Content and Competencies for Interdisciplinary Teaching of Nonfunctional Software Requirements and User Experience in Computing Courses: An Asset Mapping of Curricular References Guidelines

Josué Viana Ferreira[1], Sandro Ronaldo Bezerra Oliveira[1,*], and Carlos dos Santos Portela[2]

[1]Graduate Program in Computer Science (PPGCC), Federal University of Pará (UFPA), Brazil
[2]Faculty of Information System, Cametá Campus, Federal University of Pará (UFPA), Brazil
Email: josue.ferreira@icen.ufpa.br (J.V.F.); srbo@ufpa.br (S.R.B.O.); csp@ufpa.br (C.D.S.P.)
*Corresponding author
Manuscript received June 16, 2025; revised July 31, 2025; accepted September 1, 2025; published January 9, 2026

***Abstract*—Nonfunctional Requirements (NFRs) and User Experience (UX) are essential for developing robust, effective and user-centred software systems. However, these aspects are often addressed in a fragmented manner within Computing curricula, typically confined to specific subjects such as Software Engineering (SE) and Human-Computer Interaction (HCI). This article aims to map the content and competencies related to NFRs and UX across four key computing curriculum frameworks: the Brazilian Computing Curricula Guidelines (RF-CC-2017), the ACM/IEEE Computing Curricula 2020 (CC2020), the Software Engineering Body of Knowledge (SWEBOK v4.0) and the Software Engineering Competency Model (SWECOM 1.0). Using a structured mapping methodology supported by expert validation, the study reveals conceptual and educational similarities between the analyzed documents. The results suggest significant opportunities for interdisciplinary integration between SE and HCI in Computing education, emphasizing content alignment, practical competencies, and shared design principles. The findings offer theoretical and practical contributions to curriculum design by proposing concrete guidelines for cohesive, interdisciplinary integration of NFRs and UX in Computing programmes.**

***Keywords*—nonfunctional requirements, user experience, software engineering, human-computer interaction, computing curricula, competency mapping**

I. Introduction

Nonfunctional Requirements (NFRs) and User Experience (UX) are essential for developing robust, effective, and human-centred systems. These elements encompass attributes such as usability, accessibility, performance, and suitability for the intended context. They are crucial for ensuring software quality and sustainability [1–3]. However, despite their theoretical and practical importance, content and competencies related to NFRs and UX are not systematically or comprehensively addressed in Software Engineering (SE) and Human-Computer Interaction (HCI) curricula. This may hinder the training of professionals capable of addressing contemporary software development challenges [4].

While NFRs ensure quality attributes such as performance and reliability, UX focuses on improving user interactions by considering emotional, cognitive, and sociocultural aspects [5]. International guidelines such as SWEBOK (Software Engineering Body of Knowledge) [6] recognize the importance of these areas for a comprehensive Computing education and recommend studying these subjects to develop a quality curriculum. In the Brazilian context, the Brazilian Computer Society's Training Reference Guideline (RF-CC-2017 from SBC) [7] reinforces this perspective by advocating the integration of content between different disciplines to improve the ability of courses to address complex issues.

Due to the multidisciplinary nature of software development teams, fostering interdisciplinarity between SE and HCI is a strategic advantage in professional training. Barbosa and Silva [8] and Lima *et al.* [9] emphasize the importance of integration between these two disciplines, as their complementary approaches are key to developing successful interactive systems. These authors argue that HCI content should be incorporated into NFRs to enhance technical quality and UX. Diniz *et al.* [10] reported on an interdisciplinary project that integrated SE and HCI. In this project, students produced documentation, models and prototypes for an application. This initiative promoted integrated learning and connections with professional practice.

The originality of this research lies in its focus on the intersection between NFRs and UX—a dimension absent from previous mappings—and in its use of the most up-to-date curricular references to propose an unprecedented interdisciplinary perspective on SE and HCI education. Further details on originality will be discussed in Section III. The aim of this study is to map the assets (content and competencies) related to NFRs and UX in the main Computing curriculum references: RF-CC-2017 [7], Computing Curricula 2020 (CC2020) [11], Software Engineering Competency Model (SWECOM 1.0) [12], and Software Engineering Body of Knowledge (SWEBOK v4.0) [6]. The study seeks to identify conceptual and formative convergences that support an interdisciplinary curricular proposal. Using a structured, expert-validated mapping methodology based on reference documents, the study aims to address gaps in the literature and assist SE and HCI curriculum coordinators and professors in delivering training that better aligns with the requirements for developing high-quality, user-centred systems.

In addition to this section, the remainder of the study is structured as follows. Section II presents the theoretical foundation and addresses the SWEBOK v4.0, SWECOM 1.0, SBC's RF-CC-2017 and ACM/IEEE's CC2020 curriculum references. Section III presents works related to the scope of

doi: 10.18178/ijiet.2026.16.1.2478

the study. Section IV details the research methodology. Section V displays the results obtained from asset mapping and the peer review process used to validate the research. Section VI discusses the research results. Section VII recognizes threats to the validity of the study. Finally, Section VIII concludes the study.

II. Background

This section presents the theoretical framework underpinning the asset mapping conducted in this study, emphasizing the importance of an interdisciplinary approach that incorporates both NFRs and UX. The following references were considered: SWEBOK v4.0, SWECOM 1.0, CC2020, RF-CC-2017 from SBC and the National Curriculum Guidelines (DCNs) for Computer Science courses in Brazil [13]. These documents guide the design of curricula in this field and are widely recognized in the literature as references for defining content and developing competencies in Computing programmes.

In this regard, Kumar and Choppella [14] highlight SWEBOK 3.0 as a useful reference for structuring curricula and educational practices that develop essential SE competencies, thereby strengthening professional training. Raj *et al.* [15, 16] identify CC2020 as a milestone in advancing competency-based approaches in computing. They propose curricula integrating knowledge, skills, and dispositions that are aligned with accreditation standards and industry participation. These curricula focus on graduates' employability and technical and professional preparation. In line with ABET's (Brazilian Association of Labor Studies) student learning outcomes, the authors present an integrated interpretation of ABET and CC2020 to consolidate competency-based education. Impagliazzo *et al.* [17] reinforce the idea that CC2020 summarizes the current state of the field and sets directions for future curricula. In the SE context, they note that the SE2014 competencies, integrated into CC2020 and supported by SWECOM, are essential for such training. These competencies should be continuously refined through engagement with academic and professional communities, updating curricular guidelines and ensuring education aligns with contemporary demands in the software industry. Finally, RF-CC-2017 aims to provide Brazilian undergraduate Computer Science students with an integrated, flexible, and relevant education [18].

As the following subsections detail the four established curricular references (SWEBOK v4.0, SWECOM 1.0, CC2020, and RF-CC-2017) individually, we first present a summary of their asset structures. This overview, organized in Table 1, clarifies how each document organizes its content and competencies, and facilitates understanding of the asset mapping conducted in this study.

Table 1. Overview of key asset structures in SWEBOK v4.0, SWECOM 1.0, CC2020 and RF-CC-2017

Reference Document	Asset Structure	Description of Function in Curriculum
SWEBOK v4.0	Knowledge Area → Topics → Subtopics	Defines foundational Software Engineering knowledge and organizes content hierarchically. Focused on conceptual coverage of SE domains.
SWECOM 1.0	Skill Areas → Skills → Activities (5 levels)	Competency-based model detailing practical skills and proficiency levels for software engineers.
CC2020 (ACM/IEEE)	Courses → Knowledge Areas → Competencies → Elements of Foundation and Professional Knowledges	An international curricular guideline emphasizing competencies and application-oriented learning outcomes.
RF-CC-2017 (SBC)	Course → Training Axis → Derived Competence → Contents	A national reference for Brazilian Computing curricula that integrates DCNs 2016 and emphasizes competences aligned to local context.

A. SWEBOK Guide

SWEBOK defines the fundamental scope of knowledge in software engineering [6]. This study adopts the most recent version, SWEBOK v4.0, which organizes knowledge into Knowledge Areas. These are subdivided into Topics and Subtopics, representing the main foundations of the field, as shown in Table 2.

Table 2. SWEBOK v.4.0 assets

Asset	Definition
Knowledge Area	A characterization of Software Engineering content
Topic	Decomposition of a Knowledge Area into a content group
SubTopic	Breakdown of a Topic containing specific content

Topics related to NFRs and UX were selected for this work, given their importance in developing core competencies in SE and HCI. These topics cover essential aspects of developing robust, usable, and user-centered systems, addressing technical quality attributes and UX. This selection aligns with the CC2020 guidelines and RF-CC-2017 [7], thereby reinforcing its relevance in academic training.

B. SWECOM Competency Model

SWECOM [12] is a competency model for software engineers involved in the development and evolution of software systems. Based on the guidelines for software engineering courses [19] and SWEBOK v3.0 [20], SWECOM distinguishes between knowledge and competence, emphasizing the importance of developing practical capabilities relevant to the market.

This study focuses on the technical competencies categorized by subject area in SWECOM 1.0. Table 3 shows the model's structure and its components, which are divided into: Skill Areas, Skills, and Activities, organized into five levels of increasing proficiency.

According to Table 3, SWECOM is organized by skill area (e.g., software requirements), skills within these areas (e.g. software requirements elicitation), and activities within these skills (e.g., prototyping for requirements elicitation). These activities are specified at five levels of competence: (1) Technician, (2) Entry-Level Practitioner, (3) Practitioner, (4) Technical Leader, and (5) Senior Software Engineer.

In addition to the activities specified at the various competency levels, SWECOM [12] states that all software engineers should be able to instruct and mentor others in the methods, tools, and techniques used for these activities. For instance, an Entry-Level Practitioner may instruct or guide others on how to use configuration management tools for their activities. To this end, the SWECOM model uses the

following notations: Follow (F), Assist (A), Perform (P), Lead (L), and Create (C).

Table 3. SWECOM 1.0 assets

Asset	Definition
Skill Areas	Set of technical competencies
Skills	Skills are described in terms of activities
Activities	Activities are specified in 5 (five) competency levels: Technician; Entry-Level Practitioner; Practitioner; Technical Leader; and Senior Software Engineer

Thus, SWECOM is a valuable reference for professionals, managers and curriculum developers, guiding the development of competencies that align with contemporary demands in the software industry.

C. Computing Curricula 2020

Published by the Association for Computing Machinery (ACM) and the Institute of Electrical and Electronic Engineers (IEEE Computer Society), CC2020 is an update to the international curriculum guidelines for Bachelor's degrees in Computing. It replaces the CC2005 report [11].

CC2020 is aligned with the SWECOM 1.0 competency model and builds upon the competency definition set out in the report in order to structure the educational content of Computing courses.

For this study, the Computer Science course was selected because it focuses on disciplines directly related to the Fundamentals of Software Development, providing critical and practical training in SE [19]. This enables a correlation with SWEBOK to be established (see Section II.B).

As shown in Table 4, CC2020 is organized into four main components (assets): Courses, Knowledge Areas, Competencies, and Elements of Foundation and Professional Knowledge.

Table 4. CC2020 assets

Asset	Definition
Courses	Unit of study for a specific area
Knowledge Areas	Focus knowledge for a particular area of study
Competencies	Ability to demonstrate efficiency in a given area, job or function
Elements of Foundation and Professional Knowledges	Fundamental skills central to the individual

D. SBC Training Reference Guideline

The SBC Training Reference Guidelines (RF-CC-2017) [7] play a strategic role in guiding Computing courses in Brazil. Aligned with the 2016 DCNs [13] and based on a content and competency model, RF-CC-2017 engages in dialogue with SWECOM and CC2020. It also aligns with Brazilian guidelines and international reference documents in the Computing field.

This study analyzed the Computer Science course, which aims to train professionals capable of driving scientific and technological development [13]. The analysis focused on Software Engineering and Human-Computer Interaction content, specifically addressing NFR and UX competencies. These references organize content hierarchically into assets Courses, Training Axis, Derived Competences, and Content, as shown in Table 5.

The RF-CC-2017 guidelines are structured around twenty-five general and specific competencies and skills, as defined by the 2016 DCNs for graduates of Bachelor's degree courses in Computer Science—the subject of this study. These competencies are organized into seven training axes. Each axis corresponds to a macro competency comprising a set of competencies derived from the 2016 DCNs. The axes guide the development of skills that enable graduates to work in different Computing phases, from system conception to implementation, and to seek continuous professional development beyond advanced studies aimed at scientific and technological advancement.

Table 5. RF-CC-2017 assets

Asset	Definition
Course	Definition of expected graduate profile
Training Axis	Training graduates in generic competencies
Derived Competence	Defines development needs in line with specific content
Contents	Topic to be studied in order to develop the expected competences

The structure of a training axis comprises the following elements: a code to identify it, a title to name it, a description to contextualize the competence of the axis, the axis competence to express its main training objective, and a set of derived competences necessary to build this macro competence. General competences are indicated by the identifier CG, while those specific to the Computer Science course are identified by CE. Each derived competence consists of three sub-fields: a code comprising the letter "C" (for "competence"), followed by the axis code (from 1 to 7) and a sequential number; a classification corresponding to one of the six levels of the cognitive process of Bloom's Taxonomy Revised [21]; and content comprising a list of essential knowledge for developing the competence.

In this study, we chose to limit the analysis to Training Axis 2: Systems Development. While the contents and competences related to SE and HCI are recognized as cross-cutting in RF-CC-2017, they are also present in Axis 3 (Project Development), Axis 5 (Infrastructure Management), Axis 6 (Continuous and Autonomous Learning), and Axis 7 (Science, Technology and Innovation). However, the choice of Axis 2 is justified by its greater alignment with competences linked to NFRs and UX. This axis focuses more clearly and directly on competences related to the identification, analysis, and specification of NFRs, as well as UX design and evaluation practices. The training objective is also to equip graduates with the ability to develop robust and effective computer systems, including creating new solutions and adapting existing systems. The rationale for selecting this training axis is also supported by a study conducted by Ferreira *et al.* [4], which provided a diagnostic analysis of the teaching of NFRs and UX in Brazilian undergraduate Computer Science programmes. The study examined how these programmes align with reference documents from the top universities in the country according to the Folha University Ranking. The study used the System Development Training Axis as a basis to map content and competencies across Course Pedagogical Plans (PPCs), syllabuses, and teaching plans of Brazilian higher education institutions. This was done by considering the CC2020 curriculum, the SWEBOK 3.0 guide, and the RF-CC-2017.

Restricting the scope to Axis 2 thus enabled a more precise and coherent alignment with the study's objectives and an in-depth examination of NFRs and UX in the selected

curricular references.

E. Interdisciplinarity of NFRs and UX

The 2016 DCNs for Computer Science courses establish interdisciplinarity as an essential educational principle. The document states that curricula should be designed to include "*III—Ways of Implementing Interdisciplinarity*" [13], thereby overcoming disciplinary fragmentation and promoting knowledge integration.

These guidelines outline training requirements that are directly aligned with the themes of the NFRs and the UX. The expected results are that graduates will [13]: "*II—acquire a global and interdisciplinary vision of systems, understanding that this vision transcends the implementation details of the many components and the knowledge of application domains; V—act reflexively in the construction of computer systems, understanding their direct or indirect impact on people and society; and VI—create solutions, either individually or as part of a team, to complex problems characterized by relationships between knowledge and application domains*".

Based on the 2016 DCNs, the SBC's RF-CC-2017 emphasizes that certain content can be covered in more than one curricular unit, demonstrating its applicability in different contexts and at different levels of depth. The set of contents and subjects depends, above all, on the strategy adopted by each course to develop the expected competences.

In this context, the guidelines emphasize the need for robust, integrated training that aligns with the skills required by the technology sector, in which aspects of software quality and UX are playing an increasingly important role.

Fig. 1 therefore illustrates the relationship between the key elements employed in the asset mapping presented in this study. The Computer Science course is positioned at the centre as it interfaces with the RF-CC-2017 and CC2020 curricula, as well as the SWEBOK v4.0 and SWECOM 1.0 reference documents. As detailed in Section II.A, the knowledge areas were taken from the SWEBOK v4.0 guide. These documents were used to map content and competencies related to the NFRs and UX.

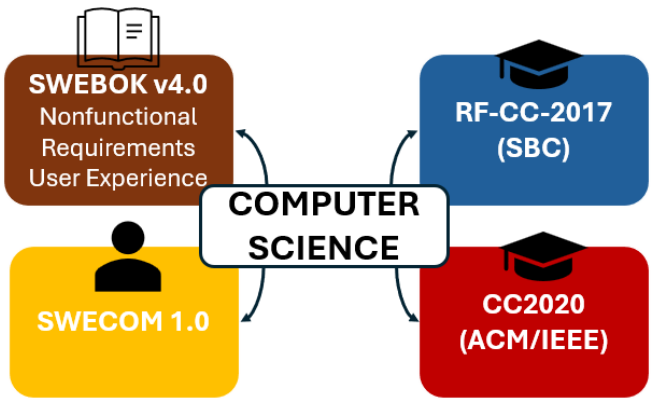

Fig. 1. Mapping core elements.

III. Related Works

Several studies have mapped content and competencies based on ACM/IEEE, SBC, and SWEBOK guidelines [22–25], applying them to various areas of computer science, such as Software Processes, Project Management, Software Requirements, and Software Design. This section presents studies that are methodologically related to the present research, highlighting their contributions, methodologies, and limitations compared to the approach developed here, which focuses specifically on NFRs and UX.

Castro and Oliveira [22] conducted a study focused on mapping content and competencies for teaching software design in Computer Science programmes. They established relationships among the CC2020 curriculum, RF-CC-2017, and the SWEBOK v3.0 guide. They combined document analysis with peer validation to identify detailed correspondences between software design topics in the three frameworks. The results revealed a strong correlation among the curricular assets, particularly between CC2020 and RF-CC-2017. This demonstrates that the SBC guidelines complement international recommendations by providing practical guidance on teaching design. The study also suggests that integrating these guidelines fosters the development of practice-oriented curricula that are aligned with the design competencies demanded by the market and current technological trends.

Colares *et al.* [23] mapped the content and competencies related to Software Process Improvement (SPI) within Computer Science programmes. They incorporated CC2013, RF-CC-2017, and SWEBOK v3.0, as well as the best practices of the Capability Maturity Model Integration for Development (CMMI-DEV v2.0) and the Brazilian Software Process Improvement Reference Model (MR-MPS-SW 2023). The methodology involved document analysis, structured asset mapping and peer validation by an experienced SE specialist. The study identified gaps in RF-CC-2017 relating to the absence of SPI-oriented content and competencies, proposing five content blocks derived from the asset mapping exercise. Supported by CMMI and MR-MPS-SW best practices, these blocks cover areas ranging from identifying process problems to assessing improvement effectiveness. They provide guidance for curricula aligned with industry demands, and national and international quality standards.

More recently, Carvalho *et al.* [24] conducted a study focusing on Software Project Management for Computer Science programmes based on CC2020, SWEBOK v3.0, and RF-CC-2017, followed by peer validation. The results highlighted the complementarity of the documents: SWEBOK v3.0 provides a robust technical foundation; CC2020 emphasizes collaborative and adaptive competencies; and RF-CC-2017 structures learning according to Bloom's Taxonomy, promoting critical and applied development. The study also demonstrated how CC2020 and RF-CC-2017 competencies can be achieved through SWEBOK v3.0 content.

Finally, Santos and Oliveira [25] used CC2020, RF-CC-2017, and SWEBOK v3.0 to map teaching resources related to Software Requirements. The study emphasizes the importance of incorporating consistent requirements topics into software engineering curricula, given that this phase is pivotal for successful software development. The methodology combined document analysis and peer validation, revealing a notable correlation between CC2020 and RF-CC-2017 with regard to competencies. Meanwhile, SWEBOK v3.0 provides additional essential detail for education in this area. The study recommends emphasizing

such content in national guidelines to strengthen student preparation.

A literature review revealed that some of the analyzed studies used outdated documents; for instance, Colares *et al.* [23] adopted CC2013. All of the studies also relied on SWEBOK v3.0. In contrast, the present research adopts the most recent version of SWEBOK (v4.0), enabling topics and subtopics relevant to RF-CC-2017 and CC2020 to be identified. This specifically addresses the subthemes of NFRs and UX. The SWECOM 1.0 model was used to map the competencies required to train professionals to address these subjects and enable an integrated analysis of the competencies proposed by SBC and ACM/IEEE.

It is important to emphasize the knowledge sub-areas selected for this research (NFRs and UX), since related studies do not directly address these sub-fields, despite their relevance to the current academic community. In this context, recent Computing education research has explored innovative approaches to developing UX competencies, emphasizing user-centred, hands-on experience. For example, Phothisonothai *et al.* [26] demonstrate that project-based learning initiatives bridge the gap between theory and practice, thereby increasing engagement and enabling objective evaluations of user experience, including usability and emotional response. Similarly, Nimkoompai *et al.* [27] demonstrate that hybrid strategies combining online self-learning, communities of practice, and applied projects can strengthen UX education and prepare students for real-world interactive design challenges in emerging curricula.

Regarding NFRs, SWEBOK v4.0 emphasizes the importance of avoiding vague and/or unverifiable requirements based on subjective judgments, such as "the software must be reliable" or "the software must be easy to use", as these can hinder understanding and the level of detail required. Li *et al.* [28] also highlight that NFRs, particularly those relating to performance, reliability, maintainability, usability, security, privacy, and customer satisfaction, are frequently overlooked in academic software projects. They found that 68.06% of analyzed projects ignored NFRs, with 76.39% overlooking security requirements. This reveals a significant deficiency in the systematic treatment of NFRs in SE education. This educational shortcoming reflects a wider professional issue: industry-focused research has shown that practitioners often undervalue NFRs compared to functional requirements. For instance, Ramos *et al.* [29] and Oliveira *et al.* [30] emphasize that overlooking NFRs during the requirements analysis phase can severely compromise software quality, often resulting in costly project failures. Together, these findings reveal a systemic and persistent underestimation of NFRs in both academic training and professional practice. This highlights the urgent need for curricular strategies that treat NFRs as core components of SE and HCI education. After all, these requirements directly impact UX [4].

In this context, Table 6 summarizes the main characteristics of the study in question and contrasts them with those of related work. This emphasizes the advances and distinctive aspects that highlight the originality of the research.

Table 6. A Comparison of this study with related works

Characteristics	[22]	[23]	[24]	[25]	This Study
Using of CC2013		X			
Using of CC2020	X		X	X	**X**
Using of RF-CC-2017	X	X	X	X	**X**
Using of SWEBOK v3.0	X	X	X	X	
Using of SWEBOK v4.0					**X**
Using of SWECOM 1.0					**X**
Knowledge sub-areas: NFRs and UX (in the SE and HCI)					**X**

IV. Materials and Methods

This section outlines the methodology adopted for asset mapping in this research, emphasizing the steps taken and outputs generated at each stage. To ensure methodological rigour, validity considerations were embedded throughout the mapping process. A structured sequence of steps, expert involvement, and transparent coding procedures were employed to enhance reliability and construct alignment. A detailed assessment of threats to validity is provided in Section VII. With this methodological foundation established, the mapping was organized according to the steps illustrated in Fig. 2.

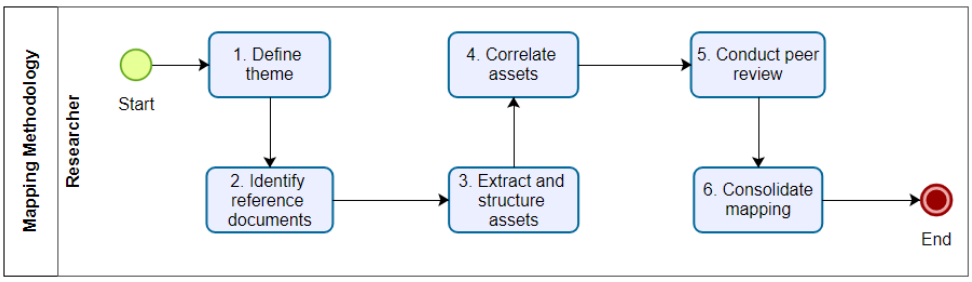

Fig. 2. Asset mapping methodology.

In Step 1, a preliminary literature review was conducted to define the scope of the mapping exercise. Studies addressing the teaching of NFRs and UX in the context of SE and HCI were analyzed. To support face validity, a previous diagnostic study [4] was considered. This study analyzed PPCs, teaching plans, and course content from Brazilian Computer Science programmes, revealing that NFR and UX content is present, albeit fragmented, within curricula. These findings reinforce the practical relevance of the present mapping exercise. Previous curriculum mapping initiatives [22, 24] were also used as a reference for the methodological approach to asset correlation. Based on this exploratory analysis, NFRs and UX were selected as the focus areas, both of which form part of the body of knowledge outlined in SWEBOK v4.0 [6].

Step 2 involved a mapping exercise based on four curricular references selected for their relevance to SE and HCI education: SWEBOK v4.0, SWECOM 1.0, CC2020,

and RF-CC-2017. These documents were then organized into core asset hierarchies (knowledge areas, content, skills, and competencies), providing the analytical basis for identifying and correlating NFR and UX topics across the references.

Step 3 involved studying and organizing the assets contained within the four selected reference documents in detail. The aim was to analyse the purpose and scope of each asset to identify potential conceptual links between the documents. To ensure the mapping remained focused on the study's core sub-areas, only assets directly related to NFRs and UX were considered. This stage established the analytical basis for correlating the content and competencies of the reference frameworks. To ensure reproducibility, explicit decision rules were established during the mapping process. An asset was included if it clearly mentioned attributes related to NFRs (e.g., performance, reliability, usability, accessibility, and security), or if it directly addressed user-centred methods, interface prototyping, evaluation, or design principles associated with UX. Assets whose relationship to NFRs or UX was deemed incompatible were planned by two researchers (the doctoral student and an expert in SE and HCI), with inclusion only occurring when consensus was reached. The alignment process was conducted via online meetings, which facilitated detailed discussion and resolution. Items outside these criteria, such as those dealing exclusively with project management or general software processes, were excluded to maintain focus on the scope of the study. To illustrate these relationships, the subtopic "Nonfunctional Requirements" in SWEBOK v4.0 was mapped to the "Software Engineering" content of RF-CC-2017, as well as to the "Requirements Analysis and Specification" and "Systems Analysis and Design" areas of CC2020, since it explicitly defines system-level constraints that impact quality. Similarly, the SWECOM 1.0 skill "Usability Testing and Evaluation" was correlated with the RF-CC-2017 competence "C.2.1" and the CC2020 "UX Design Knowledge Area", as it contributes directly to ensuring user-centred evaluation practices.

In Step 4, correspondence was established among the selected assets and explicit justifications were developed for each identified relationship. This process was guided by NFRs and UX. An experienced researcher provided support at this stage, helping to validate the conceptual consistency of the correlations. Additionally, a combined colour-coding and shape-encoding system was used to categorize and visually represent the relationships between the assets. Each colour and shape corresponded to a specific category: orange for content items, green for competency items, and gray for uncorrelated items. It is worth noting that an HCI specialist analyzed and validated the colour palette and graphical patterns used in all the figures in this study. The colour-coding system supported the construction of the asset mapping tree presented in Fig. 3. This tree highlights conceptual overlaps and gaps across the references, providing a clear visual synthesis of the interdisciplinary connections between SE and HCI in the context of NFRs and UX.

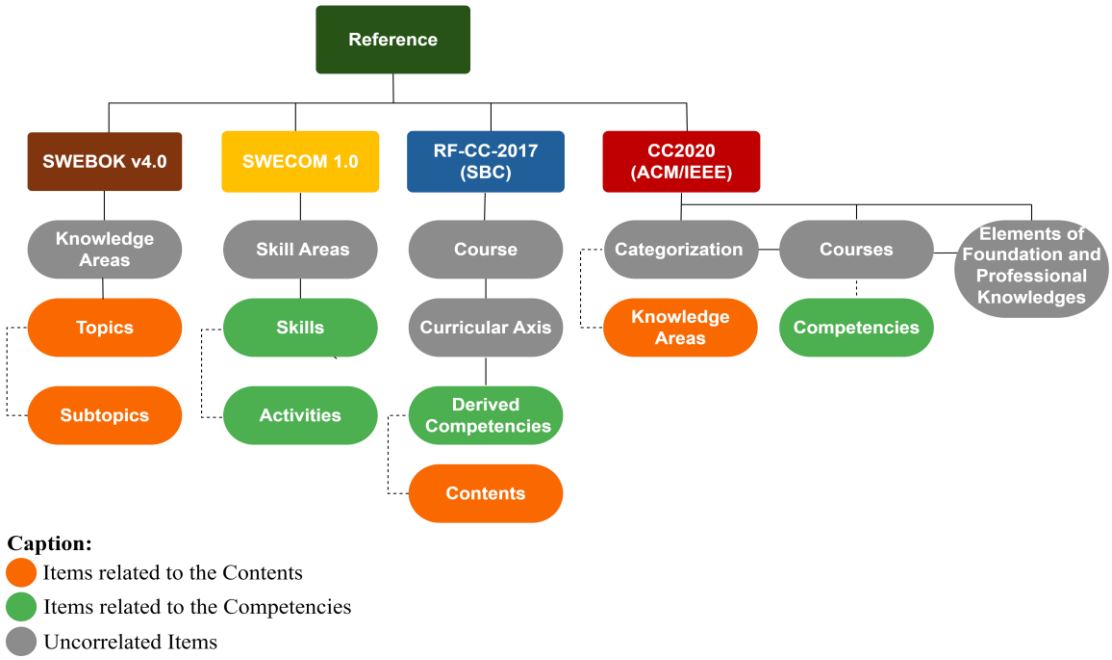

Fig. 3. Colour-coded asset mapping tree showing the relationships among SWEBOK v4.0, SWECOM 1.0, CC2020, and RF-CC-2017.

Step 5 corresponded to the peer review process, conducted by an expert with extensive experience in Software Engineering and a strong background in Computing education research. During this step, the mapping and respective justifications were presented in an online meeting, during which suggestions for improvement were discussed and recorded.

Finally, Step 6 involved consolidating the mapping and incorporating the contributions that had been identified during the review process outlined in Step 5.

V. Results

Fig. 3 shows the resulting tree structure from the asset mapping related to NFRs and UX. It highlights the correlations between SWEBOK v4.0, SWECOM 1.0, RF-CC-2017, and CC2020. This figure uses the same colour palette and coding patterns as the reference frameworks shown in Fig. 1. Brown is assigned to SWEBOK v4.0, yellow

to SWECOM 1.0, blue to RF-CC-2017 (SBC), and red to CC2020 (ACM/IEEE). All elements are represented as rounded rectangles. As detailed in Section IV, the colour coding also distinguishes between three categories of element: content-related items (orange), competency-related items (green), and uncorrelated items (gray). This visual convention ensures consistency and facilitates interpretation across all reference models.

The uncorrelated elements lie outside the scope of the NFR and UX content and competency mapping conducted in this study. These include the Knowledge Areas of SWEBOK v4.0, the Skill Areas of SWECOM 1.0, the Course and Training Axis of RF-CC-2017 and the Categorization, Courses, and Elements of Foundation and Professional Knowledge of CC2020. While these items are relevant in their original contexts, they were not included in the mapping as they are not directly applicable to NFRs and UX. Therefore, only elements that are explicitly associated with curricular content or learning outcomes in NFRs and UX were selected for correlation.

Therefore, the Subtopics derived from the Topics in SWEBOK v4.0 were linked to the Contents of RF-CC-2017 and the Knowledge Areas of CC2020. These formed the set of items that structured the content domain. Similarly, the Skills from SWECOM 1.0 were correlated with the Derived Competencies of RF-CC-2017 and the Competencies of CC2020, establishing clear links between practical abilities and training outcomes. The Activities in SWECOM 1.0, which detail the practical application of these skills, were also considered essential to the analysis, as they represent tasks that are directly related to SE and HCI, focusing on NFRs and UX.

In summary, the hierarchical organization implemented in the mapping allows for an integrated visualization of content, skills, and competencies across the different curricular references, revealing conceptual convergences, methodological complementarities, and strategic alignments for the interdisciplinary teaching of NFRs and UX.

A. SWEBOK v4.0 Asset Mapping

SWEBOK v4.0 is divided into 18 Knowledge Areas (KAs), each representing a specific SE domain. Each KA is organized hierarchically into Topics, Subtopics, and where necessary, Sub-Subtopics. It also includes references, further reading and cross-references that connect knowledge between different areas.

Table 7. Relationship between topics and subtopics in SWEBOK v4.0—Nonfunctional requirements

Topics	SubTopics
Software Requirements Fundamentals	Definition of a Software Requirement Nonfunctional Requirements Technology Constraints Quality of Service Constraints Why Categorize Requirements This Way?

Regarding NFRs, Software Requirements KA in SWEBOK v4.0 covers the subject in the 'Software Requirements Fundamentals' Topic, which is divided into the following Subtopics: Definition of a Software Requirement; Nonfunctional Requirements; Technology Constraints; Quality of Service Constraints; and Why Categorize Requirements This Way? Analyzing these contents enabled us to establish that these five Subtopics are directly aligned with the NFRs, offering a comprehensive overview of their definition, categorization and implications within SE. Table 7 shows the relationship between the Topics and Subtopics in SWEBOK relating to NFRs.

A description of each subtopic relating to the NFRs can be found below:

- **Definition of a Software Requirement:** A condition or capability that is necessary to solve a problem, achieve an objective, or ensure compliance with a formal standard. They represent the needs and constraints of various stakeholders, including customers, users, regulatory bodies, and the project team itself. Software requirements often address complex business processes, device controls, or improvements to existing systems, reflecting the intricate environments in which software operates.
- **Nonfunctional Requirements:** These requirements influence implementation decisions by defining factors such as the necessary computing platforms and database engines, the required level of accuracy, the speed of result presentation, and the storage capacity for specific types of records. They are divided into technology constraints and quality of service constraints, which are interconnected. Any modification to a nonfunctional requirement must consider its potential positive or negative impact on the others.
- **Technology Constraints:** These are nonfunctional requirements that mandate or prohibit the use of specific automation technologies or infrastructures. Examples include particular computing platforms (such as Windows, macOS, Android OS, or iOS), specific programming languages (such as Java, C++, C#, or Python), compatibility with certain web browsers (such as Chrome, Safari, or Edge), specific database engines (such as Oracle, SQL Server, or MySQL), and broader technologies (such as relational databases).
- **Quality of Service Constraints:** These define the acceptable performance levels that a software solution must achieve, without mandating the use of specific technologies. These include requirements relating to response time, throughput, accuracy, reliability, and scalability. Additionally, safety and security are critical yet often overlooked aspects that are essential to ensure the system operates properly and is protected.
- **Why Categorize Requirements This Way?** The categorization of requirements into functional and nonfunctional groups is a strategic approach that supports effective software engineering practices. As requirements from different categories often originate from different sources, various techniques must be applied to elicit, analyse, specify, and validate them. An important distinction is the 'Perfect Technology Filter': functional requirements would persist in a hypothetical environment with infinite computational resources, whereas nonfunctional requirements impose technological constraints. In large systems, nonfunctional requirements at a higher domain level can also give rise to functional requirements in related sub-domains. Security requirements in banking systems are an example of this.

Table 8 shows the relationship between SWEBOK v4.0 Topics and Subtopics aligned with UX. These topics are

included in Software Design KA, which covers basic concepts, contexts, processes, qualities of software design, strategies, and the recording and evaluation of designs. In the context of this research, UX is represented by three Topics and their respective Subtopics: "Software Design Strategies and Methods", associated with the subtopic "User-Centered Design"; "Test Levels", associated with the subtopic "Objectives of Testing"; and "Human Factors: User and Developer", associated with the subtopic "User Human Factors". Analysis of this content revealed that these three Subtopics directly promote a more effective UX by considering aspects such as the centrality of the user in the design process, the definition of testing objectives that consider interaction quality, and the incorporation of human factors in software development. These elements are fundamental to ensuring that the product meets users' needs, expectations, and limitations, while also ensuring that it is technically sound and user-friendly.

Table 8. Relationship between topics and subtopics in SWEBOK v4.0—User experience

Topics	SubTopics
Software Design Strategies and Methods	User-Centered Design
Test Levels	Objectives of Testing
Human Factors: User and Developer	User Human Factors

The description of the mapped UX topics is set out below:

- **User-Centered Design:** This multidisciplinary approach emphasizes gaining a deep understanding of users and their needs in order to design user experiences within the context of their organization and the tasks to be accomplished. The process involves gathering user requirements, creating a user flow of tasks, and decisions, designing prototypes or mock-ups of user interfaces and evaluating the design solution against the original requirements.
- **Objectives of Testing:** Software testing is conducted with explicitly stated objectives, which may vary in precision and focus. As well as verifying functional specifications, testing can assess nonfunctional properties such as performance, reliability, and usability. Other critical objectives include measuring system reliability, identifying security and privacy vulnerabilities, and evaluating usability. These require distinct approaches, each tailored to the specific testing goal.
- **User Human Factors:** Users expect software to be robust and secure, with intuitive graphical user interfaces (GUIs) that guide them through minimal, intelligent, and easy-to-follow steps to help them achieve their goals. The interface should facilitate self-learning, provide clear and complete messages, and support error recovery and interrupted operations where possible. Software engineers must understand user profiles, system functionalities, input and output interfaces, fault tolerance, and performance parameters. User interface development typically involves several iterative cycles starting with prototypes and requires robust interface devices.

B. SWECOM 1.0 Asset Mapping

In the SWECOM 1.0 asset mapping, which aligns with the objectives of this study, the target competency level is Entry-Level Practitioner or Early Career Professional. These levels are defined in SWECOM [12] and detailed in Section II.B. According to this reference, this profile corresponds to individuals who have completed higher education or have four to five years' equivalent practical experience in the field.

The document also states that software engineers should develop the ability to guide and support their colleagues in using methods, techniques, and tools related to professional practice appropriately, regardless of the technical focus of their role. In this context, the Entry-Level Practitioner is assigned the "Assist" (A) and "Perform" (P) classifications, which denote the ability to collaborate in executing tasks (A) and carrying them out under supervision (P), respectively. SWECOM's competencies related to NFRs and UX are presented below.

Table 9 shows the Skill Area, Skills, Activities, and Levels corresponding to the NFR-related competencies in the SWECOM model. The identified Skill Area is Software Requirements, which includes four specific Skills: Software Requirements Elicitation, Software Requirements Analysis, Software Requirements Specification, and Software Requirements Verification and Validation. These Skills are associated with five Activities that aim to: assist in stakeholder engagement (A); apply appropriate methods for requirements elicitation (A); perform domain analysis (A); prepare requirements documentation, including nonfunctional requirements (E); and review requirements specifications (P). These Activities are classified as Assistance (A) and Perform (P), placing them in the Basic Level Practitioner dimension.

Table 9. Skills related to nonfunctional requirements in SWECOM (entry-level)

Skill Areas	Skills	Activities	Level
Software Requirements	Software Requirements Elicitation	1. Assists in engaging different stakeholders to determine needs and requirements (A). 2. Assists in applying different methods to the project as appropriate to elicit requirements (A).	Assist (A)
	Software Requirements Analysis	1. Assists in domain analysis (A).	Assist (A)
	Software Requirements Specification	1. Prepares requirements documentation including descriptions of interfaces and functional and nonfunctional requirements (E).	Perform (P)
	Software Requirements Verification and Validation	1. Reviews specifications of requirements for errors and omissions (P).	Perform (P)

The SWECOM model includes Software Requirements Process and Product Management in the same category as Software Requirements. This skill covers the management of requirements processes and products, in line with the Requirements Engineering cycle as defined in SWEBOK v4.0. However, this skill falls outside the scope of this work, as the objective is not to map all the skills related to the Requirements Engineering process, but rather to identify those specifically associated with the NFRs in the analyzed theoretical framework.

UX is included in the Human-Computer Interaction Skill Area, which comprises the following five specific Skills:

Requirements, Interaction Style Design, Visual Design, Usability Testing and Evaluation, and Accessibility. As with the Skills related to NFRs, these competencies are at the Assist (A) and Perform (P) levels and are associated with the Entry-Level Practitioner profile.

In total, these skills encompass twenty-three tasks, covering Activities such as: identifying the stakeholders responsible for providing HCI requirements (P); assisting in identifying user interface requirements (A); designing and creating prototypes to support the identification of user interface requirements (P); documenting use cases, scenarios, interaction dialogues, and storyboards (P); assisting in identifying user input errors and error handling approaches (A); assisting in identifying interaction modes (A); establishing bidirectional traceability between interaction dialogues, storyboards, specific user interface requirements, and acceptance criteria (P); developing interface prototypes (P); assisting in designing the layout of pages or screens (A); assisting in selecting existing icons and designing new ones (A); assisting in choosing colour themes, font styles, and sizes (A); assisting in designing menus (A); creating mock-ups and sketches of interfaces (P); analyzing the design using usability checklists (P); assisting in identifying representative participants from the target user group for usability testing (A); assisting in designing usability tests (A); conducting usability tests and collecting data (P); analyzing the results of usability tests (P); assisting in identifying accessibility needs for user interfaces (A); assisting in identifying international accessibility needs (A); using appropriate tools and techniques to implement the necessary accessibility (A). The competencies aligned with UX are shown in Table 10.

Table 10. Skills related to user experience in SWECOM (entry-level)

Skill Areas	Skills	Activities	Level
Human- Computer Interaction	Requirements	1. Identifies stakeholders to provide HCI requirements (P). 2. Identifies target users and their attributes (P). 3. Assists in identifying user interface requirements (A). 4. Designs and creates prototypes for use in eliciting user interface requirements (P).	Assist (A), Perform (P)
	Interaction Style Design	1. Documents use cases, scenarios, interaction dialogs, and storyboards (P). 2. Assists in identifying user input errors and error handling approaches (A). 3. Assists in identifying interaction modes (A). 4. Establishes two-way traceability between use cases, scenarios, interaction dialogs, and storyboards and specific user interface requirements and acceptance criteria (P). 5. Develops interface prototypes (P).	Assist (A), Perform (P)
	Visual Design	1. Assists in designing page/screen layout (A). 2. Assists in selecting from existing icons and designing new icons (A). 3. Assists in selecting color theme, font styles, and font sizes (A). 4. Assists in menu design (A). 5. Creates mockups and sketches (P).	Assist (A), Perform (P)
	Usability Testing and Evaluation	1. Analyzes design with a usability checklist (P). 2. Assists in identifying representative test subjects from the target user group (A). 3. Assists in obtaining test subjects (A). 4. Assists in designing usability tests (A). 5. Conducts usability tests and collects data (P). 6. Analyzes results of usability testing (P).	Assist (A), Perform (P)
	Accessibility	1. Assists in identifying accessibility needs for user interfaces (A). 2. Assists in identifying the needs for international accessibility (languages, cultural considerations, and so forth) (A). 3. Uses the selected tools and techniques for implementing required accessibility (A).	Assist (A)

C. Correlation of NFRs between SBC, ACM/IEEE and SWEBOK v4.0

Table 11 shows the correlations established in this study between the NFRs in RF-CC-2017 and the Knowledge Areas (KAs) of ACM/IEEE (CC2020), as well as the SWEBOK v4.0 Subtopics, organized according to the asset tree structure (see Fig. 3). The NFRs in RF-CC-2017 form part of the software engineering content within the systems development training axis. In this context, the content mapped to the NFRs aligns with the KAs of the CC2020 curriculum, specifically "Requirements Analysis and Specification" and "Systems Analysis and Design". These two areas belong to the Systems Modelling category. Table 11 also shows that these assets correspond directly to the SWEBOK v4.0 subtopics, as detailed in Section V.A.

Table 11. Contents correlation from nonfunctional requirements between SBC, ACM/IEEE knowledge areas and SWEBOK v4.0 subtopics

SBC (Contents)	ACM/IEEE (Knowledge Areas)	SWEBOK (SubTopics)
Software Engineering	Requirements Analysis and Specification, and Systems Analysis and Design	Definition of a Software Requirement Nonfunctional Requirements Technology Constraints Quality of Service Constraints Why Categorize Requirements This Way?

The correlation is justified by the fact that when the NFRs are addressed within the Software Engineering and Systems Modelling content of the SBC's RF-CC-2017, they converge conceptually with the KAs of CC2020 and the theoretical structure of the SWEBOK v4.0 Subtopics. These subtopics deal with the definition, categorization, and analysis of NFRs. These subtopics emphasize that NFRs impose technological, performance, safety, and reliability restrictions and are fundamental to ensuring the quality of the final product. Additionally, the categorization proposed in the SWEBOK emphasizes the necessity of applying distinct techniques for elicitation, analysis, specification, and validation based on the nature of the requirements. This highlights that, despite often being implicit, NFRs play a pivotal role in structuring

system design and development. Therefore, the relationship between the documents is based on the conceptual equivalence and convergence of the approaches adopted regarding the strategic relevance of NFRs in software engineering.

D. UX Correlation between SBC, ACM/IEEE and SWEBOK v4.0

Table 12 shows the correlation between the UX content in RF-CC-2017 and CC2020 KAs, as well as SWEBOK v4.0 Subtopics. The analysis revealed that the UX area is incorporated into the Human-Computer Interaction content, which is part of the Systems Development training axis and NFRs. This axis highlights several characteristics, one of which is that methods, techniques, and tools must be used in the development cycle of computer systems to guarantee product quality—a guideline that is directly aligned with the UX area, which covers interface analysis, design, and testing activities.

Therefore, UX-related assets correlate with the KAs in the CC2020 curriculum, specifically those belonging to the UX Design area of the Users and Organizations category. Furthermore, as can be seen in Table 12, the assets correspond to the SWEBOK v4.0 Subtopics mapped in Section V.A.

Table 12. Contents correlation from user experience between SBC, ACM/IEEE knowledge areas and SWEBOK v4.0 subtopics

SBC (Contents)	ACM/IEEE (Knowledge Areas)	SWEBOK (SubTopics)
Human-Computer Interaction	User Experience Design	User-Centered Design
		Objectives of Testing
		User Human Factors

The correlation of UX-related assets is justified by the fact that this area is covered in the Human-Computer Interaction content of the RF-CC-2017, which is directly aligned with the UX Design knowledge area of the CC2020. Both of these areas belong to HCI. These assets also align with the conceptual structure of the SWEBOK v4.0 subtopics. This emphasizes the importance of users in the design process through activities such as gathering requirements, creating task flows, prototyping, and validation. They also establish test objectives that include nonfunctional properties such as performance, reliability, and usability. Furthermore, it demonstrates that users expect robust, intuitive, and responsive interfaces. This requires engineers to understand user profiles and interaction devices. Thus, the relationship between the documents is based on conceptual equivalence and similarity in their treatment of the importance of UX in systems development.

E. Correlation of NFRs Competences and UX in SWECOM, SBC and ACM/IEEE

In accordance with the adopted methodology, analysis of the reference documents revealed a correlation between NFR and UX-related competences based on SWECOM 1.0, RF-CC-2017, and CC2020. These correlations and their respective justifications are presented below.

The competencies relating to software requirements elicitation, analysis, and specification within the SWECOM model are closely aligned with derived competency C.2.7 in the SBC RF-CC-2017. This competency guides the analysis of requirements and their specification for specific problems, including nonfunctional ones, as well as the planning of resolution strategies. Additionally, the ACM/IEEE competencies (CC2020) cover system modelling, design logic, and the structured presentation of solutions — all of which are fundamental to representing, justifying, and communicating NFRs clearly and consistently. These three sets of guidelines share a common emphasis on systematically analyzing constraints such as performance, usability, reliability, and other critical properties that characterize NFRs in the context of software development, which justifies this correlation.

The Requirements Verification and Validation competency, also present in SWECOM, deals directly with NFRs by verifying system conformity in relation to quality criteria and suitability for use. This approach is reflected in SBC competency C.2.9, which involves analyzing system suitability based on previously defined requirements, including nonfunctional ones. This approach is reinforced by the ACM/IEEE competency, which enables clear distinction to be made between functional requirements and NFRs.

The correlation between SWECOM, SBC, and ACM/IEEE is therefore based on the recognition that NFRs not only impose technical restrictions but also guide design decisions and software quality assessment. They are also essential for ensuring the solution's sustainability and adherence to its intended use. Table 13 shows the complete matrix of convergences between the references with regard to NFRs.

Table 13. Competency mapping from nonfunctional requirements between SWECOM, SBC and ACM/IEEE

SWECOM (Skills)	SBC (Derived Competencies)	ACM/IEEE (Competencies)
Software Requirements Elicitation, Software Requirements Analysis, and Software Requirements Specification	Identify and analyze requirements and specifications for specific problems and plan strategies for their solutions (C.2.7.-CE-IV)	Analyze an industry problem to determine underlying recurrence relations and present the solution to professionals by using a variety of basic recurrence relations.
		Design a simple sequential problem and a parallel version of the same problem using fundamental building blocks of logic design and use appropriate tools to evaluate the design for a commercial organization and evaluate both problem versions.
		Present to a client the design of a simple software system using a modeling notation (such as UML), including an explanation of how the design incorporated system design principles.
Software Requirements Verification and Validation	Analyze the extent to which a computer-based system meets the defined criteria for its current and future use (suitability) (C.2.9.-CE-VIII)	Conduct a review of a set of software requirements for a local project, distinguishing between functional and nonfunctional requirements, and evaluate the extent to which the set exhibits the characteristics of good requirements.

With regard to UX, Table 14 shows the correlation between the SWECOM 1.0 competencies, the SBC RF-CC-2017 competencies, and the ACM/IEEE CC2020 competencies. In SWECOM, the competencies relating to

requirements, interaction design, visual design, usability, and accessibility testing and evaluation correspond directly to SBC competency C.2.1. This competency emphasizes applying HCI principles to evaluate and develop products for various platforms. Similarly, the CC2020 competencies emphasize the importance of designing and developing interactive applications based on user-centred design principles and using suitable tools, techniques, and terminology to optimize usability and UX in various contexts, including commercial and social organizations.

Table 14. Competency mapping from user experience between SWECOM, SBC and ACM/IEEE

SWECOM (Skills)	SBC (Derived Competencies)	ACM/IEEE (Competencies)
Requirements, Interaction Style Design, Visual Design, Usability Testing and Evaluation, and Accessibility	Apply human-computer interaction principles to evaluate and build a wide range of products including user interfaces, web pages, multimedia systems, and mobile systems (C.2.1.-CE-XIII)	Design an interactive application, applying a user-centered design cycle with related tools and techniques (modes, navigation, visual design), to optimize usability and user experience within a corporate environment.
		Design and develop an interactive application for a local charity, applying a user-centered design cycle with related vocabulary, tools, and techniques that optimize usability and user experience.
		Create a simple application, together with help and documentation, that supports a graphical user interface for an enterprise and conduct a quantitative evaluation and report the results.
		Design for a client a responsive web application utilizing a web framework and presentation technologies in support of a diverse online community.
		Adopt processes to track customer requests, needs, and satisfaction.
		Analyze and evaluate a user interface that considers the context of use, stakeholder needs, state-of-the-art response interaction times, design modalities taking into consideration universal access, inclusiveness, assistive technologies, and culture-sensitive design.
		Create and conduct a simple usability test to analyze and evaluate a user interface that considers the context of use taking into consideration universal access and culturally sensitive design.

Furthermore, CC2020 suggests that graduates should be capable of creating a basic application with guidance and documentation, developing a responsive web application for a client using frameworks and presentation technologies, monitoring client requirements and satisfaction, and analyzing interfaces in the context of use, taking into account stakeholders, modern response times, design modalities, universal access, assistive technologies, inclusion, and cultural sensitivity. Finally, it highlights the importance of creating and executing simple usability tests to evaluate interfaces from these same perspectives.

Thus, the correlation between the analyzed reference assets is justified by the conceptual and practical convergence in the treatment of UX. This is evidenced by the common emphasis on competencies that apply HCI principles, adopt user-centred methodologies, and use evaluation strategies that include usability, accessibility, and inclusion. The outcome is satisfactory UX. This demonstrates consistency in preparing professionals to design and evaluate interactive systems that consider multiple contexts and user needs.

F. Mapping Evaluation

Once the mapping of assets and justification of established relationships had been completed, the peer review process began. This step was carried out by experts in the target area who had not been involved in the mapping. The aim was to carefully evaluate the research and identify its strengths, weaknesses and areas for improvement.

As mentioned in Section IV, the mapping was carried out by the main author of this study—a PhD Computing student — in collaboration with a researcher with over 15 years' experience in SE and HCI. To validate the results, peer review was conducted by a researcher with over 25 years' experience in SE who has extensive experience of researching computer education and has published over 200 studies on teaching and learning in Computing. This researcher has also trained over 50 Master's students and over 20 doctoral students in their thesis research.

Following confirmation of the reviewer's participation, two remote meetings were held to present the research objectives and mapping results. The two researchers responsible for the mapping and the reviewer participated in these meetings.

At the first meeting, the initial version of the asset mapping was presented in detail. The reviewer then proposed a set of improvements, which the researchers analyzed and incorporated. These were validated by the reviewer at the second meeting. The main recommendations are described below:

- **Refining NFRs and UX in the context of the research:** The reviewer requested that the mapping be restricted to content and competencies exclusively related to NFRs and UX. This was justified by the need to exclude assets linked to Requirements Engineering processes, the Software Life Cycle and other areas, such as Software Design and Project Management, which lie outside the scope of this work. Based on this feedback, the 'Software Requirements Process and Product Management' competency from SWECOM 1.0, as well as the competency derived from RF-CC-2017 'Employ methodologies aimed at guaranteeing quality criteria throughout all the development stages of a computing solution' (C.2.8 and C.4.8 – CE-VII), were removed. These competencies belonged to the Systems Development Training Axis and were also included in the Systems Deployment Axis,
- **Improving the justifications for asset mapping:** The reviewer requested more detailed justifications for the mappings between Subtopics (SWEBOK v4.0), Skills (SWECOM 1.0), Content (RF-CC-2017), and Knowledge Areas (CC2020). The guidance aimed to reinforce adherence to the target contents of the NFRs and UX. These justifications have been improved and incorporated into the mapping, as shown in Section V (Results), which highlights the criteria adopted for the correlation between

the assets,

- **Revision of items:** The reviewer requested the regrouping or exclusion of certain elements to ensure an exclusive focus on NFRs and UX. In response to this request, Topics from SWEBOK v4.0 that were considered cross-cutting, such as "Software Architecture Fundamentals", "Software Design Qualities", "Construction", "Software Testing Fundamentals", "Software Testing in the Development Processes", "Application Domains", "Testing of and Testing Through Emerging Technologies", "Software Engineering Operations Planning", and "Software Quality Assurance Process", were removed,
- **Creation of a structure of relationships between assets:** The reviewer recommended developing a visual structure to make the relationships between the assets in the analyzed models explicit and facilitate understanding of these interrelationships. In response, a relationship tree was developed (see Section V), which organizes the references used and their connections. This graphic representation clarifies the equivalence between the assets, making the mapping more robust and transparent.

Therefore, Step 6 of this study incorporated all contributions from the peer review into the mapping to create the final version presented here.

VI. Discussion

The results of this study demonstrate significant convergence in the Brazilian and international curricular reference guidelines analyzed with regard to the presence and treatment of NFRs and UX in Computing courses. Correlation analysis of the RF-CC-2017, CC2020, SWEBOK v4.0, and SWECOM 1.0 documents indicates conceptual and training alignment, which could support an integrated approach to these sub-areas in the teaching process.

Regarding NFRs specifically, the associated content was found to be adequately covered in the "Systems Development" axis of RF-CC-2017 and the CC2020 KAs "Requirements Analysis and Specification" and "Systems Analysis and Design". The presence of specific Subtopics in SWEBOK v4.0, such as NFRs, technology constraints and quality of service constraints, enhances understanding of NFRs by emphasizing their categorization and impact throughout the software life cycle. This shows that NFRs are recognized as structuring elements of SE that require specific Skills for their elicitation, analysis, specification, and validation. The SWECOM 1.0 model reinforces this perspective by assigning practical activities directly related to these stages at the beginning of one's career.

Regarding UX, the analysis revealed an equally robust correlation. The HCI content in the RF-CC-2017 is aligned with the KA 'User Experience Design' of CC2020, as well as the SWEBOK v4.0 Subtopics dealing with user-centered design, objectives of testing, and user-human factors. These topics emphasize the importance of usability, accessibility, and adaptation to different contexts, thereby reinforcing the centrality of the user in the development process. The competencies described in SWECOM 1.0 support this approach by incorporating activities such as prototyping, usability testing, human factors analysis, and accessibility implementation—tasks that are usually carried out by early-career software engineers. This demonstrates that UX aspects are fundamental to the graduate profile and should not be considered optional or supplementary.

An integrated analysis of the contents and competences present in the references shows that NFR and UX can both be approached interdisciplinarily in Computing curricula. A practical example of this integration can be seen in the SWECOM "Requirements" UX competency, which involves activities such as identifying stakeholders and target users, and defining requirements for interfaces—actions that are directly linked to NFR-associated practices. Furthermore, the User-Centered Design subtopic of SWEBOK v4.0 highlights this intersection by incorporating tasks such as collecting user requirements and evaluating the design solution against the initial requirements. This suggests a convergent approach between usability and software quality. Similarly, the "Objectives of Testing" subtopic explains that interface evaluations should consider nonfunctional properties such as performance, reliability, and usability—points that strengthen the links between UX and NFRs. The SWECOM "Usability Testing and Evaluation" competency is also noteworthy as it encompasses activities such as design analysis based on usability checklists, the conception and execution of tests and the collection and analysis of results. This highlights usability as a strong point of intersection between NFRs and UX.

Furthermore, incorporating NFR and UX content and competencies into both the RF-CC-2017 and the CC2020 reinforces the idea that systems development requires the simultaneous consideration of technical robustness and UX. Adopting updated versions of the references (SWEBOK v4.0 and CC2020) and incorporating the SWECOM 1.0 competency model brings the mapping up to date, rendering it more comprehensive and applicable. This overcomes the limitations of previous studies that used outdated versions or restricted themselves to broad areas/disciplines.

Another relevant aspect is the complementary nature of the references. SWEBOK v4.0 organizes technical knowledge hierarchically into areas and sub-areas. CC2020 defines a set of competencies to be developed throughout the programme. RF-CC-2017 adapts these guidelines for the Brazilian context and aligns them with the 2016 DCNs. SWECOM 1.0 assigns proficiency levels and specifies related practical activities. This complementarity supports the feasibility of an integrated curricular approach in which NFR and UX content and competencies are addressed progressively, with clearly defined learning objectives and tangible outcomes.

However, despite this convergence, the references' notable structural differences may affect curriculum integration. For instance, SWEBOK's emphasis on hierarchical knowledge does not always align directly with SWECOM's activity-based competencies, and the scope and contextualization of CC2020 and RF-CC-2017 differ. Without deliberate alignment between theoretical coverage and practical skill development, these discrepancies may create gaps or overlaps if curricula are integrated. Recognizing and addressing these differences is crucial to ensuring coherent, effective interdisciplinary integration of NFR and UX content across SE and HCI courses.

Finally, it should be noted that the proposed mapping contributes to curriculum planning and critical reflection on

the teaching of SE and HCI. By explaining the points of convergence between the analyzed references, this study provides course coordinators and professors with the support they need to review and improve their pedagogical strategies. This enables them to incorporate NFR and UX content into SE and HCI curricula in Computing courses in a structured and strategic way.

Building on this contribution, the mapping can be applied to the design of course plans. It can serve as a guide for defining competencies and skills, as well as for structuring teaching units, their prerequisites and guiding questions. It can also inform the development of programme content and expected learning outcomes, ensuring alignment between objectives, instructional strategies, and assessment practices. Each unit can integrate content from SE and HCI, emphasizing the connection between NFRs and UX. Furthermore, the mapping supports curricular progression by enabling competencies to be distributed logically and sequentially from introductory to advanced levels. For example, initial units could focus on competencies related to requirements elicitation and user-centered design, while more advanced units could address the validation of quality requirements and usability testing. Thus, the mapping ensures that the course prepares professionals capable of developing robust, user-centered systems by consolidating essential content and guiding questions, and equipping them with the necessary practical competencies.

VII. Threats to Validity

This section analyses the potential threats to the validity of this study and presents the strategies adopted to minimize them. Four aspects of validity were considered for this purpose, in accordance with the definition proposed by Runeson and Höst [31]. Additionally, Subsection C expands the analysis to cover threats related to reviewer roles, inter-reviewer agreement, and resolution methods.

A. Internal Validity

This refers to the extent to which the results accurately reflect the reality being studied, rather than being significantly influenced by external factors. To mitigate this threat, the asset mapping process was structured as described in Section IV. As part of this process, the final version of the mapping was peer-reviewed before finalization to validate the research results.

B. External Validity

This relates to the possibility of generalizing the results to other contexts or populations. To mitigate this threat, a systematic methodology was developed for carrying out the asset mapping. The relevance and applicability of this approach was corroborated by the studies presented in Section III, which adopted similar construction processes. This increases the potential for generalization of the findings.

C. Construct Validity

This refers to the precision with which the concepts and variables under investigation are defined and measured. Given the theoretical nature of this study, a peer review process was implemented to mitigate threats to construct validity. Additionally, developing a systematized mapping methodology helped to reduce the associated risks.

Two researchers with expertise in SE and HCI carried out the mapping process, which was subsequently reviewed by a senior researcher with over 25 years' experience in Computing education. There was high agreement between the two initial mappers, with disagreements occurring in less than 10% of cases. The disagreements that did occur mainly concerned borderline features, such as cross-cutting topics in SWEBOK and SWECOM competencies, and were resolved through discussion until consensus was reached. If consensus could not be reached immediately, the case was presented to the senior reviewer for adjudication. This third-party review ensured impartiality and methodological rigour, thereby reducing interpretative bias. The structured resolution method enhanced the reproducibility and credibility of the mapping process.

D. Conclusion Validity

This relates to the confidence placed in the inferences drawn from the collected and analyzed data. The related studies discussed in Section III reinforce the consistency of the inferences made in this study, as similar results were produced when mapping assets from the ACM/IEEE, SBC, and SWEBOK references in different versions. Furthermore, peer review helped to validate the developed mapping, reducing possible interpretative biases, and strengthening the robustness of the conclusions.

VIII. Conclusion

The aim of this study was to map the content and competencies related to NFRs and UX in the Brazilian (RF-CC-2017) and international (CC2020, SWEBOK v4.0, and SWECOM 1.0) Computing curricula. A systematic methodology, validated by an expert, was used to identify conceptual and educational convergences in order to support the integration of SE and HCI across disciplines.

The results showed that NFRs and UX are widely covered in the analyzed documents and have the potential to be articulated in Computing curricula. The presence of specific topics and subtopics in SWEBOK v4.0, combined with the practical competencies set out in SWECOM 1.0, the organization by knowledge areas in CC2020, and the structure of derived competencies in RF-CC-2017 suggests an approach that could help to overcome disciplinary fragmentation and promote more integrated, contextualized training.

The analysis also revealed a strong case for the intersection between NFRs and UX given their shared focus on technical quality, usability, accessibility, and suitability for the intended context. This convergence highlights the importance of addressing this content in a coordinated manner in order to train professionals who are capable of designing, developing, and evaluating robust, human-centered systems.

The study's main limitation is its exclusive focus on documents, with no empirical validation in real educational contexts. Nevertheless, this limitation suggests avenues for future research, such as applying and refining the proposed approach. Our intention is to develop an interdisciplinary approach to teaching and learning NFRs and UX. This approach will be based on the asset mapping presented in this study, and we will conduct empirical investigations to assess

its impact on teaching and learning. Additionally, we plan to conduct a study with programme coordinators to examine the mapping's practical adequacy and collect qualitative and quantitative data on its applicability in different institutional contexts.

The main contribution of this work is its theoretical and methodological support for course coordinators and professors who wish to revise or improve the curricula of SE and HCI subjects, thereby strengthening the integration of NFRs and UX.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

J.V.F.: Conceptualization, Methodology, Formal analysis, Investigation, Writing—original draft preparation; S.R.B.O.: Conceptualization, Validation, Writing—review and editing, Supervision. C.d.S.P.: Conceptualization, Validation, Supervision. All authors had approved the final version.

FUNDING

This research was funded by the Amazon Foundation for Support of Studies and Research of Pará (FAPESPA) through the granting of a doctoral scholarship.

ACKNOWLEDGMENT

The authors would like to thank the Amazon Foundation for Support of Studies and Research of Pará (FAPESPA) for their financial support in conducting this research.

REFERENCES

[1] I. Sommerville, *Engineering Software Products: An Introduction to Modern Software Engineering*, 1st ed., Hoboken, NJ, USA: Pearson Education, 2019.

[2] R. S. Pressman and B. Maxim, *Software Engineering-9,* McGraw Hill Brasil, 2021.

[3] C. Werner, "Towards a theory of shared understanding of non-functional requirements in continuous software engineering," in *Proc. the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings,* ACM/IEEE, 2022, pp. 300–304. doi: 10.1145/3510454.3517069

[4] J. V. Ferreira, S. R. B. Oliveira, and C. do S. Portela, "Diagnosis of the teaching of software non-functional requirements and user experience in undergraduate computer science programs in Brazil," *Revista Brasileira de Informática na Educação (RBIE)*, vol. 33, pp. 643–671, 2025. doi: 10.5753/rbie.2025.5435

[5] R. Krause and M. Rosala. (2019). *User Experience Careers: What a Career in UX Looks Like Today,* 2nd ed., Nielsen Norman Group. [Online]. Available: https://media.nngroup.com/media/reports/free/UserExperienceCareers_2nd_Edition.pdf

[6] H. Washizaki, *Guide to the Software Engineering Body of Knowledge (SWEBOK Guide)*, Version 4.0, IEEE Computer Society, 2024.

[7] A. F. Zorzo, D. Nunes, E. Matos, I. Steinmacher, J. Leite, R. M. Araujo, R. Correia, and S. Martins, *Training References for Undergraduate Computer Science Courses in Brazil*, Brazilian Computing Society, Porto Alegre, Brazil, 2017.

[8] S. Barbosa and S. Silva, *Human-Computer Interaction*, 1st ed., Rio de Janeiro: Elsevier, 2010. (in Portuguese)

[9] I. F. G. Lima, M. C. Melo, W. E. Silva, and T. P. Falcão, "Interdisciplinarity and HCI: Possibilities in the context of the bachelor's degree in computing," in *Proc. Workshop on Education in HCI (WEIHC)—XVIII Brazilian Symposium on Human Factors in Computing Systems*, Brazil, 2019, pp. 82–87. doi: 10.5753/ihc.2019.8405 (in Portuguese)

[10] L. Diniz, F. Ferreira, and J. Diniz, "Interdisciplinarity in teaching software engineering and human-computer interaction using digital technologies: An experience report," in *Proc. the XXVII Workshop on Informatics in Education (WIE 2021)*, Brazil, 2021, pp. 116–127. doi: 10.5753/wie.2021.218683

[11] CC2020 Task Force, *Computing Curricula 2020: Paradigms for Global Computing Education*, New York, United States, 2020. doi: 10.1145/3467967

[12] IEEE Computer Society, *SWECOM—Software Engineering Competency Model,* Version 1.0, Washington DC: IEEE Computer Society, 2014.

[13] Brazil. (2016). *Ministry of Education. Resolution no. 05, of November 16, 2016: National Curricular Guidelines for Undergraduate Computing Courses.* Brazil: MEC. [Online]. Available: https://abmes.org.br/arquivos/legislacoes/Res-CES-CNE-005-2016-11-16.pdf

[14] M. Kumar and V. Choppella, "A study of the design and documentation skills of industry-ready CS students," in *Proc. 15th Annual ACM India Compute Conference*, pp. 23–28, 2022. doi: 10.1145/3561833.356184

[15] R. K. Raj, J. Impagliazzo, S. G. Aly, D. S. Bowers, H. Connamacher, S. Kurkovsky, B. MacKellar, T. Prickett, and M. M. Samary, "Toward competency-based professional accreditation in computing," in *Proc. 2022 Working Group Reports on Innovation and Technology in Computer Science Education*, pp. 1–35, 2022. doi: 10.1145/3571785.357412

[16] R. K. Raj, A. N. Kumar, M. Sabin, and J. Impagliazzo, "Interpreting the ABET computer science criteria using competencies," in *Proc. 53rd ACM Technical Symposium on Computer Science Education (SIGCSE)*, 2022, vol. 1, pp. 906–912. doi: 10.1145/3478431.349929

[17] J. Impagliazzo, P. Bourque, and N. R. Mead, "Incorporating CC2020 and SWECOM competencies into software engineering curricula: A tutorial," in *Proc. 2020 IEEE 32nd Conference on Software Engineering Education and Training (CSEE&T)*, 2020, pp. 1–3. doi: 10.1109/CSEET49119.2020.9206238

[18] V. Ferreira, M. Souza, and P. A. P. Júnior, "Characterization of the adequacy level of computing curricula in Brazil to the SBC competency model," in *Proc. Workshop on Computer Education (WEI)*, 2024, pp. 443–454. doi: 10.5753/wei.2024.1966

[19] ACM/IEEE. (2013). *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science.* New York, NY: Association for Computing Machinery. [Online]. Available: https://www.acm.org/binaries/content/assets/education/cs2013_web_final.pdf

[20] P. Bourque and R. Fairley. (2014). *SWEBOK: Guide to the Software Engineering Body of Knowledge*, 3rd ed. Los Alamitos: IEEE Computer Society. [Online]. Available: https://cs.fit.edu/~kgallagher/Schtick/Serious/SWEBOKv3.pdf

[21] A. P. C. M. Ferraz and R. V. Belhot, "Bloom's taxonomy: Theoretical review and presentation of the instrument's adaptations for defining instructional objectives," *Management & Production*, vol. 17, pp. 421–431, 2010. doi: 10.1590/S0104-530X2010000200015

[22] V. S. Castro and S. R. B. Oliveira, "Contents and competences for teaching software design in computer science courses: A mapping of the CC-2020, RF-CC-2017 and SWEBOK-V3.0 assets," in *Proc. 19th International Conference on Information Systems & Technology Management*, Brazil, 2022. doi: 10.5748/19CONTECSI/PSE/EDU/6979

[23] A. F. O. Colares, J. C. Furtado, and S. R. B. Oliveira, "Content and skills for teaching software process improvement in the computer science course: A mapping of ACM/IEEE, SBC, SWEBOK, CMMI and MR-MPS-SW assets," in *Proc. IEEE Frontiers in Education Conference (FIE)*, 2023, pp. 1–8. doi: 10.1109/FIE58773.2023.10343447

[24] E. C. Carvalho, S. R. B. Oliveira, and F. W. S. Garcia, "Curriculum mapping in software project management: An analysis between CC-2020, SWEBOK 3.0 and RF-CC-2017," *Pedagogical Notebook*, vol. 21, no. 13, 2024. doi: 10.54033/cadpedv21n13-340

[25] E. D. Santos and S. R. B. Oliveira, "A mapping of teaching assets from CC-2020, RF-CC-2017 and SWEBOK 3.0 related to the software requirements knowledge area," *Education and Development Notebooks*, vol. 16, no. 4, 2024. doi: 10.55905/cuadv16n4-088

[26] M. Phothisonothai, K. Orkphol, N. Pannurat, A. Limmanee, W. Lamaisri, and C. Euafua, "Towards integrating physiological data in UI/UX design learning and engineering education," in *Proc. 2024 10th International Conference on Communications and Electronics (ICCE)*, Danang, Vietnam, 2024, pp. 573–576. doi: 10.1109/ICCE62051.2024.10634623

[27] A. Nimkoompai, S. Chotirat, and S. Nuchitprasitchai, "Student-centered learning in HCI/UX/UI framework: The Thai experience," in *Proc. 2024 8th International Conference on Information Technology (InCIT)*, Chonburi, Thailand, 2024, pp. 121–126. doi: 10.1109/InCIT63192.2024.10810571

[28] Y. Li, J. Keung, X. Ma, J. Zhang, Z. Yang, and S. Liu, "Learning gaps in project-based requirements engineering education: A case study of student projects," in *Proc. 22023 International Symposium on Educational Technology (ISET)*, 2023, pp. 239–243. doi: 10.1109/ISET58841.2023.00054

[29] F. B. A. Ramos, A. Pedro, M. Cesar, A. A. M. Costa, M. B. Perkusich, H. O. de Almeida, and A. Perkusich, "Evaluating software developers' acceptance of a tool for supporting agile non-functional requirement elicitation," in *Proc. the International Conference on Software Engineering and Knowledge Engineering (SEKE)*, KSI Research, 2019, pp. 26–42. doi: 10.18293/SEKE2019-107

[30] A. Oliveira, J. Correia, W. K. G. Assunção, J. A. Pereira, R. de Mello, D. Coutinho, C. Barbosa, P. Libório, and A. Garcia, "Understanding developers' discussions and perceptions on non-functional requirements: The case of the Spring ecosystem", in *Proc. the ACM on Software Engineering,* 2024, pp. 1–22. doi: 10.1145/3643750

[31] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 2, pp. 131–164, 2009. doi: 10.1007/s10664-008-9102-8

Copyright © 2026 by the authors. This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited (CC BY 4.0).