Analyzing the Communication Patterns of Different Teammate Types in a Software Engineering Course Project

Yanye Luther
Colorado State University
Fort Collins, Colorado, United States
yanye.luther@colostate.edu

Lindsey Nielsen
Colorado State University
Fort Collins, Colorado, United States
lindsey.rebecca.nielsen@colostate.edu

Logan Cadman
Colorado State University
Fort Collins, Colorado, United States
allen.cadman@colostate.edu

Marcia Moraes
Colorado State University
Fort Collins, Colorado, United States
marcia.moraes@colostate.edu

Sudipto Ghosh
Colorado State University
Fort Collins, Colorado, United States
sudipto.ghosh@colostate.edu

Bianca Trinkenreich
Colorado State University
Fort Collins, Colorado, United States
bianca.trinkenreich@colostate.edu

Abstract

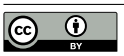
Effective communication is vital for the success of professional software engineering (SE) teams. As SE courses teach essential industry skills like teamwork and collaboration, ensuring effective communication becomes important in student projects. However, poor engagement from team members can lead to conflicts, uneven workloads, and diminished learning experiences. Teammate types such as Couch Potatoes, who contribute minimally, and Hitchhikers, who rely on others while taking credit, exacerbate these issues. In contrast, Lone Wolves work independently, potentially isolating themselves, while Good Teammates actively collaborate and contribute fairly, driving team success. In this study, we aimed to investigate the communication patterns of teammate types such as Couch Potato, Hitchhiker, Lone Wolf, or Good Teammate during a SE testing course. We applied Ordered Network Analysis (ONA) to the conversational data of the teams to examine the distinct communication patterns of students whose contributions were either perceived positively (e.g., Good Teammates) or negatively (e.g., Couch Potato, Hitchhiker, Lone Wolf) by their peers. The findings reveal distinct communication behaviors across teammate types. While Good Teammates and Couch Potatoes discussed similar content, Good Teammates communicated more frequently and consistently throughout the project. Lone Wolves seldom engaged in pleasantries, reflecting a task-focused approach, whereas Hitchhikers rarely contributed substantively to technical discussions, such as pull requests, often interacted through pleasantries. These patterns emphasize the need for early interventions and communication planning to promote accountability balance and effective student collaboration during class projects.

CCS Concepts

• **Software and its engineering**;

Keywords

Teamwork, Communication, Ordered Network Analysis



This work is licensed under a Creative Commons Attribution 4.0 International License.

FSE Companion '25, Trondheim, Norway

© 2025 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-1276-0/2025/06
https://doi.org/10.1145/3696630.3727232

ACM Reference Format:
Yanye Luther, Lindsey Nielsen, Logan Cadman, Marcia Moraes, Sudipto Ghosh, and Bianca Trinkenreich. 2025. Analyzing the Communication Patterns of Different Teammate Types in a Software Engineering Course Project. In *33rd ACM International Conference on the Foundations of Software Engineering (FSE Companion '25), June 23–28, 2025, Trondheim, Norway.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3696630.3727232

1 Introduction

Software engineering is fundamentally a socio-technical discipline [32] in which effective team communication plays a critical role in the success of software projects, contributing not only to academic achievement but also to professional development in the software engineering field [10, 24]. In educational settings and the software industry, collaboration and communication are fundamental to achieving successful outcomes in group-based course projects. The skills developed through team work during academic training directly translate into the collaborative nature of software development in real-world industry environments [24].

Working in teams provides students with the opportunity to collaborate with diverse individuals and take on various roles to complete a course project. As team members, they encounter a range of behaviors, some of which may be easier to work with than others. Even in teams where all members are collaborative, differing goals can create challenges—for instance, some students may strive for an "A" grade at all costs, while others may aim to do just enough to secure a passing "C" grade [25, 30].

Previous research on team dynamics [20] has identified several types of conflictive behaviors that can appear in teams: Couch Potatoes are individuals who contribute minimally to group efforts, often relying on others to carry the workload [19]. Lone Wolves, prefer to work independently, sometimes struggling to integrate with the team [3]. Hitchhikers are those who, like Couch Potatoes, contribute little but try to coerce the others into doing everything their way and actively seek to benefit from the group's efforts without significantly contributing [19]. Lastly, Good Teammates are described as students who work well with others, contribute to the work, and bring a positive attitude to the group [21].

Although prior research has examined conflictive behavior types in teams and their effects on group performance [3, 19, 21], to the best of our knowledge, no studies have yet investigated the distinct communication patterns—such as frequency, trends, and

context—associated with these behavior types within the context of software engineering education.

This study explored how SE students communicate during group-based course projects and whether their communication patterns align with their peers' perceptions of their behaviors. We present an empirical study exploring the following research questions:

- *RQ1*: How often does each type of teammate communicate?
- *RQ2*: What topics does each teammate type communicate during interactions?

We analyzed communication data from students collaborating on group-based class projects, focusing on their interactions within the group. In addition, we examined survey responses in which students in a SE testing class evaluated their teammates' contributions to the class project and assigned them a teammate type.

Our contributions to understanding these communication patterns can help educators identify struggling or disengaged students early and implement targeted interventions to foster more equitable collaboration. For instance, recognizing the limited or superficial contributions of Couch Potatoes and Hitchhikers can guide strategies to ensure accountability and fair workload distribution. Similarly, understanding how Good Teammates maintain productive communication or how Lone Wolves prefer independent interaction can inform practices to enhance team cohesion and optimize individual strengths. Ultimately, this analysis contributes to improving team-based learning experiences and better-preparing students for the collaborative demands of professional software engineering. Instructors can leverage tools like communication analysis and peer evaluations to identify struggling students, ensure fair workload distribution, and foster effective time management and collaboration skills.

2 Background

2.1 Course Context

The study was performed using data from the Spring 2024 semester of a senior-level computer science course focused on Software Testing at Colorado State University, which had 24 teams comprised of 118 students (89 in-person and 29 online). Colorado State University is classified as R1: Doctoral Universities – Very high research activity (Carnegie Classification System) [1].

The lectures focus on the fundamentals of systematic testing using various formalisms such as input space partitioning, graph coverage, mutation analysis, state-based testing, and testing workflows. The project provides hands-on training in test-driven development with various testing tools while using models to support various testing and implementation activities in a team of five students.

2.1.1 Course Project. In Sprint 1 (P1), students used input space partitioning to create tests, implement core business logic, and improve code coverage through graph-based coverage principles. In Sprint 2 (P2), they enhanced the P1 test suite's mutation score with Pitest, applied static analysis tools (PMD and SpotBugs), refactored code, and used automatic test generation tools (EvoSuite and Randoop). They also implemented a REST controller and mocked a database. In Sprint 3 (P3), students developed the front-end with JavaScript and React and wrote comprehensive test cases using Fitnesse and Selenium.

2.1.2 Team Creation. Students in this course were split into teams to work together on a software engineering project throughout the 16-week semester broken into three 5-week sprints. Teams consisted of about five students each.

Rather than randomly assigning students to teams, the instructor intentionally formed diverse and balanced groups. These teams were designed to reflect a mix of genders, attendance model (in-person or online), computer science skills (based on accumulated course credits), and prior experience (assessed through a questionnaire on internships, coursework, and familiarity with specific languages, tools, technologies, and processes). To give students some control over their team dynamics, each student was allowed to select one partner to be in the same group, enabling them to work with a known and trusted collaborator. Given the inclusion of online students—who often have different schedules and time zones compared to in-person students—teams were limited to no more than two remote participants. Furthermore, based on research showing that women tend to feel more motivated and confident when paired with peers of the same gender [38], teams were structured to include at least two women, ensuring that women had the opportunity for peer parity.

2.1.3 Teammate Types. In addition to the initial survey used to assess students' experiences with team formation, the instructor conducted three online surveys—one at the end of each sprint—with response rates of 100%, 98.3%, and 97.4% for the first, second, and third surveys, respectively. The questionnaire included items to evaluate how each team member perceived their teammates' contribution levels and to identify a perceived teammate type (Good Teammate, Lone Wolf, Couch Potato, or Hitchhiker).

The instructor provided an in-class explanation of teammate types to ensure clarity. Therefore, in a team of five students, each student was evaluated by the other four teammates. The teammate could be labeled with more than one teammate type. The Good Teammate label was assigned only for unanimous cases, i.e., when the entire group considered the teammate a Good Teammate. When at least one teammate considered the student a Couch Potato, Hitchhiker, or Lone Wolf, the student received that teammate type. This meant that a single student could receive multiple negative teammate type labels.

This approach was adopted to address potential biases in evaluations. It is challenging to consistently meet the criteria for being labeled a Good Teammate, and since the surveys were administered at the end of each sprint and were graded based on completion, there may have been instances where students did not fully engage with the survey process.

Additionally, the possibility of students offering lenient or favorable evaluations of their peers—either to avoid conflict or to protect teammates—was taken into consideration. Therefore, the decision to label a student as a "Negative Teammate Type" (Couch Potato, Hitchhiker, Lone Wolf) based on a single report from any one teammate helps mitigate the influence of potential bias and better reflects the diversity of perceptions within the team. We discuss the trade-offs of our approach in Section 3.6.

[1] https://carnegieclassifications.acenet.edu/

2.2 Epistemic and Ordered Network Analysis

Epistemic Network Analysis (ENA) is a network analysis technique, which can be used to examine co-occurrences of concepts (codes) in a given segment of discourse data. Those co-occurrences are considered a good indicator of cognitive connections, particularly when they are frequent [29] [18]. ENA represents the structure of connections among codes in discourse by calculating the co-occurrence of each unique pair of codes within a segment of data, and aggregates this information in a cumulative adjacency matrix. ENA represents this matrix as a vector in high-dimensional space, which is then normalized to quantify relative frequencies of co-occurrence independent of discourse length. In this process, ENA projects the networks as points into a low-dimensional space using singular value decomposition (SVD) [31]. This analytical tool hence provides network graphs, where the nodes in the model correspond to the codes in the discourse and the edges represent the relative strength of connection among codes [40].

Besides of identifying the co-occurrences of codes in qualitative data, Ordered Network Analysis (ONA) models the directional relationships between different contributions (codes), such as how one participant's message or action may influence the next [36]. These directional relationships refer to how an initial message (composed by codes) triggers a response (composed by codes), and how responses can then shape subsequent contributions, creating an ordered flow of communication. By modeling the directional relationships from initial messages to the subsequent responses, ONA can reveal many important ordered differences in individual's contributions to the collaborative process.

3 Methodology

In this section, we outline our methodology for addressing the research questions proposed in Section 1.

3.1 Dataset

We analyzed conversational data collected from students interacting about a course project.

Students were added to Microsoft Teams channels based on their team number by the instructor and Teaching Assistants (TAs). Only student messages were considered in the analysis. We retrieved 11,248 student messages from all 24 teams using the Microsoft Graph API. The data collected included the team number, student name, date and time of the message if the message started a new thread or was a reply as part of a message thread, the number and types of reactions the message received, subject of the message, and the content of the message. Only messages from designated team channels were retrieved. We did not have access to private Teams conversations between individual members or transcripts from Teams meetings and any other remote or in-person meetings. Therefore, the only indication of a team's conversation is within the designated channel created for the course.

The study was approved by the Institutional Review Board of the university. All data were deidentified so that each student was assigned a random number associated with their team, e.g., students in team 8 were S8.1, S8.2, S8.3. The replication package [16] has the dataset and the code to run the conversational data analysis.

3.2 Conversation Frequency

The data collected from the student messages included the date and time the messages were sent, allowing us to extract frequency information. We divided the messages sent during each sprint using the sprint end dates. The end date of each sprint was used as a reference point in Fig. 2 to visualize how students interacted with one another as the deadline approached. This helps highlight patterns or changes in communication behavior during critical periods of the project timeline. Next, we grouped the messages per teammate type. The frequencies were calculated for each sprint as students could have different teammate types assigned through the sprints. By applying these methods, we calculated the following metrics for each teammate type: the number of messages sent, the number of students, and the average number of messages sent.

3.3 Defining the Codebook

In order to extract potential codes and keywords from the data, we performed topic modeling using Latent Dirichlet Allocation (LDA) [4]. Table 1 shows the extracted stems of keywords for the initial codebook. The cleaning and pre-processing steps from Saravani et al. [26] were performed on all 11,248 messages from the 24 teams.

Table 1: Initial Codebook

Code	Keywords
0	worker, qualif, project, compani, method, isp, set, check, class, think
1	run, issu, tri, code, start, file, sure, like, branch, server
2	merg, look, need, fix, know, want, pr, issu, class, sorri
3	report, score, case, implement, test case, team, point, coverag, compon, reflect
4	meet, good, time, thank, team, think, sound, avail, today, like

The initial codebook identified several potential keywords. A subset of the authors of this paper then held four meetings to collaboratively revise the codebook, refining it until they reached a consensus through a negotiated agreement process [11]. For example, keywords *worker*, *qualif*, *project*, *compani* in code 0 were used for the new *project themes* code. Also, the *planning* code was inspired by keywords from code 4. The keyword *thank* in code 4 inspired the *pleasantries* code. After these revisions, the codebook was presented to the instructor for further refinement and verification. We determined that codes 1, 2, and 3 all discussed actions performed regarding the codebase in GitHub. We broke these into two distinct codes: *pull request*, which is the main unit of work and code performed by the students, and *github actions*, which describes the actions the other teammates can do in response to a pull request. The final codebook is shown in Table 2 based on the presented coding approach.

The finalized codebook was used to automatically code all 11,248 messages and to generate the well-formatted table [28] necessary to use ONA. This coded data was then represented in ONA networks using the ENA web tool [1] with the following configuration:

- Unit of analysis: teammate type
- Conversation: teammate type

Table 2: Finalized Codebook

Code	Description	Keywords	Example
pull request	Mention or discussion of a pull request (PR) in GitHub. In software engineering and specifically in this course, PRs are the main "unit of work" displayed by teammates in the form of reviewable code written by the teammate.	pr, pull request, branch, main	PR is out. This one contains a filter file that excludes, the DB file and DTOs files to help make SpotBugs more readable
github actions	Actions that teammates can perform in GitHub. Typically these are actions done by other teammates to look at another teammate's code in their PR	merge, approve, review, comment, commit	I left a comment on the PR. I'll merge it once that's addressed.
project themes	The project entailed creating an application that worked as a project management system, thus consisting of workers, qualifications, companies, and projects.	worker, qualification, company, project	I'll work on another issue for now until Project.addWorker() is done
planning	Talk about teammates getting together to meet and discuss the project.	meeting, meet, in person, remote, on a call, on the call, available, schedule, teams call, csb, lab, when2meet	How about we shoot for a 4:30 PM meeting on Friday, then?
pleasantries	An inconsequential remark made as part of a polite conversation.	thanks, thank you, thx, nice work, good work, great work, nice job, great job, good job, great idea, good idea, well done, awesome, good luck, appreciate, lgtm, kudos, please, pls, plz, yay	Great, thank you!

- Codes: *pull request*, *github actions*, *project themes*, *planning*, and *pleasantries* from Table 2
- Comparison group: teammate type
- Network type: Ordered

3.4 ONA Analytical Procedures

ONA is built on a similar mathematical framework used in ENA. The ONA algorithm accumulates the connections for each unit of analysis using the coded data. Those connections are identified using a moving window formed from a current line of data to the preceding lines within the window. In this study, the unit of analysis is the teammate type, and the moving window size is five lines. We chose a five-window size because we identified that in our conversation data, the current line is the response to the preceding four lines of conversation, stating the common ground where the connections are happening. For example, as shown in Table 3, all five messages are within the window size, where S4.2's final response *"Good idea. I'll bring it up next meeting"* is the response to the four messages preceding it. The coding process is binary, where a value of one indicates the presence of a code, and a value of zero signifies its absence.

The connections are accumulated into an asymmetric adjacency matrix where the number of connections from code A to code B may differ from the number of connections from B to A. This allows ONA to provide the directionality of the connections to codes, unlike the uni-directionality modeled in ENA [35]. ONA transforms this single matrix into a single high dimensional symmetric adjacency vector, which is then normalized and centered. The algorithm then performs a dimensional reduction using singular value decomposition (SVD) and means rotation (MR), which allows these high-dimensional vectors and matrices to be represented in a 2D space [35].

In ONA, the network nodes are placed in the space using the same optimization method used in ENA [5]. This algorithm minimizes the distance between the ONA scores and the centroids of the corresponding networks. As a result, the layout of the ONA metric space can be interpreted based on the positions of the nodes. Units with ONA points on the right side of the space tend to have more frequent connections between the codes located on the right. Similarly, units with points on the left side of the space have more frequent connections between the codes on the left.

3.5 ONA Visualization Design

3.5.1 Nodes. In ONA, the size of each node corresponds to the number of times a particular code appears as a response to other codes within the data, with larger nodes indicating more frequent responses. In Figure 1, node B is the least common response compared to A and C. The color and saturation of the circle inside each node reflect the number of self-connections for that code, which occurs when a code appears both as a response and in the ground (source of a connection) of a given window. Larger and more saturated colored circles represent codes with more frequent self-connections. In Figure 1, node A has the most self-connections than nodes B and C.

3.5.2 Edges. Directed connections in ONA are represented by edges between nodes, visualized as a pair of triangles [35]. ONA employs a "broadcast" model so the source of a connection (ground)

Table 3: Example Coded Student Discourse

Student	Message	pull request	github actions	project themes	planning	pleasantries
S4.2	*I made a PR for verifying a work has x qualifications*	1	0	1	0	0
S4.3	*I left some comments on it for you to address*	0	1	0	0	0
S4.2	*I'll work on addressing those. Thank you for reviewing that!*	0	1	0	0	1
S4.1	*I saw those comments too. I think we should all talk about it in our next meeting on Monday.*	0	1	0	1	0
S4.2	*Good idea. I'll bring it up next meeting*	0	0	0	1	1

is placed at the apex of the triangle, and the destination (response) is at the base. In other words, the directionality is read as the thinner part of the edge to the thicker end of the edge of the triangle. To make directionality clear, the dark chevrons placed inside the triangles indicate the directionality of the connection from ground to response [35]. Between any two codes there may be a bidirectional connection where the chevron only appears on the side with stronger connections. In Figure 1, node A to node C has a thicker triangle with a chevron on meaning that code A is in the common ground that code C is a response to. The dark chevron pointing towards C from A helps viewers identify that C is more often a response to A than vice versa.

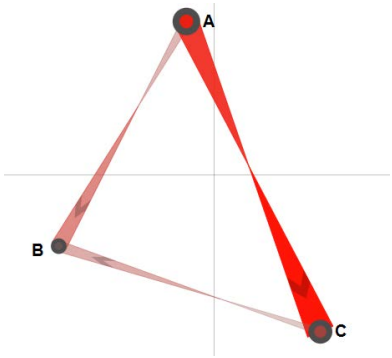


Figure 1: Sample ONA Network

3.6 Trade-Offs

We followed Robillard et al. [22] guidelines for reporting trade-offs that promotes a fair and dispassionate assessment of researchers' work.

The first decision point involved how to classify teammate types based on peer evaluations. The labeling approach required unanimous agreement to assign the Good Teammate label, while a single report was sufficient to label someone as a Couch Potato, Hitchhiker, or Lone Wolf. This asymmetry aimed to reduce bias and encourage accountability. While the unanimous requirement ensured that only universally recognized contributors were labeled as Good Teammates, it also introduced a trade-off—potentially excluding individuals seen as strong contributors by most, but not all, teammates. Similarly, assigning a negative teammate type based on a single teammate's feedback aimed to counteract leniency and encourage honest reporting. This approach valued diverse perspectives but risked overemphasizing outlier opinions, potentially misclassifying students not consistently exhibiting negative behavior. Alternatives like majority voting or weighting feedback by role or task proximity could offer a different balance between fairness and inclusivity. However, they might dilute minority insights or complicate interpretation. The chosen approach prioritized capturing diverse perspectives, recognizing that all methods have trade-offs.

The second decision point stems from refining the codebook. While automated methods reveal broad patterns, they often miss nuanced details like idioms or implicit meanings. Manual refinement helps align codes with research goals focused on actionable insights for educators. A trade-off in refining the codebook lies in balancing interpretability with technical specificity—for example, separating *pull-request* and *GitHub actions* to distinguish between code creation and review, which automation had merged. An alternative strategy could have included technical testing terms like *mutation score*, *test coverage*, or *static analysis* to highlight domain depth. However, this suits studies focused on technical performance more than team communication. Mixing technical and behavioral codes might also reduce clarity by creating competing focal points.

4 Results

4.1 RQ1: How often does each type of teammate communicate?

Table 4 presents the frequency of messages per sprint sent by each teammate type: Couch Potatoes, Hitchhikers, Lone Wolves, and Good Teammates. For each sprint, the table shows the total number of messages and the average number of messages sent by each teammate type.

Good Teammates consistently demonstrated the highest level of communication across all sprints, totaling 7,916 messages with an average of 36.99 messages per student, the higher average messages per sprint than the other teammate types. Out of the 11,248 messages sent throughout the semester, Good Teammates sent 70.3% of the total messages, Couch Potatoes sent 11.7%, Lone Wolves sent 9.6%, and Hitchhikers sent 8.4%.

In Figure 2, the dashed vertical lines indicate the end of each sprint, marking the deadlines when the students submitted their

Table 4: Message Frequency of Different Teammate Types

	Couch Potatoes			Hitchhikers			Lone Wolves			Good Teammates		
	# of Students	Total # of Messages	Average # of Messages	# of Students	Total # of Messages	Average # of Messages	# of Students	Total # of Messages	Average # of Messages	# of Students	Total # of Messages	Average # of Messages
Sprint 1	27	496	18.37	20	356	17.80	15	780	52.00	72	2821	39.18
Sprint 2	31	511	16.48	17	377	22.18	14	211	15.07	68	2434	35.79
Sprint 3	22	309	14.05	13	207	15.92	9	95	9.44	74	2661	35.96
Totals	80	1316	16.45	50	940	18.80	38	1076	28.32	214	7916	36.99

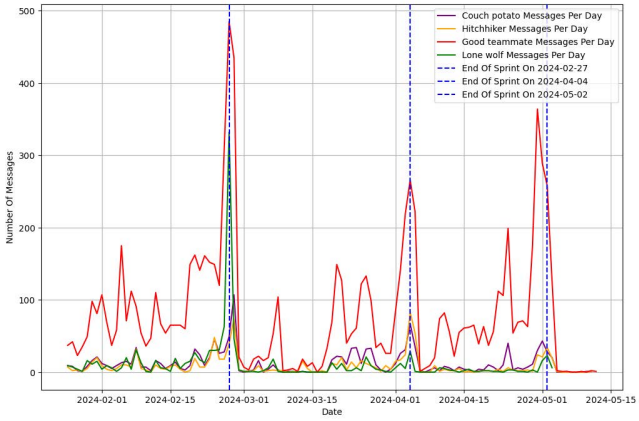


Figure 2: Messages Per Day for All Teammate Types

projects for review. We observed notable spikes in communication activity toward the end of each sprint. This pattern indicates that students tend to engage in more frequent and intensive interactions as project deliverable deadlines approach, where students are likely finalizing and submitting their work. Such behavior suggests that deadlines served as a catalyst for collaboration and problem-solving, likely driven by the urgency to finalize tasks and address any remaining issues.

RQ1

The analysis revealed that Good Teammates consistently demonstrated the highest level of communication throughout the semester. Communication activity spiked significantly toward the end of each sprint, coinciding with project submission deadlines, indicating that students engage more intensively in collaboration and problem-solving as deadlines approach.

4.2 RQ2: What topics does each teammate type communicate during interactions?

To qualitatively represent the content of the messages sent by each of the teammate types, we used Ordered Network Analysis. Figures 3 show the individual network of co-occurrences of the codes *pull request*, *github actions*, *project themes*, *planning*, and *pleasantries* within a 5 stanza moving window for each of the teammate types. Meaning these are the co-occurrences within 5 student messages sent. The unit of analysis is the teammate types to isolate the differences in the teammate types in each network. For example Figure 3d shows the connections all Good Teammates made over the entire three sprints of the course.

One way to compare individual networks is to compare the centroid of the network. The small colored squares on each network represent the centroid of the individual network. The centroid allows a network to be reduced to a single point for comparison, thus capturing all of the weighted connections between codes. The color saturation of the circle within each node are proportional to the number of self-connections for that code. In Figure 3, the *pull request* code has the most self-connections compared to all the other codes for all teammate types. Meaning after a student discusses a pull request, the next message is often also about pull requests.

4.2.1 Couch Potato Individual Network. In Figure 3a, the Couch Potato centroid indicates they discuss less *planning* and *pull request* than Good Teammates and Lone Wolves, but still more than Hitchhikers. Their centroid was also the closest to *project themes* compared to the other teammate type centroids, indicating Coach Potatoes discussed *project themes* the most. Couch Potatoes more often discussed *planning* and then *pull requests* as the next topic of discussion, and were the only teammate type to have *pull request* be the response of *planning* since all three other teammate types have the chevron pointing from *pull request* to *planning*. They engaged in lots of project-related discussions as indicated by the thick edge between *pull request* and *project themes*. However, they had many thin edges to *planning* meaning Coach Potatoes did not discuss much planning or agreeing to attend team meetings. They had

a strong edge from *pull request* to *pleasantries*, indicating saying "thank you" or "good job" in response to a pull request.

4.2.2 Hitchhiker Individual Network. In Figure 3b, the Hitchhiker centroid is the furthest away from the *pull request* code, meaning they typically did not discuss pull requests. Hitchhikers also had the weakest connection between *pull request* and *project themes* compared to the other teammate types. Hitchhikers had the thickest edges to and from *pleasantries* compared to all the other teammate types, had lots of thicker edges going towards *planning*, and had the most saturated circle within the *pleasantries* node meaning they usually mentioned another pleasantry in response to a different pleasantry. Despite not engaging in many project-related discussions, Hitchhikers discussed planning and agreeing to attend meetings and used lots of pleasantries.

4.2.3 Lone Wolf Individual Network. In Figure 3c, the Lone Wolf centroid is near *pull request* as well as *planning*. This means they discussed more of these codes compared to the other codes. Lone Wolves had the weakest connections to and from *pleasantries*. Besides not using *pleasantries*, and had good connections to all of the other codes.

4.2.4 Good Teammate Individual Network. In Figure 3d, the Good Teammate centroid is close to *pull request* code, meaning a majority of the discussions are about pull requests. Similar to Lone Wolves, Good Teammates also do not use many *pleasantries*.

4.2.5 Differences with a Good Teammate. To better visualize the differences between the teammate type individual networks, Figure 4 shows the subtracted networks for the negative teammate type options compared to the Good Teammate. A subtracted network is able to show the most salient differences between two networks by calculating the difference in connection strengths between corresponding nodes.

Figure 4a shows there are not very many differences in the content discussed by Good Teammates and Couch Potatoes as seen by the thin edges between all of the nodes. Couch Potatoes used more *pleasantries* than Good Teammates. While that, in Figure 4c), we show that Good Teammates discuss more *project themes* with *pull request* compared to Lone Wolves. Lone Wolves have more instances of responding to *pull request* with *planning* and *github actions*. Good Teammates also use more *pleasantries* than Lone Wolves. The biggest difference from a Good Teammate was that of Hitchhikers. Figure 4b shows that Hitchhikers discuss a lot more *planning* with *pleasantries* and *github actions* with *pleasantries* compared to Good Teammates that discuss more *pull request* with *project themes* and *pull request* with *planning*. In all subtracted networks, Good Teammates discuss *pull requests* and *project themes* more than all the other teammate types.

We discuss our findings in the context of existing literature in Sec. 5, drawing connections between our results and prior research.

RQ2

Good Teammates consistently discussed *pull requests* and *project themes*, which are related to technical progress. In contrast, Hitchhikers rarely engage with *pull requests* and instead start conversations with *planning* topics, such as meetings, often followed by *pleasantries*. They also discuss *GitHub actions*, typically tasks handled by others, but these conversations are interspersed with *pleasantries* rather than substantive technical contributions. Couch Potatoes, while using more *pleasantries* than Good Teammates, still engage in discussions about *pull requests*, unlike Hitchhikers. Lone Wolves stand out with the weakest connections to and from *pleasantries* among all teammate types.

5 Discussion

Good Teammates consistently demonstrated the highest levels of communication, both in terms of frequency and the relevance of content. Aligned with research that shows a great teammate focuses on achieving goals and consistently communicates with the team [9, 14], our results show that Good Teammates' messages were focused on project-related tasks, such as pull requests, project themes, and planning, reflecting a high level of engagement and collaboration. These students were not only frequent communicators but also consistently discussed important aspects of the project.

Hitchhikers exhibited a very different communication pattern from Good Teammates. While they sent fewer messages overall, the content of their communication was noticeably less focused on the project or technical tasks. Instead, Hitchhikers tended to engage in more social or low-effort discussions, such as pleasantries or general planning, rather than delving into the actual work or code-related issues. This aligns with the Hitchhiker manipulative behavior [19], and suggests that Hitchhikers may be employing a strategy to stay minimally involved in the work while still appearing somewhat engaged, particularly when deadlines approach. Their behavior represents a clear attempt to "game the system" by contributing as little as possible while avoiding being perceived as completely absent. This lack of focus on the project's core tasks is a stark contrast to the behavior of Good Teammates. Hitchhikers need to have their limits set early and high, because they have an uncanny ability to detect just how much they can get away with [19].

Couch Potatoes exhibited a communication pattern similar to Good Teammates in terms of content but used significantly more *pleasantries*. This suggests that Couch Potatoes frequently made inconsequential remarks, such as thanking or praising their teammates. While *pleasantries* play a vital role in team culture by boosting morale and fostering a sense of belonging within the team [39], they contribute less directly to project tasks. The key distinction between Couch Potatoes and Good Teammates lies in the frequency of their messages. While Good Teammates communicated regularly and consistently throughout the sprints, Couch Potatoes were far less active overall. This reduced frequency of communication significantly limited their ability to contribute at the same level as Good Teammates, despite having similar discussions about project tasks when they did engage. Couch Potatoes seem capable of contributing the right content, but their infrequent participation prevented them from fully playing an active role within their teams. The literature suggests that Couch Potatoes benefit from clear, firm expectations

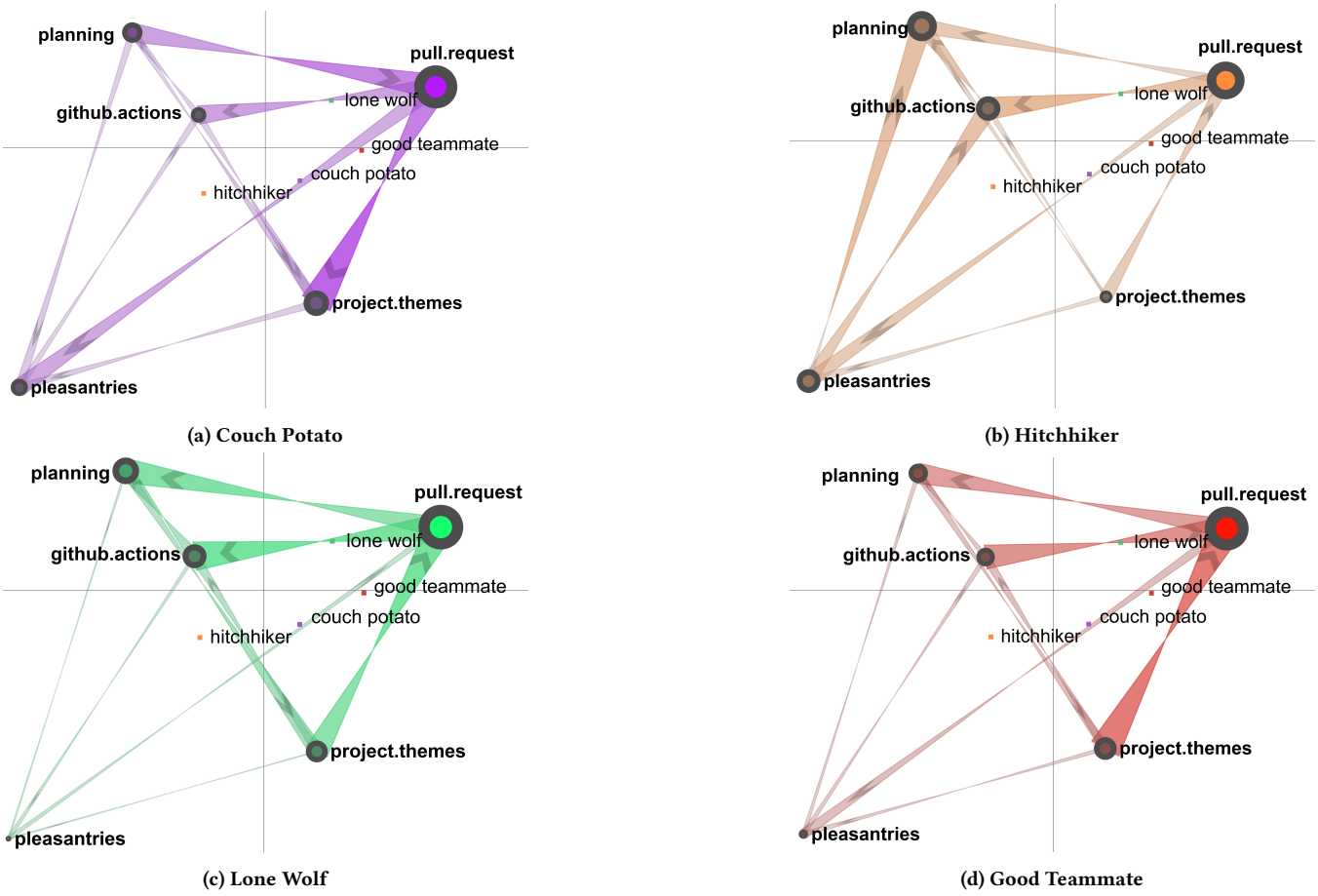


Figure 3: Individual Networks for Teammate Types

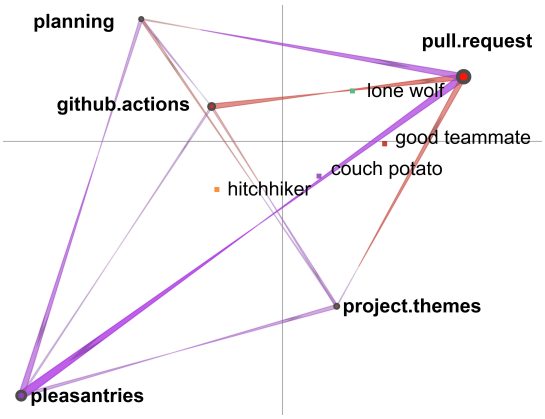
and can often be effectively guided to produce reasonably good work when provided with structured direction [19].

Aligned to research on community smells that shows that Lone Wolves carry out their work with little communication [6, 7, 34], one notable difference between Lone Wolves and other teammate types in our study is that Lone Wolves had the weakest connections to and from *pleasantries* compared to all other teammate types. Our results showed that Lone Wolves had more instances of responding to *pull request* with *planning* and *github actions*, for example, responding to a *pull request* being made with either reviewing the pull request or suggesting to meet and discuss the pull request. This could suggest that they were less likely to engage in social interactions, such as thanking or praising teammates, which might indicate a more individualistic or task-focused approach to communication. Despite this, their focus on project tasks shows they were still somewhat engaged, but their lack of social connection could limit their overall teamwork effectiveness. The literature suggests that Lone Wolves benefit from targeted mentoring and the implementation of a structured communication plan. These strategies encourage them to engage more frequently and effectively with their peers, fostering better collaboration and integration within the team [7].

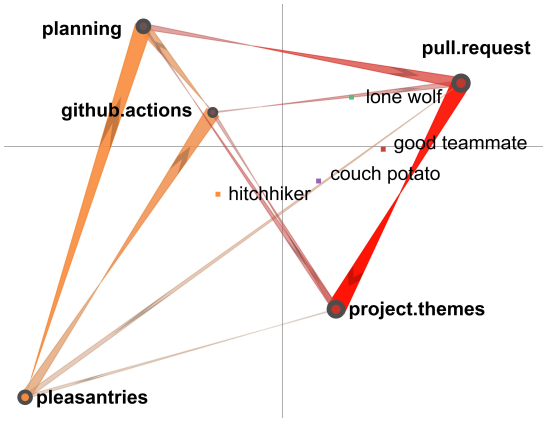
Our results align with the findings of previous studies on team communication patterns that emphasized the importance of frequent and relevant communication for team performance [17].

6 Related Works

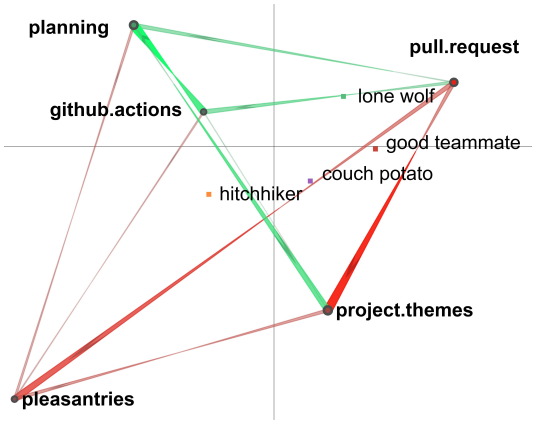
Previous studies in other domains have explored how various teammate types communicate and their impact on team dynamics. In medicine [19], Couch Potatoes were shown to burden their teammates by minimally contributing, forcing others to compensate for their lack of effort. Hitchhikers were shown as exploiting the team by avoiding meaningful contributions while still taking credit for the work, which creates resentment and undermines fairness. Lone Wolves, although capable, isolated themselves from the team, reducing collaboration and cohesion. Marketing education [3] also discussed the negative impact on student team performance when having Lone Wolves in group-based projects. Characterized by their preference for working independently and lacking trust in teammates' abilities, Lone Wolves were shown as impatient with group processes, thus hindering effective collaboration. We contribute to the state-of-the-art by analyzing the communication patterns of different teammate types in a software engineering group-based class project.



(a) Couch Potato (purple) vs Good Teammate (red)



(b) Hitchhiker (orange) vs Good Teammate (red)



(c) Lone Wolf (green) vs Good Teammate (red)

Figure 4: Subtracted Networks of Good Teammate Compared to Other Types. Different colors represent each teammate type.

In computer science, Chen et al. [8] investigated the impact of group communication patterns on students' performance. The findings showed that effective communication correlated with higher individual grades. Different than our study, which analyzed the perceived contributions of individual members as teammate types, the study analyzed teams by communication patterns of their members: "ideal" (balanced communication with active contributions from all members), "dominant leader" (a central leader communicating extensively with others), "unsocial" (minimal interaction), and "unresponsive" (irregular communication). Groups with "ideal" or "dominant leader" patterns performed significantly better, while those with "unsocial" or "unresponsive" patterns had lower grades. Besides the impact on grades, [2] investigated how communication relates to the software development lifecycle and found that developers communicate more when more bugs are present.

ENA and ONA have been used as analysis techniques in education and learning. ENA has been used to understand the social presence of a student [23], the role the student plays in the group [13, 33], and sentiment of the students' messages [27] in educational group settings. ONA has been used to model the performance of military teams learning to identify, assess, and respond to potential threats detected by radar [35]. It is also being used to understand learning tactics applied by students in an Massive Open Online Courses (MOOC) [36], as well as to unveil patterns of success and struggles in students programming assignments in a CS1 course[37]. In this paper, we used ONA to the conversational data of the teams to examine the communication patterns of students. We applied the automated methods used in [26] and [15] to extract specific content and terminology students use in chat messages exchanged during the group-based course project.

7 Implications and Future Work

The findings of this study provide actionable insights for instructors to enhance team dynamics and learning outcomes in group-based projects. Instructors can use communication analysis to identify Couch Potatoes, Hitchhikers, and Lone Wolves early, enabling targeted interventions such as mentoring and structured support to foster engagement.

Establishing communication norms and plans at the start of projects can help set expectations for frequency and type of communication. Assigning and rotating distinct roles within teams can help to ensure active participation and expose students to various aspects of teamwork, while periodic check-ins help address

communication gaps and conflicts. Encouraging task-oriented communication, focusing on topics as pull requests and planning, can be facilitated through templates or training.

Future research could delve deeper into the team climate and the reasons behind teammate evaluations, offering more nuanced insights into behaviors and perceptions of contributions. Defensive climates, characterized by judgmental communication, controlling behaviors, and a lack of empathy, can stifle open interaction and hinder collaboration within teams [12].

Analyzing conversational data through the lens of defensive communication climates could provide valuable insights into whether specific teammate types are more likely to encounter or contribute to such climates. Additionally, delving into the justifications students provide when evaluating and classifying their peers into these teammate types could uncover the underlying dynamics driving these perceptions. This approach would not only clarify how defensive climates influence team interactions but also inform strategies to foster more supportive and inclusive communication practices.

Additionally, future studies could examine how various teammate types contribute to the project by analyzing course project data on GitHub. This analysis could compare perceived contributions with actual contributions, shedding light on potential discrepancies and providing a more comprehensive understanding of team dynamics.

8 Threats to Validity

Construct validity: The categorization of teammate types (e.g., Couch Potatoes, Hitchhikers, Lone Wolves, and Good Teammates) is based on peer evaluations and perceptions. However, these classifications may be influenced by peer biases or misunderstandings of teammate behavior, potentially affecting the validity of the constructs. However, our results identified similarities between what the literature says about the behavior of each teammate type and the conversational data. For example, our results (see Sec 4.2) showed that Hitchhikers do not tend to talk about pull requests, and avoiding working on deliverables is a behavioral characteristic of a Hitchhiker [19]. Lone wolves missing *pleasantries* also aligns with the behavioral characteristic of social isolation of lone wolves [7]. Another threat is that this study leverages metrics such as the frequency and topics of messages to analyze communication patterns. While these metrics provide useful insights, they may not fully capture communication effectiveness, which could be a more critical determinant of team performance. The study assumes frequent discussions about pull requests, project planning, and task updates indicate effective collaboration. However, effective teamwork can also occur with minimal written communication, particularly when tasks are well-distributed and independently managed.

Internal validity: Being based on peer evaluations, the classification of teammate types may be influenced by interpersonal relationships, personal biases, or a lack of comprehensive observation. Students may inaccurately label peers as Couch Potatoes, Hitchhikers, or Lone Wolves based on subjective impressions rather than objective behaviors. We mitigated this bias by assigning Good Teammate only for unanimous cases, i.e., when the entire group considered the teammate a Good Teammate. As explained for construct validity, the study itself mitigates this threat by showing communication patterns associated with teammate types. However, confounding variables can play a role. Other factors, such as prior teamwork experience, technical expertise, or time management skills, might influence communication patterns and team contributions. These variables are not controlled for in the analysis, which could confound the relationship between teammate types and communication behavior. Moreover, the reliance solely on Microsoft Teams chat data may not provide a complete picture of communication. Teams could use other communication tools (e.g., email, messaging apps) or have verbal interactions that are not captured, leading to gaps in the analysis. To mitigate these internal threats, future research could triangulate data from multiple sources (e.g., additional communication tools, project management logs), incorporate more objective measures of team contributions, and explore longitudinal designs to capture temporal effects and confounding variables more accurately.

External validity: Our findings are based on data from a single course at one institution, the results may not apply to other educational contexts, courses, or institutions with different curricula or teaching methodologies. The characteristics of the cohort, such as skill levels, prior experience, and team diversity, are unique and may not represent broader populations of software engineering students. Furthermore, external factors, such as concurrent coursework or other situational stressors, might have impacted communication patterns. Although the findings from this case study should not be generalized, they suggest that communication patterns can offer valuable insights into teammate behavior. Nonetheless, further research is needed to validate the teammate type constructs and to explore whether perceived teammate types align with actual contributions.

9 Conclusion

The results of this study demonstrate that the perceptions of teammate types as Good Teammates, Couch Potatoes, Hitchhikers, and Lone Wolves are related to communication behaviors. Students who their peers classified as Good Teammates were the most consistently active communicators, engaging in task-oriented discussions related to project deliverables. In contrast, students who were classified as Couch Potatoes or Hitchhikers had lower levels of communication, and their conversation topics were less focused on project tasks. Lone Wolves did not engage in collaborative communication.

These findings underscore the importance of consistent, task-oriented communication for successful collaboration in team-based projects. They also suggest that teammate types can help to understand team dynamics and identify students who may need additional support or intervention to increase their engagement and contributions.

Educators can use these insights to foster better communication within teams, ensuring that all members contribute to discussions and work collaboratively toward shared goals. In particular, students identified as less engaged—such as Couch Potatoes and Hitchhikers—may benefit from targeted strategies that encourage more active participation, thereby improving both their individual learning outcomes and the overall success of the team.

References

[1] 2025. ENA Web Tool. https://app.epistemicnetwork.org/login.html

[2] Roberto Abreu and Rahul Premraj. 2009. How developer communication frequency relates to bug introducing changes. In *Proceedings of the Joint International and Annual ERCIM Workshops on Principles of Software Evolution (IWPSE) and Software Evolution (Evol) Workshops* (Amsterdam, The Netherlands) *(IWPSE-Evol '09)*. Association for Computing Machinery, New York, NY, USA, 153–158. doi:10.1145/1595808.1595835

[3] Terri Feldman Barr, Andrea L. Dixon, and Jule B. Gassenheimer. 2005. Exploring the "Lone Wolf" Phenomenon in Student Teams. *Journal of Marketing Education* 27, 1 (2005), 81–90. doi:10.1177/0273475304273459 arXiv:https://doi.org/10.1177/0273475304273459

[4] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[5] Dale Bowman, Zachari Swiecki, Zhiqiang Cai, Yeyu Wang, Brendan Eagan, Jeff Linderoth, and David Williamson Shaffer. 2021. The Mathematical Foundations of Epistemic Network Analysis. In *Advances in Quantitative Ethnography*, Andrew R. Ruis and Seung B. Lee (Eds.). Springer International Publishing, Cham, 91–105.

[6] Gemma Catolino, Fabio Palomba, Damian A Tamburri, Alexander Serebrenik, and Filomena Ferrucci. 2019. Gender diversity and women in software teams: How do they affect community smells?. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*. IEEE, 11–20.

[7] Gemma Catolino, Fabio Palomba, Damian A Tamburri, Alexander Serebrenik, and Filomena Ferrucci. 2020. Refactoring community smells in the wild: the practitioner's field manual. In *Proceedings of the acm/ieee 42nd international conference on software engineering: Software engineering in society*. 25–34.

[8] G.D. Chen, C.Y. Wang, and K.L. Ou. 2003. Using group communication to monitor web-based group learning. *Journal of Computer Assisted Learning* 19, 4 (2003), 401–415. doi:10.1046/j.0266-4909.2003.00045.x arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1046/j.0266-4909.2003.00045.x

[9] Katie Crawford and Desmond McEwan. 2022. Teamwork makes the dream work: How to be a great teammate. *Frontiers for Young Minds* 10 (2022), 685055.

[10] Joanna F. DeFranco and Philip A. Laplante. 2017. Review and Analysis of Software Development Team Communication Research. *IEEE Transactions on Professional Communication* 60, 2 (2017), 165–182. doi:10.1109/TPC.2017.2656626

[11] D Garrison, Martha Cleveland-Innes, Marguerite Koole, and James Kappelman. 2006. Revisiting methodological issues in transcript analysis: Negotiated coding and reliability. *The Internet and Higher Education* 9, 1 (2006), 1–8.

[12] Kathy Garvin-Doxas and Lecia J Barker. 2004. Communication in computer science classrooms: Understanding defensive climates as a means of creating supportive behaviors. *Journal on Educational Resources in Computing (JERIC)* 4, 1 (2004), 2–es.

[13] Dragan Gašević, Srećko Joksimović, Brendan R Eagan, and David Williamson Shaffer. 2019. SENS: Network analytics to combine social and cognitive perspectives of collaborative learning. *Computers in Human Behavior* 92 (2019), 562–577.

[14] Paul Luo Li, Amy J Ko, and Jiamin Zhu. 2015. What makes a great software engineer?. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, Vol. 1. IEEE, 700–710.

[15] Yanye Luther, Marcia Moraes, Sudipto Ghosh, and James Folkestad. 2024. OnDiscuss: An Epistemic Network Analysis Learning Analytics Visualization Tool for Evaluating Asynchronous Online Discussions. In *Advances in Quantitative Ethnography: 6th International Conference, ICQE 2024*. Springer.

[16] Yenye Luther, Lindsey Nielsen, Logan Cadman, Marcia Moraes, Sudipto Ghosh, and Bianca Trinkenreich. 2025. Replication Package. https://figshare.com/s/16ebaeee79fcab2d9031.

[17] Sara McComb and Deanna Kennedy. 2020. *Team Communication in Theory and Practice*. Springer International Publishing, Cham, 1–16. doi:10.1007/978-3-030-36159-4_1

[18] Marcia Moraes, James Folkestad, and Kelly McKenna. 2021. Using Epistemic Network Analysis to Help Instructors Evaluate Asynchronous Online Discussions. In *Second International Conference on Quantitative Ethnography: Conference Proceedings Supplement*. 19–22.

[19] Barbara Oakley. 2003. Coping with hitchhikers and couch potatoes on teams [Student's Corner]. *IEEE Engineering in Medicine and Biology Magazine* 22, 5 (2003), 9–10. doi:10.1109/MEMB.2003.9625655

[20] Barbara Oakley, Richard M Felder, Rebecca Brent, and Imad Elhajj. 2004. Turning student groups into effective teams. *Journal of student centered learning* 2, 1 (2004), 9–34.

[21] Christopher Ong and Kevin McGee. 2012. Good team-mates do more than help the team win: design factors that impact player concern about team-mate experience. In *Proceedings of the 24th Australian Computer-Human Interaction Conference* (Melbourne, Australia) *(OzCHI '12)*. Association for Computing Machinery, New York, NY, USA, 440–448. doi:10.1145/2414536.2414605

[22] Martin P Robillard, Deeksha M Arya, Neil A Ernst, Jin LC Guo, Maxime Lamothe, Mathieu Nassif, Nicole Novielli, Alexander Serebrenik, Igor Steinmacher, and Klaas-Jan Stol. 2024. Communicating Study Design Trade-offs in Software Engineering. *ACM Transactions on Software Engineering and Methodology* (2024).

[23] Vitor Rolim, Rafael Ferreira, Rafael Dueire Lins, and Dragan Găsević. 2019. A network-based analytic approach to uncovering the relationship between social and cognitive presences in communities of inquiry. *The Internet and Higher Education* 42 (2019), 53–65.

[24] Susan Ruff and Michael Carter. 2009. Communication learning outcomes from software engineering professionals: A basis for teaching communication in the engineering curriculum. In *2009 39th IEEE Frontiers in Education Conference*. 1–6. doi:10.1109/FIE.2009.5350442

[25] Pilar Sancho-Thomas, Rubén Fuentes-Fernández, and Baltasar Fernández-Manjón. 2009. Learning teamwork skills in university programming courses. *Computers & Education* 53, 2 (2009), 517–531.

[26] Sina Mahdipour Saravani, Sadaf Ghaffari, Yanye Luther, James Folkestad, and Marcia Moraes. 2023. Automated Code Extraction from Discussion Board Text Dataset. In *Advances in Quantitative Ethnography: 4th International Conference, ICQE 2022, Copenhagen, Denmark, October 15–19, 2022, Proceedings*. Springer, 227–238.

[27] Jennifer Scianna and Rogers Kaliisa. 2023. 'SSEEN': a networked approach to uncover connections between sentiment, social, and epistemic elements of student online forum discourse. *Educational technology research and development* (2023), 1–23.

[28] David Williamson Shaffer, Wesley Collier, and Andrew R Ruis. 2016. A tutorial on epistemic network analysis: Analyzing the structure of connections in cognitive, social, and interaction data. *Journal of Learning Analytics* 3, 3 (2016), 9–45.

[29] David Williamson Shaffer, David Hatfield, Gina Navoa Svarovsky, Padraig Nash, Aran Nulty, Elizabeth Bagley, Ken Frank, André A Rupp, and Robert Mislevy. 2009. Epistemic network analysis: A prototype for 21st-century assessment of learning. *International Journal of Learning and Media* 1, 2 (2009), 33–53.

[30] Maurício Souza, Renata Moreira, and Eduardo Figueiredo. 2019. Students perception on the use of project-based learning in software engineering education. In *Proceedings of the XXXIII Brazilian Symposium on Software Engineering*. 537–546.

[31] Gilbert W Stewart. 1993. On the early history of the singular value decomposition. *SIAM review* 35, 4 (1993), 551–566.

[32] Margaret-Anne Storey, Neil A Ernst, Courtney Williams, and Eirini Kalliamvakou. 2020. The who, what, how of software engineering research: a socio-technical framework. *Empirical Software Engineering* 25 (2020), 4097–4129.

[33] Zachari Swiecki and David Williamson Shaffer. 2020. iSENS: an integrated approach to combining epistemic and social network analyses. In *Proceedings of the tenth international conference on learning analytics & knowledge*. 305–313.

[34] Damian A Tamburri, Fabio Palomba, and Rick Kazman. 2019. Exploring community smells in open-source: An automated approach. *IEEE Transactions on software Engineering* 47, 3 (2019), 630–652.

[35] Yuanru Tan, Andrew R Ruis, Cody Marquart, Zhiqiang Cai, Mariah A Knowles, and David Williamson Shaffer. 2022. Ordered network analysis. In *International Conference on Quantitative Ethnography*. Springer, 101–116.

[36] Yuanru Tan, Andrew R. Ruis, Cody Marquart, Zhiqiang Cai, Mariah A. Knowles, and David Williamson Shaffer. 2023. Ordered Network Analysis. In *Advances in Quantitative Ethnography*, Crina Damşa and Amanda Barany (Eds.). Springer Nature Switzerland, Cham, 101–116.

[37] Yuanru Tan, Zachari Swiecki, Andrew R Ruis, and David Shaffer. 2024. Epistemic network analysis and ordered network analysis in learning analytics. In *Learning Analytics Methods and Tutorials: A Practical Guide Using R*. Springer Nature Switzerland Cham, 569–636.

[38] Bianca Trinkenreich, Ricardo Britto, Marco A Gerosa, and Igor Steinmacher. 2022. An empirical investigation on the challenges faced by women in the software industry: A case study. In *Proceedings of the 2022 ACM/IEEE 44th International Conference on Software Engineering: Software Engineering in Society*. 24–35.

[39] Bianca Trinkenreich, Marco Aurelio Gerosa, and Igor Steinmacher. 2024. Unraveling the drivers of sense of belonging in software delivery teams: Insights from a large-scale survey. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. 1–12.

[40] Szilvia Zörgő, Gjalt-Jorn Ygram Peters, Clara Porter, Marcia Moraes, Savannah Donegan, and Brendan Eagan. 2022. Methodology in the mirror: a living, systematic review of works in quantitative ethnography. In *Advances in Quantitative Ethnography: Third International Conference, ICQE 2021, Virtual Event, November 6–11, 2021, Proceedings 3*. Springer, 144–159.