023-2-66-74.

[4] M. M. Mustaquim and T. Nyström, "A System Development Life Cycle for Persuasive Design for Sustainability," in *Persuasive Technology*, T. MacTavish and S. Basapur, Eds., in Lecture Notes in Computer Science, vol. 9072. Cham: Springer International Publishing, 2015, pp. 217–228. doi: 10.1007/978-3-319-20306-5_20.

[5] A. Alzayed and A. Khalfan, "Understanding Top Management Involvement in SDLC Phases," *JSW*, pp. 87–120, May 2022, doi: 10.17706/jsw.17.3.87-120.

[6] P. Thitisathienkul and N. Prompoon, "Quality assessment method for software development process document based on software document characteristics metric," in *Ninth International Conference on Digital Information Management (ICDIM 2014)*, Phitsanulok, Thailand: IEEE, Sep. 2014, pp. 182–188. doi: 10.1109/ICDIM.2014.6991412.

[7] V. Öztürk, "Selection of appropriate software development life cycle using fuzzy logic," *Journal of Intelligent & Fuzzy Systems*, vol. 25, no. 3, pp. 797–810, 2013, doi: 10.3233/IFS-120686.

[8] M. Z. Mistarihi, "The Implementation of Structural System Analysis and Information Technology to Business Process Re-Engineering," in *Seventh International Conference on Advances in Computing, Communication, and Information Technology - CCIT 2018*, Institute

of Research Engineers and Doctors, Oct. 2018, pp. 55–61. doi: 10.15224/978-1-63248-162-7-26.
[9] M. A. Adeagbo, J. E. T. Akinsola, A. A. Awoseyi, and F. Kasali, "Project Implementation Decision Using Software Development Life Cycle Models: A Comparative Approach," *J. Comp. Sci & Applic.*, vol. 28, no. 1, Sep. 2021, doi: 10.4314/jcsia.v28i1.10.
[10] D. Kumar Saini, "Software Testing for Embedded Systems," *IJCA*, vol. 43, no. 17, pp. 1–6, Apr. 2012, doi: 10.5120/6192-8700.
[11] S. Gupta, J. Banga, S. Dabas, and Dr. M. K. Bhatia, "A Comprehensive Study of Software Development Life Cycle Models," *IJRASET*, vol. 10, no. 12, pp. 354–358, Dec. 2022, doi: 10.22214/ijraset.2022.47868.
[12] P. Mishra, A. K. Shrivastava, P. K. Kapur, and S. K. Khatri, "Modeling Fault Detection Phenomenon in Multiple Sprints for Agile Software Environment," in *Quality, IT and Business Operations*, P. K. Kapur, U. Kumar, and A. K. Verma, Eds., in Springer Proceedings in Business and Economics. Singapore: Springer Singapore, 2018, pp. 251–263. doi: 10.1007/978-981-10-5577-5_20.
[13] A. Y. Egwoh and O. F. Nonyelum, "A Software System Development Life Cycle Model for Improved Students Communication and Collaboration," *IJCSES*, vol. 8, no. 4, pp. 1–10, Aug. 2017, doi: 10.5121/ijcses.2017.8401.
[14] H. B. Yadav and D. K. Yadav, "A fuzzy logic based approach for phase-wise software defects prediction using software metrics," *Information and Software Technology*, vol. 63, pp. 44–57, Jul. 2015, doi: 10.1016/j.infsof.2015.03.001.
[15] Y. Bassil, "A Simulation Model for the Waterfall Software Development Life Cycle," *International Journal of Engineering*, vol. 2, no. 5, 2012.
[16] S. Kang and S. Kim, "CIA-level driven secure SDLC framework for integrating security into SDLC process," *J Ambient Intell Human Comput*, vol. 13, no. 10, pp. 4601–4624, Oct. 2022, doi: 10.1007/s12652-021-03450-z.
[17] M. Humayun, N. Jhanjhi, M. Fahhad Almufareh, and M. Ibrahim Khalil, "Security Threat and Vulnerability Assessment and Measurement in Secure Software Development," *Computers, Materials & Continua*, vol. 71, no. 3, pp. 5039–5059, 2022, doi: 10.32604/cmc.2022.019289.
[18] L. Almazaydeh, M. Alsafasfeh, R. Alsalameen, and S. Alsharari, "Formalization of the prediction and ranking of software development life cycle models," *IJECE*, vol. 12, no. 1, p. 534, Feb. 2022, doi: 10.11591/ijece.v12i1.pp534-540.
[19] S. Z. Iqbal and M. Idrees, "Z-SDLC Model: A New Model For Software Development Life Cycle (SDLC)," *International Journal of Engineering and Advanced Research Technology (IJEART)*, vol. 3, no. 2, 2017.
[20] P. Agarwal, A. Singhal, and A. Garg, "SDLC Model Selection Tool and Risk Incorporation," *IJCA*, vol. 172, no. 10, pp. 6–10, Aug. 2017, doi: 10.5120/ijca2017915143.
[21] N. S. A. Karim, A. Albuolayan, T. Saba, and A. Rehman, "The practice of secure software development in SDLC: an investigation through existing model and a case study: The practice of secure software development in SDLC," *Security Comm. Networks*, vol. 9, no. 18, pp. 5333–5345, Dec. 2016, doi: 10.1002/sec.1700.
[22] S. C. Misra and V. Singh, "Conceptualizing open agile software development life cycle (OASDLC) model," *International Journal of Quality & Reliability Management*, vol. 32, no. 3, pp. 214–235, Mar. 2015, doi: 10.1108/IJQRM-08-2013-0127.
[23] N. B. Ruparelia, "Software development lifecycle models," *SIGSOFT Softw. Eng. Notes*, vol. 35, no. 3, pp. 8–13, May 2010, doi: 10.1145/1764810.1764814.
[24] R. A. Majid, N. L. M. Noor, W. A. W. Adnan, and S. Mansor, "A survey on user involvement in Software Development Life Cycle from practitioner's perspectives," in *5th International Conference on Computer Sciences and Convergence Information Technology*, Seoul: IEEE, Nov. 2010, pp. 240–243. doi: 10.1109/ICCIT.2010.5711064.
[25] K. Kumar and S. Kumar, "A rule-based recommendation system for selection of software development life cycle models," *SIGSOFT Softw. Eng. Notes*, vol. 38, no. 4, pp. 1–6, Jul. 2013, doi: 10.1145/2492248.2492269.
[26] A. Satriansyah, D. Ferdiansyah, and J. Rinaldo, "Application Prototype Attendance System Garuda Indonesia's Premium Service Assistant Employees Use The Waterfall Model," *CNAHPC*, vol. 4, no. 1, pp. 35–45, Jan. 2022, doi: 10.47709/cnahpc.v4i1.1189.
[27] R. Armansyah and D. Pratiwi, "Game of the Cursed Prince based on Android," *IJCA*, vol. 179, no. 19, pp. 31–36, Feb. 2018, doi: 10.5120/ijca2018916333.
[28] A. Pataropura, "Online Crowdfunding Platform Information System," *algor*, vol. 3, no. 2, pp. 44–51, Mar. 2022, doi: 10.31253/algor.v3i2.1035.
[29] Z. R. S. Elsi, G. Rohana, and V. Nuranjani, "NEW STUDENT ADMISSIONS INFORMATION SYSTEM WITH CLIENT SERVER BASED SMS GATEWAY," *jitk*, vol. 6, no. 2, pp. 159–166, Feb. 2021, doi: 10.33480/jitk.v6i2.1377.
[30] Universitas Amikom Yogyakarta *et al.*, "The Systematic Literature Review of the spiral development model: Topics, trends, and application areas," *INJURATECH*, vol. 2, no. 2, pp. 154–171, Dec. 2022, doi: 10.34010/injuratech.v2i2.8372.
[31] C. Song, X. Dong, and C. Wang, "Spiral Tip Recognition via Deterministic Learning," *Int. J. Bifurcation Chaos*, vol. 30, no. 06, p. 2050093, May 2020, doi: 10.1142/S0218127420500935.
[32] R. Valerdi, "Lessons From the Father of Software Engineering," *Computer*, vol. 56, no. 1, pp. 133–136, Jan. 2023, doi: 10.1109/MC.2022.3219527.
[33] B.-T. Chu, H. Yu, and A. Dance, "USIS: A Unified Framework for Secured Information System Lifecycle:," presented at the 9th Joint International Conference on Information Sciences (JCIS-06), not available, 2006. doi: 10.2991/jcis.2006.246.
[34] Yuan Shi, Gao Jin-Yue, and 吉林大学物理系，长春 130023, "THE PHASE EFFECT OF A DRIVING FIELD ON SPONTANEOUS EMISSION IN V-MODEL," *Acta Phys. Sin.*, vol. 49, no. 6, p. 1081, 2000, doi: 10.7498/aps.49.1081.
[35] S. Robinson, "Simulation model verification and validation: increasing the users' confidence," in *Proceedings of the 29th conference on Winter simulation - WSC '97*, Atlanta, Georgia, United States: ACM Press, 1997, pp. 53–59. doi: 10.1145/268437.268448.
[36] G. Mazzanti, S. E. Guthrie, A. G. Marangoni, and S. H. J. Idziak, "A Conceptual Model for Shear-Induced Phase Behavior in Crystallizing Cocoa Butter," *Crystal Growth & Design*, vol. 7, no. 7, pp. 1230–1241, Jul. 2007, doi: 10.1021/cg050467r.
[37] S. Wang and P. Apostolellis, "A Student Engagement Evaluation Methodology Inspired from Usability Engineering for Extracting Course Design Requirements," in *2020 ASEE Virtual Annual Conference Content Access Proceedings*, Virtual On line: ASEE Conferences, Jun. 2020, p. 34056. doi: 10.18260/1-2--34056.
[38] F. Johannknecht, M. M. Gatzen, D. Hahn, and R. Lachmayer, "Holistic Life Cycle Costing Approach for Different Development Phases of Drilling Tools," in *Day 1 Mon, November 14, 2016*, Bangkok, Thailand: IPTC, Nov. 2016, p. D012S062R001. doi: 10.2523/IPTC-18714-MS.
[39] F. Tapia, A. McKay, and M. Robinson, "SIMULATION OF FEEDBACK LOOPS IN ENGINEERING DESIGN," *Proc. Des. Soc.*, vol. 1, pp. 2661–2670, Aug. 2021, doi: 10.1017/pds.2021.527.
[40] B. Henderson-Sellers and J. M. Edwards, "The object-oriented systems life cycle," *Commun. ACM*, vol. 33, no. 9, pp. 142–159, Sep. 1990, doi: 10.1145/83880.84529.
[41] L. Kong and T. Yuan, "Use case modeling approach for early aspect acquisition," *SIGSOFT Softw. Eng. Notes*, vol. 34, no. 4, pp. 1–6, Jul. 2009, doi: 10.1145/1543405.1543417.
[42] A. Bröckers, "Variability in Standard Software Products: Introducing Software Product Line Engineering totheInsurance Industry," in *The Essence of Software Engineering*, V. Gruhn and R. Striemer, Eds., Cham: Springer International Publishing, 2018, pp. 91–105. doi: 10.1007/978-3-319-73897-0_6.
[43] S. K. Duraisamy, B. Bass, and S. Mukkavilli, "Embedding Performance Testing in Agile Software Model," *IJSEA*, vol. 12, no. 06, pp. 1–11, Nov. 2021, doi: 10.5121/ijsea.2021.12601.
[44] G. Waja, J. Shah, and P. Nanavati, "AGILE SOFTWARE DEVELOPMENT," *IJEAST*, vol. 5, no. 12, Apr. 2021, doi: 10.33564/IJEAST.2021.v05i12.011.
[45] L. Trihardianingsih, M. Istighosah, A. Y. Alin, and M. R. Ghonim Asgar, "Systematic Literature Review of Trend and Characteristic Agile Model," *j. Teknik inform.*, vol. 16, no. 1, pp. 45–57, May 2023, doi: 10.15408/jti.v16i1.28995.
[46] D. Lahru Riatma, M. Masbahah, A. L. Megasari, and R. A. Fatsena, "Telemedicine Development for Health Center Services Using Agile Methods," *CNAHPC*, vol. 5, no. 1, pp. 46–54, Jan. 2023, doi: 10.47709/cnahpc.v5i1.1987.
[47] A. Khalid, S. A. Butt, T. Jamal, and S. Gochhait, "Agile Scrum Issues at Large-Scale Distributed Projects: Scrum Project Development At Large," *International Journal of Software Innovation*, vol. 8, no. 2, pp. 85–94, Apr. 2020, doi: 10.4018/IJSI.2020040106.