The High-Low Divide: How Teamwork Learning in Undergraduate Computer Science Is Shaped By Cultural Context

Investigating How Cultural Context Affects Teamwork Pedagogy

George-Matei Andrei[1]

Supervisor(s): Sole Pera[1]**, Merel Steenbergen**[1]

[1]**EEMCS, Delft University of Technology, The Netherlands**

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 21, 2026

Name of the student: George-Matei Andrei
Final project course: CSE3000 Research Project
Thesis committee: Sole Pera, Merel Steenbergen, Masoud Mansoury

Abstract

Teamwork is fundamental to computer science education, however, localized cultural aspects are deeply ingrained within pedagogy, shaping how collaboration is taught. As a result, fresh graduates struggle to adapt to the multicultural demands of a globalized workforce. With a basis in Hall's context theory, this research investigates the cultural boundaries impacting teamwork teaching in undergraduate computer science curricula. Utilizing deductive coding and cross-case comparison, this study analyzes 14 syllabi from a low-context institution (Delft University of Technology) and 8 syllabi from a high-context institution (KAIST). The findings reveal a definitive cultural divide, TU Delft relies on rigid textual instruction, regular supervision, and individual accountability, whereas KAIST prioritizes instructor autonomy, holistic goals, and implicit trust to foster collaboration. Crucially, explicit teamwork learning goals are predominantly missing across both cases due to a prioritization of technical skills. As a result, the course deliverables and additional assessment methods reflect the implicit learning goals for the students. These findings provide aid for curriculum designers who wish to integrate cultural context-varied teamwork strategies, bridging the gap between the academic instruction received by students and the expectations of the globalized industry.

1 Introduction

In the field of computer science, most careers are teamwork-based [8], however, there is a gap between what is expected of new hires in the industry and the abilities of fresh graduates [1, 26]. This discrepancy has become increasingly pronounced, as 80% of software engineering jobs now operate in fully remote or hybrid environments [11]. As a result, cross-cultural communication and multi-context collaboration have become a daily requirement in the workforce [31].

Hall's context theory is central to understanding the complexities of globalized work. The framework categorizes cultures based on their communication styles and relational profiles. Low-context cultures are intrinsically text-driven, direct, and individualistic, which, in computer science, translates into a reliance on explicit documentation, rigid requirements and specifications, as well as externalized task tracking. In contrast, high-context cultures prioritize implicit understanding, group unity, and relational alignment, thus relying on shared norms, consensus building, as well as verbal and nonverbal group cues over written instructions. [2, 12]. In educational settings, these cultural tendencies influence how students approach group work, creating tension between the low-context demand for individual efficiency and high-context preferences for group unity. [15, 25]

When applied to the competency gap in computer science education, Hall's theory highlights a critical challenge. While the software industry has rapidly become a more globalized, multi-context system, many undergraduate computer science curricula still rely on localized, single-context teamwork applications. This divergent design fails to teach undergraduate students how to adapt their collaborative methods when approached by different cultural working environments, creating a mismatch between academic preparation and global workforce expectations.

This issue persists due to the curricula of universities not being culturally neutral. Instead, study guides, syllabi, and university processes are all aspects deeply influenced by the local context of the institution. Previous research documents how students from diverse cultural backgrounds approach group work during projects [15, 25], finding issues such as the frustration of cultural minority members [9] or the need for early instructor mediation in cross-cultural team formations [27]. However, while these behaviors are understood, a disconnect arises about how curricula are structured to impact these dynamics. It remains unclear if and how formal teaching methods and explicit learning goals are shaped by the cultural context within which they are implemented, or if they offer preparation for computer science students to navigate the cultural shifts they will encounter in the global industry.

To address this gap, this paper investigates the explicit teaching methods and learning goals implemented by faculty members in high and low-context university environments, as well as conducting an analysis of the differences between them. The findings of this work can be used to help computer science curricula designers make informed and culturally aware choices to improve students' readiness. The investigation is framed by the following main research question and two sub-questions:

RQ How does the teaching of teamwork in undergraduate computer science curricula vary across different cultural context learning environments?

Sub-RQ1 Which methods are used to fulfill teamwork learning goals in high-context and low-context environments?

Sub-RQ2 How does cultural context affect which learning goals are matched to learning strategies in undergraduate curricula?

To answer the research questions, specific teaching methods and learning goals were collected from universities in low-context and high-context countries. Deductive coding was used in conjunction with cross-case comparison to effectively compare the results from each cultural context. The insights from these findings, together with information from previous literature, can help guide educators towards incorporating teaching teamwork in a way that puts more attention on the intercultural aspects of teamwork.

The rest of the paper is structured as follows. Section 2 focuses on previous research on teamwork in computer science and the intercultural aspects of it. The third section covers the methods used to collect and analyze data, which is then followed by a section discussing responsible research. The fifth section presents the results of the research, after which the sixth section discusses the main findings in relation to

the research questions, as well as the limitations of the work and future paths for further research. Finally, the last section presents the conclusion of the paper.

2 Related Work

The gap in expectations between the industry and what is taught to undergraduates is something that has been studied and proven repeatedly over time [1, 8]. However, this gap keeps affecting students, as the demand for skilled workers, with competencies proven through internships, keeps increasing [1], [16]. This is further exacerbated by novice developers being accustomed to smaller teams and having more independence [3], but also by current teaching methods, such as grouping students and assigning them a task, which does not give equal opportunities for students to learn teamwork equally [9], or that their collaboration will have sufficient depth [28]. These examples also show the current focus of research on individual or group outputs rather than specific pedagogical methods for teamwork in computer science.

In terms of the categorizations of cultures, Hall's context theory presents two distinct groups, high-context cultures and low-context cultures [12]. Within this spectrum, Asian cultures lean more towards high-context cultures, while Western cultures do so towards low-context cultures. In high-context cultures, such as South Korea, non-verbal cues, including interpersonal relationships, body language, and gestures, hold much more weight and need to be taken into consideration when communicating, as they can be of higher importance than words. On the other hand, in low-context cultures, such as the Netherlands, words that are explicitly communicated should be taken literally, as communication is more direct and verbose.

The cultural context that the students are part of shapes not only how they approach teamwork, but also which aspects they focus on the most. Students from countries sharing similar cultural contexts, even if they are from different countries, show no significant differences in terms of communication [24], as well as proper work division and collaboration [5]. However, when the cultural contexts of participants begin to mix, issues can begin to arise, such as the frustration of group members [9], the need for early intervention to be able to alleviate communication problems [27], or more simple issues such as differences in language proficiency [10].

Differences in communication styles, such as students from low-context cultures emphasizing individual contributions and efficiency, while students from high-context cultures prioritize group understanding [15, 25], as well as learning approaches specific to the cultural context, also show that in some cases individuals from low-context cultures end up with a more negative perception of collaborative learning [25]

Additional discrepancies can also be seen in terms of the aspects of teamwork that students from high and low context cultures put most emphasis on. Low-context cultures perceive traits such as language differences, trust, and familiarity as more important, while high-context cultures perceive leadership, communication, and cultural differences as more important [19]. Furthermore, challenges can arise in mixed teams due to differences in attitudes towards grades, influences of the course lecturer, and differences in autonomy [10]. However, if taught or directly practiced in intercultural scenarios, the expectations and importance placed on these values do change and become more aligned, as teams change their perception of these cultural dimensions [18].

3 Methodology

To investigate the difference between teaching teamwork in high-context and low-context cultures, we investigated available curricula for universities in each category. The data was collected from the universities' websites, which were transferred to ATLAS.ti for analysis.

3.1 Data collection

The universities that were selected as case studies were Delft University of Technology (TU Delft) for low-context culture countries, and KAIST for the high-context culture. They were selected due to their identity as technical universities, with both having a separate computer science curriculum. Additionally, while both universities accept international students, the majority of them are still local or from similar cultural contexts, keeping the cultural context consistent, as the TU Delft's computer science program has 41% of people in the bilingual track [22], and at KAIST, overall international students make up less than 10% of the student body [7].

The data was retrieved from each university's respective course website, from where courses containing teamwork were selected. For the courses selected from KAIST, the newest available version was chosen, with courses being selected either from the 2026 spring semester or the 2025 autumn semester. The courses chosen were considered to contain teamwork if they had a group assignment that was required to be completed for either passing the course or additional grades. As a result, 14 courses from TU Delft and 8 courses from KAIST were chosen, the full list being available in Appendix A. The sample size variation arises from differences in transparency and availability between the two universities, as at KAIST, the professors themselves dictate what is presented on the public website.

3.2 Data analysis

To analyze the information from the study guides, a cross-case comparison [32] using the cross-case matrix framework for visualization [21] was performed on the data resulting from deductive coding, in order to depict the comparison results between the two cultural contexts. A distinct set of codes was developed for each sub-research question, with codes grouped into three themes: general codes, low-context culture, and high-context culture.

Table 1 features the codes developed for the first sub-question, based on literature on teamwork methods and cultural communication styles [6, 12, 13, 17, 19, 29]. Similarly, Table 2 presents the coding scheme used in the analysis of the second sub-question, which is derived from literature surrounding intercultural collaboration and teamwork learning goals [12, 13, 25, 29].

For the second sub-question, additional focus was placed on aspects most strongly emphasized during student evaluation, as grading criteria can reflect learning goals which

are not explicitly mentioned [4]. The codes for both subquestions also include the coding of missing data to help detect additional gaps present within the syllabi.

4 Responsible Research

The assignment and analysis of the codes were conducted by a single investigator. In the absence of a secondary coder to verify inter-rater reliability, the qualitative interpretation may have a heightened risk of subjectivity. The deductive coding architecture employed aids in limiting the impacts of subjective analysis.

Regarding the ethical aspects of the research, the methodology relies exclusively on publicly available institutional documentation retrieved directly from the public websites of TU Delft and KAIST. Since the study did not include human participants, interviews, or the collection of personal data from faculty or students, the research does not pose any ethical risks related to the use of such data.

Large Language Models (LLMs) were only used to support the writing process, being limited to improvements in readability and the creation of tables. The author verified and edited the content produced before use and assumes full responsibility for the content presented.

5 Results

Deductive coding was conducted to derive findings, from which we present the main findings of the cross-case comparison in the result matrix seen in Table 3. This matrix presents the main findings in terms of each research question and provides information on the main differences observed. The results of each question will be presented separately with quotes in relevant places.

5.1 Teaching methods

In regard to the teaching methods presented in both contexts, the first big difference arises in the way the process is presented, especially in relation to input from instructors or teaching assistants (TAs).

Within TU Delft, 5 of the 14 courses have explicit weekly meetings with a supervisor or teaching assistant as a method used for monitoring the progress of each student, as well as the group as a whole. For example, the *Collaborative Development Project* specifies that: "Every group will be supervised by a TA, through mandatory weekly meetings." This created a rigid approach to collaboration, where progress, accountability, and evaluation criteria are documented and explicitly verified on a strict schedule.

On the other hand, KAIST does not have this type of formal and mandatory reporting, instead shifting to a more implicit structure. While none of the courses require regular and mandatory meetings with an instructor, they instead ask the students to reach out for feedback in cases where it's required, as detailed in 4 of the 8 syllabi.

Team formation, while still predominantly consisting of student-selected teams, presented some differences. Only one of the KAIST (*Usable Security and Privacy*) and one of the TU Delft (*Research Project*) courses opted for a preference-first approach, where students chose the topics they would prefer to work on from a set list instead of prioritizing the teammates they could pair with. In these two courses, students still had the option to select the same topics as their peers, but in the case of TU Delft, the selection leans more towards the preference side as it aims to offer good topics for each student, while in the case of KAIST, the first-come first-serve method means that students can still get placed with specific classmates. In addition to this, two courses at TU Delft have fully randomized team assignments. illustrating a more formal setup aimed towards prioritizing efficiency and simulating real conditions. KAIST had no such courses with fully randomized teammates.

A clear divergence was observed in terms of the accountability and impact of the individual compared to the entire group. TU Delft has 7 courses that focus on individual accountability. While this is heavily embedded into the assessment methods rather than the teaching methods, individual contributions have an impact on the grading metrics. In these 7 courses, the final grade can be influenced by individual technical contributions, the process, and collaborative performance throughout the project, which is evaluated through a combination of peer reviews and supervisor assessment. In contrast, KAIST has only 3 courses that explicitly account for individual contributions. Rather than utilizing supervisor monitoring throughout the project, these courses tracked individual output only on personal reflections and peer reviews, reflecting the importance of group relations and preserving the shared approach to group work.

The presence of capstone courses, meant to showcase the academic experiences gathered until then, deeply affects the way the curriculum is shaped as a whole. The capstone courses at TU Delft are predominated by explicit and task-oriented codes. They describe the deployment of version control:"Use Git to version and share source code contributions.", division of labor: "Organize and distribute tasks between team members", individual decomposition: "Break down software requirements into actionable tasks" and explicit role assignment: "Chair a meeting and take minutes". Even though these codes are most notably presented in the syllabus of *Collaborative Development Project*, they are also mirrored in the syllabi for *Software Project* and *Research Project*. General workflows, such as Agile/Scrum methods, public presentations, and collaboration tools or management software, appear in all three capstone courses.

In the case of KAIST, the curriculum lacks any formal capstone courses. Despite this absence, one of the courses mentions the use of GitHub. However, instead of functioning as a tool to supervise regular version control and contributions, it is used as a code repository for the verification of the produced content. Another course also explicitly notes that "teams should aim to follow the Agile process as closely as possible", indicating the presence of a baseline workflow despite the absence of fixed tracking.

Regarding the absence of written indicators, a notable difference appears in terms of their frequency. Several of the high-context indicators lack written mention in any of the syllabi. Codes such as `face-to-face meetings`, `synchronous work`, `pair programming`, `group opening`, or `live whiteboard` are fully absent

Subcategory	General	Low-Context Codes	High-Context Codes
Collaboration Model	team projects, project-based learning	individual decomposition, asynchronous teams, strict schedule	face-to-face meetings, synchronous work
Process & Roles	agile/scrum, team formation	explicit role assignment, defined goals	instructor input, group opening
Tools & Platforms	collaboration tools, management software	version control	live brainstorming, live presentation
Interaction & Comm.	peer/self assessment, group reflection, public presentation	division of labor, individual accountability, conflict protocol	pair programming, instructor mediation, consensus decision making

Table 1: Deductive Codebook for Sub-RQ1: Teamwork Methods

Subcategory	General	Low-Context Codes	High-Context Codes
Communication	interpersonal communication, negotiation	direct communication, clear instructions	implicit communication, indirect messaging, trust building
Collaboration	teamwork skills, group problem solving, collaborative design	individual contribution, personal responsibility	group harmony, shared goals, conflict avoidance
Skills Core	cs theory, engineering principles, leadership skill	individualist values, assertiveness, explicit comparisons	collectivist values, respect mindset, indirect feedback

Table 2: Deductive Codebook for Sub-RQ2: Leaning Goals

from the entirety of the KAIST syllabi. Within the case of TU Delft, explicit mentions of conflict protocols or asynchronous teams were also absent.

Cross-cultural code overlap was also minimal between the two cases. Only 2 of the courses at KAIST contained the low-context code defined goals, presented through specific, intermediate assessment points that students must gradually build towards. In contrast, 5 of the courses from TU Delft contained an overlap with the high-context code pair programming in regards to how students should approach their lab work.

Regarding the general baseline codes, team projects appeared across all of the combined 22 syllabi. Conversely, the least frequent baselines were project-based learning, appearing in only 2 TU Delft syllabi, and group reflection, appearing in only 3 KAIST syllabi. This lower presence can be attributed to overlap within the course documentation, where more specific or explicit codes took precedence.

5.2 Learning goals

The clearest result in terms of learning goals is the prevalence of missing explicit teamwork learning goals across both universities. KAIST has only 2 of the 8 analyzed courses that explicitly mention teamwork as a formal learning goal. However, they rely on abstract and holistic wording, such as "Propose and develop a team project" and "Develop and communicate design or research ideas collaboratively", being classified under general codes.

Similarly, at TU Delft, explicit teamwork goals are absent from 11 of the 14 syllabi, appearing within the capstone courses of *Collaborative Development Project* and *Software Project*, as well as *Human Computer Interaction*. While mirroring KAIST in the majority of the teamwork learning goals aligning with the general codes, the concentration is much larger within the capstone courses.

A notable difference arises in the cultural context-specific codes present within these courses. While KAIST presents no codes from either cultural context in either courses, TU Delft incorporates specific collaboration based low-context codes with individual contribution and personal responsibility being mentioned ("Chair a meeting and take minutes", "Organize and distribute tasks between team members"). This reflects the splitting of work within a professional workflow. In contrast, the course *Collaborative Development Project* also has an overlap with high-context indicators, as group harmony is represented through the encouragement of students to "Make agreements on how to work in a team".

The widespread omission of teamwork learning goals is driven by an explicit prioritization of the Skills Core over collaboration. Across all 22 of the syllabi analyzed, learning goals were dominated by the codes cs theory or engineering principles. In the presented texts, teamwork is rather used as a method of delivering a more complex technical product rather than being treated as a standalone skill.

As a consequence, the deliverables mentioned within these

Pedagogical Dimension	Low-Context Culture (TU Delft)	High-Context Culture (KAIST)	Cross-Case Comparison
Sub-RQ1: Teamwork Methods	Rigid schedules and monitoring, as well as explicit individual tracking within capstone courses	Student-initiated reporting and a higher concentration of missing explicit teaching methods	Both share a technical baseline, but the methods diverge with the presence of additional low-context specific methods
Sub-RQ2: Learning Goals	Highly focused goals related to individual contribution and computer science-specific goals	Holistic learning goals and a heavier reliance on implicit group alignment for collaborative work	Prevalent absence of explicit teamwork goals for both universities, with deliverables and assessment methods acting as implicit goals
Inductive Findings	Rigid templates for syllabi using more formal descriptive language	Professor autonomy for syllabi construction, with high variance in availability, structure, and formality of language	Structure and language use directly reflect the low-context direct communication and high-context implicit trust

Table 3: Cross-Case Comparison Matrix of Teamwork Pedagogy Across Cultural Contexts

syllabi also act as reflections of implicit learning goals. At TU Delft, even though the majority of courses still have missing explicit information on which aspects will be graded, the predominance of the `Skills Core` general codes is observed yet again, with only peer feedback being the most common mention of teamwork impacting assessment. The main exception is *Collaborative Development Project*, which describes an individual grading component, directly correlating to the focus on `individual contribution` and `personal responsibility` identified within its formal learning goals.

In contrast, the courses at KAIST feature highly explicit parameters regarding peer evaluation and diverse group deliverables. Out of the 8 courses, 5 contain explicit mentions of peer reviews, individual task grading, or collaborative components outside of the main software project. Specifically, 3 of these 5 courses (*Introduction to Software Engineering*, *Artificial Intelligence Based Software Engineering*, and *Introduction to Social Computing*) mandate individual deliverables that partially incorporate peer feedback and track `individual contributions`. While this introduces low-context tendencies into the curriculum, courses like *Introduction to Social Computing* relegate these tracking metrics to a comparatively small percentage of the final grade. Furthermore, 3 of these 5 courses (*Introduction to Social Computing*, *Introduction to Human Computer Interaction*, and *Usable Security and Privacy*) feature alternative assessment blocks consisting of teamwork-centric tasks, such as in-class group activities, studio participation, or collaborative user studies. These structures contribute significantly to general codes while implicitly reinforcing the value of `shared goals`

In contrast, at KAIST, 5 of the 8 courses have explicit mentions of peer reviews, individual deliverables, or other aspects pertaining to group work besides projects. Specifically, 3 of the 5 courses (*Introduction to Software Engineering*, *Artificial Intelligence Based Software Engineering*, *Introduction to Social Computing*) have individual deliverables that incorporate peer feedback and track `individual contributions`. This introduces low-context tendencies. However, *Introduction to Social Computing* has these metrics as a comparatively small percentage of the final grade. Furthermore, 3 of these 5 courses (*Introduction to Social Computing*, *Introduction to Human Computer Interaction*, *Usable Security and Privacy*) have additional assessment criteria besides the project. These consist of teamwork-centric tasks, such as in-class group activities, studio participation, or user study participation, contributing to both general codes while also introducing the value of `shared goals`.

5.3 Inductive Findings

Beyond the findings related to the explicit analysis of the data, other findings related to the structure and document design of the syllabi. The layout, standardization, and execution of the study guides across both universities reflect the high and low context divide.

In terms of formal structure, TU Delft enforces a very rigid structure of the syllabus that needs to be implemented. Each syllabus contains separate sections detailing the course description, learning goals, teaching methods, assessment, and contact channels. Furthermore, the study guides can explicitly present circular dependencies such as expected prior knowledge, course prerequisites, or co-requisites.

In contrast, KAIST presents a more informal approach to the writing of each syllabus. Individual professors are responsible for writing the syllabus of their course, meaning the structure, level of detail, and layout of each syllabus is left to their discretion. This is evident in the structure and level of detail observed in each syllabus, with extremes being that some syllabi are entirely missing from the public course website, while others provide full grading rubrics for every assignment. These structures reflect the cultural context directly. The TU Delft aims for explicit instructions and strict layouts mirroring low-context practices, while the high-context structure of KAIST leans towards a framework based on trust in the teaching staff, offering autonomy and displaying implicit accountability.

Additionally, an analysis of the writing style of the syllabus reveals a divergence in how the learning goals are phrased. At KAIST, the phrasing of learning goals, particularly ones related to teamwork, is more commonly open-ended (“Develop

and communicate design or research ideas collaboratively") or conveyed in a less formal, more conversational manner ("We will try to also expand into various ML algorithms").

Conversely, the TU Delft has much more granular and descriptive goals, which can notably be seen for courses such as *Signal Processing*, *Research Project*, and *Collaborative Development Project*. This difference in style demonstrates that even for similar competencies, the low-context curriculum is more explicit to eliminate ambiguity, whereas the high-context curriculum relies on social alignment and contextual understanding to achieve those goals.

6 Discussion

This analysis investigated which aspects of high and low context cultures are most prevalent in how teamwork is taught in universities in different cultural contexts. In the following subsections, we discuss these findings in relation to the posed research questions.

6.1 Synthesis of Research Questions

The teaching of teamwork in undergraduate computer science curricula varies greatly depending on the cultural context in which the institution is placed. The approach used by low-context environments is direct and structurally supervised, placing a strong emphasis on individualistic values and concrete workflows. In contrast, high-context environments rely on a pedagogical system driven by interpersonal relationships, collective group cohesion, and relational trust. As a result, while both frameworks share a unified technical baseline, they diverge in their structural and administrative execution.

Sub-RQ 1: Which methods are used to fulfill teamwork learning goals in high-context and low-context environments? TU Delft exhibits a highly structured and direct approach to teaching teamwork. The enforcement of a strict schedule in conjunction with the presence of randomized teams and widespread incorporation of individual accountability reflects a low-context strategy to clarity in teaching teamwork through explicit communication and defined responsibilities. These methods require students to rely on formal workflows to collaborate rather than on pre-existing social relationships.

These findings are consistent with Hall's interpretations of low-context cultures, with information being communicated directly and social interactions being defined structures. The focus on individual contributions suggests that teamwork is primarily perceived as an optimized process, which leads students to be accustomed to performance indicators reflected in professional environments.

Despite this, the presence of methods such as pair programming indicates a deliberate focus on including collaborative elements to mitigate exaggerated individualism. However, these interactions are included as a requirement, suggesting that high-context practices can be incorporated into low-context settings. Since a cohesive group is not the base in a low-context setting, these practices are often formalized instead of allowing the environment to arise naturally from the group itself.

This presents a clear design focus to help graduates prepare for highly regulated tech environments, focusing on clarity and accountability. Previous research has shown that teams with a diverse cultural background frequently experience misunderstandings due to differences in communication [25]. Due to this, students taught in low-context settings can experience difficulties in environments that depend on interpersonal relationships and implicit communication. This indicates the need to prepare students to work in professional contexts with multiple communication styles in order to not cause friction.

In contrast, KAIST showcases an approach that prioritizes student autonomy and group cohesion. The absence of explicit oversight and the focus on student-led team formation relate to aspects associated with high-context cultures, as social relationships and common understanding play a larger role in collaboration.

The high density of `missing` codes present in the syllabi supports this interpretation. The lack of explicit documentation indicates that the rules related to collaboration are considered self-evident and do not require being formally written, instead of being interpreted as a lack of teamwork practices [29]. From the perspective of Hall, in high-context scenarios, overspecification may even be considered redundant and over-restrictive. As a result, students are made to develop collective responsibility and mutual trust, teamwork in this case leaning towards being a social aspect, promoting group cohesion and coordination without external intervention.

However, this approach also introduced a vulnerability, as there is no safety net for teams that cannot reach higher levels of collaboration. In the event of interpersonal friction or uneven work distribution, students often need to resolve these team issues internally, without the assistance of an instructor. This can isolate cultural minorities who are not accustomed to an approach based on implicit communication or do not share the same underlying assumption related to teamwork [9].

These findings suggest that while the high-context approach helps build trust and collective responsibility, it can not only risk having these habits misinterpreted as an overreliance on groups, but also may not provide enough support for students requiring explicit guidance. As a result, including autonomy and clear expectations could be essential to improve their assimilation into multicultural work environments.

Finally, the commonality of general codes for both universities showcases the shared baseline of computer science education. Although the core aspects of the technical information presented remain identical, the methods used to teach this information diverge between cultures. TU Delft makes use of explicit and individualized methods to ensure contributions, while KAIST relies on implicit dynamics and group autonomy to achieve the same targets.

While each approach produces strengths valued within its cultural context, the results suggest that the curricula should incorporate multi-context methods of collaboration to narrow the gaps created. This can help students develop skills required to approach both implicit and explicit methods of collaboration, reducing the competency gap found in previous research [3].

Sub-RQ 2: How does cultural context affect which learning goals are matched to learning strategies in undergraduate curricula? The results indicate that the cultural context also influences the way teamwork is conceptualized within the curriculum. The focus on teamwork as a learning goal is very low compared to the presence of collaborative activities, showing that teamwork functions as a method for creating a product. This suggests that collaboration is treated as a tool for working rather than as a standalone skill that needs explicit teaching.

This aligns with previous work showing that technical skills are prioritized over collaborative ones [20]. Similarly, assessment and feedback also focus on technical performance [30], indirectly shifting the focus of students onto product quality rather than on the process undergone. These studies, together with the findings of this paper, show that teamwork is implicitly expected to arise from group work despite collaboration requiring additional competencies.

As a result, culturally context-specific codes become less visible within the explicit learning goals than in the methods used, indicating that, while technical objectives are generally universal, the mechanisms used to achieve them are influenced by culture.

KAIST's total absence of explicit high and low-context codes reflects the implications of high-context communication. The treatment of interpersonal relationships, consensus building, and collaboration as implicit social norms rather than formal ones aligns with Hall's framework. As a result, the absence of teamwork goals should not be treated as a deficit in the curriculum, but as evidence that collaborative behavior forms naturally within each group.

In contrast, TU Delft reflects the low-context approach of defining responsibilities and behaviors by having the majority of the teamwork learning goals present within capstone courses. In these courses, collaboration is split into more concise tasks focusing on personal responsibility and individual contribution. The relational goals found, such as group harmony, are also formalized into explicit agreements. This suggests that collaboration becomes structured, reflecting the low-context tendencies to reduce ambiguity.

These differences support Hall's framework that cultural context shapes communication as well as the degree of explicitness in social relationships. Even though both institutions aim to teach teamwork, they diverge in terms of interpreting collaboration as a skill requiring a formal and rigid approach or as a naturally arising behavior.

The implicit learning goals found within the deliverables for each course offer a more detailed view of the culture-specific differences. Although both institutions deploy similar technical tools, the intended use varies considerably.

At TU Delft, the clear focus is on individual validation, with peer reviews and individual components relating again to an implicit goal of personal accountability. In this case, the collaborative platforms' version control utility is also used as a monitoring tool to measure individual key performance indicators, such as the number of commits and lines of code, which enforce a student's focus on personal contributions during teamwork.

KAIST instead places emphasis on collective activities such as studio and user study participation, as well as in-class group activities. These tools support shared goals through cohesion rather than acting as tools for monitoring. GitHub is used as a delivery method instead of a tracking tool, indicating that greater trust is placed in the team's dynamics. These findings suggest that the high-context system places greater importance on the collective outcome rather than the individual accountability prioritized by the low-context system.

These differences show that identical technologies used do not reflect identical values. Instead, the tools themselves become affected by the context-specific values in terms of responsibility and trust. This expands beyond Hall's framework by demonstrating that cultural context also shapes the meaning placed on each technical aspect.

An important finding is the mismatch between the explicit goals presented within curricula and the demands expected in the industry. The emphasis on the technical output rather than the collaborative process risks conveying the impression that teamwork is a less relevant aspect, rather than one integral to it. Previous research also shows that graduates experience difficulties due to collaboration being a central part of work compared to their studies [3, 26]. The findings in this paper show that these challenges can be emphasized by culture-specific assumptions present within the learning environment. Students from low-context systems can become reliant on explicit individual tracking, while those from high-context systems may struggle in environments focused on performance metrics and formal reporting.

When taken as a whole, the results show that teamwork requires deliberate teaching instead of being a skill that can be built individually. As computer science becomes increasingly globalized and multicultural, curricula can benefit from treating collaboration as an objective rather than a tool to use. Helping students reflect on communication styles, conflict management, and other culturally diverse approaches to teamwork will allow them to develop the skills needed for the professional context.

6.2 Structural and Textual Explicitness

While the explicit contents of the curricula have been analyzed, the structural and textual tone of the syllabi also reflect the cultural context, requiring a more holistic view. Within Hall's framework, a document is always culturally impacted by the method in which it is presented and transmitted. The degree of standardization and textual formality reveals overarching aspects that would otherwise be missed. Thus, investigating how an institution structures and words the syllabi establishes an additional baseline for communication that professors and students both interpret and reflect, shaping the implicit boundaries of collaboration.

For TU Delft, the rigid syllabus structure functions as a tool to mediate trust in the individual through explicit wording and standardized procedures. By enforcing regular subsections, the university reduces ambiguity for both instructors and students. The syllabus functions as a higher importance document, reflecting Hall's findings that low-context cultures rely on heavily centralized and explicit communication to ensure stability.

The standardized textual environment further suggests that teamwork is framed as a process supported by clearly defined procedures and documented expectations. Consequently, conflict resolution and collaboration are presented as problems that can be addressed through formal mechanisms rather than primarily through interpersonal negotiation. This interpretation aligns with the broader low-context emphasis on transparency and the reduction of uncertainty through explicit communication.

Conversely, the decentralized structure at KAIST offers a framework based on trust and autonomy. By offering professors discretion over the structure and content of their syllabi, the university relies on implicit competence and professional relationships with its faculty. Thus, the syllabus is treated as a flexible outline supported by personal interaction rather than formal documentation.

This interpretation is reinforced by the language used within the analyzed syllabi. Open-ended objectives and relatively informal wording contrast with the more explicit and standardized language observed at TU Delft. In accordance with Hall's framework, this suggests that the high-context environment places less emphasis on conveying all relevant information through text alone. Instead, classroom interactions and shared understandings become an important means through which expectations surrounding collaboration are communicated and negotiated.

This is additionally confirmed in the language used in the syllabi. Open-ended objectives and the informal language used contrast with the explicit language used at TU Delft. In accordance with Hall's framework, this reflects the low emphasis of the high-context environment on conveying important information only through text. Instead, it relies on social interaction and mutual agreements as a means to communicate the aspects related to collaboration.

These observations suggest that cultural context also influences how information is structured and communicated. The syllabi contain context-specific assumptions related to trust and the role of formal documentation. As a result, the curriculum related to teamwork also extends to the textual and organizational structures through which students interact with it. This highlights that differences in teamwork education can also arise from the implicit communication used by institutions, not just by the explicitly presented curricula.

6.3 Limitations

The primary limitation lies in the limited sample size in terms of the courses analyzed. While 22 combined syllabi were sufficient for cross-case comparison, the sample size remains limited, especially for the high-context scenario, where only 8 courses were analyzed compared to the 14 for the low-context setting.

Further affecting this comparison is the absence of formal capstone courses from the KAIST curriculum. Since TU Delft capstone courses contained the highest concentration of both explicit teaching methods and learning goals associated with teamwork, including context-specific codes, this difference limits the quality of the cross-case comparison. However, as noted in the discussion, this divergence is itself an indication of a high-context institution.

Another major limitation is the reliance on analyzing the written public syllabi of each university. The documented information captures the intended curriculum but does not account for how it's delivered in practice. This boundary is explicitly indicated by the prevalence of the `missing` code, though the absence of explicit instructions could also indicate a higher reliance on specific classroom practices.

Finally, in terms of institutional and geographical scope, a single university cannot serve as a representation for an entire nation or cultural context. The practices at TU Delft and KAIST could be heavily influenced by local external aspects, resulting in these findings not being directly generalizable to the broader cultural context of computer science programs.

6.4 Future Work

To build upon the current exploration done by this paper, the clearest path for future research should be improving the depth and modality of the data. The most vital next step to improve would be conducting semi-structured interviews with both faculty members and students. This expansion would allow the capture of additional implicit assumptions and unwritten learning goals that were flagged by the `missing` code, helping provide a more complete view of the omitted data.

Furthermore, future research could expand on the scope of how these culture-specific variables affect multi-context and multicultural environments. Investigating how teamwork is taught in international programs or institutions with diverse student bodies would provide an additional dimension of comparison.

Analysis of teamwork assessment methods in different cultural contexts would also be a relevant addition to future investigations. Seeing how individualistic metrics and collectivist team grades are represented in cultural contexts would yield additional insights into the cultural difference between approaches. Investigating how they affect the skills and readiness of students would also lead to greater findings on how computer science education can adapt to the globalized, cross-cultural industry.

7 Conclusion

This study investigated the cultural differences affecting teamwork pedagogy in undergraduate computer science education, mapped to the teaching methods and learning goals used. The findings indicate a definitive divide in how collaboration is split across separate cultural contexts.

The low-context system, investigated through TU Delft, utilizes explicit textual instruction, regular supervision, and rigid individualist accountability to manage group work. In contrast, the high-context case, represented through KAIST, instead relies on professor autonomy, holistic objectives, and implicit trust between students to achieve group cohesion.

A pervasive baseline was found across both cases, that explicit teamwork learning goals are omitted in favor of technical skills. This resulted in the collaborative skills being taught less explicitly, instead being reflected in the deliverables and rubrics of each course as implicit learning goals.

The contributions of this research provide insights for computer science education in a more globalized industry. By presenting how cultural aspects are ingrained into the structure of

the curricula, this work can provide a lens through which curriculum designers can evaluate their own programs in terms of demonstrating a single-context bias. These findings expose a gap in students' preparation, as there is negligible overlap targeted towards preparing undergraduate students for intercultural collaboration. By identifying the missing aspects, this work allows for future designs to deliberately integrate context-varied methods and goals to help in bridging the gap between the academic instruction they receive and the global workforce.

References

[1] 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. *SIGCSE 2024 - Proceedings of the 55th ACM Technical Symposium on Computer Science Education* 1 (March 2024), 74–80. https://doi.org/10.1145/3626252.3630857

[2] Carol M. Barnum. 2020. *Usability Testing Essentials: Ready, Set...Test!* (2nd ed.). Morgan Kaufmann, Burlington, MA.

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

[4] John Biggs. 1996. Enhancing teaching through constructive alignment. *Higher Education* 32 (Oct. 1996), 347–364.

[5] Ivana Bosnić and Igor Čavrak. 2019. Project work division in agile distributed student teams: who develops what?. In *Proceedings of the 14th International Conference on Global Software Engineering* (Montreal, Quebec, Canada) *(ICGSE '19)*. IEEE Press, Piscataway, NJ, USA, 152–161. https://doi.org/10.1109/ICGSE.2019.00038

[6] EFL Cafe. 2023. *High-Context and Low-Context Cultures: Impact on English Language Education-context-and-low-context-cultures-impact-on-english-language-education*. Technical Report. EFL Cafe. https://eflcafe.net/high-context-and-low-context-cultures-impact-on-english-language-education/.

[7] KAIST Community-Global Connection. 2026. Quick Facts. https://k-connect.kaist.ac.kr/nation_det/quick_facts

[8] Michelle Craig, Phill Conrad, Dylan Lynch, Natasha Lee, and Laura Anthony. 2018. Listening to early career software developers. *Journal of Computing Sciences in Colleges* 33 (April 2018), 138–149. Issue 4. https://doi.org/10.5555/3199572.3199591

[9] Isabella Graßl, Stephan Krusche, and Gordon Fraser. 2023. Diversity and Teamwork in Student Software Teams. *ACM International Conference Proceeding Series* 1 (June 2023), 110–119. https://doi.org/10.1145/3593663.3593687

[10] Rashina Hoda, Muhammad Ali Babar, Yogeshwar Shastri, and Humaa Yaqoob. 2017. Socio-Cultural Challenges in Global Software Engineering Education. *IEEE Transactions on Education* 60 (Aug. 2017), 173–182. Issue 3. https://doi.org/10.1109/TE.2016.2624742

[11] Paul Ilyusenko. 2024. Will Software Developers Return to the Office? Post-Pandemic Survey and Expert Opinion. https://www.scnsoft.com/software-development/will-software-developers-return-to-the-office

[12] D. A. Jameson. 1989. Book Reviews : Hidden Differences: Doing Business with the Japanese. Edward T. Hall and Mildred Reed Hall. Garden City, NY: 1987. Anchor Press/Doubleday 172 pages. Reviewed by Joel P. Bowman Western Michigan University. *In Proceedings of the Fourth international Workshop on Computing Education Research (ICER '08)* 26 (1989), 83–85. Issue 1. https://doi.org/10.1177/002194368902600107

[13] Dan Jiang, Bettina Dahl, and Xiangyun Du. 2023. A Systematic Review of Engineering Students in Intercultural Teamwork: Characteristics, Challenges, and Coping Strategies. *Education Sciences* 13, 6 (May 2023), 25 pages. https://doi.org/10.3390/educsci13060540

[14] KAIST. 2026. KAIST ERP. https://erp.kaist.ac.kr/com/lgin/SsoCtr/initExtPageWork.do?link=estblSubjt

[15] Kyong-Jee Kim and Curtis J. Bonk. 2002. Cross-cultural Comparisons of Online Collaboration. *Journal of Computer-Mediated Communication* 8, 1 (10 2002), JCMC814. https://doi.org/10.1111/j.1083-6101.2002.tb00163.x

[16] Wendy A. Lawrence-Fowler, Laura M. Grabowski, and Christine F. Reilly. 2015. Bridging the divide: Strategies for college to career readiness in computer science. *2015 IEEE Frontiers in Education Conference (FIE)* FIE 2015 (Oct. 2015), 1–8. https://doi.org/10.1109/FIE.2015.7344317

[17] Robert Lingard and Shan Barkataki. 2011. Teaching teamwork in engineering and computer science. In *Proceedings of the 2011 Frontiers in Education Conference (FIE)*. IEEE, Piscataway, NJ, USA, F1C–1–F1C–5. https://doi.org/10.1109/FIE.2011.6143000

[18] Daniel Moritz Marutschke, Patricia Brockmann, and Victor Kryssanov. 2024. Students' Perception of the Impact of Cultural Dimensions on Global Software Engineering. In *Proceedings of the 15th International Conference on E-Education, E-Business, E-Management and E-Learning (IC4e 2024)*. Association for Computing Machinery, New York, NY, USA, 154–159. https://doi.org/10.1145/3670013.3670067

[19] Daniel Moritz Marutschke, Victor V. Kryssanov, and Patricia Brockmann. 2022. Balanced, Unbalances, and One-Sided Distributed Teams - An Empirical View on

Global Software Engineering Education. *IEICE Transactions on Information and Systems* E105D (2022), 2–10. Issue 1. https://doi.org/10.1587/transinf.2021MPP0002

[20] Sharon Mason. 2020. Collaborative Learning in Computing Education: Faculty Perspectives and Practices. In *Proceedings of the 2020 ACM Conference on International Computing Education Research* (Virtual Event, New Zealand) *(ICER '20)*. Association for Computing Machinery, New York, NY, USA, 136–146. https://doi.org/10.1145/3372782.3406254

[21] Matthew B. Miles, A. Michael Huberman, and Johnny Salda na. 2019. *Qualitative Data Analysis: A Methods Sourcebook* (4 ed.). SAGE Publications Inc., Thousand Oaks, CA. 380 pages. https://study.sagepub.com/miles4e

[22] Delft University of Technology. 2026. Selection procedure. https://www.tudelft.nl/en/onderwijs/opleidingen/bachelors/computer-science-and-engineering/bachelor-of-computer-science-and-engineering/from-application-to-enrollment/selection-procedure

[23] Delft University of Technology. 2026. Studiegids — Study Guide / BSc Computer Science and Engineering. https://studiegids.tudelft.nl/opleidingen/study-guide/educations/13603

[24] Maria Paasivaara, Kelly Blincoe, Casper Lassenius, Daniela Damian, Jyoti Sheoran, Francis Harrison, Prashant Chhabra, Aminah Yussuf, and Veikko Isotalo. 2015. Learning Global Agile Software Engineering Using Same-Site and Cross-Site Teams. In *Proceedings of the 37th International Conference on Software Engineering*, Vol. 2. IEEE Computer Society, Washington, DC, USA, 285–294. https://doi.org/10.1109/ICSE.2015.157

[25] Vitaliy Popov, Omid Noroozi, Jennifer B. Barrett, Harm J.A. Biemans, Stephanie D. Teasley, Bert Slof, and Martin Mulder. 2014. Perceptions and experiences of, and outcomes for, university students in culturally diversified dyads in a computer-supported collaborative learning environment. *Computers in Human Behavior* 32 (March 2014), 186–200. https://doi.org/10.1016/J.CHB.2013.12.008

[26] Alex Radermacher and Gursimran Walia. 2013. Gaps between industry expectations and the abilities of graduates. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education* (Denver, Colorado, USA) *(SIGCSE '13)*. Association for Computing Machinery, New York, NY, USA, 525–530. https://doi.org/10.1145/2445196.2445351

[27] Jorge Cristancho Rodriguez, Aggrawal Sakhi, Devang Patel, and Magana J. Alejandra. 2025. Teamwork Interactions and Cultural Orientations of Software Development Teams. *International Journal of Engineering Education* 41 (2025), 1289–1307. Issue 5.

[28] 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 (March 2010), 473–492. Issue 4. https://doi.org/10.1007/s10212-010-0026-5

[29] Liu Tong and Tian Yuqing. 2020. Applying Hall's High Context and Low Context Cultures Model to Analysis the Implications of Cultural Differences on Functioning in Cross-cultural Groups. *Academic Journal of Humanities & Social Sciences* 3 (Sept. 2020), 128–133. https://doi.org/10.25236/AJHSS.2020.030813

[30] Rebecca Vivian, Katrina Falkner, and Nickolas Falkner. 2013. Analysing computer science students' teamwork role adoption in an online self-organised teamwork activity. In *Proceedings of the 13th Koli Calling International Conference on Computing Education Research* (Koli, Finland) *(Koli Calling '13)*. Association for Computing Machinery, New York, NY, USA, 105–114. https://doi.org/10.1145/2526968.2526980

[31] Xuan Yang and Youfu Wang. 2026. The impact of digital collaboration tools on inclusive leadership in multicultural teams in the context of global remote work: a psychological perspective on empathy, cohesion, and cross cultural communication. *Frontiers in Psychology* 17 (March 2026), 1–15. https://doi.org/10.3389/fpsyg.2026.1738857

[32] Robert K. Yin. 2017. *Case Study Research and Applications: Design and Methods* (6 ed.). SAGE Publications Inc., Thousand Oaks, CA. 352 pages.

A Selected course list

A.1 TU Delft

- Collaborative Development Project
- Computer Organization
- Software Project
- Software Engineering Methods
- Signal Processing
- Digital Systems
- Big Data Processing
- Data Mining
- Computational Intelligence
- Research Project
- Collaborative Artificial Intelligence
- Computer Security
- Algorithms for NP-Hard Problems
- Human Computer Interaction

[23]

A.2 KAIST

- Inroduction to Social Computing
- Introduction to Software Engineering
- Human Computer Interaction
- Usable Security and Privacy
- Diffusion and Flow Models
- Artificial Intelligence Based Software Engineering
- Computing and Platforms for Care and the Underprivileged
- Algorithms Design and Analysis for NP-Hard Problems

[14]