Grenze International Journal of Engineering and Technology, June Issue

Knowledge Sharing in Software Development: A Conceptual Framework

Mitali Chugh
UPES, Dehradun, India
Email: mitalichugh11@gmail.com

***Abstract*— Software development is immensely dependent on knowledge. Thus, software development organizations line up knowledge management (KM) to realistically apply the knowledge and renovate solutions for addressing challenges through the different stages of software development. KM facilitates these organizations to obtain and maintain a competitive advantage, specifically from the perspective of agile software development, wherever it nurtures collaboration and management among collocated or cross-functional development teams.**

The focus of this study is to investigate knowledge sharing in agile software projects and to identify the KS strategies and practices employed by agile teams and practitioners in the software development environment. A conceptual framework has been proposed to facilitate knowledge sharing within both collocated and distributed agile teams. This framework draws upon established theories and practices from knowledge management, organizational behavior, and agile methodologies. It presents a structured approach to understanding and addressing the various factors that influence knowledge sharing within agile teams. The conceptual framework covers elements such as communication, knowledge repositories, iterative development, customer collaboration, and engineering practices.

In essence, this paper acts as a valued source for researchers, practitioners, and software development teams pursuing to enhance knowledge sharing in the agile context. By implementing the potential elements of the conceptual framework, organizations can unlock new possibilities for efficient knowledge transfer, collaboration, and innovation, ultimately driving the success of agile software development projects in an increasingly dynamic and technologically advanced world.

***Index Terms*— first term, second term, third term, fourth term, fifth term and sixth term.**

I. INTRODUCTION

In the present dynamic scenario, software development organizations must acclimatize their both tangible and intangible resources to be agile and robust to cope with the need to develop rapidly (Sepashvili, 2020). A well-organized Knowledge sharing process facilitates dynamic capabilities to enhance performance (Cyfert et al., 2021; Demigha & Kharabsheh, 2019; Michailova & Zhan, 2015). The traditional software development methods are apt for the systems that must be developed in phases i.e., the next phase begins when the previous phase ends. In the present scenario, the necessities and goals of the software projects are so frequently altered that the exhaustive documentation becomes rapidly inappropriate and is a hurdle in project development (Huie et al., 2020). For instance, extensive documentation is given lesser significance as compared to software development. This leads

Grenze ID: 01.GIJET.10.2.208_1

to the condition where experience is the primary source for knowledge use and maintenance instead of comprehensive documentation (Gheorghe et al., 2020).

In agile software development documentation focuses primarily on the knowledge sharing that directly happens within local or distributed teams when various project events are conducted. This leads to several challenges for organizational knowledge management and sooner or later may affect the dynamic and innovative capability of the organization (Huie et al., 2020). Agile development predominantly promotes verbal communication and counts on the experiences of practitioners (Alsaqqa et al., 2020). Hence, in agile development tacit or conceptual knowledge is prioritized over explicit knowledge that can be easily lost as it is present only in the mind of team members and stakeholders (de Castro et al., 2021; Gamble, 2020). The sharing of explicit knowledge is easier as it is communicated and encoded in an email, book, or any other electronic media, etc. However, tacit knowledge sharing is subjective to individual attributes such as trust, motivation, individual perception, etc. (Cho et al., 2020). As a result, organizations are facilitating knowledge sharing by making it a key component of their policies, boosting innovation, and sharing proficiencies among members of the team (Sardjono, 2020).

We are focused on the following research questions:

1. How do development teams share knowledge in an agile software project?
2. What factors influence sharing knowledge in an agile software project?

The paper is structured as: Section 2 includes the background work. Section 3 describes the strategies adopted by software development teams for knowledge creation followed by a detailed discussion of how knowledge sharing is facilitated among team members and proposes a conceptual framework to share knowledge in section 4. Section 5 concludes the paper along with a brief account of future work. We are requesting that you follow these guidelines as closely as possible.

TABLE I. TYPE SIZES FOR CAMERA-READY PAPERS

Type size (pts.)	Appearance		
	Regular	Bold	Italic
8	Table captions,[a] table superscripts		
8	Section titles, tables, table names, first letters in table captions,figure captions, footnotes, text subscripts, and superscripts		
9	References, authors' biographies	Abstract	
9	Authors' affiliations, main text, equations, first letters in section titles		Subheading
10	Authors' names		
20	Paper title		

As always with a conversion to PDF, authors should *very carefully* check a printed copy.

II. BACKGROUND WORK

Agile approaches have been adopted to foster team management, productivity, and knowledge sharing. Agile practices such as pair programming, release and sprint planning, customer collaboration, cross-functional teams, daily scrum meetings, and project retrospectives facilitate agile teams in knowledge sharing (M. Razzak et al., 2014). On the other hand, if the agile teams are distributed sprint planning, daily scrums, sprint reviews, and project retrospective meetings are practiced (Dorairaj et al., 2012). In literature, researchers have presented diverse views on knowledge sharing. Ghobadi and Mathiassen (2016) have proposed an empirical model for managing knowledge-sharing risks in agile software development (Ghobadi & Mathiassen, 2017). The model suggests conceptions and practices for evaluating a project's knowledge-sharing risk profile and expressing a complete resolution approach for risk mitigation. The outcomes reveal how various knowledge-sharing risk management profiles affect project performance outcomes. Moe et al. (2016) discuss the enablement of knowledge sharing in virtual teams (Moe et al., 2016). They have presented a discussion on the usage of the framework for facilitating shared knowledge employed for the distributed virtual agile team in a Product Centre at DNV GL. Waheed et al. (2019) highlight documentation and knowledge vaporization problems in distributed software development teams and propose a framework to deal with the stated problems. The proposed framework is evaluated through an industrial case study and concludes by evaluating the proposed framework using a SUS questionnaire and thematic

analysis that enhances knowledge sharing (Waheed et al., 2019a). Nielsen, Winkler, and Norbjerg (2017) have also proposed a framework for knowledge sharing in the DevOps domain for small and large IT organizations. The framework is helpful in knowledge sharing between the Dev and Ops teams (Nielsen et al., 2017). Cram and Marabelli (2018) in their study have investigated the benefits of knowledge sharing in projects that use both traditional and agile development approaches. Their findings show that the implementation of a hybrid approach permitted the project to exploit the traditional practices that were functioning fine, disable techniques that were floundering, and explore the usage of agile practices in particular capacities (Cram & Marabelli, 2018). Imam and Zaheer (2021) have studied the influence on project success of knowledge sharing and cohesion. The outcomes of the study reveal amplified project success with shared leadership, cohesion, and knowledge sharing (Imam & Kashif, 2021). Similarly, Raziq et al. (2020) have shown the mediating role of knowledge sharing on project success and organizational structure (Raziq et al., 2020). Few other studies in the literature have comprehensively discussed the facilitators and barriers to knowledge sharing (Ali Ekber Akgün Halit Keskin Hayat Ayar Zeki Okunakol, 2017; Karagoz et al., 2020; Khoza, 2017; Prabodha & Vasanthapriyan, 2019). Hence, we can infer the significance of knowledge sharing in different dimensions of agile software development.

III. Knowledge-Sharing Practices In Agile Software Development Teams

Agile software development is the process in which self-organizing and cross-functional teams work to develop software solutions. Several agile methodologies have come into existence as Scrum, Kanban, Extreme programming, crystal methodology, etc. Each method characterizes a specific type of agility; however, these methods are complementary to each other in managing software development. Next, we review the agile practices that are prevalent in literature, nevertheless, without centering on one explicit agile method.

Knowledge sharing in agile development processes is facilitated by various practices such as release and iteration planning, pair programming and pair rotation, on-site customers in the case of XP (Beck, 2000) daily Scrum meetings, cross-functional teams, and project retrospectives in Scrum. The developers and on-site customers share knowledge related to commercial aspects and requirements by using release and iteration planning. At the inception of the project release planning meeting is organized and the project schedule is prepared by breaking the project into smaller iterations and releases. When an iteration begins developers and customers decide the course of action for starting few weeks. As a result of the discussion developers break features into tasks and estimate the efforts required for accomplishing each feature. To predict whether the developers can complete the work proposed by the customer in the upcoming iteration the available working hours in the forthcoming iteration and velocity are taken into consideration. When both customers and developers agree, and no renegotiation must be done a detailed requirement analysis is carried out and a development team member works on the implementation of a feature. The face-to-face interaction between developers and customers enhances trust and thus assists developers in better understanding the requirements. The feedback is directly from customer representatives, so it is better when compared to information exchanged or retrieved through documents.

The development team is cross-functional however it does not mean that all the team members can perform each other's tasks or no two team members can have intersecting abilities. A T-shaped individual knowing broader, and specialized fields is preferred. In addition, rapid iterations, and demonstrations of the developed software to the customers expedite clearing misinterpretations. Another agile practice of pair programming is a vital way of knowledge sharing. Pair team members' designs and look at a piece of code to sort out the problems and at the same time enhance understanding. Often the pairs are rotated as recommended in XP, to widen the influence and share both explicit and tacit knowledge between the pair. This consists of task-related knowledge, contextual knowledge, and social resources. Recently, in a few years mob programming, a substitute for pair programming has developed where the entire team works on designing, coding, and testing together focusing on resolving the same problem at the same time.

An additional way to share knowledge about implementation is peer review of change sets i.e., code review. Debating change sets through pull requests is an innovative way to comprehend new attributes added to the system, checking coding standards and whether the alterations follow architectural principles. One could contend that code reviews are done simultaneously when doing pair and mob programming, however, having a conversation when integrating changes brings up a considerably changed focus.

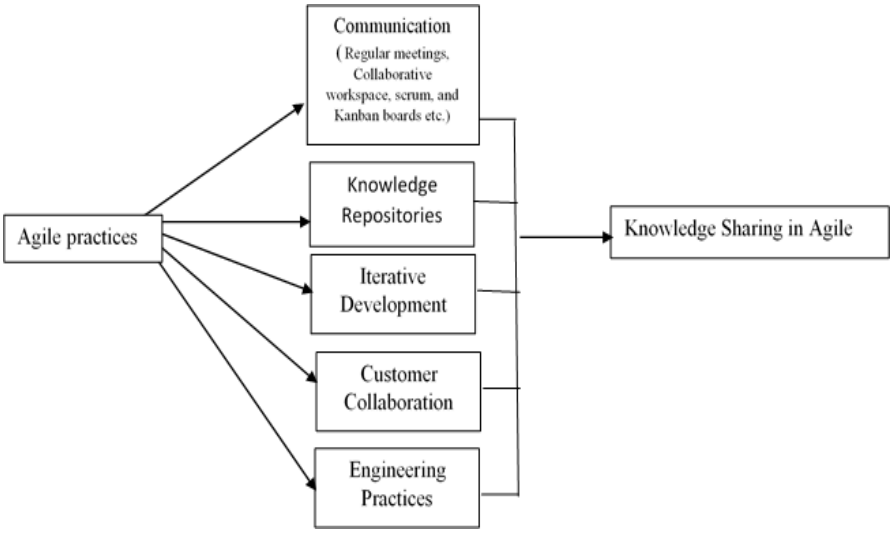
The pair and the mob programming sessions enable communication in a pair or a group, daily Scrum meetings are amongst the best ways to ease communication in the whole team. In daily scrum meetings, work has progressed ever since the last meeting is discussed along with the aims set for the day. The team members also discuss their task-related complications and give recommendations to the teammates related to their tasks. Daily scrum meetings

are the best way to keep teammates updated about one's work. In addition, it also educates each other about the expertise of teammates and boosts regular communication. Teammates come to know whom to interact with when they need information while working on the inexperienced parts of the system. In the cross-functional agile teams' collaboration and knowledge sharing are better which leads to reduced product development time.
Some of the agile methods support constant learning as project retrospectives that help to identify the factors that are facilitators and barriers to the ongoing development process. The phases where the teams come across barriers as overlong stand-up meetings. In cases where team members face obstacles in the current process, such as lengthy stand-up meetings, retrospectives offer the chance for the problems to be conferred and tackled all through the development rather than at the end of the project.

IV. A Conceptual Model To Facilitate Knowledge Sharing

Information sharing and continuous communication among the team members of an agile software development project are important. Both formal and informal communication facilitate information sharing among agile team members. Review meetings and software requirement specification documents foster formal communication, however informal communication is enabled through emails, telephone calls, video calls, face-to-face meetings, etc. (Ahmad et al., 2018). The face-to-face meetings enhance trust among team members (Huri et al., 2019) and thus improve the software development process and product. Agile development needs frequent communication and sharing of information among team members and clients. In addition, agile development involves internal communication between project leads and developers and external communication between clients and developers. Another way for knowledge sharing is technical forums that are like communities of practice meant to build connections among development team members to assist speedy problem solving and decrease the time of development as the team members are not trapped on regular matters. Technical forums have been reported to support knowledge sharing among large-scale team members (Ahmad et al., 2018; Khalil & Khalil, 2020a). Moreover, teams use Kanban and Scrum boards for both local and distributed teams to share knowledge (Khalil & Khalil, 2020b). The studies in the literature consider communication as a significant practice for knowledge sharing in agile software teams (Ahmad et al., 2018; Batra et al., 2019; Waheed et al., 2019b).
Although in agile development interactions are more relied on when compared to exhaustive documentation, knowledge repositories are significant in the process of development. Knowledge repositories capture, store, retrieve, reuse, and share knowledge automatically as an online database. To enhance client and development team interactions knowledge repositories support knowledge sharing regarding the product under development as it is manageable, adaptable, and heterogeneous. The organizations use tools such as wikis for data storage and sharing, this permits information access on former projects hence improving knowledge sharing between agile teams. Knowledge repositories also decrease training costs as explicit knowledge on prior projects is available that prevents the repetition of the same inaccuracies in future projects. Among the distributed locations repositories like JIRA, Wiki, Confluence, GitHub and Redmine, etc. are used. These types of repositories provide efficient mechanisms to access codified knowledge. Many studies in the literature consider knowledge repositories significant for knowledge sharing (M. A. Razzak & Ahmed, 2014; Tams et al., 2020).
The knowledge-sharing initiatives are supported by Agile practices of regular communications and feedback. Furthermore, the developers promptly exchange and bring together their ideas and pass on best practices and codified knowledge. This needs a decent IT infrastructure that is facilitated as an element of knowledge management initiatives to foster informal interaction to enhance network development. The IT tools are essential for agile engineering practices (unit-testing and continuous integration) specifically while development events are performed in geographically distributed locations. Hence, agreeing with the view of researchers, this study also considers engineering practices to support knowledge sharing in agile software development (Khalil & Khalil, 2020b; Ramesh et al., 2010; Singh et al., 2014).
The agile methodology uses iterative development (Khalil & Khalil, 2020b). The iterative development activities such as iteration planning meetings, daily meetings, sprint review meetings, pair programming, quality function deployment, retrospective meetings with the customers, unit testing, and continuous integration are conducted by agile teams to support knowledge sharing. In addition, story cards are presented on the boards in the workspace. The stakeholders and developers also perform brainstorming sessions to discuss challenges and enhance the process of development. In the iterative planning meeting with the customer, functional requirements are discussed. Pair programming and engineering practices such as unit testing and continuous integration are also used to deliver codes with fewer defects. Thus, considering the iterative development, knowledge sharing is enhanced (Khalil & Khalil, 2020a; M. A. Razzak & Ahmed, 2014).

II The clients play an important role in the development process and support deciding the project direction. In the agile development process in every phase of the project, clients are engaged with the developers. The clients are involved in presenting the requirements and performing acceptance testing. The clients are engaged in issue tracking systems, which support the project manager and the developers to early iteration (Khosravi et al., 2018; M. A. Razzak & Ahmed, 2014).



Proposition 1: Knowledge is a vital resource that is shared across varied levels and tasks in the organization. To facilitate knowledge sharing, a general understanding is required that what constitutes knowledge in the organization.
Proposition 2: The factors critical to knowledge sharing in organizations are communication, knowledge repositories, engineering practices, iterative development, and customer collaboration.
Proposition 3: All factors are correlated and collectively they establish a most favourable environment for knowledge sharing within an organization.

V. CONCLUSIONS

The significance of human viewpoint on organizational knowledge sharing is increasing and hence the prospects for researchers in the domain to advance their understanding of knowledge sharing is also growing. The literature survey that led to the conceptual model creation for knowledge sharing explained in this paper put forward opportunities for research in the knowledge-sharing process and factors that influence this process. This research aims to find the knowledge-sharing techniques applied by practitioners in distributed agile projects. To perform knowledge-sharing activities in an agile project, teams practice different types of approaches. But, in general, we found that different types of knowledge-sharing techniques are applied by practitioners to perform knowledge-sharing activities in distributed agile projects. To perform knowledge sharing in an agile project, team members practice pair programming, customer collaboration, Scrum/Kanban boards, technical forums, and technical discussion techniques. Also, knowledge sharing among team members of geographically distributed teams is supported by different types of techniques: repositories, pair programming, version control, screen sharing, daily scrums, common chat rooms, technical forums, etc.
This paper has presented a model that describes knowledge sharing between individuals, identifying factors that have a significant influence on the knowledge-sharing process.
In future works the conceptual framework can be enhanced by incorporating Enhanced Natural Language Processing (NLP) for improved knowledge extraction, understanding, and contextualization within software development repositories, collaborative AI and ML systems in the conceptual framework can enhance collaborative decision-making, model-sharing, and collective knowledge acquisition to enhance the overall knowledge-sharing capabilities of the team. Immersive and Interactive VR Environments facilitate knowledge sharing in software development by supporting the designing of virtual collaborative spaces where team members can work together in a shared virtual environment, enhancing communication, visualization, and interactive problem-solving.

REFERENCES

[1] E. Sepashvili, “Supporting Digitalization: Key Goal for National Competitiveness in Digital Global Economy,” 2020.
[2] S. Demigha and R. Kharabsheh, “Knowledge sharing in an Agile Organization,” in Proceedings of the European Conference on Knowledge Management, 2019, pp. 267–77.
[3] S. Michailova and W. Zhan, “Dynamic capabilities and innovation in MNC subsidiaries,” J. World Bus., vol. 50, no. 3, pp. 576–83, 2015.
[4] S. Cyfert, A. Chwiłkowska-Kubala, W. Szumowski, and R. Mi skiewicz, “The process of developing dynamic capabilities: The conceptualization attempt and the results of empirical studies,” PLoS One, vol. 16, no. 4, 2021.
[5] C. P. Huie, T. Cassaberry, and A. K. Rivera, “The impact of tacit knowledge sharing on job performance,” Int. J. Soc. Educ. Sci., vol. 2, no. 1, pp. 34–40, 2020.
[6] A.-M. Gheorghe, I. D. Gheorghe, and I. L. Iatan, “Agile software development,” Inform. Econ., vol. 24, no. 2, pp. 90–100, 2020.
[7] S. Alsaqqa, S. Sawalha, and H. Abdel-Nabi, “Agile software development: Methodologies and trends,” Int. J. Interact. Mob. Technol., vol. 14, no. 11, pp. 246–70, 2020.
[8] R. O. de Castro, C. Sanin, E. Szczerbicki, and A. Levula, “Where did Knowledge Management Go?: A Comprehensive,” Cybern. Syst., vol. 52, no. 5, pp. 461–76, 2021.
[9] J. R. Gamble, “Tacit vs explicit knowledge as antecedents for organisational change,” J. Organ. Chang. Manag., vol. 33, no. 6, pp. 1123–41, 2020.
[10] S. Y. Cho, J. Happa, and S. Creese., “Capturing tacit knowledge in security operation centers,” IEEE Access, vol. 8, no. 4, pp. 2021–41, 2020.
[11] W. Sardjono, “Analysis of application of Zachman framework for knowledge management systems success optimization.,” in 2020 International Conference on Information Management and Technology, 2020, pp. 277–282.
[12] M. Razzak et al., “Knowledge Management in Distributed Agile Software Development Projects To cite this version : HAL Id : hal-01369807 Knowledge Management in Distributed Agile Software Development Projects : Techniques , Strategies and Challenges,” in 2nd IFIP International Workshop on Artificial Intelligence for Knowl- edge Management (AI4KM), 2014, pp. 107–131.
[13] S. Dorairaj, J. Noble, and P. Malik, “Knowledge Management in Distributed Agile Software Development,” in Agile Conference (AGILE), 2012, pp. 64–73. [Online]. Available: http://ecs.victoria.ac.nz/twiki/pub/Main/TechnicalReportSeries/ECSTR12-08.pdf
[14] S. Ghobadi and L. Mathiassen, “Risks to Effective Knowledge Sharing in Agile Software Teams: A Model for Assessing and Mitigating Risks,” Inf. Syst. J., vol. 27, no. 6, pp. 699–731, 2017, doi: 10.1111/isj.12117.
[15] N. B. Moe, T. E. Faegri, D. S. Cruzes, and J. E. Faugstad, “Enabling knowledge sharing in agile virtual teams,” Proc. - 11th IEEE Int. Conf. Glob. Softw. Eng. ICGSE 2016, pp. 29–33, 2016, doi: 10.1109/ICGSE.2016.30.
[16] S. Waheed, B. Hamid, N. Z. Jhanjhi, M. Humayun, and N. A. Malik, “Improving knowledge sharing in distributed software development,” Int. J. Adv. Comput. Sci. Appl., vol. 10, no. 6, pp. 434–443, 2019, doi: 10.14569/ijacsa.2019.0100656.
[17] P. A. Nielsen, T. J. Winkler, and J. Nørbjerg, “Closing the IT development-operations gap: The devops knowledge sharing framework,” CEUR Workshop Proc., vol. 1898, 2017.
[18] W. A. Cram and M. Marabelli, “Have your cake and eat it too? Simultaneously pursuing the knowledge-sharing benefits of agile and traditional development approaches,” Inf. Manag., vol. 55, no. 3, pp. 322–339, 2018, doi: 10.1016/j.im.2017.08.005.
[19] H. Imam and M. K. Zaheer, “Shared leadership and project success: The roles of knowledge sharing, cohesion and trust in the team,” Int. J. Proj. Manag., vol. 39, no. 5, pp. 463–473, 2021, doi: 10.1016/j.ijproman.2021.02.006.
[20] M. M. Raziq, M. Ahmad, M. Z. Iqbal, M. Ikramullah, and M. David, “Organisational Structure and Project Success: The Mediating Role of Knowledge Sharing,” J. Inf. Knowl. Manag., vol. 19, no. 2, pp. 1–22, 2020, doi: 10.1142/S0219649220500070.
[21] Ali Ekber Akgün Halit Keskin Hayat Ayar Zeki Okunakol, “Knowledge sharing barriers in software development teams: a multiple case study in Turkey,” Kybernetes, vol. 46, no. 4, pp. 1–35, 2017.
[22] Y. Karagoz, N. Whiteside, and A. Korthaus, “Context matters : enablers and barriers to knowledge sharing in Australian public sector ICT projects,” J. Knowl. Manag., vol. 24, no. 8, pp. 1921–1941, 2020, doi: 10.1108/JKM-12-2019-0691.
[23] L. Khoza, “Factors negatively influencing knowledge sharing in software development,” South African J. Inf. Manag., vol. 19, no. 1, pp. 1–9, 2017.
[24] J. Prabodha and S. Vasanthapriyan, “ANALYSIS OF KNOWLEDGE SHARING BARRIERS IN SRI LANKAN SOFTWARE COMPANIES,” Int. J. Knowl. Manag., vol. 15, no. 4, pp. 78–93, 2019.
[25] K. Beck, Extreme Programming Explained: Embrace Change., 2nd ed. Addison-Wesley Professional, Reading, 2000.
[26] M. O. Ahmad, V. Lenarduzzi, M. Oivo, and D. Taibi, “Lessons learned on communication channels and practices in Agile Software Development,” Proc. 2018 Fed. Conf. Comput. Sci. Inf. Syst. FedCSIS 2018, vol. 15, pp. 929–938, 2018, doi: 10.15439/2018F72.
[27] M. Huri, B. Orcid, B. English, and S. T. Orcid, “FACE-TO-FACE COLLABORATIVE LEARNING GROUPS,” no. July, pp. 0–3, 2019.

[28] C. Khalil and S. Khalil, “Exploring knowledge management in agile software development organizations Exploring knowledge management in agile software development organizations,” no. June, 2020, doi: 10.1007/s11365-019-00582-9.
[29] C. Khalil and S. Khalil, “Exploring knowledge management in agile software development organizations Exploring knowledge management in agile software development organizations,” Int. Entrep. Manag. J., no. June, 2020, doi: 10.1007/s11365-019-00582-9.
[30] D. Batra, W. Xia, and M. Zhang, “Collaboration in Agile Software Development : Concept and Dimensions,” no. March, 2019, doi: 10.17705/1CAIS.04120.
[31] S. Waheed, B. Hamid, N. Z. Jhanjhi, M. Humayun, and N. A. Malik, “Improving Knowledge Sharing in Distributed Software Development,” no. January, 2019, doi: 10.14569/IJACSA.2019.0100656.
[32] S. Tams, A. Dulipovici, J. B. Thatcher, K. Craig, and M. Srite, “The role of basic human values in knowledge sharing: How values shape the postadoptive use of electronic knowledge repositories,” J. Assoc. Inf. Syst., vol. 21, no. 1, pp. 201–237, 2020, doi: 10.17705/1jais.00597.
[33] M. A. Razzak and R. Ahmed, “Knowledge sharing in distributed agile projects: Techniques, strategies and challenges,” 2014 Fed. Conf. Comput. Sci. Inf. Syst. FedCSIS 2014, vol. 2, pp. 1431–1440, 2014, doi: 10.15439/2014F280.
[34] A. Singh, K. Singh, and N. Sharma, “Agile knowledge management: a survey of Indian perceptions,” Innov. Syst. Softw. Eng., vol. 10, no. 4, pp. 297–315, 2014, doi: 10.1007/s11334-014-0237-z.
[35] B. Ramesh, L. Cao, and R. Baskerville, “Agile requirements engineering practices and challenges: an empirical study,” Infomation Syst. Journal. Syst. J., vol. 20, pp. 449–480, 2010.
[36] Khosravi, A. R. C. Hussin, and M. Nilashi, “Toward software quality enhancement by Customer Knowledge Management in software companies,” Telemat. Informatics, vol. 35, no. 1, pp. 18–37, 2018, doi: 10.1016/j.tele.2017.09.007.