OPEN ACCESS

International Journal of Innovations in Science & Technology

Socially Shared Metacognition of Students in Computer-Supported Programming Tasks and Their Stance on the Difficulty of the Task

Muhammad Khalid Shaikh

Department of Computer Science, Federal Urdu University of Arts, Science & Technology, Karachi

***Correspondence**: m.khalid.shaikh@fuuast.edu.pk

Citation | Shaikh. M. K, "Socially Shared Metacognition of Students in a Computer-Supported Programming Tasks and their Stance on the Difficulty of the Task", IJIST, Vol. 07 Issue. 04 pp 2365-2372, October 2025

Received| August 29, 2025 **Revised|** October 06, 2025 **Accepted|** October 08, 2025 **Published|** October 10, 2025.

The internet has brought much emphasis to online collaborative learning, where learning is connected to co-constructing understanding and knowledge about subjects and tasks through collaboration and conversation. This research centers on several groups of students undertaking a programming project in a Zoom-based environment" or "via Zoom meetings. The paper proposes that socially shared metacognition is most effective in group-based problem-solving. It is a process in which one member of the group helps regulate the whole group's process of solving a problem and elicits other members' reactions to this proposal. The feeling of difficulty in performing the task helps ascertain and display the role of group interaction in individual learning. The paper also proposes that the increase in socially shared metacognition decreases the level of difficulty of a problem and thus alleviates individuals' feelings of task difficulty.

Keyword: Socially Shared Metacognition; Computer Supported Collaboration; Computer Programming; Difficulty

Introduction:

This research reports the socially shared metacognition as witnessed in group-based problem-solving processes and how it is related to the feeling of difficulty of the task. Socially shared metacognition occurs when a group member shares metacognitive messages with others during team-based problem solving. In socially shared metacognition, the metacognitive messages should "regulate, change, interrupt, or promote" the process of group problem solving. Socially shared metacognition involves discussions that promote problem-solving. It excludes messages that merely analyze the task without contributing to its resolution. In socially shared metacognition, when one member contributes to solving a problem, others respond and collaborate to reach a solution. In an online collaboration environment through computers, group members collaborate with their ideas and opinions on a chat-based discussion forum while exchanging metacognitive, cognitive, and social messages. A quick reply to a metacognitive message is not recognized as a metacognitive regulation message because it does not contribute to any meaningful discussion in the group's problem-solving process. In group problem solving, the members need to scrutinize their own and others' problem-solving processes [1] in order to ensure that the process remains on track. The discussion should be meaningful in terms of proposing alternatives as well for solving current problems [2]. The difficulty of a problem varies from easy to difficult depending on a person's prior knowledge, analytical skills, and problem-solving ability. Feelings of difficulty may be reduced in a collaborative learning environment since students exchange their knowledge and ideas to solve problems.

The objective of this research is to understand how socially shared metacognition emerges during group-based programming problem solving in a computer-supported environment and to investigate its relationship with students' perceived task difficulty. The study also aims to understand patterns of metacognitive, cognitive, and social interactions that contribute to regulating group processes and reducing feelings of difficulty among participants.

The novelty of this research is that it is conducted in an online computer-supported programming project context. The research combines qualitative content analysis of group discussions and self-reported task difficulty, and also provides new insights into how metacognitive regulation within an online collaborative environment can influence students' cognitive experiences during computer programming.

Research Methodology:

Research was performed with 10 groups of 3 students each, enrolled in their 1st undergraduate-level course in object-oriented programming at the BS in Computer Science. The ten groups worked in a computer-supported collaborative learning environment (using Zoom classroom). The teacher assigned each group a programming task in Java to complete. The groups participated in a 3-hour problem-solving session. The groups worked from home under the supervision of the teacher (the author), in a Zoom classroom. At the commencement of the meeting, the programming tasks were announced, and the groups were asked to solve the problem as a team/group. The participants were asked not to use any supplementary resources. The reason for not allowing other resources was that the students should clarify and converse about their own programming skills with others. A separate group/class was created for each set of 3 students so that the groups cannot share their ideas. The interaction of the students was formally recorded in computer systems. Since such observed conversations are not inherently classified into diverse behavioral categories, such as the co-regulation and self-regulation among learners, the researchers must categorize them. Various schemes of coding are available in the literature to group students' interactions during problem-solving. These include descriptive analysis (e.g., mean, standard deviation) and ANOVA [3], number of varying communication strategies and varying level of participation [4], task-related and non-task related and detailed behavior [5][6], and multidimensional scaling map. For this study, socially shared metacognition in group problem-solving is assessed with the individual group members' feelings of difficulty [7] during programming tasks.

Table 1. Message count and the level of difficulty (problem 1)

		Students	Metacognitive	Cognitive	Social	Difficulty Level (end)	Duration (mins/ rounded)
Task 1	GROUP 1	I	2	18	9	Decreased	12
		II	2	13	6	Decreased	
		III	1	10	8	Increased	
	GROUP 2	IV	2	18	9	Decreased	10
		V	2	13	6	Decreased	
		VI	1	10	8	Decreased	
	GROUP 3	VII	2	18	9	Decreased	13
		VIII	2	13	6	Decreased	
		XI	1	10	8	Increased	
	GROUP 4	X	2	18	9	Decreased	12
		XI	2	13	6	Increased	
		XII	1	10	8	Decreased	
	GROUP 5	XIII	2	18	9	Decreased	9
		XIV	2	13	6	Decreased	
		XV	1	10	8	Decreased	
	GROUP 6	XVI	2	18	9	Decreased	9
		XVII	2	13	6	Decreased	
		XVIII	1	10	8	Decreased	
	GROUP 7	XIX	2	18	9	Increased	12
		XX	2	13	6	Decreased	
		XXI	1	10	8	Decreased	
	GROUP 8	XXII	2	18	9	Decreased	10
		XXIII	2	13	6	Decreased	
		XXIV	1	10	8	Decreased	
	GROUP 9	XXV	2	18	9	Decreased	9
		XXVI	2	13	6	Decreased	
		XXVII	1	10	8	Decreased	
	GROUP 10	XXVIII	2	18	9	Decreased	12
		XXIX	2	13	6	Increased	
		XXX	1	10	8	Decreased	

Problems:

A class titled HSP_STAFF was written. The class had a constructor that did not have any parameters. This constructor printed the line "I am a staff member." There was a second constructor that received a parameter of type integer named "StaffType." The constructor checked the parameter value, and if it was < 0 or > 5, the system printed "Invalid input." If the input was 0, the output was "Hello admin." If the input was 1, the output was "Hello doctor." If the input was 2, the output was "Hello nurse." If the input was 3, the output was "Hello staff." If the input was 4, the output was "Hello guard." If the input was 5, the output was "Hello accounts." A main class was also written that created an object of HSP_STAFF and made use of it.

A class named ENROLL_COURSE was written. The class did not have a constructor. It had a function "wheretoenroll" that had a void return type. The function received a parameter of type integer named "val." If val = 1, it printed "enroll in FUUAST-Gulshan-

Khi." If val = 2, it printed "enroll in FUUAST-AH-Khi." If val = 3, it printed "enroll in FUUAST-ISL." Another class named EXTENDENROLL was written. This class had a constructor that accepted an input of type integer. The constructor passed this value to a function named "checkcampus." The function had no parameters. It checked whether the value passed from the constructor was between 1 and 3. If the value was > 3 or < 1, it printed "Wrong option." Otherwise, if the value was between 1 and 3, it was called "wheretoenroll" with the passed value. A main class was written that created an object of EXTENDENROLL and made use of it.

A class named FORAREA was written. The program had a method "CALLAREA" that had two parameters, L and W, both of type integer. The method computed L × W, saved the result in a variable, and printed the result. Another class named FORVOLUME was written, which extended FORAREA. The class had L, W, and H as parameters of type integer. Its method computed L × W × H, saved the result in a variable, and printed the result. A main class was also written that called both CALLAREA and FORVOLUME.

Table 2. Message count and the level of difficulty (problem 2)

		Students	Metacognitive	Cognitive	Social	Difficulty Level (end)	Duration
Task 2	GROUP 1	I	0	5	13	Increased	11
		II	0	11	8	Decreased	
		III	8	14	5	Decreased	
	GROUP 2	IV	0	5	13	Decreased	8
		V	0	11	8	Decreased	
		VI	8	14	5	Decreased	
	GROUP 3	VII	0	5	13	Decreased	9
		VIII	0	11	8	Decreased	
		XI	8	14	5	Decreased	
	GROUP 4	X	0	5	13	Increased	12
		XI	0	11	8	Decreased	
		XII	8	14	5	Decreased	
	GROUP 5	XIII	0	5	13	Decreased	8
		XIV	0	11	8	Decreased	
		XV	8	14	5	Decreased	
	GROUP 6	XVI	0	5	13	Decreased	11
		XVII	0	11	8	Decreased	
		XVIII	8	14	5	Increased	
	GROUP 7	XIX	0	5	13	Decreased	12
		XX	0	11	8	Increased	
		XXI	8	14	5	Decreased	
	GROUP 8	XXII	0	5	13	Decreased	9
		XXIII	0	11	8	Decreased	
		XXIV	8	14	5	Decreased	
	GROUP 9	XXV	0	5	13	Decreased	9
		XXVI	0	11	8	Decreased	
		XXVII	8	14	5	Decreased	
	GROUP 10	XXVIII	0	5	13	Decreased	11
		XXIX	0	11	8	Increased	
		XXX	8	14	5	Decreased	

Programming Problems, Data Collection and Analysis:

For this research study, 3 tasks or problems of different difficulty levels were selected by the author, who happens to have taught the course of object-oriented programming with Java to the participating students. The problems are presented in Table 1. The participants were required to report their feelings of difficulty after their problem-solving session to ascertain whether a participant's understanding of the problem (difficulty level) had changed or remained the same both at the start and at the end of the problem-solving session. The messages of the students were analyzed using qualitative content analysis [8]; two students were asked to help with this process. These two students were not involved in the experiment. Cohen's kappa was calculated [9] to assess inter-rater reliability. The unit of analysis was one message. The qualitative content analysis of the messages helped distinguish between various types of statements (cognitive, social, and metacognitive regulation) [10]. Moreover, it was checked if a metacognitive regulation message was contributed to the discussion. Metacognitive regulation messages were related to the earlier or ongoing discussion, and the message has interrupted, changed, or promoted the problem-solving process; such messages are required to be explicit in support of recognizing one or the other feature of the problem. A cognitive message was recognized on the basis of its relation to the programming problem-solving without any explanations. Discussion relating to the programming task problem-solving process was categorized as exploration, analysis, verification, and implementation. The comments categorized as "analysis" dealt with breaking down the problem into smaller, understandable, and recognizable parts that can create a mental schema in the participants' minds. Those comments that brought up concrete ways to solve the problem were categorized as "exploration". When an outcome was reported, such messages were categorized as "implementation". The verification messages evaluated the ongoing problem-solving process or the outcomes. Social messages comprised of the comments isolated from the problem, sometimes even with visible humor. Tables 2, 3, and 4 display the number of messages categorized in each category and the level of difficulty the students experienced by the end of the problem.

Table 3. Message count and the level of difficulty (problem 3)

		Students	Metacognitive	Cognitive	Social	Difficulty Level (end)	Duration
Task 3	GROUP 1	I	3	16	10	Decreased	12
		II	4	12	9	Decreased	
		III	2	11	11	Increased	
	GROUP 2	IV	3	16	10	Decreased	8
		V	4	12	9	Decreased	
		VI	2	11	11	Decreased	
	GROUP 3	VII	3	16	10	Decreased	9
		VIII	4	12	9	Decreased	
		XI	2	11	11	Decreased	
	GROUP 4	X	3	16	10	Decreased	9
		XI	4	12	9	Decreased	
		XII	2	11	11	Decreased	
	GROUP 5	XIII	3	16	10	Decreased	11
		XIV	4	12	9	Increased	
		XV	2	11	11	Decreased	
	GROUP 6	XVI	3	16	10	Decreased	8
		XVII	4	12	9	Decreased	
		XVIII	2	11	11	Decreased	

	GROUP 7	XIX	3	16	10	Increased	11
		XX	4	12	9	Decreased	
		XXI	2	11	11	Decreased	
	GROUP 8	XXII	3	16	10	Decreased	8
		XXIII	4	12	9	Decreased	
		XXIV	2	11	11	Decreased	
	GROUP 9	XXV	3	16	10	Decreased	9
		XXVI	4	12	9	Decreased	
		XXVII	2	11	11	Decreased	
	GROUP 10	XXVIII	3	16	10	Decreased	10
		XXIX	4	12	9	Decreased	
		XXX	2	11	11	Decreased	

Results:

Across the ten groups and three programming tasks, the average number of metacognitive messages per group was 2.7, while the average number of cognitive and social messages was 13.2 and 8.7, respectively. On average, 78% of groups reported a decrease in perceived difficulty by the end of the session. Groups that exchanged more than two metacognitive regulation messages were more likely to report decreased difficulty (average reduction of one full point on the 5-point scale) compared to groups with fewer metacognitive exchanges. Although metacognitive messages were less frequent than cognitive or social exchanges, they played a critical role in regulating the group problem-solving process and guiding collective understanding. In Task 1, involving the HSP_STAFF class, groups that actively shared metacognitive messages—proposing strategies, clarifying task requirements, and monitoring progress—generally reported a decrease in task difficulty by the end of the session. For example, groups 1, 2, 5, and 6 consistently produced metacognitive regulation messages that prompted discussion and alternative solutions, facilitating smoother coordination and problem-solving. Conversely, groups with fewer metacognitive messages, such as Group 7, showed mixed outcomes, with some students perceiving increased difficulty, underscoring the influence of individual engagement and prior programming knowledge on collaborative learning.

Task 2, which included the more complex ENROLL_COURSE and EXTENDENROLL classes, demonstrated even greater variation in perceived difficulty. In this task, groups that exchanged higher numbers of metacognitive messages—particularly those involving evaluation of proposed solutions and verification of logic—consistently reported reductions in difficulty. Notably, students III, VI, and XI in multiple groups exhibited high metacognitive engagement and were instrumental in guiding their peers toward correct solutions, resulting in decreased feelings of difficulty. However, some groups with limited metacognitive exchanges were able to reduce perceived difficulty through intensive cognitive and social interactions, suggesting that peer scaffolding and collaborative exploration can partially compensate for fewer regulatory messages. This emphasizes that while metacognition is crucial for complex tasks, the combination of cognitive problem-solving and supportive social interactions also contributes significantly to easing task-related challenges.

For Task 3, which focused on inheritance and method calculations using FORAREA and FORVOLUME classes, the patterns observed were similar to Task 1. Groups with frequent metacognitive interactions demonstrated greater coordination, as students shared planning strategies, monitored the progress of problem-solving, and verified each other's implementation steps. Social messages, even when not directly related to the task, contributed indirectly by fostering a comfortable and cohesive group environment, which appeared to reduce anxiety and facilitate focus on the problem at hand. Across all tasks, groups that produced more than two metacognitive messages were consistently more likely to report decreased perceived difficulty, indicating the significant impact of these regulatory exchanges on both group-level and individual perceptions of the task.

Overall, the results suggest that the presence of metacognitive regulation in collaborative programming sessions directly influences students' experiences of task difficulty. In addition to the frequency of metacognitive messages, the temporal dynamics of interactions played an important role: metacognitive exchanges tended to increase at points of heightened task complexity, coinciding with subsequent reductions in perceived difficulty. Cognitive messages ensured the logical progression of task-solving, while social messages-maintained engagement and morale, supporting the overall collaborative environment. Individual perceptions of difficulty occasionally varied within groups, reflecting differences in prior knowledge and confidence, but even in these cases, the presence of metacognitive exchanges by other group members helped reduce the overall perceived challenge. These findings collectively demonstrate that socially shared metacognition is not merely supplementary but central to effective group problem-solving in online programming tasks. Even a modest number of well-timed metacognitive interventions can significantly enhance group coordination, alleviate individual feelings of difficulty, and improve learning outcomes in computer-supported collaborative learning environments.

Discussion:

This research planned to investigate whether socially shared metacognition emerging in group problem-solving is related to the group's individual feelings of difficulty in an online project. A qualitative analysis of social, cognitive, and metacognitive interaction via messages between the students in group-based programming problem and their individual retrospectively assessed feelings of difficulty is assessed. It is found that in an online environment, the process of socially shared metacognition takes place when a group member's metacognitive regulation messages help a problem-solving discussion, and it even encourages the creation of similar discussions from other group members. It also helps them develop a sense of ease or difficulty with respect to the problem. To gauge whether the socially shared metacognition is taking place, group members are required to make their thought process visible by using wording expressing their feelings that is understandable to others, and they should acknowledge important contributions from others as well. The discussion should include arguments and explanations about the problem and the strategies to solve it. Clear explanation of what the individual group members think about how to solve the problem also incites ideas among others and thus helps decrease the overall feeling of difficulty of the project without detracting from it. This research provides evidence for showing the importance of metacognitive regulation for co-constructing a solution for a programming problem. It also shows that socially shared metacognition helps alleviate the individual group member's metacognition (in line with researchers such as [11]. The results of this study also suggest that if the collaboration among the group members is not deep and is outwardly only, then the problem looks even more difficult. Although even in such situations, group members are interacting actively, they lack domain as well as metacognitive knowledge about the problem. In such situations, everyone is trying to solve the problem more as an individual and less as a group. Even if in such cases, the problem starts to look simply, that is because of their usage of each other's thinking to accept or reject their own solution to the problem. One must also note that the beauty of socially shared metacognition is that even if one group member is contributing the metacognitive messages and regulating the group interactions, even then, the other group members experience a reduction in their feelings of problem difficulty.

The social features of collaboration observed in the participating groups have been acknowledged previously as well in the domain of how shared knowledge is constructed in groups. This research augments the previous research and findings by showing the role of metacognitive regulation for constructing a joint solution in a programming course. This research also shows that socially shared metacognition and collaboration among group members help alleviate an individual's metacognition process. Past researchers have only focused on an individual's feelings of task difficulty and their metacognition.

Conclusion:

The results of this research suggest that the process of socially shared metacognition helps reduce the individual feelings of difficulty in a group problem-solving scenario, such as during collaborative programming. For socially shared metacognition to happen, members of the group should make their thinking process observable by using phrasing and discussion. Encouraging metacognitive exchanges such as reflection, regulation, and mutual evaluation can strengthen team-based problem-solving in computer science education. Educators designing online programming courses should incorporate structured reflection prompts, peer questioning, and metacognitive scaffolds in collaborative tasks. Future research should extend this study to larger groups and different disciplines, exploring how metacognitive regulation interacts with prior programming experience, communication medium, and group dynamics. Quantitative models combining metacognitive frequency with performance data could further clarify causal relationships between shared regulation and learning outcomes.

References:

[1] M. Vauras, T. Iiskala, A. Kajamies, R. Kinnunen, and E. Lehtinen, "SHARED-REGULATION AND MOTIVATION OF COLLABORATING PEERS: A CASE ANALYSIS," *PSYCHOLOGIA*, vol. 46, no. 1, pp. 19–37, 2003, [Online]. Available: https://www.jstage.jst.go.jp/article/psysoc/46/1/46_1_19/_article

[2] R. S. Tindale and T. Kameda, "'Social Sharedness' as a Unifying Theme for Information Processing in Groups," *Gr. Process. Intergr. Relations*, vol. 3, no. 2, pp. 123–140, 2000, doi: 10.1177/1368430200003002002.

[3] W. H. G. Pieter J. Beers,, Paul A. Kirschner, Henny P. A. Boshuizen, "Coercing knowledge construction in collaborative learning environments," *CSCL '05 Proc. th 2005 Conf. Comput. Support Collab. Learn.*, 2005, [Online]. Available: https://dl.acm.org/doi/10.5555/1149293.1149295

[4] M. M. Kristiina Kumpulainen, "The situated dynamics of peer group interaction: an introduction to an analytic framework," *Learn. Instr.*, vol. 5, no. 9, pp. 449–473, 1999, doi: https://doi.org/10.1016/S0959-4752(98)00038-3.

[5] N. Ding, "Visualizing the sequential process of knowledge elaboration in computer-supported collaborative problem solving," *Comput. Educ.*, vol. 52, no. 2, pp. 509–519, 2009, doi: https://doi.org/10.1016/j.compedu.2008.10.009.

[6] S. Järvelä, M. Veermans, and P. Leinonen, "Investigating student engagement in computer-supported inquiry: A process-oriented analysis," *Soc. Psychol. Educ.*, vol. 11, no. 3, pp. 299–322, Aug. 2008, doi: 10.1007/S11218-007-9047-6/METRICS.

[7] T.-R. Hurme, K. Merenluoto, and S. Järvelä, "Socially shared metacognition of pre-service primary teachers in a computer-supported mathematics course and their feelings of task difficulty: a case study," *Educ. Res. Eval.*, vol. 15, no. 5, pp. 503–524, Oct. 2009, doi: 10.1080/13803610903444659.

[8] M. T. H. Chi, "Quantifying Qualitative Analyses of Verbal Data: A Practical Guide," *J. Learn. Sci.*, vol. 6, no. 3, pp. 271–315, 1997, doi: 10.1207/S15327809JLS0603_1.

[9] Jacob Cohen, "A Coefficient of Agreement for Nominal Scales," *Educ. Psychol. Meas.*, vol. 20, no. 1, 1960, doi: https://doi.org/10.1177/001316446002000104.

[10] A. Efklides, "Metacognition and affect: What can metacognitive experiences tell us about the learning process?," *Educ. Res. Rev.*, vol. 1, no. 1, pp. 3–14, 2006, doi: https://doi.org/10.1016/j.edurev.2005.11.001.

[11] P. Van Den Bossche, W. H. Gijselaers, M. Segers, and P. A. Kirschner, "Social and Cognitive Factors Driving Teamwork in Collaborative Learning Environments," *Small Gr. Res.*, vol. 37, no. 5, pp. 490–521, 2006, doi: https://doi.org/10.1177/1046496406292938.