Mapping learning objectives of project-based undergraduate software engineering courses to CC2020 competency model

Ahmad D. Suleiman
Electrical & Computer Eng
Clarkson University
Potsdam NY, USA
suleimad@clarkson.edu

Daqing Hou
Electrical & Computer Eng
Clarkson University
Potsdam NY, USA
dhou@clarkson.edu

Yu Liu
Electrical & Computer Eng
Clarkson University
Potsdam NY, USA
yuliu@clarkson.edu

Jan DeWaters
Inst for STEM Education
Clarkson University
Potsdam NY, USA
jdewater@clarkson.edu

Mary M Small
Inst for STEM Education
Clarkson University
Potsdam NY, USA
mmsmall@clarkson.edu

Juliana G de Souza
Department of Computer Science
Virginia Commonwealth Univ.
Richmond VI, USA
desouzajg@vcu.edu

David Shepherd
Department of Computer Science
Louisiana State University
Baton Rouge LA, USA
davidshepherd@gmail.com

***Abstract*—This qualitative research performs a thematic analysis of the learning objectives in existing project-based undergraduate software engineering courses to align them with the competency model defined in the Computing Curricula 2020 reports (CC2020). This study identifies the trends, strengths, and gaps in how the reviewed course learning objectives cover the knowledge, skill, and disposition components of the CC2020 competency model. The learning objectives were categorized according to knowledge elements, skills, and dispositions as defined in the CC2020 competency model. Our analysis shows that 54% of knowledge elements from the reviewed learning objectives do not have any skill level specified and overall, only two out of the eleven dispositions in CC2020 are specified ("Collaboration" and "Professional"). We also find that technical knowledge elements from the software development category (e.g., software process, software design, and software quality, verification & validation) and systems modeling category (e.g., systems analysis & design, and requirements analysis and specification), probably unsurprisingly, are covered the most often. Similarly, collaboration & teamwork, and oral & written communication are unsurprisingly the most common professional & foundational knowledge elements in the reviewed course's learning objectives as they are essential to project-based learning. Although they are essential for the completion of a successful software project, knowledge elements such as time management, security technology & implementation, and user experience design are rarely mentioned. We discuss the implications of our findings on course design.**

Keywords— Learning Outcomes, Learning Objectives, Project-based learning, Competency, CC2020, Software Engineering

I. INTRODUCTION

Project-based Learning (PjBL) is a learning approach that has been widely applied to developing the competency needed by the industry. PjBL provides students with the opportunity to apply their knowledge and skills in the context of authentic real-life projects [1]. PjBL is mostly used to support learning at the higher levels of the revised Bloom's Taxonomy [2], from "applying" to "creating", and is effective in fostering higher-order thinking skills in students [3][4][5].

Funding agency: United States National Science Foundation.

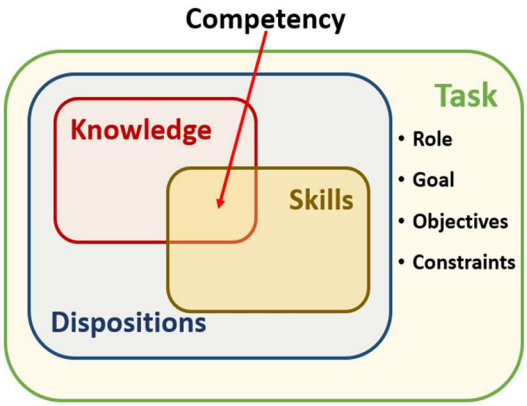

Fig. 1. Conceptual structure of CC2020 competency model which defines competency as composed of knowledge, skills, and dispositions in the context of performing a task [6].

Building a good software project for PjBL requires a significant effort from the course instructor. Nonetheless, it is also crucial to design a course project in a way that aligns with and supports the course's set learning objectives. If the project fails to align with and support these objectives, the purpose of using a project-based approach becomes futile. Thus, it is imperative for educators to understand the commonly used learning objectives in software engineering project courses and to better design projects that effectively target the course learning objectives.

On the other hand, ACM and IEEE Computer Society have issued the Computing Curricula 2020 (CC2020) [6] that advocates for a transition of computing learning and education from the current knowledge-based one [7] to a competency-based one. CC2020 defines competency as composed of four dimensions, i.e., knowledge, skills, dispositions, and task, where the former three are observed within the performance of a specific task. Knowledge is the "know-what" dimension of competency. It is proficiency in core concepts and content and

the application of learning to new situations. Skills are the "know-how" dimension of competency that defines the ability to carry out tasks with determined results. Dispositions are the "know-why" dimension of the competency that defines one's intellectual, social, or moral tendencies. Fig. 1 illustrates the conceptual structure of the competency model. CC2020 presents suggested elements of each of these dimensions that encompass all curricula in computing educational programs. Crucially the purpose/goals of a course are expressed through statements of learning objectives/outcomes. In contrast to a competency approach, the learning outcomes in knowledge-based learning tend to be expressed in terms of knowledge units and skills only. They thus fall short when we are faced with challenges like assessing the development of professional graduate attributes or performing comparisons of educational programs [8].

To illustrate the process for developing a competency-based curriculum, Clear et al. [8] demonstrate how to design computer science competency statements from the CS2013 [9] curriculum model.

The disposition dimension of CC2020 is needed to satisfy the professional expectations of a modern workplace. Bowers et al. [10] make the first formal attempt to link CC2020 attitudes to an existing employer-centered skills framework, i.e., the Skills Framework for the Information Age (SFIA) [11]. SFIA is a widely adopted competency framework with users all over the world. SFIA defines professional skills and competencies required across the broad field of computing. Bowers et al. [10] confirmed that the CC2020 dispositions cover all these behavioral and professional qualities in the SFIA framework.

This paper presents a case study by applying thematic analysis for mapping existing project-based software engineering courses to the CC2020 model. This paper for the first time presents a summary of the learning objectives that are commonly used in existing software engineering project courses as well as discusses the implications of the identified trend and gaps in the coverage of CC2020 model, on course design. To our best knowledge, there has been no previous study that maps the learning objectives of existing courses to the CC2020 competency model.

The rest of the paper is organized as follows. Section II describes our research methodology and process. Section III presents our results with discussion. Lastly, Section IV concludes the paper.

II. Research Methodology and Process

We use thematic analysis as our research methodology to map learning objectives in existing project courses to the CC2020 competency model. In this section, we start by stating our research question, then we explain the process of search, selection, and analysis of learning objectives of existing courses. Finally, we highlight our initial codes and the coding process employed. The results of our thematic analysis are presented in Section III.

A. Research Question

Our qualitative research process is guided by the research question "To what extent do the learning objectives of current software engineering project courses cover the three components of the CC2020 competency model?"

B. Search

To find the software engineering courses, we use two approaches: (a) we performed Google search with the following keywords: '*software engineering*', '*course*', '*syllabus*', and '*class*'; and (b) to find more courses, we searched the course catalog of a non-exhaustive list of universities with the keywords '*software engineering*'. A further search was also required for individual course titles to get the most recent syllabus and for other publicly available course information.

C. Selection

From the search result, courses were selected based on the following criteria: (a) the course must be a higher education course; (b) the course must have publicly available information, at least a course name, and a course description, (c) the course title must contain "*software engineering*" to indicate it's a software engineering course and (d) the course must have a major project. The selection process yielded a total of 31 syllabi in software engineering, 29 of which were gathered via an online search in various university course catalogs, and 2 were from other educational research papers.

D. Analyses

Thematic analysis [12] was used to analyze the data collected from the search process. This process includes six phases, namely, familiarization with the data, generating initial codes, searching for themes, reviewing potential themes, defining, and naming themes, and finally producing a report [13].

In addition to the "learning objective" section of a course, course learning objectives were also inferred from other publicly available materials such as course descriptions, project descriptions, instruction topics, etc. The inferring process follows a careful textual reading for any emphasis, expectation, or achievement of learning. 8 out of 31 courses were inferred. All data, including the raw materials and the inferred objectives, can be made available upon request to the first author.

E. Initial Codes

The learning objectives were coded according to knowledge elements, skills, and dispositions as defined in the CC2020 competency model.

1) Knowledge Element: Knowledge is the "know-what" dimension of competency as a factual understanding. This dimension reflects the enumerated subject matter that teachers catalog as topics in their syllabi. An element of knowledge designates a core concept essential to competency. CC2020 competency model encompasses concepts that are *technical* (computing concepts), *foundational and professional* (indicative of a workplace), and *domain-specific* (the task setting). Six categories of technical knowledge elements and thirteen foundational & professional knowledge elements are highlighted in Table I and Table II respectively.

TABLE I. TECHNICAL KNOWLEDGE ELEMENTS [6]

Category	Computing Knowledge Area	
	Code	*Name*
1. Users and Organizations	K(C1.1)	Social Issues and Professional Practice
	K(C1.2)	Security Policy and Management
	K(C1.3)	IS Management and Leadership
	K(C1.4)	Enterprise Architecture
	K(C1.5)	Project Management
	K(C1.6)	User Experience Design
2. Systems Modeling	K(C2.1)	Security Issues and Principles
	K(C2.2)	Systems Analysis and Design
	K(C2.3)	Requirements Analysis and Specification
	K(C2.4)	Data and Information Management
3. Systems Architecture and Infrastructure	K(C3.1)	Virtual Systems and Services
	K(C3.2)	Intelligent Systems (AI)
	K(C3.3)	Internet of Things
	K(C3.4)	Parallel and Distributed Computing
	K(C3.5)	Computer Networks
	K(C3.6)	Embedded Systems
	K(C3.7)	Integrated Systems Technology
	K(C3.8)	Platform Technologies
	K(C3.9)	Security Technology and Implementation
4. Software Development	K(C4.1)	Software Quality, Verification, and Validation
	K(C4.2)	Software Process
	K(C4.3)	Software Modeling and Analysis
	K(C4.4)	Software Design
	K(C4.5)	Platform-Based Development
5. Software Fundamentals	K(C5.1)	Graphics and Visualization
	K(C5.2)	Operating Systems
	K(C5.3)	Data Structures, Algorithms, and Complexity
	K(C5.4)	Programming Languages
	K(C5.5)	Programming Fundamentals
	K(C5.6)	Computing Systems Fundamentals
6. Hardware	K(C6.1)	Architecture and Organization
	K(C6.2)	Digital Design
	K(C6.3)	Circuits and Electronics
	K(C6.4)	Signal Processing

2) Skill Level: Skills introduce the capability of applying knowledge to actively accomplish a task. Hence, a skill expresses an element of knowledge as acted upon with proficiency to define the "know-how" dimension of competency. CC2020's definition of competency has adopted the 2001 revision of Bloom's Taxonomy of six levels of the cognitive process [2] to specify the degree of skill expected in successful task accomplishment. Table III summarizes an ordered sequence of six cumulative levels of skill (cognitive skill) together with abbreviated definitions. For instance, students create new software components or modules that extend the functionality of existing systems or introduce entirely new capabilities at the "Creating" level. They combine their programming skills to build innovative features, modules, or libraries. As another example, requirement validation and quality assurance require activities that are usually performed at the "Evaluating" and "Analyzing levels respectively.

3) Disposition: Dispositions frame the "know-why" dimension of competency and prescribe a temperament of quality of character in task performance. Dispositions moderate the behavior of applying "know-what" that becomes "know-how". Dispositions control whether and how an individual is inclined to use his/her skills. Dispositions can denote the values and motivations that guide applying knowledge while designating the quality of knowing indicative of a standard of professional performance [14]. Table IV displays the eleven dispositions of CC2020, which were derived from the literature by the CC2020 task force [6]. For instance, students demonstrate a "Collaborative" disposition by active participation in a team project, sharing knowledge and expertise, giving or receiving feedback, task allocation and coordination, effective communication, etc. Another example is a "Responsive" disposition, which is all about respecting the timing constraints for communication and activities required to accomplish the project's goals. As a final example, the "Meticulous" disposition is marked by precise attention to details, which helps achieve thoroughness and accuracy when accomplishing a task. For example, students testing edge cases and providing quality documentation can be considered as being "Meticulous".

TABLE II. PROFESSIONAL & FOUNDATIONAL KNOWLEDGE ELEMENTS [6]

Professional & Foundational Knowledge Element		Meaning
Code	*Name*	
K(P1)	Oral Communication & Presentation	Conveying a message orally using real-time presentations with visual aids related to audience interests and goals
K(P2)	Written Communication	Use of a written form of interaction between people and organizations that provides an effective way of messaging
K(P3)	Problem Solving and Troubleshooting	A logical and orderly search for the source of a unit problem and making the unit operational again
K(P4)	Project and Task Organization and Planning	A process to provide decisions about a project concerning the organization and planning to achieve a successful result
K(P5)	Collaboration and Teamwork	Apportion challenging tasks into simpler ones and then work together to complete them efficiently
K(P6)	Research and Self-Starter/Learner	Someone who begins or undertakes work or a project without needing direction or encouragement to do so
K(P7)	Multi-Task Prioritization and Management	Processing several issues or tasks at once while arranging them according to importance to do a specific one first
K(P8)	Relationship Management	A strategy to maintain an ongoing level of engagement usually between a business and its customers or other businesses
K(P9)	Analytical and Critical Thinking	A mental process of simplifying complex information into basic parts and evaluating results to make proper decisions
K(P10)	Time Management	An ability to use a person's time in an effective or productive manner to work efficiently
K(P11)	Quality Assurance / Control	Use of techniques, methods, and processes to identify and prevent defects according to defined quality standards
K(P12)	Mathematics and Statistics	Use of numbers and theories abstractly especially in the collection and analysis of numerical data
K(P13)	Ethical and Intercultural Perspectives	Ethical perspectives of the different viewpoints someone uses to view a problem in the context of individual human values

TABLE III. LEVELS OF COGNITIVE SKILLS BASED ON BLOOM'S TAXONOMY [6]

Skill Level		Definition	Verbs
Code	*Name*		
B-I	Remembering	Exhibit memory of previously learned materials by recalling facts, terms, basic concepts, and answers.	Choose, Define, Find, How, Label, List, Match, Name, Omit, Recall, Relate, Select, Show, Spell, Tell, What, When, Where, Which, Who, and Why
B-II	Understanding	Demonstrate understanding of facts and ideas by organizing, comparing, translating, interpreting, giving descriptions.	Classify, Compare, Contrast, Demonstrate, Explain, Extend, Illustrate, Infer, Interpret, Outline, Relate, Rephrase, Show, Summarize, and Translate
B-III	Applying	Solve problems in new situations by applying acquired knowledge, facts, techniques, and rules in a different way.	Apply, Build, Choose, Construct, Develop, Experiment, with, Identity, Interview, Make, use, of, Model, Organize, Plan, Select, Solve, and Utilize
B-IV	Analyzing	Examine and break information into parts by identifying motives or causes. Make inferences and find evidence to support solutions.	Analyze, Assume, Categorize, Classify, Compare, Conclusion, Contrast, Discover, Dissect, Distinguish, Divide, Examine, Function, Inference, Inspect, List, Motive, Relationships, Simplify, Survey, Take part in, Test for, Theme
B-V	Evaluating	Present and defend opinions by making judgments about information, validity of ideas, or quality of material.	Agree, Appraise, Assess, Award, Choose, Compare, Conclude, Criteria, Criticize, Decide, Deduct, Defend, Determine, Disprove, Estimate, Evaluate, Explain, Importance, Influence, Interpret, Judge, Justify, Mark, Measure, Opinion, Perceive, Prioritize, Prove, Rate, Recommend, Rule on, Select, Support, Value
B-VI	Creating	Compile information together in a different way by combining elements in a new pattern or by proposing alternative solutions.	Adapt, Build, Change, Choose, Combine, Compile, Compose, Construct, Create, Delete, Design, Develop, Discuss, Elaborate, Estimate, Formulate, Happen, Imagine, Improve, Invent, Make up, Maximize, Minimize, Modify, Original, Originate, Plan, Predict, Propose, Solution, Solve, Suppose, Test, Theory

F. Coding Process

Coding the learning objectives and other information in course syllabi was an iterative process and required a deep understanding of both the initial codes and the data. Initially, relevant information from the syllabi was fetched as excerpts. These excerpts were then mapped to the appropriate CC2020 competency model component. An excerpt could be mapped to more than one competency component. In the rest of this subsection, we illustrate how each CC2020 component was coded with examples.

1) Coding Knowledge Elements: As illustrated in Fig. 2, excerpts from the course's learning objectives were mapped to both initial knowledge elements. The first excerpt "software development processes" was mapped to technical knowledge element K(C4.2), i.e., "Software Process". The sixth excerpt "Quality assurance" was mapped to technical knowledge element K(P11), i.e., "Quality Assurance/Control".

TABLE IV. LEARNING DISPOSITIONS [6]

Disposition Element		Elaboration
Code	*Name*	
D-1	Proactive	With Initiative/Self-Starter Shows independence. Ability to assess and start activities independently without needing to be told what to do. Willing to take the lead, not waiting for others to start activities or wait for instructions.
D-2	Self-Directed	Self-motivated/ Self-Directed Demonstrates determination to sustain efforts to continue tasks. Direction from others is not required to continue a task toward its desired ends.
D-3	Passionate	With Passion/Conviction Strongly committed to and enthusiastic about the realization of the task or goal. Makes the compelling case for the success and benefits of task, project, team, or means of achieving goals.
D-4	Purpose-Driven	Purposefully engaged / Purposefulness Goal-directed, intentionally acting and committed to achieving organizational and project goals. Reflects an attitude towards the organizational goals served by decisions, work, or work products. e.g., Business acumen.
D-5	Professional	With Professionalism / Work Ethic. Reflecting qualities connected with trained and skilled people: Acting honestly, with integrity, commitment, determination, and dedication to what is required to achieve a task.
D-6	Responsible	With Judgement / Discretion / Responsible / Rectitude Reflect on conditions and concerns, then act according to what is appropriate to the situation. Making responsible assessments and taking action using professional knowledge, experience, understanding, and common sense. E.g., Responsibility, Professional astuteness.
D-7	Adaptable	Adaptable / Flexible / Agile Ability or willingness to adjust approach in response to changing conditions or needs.
D-8	Collaborative	Collaborative Team Player / Influencing Willingness to work with others, engaging appropriate involvement of other persons and organizations helpful to the task. Striving to be respectful and productive in achieving a common goal.
D-9	Responsive	Responsive/Respectful Reacting quickly and positively. Respecting the timing needs for communication and actions needed to achieve the goals of the work.
D-10	Meticulous	Attentive to Detail Achieves thoroughness and accuracy when accomplishing a task through concern for relevant details.
D-11	Inventive	Exploratory / Inventive Looking beyond simple solutions; Examining alternative ideas and solutions. Seeks, produces, and integrates appropriate alternative

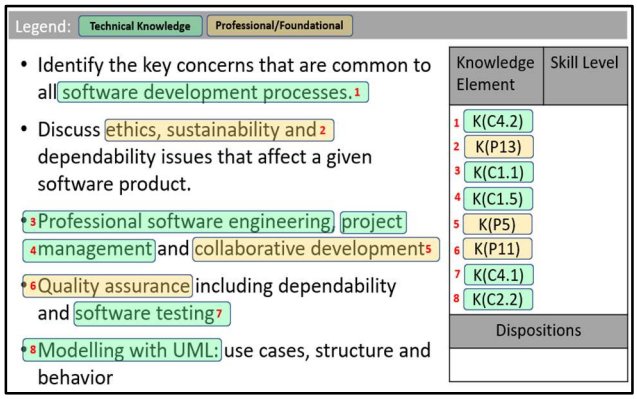

Fig. 2. Example showing the process of coding knowledge elements where Technical Knowledge elements are colored in Green and Foundational and Professional Knowledge in Yellow. Notice the excerpts (left) and their codes (right) are matched by index numbers.

2) Coding Skill Level: After the learning objective was mapped to a knowledge element, a verb was searched within the excerpt to signify the skill level for the knowledge element. When no verb could be mapped to any skill level, then the knowledge element would have no skill level attached to it. When multiple skill levels were found for a single knowledge element, the maximum skill level was selected. The verbs searched for in each skill level are highlighted in Table III. Fig. 3 shows the process of adding skill level to the previous example.

3) Coding Dispositions: As illustrated in Fig. 4, excerpts from the course's learning objectives were mapped to the 11 dispositions. The excerpt "professional software engineering" was mapped to the disposition D-5, i.e., "Professional". As shown by this example, an excerpt for a disposition (D-5, Professional, Table IV) could also overlap with the excerpt of a knowledge element (K(C1.1), Social Issues and Professional Practices, Table I).

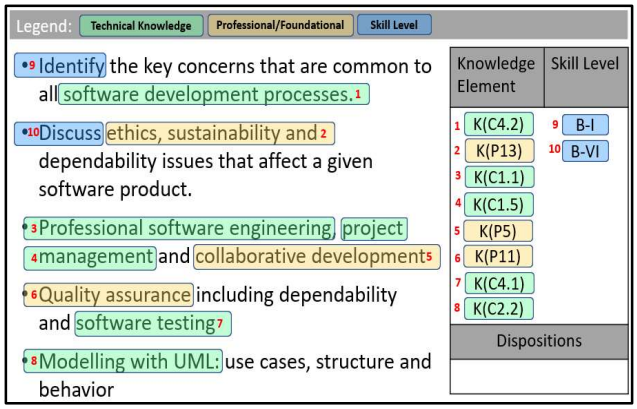

Fig. 3. Example showing the process of coding skill levels which are shown in Blue. Notice the excerpts (left) and their codes (right) are matched by index numbers.

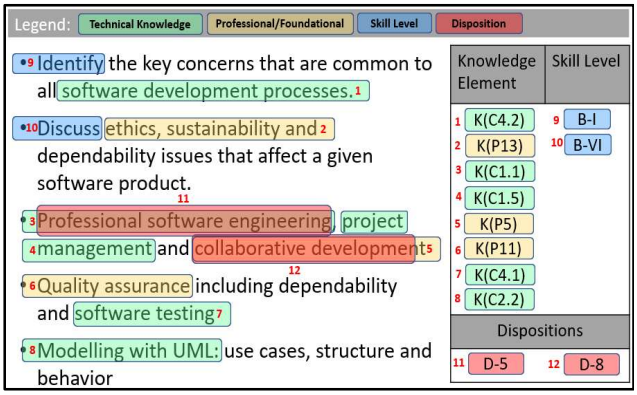

Fig. 4. Example showing the process of coding dispositions which are shown in Red where Excerpt colored in red are mapped to Disposition. Notice the excerpts (left) and their codes (right) are matched by index numbers.

III. Result And Discussion

The thematic analysis of the 31 courses yielded 187 technical knowledge element excerpts and 146 professional/foundational knowledge element excerpts. From these we have identified the following five main themes about software engineering project courses:

1. It's all about "Software Development".
2. "Software Fundamentals" are rarely covered.
3. Professional and foundational knowledge are key.
4. Skill level is often not specified.
5. Only two out of the eleven dispositions are specified.

A. Theme I: It's All About "Software Development"

Probably unsurprisingly, the technical knowledge element category "Software Development" is covered by all the courses. The knowledge elements under this category covered more than half of the technical knowledge excerpts (see Fig. 5), with all of them occurring in nearly all courses, except for "Platform Development" which occurs in none.

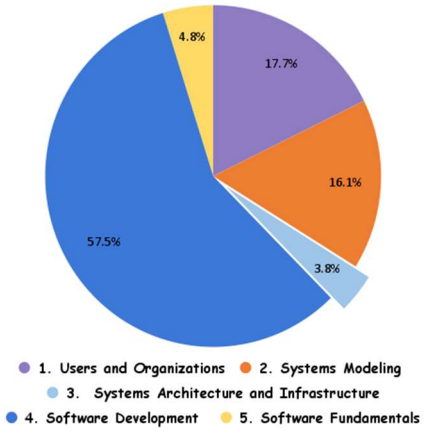

Fig. 5. Percentage of excerpts that belong to each category of technical knowledge elements.

"Software Quality, Verification, and Validation" is the most common knowledge element, occurring in all but one course. This shows how project-based software engineering courses focus on a form of software quality assurance. Several testing types and techniques have been mentioned. An example from the excerpts is shown below:

> *"Discuss various testing techniques such as white box and black box testing, Distinguish between different types and levels of testing (for instance, unit, integration, systems, and acceptance)"* - Course 4

Some courses mentioned giving students hands-on experience in writing tests, bug finding, code inspection, code reviews, etc. Another example from the excerpts is as follows:

> *"Gain experience writing tests, testing (functional testing, structural testing, testing strategies), code refactoring, and debugging."* - Course 1

"Software Process" is another knowledge element that occurs in most courses (83%) (See Table V). Courses often highlight giving students understanding and guidance in various software development lifecycle (SDLC) techniques and phases. The most common ones are agile, scrum, and waterfall. Examples from the excerpt are:

> *"Describe and compare the various mainstream software development methods, examines the software development life cycle, Agile methodologies"*- Course 8

> *"Guide the student through the waterfall development model"* - Course 30

"Software Modeling and Analysis" and "Software Design" are other high-occurring knowledge elements. This is probably also not surprising in project-based courses. The learning objectives highlight knowledge in software analysis, design, development, and implementation. Software modeling techniques are also covered, especially Unified Modeling Language (UML). Here is an example from the excerpts:

> *"Software design… Modeling with UML… Select appropriate process models, approaches, and techniques to manage a given software development process and justify the choices, Identify the key concerns that are common to all software development processes."* - Course 11

B. Theme II: "Software Fundamentals" Are Rarely Covered

Although technical knowledge elements under "Software Fundamentals" such as "Programming Languages", "Data Structures and Algorithms" etc., are essential for a successful software project, they are rarely found in a project-based software engineering course. Only about 5% of the technical excerpt maps to this category (See Fig. 4). The reason might be that many software engineering courses have prerequisites that cover these fundamentals. This allows students to have sufficient prior computing fundamental knowledge needed for a successful software project.

TABLE V. OCCURRENCE OF TECHNICAL KNOWLEDGE ELEMENTS

Category	Computing Knowledge Area[a]		Percentage
	Code	*Name*	
1. Users and Organizations	K(C1.1)	Social Issues and Professional Practice	41.94%
	K(C1.3)	IS Management and Leadership	3.23%
	K(C1.4)	Enterprise Architecture	0.00%
	K(C1.5)	Project Management	54.84%
	K(C1.6)	User Experience Design	6.45%
2. Systems Modeling	K(C2.1)	Security Issues and Principles	9.68%
	K(C2.3)	Requirements Analysis and Specification	87.10%
3. Systems Architecture and Infrastructure	K(C3.1)	Virtual Systems and Services	3.23%
	K(C3.6)	Embedded Systems	9.68%
	K(C3.9)	Security Technology and Implementation	9.68%
4. Software Development	K(C4.1)	Software Quality, Verification, & Validation	96.77%
	K(C4.2)	Software Process	83.87%
	K(C4.3)	Software Modeling and Analysis	77.42%
	K(C4.4)	Software Design	87.10%
	K(C5.3)	Data Structures, Algorithms, & Complexity	9.68%
	K(C5.4)	Programming Languages	9.68%
	K(C5.5)	Programming Fundamentals	9.68%

[a.] Missing knowledge elements have 0% occurrence.

Three courses highlight their learning objective to cover computer algorithm principles. All three occurrences are at the "Applying" skill level, indicating that students are given the opportunity to use their previous algorithm knowledge in a software project. An example from the excerpts is:

> *"Gain experience in applying various Computer Science methods and algorithms, as learned in earlier courses, to large-scale software development."* - Course 18

Two of the three courses that cover the "Programming Languages" knowledge element cover C++ programming and Python scripting, which are necessary for the completion of their respective software projects. The other course focuses on a not-so-traditional learning language Ruby on Rails. See the excerpt below:

> *"Apply the key ideas of learning a new framework to construct and deploy simple Rails applications, Apply the key ideas of learning a new language in order to construct programs in Ruby"* Course 7

Other programming fundamentals that are also found are design patterns, object-oriented programming, and other programming techniques using Java and Typescript.

C. Theme III: Professional and Foundational Knowledge Are Key

Knowledge of the computing discipline alone cannot adequately educate graduates for successful careers. While disciplinary knowledge sets computer experts apart from other professionals, there are many additional knowledge domains that are foundational, or normative in society and the workplace, as opposed to only technical [6]. The result proves just that, as professional and foundational knowledge has nearly as much excerpt as technical knowledge excerpts (145).

In nearly all the courses, there is a focus on some kind of communication, either "Written Communication" or "Oral Communication & Presentation". Written communication usually comes in the form of a project report and documentation. Oral communication often comes as milestone/final presentations, meetings, and interviews. An example excerpt is shown as follows:

> *"Students are expected to demonstrate the ability to understand and document customer requirements and design a working product within given constraints, The ability to communicate effectively with the development team and all stakeholders."* - Course 24

TABLE VI. OCCURRENCE OF PROFESSIONAL & FOUNDATIONAL KNOWLEDGE ELEMENTS

Professional and Foundational Knowledge		**Percentage**
Code	***Name***	
K(P1)	Oral Communication & Presentation	64.52%
K(P2)	Written Communication	64.52%
K(P3)	Problem-Solving and Troubleshooting	12.90%
K(P4)	Project and Task Organization and Planning	93.55%
K(P5)	Collaboration and Teamwork	90.55%
K(P6)	Research and Self-Starter/Learner	3.23%
K(P7)	Multi-Task Prioritization and Management	3.23%
K(P8)	Relationship Management	0%
K(P9)	Analytical and Critical Thinking	12.90%
K(P10)	Time Management	9.68%
K(P11)	Quality Assurance / Control	83.87%
K(P12)	Mathematics and Statistics	3.23%
K(P13)	Ethical and Intercultural Perspectives	23.58%

Other frequently used professional and foundation knowledge elements include "Project and Task Organization and Planning", "Teamwork and Collaboration", and "Quality Assurance / Control". A small number of courses focus on problem-solving and troubleshooting skills, analytical and critical thinking skills, and ethical & intercultural perspectives. An example from the courses is shown below:

> *"Understanding of professional, ethical, legal, social issues and responsibilities."* - Course 20

Time management is surprisingly less emphasized, having appeared in just three courses. Software engineering requires effective time management since it has a direct bearing on project outcomes and team morale. One of the examples from the excerpts is as follows:

> *"Gain experience in group-based software development and develop communication, planning, and time-management skills."*- Course 18

D. Theme IV: Skill Level is Often Not Specified

Our thematic analysis yielded a combined 331 technical, professional, and foundational knowledge elements. About 54% of them have no associated skill level specified (i.e., no verb in the excerpt can be mapped to Bloom's taxonomy) as illustrated in Table VII.

TABLE VII. KNOWLEDGE ELEMENTS WITH SKILL LEVEL SPECIFIED

Knowledge	***Count***	***No of Skill***	**Percentage**
Technical	186	70	37.63%
Professional/Foundational	145	80	55.17%
Total	331	150	45.32%

E. Theme V: Only Two Out of the Eleven Dispositions are Specified.

A well-structured competency needs to have a disposition, which distinguishes a competency from a learning outcome in an obvious way. As a result, it significantly increases the expressiveness of learning goals and adds language that is typical of professional expectations [6]. Only two out of the 11 dispositions provided by CC2020 ("Collaboration" and "Professional") are found in the software engineering courses.

The collaboration disposition is found in nearly all (90%) of the courses. Students perform the course project in groups of varying sizes ranging from small to large. Because software development is a complicated process requiring the involvement of many stakeholders, collaboration is an essential part of software engineering. An example from the excerpts is:

> *"Collaborative development: Students have worked collaboratively in a team to implement and fully test detailed designs and code. An ability to function effectively on teams to accomplish a common goal."* - Course 10

Another crucial component of software engineering is being "Professional" since it guarantees that software engineers uphold a high standard of quality and moral behavior. Professionalism includes a variety of abilities and conduct, such

as effective communication, time management, moral conduct, dedication, and determination. One-third of the courses specified professional disposition. An example from the excerpts is given below:

> *"Demonstrate appropriate professional conduct. Discuss professional codes of conduct of Computer Scientists and Engineers."* - Course 17

IV. Conclusion

As the PjBL approach becomes more widely used in software engineering courses, it is important for educators to understand the commonly used learning objectives in such project-based courses and to better design projects that effectively target the appropriate course learning objectives. This study utilizes thematic analysis to perform a case study where learning objectives/outcomes in existing project-based software engineering courses were mapped to the CC2020 competency model. The learning objectives were categorized according to knowledge elements, skills, and dispositions as defined in the CC2020 competency model.

The major contributions of this study include the following:

- A summary of commonly used course learning objectives is presented.
- We show that 54% of knowledge elements from the reviewed course learning objectives do not have any skill level specified.
- We highlight that only two out of the eleven CC2020 dispositions ("Collaboration" and "Professional") are specified in the surveyed project-based software engineering courses.

Furthermore, the findings of this study have implications for project and course design. For example, the summary of commonly used course learning objectives is available for future instructors to consider when designing their own course projects, enabling them to target project-based software engineering courses more effectively to learning objectives. On the other hand, the gaps identified in this study can help instructors construct higher-quality course learning objectives and competency statements.

This study could be limited by the relatively small number of courses surveyed (31). Future research could consider a larger number of project-based courses within computing education. This broader coverage would further increase the validity of this study and enable comparison across different disciplines in computing education.

Acknowledgment

This work is partially supported by the U.S. National Science Foundation Awards DUE-2111318 and DUE-2111294.

References

[1] D. Kokotsaki, V. Menzies, and A. Wiggins, "Project-based learning: A review of the literature," *Improving Schools*, vol. 19(3), pp. 267–277, 2016. https://doi.org/10.1177/1365480216659733

[2] D. R. Krathwohl, "A Revision of Bloom's Taxonomy: An Overview, Theory Into Practice," *Theory into Practice* vol. 41(4), pp. 212-218, 24 June 2002. https://doi.org/10.1207/s15430421tip4104_2

[3] A. Churches, "Bloom's taxonomy blooms digitally," *Tech & Learning*, vol. 1, pp 1-6. 2008.

[4] J. Stayanchi, "Higher order thinking through Bloom's taxonomy," *Kwansei Gakuin University Humanities Review*, vol. 22, pp. 117-124, 2007.

[5] Abosalem, Y., 2016. Assessment techniques and students' higher-order thinking skills. International Journal of Secondary Education, 4(1), pp.1-11.

[6] CC2020 Task Force, "Computing Curricula 2020: Paradigms for Global Computing Education," *Association for Computing Machinery*, New York, 2020.

[7] The Joint Task Force for Computing Curricula 2005, "Computing Curricula 2005", *ACM, IEEE CS, IEEE SC*, 2005.

[8] A. Clear, T. Clear, A. Vichare, T. Charles, S. Frezza, M. Gutica, B. Lunt, F. Maiorana, A. Pears, F. Pitt, and C. Riedesel, "Designing computer science competency statements: A process and curriculum model for the 21st century" *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education*, pp. 211-246, 2020. https://doi.org/10.1145/3437800.3439208

[9] Joint Task Force on Computing Curricula, "Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science," *Association for Computing Machinery (ACM) and IEEE Computer Society*, 2013.

[10] D. S. Bowers, M. Sabin, R. K. Raj, and J. Impagliazzo, "Computing Competencies: Mapping CC2020 Dispositions to SFIA Responsibility Characteristics," *IEEE Global Engineering Education Conference (EDUCON)*, Tunis, Tunisia, pp. 428-437, 2022. https://doi.org/10.1109/EDUCON52537.2022.9766565

[11] The SFIA Foundation, "SFIA8: All skills A–Z," 2021. https://sfia-online.org/en/sfia-8/all-skills-a-z

[12] V. Braun, and V. Clarke, "Using thematic analysis in psychology" *Qualitative Research in Psychology*, vol. 3(2), pp. 77–101, 2006. https://doi.org/10.1191/1478088706qp063oa

[13] V. Clarke, V. Braun, and N. Hayfield, 2015. "Thematic analysis," *Qualitative psychology: A practical guide to research methods*, vol. 3, pp. 222-248.

[14] S. Frezza, T. Clear, and A. Clear, "Unpacking Dispositions in the CC2020 Computing Curriculum Overview Report," *IEEE Frontiers in Education Conference (FIE)*, Uppsala, Sweden, pp. 1-8, 2020. https://doi.org/10.1109/FIE44824.2020.9273973