Knowledge Sharing Framework for Modern Code Review to Diminish Software Engineering Waste

Nargis Fatima[1], Sumaira Nazir[2,] Suriayati Chuprat[3]
Razak Faculty of Technology and Informatics
University Technology Malaysia (UTM), Kuala Lumpur, Malaysia[1, 2, 3]
Faculty of Engineering and Computer Science
National University of Modern Languages (NUML), Islamabad, Pakistan[1, 2]

***Abstract*—Modern Code Review (MCR) is a quality assurance technique that involves massive interactions between team members of MCR. Presently team members of MCR are confronting with the problem of waiting waste production, which results in their psychological distress and project delays. Therefore, the MCR team needs to have effective knowledge sharing during MCR activities, to avoid the circumstances that lead the team members to the waiting state. The objective of this study is to develop the knowledge sharing framework for MCR team to reduce waiting waste. The research methodology used for this study is the Delphi survey. The conducted Delphi survey intended to produce the finalized list of knowledge sharing factors and to recognize and prioritize the most influencing knowledge sharing factor for MCR activities. The study results reported 22 knowledge sharing factors, 135 sub-factor, and 5 categories. Grounded on the results of the Delphi survey the knowledge sharing framework for MCR has been developed. The study is beneficial for software engineering researchers to outspread the research. It can also help the MCR team members to consider the designed framework to increase knowledge sharing and diminish waiting waste.**

Keywords*—Knowledge sharing; modern code review; software engineering wastes; waiting waste; lean software development

I. INTRODUCTION

Software engineering is known as a systematic application of engineering approaches to the development of software [1]. It highly involves social interaction among stakeholders for the development of cost-effective software [2]. It includes sub-activities such as software requirement recognition, software modeling, software testing, inspections, and Modern Code Review (MCR) [3]. These activities yield wastes for instance rework, defect, needless composite solution, waiting, extra or erroneous feature, and mental distress [3], [4]. The various perception of wastes available in the literature are given in Table I.

MCR, a lightweight software engineering activity, has its origin from Fagan's review process [5], [6] and is largely known since 2013 [5], [7]. Fagan's review process is a heavyweight code inspection that requires face to face communications between team members [8]. While MCR is informal, easy-going, and supported through review tools [5], [9]. MCR aims to improve software quality through the improvement of source code quality [5], [10], [11]. It is being practiced by numerous organizations, for instance, Microsoft, Google, etc. [9], [12].

Though MCR has overcome the inadequacies of Fagan's review process [16] and is aimed to enhance the source code quality through widespread knowledge sharing between team members of MCR [5], [9], [12], [10], however, the MCR produces waiting waste due to lack of knowledge sharing [4], [9], [17], [18], [19], [20].

Even though the existing research has paid attention to knowledge sharing concerning software engineering activities [21], [22], [23] however, knowledge sharing in the context of MCR warrants attention from the researchers [9], [10], [12], [24], regarding explorations of knowledge sharing factors for MCR activities [25]. No, schematized inquiries are available about that knowledge sharing facet concerning MCR to decrease waiting waste. Thus, to minimize waiting waste, this study aims to develop a knowledge sharing framework for MCR.

This study is the an extension of our previous work that involved the identification of knowledge sharing factors for MCR through Systematic Literature Review (SLR) and expert review [25]. The result of SLR and expert review are reported in [24], [25]. In our previous studies, the SLR [24], [25] was performed to identify the knowledge sharing factors from the literature and the expert review [25] has been performed to validate the identified list of knowledge sharing factor. In this study, the Delphi survey has been conducted with experts from the industry to finalize the list of knowledge sharing factors, sub-factors and categories for their practicality concerning the industry, to identify and prioritize the most influential knowledge sharing factors for MCR activities, to get suggestion about naming conventions, grouping, and sub-grouping of provided knowledge sharing factors, sub-factors, and categories, to recognize new industry-based knowledge sharing factors, with their associated sub-factors, and categories in the context of MCR. The results of the Delphi survey have been utilized to develop Knowledge sharing framework for MCR to minimize waiting waste.

The remaining paper is organized as Section II describes the research background. The research methodology is discussed in Section III while Section IV introduces the results of the Delphi study. Section V highlights the study conclusion. Section VI highlights future work suggestions. Section VII highpoints the study contribution.

TABLE I. DEFINATION OF OF WASTES FROM LITERATURE

Definition	Reference
"All activities and work products that do not contribute to customer value" or "Everything that is not consider valuable" or "Non- efficient way of working" or "Everything that does not make it to the release i.e. product feature/qualities not delivered and were a waste of time to investigate and or develop"	[2]
"Activities that absorb resources and increase cost without adding value".	[13]
"Any Bottlenecks" or "Waste is anything that does not add value to a product, value as perceived by the customer".	[2], [14]
"Something happens against the flow".	[2], [15]

II. BACKGROUND

Software engineering is a well-disciplined approach to develop quality software [26]. It is social as well as a technical activity that integrates additional activities [3], [27] such as software requirement recognition, software modeling, software testing, inspections, and MCR. These activities generate several wastes [2], [3], [4]. Waste may lead to mental distress, project delays, and software failure. The research on waste recognition and reduction has been started in the 1980s when Toyota revolutionized the automobile industry with a "Lean Manufacturing" [4], [14]. In the year 2000, the lean manufacturing concept was shifted from manufacturing to software engineering domain [28] and was named as lean software development. Since then numerous researches have been reported in the software engineering domain focusing on waste recognition and reduction [2], [3], [4].

In the software engineering domain several wastes have been identified for instance extra or erroneous features, "task switching, defects, "relearning and handoff", needless composite solutions, rework, "extraneous cognitive load, and waiting [2], [3], [4]. Though each software engineering activity includes different software engineering actions, therefore, each activity can generate various distinct wastes [3]. MCR is a critical software engineering activity to improve code quality [5], [29], [30]. In this activity, the reviewer reviews the source code, prior to sending it to the code repository. MCR is supported with the aid of review tools, for instance, Code flow, such as Gerrit, Review board, Phabricator, etc. [5], [9], [10], [12], [31]. Fig. 1 represents the MCR process overview.

It is claimed that waste such as extra or erroneous features, defects, needless composite solutions, rework, and waiting are generated during MCR [2], [3], [4]. It is also conveyed that if the organization needs to minimize one waste, then the organization must emphasize waiting waste [2], [15], [28]. Waiting waste deals with delay between two consecutive activities [3], [4]. For instance, in the case of MCR time delay between source code submission for review by the author and receiving feedback from the reviewer [9], [10]. It is stated that one of the reasons behind waiting waste in MCR is a lack of knowledge sharing [4], [9], [18], [19], [32]. The waiting waste affects the efficiency and productivity of the developers [2], [3], [9], [18], [19], [32], and it also leads to project delays [2].

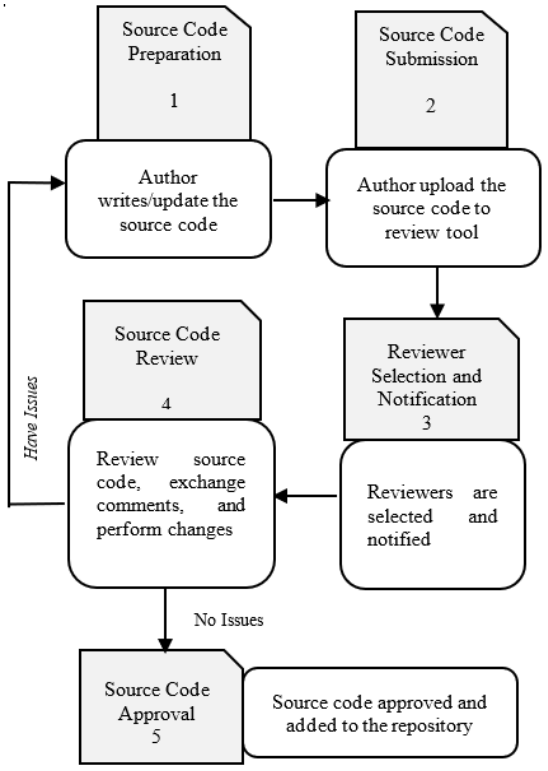


Fig. 1. MCR Process Overview [10].

To diminish the waiting waste it is mandatory to focus on knowledge sharing [2], [3], [4], [33] in MCR. It is reported that knowledge sharing among team members can be augmented by recognizing the factors that can influence knowledge sharing [9], [10], [12], [24]. Considering those factors can aid in knowledge sharing among MCR team members.

Limited researches have been performed concerning knowledge sharing in MCR highlighting the significance of knowledge sharing [10], [12], [34]. For instance, Sadowski et al., (2018), quantify knowledge sharing by looking at comments and files edited or reviewed. They reported that developers build experience through knowledge sharing while working at Google [12]. Similarly, Bosu et al., (2017), stated that code review allows senior developers to mentor newcomers. They conveyed that experienced developers can also enhance their skills while sharing knowledge [10]. Likewise, Rigby and Bird (2013) have explored knowledge sharing facet in code review. They measure the amount of knowledge shared among the MCR team through the number of files known to the developer before and after the source code review [34]. The literature shows that, although existing studies [10], [12], [34] have provided attention towards knowledge sharing in MCR, however, no framework or guidelines are available for effective knowledge sharing in MCR that can help MCR team to reduce software engineering waiting waste. Therefore, this study aims to develop the knowledge sharing framework for MCR to reduce software engineering waiting waste.

III. Research Methodology

Delphi survey has been performed as a research methodology for this study. This methodology is a less costly and relatively competent way to get consensus from the opinions of the experts [35]. It typically involves iterative questionnaires directed to individual experts in such a way that their anonymity is preserved. Feedback received in the Delphi survey after each questionnaire iteration continues until consensus is achieved. The Delphi output is the consensus among the experts along with their observations on the questionnaire items.

A. Objective of Delphi Survey Conduction

A two-round Delphi study has been performed 1) to evaluate the practicality of the identified knowledge sharing factors, sub-factors and their categories in the context of MCR with industry 2) to recognize and prioritize the most influential knowledge sharing factors concerning MCR activities 3) to get suggestion about naming conventions, grouping, and sub-grouping of provided knowledge sharing factors, sub-factors and categories 4) to recognize new industry-based knowledge sharing factors, with their associated sub-factors, and categories in the context of MCR. Delphi survey was conducted based on the guidelines given by [36]. The steps involved in the Delphi survey are detailed in subsections.

B. Delphi Experts' Selection

The selection of the experts to participate in the Delphi study is a very important and critical aspect as the output of the Delphi survey relies on the experts' opinions [36]. Based on the experts' selection requirement conveyed in literature [36], in this study, the experts were selected based on the criteria such as (1) Expert have experience of more than 8 years in the software industry, (2) Expert should have experience in MCR, (3) Expert should have knowledge of wastes in context of software engineering and knowledge sharing. Other selection criteria involve their willingness to participate in the survey as well as enough time to provide feedback [36].

C. Delphi Panel Size

Panel size deals with the number of experts to participate in the study. The panel size varies from a few to hundreds of experts [36]. The size of the panel for the Delphi study is variable. It is conveyed that with a homogenous group of people, ten to fifteen experts might be enough [37]. We requested fifteen experts to participate in the survey. Ten experts showed their interests and willingness to participate.

D. Delphi Rounds

The conducted Delphi survey involved two rounds. The expert's input was collected through questionnaires. The experts were explained each provided knowledge sharing factor, sub-factor, and category to make sure that all of the experts have a shared understanding of knowledge sharing factors. It is conveyed that in the Delphi study, most convergence of panel responses occurs between round one and two [38]. In this study, the consensus among the experts was achieved in two rounds.

E. Delphi Questionnaire Plan

The questionnaire for Round 1 involved four sections. Section A aimed to collect demographic information from the experts. Section B of the questionnaire was composed of a list of knowledge sharing factors, related sub-factors, and categories generated as a result of our previous study based on SLR and expert review [24], [25]. In Section B the experts were also questioned to score the knowledge sharing factors for their practicality and level of influence for MCR activities. Section C was designed to obtain new knowledge sharing factors, sub-factors or categories that should be included in the list. Section C also aimed to collect suggestions about naming conventions, grouping, and sub-grouping of the provided knowledge sharing factors, related sub-factors, or categories. Section D aimed to obtain information about recent real project examples for which the experts had performed MCR activities and experienced the factors influencing knowledge sharing. This section was specifically designed for generating the scenario that was later used in the experiment to validate the developed knowledge sharing framework.

The questionnaire for Round 2 involved three Sections. Section A aimed to evaluate the practicality of knowledge sharing factors finalized after Delphi survey Round 1. The finalized list contains the changes made based on the recommended suggestion of the experts in Round 1. This Section also aimed to evaluate the influence level of listed factors for each MCR activity. Section B aimed to get any new factors, related sub-factors, and categories that should be included in the list. In Section B the experts were also requested to mention the suggestions about the naming conventions grouping and sub-grouping of the provided knowledge sharing factors, sub-factors, and categories. Section C aimed to obtain information about recent real project examples for which the experts had performed MCR activities and experienced the factors influencing knowledge sharing.

F. Pilot Study

The questionnaires were evaluated by five software engineering researchers for their understanding and clarity as it is conveyed that if the questionnaires are used in research, then they should be pretested for length, clarity, and overall adequacy [39]. In the pilot test of this study, the received response was positive and no changes were suggested.

G. Data Analaysis Procedure

Descriptive statistics have been performed in this study as it is a rudimentary analytical approach. These give a basic quantitative strategy for examination and produce a general overview of the outcomes [40].

To score the practicality of knowledge sharing factors and to evaluate the level of influence of knowledge sharing factors for each MCR activity, a five-point Likert scale that is from 1 to 5 (Very High- 5, High - 4, Moderate - 3, Low- 2, Very Low – 1) was provided. For calculating the practicality of knowledge sharing factors and to recognize the most influential Knowledge sharing factors for MCR activities, the mean values were grouped into the discrete categories as shown in Table II for MCR activities.

TABLE II. GROUPING OF MEAN VALUES TO MEASURE PRACTICALITY

Mean Score =X	Practicality Level	Influence Level
4.0≤X≤ 5.0	Very High	Most Influential
3.0≤X< 4.0	High	Influential
2.0≤X< 3.0	Moderate	Moderate
1.0≤X< 2.0	Low	Weakly Influential
0≤X< 1.0	Very Low	Not Influential

The mean practicality and mean influential values of sub-factors were premeditated initially and then the found mean values were further transformed into a single composite mean value showing composite mean practicality and composite mean influence value for the associated knowledge sharing factors.

To get the consensus of the practicality and the influential values of knowledge sharing factors we used the standard deviation as shown in Table III. Initially, we calculated the standard deviation of the sub-factors that were further transformed into a single composite standard deviation for the associated knowledge sharing factor. Based on the obtained composite standard deviation of the knowledge sharing factors we come up with the consensus level among the experts. We formulated equation (1) based on guidelines given by [41] [41] to calculate the composite standard deviation of knowledge sharing factors.

$$SD(KSF) = \sqrt{\frac{(SD\ (SbF_1))^2 + \cdots + \ (SD\ (SbF_k))^2}{k}} \quad (1)$$

Where '*SD*' denotes to standard deviation, '*KSF*' refers to knowledge sharing factor. '*SbF*' refers to the sub-factor of the associated knowledge sharing factor and it ranges from 1 to k, '*k*' refers to the total number of sub-factors for associated knowledge sharing factors.

Table III represents the level of consensus used in this study. A standard deviation between '0' and '1' shows that the experts scoring is very close to each other, whereas a higher standard deviation showed that the experts' scoring was spread out over a large range [35].

H. Data Collection and Analaysis Methods

This section presents the data collected from Delphi experts and the analysis of the data collected depending on the analysis procedure defined in sub-section 'G'. The performed Delphi study involved two rounds. The details concerning data collection are discussed in the following sub-sections.

TABLE III. DECISION CRITERIA FOR THE LEVEL OF CONSENSUS

Standard Deviation (SD=X)	Level of Consensus
0 ≤ X <1	High
1 ≤ X <1.5	Fair Level
1.5 ≤ X <2	Low Level
2 < X	No Consensus

I. Delphi Round 1

In the Delphi Round 1, the questionnaire was given to the experts. They were given one week to complete the questionnaire. The phone calls were made to make sure that all experts were aware of the feedback submission date and time for Round 1. Round 1 of the Delphi survey was completed in two weeks. Round 1 aimed to collect demographic information from the experts. It also aimed to evaluate the list of provided knowledge sharing factors, related sub-factors, and categories for their naming convention, grouping, and sub-grouping which was generated as a result of our previous study based on SLR and expert review [24], [25], [42]. Round 1 involves the evaluation of the knowledge sharing factors for their practicality for the complete MCR process as well as their influence level for each MCR activity. In Round 1, the experts were also enquired to state any new industry-based knowledge sharing factors, related sub-factors, and categories that should be included in the list. The scale used to score the practicality and influence level is given in sub-section 'G'. The details about the Round 1 questionnaire is provided in sub-section 'E'. In Delphi Round 1 some recommendations were suggested by the expert so we need to conduct another Delphi round to have consensus on the suggested changes among the experts.

J. Delphi Round 2

In Round 2, the experts were given the summary of results obtained in Round 1. In Delphi Round 2 the experts were enquired to evaluate the level of practicality as well as the level of influence of subsequent knowledge sharing factors finalized after Round 1 for each MCR activity. In Round 2, the analysis method and the scoring scale was similar as in the case of Round 1. The details about the Round 2 questionnaire is provided in sub-section 'E'. Round 2 also took 2 weeks to be completed. In Round 2 the consensus was obtained for all the knowledge sharing factors therefore we stopped at the Delphi Round 2.

IV. RESULTS

This section presents the results obtained in the two Rounds of Delphi study. The results were then analyzed, and composite mean values of knowledge sharing factors were calculated based upon the mean values of their associated sub-factors. Similarly, the mean influential values of knowledge sharing factors were calculated based upon the mean influential values of their associated sub-factors. The practicality level of each knowledge sharing factors along with the standard deviation for Delphi Round 1 and Delphi Round 2 are shown in Fig. 2 and Fig. 3. Fig. 2 shows that all the provided knowledge sharing factors in both rounds were perceived as practical by the experts as the composite mean value of all the factors lies between 3 and 5. Fig. 3 shows that the level of consensus was increased in Round 2 for the practicality of the identified knowledge sharing factors among the experts.

Table IV shows the ranking of knowledge sharing factors for their level of practicality.

Regarding the most influential knowledge sharing factors, the mean influential values of sub-factors of each knowledge sharing factor in final Delphi Round for; Source Code Preparation values were from 1.4 to 5.0, Source Code Submission values were from 1.4 to 5.0, Reviewer Selection and Notification values were from 1.1 to 5.0, Source Code Review ranges from 2.4 to 5.0, Source Code Approval values were from 2.0 to 5.0. The most influential factors were identified by calculating the composite mean influential value of their connected sub-factors. The factors with composite mean values equal to or above 4.00 were considered as the most influential factors for particular MCR activity. The most influential factors grounded on their composite mean values for each MCR activity after the final Delphi Round are shown in Tables V to IX along with the standard deviation.

Based on the Delphi survey results we formulated a knowledge sharing framework for MCR to diminish waiting waste. The developed framework constitutes knowledge sharing factors, sub-factors, and categories as well as the most influential knowledge sharing factors for each MCR activity. The developed knowledge sharing framework is attached in Appendix A.

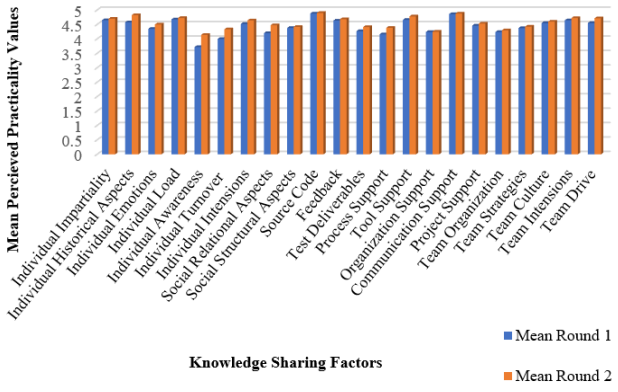


Fig. 2. Composite mean Perceived Value of Practicality of Knowledge Sharing Factors (Round 1 and Round 2).

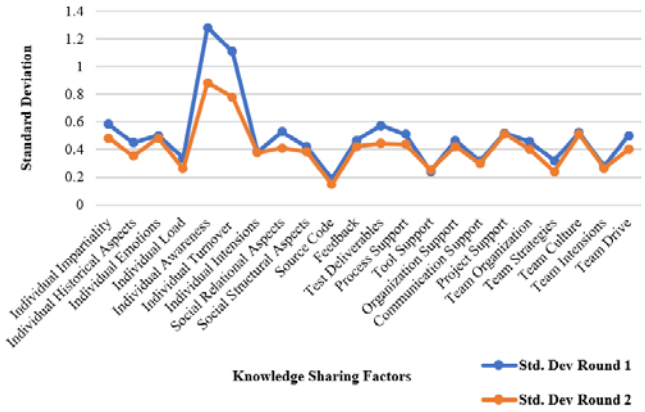


Fig. 3. Consensus Level among the Panelists for mean Perceived Values of Practicality of Knowledge Sharing Factors (Round 1 and Round 2).

TABLE IV. RANKING OF KNOWLEDGE SHARING FACTORS FOR PERCEIVED LEVEL OF PRACTICALITY

Knowledge Sharing Factors	Composite Mean Practicality Values	Standard Deviation	Rank
Source Code	4.9	0.146176337	1
Communication Support	4.88	0.298142397	2
Individual Historical Aspects	4.825	0.353553391	3
Tool Support	4.78	0.253859104	4
Individual Load	4.725	0.263523138	5
Team Intensions	4.722	0.265274142	6
Team Drive	4.714	0.402373908	7
Individual Impartiality	4.7	0.483045892	8
Feedback	4.68	0.423515147	9
Individual Intensions	4.64	0.377123617	10
Team Culture	4.6	0.510990324	11
Project Support	4.53	0.512799145	12
Individual Emotions	4.5	0.483045892	13
Social Relational Aspects	4.475	0.411636301	14
Team Strategies	4.425	0.241522946	15
Social Structural Aspects	4.4167	0.382486988	16
Test Deliverables	4.411	0.443053379	17
Process Support	4.383	0.436738756	18
Individual Turnover	4.333	0.779363463	19
Team Organization	4.3	0.402768199	20
Organization Support	4.25	0.421637021	21
Individual Awareness	4.14	0.880656321	22

TABLE V. INFLUENTIAL LEVEL OF KNOWLEDGE SHARING FACTORS FOR SOURCE CODE PREPARATION

Most influential Knowledge Sharing Factors	Composite Mean Influential Value	Standard Deviation	Rank
Source Code	4.92	0.170469437	1
Tool Support	4.63	0.357460176	2
Individual Historical Aspects	4.6	0.349602949	3
Team Strategies	4.57	0.437797518	4
Team Drive	4.44	0.311167795	5
Team Organization	4.44	0.359010987	6
Organization Support	4.4	0.357460176	7
Individual Load	4.37	0.337474279	8
Project Support	4.33	0.434613494	9
Feedback	4.1	0.333333333	10
Test Deliverables	4.08	0.293972368	11
Process Support	4.06	0.380058475	12
Individual Intensions	4.06	0.418993503	13
Individual Awareness	4	0.837987006	14

TABLE VI. INFLUENTIAL LEVEL OF KNOWLEDGE SHARING FACTORS FOR SOURCE CODE SUBMISSION

Most influential Knowledge Sharing Factors	Composite Mean Influential Value	Standard Deviation	Rank
Tool Support	4.51	0.202758751	1
Source Code	4.47	0.395487366	2
Test Deliverables	4.45	0.472712164	3
Team Strategies	4.4	0.45338235	4
Process Support	4.25	0.275546595	5
Project Support	4.18	0.215165741	6
Organization Support	4.1	0.298142397	7
Individual Historical Aspects	4	0.387298335	8

TABLE VII. INFLUENTIAL LEVEL OF KNOWLEDGE SHARING FACTORS FOR REVIEWER SELECTION AND NOTIFICATION

Most influential Knowledge Sharing Factors	Composite Mean Influential Value	Standard Deviation	Rank
Individual Historical Aspects	4.88	0.300462606	1
Social Structural Aspects	4.83	0.36004115	2
Social Relational Aspects	4.77	0.411636301	3
Individual Impartiality	4.7	0.471404521	4
Tool Support	4.52	0.194365063	5
Team Strategies	4.47	0.418330013	6
Team Culture	4.4	0.45338235	7
Organization Support	4.3	0.223606798	8
Project Support	4.15	0.129099445	9
Source Code	4.13	0.359248979	10
Process Support	4.1	0.403686714	11

TABLE VIII. INFLUENTIAL LEVEL OF KNOWLEDGE SHARING FACTORS FOR SOURCE CODE REVIEW

Most influential Knowledge Sharing Factors	Composite Mean Influential Value	Standard Deviation	Rank
Source Code	4.961	0.191708468	1
Individual Load	4.925	0.263523138	2
Test Deliverables	4.822	0.3022549	3
Communication Support	4.76	0.4163332	4
Individual Intensions	4.7	0.27080128	5
Tool Support	4.7	0.266666667	6
Feedback	4.68	0.191070501	7
Individual Impartiality	4.65	0.5	8
Team Intensions	4.53	0.210818511	9
Process Support	4.43	0.370185139	10
Individual Emotions	4.35	0.341565026	11
Individual Historical Aspects	4.3	0.278886676	12
Project Support	4.2	0.344265186	13
Organization Support	4.175	0.383695481	14
Team Strategies	4.1	0.25819889	15
Team Drive	4	0.338061702	16

TABLE IX. INFLUENTIAL LEVEL OF KNOWLEDGE SHARING FACTORS FOR SOURCE CODE APPROVAL

Most influential Knowledge Sharing Factors	Composite Mean Influential Value	Standard Deviation	Rank
Source Code	4.884	0.269535847	1
Individual Historical Aspects	4.75	0.353553391	2
Team Strategies	4.65	0.5	3
Tool Support	4.6	0.274873708	4
Project Support	4.53	0.327730693	5
Process Support	4.5	0.36004115	6
Organization Support	4.4	0.45338235	7
Team Culture	4.2	0.414996653	8
Individual Impartiality	4	0.459468292	9

V. CONCLUSION

Knowledge sharing plays a significant role in the minimization of waiting waste. This study involves statistical analysis of knowledge sharing factors to identify the list of most influential knowledge sharing factors for MCR activities. The study results reported 22 knowledge sharing factors, 135 sub-factor, and 5 categories. The obtained results were expressed as a knowledge sharing framework for MCR to diminish software engineering waste. This framework will guide the software engineers involved in MCR activities to effectively share knowledge and reduce the production of waiting waste.

VI. FUTURE WORK DIRECTIONS

This developed knowledge sharing framework is specific to the MCR activity of Software Engineering to diminish waiting waste. The study can be further extended to other software engineering activities to minimize waiting waste in other software engineering activities for instance requirement engineering, modeling, and testing. This research study delivers a list of factors influencing knowledge sharing in MCR to diminish waiting waste. Our ongoing research activities are 1) to validate the developed knowledge sharing framework regarding minimization of waiting waste through experiment, 2) to develop a web-based knowledge sharing framework for MCR to have an electronic knowledge sharing guideline for software engineers involved in MCR activities to minimize waiting waste.

VII. CONTRIBUTION

The investigation contributed to software engineering body of knowledge (SWEBOK), knowledge base software engineering (KBSE), and green software engineering (GREEN SE) by stressing the significance of knowledge sharing, most influencing knowledge sharing factors, and by providing the knowledge sharing framework for MCR to diminish waiting waste. The work can guide software engineers to effectively share knowledge by managing the undesirable facets of identified factors.

REFERENCES

[1] DeFranco and P. A. Laplante, “Review and Analysis of Software Development... - Google Scholar,” IEEE Trans. Prof. Commun., vol. 00, no. 00, pp. 1–18, 2017.

[2] H. Alahyari, T. Gorschek, and R. Berntsson Svensson, "An exploratory study of waste in software development organizations using agile or lean approaches: A multiple case study at 14 organizations," Inf. Softw. Technol., vol. 105, no. August 2018, pp. 78–94, 2019.

[3] T. Sedano and P. Ralph, "Software Development Waste," in Proc. IEEE/ACM 39th International Conference on Software Engineering, 2017.

[4] N. Fatima, S. Nazir, and S. Chuprat, "Software engineering wastes-A perspective of modern code review," ACM Int. Conf. Proceeding Ser., pp. 93–99, 2020.

[5] A. Bacchelli and C. Bird, "Expectations, outcomes, and challenges of modern code review," in Proc. International Conference on Software Engineering, 2013, pp. 712–721.

[6] S. Nazir, N. Fatima, and S. Chuprat, "Situational factors affecting Software Engineers Sustainability: A Vision of Modern Code Review," in 6th IEEE International Conference on Engineering Technologies and Applied Sciences (ICETAS) , in press, 2019.

[7] S. Nazir, N. Fatima, and S. Chuprat, "Does Project Associated Situational Factors have Impact on Sustainability of Modern Code Review Workforce?," in 6th IEEE International Conference on Engineering Technologies and Applied Sciences (ICETAS) , in press, 2019, pp. 1–5.

[8] M. E. Fagan, "Design and code inspections to reduce errors in program development," IBM Syst. J., vol. 38, no. 2.3, pp. 258–287, 1999.

[9] L. MacLeod, M. Greiler, M. A. Storey, C. Bird, and J. Czerwonka, "Code Reviewing in the Trenches: Challenges and Best Practices," IEEE Softw., vol. 35, no. 4, pp. 34–42, 2018.

[10] A. Bosu, J. C. Carver, C. Bird, J. Orbeck, and C. Chockley, "Process Aspects and Social Dynamics of Contemporary Code Review: Insights from Open Source Development and Industrial Practice at Microsoft," IEEE Trans. Softw. Eng., vol. 43, no. 1, pp. 56–75, 2017.

[11] S. Nazir, N. Fatima, and S. Chuprat, "Modern code review benefits-primary findings of a systematic literature review," in ACM International Conference Proceeding Series, 2020, pp. 210–215.

[12] C. Sadowski, E. Söderberg, L. Church, M. Sipko, and A. Bacchelli, "Modern code review: : A Case Study at Google," in Proc. ACM/IEEE 40th International Conference on Software Engineering: Software Engineering in Practice, 2018, pp. 181–190.

[13] M. V. P. Pessôa, W. Seering, and E. Rebentisch, "Understanding the waste net: A method for waste elimination prioritization in product development," Proc. DETC '08, vol. 55, no. 21, pp. 1–9, 2008.

[14] S. Mujtaba, R. Feldt, and K. Petersen, "Waste and lead time reduction in a software product customization process with value stream maps," Proc. Aust. Softw. Eng. Conf. ASWEC, pp. 139–148, 2010.

[15] J. Urrego, R. Munoz, M. Mercado, and D. Correal, "Archinotes: A global agile architecture design approach," Lect. Notes Bus. Inf. Process., vol. 179 LNBIP, pp. 302–311, 2014.

[16] S. Nazir, N. Fatima, and S. Malik, "Effective hybrid review process (EHRP)," Proc. - Int. Conf. Comput. Sci. Softw. Eng. CSSE 2008, vol. 2, pp. 763–771, 2008.

[17] G. Gousios, M.-A. Storey, and A. Bacchelli, "Work practices and challenges in pull-based development," in Proc. 38th International Conference on Software Engineering, 2016, pp. 285–296.

[18] E. W. dos Santos and I. Nunes, "Investigating the Effectiveness of Peer Code Review in Distributed Software Development," in Proc. 31st Brazilian Symposium on Software Engineering, 2017, pp. 84–93.

[19] D. M. German, U. Rey, and J. Carlos, "' Was my contribution fairly reviewed ?' A Framework to Study the Perception of Fairness in Modern Code Reviews," in Proc. ACM/IEEE 40th International Conference on Software Engineering Synthesizing, 2018, no. 2, pp. 523–534.

[20] L. Novikova, "Poor knowledge sharing is the second biggest challenge for software development teams," 2019. [Online]. Available: https://blog.onebar.io/poor-knowledge-sharing-is-the-second-biggest-challenge-for-software-development-teams-a4843f9b9aa. [Accessed: 10-Aug-2019].

[21] R. Anwar, M. Rehman, K. S. Wang, A. Amin, and R. Akbar, "Conceptual framework for implementation of knowledge sharing in global software development organizations," ISCAIE 2017 - 2017 IEEE Symp. Comput. Appl. Ind. Electron., pp. 174–178, 2017.

[22] X. Chen, Y. Zhou, D. Probert, and J. Su, "Managing knowledge sharing in distributed innovation from the perspective of developers: empirical study of open source software projects in China," Technol. Anal. Strateg. Manag., vol. 29, no. 1, pp. 1–22, 2017.

[23] N. S. Safa and R. Von Solms, "An information security knowledge sharing model in organizations," Comput. Human Behav., vol. 57, pp. 442–451, 2016.

[24] N. Fatima, S. Nazir, and S. Chuprat, "Knowledge sharing, a key sustainable practice is on risk: An insight from Modern Code Review," in 2019 6th IEEE International Conference on Engineering Technologies and Applied Sciences (ICETAS), 2019, pp. 1–6.

[25] N. Fatima, S. Nazir, and S. Chuprat, "Knowledge sharing factors for modern code review to minimize software engineering waste," Int. J. Adv. Comput. Sci. Appl., vol. 11, no. 1, pp. 490–497, 2020.

[26] P. Bourque and R. E. Fairley, Software Engineering - Body of Knowledge. 2014.

[27] S. Nazir, N. Fatima, and S. Chuprat, "Individual Sustainability Barriers and Mitigation Strategies: Systematic Literature Review Protocol," in 2019 IEEE Conference on Open System, ICOS 2019, 2019, pp. 1–5.

[28] M. Poppendieck and T. Poppendieck, Lean Software Development: An Agile Toolkit. 2003.

[29] N. Fatima, S. Nazir, and S. Chuprat, "Individual, Social and Personnel Factors Influencing Modern Code Review Process," 2019 IEEE Conf. Open Syst. ICOS 2019, pp. 40–45, 2019.

[30] S. Nazir, N. Fatima, and S. Chuprat, "Situational factors for modern code review to support software engineers' sustainability," Int. J. Adv. Comput. Sci. Appl., vol. 11, no. 1, pp. 498–504, 2020.

[31] N. Fatima, S. Chuprat, and S. Nazir, "Challenges and Benefits of Modern Code Review-Systematic Literature Review Protocol," in Proc. International Conference on Smart Computing and Electronic Enterprise, 2018, pp. 1–5.

[32] O. Kononenko, O. Baysal, and M. W. Godfrey, "Code Review Quality: How Developers See It," in Proc. International Conference on Software Engineering, 2016, pp. 1028–1038.

[33] A. Ram, Achyudh; Sawant, Anand; Castelluccio, Marco; Bacchelli, "What Makes a Code Change Easier to Review? An Empirical Investigation on Code Change Reviewability," in Proc. ESEC/FSE, 2018.

[34] P. C. Rigby and C. Bird, "Convergent Contemporary Software Peer Review Practices Categories and Subject Descriptors," in Proc. ESEC/FSE, 2013, pp. 202–212.

[35] H. A. von der Gracht, "Consensus measurement in Delphi studies. Review and implications for future quality assurance," Technol. Forecast. Soc. Change, vol. 79, no. 8, pp. 1525–1536, 2012.

[36] G. J. Skulmoski, F. T. Hartman, and Jennifer Krahn, "The Delphi Method for Graduate Research," J. Inf. Technol. Educ., vol. 6, 2007.

[37] T. Hatcher and S. Colton, "Using the internet to improve HRD research: The case of the web-based Delphi research technique to achieve content validity of an HRD-oriented measurement," J. Eur. Ind. Train., vol. 31, no. 7, pp. 570–587, 2007.

[38] H. W. Lanford, Technological forecasting methodologies; a synthesis. New York: American Management Association, 1972.

[39] D. F. Polit and C. T. Beck, Nursing research: generating and assessing evidence for nursing practice. Lippincott Williams and Wilkins, Philadelphia, 9th ed. LWW, 2011.

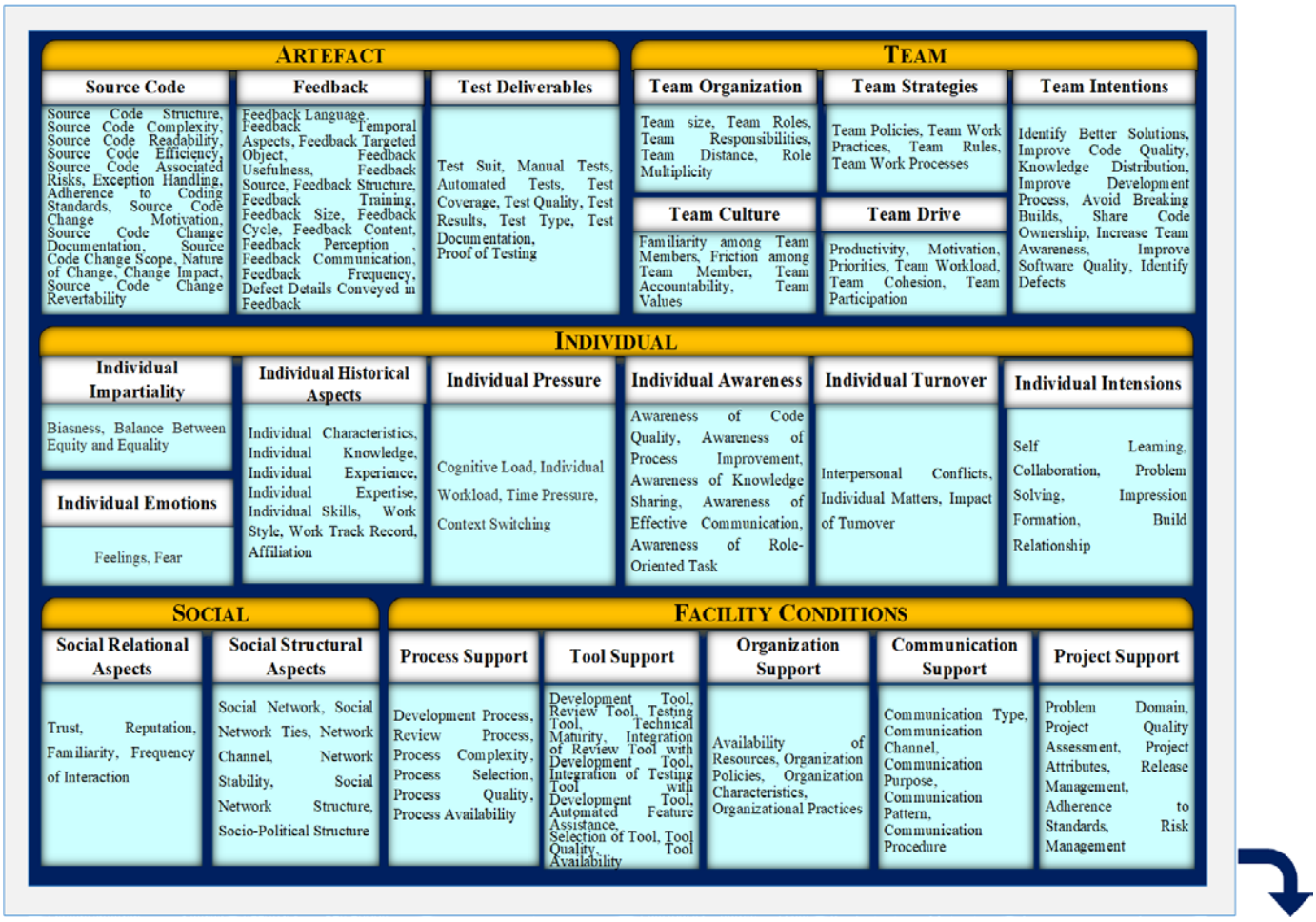
[40] S. G. Naoum, Dissertation research and writing for construction students, Second Edition, 2nd ed. Oxford: Butterworth-Heinemann, 2012.

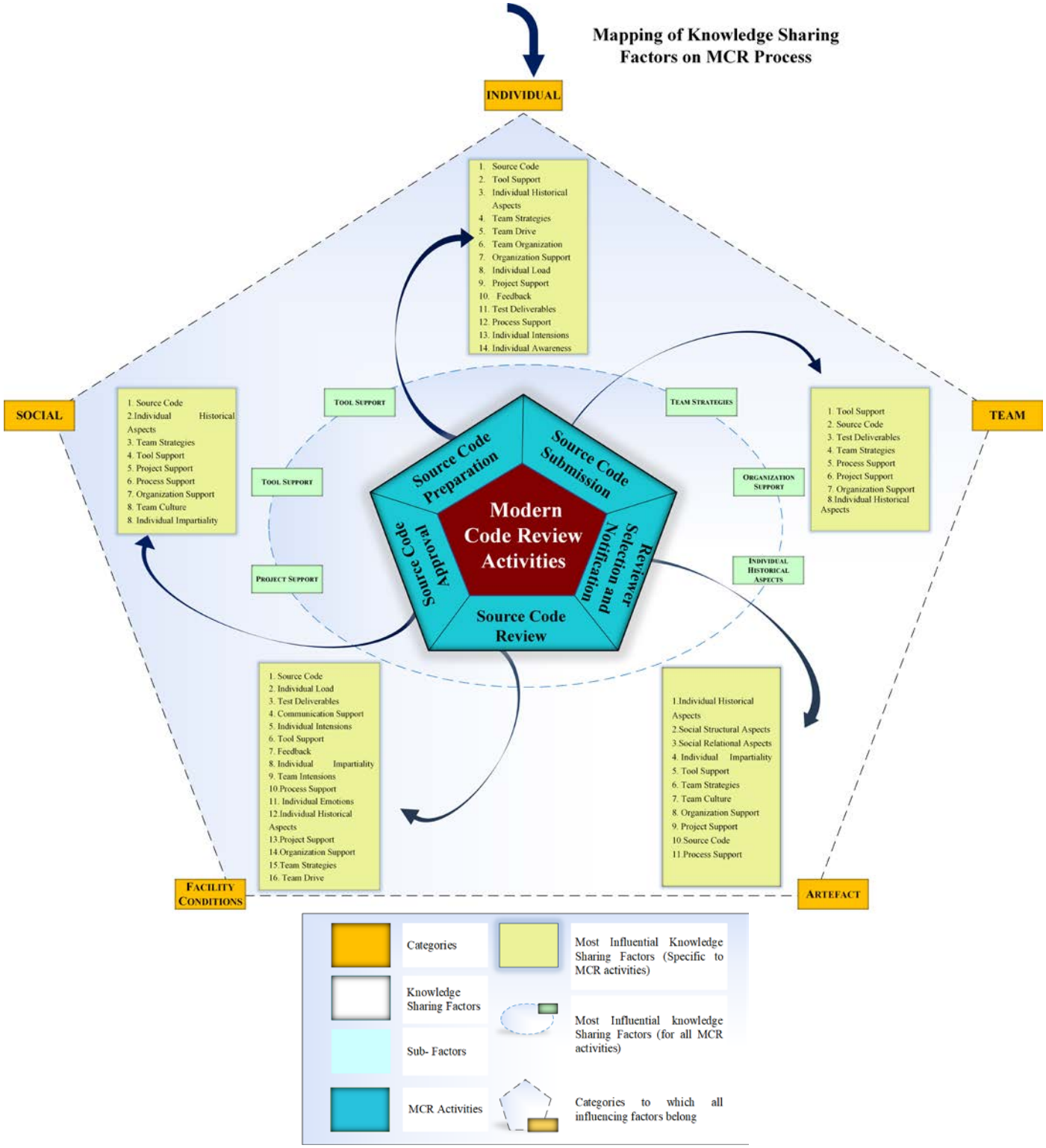
[41] J. Cohen, Statistical power analysis for the behavioral science, 2nd ed. New: Lawrence Erlbaum Associates, 1988.

[42] N. Fatima, S. Nazir, and S. Chuprat, "Understanding the Impact of Feedback on Knowledge Sharing in Modern Code Review," in 6th IEEE International Conference on Engineering Technologies and Applied Sciences (ICETAS) , In Press, 2019, pp. 1–5.

APPENDIX A

KNOWLEDGE SHARING FRAMEWORK FOR MODERN CODE REVIEW




Mapping of Knowledge Sharing Factors on MCR Process
INDIVIDUAL
1. Source Code
2. Tool Support
3. Individual Historical Aspects
4. Team Strategies
5. Team Drive
6. Team Organization
7. Organization Support
8. Individual Load
9. Project Support
10. Feedback
11. Test Deliverables
12. Process Support
13. Individual Intensions
14. Individual Awareness
SOCIAL
1. Source Code
2.Individual Historical Aspects
3. Team Strategies
4. Tool Support
5. Project Support
6. Process Support
7. Organization Support
8. Team Culture
8. Individual Impartiality
TEAM
1. Tool Support
2. Source Code
3. Test Deliverables
4. Team Strategies
5. Process Support
6. Project Support
7. Organization Support
8.Individual Historical Aspects
TOOL SUPPORT
TOOL SUPPORT
PROJECT SUPPORT
TEAM STRATEGIES
ORGANIZATION SUPPORT
INDIVIDUAL HISTORICAL ASPECTS
Source Code Preparation
Source Code Submission
Reviewer Selection and Notification
Source Code Review
Source Code Approval
Modern Code Review Activities
FACILITY CONDITIONS
1. Source Code
2. Individual Load
3. Test Deliverables
4. Communication Support
5. Individual Intensions
6. Tool Support
7. Feedback
8. Individual Impartiality
9. Team Intensions
10.Process Support
11. Individual Emotions
12.Individual Historical Aspects
13.Project Support
14.Organization Support
15.Team Strategies
16. Team Drive
ARTEFACT
1.Individual Historical Aspects
2.Social Structural Aspects
3.Social Relational Aspects
4. Individual Impartiality
5. Tool Support
6. Team Strategies
7. Team Culture
8. Organization Support
9. Project Support
10.Source Code
11.Process Support
Categories
Knowledge Sharing Factors
Sub- Factors
MCR Activities
Most Influential Knowledge Sharing Factors (Specific to MCR activities)
Most Influential knowledge Sharing Factors (for all MCR activities)
Categories to which all influencing factors belong