Delft University of Technology

Document Version
Final published version

Licence
CC BY

Citation (APA)
Steenbergen, M., Pera, M. S., & Aivaloglou, E. (2026). Teamwork for Computer Scientists in Education and Practice. In *ITiCSE 2026 - Proceedings of the 31st ACM Conference on Innovation and Technology in Computer Science Education V. 1* (pp. 65-71). Association for Computing Machinery (ACM). https://doi.org/10.1145/3803400.3809339

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
In case the licence states “Dutch Copyright Act (Article 25fa)”, this publication was made available Green Open Access via the TU Delft Institutional Repository pursuant to Dutch Copyright Act (Article 25fa, the Taverne amendment). This provision does not affect copyright ownership.
Unless copyright is transferred by contract or statute, it remains with the copyright holder.

Sharing and reuse
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.

Teamwork for Computer Scientists in Education and Practice

Merel Steenbergen
Delft University of Technology
Delft, The Netherlands
M.A.Steenbergen@TUDelft.nl

Maria Soledad Pera
Delft University of Technology
Delft, The Netherlands
M.S.Pera@TUDelft.nl

Efthimia Aivaloglou
Delft University of Technology
Delft, The Netherlands
E.Aivaloglou@TUDelft.nl

Abstract

Teamwork is an integral part of computer science, but group projects in education do not always teach teamwork skills tailored to the context of software project teams. As a result, early-career computer scientists often struggle with team projects that involve interdependent tasks. This research focuses on identifying which teamwork skills are important to computer scientists at the start of their careers and how they perceive the career-education gap regarding teamwork. We conducted semi-structured interviews with 19 early-career computer scientists and analysed them using deductive and inductive coding. *Social skills* were perceived as the most important by the participants. Although *leadership* and *coordination* were among the most prevalent in university education, they were considered less important in industry. These findings can be used to motivate educational design choices in university group projects, focusing on teaching the teamwork skills that will be relevant to students' careers.

CCS Concepts

• **Social and professional topics** → **Computing education**; **Software engineering education**; • **Applied computing** → **collaboration learning**.

Keywords

computing education, software engineering education, teamwork, collaboration, group projects

ACM Reference Format:
Merel Steenbergen, Maria Soledad Pera, and Efthimia Aivaloglou. 2026. Teamwork for Computer Scientists in Education and Practice. In *Proceedings of the 31st ACM Conference on Innovation and Technology in Computer Science Education V. 1 (ITiCSE 2026), July 10–15, 2026, Madrid, Spain.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3803400.3809339

1 Introduction

Working in teams is the default in most Computer Science (**CS**) positions, making teamwork a crucial soft, i.e., professional, skill for computer scientists entering the workforce [6]. With the recent AI surge, more menial CS-related tasks that programmers are currently responsible for may become less important as they are replaced or supported by AI. Therefore, collaboration and teamwork may become even more important skills for a computer scientist as the focus of relevant skills shifts [15]. However, a well-documented gap exists between CS education and industry demands regarding teamwork, among other skills [3, 22, 26].

Teamwork skills in CS are typically taught through group-based project courses that emphasize soft skills, such as client interaction and teamwork, and technical skills, including programming [20]. However, students fail to see the relevance of these soft skills and avoid learning teamwork strategies in favour of technical skills [16]. Some unique challenges for effective team collaboration arise in CS-related projects and exacerbate the need for tailored teamwork education. For instance, the large gender gap [1], higher reported rates of neurodivergence [32], and a discrepancy in (programming) experience [7] lead to unbalanced teams. Furthermore, the software development life cycle requires teamwork tasks using CS-specific knowledge, such as code reviewing [23]. To make students more involved in learning teamwork, teaching methods need to be adapted to take these distinct challenges into account. However, based on existing literature and current practices, there is currently a lack of sufficient understanding of which aspects of teamwork are relevant to both CS university projects and industry teams.

In this work, we probe which teamwork skills are relevant to early-career computer scientists. For this, we consider how computer scientists learn teamwork skills during their education and which aspects of teamwork potentially lead to struggles when entering the workforce. Our work is framed by the following research questions (**RQs**):

RQ1 Which teamwork skills do early-career computer scientists perceive as important for their career?
RQ2 How do computer scientists perceive teamwork to be embedded in their university curriculum?
RQ3 How do computer scientists perceive the gap between education and required teamwork skills in the workplace?

To answer these questions, we conducted semi-structured interviews with 19 early-career computer scientists. We analysed the collected data using a combination of deductive and inductive coding, supported by KJ diagrams to organize and synthesize emerging themes. The insights from these interviews, guided by teamwork skills derived from existing literature, can inform course design and guide educators in focusing teaching for teamwork in CS degrees. Furthermore, this work contributes to the CS education research community (and beyond) by publishing the anonymous dataset[1] for other researchers to explore and build upon our findings.

2 Related Work

Employers tend to prioritise different soft skills than graduates do [12, 28], which can lead students to unintentionally develop skills that are less relevant to industry needs. This mismatch is reflected in novice software developers who indicate struggling with teamwork,



This work is licensed under a Creative Commons Attribution 4.0 International License.
ITiCSE 2026, Madrid, Spain
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2634-7/2026/07
https://doi.org/10.1145/3803400.3809339

[1] https://doi.org/10.4121/080f2a22-2b73-4dff-ab27-e2a0bf294980

as they are accustomed to smaller teams and more independent tasks requiring less actual collaboration [4]. This aligns with findings showing that CS students are inclined to approach a project by dividing tasks and working on them separately, minimizing contact and limiting skill development [31]. Therefore, grouping students and assigning them a task does not guarantee that students will learn collaboratively as a team [29], nor does it ensure that students will have equal opportunities to practice teamwork skills [11].

Despite teamwork being the default in the CS industry, students often show a preference for individual work. This reluctance is largely driven by concerns about social loafing and its potential negative impact on group grades [13, 18]. Teaching teamwork skills tailored to CS might mitigate these challenges and help students learn to work collaboratively, rather than individually in parallel.

Recently, an integrative framework has been introduced as a means to support assessment of specific teamwork skills per student [24]. Part of this framework specifically allows teachers to both detect and record the collaboration skills that an individual student has developed, linked to the context they appear [8]. Similar competency models have been developed in other domains, such as federal organizations [9]. Interestingly, human resource experts and experienced collaborators disagree on which skills, attributes, and attitudes are collaborative in nature [9]. However, literature identifies constructing shared knowledge, negotiation/coordination, and maintaining team function as the core facets of collaboration [30]. Together, these competency models form a basis for deconstructing teamwork into multiple skills and identifying which of these are most relevant in CS careers.

3 Methodology

To understand the relevance of teamwork skills in CS, we conducted online, semi-structured interviews guided by core literature on teamwork skills, summarised in Table 1. The meetings were recorded and transcribed using Microsoft Teams. The transcripts were manually corrected and anonymised, after which the recordings were deleted. Each interview took between 25 and 50 minutes.

Participants. Craig et al. define early-career computer scientists as "… graduates from four-year degree programs in Computer Science with between one and five years of industry experience" [6]. We extend this definition to account for different program lengths and the first year of working: A recent (less than 5 years) CS graduate with industry experience.

To recruit participants, we contacted educators from our professional networks to share the invitation with potential participants. Ultimately, 19 participants, from a diverse set of institutions and with varying nationalities and job titles, volunteered to take part in the study; their demographics are presented in the data repository. No remuneration was offered for taking part in the interviews. While recruitment started from the author's network, we aimed to reflect a wide range of geographic locations, work settings, and educational backgrounds to increase the diversity of the perspectives. Despite the global and diverse nature of early-career computer scientists, recurring patterns began emerging after 16 interviews, after which no new codes were added in the final three interview analyses, suggesting thematic saturation had been achieved.

Protocol. The interviews started with gathering consent, as approved by the ethics committee of the authors' institution. After inquiring about the participants' background in CS, we asked participants to describe situations in their education and career where each teamwork skill from Table 1 affected the team process. Since we expect to identify problems with teamwork when social loafers are present [13], some specific questions were asked about these situations. Finally, participants were tasked with prioritising teamwork skills and reflecting on what they felt was missing from their education regarding teamwork. The full interview protocol can be found in the data repository.

Data Analysis. To analyse the transcripts, we used a KJ diagrams approach [25], so we could combine the deductive codes, derived from the skills and RQs, with inductive coding results. Relevant quotes were labelled to create codes. Those codes were grouped into categories without considering the quotes belonging to those labels. These were grouped into families with relations, which were used to create written explanations of the data.

The interviews were conducted by the first author, who has a master's degree in Computer Science. To avoid bias, an independent researcher was invited to collaborate on the analyses. This researcher, with a non-CS degree from a different university, is working in the field of Engineering Education. Due to time constraints, only one interview was independently labelled by both researchers. We found a Cohen's kappa of 0.85, indicating a high level of agreement [17]. Disagreements were discussed until consensus was reached. This means that even though only one interview has been analysed by two researchers (~5%), the labels are adjusted accordingly for further coding by the lead researcher.

4 Results

Both deductive and inductive analyses were conducted to derive findings. We first present the deductive coding results, which explore the relevance of teamwork skills for early-career computer scientists and how these skills are integrated into CS education. This is followed by our reports on inductive findings, exploring the differences between career and education when it comes to teamwork. The results are accompanied by anonymous participant quotes. For interviews not conducted in English, quotes translated from their original language, Dutch, are noted with an asterisk.

4.1 Important Teamwork Skills for Career

Social skills were indicated by 10 participants as the most important teamwork skill that heavily influences other teamwork skills. Of these, six participants put *social skills* as the absolute most relevant skill (P1, P5, P6, P12, P13, P16); four others spoke about *social skills* combined with other skills (P4, P8, P14, P17). They indicate that social people will be more inclined to have the traits a good leader needs (*leadership*), offer help more readily (*assistance*), and keep negotiations civil (*negotiation*). Participants considered social skills to be almost synonymous with empathy, as demonstrated by P6's quote below.

> *"When you are negotiating with someone... I mean, we are still human. We value social skills and seem to value the arguments of the other person when we see how important it is to them." (P6*)*

Skill	Indicators	Sources
Assisting	Assisting others, asking for help, offering help, explaining things to others, giving feedback, reassigning or taking over tasks when necessary	[9], [30]
Gathering and Sharing Information	Sharing one's own information, giving updates, summarising process, doing research, sharing conclusions, asking for clarifications, not sharing irrelevant information	[9], [24], [30]
Coordination	Deciding project focus, dividing tasks, goal setting, monitoring progress, referencing team agreements, setting deadlines	[8], [24], [30]
Negotiation	Seeking win-win solutions, finding common ground, providing reasons to accept or reject an idea, welcoming conflict for the purpose of gaining new perspective, offer alternative solutions	[8], [24], [28], [30]
Leadership	Communicating project vision, taking the lead in meetings, allocating duties in cooperation with team members, promoting decision-making	[9], [28]
Social Skills	Building relationships, understanding other perspectives, showing empathy, using socially appropriate language	[8], [9], [24], [28]

Table 1: Teamwork skills guiding the interviews

Five participants indicated that *social skills* positively affect *negotiations* and are additionally good qualities for a leader to have. Four interviewees concretely discussed how the *social skills* of a leader can impact the team either positively or negatively. Three of them also explicitly discussed *assistance* in this context.

> *"I think he's a good leader, because he really wants to help us and he's very empathetic." (P4)*

> *"Leadership should not only be telling people what to do, but also helping them out." (P16)*

Gathering and sharing information was regarded the second-most important teamwork skill, prioritised by P9, P10, and P11, and combined with other skills by P3 and P19. Where P3 mainly focused on the aspect of making strong arguments for your opinion, P19 regarded it as more general communication skills.

Other teamwork skills were less prioritised. Only P2 felt that *coordination* was the most crucial skill. P3, P14, and P19 briefly mentioned it in their discussion of crucial skills in combination with others. Coordination was perceived to be highly overlapping with *leadership* by thirteen out of nineteen interviewees, especially in a professional context. An explanation for this is that participant consider their manager as their leader. Managers are often responsible for assigning tasks to team members and keeping track of progress, two of the most important coordination tasks.

> *"My manager's very good at giving us tasks, a lot of teamwork is because of him rather than individual team mates." (P10)*

Negotiation (P3, P17) and *leadership* (P14, P19) were both mentioned just twice as most important, always in combination with other skills. In professional contexts, overlaps between *leadership* and *negotiation* can occur when *negotiation* between team members is overridden by *leadership*, such as managers.

> *"And there are times that even though our team has agreed on an approach, we still have to change it based on the needs of someone senior." (P8)*

> *"One of the parts in a workplace that annoy me, is someone senior will block your code because they think it should be implemented another way." (P11)*

The teamwork skill least often indicated as most important is *assistance*. Only P15 mentioned it independently, and P8 discussed it in combination with other skills. *Gathering and sharing information* can support *assistance*, because when everyone has their own area of knowledge, *assistance* can be more tailored. Two other participants, P7 and P18, considered all teamwork skills equally important.

> *"In an organisation, you'll go to the same people because they can solve the questions." (P5*)*

> *"One person knows this topic, another person knows that topic. They come in handy for assistance." (P7)*

4.2 Learning Teamwork Skills in University

Assistance. None of the participants indicated having learned assisting skills. However, P3 and P19, from different universities, specifically mentioned that pair programming was very encouraged as a means for assistance.

> *"They encouraged actually paired programming where one person was the driver, one person being the navigator." (P19)*

P4 and P6 spoke about implicitly learning how to ask questions, for example because of being encouraged by peers. Others (P2, P11, P13, P18) indicated not learning this in university, but already having experience with assistance taught at home or in high school. P2, P11, and P18 indicated being helpful could be a character trait rather than a skill.

> *"If someone is helpful, that doesn't necessarily mean that they have learned that during their education. But mostly it's just their personal traits." (P2)*

> *"I think asking for help goes back as early as you know your childhood, where if you weren't tall enough, you'd ask someone taller to help you grab something from a shelf." (P11)*

Gathering and sharing information. Implicit learning either during education or during work were the most commonly mentioned by the participants. All participants indicated never having lectures or lessons about this skill.

"There weren't a lot of workshops or seminars sort of thing that I did in university." (P7)

Coordination. Participants' experiences varied from having taken dedicated courses to never having been graded on the process of a project. Whereas P11 cannot imagine how teaching practical project coordination would be feasible in university due to the scope and duration required, P14 describes their experience with exactly such a course. P19 also argues the need for mock projects where students can feel free to fail.

"You can't tell someone how to plan and coordinate, expect them to do it and then try to grade them on it in a four month time span, right? I think it comes from being given a problem that's big enough that you can't complete it without planning and coordination, and you can't complete it alone." (P11)

"We had a mock project in collaboration with the company. We had to coordinate how to efficiently do this with 12 people. We had weekly meetings with a representative of the company throughout six or seven weeks." (P14)

Negotiation. Four participants (P1, P3, P4, P18) explicitly indicated not learning this in university, and two others (P6, P16) said they got experience in other ways than teacher-guidance. P18 does note it would have been beneficial to learn effective negotiations.

"If there is a way to teach this kind of conflict resolution and giving constructive feedback, it'd be a very valuable skill to have that's missing in a lot of undergraduate degrees." (P18)

Leadership. P2, P3, and P4 specifically indicated *leadership* was not formally taught. Other participants (P5, P6, P12) spoke about their experience with project courses, mainly the ones using Scrum, where team roles were rotated every meeting. They identified the student serving as Scrum Master as the team leader, but did not report receiving leadership instruction by teachers. P9 and P10 spoke about courses teaching the theory behind leadership, but indicated them as not very effective for practical application.

"I don't know how well that was taught. (…) It's like you said: In theory. (…) And then when the time comes, it's like that's not how it works." (P9)

Social Skills. None of the participants indicated they had been taught *social skills* in school. However, most of them do agree that they should be taught. Such as P16 says:

"In university, they teach you how to write a paper, academic stuff. It's different when you go into industry. You are not just a coder or software developer, you don't sit at your PC all day. You need to go into meetings, you need to know how to talk to people. That's why I think social skills is really important, because you will not be in your own bubble forever." (P16)

P6, P9, P13, and P15 specifically discussed *social skills* and neurodivergence. The participants discussed that learning social skills could be an advantage to neurodivergent students, but also to neurotypical students working with neurodivergent students, as shown from P6's interview excerpt. P19 has a similar view, where they understand that neurodivergent individuals might have a harder time picking up social cues, but that does not absolve them of being considerate towards others.

"I have a colleague on the spectrum and he's bad at communicating, but I've noticed that's only non-verbally. If you're clear about your intentions and questions, it's much better. But it does require a different approach. We aren't taught that at university." (P6)

"My brother-in-law is autistic, so he doesn't always pick up on all the social cues, but he tends to be very considerate and tends to try to be patient with other people. (…) I do understand that a lot of computer science students have a bit more difficult time picking up on social cues. I find that kindness and patience and consideration stuff have always been bigger problems I've had with other people." (P19)

4.3 The Career-Education Gap

Scope. A recurring topic in the interviews was the difference in scope between university projects and professional projects, mainly in regard to *coordination* and *sharing information*. Participants P5, P7, and P12 noted that under-sharing is quite difficult in a short project with a small team and well-defined scope, which is usually the case in university. However, it becomes significantly harder to share and gather all relevant information in a large organisation.

"In a study project, it isn't difficult to share that information with five people. If you extrapolate that to an organisation of a thousand people where there has been misinformation for years... During your education you are not prepared for the amount of misinformation there is." (P5)*

"During the studies you have a piece of perfect knowledge. However, here I am in a situation where you don't have that context or you don't even know if you have the correct information." (P5)*

Hierarchy. Another main difference is the hierarchy found in most workplaces, but not in universities. Although educators in university could be equated to managers in the workplace, educators are not responsible for the projects their students deliver, while managers often carry responsibility towards superiors or clients. Furthermore, team *leadership* is well-defined in industry, while in university it is usually determined by the team itself.

"In school we're all studying the same and somebody takes the lead in meetings. And then flipping to a manager, it's not the same. (…) I don't have the same authority." (P12)

An effect of this hierarchy propagates to *assistance*. Teachers and teaching assistants are more often than not readily available to provide help in university and actively encourage asking questions. Team members are all on the same level and help each other using methods such as peer feedback. However, multiple participants (P7, P8, P12, P15) indicated feeling more self-conscious about asking for assistance in professional environments.

"In university everybody's your peer. You're all at the same level and the knowledge levels. There's a bit of hesitation once it comes to a company." (P7)

"The difference between education and work, is the formality. Just because we discuss between friends during my education it was easier to communicate and during my work experience it was more professional." (P15)

Pressure. Pressure introduced by the hierarchy in professional environments affects the participants' freedom to shape their work. Whereas in university they had the choice to work on the tasks they preferred, their managers are the ones to assign tasks in their careers. This was indicated by P11, P12, and P15, who all have different jobs.

> *"Things come down from on top, (…) and then eventually it gets to the people who implement things like their software engineers." (P11)*

> *"In this job, it's like we need to do this, could you do that? (…) And in school, I had more to say what I wanted to do." (P12)*

This pressure manifests in a difference in motivation and priorities too. In university, students can get away with doing the bare minimum without major consequences, but in professional life they feel more external motivation because of the expectations of their superiors and client. This was discussed by P4, P11, P14, and P15.

> *"So in school there's just people who don't really communicate. (…) In the workplace, I've seen it less. (…) I think it happens a lot less because their livelihood depends on it." (P11)*

> *"The main difference between education and work is the commitment to the client. So you feel pressure at work." (P15)*

Another effect of the pressure, according to P3, is less conflict because of the need to maintain a professional attitude. However, P19 experienced higher pressure to perform in university. This might be due to a difference in careers, as they have jobs with different responsibilities and expectations. Although in different contexts, both participants hint that the pressure to perform leads to conflict, instead of constructive negotiations.

> *"I think it matters a lot whether you like someone when we're talking about constructive criticism. That is a big difference between uni projects and work. In uni you can just not like someone and therefore not like their feature too." (P3*)*

> *"(At work) It didn't feel like the stakes were so high. (…) whereas I feel like sometimes the stakes are so high in academia, everyone wants the high grades, (…). So I find that it's easier to fall into emotional conflicts in the academic setting." (P19)*

Job interviews. Another noteworthy outcome is that the composition of the industry workforce differs from that of the university population. Job interviews introduce some filtering and bias into the workplace. Interestingly, this seems to affect our early-career computer scientists differently, as shown by the quotes below. Notably, P11 (male) and P4 (female) both discuss experiences in big tech companies, yet their views differ.

> *"When you go to these companies, you meet people who've been vetted before multiple times, really. (…) As a result, I haven't really met anyone who isn't a great communicator in the workplace. (…) I see poor communicators more often in school than in than in the workplace." (P11)*

> *"You might encounter people who display sexism, this happened to me mostly in industry and not in university. (…) In industry you encounter so many different types of people that I think the issues go a bit into more depth and person." (P4)*

Dealing with slackers. More differences between teamwork in education and career were found in the strategies for dealing with slackers. Most participants indicated using ineffective methods in university, such as confrontation or ignoring the problem. Five participants (P1, P5, P6, P9, P12) regretted that and would currently approach a slacker with more compassion and try to understand their situation before judging, so they could offer help. Most participants show some level of growth and would currently show more compassion, truly trying to help everyone achieve team goals, rather than focusing on individual contributions.

> *"I think the big difference is that I now realise that there are more reasons someone doesn't contribute than laziness." (P1*)*

Seven participants (P2, P10, P11, P13, P14, P15, P18) said to involve a supervisor if a colleague was slacking. Some of them even considered dealing with slackers to not be their responsibility and indicated that involving a supervisor or manager would be their first approach to dealing with a slacker.

> *"Your manager typically takes care of it for you. That person will usually get fired. (...) If someone isn't pulling their weight, I don't want to be the one to call them out." (P11)*

5 Discussion

Our analysis of the interviews revealed which teamwork skills are relevant to early-career computer scientists and their reflections on education for these skills. Below, we discuss these findings in relation to our RQs and prior literature.

RQ1: 'Which teamwork skills do early-career computer scientists perceive as important for their career?' *Social skills*, i.e., showing empathy and consideration for others, were considered most important. We attribute this to good *social skills* having a positive effect on the other skills. For example, participants observed that negotiations became disruptive to the team process when team members were overly invested in their own solutions, making them unreceptive to feedback, which they tied to a perceived lack of *social skills*. External frameworks support this connection: Succi and Canovi [28] list *Conflict Management & Negotiation Skills* within the broader category of social skills, and Durán and Amandi [8] categorise listening as an attribute closely related to *social skills*.

Remarkably, while participants consider *coordination* and *leadership* as highly overlapping, the literature that these skills emerged from are unconnected. Coordination is described through indicators such as guiding group processes, promoting decision-making, planning activities, allocating subtasks and roles, and monitoring execution [8, 24, 30]. Leadership has as indicators taking initiative, positioning oneself as the leader, and motivating and guiding others [9, 28]. This reflects a difference in perspective: leadership assumes responsibility by one person, whereas coordination can be a collective team effort. Whether leadership is a necessary skill for teamwork in CS curricula depends on the underlying view of collaboration: either as a team effort, or as collective efforts of individuals. In educational teamwork contexts, leadership may not be essential, especially if the manager will assume team-leadership in entry CS positions. However, shifting students to view from a group of individuals to a team, and teaching *coordination* skills more complex than task distribution might benefit all students.

RQ2: 'How do these computer scientists perceive teamwork to be embedded in their university curriculum?' Participants generally felt that teamwork skills were only partially embedded in their university curriculum, and in some cases, not addressed at all. While many courses attempted to integrate *coordination*, practical courses were often too small in scale to accurately mirror real-world projects, as also reflected by the literature [4]. As evident from the interviews, university group projects tend to emphasize task division over the more complex dynamics of *coordination*, resulting in group work rather than true collaboration [24, 29]. This is reflected in the practice of weekly shifting tasks and roles, whereas tasks, roles, and *leadership* are defined according to individual strengths and skills in the real world. These findings align with the literature, where employers indicate the need for less theory and more practice in university curricula [12].

Social skills were described as most important, yet no participant recalled explicit teaching in this area. Some indicated they would appreciate the integration of *social skills* in the curriculum. Especially since people with Autism Spectrum Disorder are more likely to enrol in STEM disciplines, such as CS [32]. Teaching *social skills* could be tailored to expressing consideration for peers, even when social cues are not picked up on. Students with autism already spend a lot of time learning *social skills* outside of their coursework [19]. It could be beneficial to introduce well-defined moments to practice the skills learnt in these extracurricular activities and integrate these into the coursework. An alternative explanation is that teamwork skills are embedded in course design, but students do not recognize them as such. Prior research shows that perceived learning gains do not always reflect actual learning outcomes [21]. Nevertheless, the difficulties described by our participants suggest that key teamwork skills are not sufficiently taught and more focus on interpersonal skills is necessary.

RQ3, 'How do these computer scientists perceive the gap between their education and required teamwork skills in the workplace?' Participants reported that workplace projects typically involve larger teams, bigger code bases, and longer timelines than academic assignments. This affects *coordination* and *gathering and sharing information*. In university, students mostly take the approach of dividing the tasks, working on those individually, and merging the code at the end of the week [31]. However, literature shows that this can lead to surprises when working in industry projects where this is not feasible and therefore, university projects to maintain legacy applications are particularly valued, as they closely resemble real-world projects [6].

Professional environments involve responsibility to superiors and clients. This introduces hierarchy and makes participants feel insecure about asking questions at work, aligning with prior research that noted the difficulty of identifying experts in multi-developer teams compared to the easy access to professors and teaching assistants in academia [6]. While this problem might not be specific to CS, pair programming could be a solution to this in CS contexts. Pair programming leads to a community that encourages question-asking in a safe environment, and could therefore be valuable to integrate in education and early-career paths [27]. Furthermore, it increases confidence, which was also mentioned as a barrier to asking questions to superiors [10, 14].

A positive consequence of this pressure is the fact that participants felt more forced to maintain a professional attitude. In most cases, this led to more constructive *negotiation* rather than disruptive conflict. Furthermore, most participants show a more mature and productive way of dealing with slackers, showing more empathy. Pair programming might increase the self-efficacy of slackers, making them more inclined to ask for help when needed [4, 10]. Still, teamwork education should incentivise students to uphold this professional attitude, in order to prepare them for the job market and guide them from disruptive conflicts to constructive ones.

Unfortunately, the filtering that job interviews do can introduce bias in the workplace. STEM, and therefore CS, are known for their low diversity in socioeconomic status and gender [2]. Job interviews reinforce this lack of diversity, also known as the leaky pipeline [5]. Minorities may face frustrations during teamwork, and this effect may be more pronounced in the workplace [11].

Limitations. Participants' reflections on university training are specific to their own programs and institutions; they might be influenced by their own biases, perceptions, and memory limitations. Further, social desirability biases might have influenced participants' expression of potential shortcomings. Reported outcomes are based on 19 interviews from participants worldwide. Although offering a starting point, their views may not capture the broad variation in CS-related job titles and university locations. Additionally, the distribution of universities is skewed; Europe, and specifically the Netherlands, is over-represented; Asia, Africa, and South America are under-represented, and Oceania not represented. This imbalance is largely due to challenges in reaching participants. Note that the search for participants began with our professional network and participation was voluntary, which might have introduced bias towards extraverted participants with positive experiences.

The insider-perspective of the lead researcher may have affected the lens through which the data has been analysed. To mitigate potential bias, we employed the KJ diagram method to analyse the data, and included a second coder to provide independent verification of the coding scheme. Unfortunately, the second coder was only able to code one interview, not providing enough coverage to truly claim inter-rater reliability.

6 Conclusions

With this study, we identified the teamwork skills that early-career computer scientists consider important for their careers, and the shortcomings they perceive in their educational preparation. Among participants, *social skills* emerged as the most important, largely because of their influence on other teamwork skills, such as *negotiation* and *assistance*. Participants also perceived *leadership* and *coordination* as less important at this stage of their careers, while those skills were more prevalent in university education.

Outcomes from this work offer insights into how early-career computer scientists perceive the gap between their academic preparation and professional teamwork demands. They also provide valuable starting points for the design of educational programmes and courses. In particular, they highlight limitations in how current academic projects prepare students for real-world teamwork. More practice-oriented learning experiences would benefit CS students in developing teamwork skills essential for their early careers.

References

[1] Rukiye Altin and Andreas Mühling. 2024. Why Female Students Are Dropping out of CS Programs. In *Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1*. ACM, Milan Italy, 304–310. doi:10.1145/3649217.3653635

[2] K. Arbuckle, E. J. Bethell, D. J. Hawthorn, K. Hunt, M. Khera, Z. Lewis, J. Mitchell, M. H. Nicholl, and L. A. Reynolds. 2025. Low socioeconomic status is an under-recognised source of challenges in academia. *Journal of Zoology* 325, 4 (2025), 267–275. doi:10.1111/jzo.13250

[3] Nimmi Arunachalam, Stephanie J. Lunn, Mark Weiss, Jason Liu, and Giri Narasimhan. 2024. Foot in the Door: Developing Opportunities for Computing Undergraduates to Gain Industry Experience. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2024)*. Association for Computing Machinery, New York, NY, USA, 74–80. doi:10.1145/3626252.3630857

[4] Andrew Begel and Beth Simon. 2008. Novice software developers, all over again. In *Proceedings of the Fourth international Workshop on Computing Education Research (ICER '08)*. Association for Computing Machinery, New York, NY, USA, 3–14. doi:10.1145/1404520.1404522

[5] Christina Bock. 2022. Diversity, Equality, and Inclusion in Our Professions: A Thin and Leaky Pipeline. *The Electrochemical Society Interface* 31, 1 (March 2022), 41. doi:10.1149/2.F08221IF

[6] Michelle Craig, Phill Conrad, Dylan Lynch, Natasha Lee, and Laura Anthony. 2018. Listening to early career software developers. *J. Comput. Sci. Coll.* 33, 4 (April 2018), 138–149.

[7] Jialin Cui, Runqiu Zhang, Ruochi Li, Fangtong Zhou, Yang Song, and Edward Gehringer. 2024. How Pre-class Programming Experience Influences Students' Contribution to Their Team Project: A Statistical Study. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1*. ACM, Portland OR USA, 255–261. doi:10.1145/3626252.3630870

[8] E. B. Durán and A. Amandi. 2009. Personalised collaborative skills for student models. *Interactive Learning Environments* 19, 2 (Mar 2009), 143–162. doi:10.1080/10494820802602667

[9] Heather Getha-Taylor. 2008. Identifying Collaborative Competencies. *Review of Public Personnel Administration* 28, 2 (Mar 2008), 103–119. doi:10.1177/0734371X08315434

[10] Isabella Graßl and Gordon Fraser. 2024. Equitable Student Collaboration in Pair Programming. In *Proceedings of the 46th International Conference on Software Engineering: Software Engineering Education and Training* (Lisbon, Portugal) *(ICSE-SEET '24)*. Association for Computing Machinery, New York, NY, USA, 274–285. doi:10.1145/3639474.3640086

[11] Isabella Graßl, Stephan Krusche, and Gordon Fraser. 2023. Diversity and Teamwork in Student Software Teams. In *Proceedings of the 5th European Conference on Software Engineering Education* (Seeon/Bavaria, Germany) *(ECSEE '23)*. Association for Computing Machinery, New York, NY, USA, 110–119. doi:10.1145/3593663.3593687

[12] Dianne Hagan. 2004. Employer satisfaction with ICT graduates. In *Proceedings of the Sixth Australasian Conference on Computing Education - Volume 30 (ACE '04, Vol. 30)*. Australian Computer Society, Inc., AUS, 119–123.

[13] David Hall and Simone Buzwell. 2013. The problem of free-riding in group projects: Looking beyond social loafing as reason for non-contribution. *Active Learning in Higher Education* 14, 1 (3 2013), 37–49. doi:10.1177/1469787412467123

[14] Brian Hanks, Sue Fitzgerald, Renée McCauley, Laurie Murphy, and Carol Zander. 2011. Pair programming in education: a literature review. *Computer Science Education* 21, 2 (2011), 135–173. doi:10.1080/08993408.2011.579808

[15] Michael Jones, Sandra Idrovo-Carlier, and Alfredo J. Rodriguez. 2022. Automation in Colombia: assessing skills needed for the future of work. *Higher Education, Skills and Work-Based Learning* 12, 2 (01 Jan 2022), 225–240. doi:10.1108/HESWBL-01-2021-0003

[16] Yekaterina Kharitonova, Yi Luo, and Jeho Park. 2019. Redesigning a Software Development Course as a Preparation for a Capstone: An Experience Report. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19)*. Association for Computing Machinery, New York, NY, USA, 153–159. doi:10.1145/3287324.3287498

[17] Mary L. McHugh. 2012. Interrater reliability: the kappa statistic. *Biochem Med (Zagreb)* 22, 3 (2012).

[18] Robert Mcquade, Esther Ventura-Medina, Sally Wiggins, Gillian Hendry, and Tony Anderson. 2020. *Students' strategies for managing social loafers in PBL: Interactional means of dealing with unequal participation in group work*. Purdue University Press, 275–297. doi:10.2307/j.ctvs1g9g4.14

[19] Makayla Moster, Ella Kokinda, Matthew Re, James Dominic, Jason Lehmann, Andrew Begel, and Paige Rodeghero. 2022. "Can you help me?": an experience report of teamwork in a game coding camp for autistic high school students. In *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Software Engineering Education and Training* (Pittsburgh, Pennsylvania) *(ICSE-SEET '22)*. Association for Computing Machinery, New York, NY, USA, 50–61. doi:10.1145/3510456.3514154

[20] Damla Oguz and Kaya Oguz. 2019. Perspectives on the Gap Between the Software Industry and the Software Engineering Education. *IEEE Access* 7 (2019), 117527–117543. doi:10.1109/ACCESS.2019.2936660 Conference Name: IEEE Access.

[21] Adam M. Persky, Edward Lee, and Lauren S. Schlesselman. 2020. Perception of Learning Versus Performance as Outcome Measures of Educational Research. *American Journal of Pharmaceutical Education* 84, 7 (2020), ajpe7782. doi:10.5688/ajpe7782

[22] Alex Radermacher and Gursimran Walia. 2013. Gaps between industry expectations and the abilities of graduates. In *Proceedings of the 44th ACM technical symposium on Computer science education*. ACM, Denver, CO, USA, 525–530. doi:10.1145/2445196.2445351

[23] Christopher Scaffidi. 2018. Employers' Needs for Computer Science, Information Technology and Software Engineering Skills Among New Graduates. *International Journal of Computer Science, Engineering and Information Technology* 8, 1 (Feb. 2018), 01–12. doi:10.5121/ijcseit.2018.8101

[24] Verena Schürmann, Nicki Marquardt, and Daniel Bodemer. 2024. Conceptualization and Measurement of Peer Collaboration in Higher Education: A Systematic Review. *Small Group Research* 55, 1 (Feb 2024), 89–138. doi:10.1177/10464964231200191

[25] Raymond Scupin. 1997. The KJ Method: A Technique for Analyzing Data Derived from Japanese Ethnology. *Human Organization* 56, 2 (1997), 233–237. doi:10.17730/humo.56.2.x335923511444655

[26] Chris Simmons and Lakisha Simmons. 2010. Gaps in the computer science curriculum: An exploratory study of industry professionals. *Journal of Computing Sciences in Colleges* 25 (Jan. 2010), 60–65.

[27] Beth Simon and Brian Hanks. 2007. First year students' impressions of pair programming in CS1. In *Proceedings of the Third International Workshop on Computing Education Research* (Atlanta, Georgia, USA) *(ICER '07)*. Association for Computing Machinery, New York, NY, USA, 73–86. doi:10.1145/1288580.1288591

[28] Chiara Succi and Magali Canovi. 2020. Soft skills to enhance graduate employability: comparing students and employers' perceptions. *Studies in Higher Education* 45, 9 (Sep 2020), 1834–1847. doi:10.1080/03075079.2019.1585420

[29] Mark Summers and Simone Volet. 2010. Group work does not necessarily equal collaborative learning: evidence from observations and self-reports. *European Journal of Psychology of Education* 25, 4 (2010), 473–492. doi:10.1007/s10212-010-0026-5

[30] Chen Sun, Valerie J. Shute, Angela Stewart, Jade Yonehiro, Nicholas Duran, and Sidney D'Mello. 2020. Towards a generalized competency model of collaborative problem solving. *Computers & Education* 143 (Jan. 2020), 103672. doi:10.1016/j.compedu.2019.103672

[31] Anna van der Meulen and Efthimia Aivaloglou. 2021. Who Does What? Work Division and Allocation Strategies of Computer Science Student Teams. In *Proceedings of the 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*. Association for Computing Machinery, New York, NY, USA, 273–282. doi:10.1109/ICSE-SEET52601.2021.00037

[32] Xin Wei, Jennifer W. Yu, Paul Shattuck, Mary McCracken, and Jose Blackorby. 2013. Science, technology, engineering, and mathematics (STEM) participation among college students with an autism spectrum disorder. *Journal of Autism and Developmental Disorders* 43, 7 (2013), 1539–1546. doi:10.1007/s10803-012-1700-z