Authentic Individual Assessment for Team-based Software Engineering Projects

Laura Tubino
Deakin Learning Futures
Deakin University, Geelong
Geelong, Victoria
laura.t@deakin.edu.au

Andrew Cain, Jean-Guy Schneider
Dhananjay Thiruvady
Niroshinie Fernando
School of Information Technology
Deakin University, Geelong
Geelong, Victoria
{andrew.cain,jeanguy.schneider}@deakin.edu.au
{dhananjay.thiruvady,niroshinie.fernando}@deakin.edu.au

ABSTRACT

In order to give students an authentic learning experience and better prepare them for the life-long learning required in contemporary workplaces, educational institutions increasingly use project-based learning in teams. However, this poses the challenge of developing authentic and equitable assessment criteria that reflect individual contributions in a team-work setting without jeopardizing project outcomes. We present a novel and innovative portfolio-based assessment framework that focuses on qualitative outcomes and ensures that the level of achievement reflects a student's competency across each of the defined learning dimensions, including professional behaviour, teamwork, process and relevant contributions towards project deliverables. In this paper, we present the main motivation behind devising and introducing the framework and also reflect on the educational outcomes and challenges of implementing the framework in the context of two final year Software Engineering project units.

ACM Reference Format:
Laura Tubino, Andrew Cain, Jean-Guy Schneider, Dhananjay Thiruvady, and Niroshinie Fernando. 2020. Authentic Individual Assessment for Team-based Software Engineering Projects. In *Software Engineering Education and Training (ICSE-SEET'20), May 23–29, 2020, Seoul, Republic of Korea.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3377814.3381702

1 INTRODUCTION

The gap between software education and software industry's expectations of new hires is widely recognised [1, 23, 43]. One way in which higher education is addressing this, is the inclusion of project-based learning (PBL) units in Software Engineering degrees, which in fact has widely increased as Universities focus on better aligning academic curricula with industry expectations [1, 6, 19, 31].

PBL prepares students to solve real-world, ill-structured problems, improving students outcomes on conceptual understanding, problem solving and metacognitive skills [33] and giving them insights into technology, product development, and teamwork [11]. Although PBL units provide many benefits, working with real-world, ill-structured problems can also bring many challenges.

Challenges for curriculum design and assessment, reported by PBL supervisors, are (i) writing objectives, (ii) using appropriate assessment strategies, (iii) applying sound assessment practices, (iv) sampling material appropriately, and (v) controlling for mismeasurement of student achievement [28]. Design and adoption challenges such as directing students to do quality work, and monitoring and providing assistance to teams have also been widely reported [35]. These observations are aligned with those reported in the Computer Science capstone unit literature, which frequently involves PBL, with a survey on capstone unit issues identifying learning goals and student evaluation as two of the six most common themes in the literature [19].

In our Postgraduate Software Engineering capstone project units, which involve PBL coupled with industry collaboration, we have come across the same challenges reported above. Due to the scale of our capstone units, which currently involve app. 300 students and 30 supervisors, we decided that the best approach to solve these issues was to focus on assessment, building on our understanding that assessment drives learning and students' actions [5, 34].

The assessment framework we propose in this paper is motivated by the need to support, in particular, the added complexity that comes from working in teams on real-world, ill-structured problems. The assessment framework is designed to guide students' behaviour on the project and facilitate supervisors' work on monitoring and supporting teams. It also enables individual contributions in a team to be identified and rewarded, and encourages teamwork while avoiding any extra work, unrelated to the project, thus concentrating teams' efforts on the delivery of a functional Software Engineering artefact.

The rubric design we propose as part of the assessment framework includes twelve assessment criteria which we consider essential for team projects to be successful. These are divided into three broad categories: (i) basic, defining the criteria students need to comply with in order to demonstrate engagement with the project, (ii) core, referring to project practices that need to be enacted, and (iii) advanced, referring to adaptability, or approaches that allow teams to respond to novel or changing situations. For each assessment criteria, we propose defining qualitatively different levels,

ICSE-SEET'20, May 23–29, 2020, Seoul, Republic of Korea

ACM ISBN 978-1-4503-7124-7/20/05…$15.00
https://doi.org/10.1145/3377814.3381702

guided by affective and cognitive characteristics of three types of students commonly observed in higher education. This rubric is coupled with a Task-Oriented Portfolio Assessment model [13] underlined by constructive alignment principles [5]. This model consists of "outcome-based" grading, frequent formative feedback and delayed summative assessment.

The rest of this paper is organized as follows: in Section 2 we discuss related work. In Section 3, we describe our proposed assessment framework, defining its different components and their interactions, in a way that can be applied into different PBL experiences across a Software Engineering degree. We present the results of a preliminary evaluation of our framework in a capstone context in Section 4 and highlight the lessons learned in Section 5. We conclude the paper in Section 6 with a summary of the main observations and an outlook on future work.

2 RELATED WORK

In this section, we briefly review the goals of using PBL in Software Engineering units and the approaches to assessment which are commonly used in PBL, helping us to inform our proposed assessment framework. Constructive alignment and an associated assessment model, the task oriented portfolio assessment (TOPAM), are also presented, as forming the basis of our assessment model.

2.1 Reasons for using Team Projects in Software Engineering

The main reasons for incorporating PBL strategies into a Software Engineering unit is to better prepare students for contemporary workplaces and for the life-long learning required to succeed in these. PBL complements classroom education by providing authentic Software Engineering experiences as a context where students cannot only apply the concepts learned in the classroom and further develop their skills, but also where they can work collaboratively and develop adaptability, essential to their professional life [3].

PBL helps students apply their learning to realistic problems, improving motivation, helping integrate and consolidate their knowledge, and highlighting knowledge gaps [10, 33, 42]. In addition, studies have shown that PBL units have a strong impact on affective learning outcomes [32, 33], with positive impact on students' attitudes towards Software Engineering topics such as Agile practices, project planning and technical challenges [14, 27, 32, 37, 39], respectively.

In PBL, through working in teams, students apply and further develop communication skills, teamwork skills and a better understanding of issues of professional practice [10, 30, 42]. In addition, by presenting students with real-world, ill-structured problems, PBL gives students the opportunity to develop and demonstrate their ability to effectively perform in a novel or dynamic and changing situation, reflecting adaptability and life-long learning skills.

2.2 Assessment for Team Projects

There are a plethora of assessment models being applied to team projects. The main things to consider when designing team project assessments include what will be graded and how to allocate a grade fairly to each member within the team. In the following paragraphs, we will discuss some of the key related work in this area.

In terms of what will be graded, it is common to focus on the process, the final product, or a combination of these. Typical submissions for the assessment of process include artefacts such as proposals, plans, literature reviews, final report and oral presentations [19, 25, 36]. The issue with these assessment tasks is that teams usually end up developing very good documentation, but not a good product [15]. In this respect, assessing different aspects of the project through smaller artefacts or tasks can give indications of the process without jeopardising project outcomes. Select project artefacts that can be used as authentic assessment tasks for this purpose are collaboration tools, version management tools, activity logbooks, meeting minutes, and progress reports [19].

Another common approach is to focus on the final product [15, 20]. However, this is likely to result in the product becoming the central focus of the unit, as opposed to the knowledge and skills gained by producing it [33]. Assessment of techniques, as opposed to product features, has been trialed by a teaching team in response to the perception that Software Engineering students were following hacks to achieve a good product instead of following best practices [20]. Using a combination of assessment tasks that include product features or the final product, process artefacts and sometimes even a final exam is very common [19, 20, 36].

In terms of allocating a fair grade to each team member, a team grade can be allocated to every student, or it can be scaled using self-evaluation, peer-evaluation and/or instructor-evaluation to determine the contribution of each member and assign a grade that better reflects each student's engagement [36]. Scaling the grade so each member is graded according to their contribution seems fair, as it has been observed that within a group some students can spend two to five times more time working on the project [36], and that self-evaluation and peer-evaluation are usually aligned to instructor observations [19]. On the other hand, instructor observations are sometimes labeled as biased and supervisors are excluded from grading students in the name of objectivity [25].

In line with the principles of constructive alignment, which we describe in Section 2.3, whichever assessment task is used must provide evidence of each student's achievement of specific learning outcomes at different levels. For this reason, it is essential that assessment tasks ensure that individual contributions can be identified and evidence individual achievement of the different learning outcomes. Neither team-produced artefacts nor any of the methods for scaling a team grade could enable this to be achieved. Instead, focusing on smaller artefacts or tasks such as collaboration tools, version management tools, activity logbooks, meeting minutes and progress reports [19, 25] would better meet constructive alignment principles of identifying contributions that can demonstrate individual achievement of specific learning outcomes.

2.3 Constructive Alignment

Constructive alignment [5] provides a set of principles designed to elicit quality learning from students. These principles combine a constructivist view of learning, with an aligned approach to curriculum and assessment. The constructivist view of learning places the emphasis on the student in building their own knowledge and skills, requiring students to be actively engaged with their learning, as knowledge cannot be developed through communication alone.

PBL qualifies well as a constructivist method, as students construct meaning by working on a project, rather than absorbing knowledge presented to them.

In constructive alignment, the constructivist view is coupled with aligned curriculum, where student activity and assessment are focused around learning outcomes expressed using active verbs designed to require cognitive levels appropriate to the unit and its context. In this way, assessment helps motivate students to undertake the necessary activity in order for them to achieve the learning outcomes.

The principles of constructive alignment stem from a systems perspective of teaching and learning, with assessment acting as the key driver of students. Educators wanting to impact student behaviour are therefore best directed to focus on assessment, and to ensure that this assessment elicits the kinds of behaviours and activities that are expected to help students attain the required learning. The benefits of constructive alignment have been reported in many contexts, including discussions on the need for constructive alignment in Software Engineering education [2], demonstrations of its application across programming and software architecture [12], as well as in designing real-life Software Engineering projects [17].

2.4 Task-Oriented Portfolio Assessment

Learning systems based upon constructive alignment aim to utilise a student centred view and aligned curriculum to provide an environment that encourages and rewards students for achieving learning outcomes. Task-Oriented Portfolio Assessment model (TOPAM) [13] is one such learning system that has been applied successfully to teaching within Computer Science and Software Engineering [12]. This model actions constructive alignment through assessment of a portfolio of work generated by student engagement with unit tasks during the teaching period.

Central to TOPAM is the use of frequent formative feedback, criterion referenced assessment, and delayed summative grading. These principles aim to strengthen the application of constructive alignment by aligning student activity with assessment criteria that define quality standards for achievement at each grade standard (*e.g.*, Pass, Credit, Distinction and High Distinction)[1]. Assessment criteria provide students with clear qualitative standards to achieve, with tasks scaffolding learning to help ensure each student is likely to achieve the standard associated with their own grade aspirations.

Delayed summative grading aims to ensure that student grades, the focus of student attention, are calculated from a single assessment *after* the conclusion of the teaching period. This enables work during the teaching period to be formative, with students able to highlight issues they have with their understanding without fear of a mark-based penalty. At the same time, this delayed summative grading also reduces numeric game playing where students potentially skip low weighted items due to a poor perceived payback. Instead, students' focus is directed to the quality standards required for each grade, and the tasks associated with helping them achieve this standard. As a result, the specification of quality attributes associated with each grade outcome become a critical component in unit learning and assessment design.

[1]These levels of achievement are typical for a Higher Education institution in Australia and New Zealand.

3 PROPOSED ASSESSMENT FRAMEWORK

The assessment framework we propose aims to ensure students achieve the learning goals of PBL by providing a template for designing assessment that supports students through the complexity of working in a team on ill-structured problems.

The rubric resulting from this framework focuses students' attention on the main elements that each of them needs to concentrate on to successfully complete the project with their team, allowing for individual assessment in a team environment. In fact, rubrics should not only be used as a grading tool but, more importantly, as a self-assessment tool, directing students' behaviour and informing them of the expectations, thus ensuring they understand what evidence they need to provide to demonstrate their achievements. Rubrics also serve to make students aware of what the next step looks like, directing them to strive for it.

The accompanying assessment model we propose is based on the TOPAM [13], which is an "outcome-based" approach building upon constructive alignment that includes frequent formative feedback and delayed summative grading. Within this model, the tasks and rubric are used to direct student engagement with the project, with the frequent formative feedback loop helping ensure staff and students alike are aware of progress and individual contributions.

In this section, we describe the core components of our proposed assessment framework, and their interactions in helping guide student behaviour within PBL units.

3.1 Defining Assessment Criteria

In our assessment framework, twelve criteria are presented we consider as essential for team projects to be successful. Together, these criteria enable individual assessment while requiring teamwork, and at the same time, encouraging students to improve their teamwork, process, and technical knowledge, and skills, respectively.

The twelve criteria are divided into three broad categories: **basic**, **core**, and **advanced**, and are presented at the left hand side of the rubric shown in Figure 1. Individually, each criteria is designed to guide one aspect of students' behaviour. In the rubric, descriptors across the grade range indicate how each assessment criteria can be demonstrated at different achievement levels.

Within the rubric, the basic assessment criteria broadly relate to **complying** with basic project practices and address *behaviour* rather than skills or knowledge. The criteria include **professional behaviour**, **accountability**, **attendance**, and **engagement**. Their aim is for students to demonstrate engagement with the project. These criteria exist only at the lower grade levels, as they are setting the foundation for the project work with the focus for assessment moving to criteria more specifically related to the project work.

At the core of the rubric are criteria related to **enacting** effective project practices, and relate to knowledge and skills developed throughout their Software Engineering degree. The criteria are defined around **communication**, **teamwork**, **process**, **product**, and **Software Engineering skills**. Together these criteria relate to each student's ability to engage with stakeholders, facilitate and participate in teamwork activities, engage with appropriate project processes, contribute to product outcomes, and demonstrate the achievement of technical Software Engineering knowledge and skills. These criteria relate to work within the project, with most criteria extending up to the High Distinction category.

The advanced criteria within the rubric then relate to the criteria that require **adaptability**. Here students need to demonstrate their ability to effectively perform in a novel or dynamic and changing situation by adapting their approach to the project and teamwork components, based upon their analysis of factors influencing the project in their specific context. This includes **self-awareness**, **project management** and **mentoring** criteria. To demonstrate achievement of these dimensions, students need to display metacognitive skills. These students are needed as leaders in their teams, setting the vision. If the project is ill-structured and ambiguous, and no student in a team is at this level, then the supervisor needs to support the team or take on this role.

3.2 Levels of Achievement

It is important to consider our own students when designing a rubric in order to ensure that expectations described in the rubric match our students ability and learning potential. Here we will describe three types of students commonly observed in higher education. The characteristics of these three student groups align well with the characteristics of people described in Kegan's constructive-developmental framework as operating in the second, third and fourth developmental stages, whose distribution matches that of the higher education population [21].

The first group is composed of students who do not feel responsibility towards others and make decisions mainly based on immediate benefits or consequences. These students fail to plan for the long-term and are those we say "can do allocated work with guidance." Therefore, this group needs authority figures or rules to direct them. This first group of students can also be said to align with the lower level of Bloom's affective taxonomy: receiving [22], and thus will accept responsibilities, but not seek them. When it comes to the cognitive dimension, we expect these students to apply their skills and knowledge, or to operate at the third level of Bloom's cognitive taxonomy [7].

The second group is formed by students who take others into account and consider long-term consequences of their actions. These students understand other's points of view, even if different from their own, and can take on specific roles, under the guidance of others. These students are those we say "perform their role in the project." Students in this group are good team members but will probably not perform well in leadership roles without support as they need others as sources of validation, orientation and authority. This group of students can also be said to align with the second level of Bloom's affective taxonomy: responding [22], and thus will assist and cooperate. When it comes to the cognitive dimension, we expect these students to not only apply but also analyse how their skills and knowledge need to be applied, or to operate at the fourth level of Bloom's cognitive taxonomy [7].

The last group is composed by students who have a big picture view. Students in this group can see things from different perspectives, which makes them able to identify needs, plan, coordinate and evaluate situations. We say these students "can see themselves in the team and assist others." This group of students can also be said to have internalised the expected values or attitudes, and demonstrate this by being proactive and self-motivated. They align with the third, fourth or fifth levels of Bloom's affective taxonomy: valuing, organising or characterising, respectively [22]. At the cognitive level, we expect these students to present high order thinking such as evaluating, synthesising and creating [7].

Based on the three type of students described above as being present in higher education, we have defined the following levels of achievement: Pass (P), Credit (C) and Distinction (D), matching the first, second and third type of student, respectively. The Fail (F) is awarded to students who fail to engage and contribute to the project; these students are described as exhibiting a "could not care less" attitude. The High Distinction (HD), on the other hand, is awarded to students who demonstrate the characteristics of the third type of student (D students) plus **excellence** in specific areas, as illustrated in Figure 1. We distinguish between Low-HD (achievement of a level of excellence in one assessment criterion) and High-HD (achievement of excellence in two or more criteria).

3.3 Defining the Grading Criteria that will drive Student Behaviour

Once the rubric assessment criteria (left column) and the levels of achievement (top row) are defined, descriptors for each criterion across the levels of achievement need to be developed. These descriptors guide students' behaviour and illustrate how to evidence the expected level of achievement. To describe how the levels of achievement are actualised under each criteria, the characteristics of each correlating student type need to first be considered. Next, the activities and tasks students will need to perform in the team-based Software Engineering project in which the assessment will be applied need to be identified. Finally, the tasks that can evidence the achievement of the different assessment criteria need to be mapped to these student types to describe the behaviour expected for each criteria at each level.

In order to illustrate our approach, we will discuss three examples, one for each category basic, core and advanced, to highlight how the grading criteria are applied.

As the first example, consider the first item under the basic category, namely **Professional Behaviour**, which is further divided into two criteria; (i) complying with the acceptable code of conduct as specified by unit staff, and (ii) impeding the project progress. As mentioned before, these criteria, being classified as basic, have the purpose of guiding student behaviour to comply with basic project practices and are not meant to impact on the assessment any further. This is evidenced by the fact that achievement level for criteria in the basic category are mainly at the P and C levels.

The first criterion under Professional Behaviour relates to courtesy and respect for others, as well as personal integrity. Being a fundamental aspect in project and team work, there is only one grade level of P for this criterion, failure to comply results in a F and can be determined by the unit chair at any time. For the second criterion, impeding the project progress, two grade levels can be assigned: P and C. To obtain a P, the student needs to demonstrate that they have not intentionally obstructed work in the project. Actions that led to obstruction of project tasks are allowed only if due to negligence and only once. To go beyond a P level and achieve

	Criteria	Fail	Pass	Credit	Distinction	High Distinction
		"could not care less"	*"can do allocated work with guidance"*	*"performs their role in the project"*	*"can see themselves in the team and assist others"*	*"can demonstrate excellence"*
		(must meet one criteria)	*(must meet all criteria)*	*(must meet all criteria)*	*(must meet all criteria)*	*(low: one of the below; high: two of the below.)*
Basic	Professional Behaviour	Fails to conduct self in a professional and respectful manner, based on issues raised to supervisors by project stakeholders and/or other students	Conducts self in a professional and respectful manner throughout the duration of the project	Conducts self in a professional and respectful manner throughout the duration of the project	Conducts self in a professional and respectful manner throughout the duration of the project	
		Willingly blocks or impedes the project team's progress	Blocks or impedes the project team's progress due to negligence no more than once through the project duration	Helps progress project tasks and collaborates with others to ensure work undertaken does not impede others	Helps progress project tasks and collaborates with others to ensure work undertaken does not impede others	
	Accountability (Worklogs)	Demonstrates less than 100 hours of project work during the teaching period (max. 30 hours of upskilling on core technologies) or not supervisor approval for less hours	Demonstrates at least 100 hours of project work during the teaching period unless has supervisor approval for less hours (max. 30 hours of upskilling on core technologies)	Demonstrates at least 120 hours of project work during the teaching period unless has supervisor approval for less hours (max. 30 hours of upskilling on core technologies)	Demonstrates at least 120 hours of project work during the teaching period unless has supervisor approval for less hours (max. 30 hours of upskilling on core technologies)	
	Attendance	Fails to attend at least 75% of all scheduled meetings (supervisor, team) in a timely fashion without supervisor approval	Attends >75% of all scheduled meetings (supervisor, team) in a timely fashion, or has supervisor approval not to attend	Attends >90% of all scheduled meetings (supervisor, team) in a timely fashion, or has supervisor approval not to attend	Attends >90% of all scheduled meetings (supervisor, team) in a timely fashion, or has supervisor approval not to attend	
	Engagement	Often fails to participate in team activities, shows little to no commitment to the project, or is unable to demonstrate or discuss progress made	Participates in team activities and is able to discuss individual work completed and its status	Actively involved in team activities, demonstrates commitment to the project, and is able to discuss how work completed fits into the project	Actively involved in team activities, demonstrates commitment to the project, and contributes to planning future activities	
Core	Communication	Regularly fail to respond to queries in a timely manner in accordance with team agreements; Use of inappropriate tools and/or language in communication	Respond to queries in a timely manner in accordance with team agreements; Use appropriate tools and language in communication	Demonstrate evidence of discussing project-relevant information with project stakeholders	Demonstrate evidence of successfully communicating key project-relevant information to external audiences	Outstanding engagement with key project stakeholders in order to clarify current project deliverables and/or direction
	Teamwork	Fails to accept tasks by either the team or the supervisor and/or copies other students work	Accepts and completes allocated work within the given timeframe, but does not show initiative	Plans and takes ownership of activities contributing to project progress	Plans and actively works to support other team members to achieve completion of project tasks	Outstanding support of other students in achieving their personal and project goals
	Process	Fails to demonstrate the ability to follow the process (i) documented by the team and (ii) required by the Unit (as outlined in the Unit Guide)	Demonstrates the ability to follow the process (i) documented by the team and (ii) required by the Unit (as outlined in the Unit Guide)	Demonstrates ability to accurately and effectively follow the process (i) documented by the team and (ii) required by the Unit (as outlined in the Unit Guide)	Demonstrates ability to effectively follow team processes; contribution to the definition and/or ongoing improvement of processes	
	Product	Fails to contribute towards any project deliverables (as given in the team's agreed project scope)	Contributes towards some project deliverables (as given in the team's agreed project scope)	Significant contribution towards several project deliverables (as agreed in the team's project scope) throughout the project	Consistent and significant contribution towards multiple project deliverables (as agreed in the team's project scope)	Exceptional, sustained contributions to the project (process or deliverables) or support for other squad(s)
		Quality of delivered work is below an acceptable standard	Quality of work delivered is at an acceptable standard	Quality of work delivered is consistently a high standard	Quality of work delivered is consistently of industry standard	Quality of work is consistently at a very high to exceptional standard
	Software Engineering Skills	Fails to demonstrate adequate Software Engineering knowledge or skills relevant to their discipline	Demonstrates Software Engineering knowledge and skills relevant to their discipline at a basic level	Demonstrates proficiency with Software Engineering relevant to their discipline and degree course	Demonstrates proficiency with Software Engineering relevant to their discipline and degree course	Demonstrates outstanding skills and/or knowledge of Software Engineering relevant to their discipline and degree course
Advanced	Self-Awareness	Fails to demonstrate awareness of personal characteristics	Identifies personal strengths and weaknesses	Identifies how personal strengths/weaknesses impact on project outcomes	Identifies ways to use strengths and address/improve on weaknesses to improve project outcomes	Identifies self and others strengths and weaknesses and plans/behaves in ways that improve project outcomes
	Project Management	Fails to demonstrates knowledge and basic application of core project management activities relevant to their discipline	Demonstrates knowledge and basic application of core project management activities relevant to their discipline	Can organize a small group of team members to successfully complete a project-relevant activity	Active participation in project scoping; prioritizing, planning, delegating and executing identified activities	Outstanding and sustained initiative in project scoping; prioritizing, planning, delegating, executing and monitoring activities
	Mentoring	Fails to evidence any mentoring activities	Provide basic guidance to other students in the project in at least one aspect relevant to the project	Can show evidence of mentoring of other team members an area relevant to the project	Demonstrates evidence that mentoring activities lead to improved project outcomes for other team members	Demonstrates evidence that mentoring activities lead to other team members becoming successful mentors

Figure 1: Assessment criteria presented as a rubric.

a C level, the student must further demonstrate having proactively taken steps to collaborate with other team members and ensuring that their individual work does not cause delays to the project.

Next, let us consider **Product**, which is under the core category, and is further divided into two criteria, one relating to the actual contribution of the student to the product, and the other to the quality of the contribution. Here both criteria are scaffolded across the different levels of achievement, from P to HD, based on increasing requirements.

In the first criterion, relating to individual contribution, a student who does not make any contributions to the project, will be assigned F. To achieve a P, the student must evidence that they contributed to at least some deliverables, aligning with the description of our first group of students, who need rules to direct them and accept responsibilities, but do not seek them. A student evidencing significant contributions, in several deliverables can achieve a C, matching the second group of students described above, expected to be responsible team members who assist and cooperate. To achieve a D, a student is required to have *consistently* made significant contributions, reflecting a proactive and self-motivated student. The HD level in this criterion mandates either exceptional or continuing individual contributions to the project, or providing significant support to students on other project teams.

In the second criterion, relating to the quality of individual contribution, a student whose contribution is below an acceptable standard would receive an F grade. To achieve a P grade, the student is required to have delivered an acceptable standard of work, this requires the student to apply knowledge and skills, as expected from our first group of students as described above. Work that is delivered consistently at high quality standard will be graded at C level, this requires some degree of analysis, not only application of knowledge and skills. A student who consistently evidences delivery of industry standard work will achieve a D level, demonstrating a certain degree of high order thinking such as evaluation. Exceptional quality of work will be awarded an HD grading.

The third and final example we will discuss is **Project Management**, part of the advanced category. This criterion is also scaffolded across all achievement levels, from P to HD. To obtain a P, the student is required to demonstrate knowledge and application of key project management activities at a basic level. This is expected of someone who is following orders and applying knowledge and skills, but not putting any additional effort on the project. Progressing further, if the student evidences organising a small sub-group of the team to complete a project activity, they can be awarded a C grade for the category, reflecting a student who is committed to the team and the project and willing to accept responsibility. A student who actively participates in a number of project management activities including scoping, prioritizing, planning, delegating and executing, will be demonstrating being proactive and having a big picture view, or high order thinking, and thus will achieve a D level. If a student demonstrates consistent participation of the aforementioned activities at outstanding level, coupled with monitoring their progress, they can be awarded an HD.

As explained in these examples, and as can be observed in Figure 1, the different levels of achievement are qualitatively different and are scaffolded such that the requirements in the higher levels implicitly encompass the requirement of lower levels. Therefore, by achieving a D in a particular assessment criteria, the student will be demonstrating the achievement of that assessment criteria at the D level and at all the levels below D (*i.e.* Credit and Pass).

The reader may note that the rubric presented in this paper (*cf.* Figure 1) was created for a capstone setting, in the final year of a Software Engineering related degree, considering the scope and constraints within this particular setting. It should thus be considered that while many of the specific descriptions presented for the different levels of achievement are transferable across a variety of settings, they can and should be adapted to suit the student types and constraints within the learning context that it is going to be used in.

3.4 Outcome-based Grading

To ensure they meet the breadth of skills required in a team project, students aiming for a grade between P and D need to demonstrate achievement of **all** assessment criteria at the level they are targeting and all lower levels down to the P standard. The criteria at these levels are therefore aimed to be achieved as a set. The overall grade is thus said to be "outcome-based", as the student needs to demonstrate that level of achievement across all criteria. For P, C and D levels, the student must demonstrate meeting all criteria at that particular level to be awarded that overall grade. For example, to be awarded an overall P grade, the student must evidence all the behaviours in all of the criteria as described in the "Pass" column.

However, to obtain an HD grade, the student needs to meet all criteria at a D level, and additionally, evidence one (for low-HD) or two or more (for high-HD) criteria at the HD level of achievement. In fact, the HD level of achievement is appropriately "aspirational", so choosing what HD criteria to address allows students to focus on aspects of their project they excel at, or skills they particularly want to develop to a high standard. In all cases, these build upon the D standard that already provides a solid demonstration of effective teamwork within the project.

3.5 The Assessment Model

The assessment model described in this section is based on the TOPAM [13]. We argue that this assessment model is appropriate for PBL as it includes frequent formative feedback, supporting supervisor's monitoring of the project without much intrusion, and delayed summative grading, ensuring the focus of students is maintained across the project, until the delivery of the final product (or products). In TOPAM students analyse the rubric and identify the level they plan to achieve, this must be conveyed to their supervisor early on. Frequent formative feedback loops help ensure students receive the support they require to achieve their targeted level of achievement.

Figure 2 illustrates the main components of the assessment model: **students** who collaborate on a **project** are supported by a **supervisor** through frequent formative feedback. The feedback is guided by the unit's **tasks** and the assessment criteria expressed within the unit's **rubric**, but also by the **product**, thus the delivery of a functional product is not jeopardised by the assessment.

During a teaching period, each student works through a series of *assessment tasks* designed to help them achieve the learning

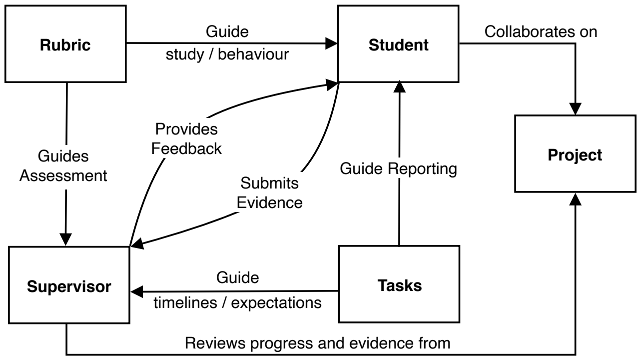


Figure 2: Visualisation of aspects within the proposed approach.

outcomes through effective use of formative feedback [13, 16]. Each task would typically have intended learning outcomes, allowing a student's submission to demonstrate a level of competency in these learning outcomes. Timely submissions are followed by a *formative feedback loop*. Project supervisors provide feedback on the work completed (as evidenced in task submissions), but do not ask for re-submissions – they rather ask for *improvements* to work artefacts, processes (individual or at the team level) etc. with the aim that any issues identified are "fixed" at the project level and evidenced in future task submissions. Only if a task submission completely fails to address the intended learning outcomes will a re-submission be requested.

Delayed grading means that the level achieved by a student will not be determined until the end of the teaching period, when each student prepares a Learning Summary Report. In the Learning Summary Report students reflect on how their behaviour and the tasks they have completed demonstrate they have met each of the unit learning outcomes to a *self-assessed* grade outcome [13].

In principle, delayed grading implies that students could submit their tasks, for the first time, at the end of the teaching period. However, it has been shown that without regular check-points, submitting only a final learning portfolio does not lead to the desired outcomes [13], as timely feedback is tight to regularly demonstrate completed work. Thus, in our model, students must incrementally report on and demonstrate their achievements towards the project goals throughout the teaching period.

3.6 Assessment Tasks

As TOPAM was originally intended for more traditional course-work units, we adapted the approach to make it more suitable for team-based project work. These changes are designed to ensure project work to remain the main focus of attention (and not the submission of assessment tasks). In line with what was reported in [40], we opted for a model where the assessment tasks themselves, in general, do not include a full body of work completed, but mainly contain (i) a brief summary of and a reflection about *project work* completed within a time period, plus (ii) additional "pointers" (or *evidence*) where the work artefacts can be found. Common to project work in many settings is the use of *repositories* where work artefacts are stored (*e.g.*, a version-controlled source code repository; defect tracking system; document repository; wiki). In this model, "pointers" are used in tasks submission that reference artefacts which can be found in any of the team's repositories.

We further introduced a number of *group* tasks (*e.g.*, an iteration plan) to indicate that individual work is always in the context of what a project team needs to achieve. The entire project team is responsible of compiling and submitting these tasks, and they can be included in individual Learning Summary Reports if a student wished to make explicit reference to them. Ultimately, the aim is that each student has a collection of evidence stemming from project work they can refer to when compiling their Learning Summary Report, and in combination with comments and feedback from the project supervisor, this will form the basis for a holistic assessment at the end of the teaching period. This process is aligned with the framework presented in Figure 2.

4 APPLICATION AND EVALUATION IN A CAPSTONE CONTEXT

The intention behind the assessment framework proposed in this study was to address the issues commonly present in Software Engineering PBL team units: evaluating students authentically and equitably in a team setting without jeopardizing the quality of the project deliverables. As a first trial, we introduced the individual assessment approach described in this paper into our Postgraduate capstone project units during the second teaching period in 2019. Across these units, a total of 27 project teams, each composed of 9 to 12 students, worked on a variety of different projects. We further divided the teaching period into four 3-week time periods (or *iterations*) with a "preparation" week before the first and some "buffer time" after the last iteration. The student teams were required to adopt an agile development methodology (*e.g.* XP [4] or SCRUM [41]) and advance project deliverables within each iteration. The iteration end-points were earmarked for formative feedback by the project supervisors

4.1 Assessment Framework

As part of the routine evaluation of our units, we surveyed the students on their perception of how the team-based projects were run. To this end, a survey was designed for students with 26 ordinal choice questions and 2 open-ended textual questions. Five of the 26 ordinal questions were concerned with assessment (reported in this paper) with the remaining ordinal questions relating to other concerns in these units (dealt with in an accompanying paper [38]). The open-ended questions called for two aspects of the assessment that worked well, and two aspects that could be improved.

The survey was conducted over the last two weeks of the teaching period *prior* to results being released and 92 students responded to the survey (response rate of 30.5%). Here, we will report on the survey results of the assessment-specific questions.

4.1.1 Rubric clarity. First of all, we wanted to know from the students whether they understood all aspects of the assessment rubric (as presented in Figure 1) and its application in their unit. Five students where in disagreement (with 1 student responding as strongly disagree), 10 students neither agreed nor disagreed, with the remaining students (84% of respondents) stating they either agree (41 respondents) or strongly agree (36 respondents) with the statement. Details of the response distribution are given in Figure 3a.

Although it is very pleasing to see that most students believe they understood the assessment rubric and how it was applied, there is a concern that their perception does not reflect reality and that actually, not everybody who responded positively to this question understood the way they were assessed. In fact, in the Learning Summary Reports that students submitted at the end of the teaching period, about 65% of all students indicated they deserved HD for their contributions, with only about 15% of students judging their contributions to be at the lower end of the spectrum. In the end, significantly less than 65% of all students received an HD grade. There may be a facet of reasons for students overestimating their achievements, one of them likely to be a lack of clear understanding of the assessment rubric and how it is applied.

4.1.2 Equitability. One of the main motivations of our work was to create an assessment model that allowed us to accurately capture each student's contribution to a team project, irrespective to the contributions of other team members. Only 10 students (11% or respondents) responded to a corresponding question negatively whereas 62 students (68% of respondents) were in agreement that the assessment approach does accurately capture individual contributions. The remaining 20 students (21% of respondents) neither agreed nor disagreed. In addition, a few students mentioned equitability as one of the aspects of the assessment that worked well in the survey's open-ended question *("The marking based on individual contributions was a good concept even if it was a team project. It helped every member to take part in task completion."; "Assessing by Individual contribution as well as the team work").* Details of the responses are given in Figure 3b.

We observed that despite our efforts to get students to understand that the assessment was individual (group tasks are assessed relative to the individuals' contribution to the task), some students were still under the impression that the group-based assessment would results in a group mark. This in large parts could have been due to other units having a similar structure where group-based tasks were given a group mark. Early indications suggest that grades were reflective of each student's achievement, but this will require further scrutiny in future teaching periods.

4.1.3 Promoting Teamwork. It was essential for us to ensure that the proposed assessment model did not encourage students to focus too much on individual work, discouraging collaborative group work. Therefore, we asked our students whether they believed the assessment model and the rubric encouraged and rewarded teamwork. A significant majority of students (or 77% of respondents) agreed that teamwork was encouraged and rewarded by the proposed assessment model (with 26% in strong agreement), only 7 students felt negatively about the approach in this regard. Furthermore, more than one in five students identified teamwork as one of the aspects of the assessment that worked well in the open-ended question *("Team work and product accountability", "1. The project helped me learn new IT skills like React Native – 2. Working with a real-time client helped me in exposure and also learned to work in a team").* Details of the response distribution are given in Figure 3c.

These responses indicate that despite the fact that students are assessed individually in a group project (something they may not have experienced before), they didn't perceive teamwork as being discouraged.

4.1.4 Product Delivery. We also wanted to find out from our students whether the assessment model and rubric had guided their approach and behaviour towards project work (*cf.* Figure 4a). An overwhelming majority of students responded positively to this question (82% agree or strongly agree) with only 4% of respondents disagreeing. However, these responses do not give us a direct indication of whether students believe the assessment model and rubric guided them in a positive or negative way. Details of the responses to this aspect are given in Figure 4a.

We do not yet have sufficient data to make an informed judgement on the impact of the assessment framework on meeting project expectations. However, investigating this impact is something we intend to do as part of future work as one of the motivations behind the design of this assessment framework was for assessment tasks to not jeopardize project work.

4.1.5 Authenticity. Finally, we wanted to find out whether students felt that the assessment approach reflected reporting and appraisal in real-world working environments. Similar to the previous question, a significant majority of the students (72% of respondents) agreed that the assessment was in line with appraisal used in real-world working environments, with only 8% in disagreement. About half of the respondents (45 students) indicated that they had prior experience in industry-based group projects, and if we restrict the responses to only this group of students, the overall distribution only marginally changes: 34 students (76% of students with industry experience) responded in agreement whereas only 4 students (*i.e.* 9%) disagreed. These responses give us confidence that we are indeed creating a learning environment with an authentic assessment approach. Details of the responses to this aspect are given in Figure 4b.

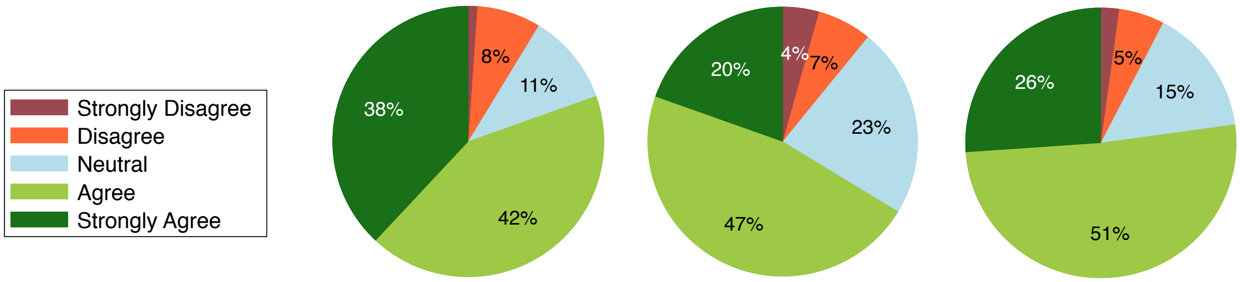
Although the responses from the students to the above-mentioned questions were very positive, these may have to be taken with a bit of caution. We stipulate that if we conducted the survey after results were released, the results for the first two assessment questions may not be quite as positive. All in all, we have very positive "first impressions" of the proposed assessment rubric and model.

4.2 Implementation

In the implementation of this assessment framework a few challenges were identified, mostly related to supervision and feedback, respectively.

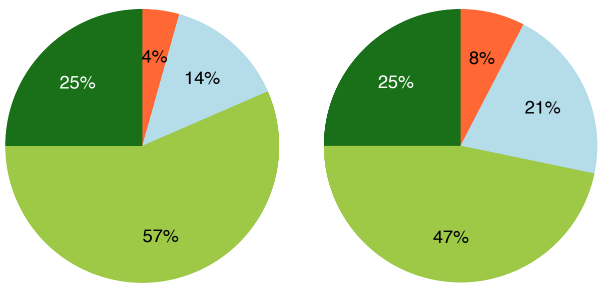
4.2.1 Consistency Amongst Supervisors. In these units, each of the 27 project teams had a unique project supervisor (*i.e.* an academic staff member). In order to ensure supervisors understood the rubric and were able to apply it accurately, we ran training sessions prior to the teaching period and we conducted a moderation session at the end of the teaching period. During moderation, supervisors came together and discussed the preliminary grades they had given, with the aim to adjust any inconsistencies as much as possible.

Due to its "outcome-based" nature, applying the assessment criteria presented in our proposed framework requires a change in mindset. It was indeed straightforward for supervisors to identify the level each student had achieved for each assessment criteria.



(a) (b) (c)

Figure 3: Responses to (a) – "I understood all aspects of the assessment rubric and how they were applied"; (b) – "Assessment in this unit accurately captured each student's contributions, independent of the contributions of other team members"; and (c) – "The Assessment Tasks and Rubric encouraged and rewarded teamwork".



(a) (b)

Figure 4: Responses to (a) – "The Assessment Tasks and Rubric helped guide my approach and behaviour to project work"; and (b) – "The assessment used in this unit reflects reporting and appraisal in real-world working environments".

However, applying the rule of grading students at the level where **all** assessment criteria are achieved is non-trivial. For example, some students may achieve a D in most criteria but fail to achieve one other criterion, in which case a fail grade should be awarded. This contradicts standard practice, where results are accumulated and not directly related to achievement of learning outcomes. Consistent with what has been reported in other literature [25], we observed that supervisors tend to grade their students based on their perception of the students' contributions rather than the evidence presented through assessment submissions. This is understandable in a PBL unit where a close working relationships can develop amongst teams and supervisors, and this is where our "outcome-based" assessment model introduces further objectivity to grading.

4.2.2 Untimely Supervisor Feedback. In an agile environment, with three week iterations, it is imperative that the supervisors focus on providing feedback promptly once work is submitted for formative feedback. This helps ensure that students can effectively take the feedback into account, and adjust their behaviour and work practices. This is different to standard assessment practice, where assignment results are not published for several weeks after the submission date. To resolve this issue, supervisors' engagement needs to be constantly monitored, and if it is found that supervisors are falling behind, they must be repeatedly encouraged to provide timely feedback.

This aspect, in some situations, could be linked to a perceived overload of assessment in our framework. The main overheads, as raised by supervisors and students, were the retrospectives at the end of each iteration, which require careful reflection and documentation. However, studies in agile methodologies have shown that the retrospectives are a critical component of successful projects [40] and hence these tasks are necessary. A potential solution to this is to have more structured retrospective submission, to help make it easier (and quicker) for supervisors to provide feedback.

4.2.3 Students not Incorporating Feedback. The TOPAM assessment model, underlying our assessment framework, prescribes the inclusion of continuous assessment submissions and feedback loops. This is designed so that students receive timely feedback and can redirect their efforts and achieve their target grade, promoting and developing their self-efficacy [24]. In Software Engineering PBL, the main feedback loops occur at the end of each iteration, in response to individual and group retrospectives. It is expected that students will reflect on this feedback, ask for clarification if required and take the feedback into account for the next iteration. However, it can happen that students do not incorporate their supervisor's feedback in their work. Rather, a large number of students will focus their efforts as they did in the previous iteration, leading to the same, potentially limited, outcomes.

This inability or unwillingness to take feedback into account, widely observed in higher education [18], is further reflected in students' Learning Summary Reports, which they submit at the end of the teaching period. Here, students can indicate a particular grade that they deserve, despite having been provided with written feedback that explicitly indicates they are not on track to achieving that target grade. Impacting on this could be a failure to understand the "outcome-based" nature of this assessment framework, although it is explained multiple times during the teaching period.

The solution to this is not straightforward, students should have ideally developed strategies for productive use of feedback throughout their degree. The teaching team could emphasise the value of self-reflection and of incorporating feedback. Supervisors could also reach out more, to make sure students understand the feedback provided. However, time-wise it is not possible for supervisors to do this and, being capstones part of students' final year, this should not be necessary. Another possibility is to provide incentives for peers to support and mentor less engaged students in understanding the benefits of continuous self-reflection and on the value of feedback.

5 LESSONS LEARNED

Given the challenges described above, there are several lessons we have taken away that will inform our future approach. None of these relate to the assessment rubrics but rather to their implementation.

5.1 Assessment Clarification

During the first weeks of the teaching period, students are presented with the rubrics and the various criteria are thoroughly described by the teaching team. Furthermore, students are asked to sign a grading agreement in which they acknowledge their understanding of the assessment rubric and the assessment model. However, once the project work gets underway, the assessment falls into the background and student lose sight of the various criteria designed to achieve teamwork and successful project outcomes.

Regarding students' awareness of the assessment, one possible reason for students loosing sight of the assessment criteria is that the supervisors may not be emphasising their importance regularly enough. In the future, supervisors will be encouraged to regularly reiterate assessment criteria and provide timely guidance, during the weekly meetings and at the end of each iteration, and to make sure that the feedback being provided aligns with the assessment criteria and that this is made clear to students.

5.2 Student Self-Efficacy

Although a few students underestimated their abilities and were redirected to revise their target grade upwards, the vast majority of students overestimated their ability to achieve the grade they initially proposed. This would not be an issue if they had taken into account their supervisor's feedback to either revise their target grade or increase their efforts and adapt their performance accordingly. However, many students did not respond to feedback and continued expecting to achieve their initial target grade even though it was clear that they were not performing at that level.

Our proposed assessment model is designed to promote and develop self-regulated learning, an active process in which students set goals for their learning and monitor their thinking, behaviours and motivation, while responding to external feedback and their environment [29]. Self-regulation is essential for students to be able to continue learning outside of formal education [9]. For this reason, we do not want to fix this problem artificially, we rather want to help students understand the importance of this process.

On our proposed assessment framework, student's self-assessment, reflections on their learning, and justification of their proposed grade, conform their Learning Summary Report. A parallel can be drawn between this document and typical job applications where applicants need to demonstrate how their experience meets the job requirements. In the future, we will focus students' attention on the Learning Summary Report from the start of the teaching period, providing examples and being explicit on the role of this document on demonstrating their perceived achievements. A possibility would be to make this a living document they populate regularly but we need to think of a way of achieving this without adding unnecessary overhead.

5.3 Supervisor Engagement

Student self-motivation is one of the key factors for successful project outcomes, supervisor engagement another. We have found that student teams with highly-engaged supervisors are more likely to result in successful outcomes for all stakeholders.

We are planning on including additional training and matching projects more accurately to supervisor skills so that supervisors feel better equipped to support their teams, which we believe will increase their engagement. Other strategies we envisage applying to improve supervisors' engagement are assigning them more time for preparing and providing feedback, and more importantly, getting them to work together in small working groups [8, 26] so they can support each other, discuss and reflect on issues as they come across.

6 CONCLUSIONS AND FUTURE WORK

Motivated by the need to expose students in tertiary education to the complexities that come from working in teams on real-world, ill-structured problems and the lack of suitably authentic assessment models, we have proposed a novel assessment framework for project-based learning. The purpose of the framework is to evaluate students authentically and equitably in a project team without affecting the quality of the project outcomes. The core of the framework are (i) a rubric including twelve assessment criteria covering key aspects related to project work, engagement and adaptability, and (ii) a Task-Oriented Portfolio Assessment model underlined by constructive alignment principles to guide student behaviour into the direction of key learning. The framework was introduced to student-teams working on different capstone projects across a Software Engineering related degree, and the initial evaluation suggests that we have met our aim of individually assessing student outcomes whilst promoting teamwork.

Can the proposed framework be applied in a different setting? In this work, we have presented the successful application thereof in the context of Software Engineering capstone units, but are yet to adapt the specific grade rubrics to different kinds of PBL settings. This is something we intend to do as part of our future investigations and we are particularly interested in identifying to what extent the rubric can be generalised. Further, by not having students solely focus on working on project deliverables, but also focusing on learning outcomes, there is a likely impact on the "quantity" (and possibly quality) of deliverables produced. At present, we do not have sufficient insights into the magnitude of this impact and thus into how any negative impact can be mitigated. This is something we intend to look into in further investigations. Finally, we need to collect longitudinal data on the effects of specific assessment criteria on students' behaviour to understand whether we need to revise them in order to improve students' skills, learning outcomes, and project outcomes.

REFERENCES

[1] Cheryl L. Aasheim, Susan Williams, and E. Sonny Butler. 2009. Knowledge and Skill Requirements for it Graduates. *Journal of Computer Information Systems* 49, 3 (2009), 48–53. https://doi.org/10.1080/08874417.2009.11645323

[2] Jocelyn Armarego. 2009. Constructive Alignment in SE education: aligning to what? In *Software Engineering: effective teaching and learning approaches and practices*, Heidi Ellis, Steven A. Demurjian, and J. Fernando Naveda (Eds.). IGI Global, Hershey, PA, 15–37. https://doi.org/10.4018/978-1-60566-102-5.ch002

[3] María C. Bastarrica, Daniel Perovich, and Maira M. Samary. 2017. What can students get from a software engineering capstone course?. In *Proceedings of 39th International Conference on Software Engineering: Software Engineering Education and Training Track (ICSE-SEET)*. ACM/IEEE, 137–145.

[4] Kent Beck. 1999. *Extreme Programming Explained: Embrace Change.* Addison-Wesley.

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

[6] Wolf-Gideon Bleek, Carola Lilienthal, and Axel Schmolitzky. 2005. Transferring Experience from Software Engineering Training in Industry to Mass University Education – The Big Picture. In *Proceedings of the 18th Conference on Software Engineering Education & Training* (Ottawa, Canada) *(CSEET '05)*. IEEE, 195–203. https://doi.org/10.1109/CSEET.2005.38

[7] Benjamin S. Bloom et al. 1956. Taxonomy of educational objectives. Vol. 1: Cognitive domain. *New York: McKay* (1956), 20–24.

[8] David Boud and Heather Middleton. 2003. Learning from others at work: communities of practice and informal learning. *Journal of workplace learning* 15, 5 (2003), 194–202.

[9] David Boud and Rebeca Soler. 2016. Sustainable assessment revisited. *Assessment & Evaluation in Higher Education* 41, 3 (2016), 400–413.

[10] Jill Bradley-Levine, Beth Berghoff, Joy Seybold, Rachel Sever, Susan Blackwell, and Azure Smiley. 2010. What teachers and administrators "need to know" about project-based learning implementation. In *Annual Meeting of the American Educational Research Association. Denver, CO.*

[11] Bernd Bruegge, Stephan Krusche, and Lukas Alperowitz. 2015. Software Engineering Project Courses with Industrial Clients. *ACM Transactions on Computing Education (TOCE)* 15, 4 (Dec. 2015), 17:1–17:31. https://doi.org/10.1145/2732155

[12] Andrew Cain and Muhammad A. Babar. 2016. Reflections on Applying Constructive Alignment with Formative Feedback for Teaching Introductory Programming and Software Architecture. In *2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C)*. 336–345.

[13] Andrew Cain, John Grundy, and Clinton J. Woodward. 2018. Focusing on learning through constructive alignment with task-oriented portfolio assessment. *European Journal of Engineering Education* 43, 4 (2018), 569–584. https://doi.org/10.1080/03043797.2017.1299693 arXiv:https://doi.org/10.1080/03043797.2017.1299693

[14] Joanna C. Dunlap. 2005. Problem-based learning and self-efficacy: How a capstone course prepares students for a profession. *Educational Technology Research and Development* 53, 1 (2005), 65–83.

[15] T. Germain, Pierre N. Robillard, and Mihaela Dulipovici. 2002. Process Activities in a Project Based Course in Software Engineering. In *Proceedings of 32nd Annual Frontiers in Education*, Vol. 3. Boston, MA. https://doi.org/10.1109/FIE.2002.1158710

[16] Graham Gibbs and Claire Simpson. 2004. Conditions under which assessment supports students learning. *Learning and Teaching in Higher Education* 1, 1 (2004), 3–31.

[17] Andreas Heberle, Rainer Neumann, Ingo Stengel, and Stefanie Regier. 2018. Teaching agile principles and software engineering concepts through real-life projects. In *2018 IEEE Global Engineering Education Conference (EDUCON)*. 1723–1728. https://doi.org/10.1109/EDUCON.2018.8363442

[18] Anders Jonsson. 2013. Facilitating productive use of feedback in higher education. *Active learning in higher education* 14, 1 (2013), 63–76.

[19] Robert F. Dugan Jr. 2011. A survey of Computer Science Capstone Course Literature. *Computer Science Education* 21, 3 (Sept. 2011), 201–267. https://doi.org/10.1080/08993408.2011.606118

[20] Siim Karus. 2019. Experience report on a move to techniques-oriented student project grading. In *Proceedings of the 41st International Conference on Software Engineering: Software Engineering Education and Training.* IEEE Press, 138–146.

[21] Robert Kegan. 1994. *In over our heads: The mental demands of modern life.* Harvard University Press.

[22] David R Krathwohl. 1984. *Taxonomy of educational objectives: Handbook 2: Affective domain.* Longman Publishing Group.

[23] Marco Kuhrmann, Joyce Nakatumba-Nabende, Rolf-Helge Pfeiffer, Paolo Tell, Jil Klünder, Tayana Conte, Stephen G. MacDonell, and Reina Hebig. 2019. Walking through the method zoo: does higher education really meet software industry demands?. In *Proceedings of the 41st International Conference on Software Engineering: Software Engineering Education and Training* (Montreal, Canada) *(ICSE-SEET '19)*. IEEE Press, 1–11. https://doi.org/10.1109/ICSE-SEET.2019.00009

[24] Check-Yee Law, John Grundy, Andrew Cain, Rajesh Vasa, and Alex Cummaudo. 2017. User Perceptions of using an open learner model visualisation tool for facilitating self-regulated learning. In *Proceedings of the Nineteenth Australasian Computing Education Conference.* ACM, 55–64.

[25] Justine Lawson, Mohammad G. Rasul, Prue Howard, Fae Martin, Roger Hadgraft, and Rob Jarman. 2015. Getting it right: The case for supervisors assessing process in capstone projects. *International Journal of Engineering Education* (2015).

[26] Margaret C. Lohman. 2006. Factors influencing teachers' engagement in informal learning activities. *Journal of workplace learning* 18, 3 (2006), 141–156.

[27] Viljan Mahnic. 2010. Teaching Scrum through team-project work: Students' perceptions and teacher's observations. *International Journal of Engineering Education* 26, 1 (2010), 96.

[28] Larry J. McKenzie, Michael S. Trevisan, Denny C. Davis, and Steven W. Beyerlein. 2004. Capstone Design Courses and Assessment: A National Study. In *Proceedings of the 2004 American Society of Engineering Education Annual Conference & Exposition.* 1545–1562.

[29] David J. Nicol and Debra Macfarlane-Dick. 2006. Formative assessment and self-regulated learning: A model and seven principles of good feedback practice. *Studies in higher education* 31, 2 (2006), 199–218.

[30] Aletta Nylén, Mats Daniels, Arnold Pears, Åsa Cajander, Roger McDermott, and Ville Isomöttönen. 2018. Why are We Here? The Educational Value Model (EVM) as a Framework to Investigate the Role of Students' Professional Identity Development. In *2018 IEEE Frontiers in Education Conference (FIE)*. IEEE, 1–8.

[31] Maria Paasivaara, Jari Vanhanen, and Casper Lassenius. 2019. Collaborating with Industrial Customers in a Capstone Project Course: The Customers' Perspective. In *Proceedings of the 41st International Conference on Software Engineering: Software Engineering Education and Training* (Montreal, Canada) *(ICSE-SEET '19)*. IEEE Press, 12–22. https://doi.org/10.1109/ICSE-SEET.2019.00010

[32] Maria Paasivaara, Dragoş Vodă, Ville T Heikkilä, Jari Vanhanen, and Casper Lassenius. 2018. How does participating in a capstone project with industrial customers affect student attitudes?. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET '18)*. ACM, 49–57. https://doi.org/10.1145/3183377.3183398

[33] Michael J. Prince and Richard M. Felder. 2006. Inductive Teaching and Learning Methods: Definitions, Comparisons, and Research Bases. *Journal of Engineering Education* 95, 2 (2006), 123–138.

[34] Paul Ramsden. 1992. *Learning to Teach in Higher Education.* Psychology Press.

[35] Daniel G. Rees Lewis, Elizabeth M. Gerber, Spencer E. Carlson, and Matthew W. Easterday. 2019. Opportunities for educational innovations in authentic project-based learning: understanding instructor perceived challenges to design for adoption. *Educational Technology Research and Development* 67, 4 (Aug. 2019), 953–982.

[36] Debbie Richards. 2009. Designing project-based courses with a focus on group formation and assessment. *ACM Transactions on Computing Education* 9, 1 (2009), 2–1–2–40. https://doi.org/10.1145/1513593.1513595

[37] Guillermo Rodriguez, Àlvaro Soria, and Marcelo Campo. 2016. Measuring the Impact of Agile Coaching on Students' Performance. *IEEE Transactions on Education* 59, 3 (Aug. 2016), 202–209. https://doi.org/10.1109/TE.2015.2506624

[38] Jean-Guy Schneider, Peter Eklund, Kevin Lee, Feifei Chen, Andrew Cain, and Mohamed Abdelrazek. 2020. Adopting Industry Agile Practices in Large-scale Capstone Education. In *Proceedings of IEEE/ACM 42nd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*. ACM, Seoul, South Korea. to appear.

[39] Jean-Guy Schneider and Lorraine Johnston. 2003. eXtreme Programming at Universities – An Educational Perspective. In *Proceedings ICSE 2003*. IEEE Computer Society Press, Portland, Oregon, 594–599.

[40] Jean-Guy Schneider and Rajesh Vasa. 2006. Agile Practices in Software Development – Experiences from Student Projects. In *Proceedings of the 17th Australian Software Engineering Conference (ASWEC '06)*, Jun Han and Mark Staples (Eds.). IEEE Computer Society Press, Sydney, Australia, 401–410.

[41] Ken Schwaber and Mike Beedle. 2002. *Agile Software Development with Scrum.* Prentice Hall.

[42] John W. Thomas. 2000. A review of research on project-based learning. (2000).

[43] Eray Tuzun, Hakan Erdogmus, and Izzet Gokhan Ozbilgin. 2018. Are Computer Science and Engineering Graduates Ready for the Software Industry? Experiences from an Industrial Student Training Program. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering Education and Training* (Gothenburg, Sweden) *(ICSE-SEET '18)*. ACM, New York, NY, USA, 68–77. https://doi.org/10.1145/3183377.3185754