Mapping Competency Categories in Dutch Bachelor's Computer Science Curricula

Mihnea Nedelcu

Supervisor(s): Dr. Sole Pera, Merel Steenbergen

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: Mihnea Nedelcu
Final project course: CSE3000 Research Project
Thesis committee: Dr. Sole Pera, Merel Steenbergen, Dr. Masoud Mansoury

Abstract

Computer Science graduates are expected to develop both technical knowledge and broader professional competencies. This paper examines how these competencies are represented in the documented curricula of three Dutch Bachelor's Computer Science programmes: Delft University of Technology, Eindhoven University of Technology, and Leiden University. We apply document-based curriculum mapping across nine competency categories and three documentary levels: programme goals, curriculum structure, and visible assessment practices. The results show broad documentary visibility, especially at the curriculum-structure level, where all nine categories are explicitly visible in all three programmes. However, visibility does not imply curricular intensity. Technical foundations and problem solving are documented more consistently and frequently than broader competencies, while assessment evidence is less frequent and less explicit. The study shows where cross-level traceability is stronger or thinner, and highlights the need to distinguish documentary presence from systematic competency development.

1 Introduction

Computer Science graduates are expected to work in environments where technical knowledge alone is not sufficient. Collaboration, communication, ethical reasoning, professional responsibility, and awareness of societal impact are now recognized as essential for effective practice [5; 6]. Yet several studies show that early-career software developers often feel underprepared for these competencies. Craig et al. [5] report that newcomers struggle with teamwork and professional communication, while Garcia et al. [6] document a persistent academia-industry gap in non-technical skills and professional dispositions across countries. These gaps can affect project success, team dynamics, and the ability of computing systems to serve society responsibly, so it is important to examine how Computer Science programmes make them part of graduate preparation.

Because a bachelor's degree is meant to prepare graduates for professional practice, programme documents provide an important point of access to how this preparation is formally organised. This is especially relevant in the Dutch higher education context, where programmes accredited by the Dutch-Flemish Accreditation Organisation (NVAO) are required to define intended learning outcomes and show constructive alignment between those outcomes, the curriculum, and assessment [9]. If a competency is only stated as a broad outcome, it remains unclear whether the curriculum gives students structured opportunities to develop it. The question is therefore whether competency development can be traced from programme-level goals into curriculum structure and, where available, assessment documentation.

We examine this question through a document-based curriculum mapping study of three Dutch Bachelor's Computer Science programmes. Curriculum mapping is suitable for this purpose because it makes the relationships between educational goals, content, sequencing, and assessment explicit [2]. We apply this logic by comparing programme-level goals and intended learning outcomes with curriculum structure and visible assessment information.

The main research question is:

> How are selected competency categories represented across programme goals, curriculum structure, and visible assessment practices in Dutch Bachelor's Computer Science programmes?

This question is addressed through two subquestions:

1. Which selected competency categories are present in the programme-level goals and intended learning outcomes of Dutch Bachelor's Computer Science programmes?
2. How are these competency categories reflected in curriculum structure, such as course types, project courses, electives, and bachelor projects?

Visible assessment practices are included in the main alignment analysis, but not treated as a separate subquestion. Assessment evidence is interpreted as an indicator of visible documentary alignment, not as a necessary condition for the presence of a competency.

The contribution is a document-based, comparative mapping of selected Dutch Bachelor's Computer Science programmes. We use a coding framework informed by curriculum mapping theory [2], CS2023 dispositions [7], and prior work on non-technical competencies in computing [5; 6]. The framework covers nine competency categories, ranging from technical foundations to independent learning, across three levels of the documented curriculum: programme goals, curriculum structure, and visible assessment practices.

Our analysis is limited to what is visible in public curriculum documentation. It does not evaluate teaching quality, rank universities, or measure student learning outcomes. Instead, it identifies where competency development is explicit, implicit, absent, or undocumented, providing a basis for making Computer Science curricula more transparent.

The rest of the paper is structured as follows. Section 2 presents the background. Section 3 describes the methodology and the analytical framework. Section 4 reports the results of the curriculum mapping. Section 5 reflects on responsible research. Section 6 discusses the findings and limitations. Section 7 concludes the paper and suggests directions for future work.

2 Background and Related Work

Computer Science education increasingly frames graduate capability as more than the accumulation of technical knowledge. CS2023, the most recent international curriculum guideline, articulates this capability through knowledge, skills, and professional dispositions [7]. Dispositions matter because they separate being familiar with a topic, or applying it when prompted, from the inclination to engage with it independently in professional work. Since dispositions mature through repeated encounters in meaningful disciplinary contexts, CS2023 attaches them to knowledge areas rather than

isolated tasks [7]. Graduate capability therefore cannot be located in technical content alone, nor inferred from single course-level activities, but must be traced as a feature distributed across a programme.

Within the Netherlands, the accreditation context is set by the Dutch-Flemish Accreditation Organization (NVAO), which requires programmes to formulate intended learning outcomes and justify them against the programme's level, orientation, discipline, professional field, and international requirements [9]. Since NVAO prescribes the logic by which outcomes must be motivated rather than the substantive competencies a Computer Science programme should pursue, national accreditation expectations and discipline-specific guidance play complementary roles. Accreditation establishes that outcomes must be justified and aligned, while guidance such as CS2023 can supply the substantive vocabulary through which such justification is interpreted.

Empirical work suggests that preparing graduates for professional entry remains difficult, particularly for competencies beyond technical knowledge. Early-career software developers report difficulty with teamwork and professional communication once they begin development work [5], and a comparable academia–industry gap in non-technical skills and professional dispositions recurs internationally [6]. Efforts to connect academic programme design with workplace practice, including structured industry experience, confirm that this transition remains an active concern in computing education research [3]. A frequent explanation is that programmes expose students to professional competencies without deliberately scaffolding them, so that collaboration or communication may be encountered without being progressively developed [6]. Exposure to a competency and its visible, structured development are therefore distinct.

Collaboration offers a concrete illustration. Studies of group programming assignments in Dutch universities show that collaboration goals are not always aligned with how work is assessed, and that a shared product grade can obscure unequal individual contributions [1]. By contrast, teamwork becomes a clearer and more assessable objective when courses are deliberately organized around explicit roles, structured peer interaction, and process-focused assessment [10]. The presence of group work is therefore weak evidence that collaboration is supported or evaluated; stronger evidence requires signs of intentional design.

A terminological difficulty complicates the interpretation of such evidence. Terms including *soft skills*, *non-technical skills*, *professional skills*, and *professional dispositions* are applied inconsistently and are not interchangeable: skills denote learned behaviours that can be practised and assessed, whereas dispositions denote orientations that shape how graduates approach professional situations [6]. When curriculum documents do not clearly mark this distinction, a broader umbrella notion such as a *competency category* is analytically useful because it allows skill-based and dispositional evidence to be considered together without forcing documentary language into categories it does not itself observe.

Making curricular relationships explicit requires an appropriate method. Curriculum mapping surfaces connections among educational goals, content, sequencing, and assessment, so that a curriculum can be examined systematically [2]. Arafeh cautions that outcome mapping alone is insufficient, because a programme may declare appropriate outcomes while still exhibiting gaps in where content appears, how it is sequenced, and whether it is assessed [2]. This aligns with constructive alignment, under which meaningful learning depends on a coherent relationship between intended outcomes, learning activities, and assessment [4]. Together, these perspectives imply that curriculum analysis should attend to outcomes, curriculum structure, and assessment in combination.

Document-based comparison has precedent within computing education. Miedema et al. [8] compare curriculum recommendations, course syllabi, and industry needs in data systems education, identifying discrepancies between recommended content and what course documentation made visible. Such work shows that comparing documents against expected or recommended content can expose gaps that outcome statements would conceal. Curriculum documents therefore constitute meaningful evidence of how educational priorities are represented and communicated, while remaining distinct from enacted teaching and student experience.

3 Methodology

This study applies a document-based curriculum mapping approach to three Dutch Bachelor's Computer Science programmes. Evidence was coded using the nine predefined competency categories introduced in Section 3.1 and examined across three documentary levels: programme goals (L1), curriculum structure (L2), and visible assessment practices (L3). These levels correspond to the core elements of curriculum mapping and constructive alignment: what programmes intend students to learn, where that learning is embedded in the curriculum, and how it is made visible through assessment [2; 4].

For each programme, publicly available curriculum documents were collected (Section 3.2), segmented and coded for the nine competency categories at each level (Section 3.3), and aggregated into per-programme visibility matrices and frequency distributions (Section 3.4).

3.1 Analytical Framework

The analytical framework consists of nine competency categories derived from CS2023 [7], the NVAO accreditation framework [9], and the competency clusters most consistently identified in prior empirical work [1; 5; 6; 8; 10]. We used a deductive coding approach, in which the nine competency categories were defined before coding to allow the three programmes to be compared against the same externally grounded framework.

The set is intended to span the full range of a graduate profile entering the workforce: it includes the technical core (*technical foundations*, *problem solving*) as a baseline, *research skills* for the academic orientation expected in the Dutch context [9], *collaboration* and *communication* as the most consistently reported transition gaps [5; 6], *professional practice* and *ethics and societal impact* as competencies CS2023 treats as integral rather than optional [7], and *interdisciplinarity* and *independent learning*

Table 1: Competency categories used in the analytical framework.

Competency categories and working definitions
C1 Technical foundations [7; 9]. Core CS knowledge, including programming, algorithms, data structures, systems, and computing concepts.
C2 Problem solving [7; 9]. Analytical thinking, abstraction, decomposition, modelling, and solution development.
C3 Research skills [9]. Systematic inquiry, literature use, methodology, scientific reasoning, and thesis-related research practice.
C4 Collaboration [7; 1; 10]. Teamwork, shared responsibility, peer coordination, and collaborative technical work.
C5 Communication [5; 6; 7]. Written, oral, visual, and technical communication, including documentation and presentation.
C6 Professional practice [7; 9; 3]. Planning, project management, workplace-relevant conduct, responsibility, and professional behaviour.
C7 Ethics and societal impact [7; 9]. Ethical reasoning, societal awareness, responsible computing, and reflection on technological consequences.
C8 Interdisciplinarity [7; 9]. Connecting CS knowledge and methods to other domains, stakeholders, and application contexts.
C9 Independent learning [7; 9]. Self-directed study, reflection, autonomy, and continued development beyond the degree programme.

as capabilities both CS2023 and NVAO expect graduates to carry into broader contexts and beyond the programme [7; 9]. Table 1 defines each category. The framework thus combines an international disciplinary source, a national accreditation source, and empirical studies of the academia–industry gap, as no single source covers the full purpose of the study.

3.2 Programmes and Documents

Three Dutch Bachelor's Computer Science programmes were selected: Delft University of Technology (TU Delft), Eindhoven University of Technology (TU/e), and Leiden University. The selection is purposive rather than statistically representative, so the findings are not intended to generalise to all Dutch Computer Science programmes.

For each programme, documentation was collected from publicly available sources, including official programme websites, online study guides, course catalogues, and bachelor project or thesis documentation where available. Only documentation for the 2025–2026 academic year was included, so that the analysis refers to a single curriculum version. The document types collected and the curriculum levels they primarily inform are shown in Table 2. For each document, the programme name, document type, source URL, access date, and language were recorded. Documents were collected within a single time window to create a consistent snapshot of each programme's documented curriculum.

Where documentation was available in both Dutch and English, the original-language version was used as the primary source. The alternate version was consulted only when wording differed in meaning.

3.3 Coding Procedure

The basic coding unit is a document segment: any passage in the collected documentation that contains identifiable evidence for one or more competency categories. A segment may be as short as a single learning objective or as large as a course-description paragraph, depending on the smallest unit at which the evidence retains its meaning.

Table 2: Document types and curriculum levels used in the analysis.

Document type	Primary curriculum level informed
Programme descriptions and graduate profiles	L1: programme goals and intended learning outcomes.
Curriculum overviews and study guides	L2: curriculum structure, sequencing, and distribution of course types.
Course catalogue entries and course descriptions	L2: course-level curriculum structure; L3 when assessment information is included.
Bachelor project or thesis manuals	L2: capstone structure; L3: visible assessment practices.
Assessment descriptions, rubrics, or grading criteria	L3: visible assessment practices.

Each segment was coded along three dimensions: first, which competency category or categories it provides evidence for; second, the curriculum level at which the evidence appears, namely programme goals (L1), curriculum structure (L2), or visible assessment (L3); and third, the directness of the evidence.

Directness was classified as either explicit or implicit. Explicit evidence is present when a competency is named directly or described through a close synonym, so that terms such as "teamwork", "ethical reasoning", and "research methods", and their Dutch equivalents, qualify as explicit. Implicit evidence is present when the competency is not named directly but the described activity or outcome demonstrably requires it. Implicit coding was applied conservatively: a generic reference to a "project" is not coded as evidence for collaboration unless the description specifies collaborative activity. Each implicit code was accompanied by a brief written justification to support consistency across documents.

3.4 Aggregation

After coding, evidence was aggregated into programme-level summaries. At the course level, a competency was considered visible if at least one explicit segment was coded for it, or if multiple implicit segments converged on the same competency across the course documentation. A single borderline implicit segment was not treated as sufficient. At the programme level, evidence was aggregated into a 9×3 visibility matrix for each programme, where the rows represent the nine competency categories and the columns represent the three curriculum levels (L1, L2, L3).

Table 3: Visibility codes used in the programme-level matrices.

Code	Meaning
E	Explicit: at least one explicit source provides evidence for the competency at that level.
I	Implicit: evidence is implicit but sufficiently consistent across multiple segments/sources.
N	Not visible: documentation was available but contained no evidence for the competency at that level.
NA	Not available: the documentation needed to assess that level was missing or insufficient.

Each matrix cell was assigned one of four visibility codes, shown in Table 3. The NA code distinguished unavailable documentation from available documentation containing no

competency evidence. No matrix cells required this code in the final coding, but it remained part of the scheme to make the treatment of missing documentation explicit.

The programme-level matrices were used to compare evidence across documentary levels. L1 codes indicated whether each competency was visible in programme goals and intended learning outcomes, L2 codes indicated visibility in curriculum structure, and L3 codes indicated visibility in assessment-practice documentation. This made it possible to identify whether evidence for a competency appeared across multiple levels or was limited to one or two levels.

For L2 evidence, an additional curriculum-location analysis was performed. Courses containing L2 evidence were classified as compulsory or core courses (C), route-dependent required choices (R), electives or free-choice courses (E), project-based courses (P), or bachelor thesis or project components (B). These labels indicate where evidence was found and do not change the visibility code. When a competency appeared across every applicable location type for a programme, the location was recorded as *all*.

The frequency-distribution tables were calculated from complete course-by-competency grids for curriculum structure and assessment information. For each course entry, all nine competency categories were considered at the relevant level, with missing categories receiving N. As a result, each course contributes the same number of judgements per level. The percentages report the distribution of E, I, and N codes across these judgements, showing how widely each competency category appears across the analyzed documentation to contrast the visibility tables which only indicate presence.

4 Results

This section reports the empirical patterns produced by the curriculum mapping procedure. It shows how the selected competency categories appear across the documented curriculum and distinguishes between visibility at a curriculum level and frequency across individual curriculum and assessment entries.

Programme-level visibility

At the programme-goal level, most competency categories are explicitly visible across the three programmes. Six of the nine categories are coded as explicit for TU Delft, TU/e, and Leiden: technical foundations, problem solving, research skills, communication, ethics and societal impact, and independent learning.

The remaining three categories show more variation. Collaboration is explicit in TU Delft, not visible in TU/e, and implicit in Leiden. The Leiden code was treated as implicit because the programme goals refer to *"work being carried out alone or in a team"*, but do not directly name collaboration as a graduate competency. Professional practice is explicit in TU Delft, implicit in TU/e, and not visible in Leiden. For TU/e, the implicit code is based on wording about graduates being able to *"determine requirements for medium scale software projects"*, which points to software project practice without explicitly naming professional practice. Interdisciplinarity is implicit in TU Delft and Leiden, but not visible in TU/e. Table 4 gives the full visibility pattern.

Table 4: L1 visibility of competency categories in programme goals and intended learning outcomes.

Competency category	TU Delft	TU/e	Leiden
C1 Technical foundations	**E**	**E**	**E**
C2 Problem solving	**E**	**E**	**E**
C3 Research skills	**E**	**E**	**E**
C4 Collaboration	**E**	**N**	**I**
C5 Communication	**E**	**E**	**E**
C6 Professional practice	**E**	**I**	**N**
C7 Ethics and societal impact	**E**	**E**	**E**
C8 Interdisciplinarity	**I**	**N**	**I**
C9 Independent learning	**E**	**E**	**E**

Curriculum-structure visibility and location

At the curriculum-structure level, all nine competency categories are explicitly visible in all three programmes. This means that each category has at least one explicit source of evidence in the documented curriculum composed of course descriptions, curriculum components, project courses, electives, or bachelor project documentation.

The curriculum-location results add detail to this general visibility pattern. Problem solving and collaboration are the most widely distributed categories, appearing across all applicable course-location types in all three programmes. The other categories are also visible across multiple locations, but with more programme-specific distributions. Table 5 shows the full distribution of L2 evidence across curriculum-location types.

Table 5: Curriculum locations of L2 competency evidence.

Competency	TUD	TU/e	LEI
C1 Technical foundations	all	C,R,E,P	C,E,P
C2 Problem solving	all	all	all
C3 Research skills	all	C,R,E,B	C,E,B
C4 Collaboration	all	all	all
C5 Communication	C,P,B	all	all
C6 Professional practice	C,R,P,B	all	all
C7 Ethics/societal impact	all	C,E	C,E,B
C8 Interdisciplinarity	C,R,E,P	R,E	C,E
C9 Independent learning	all	all	C,E,B

Note. C = compulsory/core, R = route-dependent required choice, E = elective/free choice, P = project-based course, B = bachelor thesis/project.

Assessment-practice visibility

Assessment-practice visibility is less consistent than curriculum-structure visibility. TU Delft has explicit assessment-practice evidence for eight categories, with interdisciplinarity coded as not visible. TU/e has explicit assessment-practice evidence for collaboration and communication, while the remaining categories are coded as implicit. Leiden has explicit assessment-practice evidence for technical foundations, problem solving, research skills, collaboration, communication, and independent learning; professional practice and ethics and societal impact are implicit, while interdisciplinarity is not visible.

Table 6: L3 visibility of competency categories in visible assessment practices.

Competency category	TU Delft	TU/e	Leiden
C1 Technical foundations	**E**	**I**	**E**
C2 Problem solving	**E**	**I**	**E**
C3 Research skills	**E**	**I**	**E**
C4 Collaboration	**E**	**E**	**E**
C5 Communication	**E**	**E**	**E**
C6 Professional practice	**E**	**I**	**I**
C7 Ethics and societal impact	**E**	**I**	**I**
C8 Interdisciplinarity	**N**	**I**	**N**
C9 Independent learning	**E**	**I**	**E**

Frequency Distribution of Visibility Codes

The frequency distributions show that programme-level and curriculum-level presence does not mean that competencies appear equally often across the analyzed documentation. In curriculum-structure documentation, technical foundations has the highest explicit visibility in all three programmes: 100% for TU Delft, 95% for TU/e, and 87% for Leiden. Problem solving is also frequently visible, but often through a combination of explicit and implicit evidence.

Table 7: Frequency distribution of visibility codes for TU Delft. Competency names are defined in Table 1. N = 35

	Curriculum structure			Assessment practices		
Comp.	**E**	**I**	**N**	**E**	**I**	**N**
C1	**100%**	0%	0%	29%	**63%**	9%
C2	**60%**	40%	0%	3%	23%	**74%**
C3	9%	17%	**74%**	6%	3%	**91%**
C4	23%	9%	**69%**	20%	9%	**71%**
C5	11%	6%	**83%**	11%	9%	**80%**
C6	11%	20%	**69%**	9%	0%	**91%**
C7	20%	6%	**74%**	3%	3%	**94%**
C8	9%	11%	**80%**	0%	0%	**100%**
C9	14%	29%	**57%**	3%	14%	**83%**
Mean	29%	15%	**56%**	9%	14%	**77%**

The remaining categories are less frequent across course-level documentation. Research skills, collaboration, communication, professional practice, ethics and societal impact, interdisciplinarity, and independent learning all contain substantial shares of non-visible codes in at least one programme. This pattern is strongest for interdisciplinarity and ethics and societal impact, which have high non-visible percentages in the curriculum-structure grids.

Assessment-practice documentation shows lower explicit visibility than curriculum-structure documentation. The mean explicit percentages for curriculum structure are 29% for TU Delft, 20% for TU/e, and 24% for Leiden. For assessment practices, the corresponding explicit percentages are 9%, 2%, and 4%. Across the three programmes, assessment evidence is therefore more often implicit or not visible than explicit. Tables 7–9 show the full frequency distributions.

Table 8: Frequency distribution of visibility codes for TU Eindhoven. N = 37

	Curriculum structure			Assessment practices		
Competency	**E**	**I**	**N**	**E**	**I**	**N**
C1	**95%**	0%	5%	0%	**95%**	5%
C2	30%	**65%**	5%	0%	22%	**78%**
C3	5%	16%	**78%**	3%	5%	**92%**
C4	14%	8%	**78%**	5%	14%	**81%**
C5	14%	11%	**76%**	8%	14%	**78%**
C6	11%	11%	**78%**	0%	8%	**92%**
C7	3%	3%	**95%**	0%	3%	**97%**
C8	3%	5%	**92%**	0%	3%	**97%**
C9	5%	11%	**84%**	0%	8%	**92%**
Mean	20%	14%	**66%**	2%	19%	**79%**

Table 9: Frequency distribution of visibility codes for Leiden University. N = 45

	Curriculum structure			Assessment practices		
Comp.	**E**	**I**	**N**	**E**	**I**	**N**
C1	**87%**	0%	13%	16%	**67%**	17%
C2	31%	**56%**	13%	2%	24%	**76%**
C3	16%	13%	**71%**	4%	4%	**92%**
C4	16%	11%	**73%**	7%	24%	**69%**
C5	20%	13%	**67%**	7%	24%	**69%**
C6	11%	16%	**73%**	0%	9%	**91%**
C7	16%	7%	**78%**	0%	7%	**93%**
C8	4%	20%	**76%**	0%	0%	**100%**
C9	13%	9%	**78%**	2%	9%	**89%**
Mean	24%	16%	**60%**	4%	19%	**77%**

5 Responsible Research

This study analyses publicly available curriculum documents only. No human participants were involved, no personal data was collected, and no consent procedures were required. The three programmes are identified by name because institutional context is necessary for cross-programme comparison and because the analysed sources are public by design. The analysis does not rank programmes or make claims about teaching quality, staff conduct, or student experience. Observations about absent or weak evidence refer only to the documentation analysed.

Researcher positionality was also considered. The researcher is enrolled in the Bachelor's Computer Science programme at TU Delft, one of the three programmes under study. Prior familiarity with TU Delft's curriculum structure and terminology may have influenced how its documents were read relative to those of TU/e and Leiden. To mitigate this, the same competency definitions, coding procedure, and decision rules were applied across all three programmes, and TU Delft's documentation was treated with the same procedural distance as the other two cases.

AI tools were used during the writing process to support language editing, phrasing, and structural refinement, as well as for table formatting. All substantive decisions, interpretations, coding choices, and final revisions remained the author's responsibility.

Language interpretation formed a further source of possi-

ble bias. One of the three programmes publishes primary documentation predominantly in Dutch. Dutch documents were interpreted by the researcher, who is not a native speaker but has sufficient reading proficiency for academic and administrative texts. Where terminology was uncertain, an English version was consulted when available, and Dutch terms were interpreted against the competency definitions in Table 1.

Several steps were taken to support reproducibility. For each collected document, metadata was recorded, including programme name, document type, source URL, access date, language, and academic year. Only documentation from the 2025–2026 academic year was included, and all sources were collected within a single time window. Coding was performed by a single coder, so subjective judgement cannot be fully eliminated, particularly for implicit evidence. To reduce this risk, implicit codes were accompanied by written justifications and checked against the predefined category definitions and explicit/implicit coding rules described in Section 3.3.

6 Discussion

In this section, we reflect on what the mapping shows about how the selected competencies are represented across the three programmes' documented curricula, and note the limitations of our methodology.

6.1 SQ1: Which selected competency categories are present in the programme-level goals and intended learning outcomes of Dutch Bachelor's Computer Science programmes?

Technical foundations, problem solving, research skills, communication, ethics and societal impact, and independent learning are explicitly present for all three universities, while collaboration and professional practice are explicit once, and interdisciplinary is explicit in none, yet appears implicitly in two. The consistently named categories are those that can be stated relatively directly as programme-level outcomes: disciplinary knowledge, analytical ability, research competence, communication, ethical awareness, and independent learning. By contrast, the unevenly visible categories depend more on how programmes describe practice-oriented work: collaboration as teamwork, professional practice as software-project responsibility, and interdisciplinary as cross-domain application. This partly overlaps with what the literature identifies as recurring tensions in the academia-industry gap, especially for collaboration and professional practice [6].

Where competencies are weakly coded, they appear absorbed under task or technical descriptions: collaboration as "work carried out alone or in a team" and professional practice as "determine requirements for medium-scale software projects". As such, competencies are treated as by-products of the technical work and are not conceptualized as goals themselves. Because programme goals are the anchor that curriculum and assessment align to [4], an unnamed competency makes cross-level alignment harder to trace. This offers one possible curriculum-documentation pattern related to the gap identified in prior work, without showing whether the competencies are deliberately developed in the enacted curriculum.

6.2 SQ2: How are these competency categories reflected in curriculum structure?

At the curriculum structure level, all nine categories are explicitly visible in every programme, yet they divide sharply on frequency. The technical core is widespread, with technical foundations being explicit in 87-100% of course entries, and problem solving in almost as many with implicit labels included. The seven broader competencies are present in the curriculum without being prevalent in it, with much smaller frequency rates, with explicit evidence in only single digits to the low twenties and coded not-visible in the majority.

This shared pattern is more important than the differences between programmes, although those differences still qualify the result. TU/e shows the clearest deviation: ethics and societal impact and interdisciplinary are almost absent from its documented curriculum structure, with up to 95% of judgements coded not-visible. TU Delft and Leiden show the same categories somewhat more often, though still not frequently. Other differences are harder to interpret. Independent learning, for example, is more visible at TU Delft than in the other two programmes, but mainly through implicit evidence, so this may reflect differences in documentation style or coding sensitivity rather than a clear curricular emphasis. Since course documentation varies in detail and terminology, these programme-level differences are best read as differences in documentary visibility. Miedema et al. also report variation in what course documentation makes visible [8]. The differences therefore modulate the shared pattern without overturning it.

The frequency distributions give documentary support to the distinction Garcia et al. draw between exposure and scaffolding [6]. Technical competencies appear repeatedly and explicitly, while broader competencies appear more thinly and often implicit. Collaboration illustrates this point. The curriculum location analysis shows collaboration across all course-location types, but the frequency tables show that it remains relatively infrequent. This suggests that group work is spread across the curriculum, but not frequently enough to show a sustained design principle. The stronger forms of evidence discussed in Section 2, such as explicit roles, structured peer interaction, and process-focused assessment, are precisely what infrequent and mostly implicit documentation cannot show [1; 10]. Thus, the curriculum-structure results indicate that these competencies are visible across the programmes, while providing more limited evidence of their systematic development within the documented curriculum.

6.3 RQ: How are selected competency categories represented across programme goals, curriculum structure, and visible assessment practices?

Across the three levels, the selected competencies are broadly and consistently aligned. The technical and analytical core is strongly represented in programme goals and curriculum structure, while the broader professional and human com-

petencies appear less consistently across levels and programmes. Alignment is weakest at the assessment level, and this is also where the programmes diverge most: of the nine categories, only collaboration and communication remain explicitly coded at assessment across all three (Table 6). These are also the only two competencies whose non-visibility holds roughly constant from curriculum structure to assessment, while for the other broader competencies it climbs sharply at this level, by roughly 15 to 25 points, with a smaller rise at Eindhoven, where the documentation is already largely not-visible. One possible reason is that the most detailed public assessment information is concentrated in project and thesis contexts, where collaboration and communication are more likely to be named and assessed, and where public documentation is usually the most frequent regarding assessment.

The thinner representation of the broader competencies reflects how they appear in curriculum documents. As noted in Section 6.1, these competencies are less consistently stated directly as programme-level outcomes, and require more attention to be scaffolded into a course in a structured way than technical content, so they surface less often and less consistently across the documentation. Where they do appear, the documentation reveals the pattern but cannot establish clear intent to develop them. Infrequent assessment evidence furthers this point. This ties back to Section 6.2, where collaboration was seen to become a clearer and more assessable objective only when a course is deliberately organized around it; the broad but infrequent evidence seen here is what the absence of such design hints to leave behind.

Read across all three levels together, the broader competencies are represented unevenly throughout the documented curriculum, especially when compared with the more stable technical core. This offers a curriculum-documentation perspective on the academia-industry gap that motivated the study [5; 6]: where that earlier work observes the gap in early-career developers and in industry expectations, the present mapping shows how some broader competencies are named, distributed, and assessed less consistently than technical competencies. The consistent thread across the three programmes is therefore layered: technical foundations and problem solving show the strongest cross-level stability; collaboration and communication become most clearly aligned when attached to assessed project work, even though collaboration is unevenly framed in programme goals; and the remaining broader competencies are more irregularly documented as the analysis moves from goals and curriculum structure toward assessment.

6.4 Limitations

Several methodological limitations qualify these findings, concerning the sample and time frame, the analytical framework, and the documentary basis.

The three programmes were selected purposively rather than as a statistical sample, so the findings do not generalize to Dutch Computer Science programmes as a whole. Our analysis also covers only the 2025–2026 documentation, so the snapshot may not reflect more recent or planned revisions, and a single year cannot show how the curriculum changes over time or how competencies progress across a degree.

The competency categories were derived deductively from existing frameworks without an inductive pass, so any competency outside these categories is not captured. The boundaries and granularity of the categories also shape the frequency results, so the observed patterns partly reflect how the categories were defined.

The study draws only on published documentation, an incomplete record of the curriculum: the material that most directly shows how competencies are developed and assessed is often not public, so the picture is thinnest at the assessment level. Programmes also differ in how much they publish publicly, so some apparent differences between them reflect documentation practice rather than the curriculum itself. The documents, finally, show how competencies are written into the curriculum, not how they are taught or learned.

7 Conclusions and Future Work

We examined how selected competency categories, spanning the technical core and broader professional competencies, are represented across the documented curricula of three Dutch Bachelor's Computer Science programmes. To do this, we mapped public curriculum documentation from TU Delft, TU Eindhoven, and Leiden University across three levels: programme goals, curriculum structure, and visible assessment practices.

Our central finding is that the selected competencies are broadly visible, but with different levels of documentary strength. Six of the nine categories are explicitly named in the goals of all three programmes, and all nine are explicitly visible at the curriculum-structure level. However, the frequency patterns show a much sharper division. Technical foundations dominates the course-level documentation, appearing explicitly in 87–100% of course entries, while the broader competencies usually appear in much smaller shares. Assessment documentation narrows the pattern further: only collaboration and communication remain explicit across all three programmes at the assessment level. The documented curricula therefore show broad cross-level traceability, but this traceability varies strongly in intensity. Collaboration and communication are the broader competencies with the most stable visible assessment evidence, plausibly because they are made more visible through project-oriented assessment documentation. Several other broader competencies become less consistently visible, especially at the assessment-documentation level.

The contribution of this study is to make this unevenness visible as a curriculum-documentation pattern. Earlier work identifies broader professional competencies as a persistent point of tension between university preparation and professional practice. Our mapping shows one way this tension can appear in curriculum documentation before graduation: broader competencies may be included in curriculum documents, yet appear sparsely across courses and less frequently or less explicitly in assessment descriptions. For curriculum design, this suggests that programmes should look beyond whether a competency is mentioned and examine its distribution, explicitness, and assessment visibility across the de-

gree. For curriculum analysis, it shows that visible alignment should be read not only through presence across levels, but also through the frequency and strength of the evidence at each level.

These conclusions remain limited to the scope of the study. We analysed three purposively selected programmes, one academic year, and public documentation only; the coding was deductive and conducted by a single researcher. The findings therefore describe how competencies are written into curricula, not how they are taught, practised, or learned. Future work should examine course-level materials more systematically, including non-public assignments, rubrics, project briefs, lecture materials, and detailed assessment criteria. Interviews with teachers, students, and curriculum coordinators could further clarify how documented competencies are enacted in practice. Extending the analysis to more programmes and multiple academic years would also show whether these documentary patterns are specific to the selected cases or reflect broader developments in Dutch Computer Science education.

References

[1] Efthimia Aivaloglou and Anna van der Meulen. An empirical study of students' perceptions on the setup and grading of group programming assignments. *ACM Transactions on Computing Education (TOCE)*, 21(3):1–22, 2021. doi: 10.1145/3440994.

[2] Sousan Arafeh. Curriculum mapping in higher education: a case study and proposed content scope and sequence mapping tool. *Journal of Further and Higher Education*, 40(5):585–611, 2016. doi: 10.1080/0309877X.2014.1000278.

[3] Nimmi Arunachalam, Stephanie J Lunn, Mark Weiss, Jason Liu, and Giri Narasimhan. 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*, pages 74–80, 2024. doi: 10.1145/3626252.3630857.

[4] John Biggs. Enhancing teaching through constructive alignment. *Higher education*, 32(3):347–364, 1996. doi: 10.1007/BF00138871.

[5] Michelle Craig, Phill Conrad, Dylan Lynch, Natasha Lee, and Laura Anthony. Listening to early career software developers. *J. Comput. Sci. Coll.*, 33(4):138–149, 2018. doi: 10.5555/3199572.3199591.

[6] Rita Garcia, Andrew Csizmadia, Janice L Pearce, Bedour Alshaigy, Olga Glebova, Brian Harrington, Konstantinos Liaskos, Stephanie J Lunn, Bonnie Mackellar, Usman Nasir, et al. An international examination of non-technical skills and professional dispositions in computing–identifying the present day academia-industry gap. In *2024 Working Group Reports on Innovation and Technology in Computer Science Education*, pages 124–174. 2025. doi:10.1145/3689187.3709610.

[7] Amruth N Kumar, Rajendra K Raj, Sherif G Aly, Monica D Anderson, Brett A Becker, Richard L Blumenthal, Eric Eaton, Susan L Epstein, Michael Goldweber, Pankaj Jalote, et al. Computer science curricula 2023, 2024. doi: 10.1145/3664191.

[8] Daphne Miedema, Toni Taipalus, Vangel V Ajanovski, Abdussalam Alawini, Martin Goodfellow, Michael Liut, Svetlana Peltsverger, and Tiffany Young. Data systems education: Curriculum recommendations, course syllabi, and industry needs. In *2024 Working Group Reports on Innovation and Technology in Computer Science Education*, pages 95–123. 2025. doi: 10.1145/3689187.3709609.

[9] NVAO. Assessment framework for the higher education accreditation system of the Netherlands, 2018.

[10] Pilar Sancho-Thomas, Rubén Fuentes-Fernández, and Baltasar Fernández-Manjón. Learning teamwork skills in university programming courses. *Computers & Education*, 53(2):517–531, 2009. doi: 10.1016/j.compedu.2009.03.010.