UNIPED

UNIVERSITETSFORLAGET

RESEARCH ARTICLE

ÅRGANG 41, NR. 2-2018, S. 147–163
ISSN ONLINE: 1893-8981
DOI: 10.18261/ISSN.1893-8981-2018-02-06

# Project-Based Learning in IT Education: Definitions and Qualities

## Prosjekt-basert læring i IT utdannelse: definisjoner og kvaliteter

Guttorm Sindre
Department of Computer Science (IDI), Norwegian University of Science and Technology (NTNU), Trondheim, Norway
Center for Excellent IT Education (SFU-Excited), Trondheim, Norway
guttorm.sindre@ntnu.no

Michail Giannakos
Department of Computer Science (IDI), Norwegian University of Science and Technology (NTNU), Trondheim, Norway
Center for Excellent IT Education (SFU-Excited), Trondheim, Norway
michailg@ntnu.no

Birgit R. Krogstie
Department of Computer Science (IDI), Norwegian University of Science and Technology (NTNU), Trondheim, Norway
Center for Excellent IT Education (SFU-Excited), Trondheim, Norway
birgit.r.krogstie@ntnu.no

Robin Munkvold
Department of Media Technology, Nord University (NORD), Steinkjer, Norway
Center for Excellent IT Education (SU-Excited), Trondheim, Norway
robin.munkvold@nord.no

Trond Aalberg
Department of Computer Science (IDI), Norwegian University of Science and Technology (NTNU), Trondheim, Norway
Center for Excellent IT Education (SFU-Excited), Trondheim, Norway
trondaal@ntnu.no

ABSTRACT

To prepare IT graduates for professional careers, their education must provide them with "real-life" experience. There are many tasks beyond those of core software engineering and development for which students need training: project management, team building, soft-

ware estimation and planning, progress tracking, and communication. Project-based learning offers ways to transfer learning of foundational and practical knowledge into "real projects for real clients". The emphasis on projects in IT education follows naturally from the fact that projects are the main working style in the IT industry. IT study programs maintain a share of more traditional courses in which the dominant learning activities are textbook reading, lectures and weekly exercises, so the education as such is not fully project-based. Furthermore, it is difficult to turn to an entirely project-based style when there are compulsory courses from different disciplines, for instance generic mathematics courses with a huge number of students. Hence, despite the numerous benefits of project-based learning, several practical challenges need to be addressed when implementing this strategy *in a hybrid educational context. The question posed in this article is: How can project-based courses be implemented under different educational contexts to support IT students' gradual development into professional practitioners?* Projects can be extended between or across different courses in different ways, each with potential advantages and challenges. For instance, projects can be supported in the contemporary learning systems for fast feedback and continuous monitoring of the progress. Projects can also be shared among several courses taking place in the same semester, or among courses taking place in consecutive semesters. Even more radically – extrapolating the idea of a student e-portfolio – one could consider each student's entire university education as a project, the project goals being the learning goals for the study program in question, and single courses being considered as work packages contributing to these overall goals. This might motivate students and allow them to appreciate the whole picture rather than looking at courses as narrow and isolated achievements with a strong focus on exams and grades. The paper provides a classification of different ways in which project learning can be taken beyond the single course context even in study programs where it is (at least in the short term) unavoidable to have a mixture of projects and more traditional teaching.

Keywords
project-based learning, IT education, university education, qualities

## SAMMENDRAG

For å forberede nyutdannede IT-kandidater til deres fremtidige yrkeskarriere er det avgjørende å gi studentene relevant og realistisk erfaring i løpet av utdanningen. Det er mange elementer i tillegg til ren programvareutvikling som er viktige i denne sammenheng, deriblant prosjektledelse, teambygging, kommunikasjon, sporing av fremdrift og programvare-estimering og –planlegging. Prosjektbasert læring tilbyr flere metoder for tilegning av grunnleggende kunnskaper og ferdigheter nødvendig innen «reelle prosjekter for virkelige kunder». Vektleggingen av prosjekter i IT-utdanninger følger naturlig av at prosjekter er den mest sentrale arbeidsmåten i IT-bransjen, selv om vi fortsatt finner en stor andel tradisjonelle kurs hvor dominerende læringsaktiviteter er lærebok-baserte med forelesninger og ukentlige øvelser. Videre er det vanskelig å endre til en fullstendig prosjektbasert stil når det er obligatoriske kurs i ulike fag, for eksempel matematikk, med et stort antall studenter. Til tross for at det er mange fordeler med prosjektbasert læring er det altså mange utfordringer som må tas hensyn til. Spørsmålet som stilles i denne artikkelen er: *Hvordan kan*

*prosjektbaserte kurs implementeres under ulike pedagogiske kontekster for å støtte utdanningen av IT-kandidater?* Prosjekter kan utvides og utføres mellom eller på tvers av ulike kurs på ulike måter, hver med gitte fordeler og utfordringer. Eksempelvis kan prosjekter støttes gjennom moderne læringssystemer med rask tilbakemelding og kontinuerlig overvåking av progresjonen. Prosjekter kan også deles mellom flere kurs i samme semester, eller blant kurs som foregår i påfølgende semestre. Det er også mulig å se en students utdanning som et prosjekt i seg selv – definert som en e-portefølje – hvor prosjektmålene for utdanningsprogrammet og de enkelte kurs inngår som arbeidspakker tilknyttet det overordnede målet. Dette kan virke motiverende da studentene gjennom et slikt perspektiv får en mer overordnet forståelse for sin utdanning i stedet for å kun ha fokus på de enkelte kurs. I denne artikkelen presenteres en klassifisering av hvordan prosjektbasert læring kan implementeres ut over de enkelte kurs i utdanningen, selv i utdanninger som legger opp til en miks av prosjektbaserte og mer tradisjonelle læringsaktiviteter.

Nøkkelord
prosjektbasert læring, IT-utdanning, universitetsutdanning, kvalitet

## INTRODUCTION AND STATUS OF KNOWLEDGE

There is a growing demand for IT professionals. Norway will also face such an increase (IKT kompetanse, 2014), the need for IT professionals being expected to rise to over 55 000 by 2030 (from 17 000 by 2000). With a projection of the current rhythm of IT graduates, the public and private sectors will lack more than 10 000 IT professionals by 2030 (IKT kompetanse, 2014). Along the same lines, the U.S. Bureau of Labor Statistics predicts that by 2020 one of every two STEM (Science, Technology, Engineering, and Mathematics) jobs will be in IT (Bureau of Labor Statistics, 2014–2015). Currently there is estimated a shortage of 864 000 IT professionals (ICT Skills Action Plan, 2014–2018) across the EU and the European Economic Area (EEA). Despite the positive career prospects, school students seem reluctant to pursue a career in computer science, information science and technology (we use the term IT). Although the numbers for students enrolling for IT courses in high and middle schools are increasing (Roberts, 2011) there is still a huge gap to meet the demands of a growing job market in IT (http://goo.gl/gxfmI).

Prior research reports a lack of awareness of the discipline coupled with negative attitudes towards IT among school students (Grover, Pea, and Cooper, 2014; Hewner and Guzdial, 2008). In particular, for female students, studies have shown that beliefs related to isolation, lack of community support, and lack of creativity are significant factors of discouragement and decrease females' interest in classes, activities and careers related to IT (Grover, Pea, & Cooper, 2014; Tai et al., 2006). Knowledge about what a job in IT may be like, is limited and often focused on the "lonely programmer" stereotype. Whereas this lack of knowledge applies to both genders, it is believed to have a stronger effect on girls' decision about whether to become an IT student (Clayton et al. 2009). Curriculum development and subjects-approaches selection was found to shape students' perceptions of IT significantly (Taub, Ben-Ari, and Armoni, 2009). In particular, for IT studies, several negative beliefs and stereotypes exist related to the nature of the IT profession. These beliefs often

prevent young people from seeking IT education and careers, or even lead them to drop out from IT (or from majoring in IT) (Rosson et al., 2011). A recent empirical investigation at NTNU, confirms that the situation is similar in the Norwegian higher IT education (Giannakos et al., 2017; Giannakos et al., 2016; Pappas et al., 2016). Thus, a disinclination towards studying IT disciplines implies that more research is needed to investigate how students could be retained.

The complete spectrum of needed skills for future IT professionals is determined by the tasks that are required in their work environment, and as the complexity of technology increases, the skills of related professionals must also increase (Yu et al., 2016). Thus, in addition to IT hard skills, 21st century IT professionals must possess a robust understanding of data analysis, entrepreneurship and other skills related to 21st century skills (Wang et al., 2016; Binkley et al., 2012; ACM/IEEE-CS, 2015). This should be reflected in the training of future university students, particularly in IT for which training should be more than attaining technical knowledge. The needs of the IT sector today go far beyond those of core software development. Employers require graduates who can integrate the knowledge from a variety of disciplines and address complex problems with creative problem-solving and innovative thinking (Hogue et al., 2015). Hence, IT students need to develop computational thinking, critical thinking, and creativity skills through "real-life" experiences (ACM/IEEE-CS 2015). These skills will lead them to attain the needed competences, like project management, team building, software estimation and planning, progress monitoring and communication (Börstler and Hilburn, 2015).

Unlike other study programs such as engineering or medicine, IT study programs do not require rigorous post-graduation work experience before the candidates practice independently. Traditionally, IT programs have been designed to separate the academic learning from professional practice in the form of internships and co-operative placements (Clear et al, 2012). This design creates a discrepancy, not only between topics within the study program but most importantly between industry expectations and the skills of graduates (Hogue et al., 2015). Today's IT industry requires candidates who have a combination of strong theoretical knowledge and diverse technical skills and competences.

Existing higher IT education programs have been criticized for not providing enough "soft skills" including leadership, project management, critical thinking, and change management. In the most recent curriculum guidelines from the ACM/IEEE Task Force on Computing Curricula (ACM/IEEE-CS, 2013), there has been an increased emphasis on the importance of the development and mastery of problem-solving skills integrated with real-world group-based project learning activities (Cameron, 2014). The so-called project-based learning approach has been developed to address these shortcomings of traditional pedagogical approaches.

Project Based Learning (PJBL) is a pedagogical technique frequently used in IT education. Many educators have designed and successfully applied PJBL approaches. In the context of PJBL, there are many suggestions on how to influence parameters like motivation, problem generation or presentation, limitations, expectations etc. to achieve better results (Romeike, 2008). Educators and researchers have distinguished between Problem-Based Learning (PBL) and PJBL, although no single distinction is universally adopted. Indeed, it can be argued that the two share more similarities than differences. Barron et al. (1998) suggest that

PBL is the scaffold to PJBL. PJBL approaches often start with a "driving question" where the task has authenticity because it is based in real-world problems (Stefanou et al., 2013).

In engineering education, PJBL has been found more appropriate than PBL, because projects are representative of the way engineers work and fit the hierarchical nature of engineering syllabus (Mills and Treagust, 2003). Software can be developed with cheap equipment which students nowadays have anyway (e.g., PCs), and has no cost for raw materials. Hence IT studies lend themselves particularly well to engineering projects, as students can make applications for real stakeholders to perform useful tasks, which increases motivation (Blumenfeld et al., 2013). For instance, students of marine or civil engineering are less likely to make real ships or buildings, only scaled down models.

Although PJBL has a great potential, the implementation needs to be carefully considered along many parameters in order to be effective. Examples include the project type (e.g. an industry-driven or instructor-made), group features (e.g. size), group management and student motivation. Also, class organization might influence PJBL effectiveness. A class could be formed with a traditional top-down approach, where the instructor first presents the theory and then the problem in order to allow students to put the theory into practice. Alternatively, the instructor introduces the theory bottom-up within a project framework. Moving away from the single course approach, PJBL can be applied across many courses, semesters and even departments with different objectives, competences, preconditions and standards. Given all these options and possibilities, several practical challenges need to be met while attempting to implement a PJBL strategy in higher education. The aim of this article is to present different PJBL strategies employed at IDI/NTNU, IIE/NTNU and Media Technology at Nord University as well as to describe the main PJBL qualities and map them into a classification, which will allow researchers and practitioners to make informed decisions on the following research question:

> How can project-based courses be implemented under different educational contexts to support IT students' gradual development into professional practitioners?

The rest of paper is structured as follows: in the next section, NTNU's and NORD's PJBL experiences are outlined; the third section presents the PJBL classification; the fourth section discusses the proposed classification, while the last section summarizes the contribution of this paper and makes recommendations for future research.

## PUTTING PROJECT-BASED LEARNING INTO PRACTICE

The *Department of Computer Science* (IDI) at the Norwegian University of Science and Technology (NTNU) has a long tradition of including team projects as an essential component in the education of bachelor and master candidates in computing. Some of these projects are quite big, for example an entire course is organized as a group project. This is the case for the 7.5 ECTS credits Experts in Team (EiT) project taken by all our students in the 8th semester (Jaccheri and Sindre, 2007); another example is the 15 ECTS credits "customer-driven project" in the 7th semester for the computer science students (Andersen et al., 1994), and in the 6th semester for the bachelor informatics students (Krogstie and Divi-

tini, 2009). The latter students have also had a full course team project in their 3rd semester. In addition, to the "pure" PJBL courses, there have been experiences with many smaller projects in the context of more traditional lecture courses. Those projects either spanned several courses, taking place in the 4th semester of both the aforementioned study programs (Sindre et al., 2003c), or were embedded inside single courses, exploring more focused tasks like programming (Sindre, Line and Valvåg, 2003b), requirements elicitation (Sindre, 2005), and document review (Sindre et al., 2003a). PJBL classes have also been offered in a more flexible manner, using the form of an intensive course. IDI has provided numerous two-or-three-day PJBL tutorials on game/animation development [e.g. (Giannakos, Jaccheri and Proto 2013)].

Within the *institute for Media Technology* at Nord University, students experience PJBL already from the first semester. Nord's bachelor program Games and Experience Technology gives the students a real-world experience through a concept called Game Lab through the studies. Game lab is a video game company simulator where the students are given different roles as in a real company, and where they are required to show up for work and produce according to weekly goals. The progress is evaluated each week by an executive committee, consisting of teachers from the school and people working in the industry. If the reported status is found unacceptable, the company may be dissolved, and the students will either have to come up with a different project or they will have to "apply" for a job at one of the other Game Lab companies at the school.

The Game Lab operates every semester during the bachelor program. It functions as an "umbrella subject" in which it is expected that the students will be using their knowledge and skills from other subjects taught in other classes within the bachelor's program to solve different issues within Game Lab. There is a strong focus on defining real projects in collaboration with enterprises in the region, and assignments are implemented as work challenges within Media projects and Game Labs (film.nord.no and games.nord.no).

The *former Department of Informatics and e-Learning* (IIE) (currently merged with the Department of Computer Science) at NTNU similarly includes PJBL as a cornerstone approach in their Bachelor and Master in IT programs. The computer science engineer bachelor program includes intensive agile programming projects. In courses on the bachelor's as well as the master's level Concurrent Design (CCD) is used as a process framework. CCD is based on collaborative work among teams with different competences and skills and includes intensive, facilitated sessions of synchronous, collocated work. The focus on multidisciplinary collaboration helps the students integrate previously gained knowledge from various areas. CCD is well adapted to the composition of study programs at IIE, in which many students have backgrounds and/or attend programs that are less technical and more focused on IT implementation and the use of IT to support collaboration. For instance, the Master of ICT-based learning at IIE enrolls students who have bachelor's degrees within engineering (e.g. machine, electronics, logistics), IT maintenance and IT-supported organization development. At the master level, PJBL by use of CCD helps student teams utilize and integrate knowledge from the various bachelor studies, in the context of the graduate-level syllabus, e.g. on project management and computer-supported cooperative work. For several years the school has also successfully made use of special lab facilities designed to support innovative teaching approaches: in the room; students are

organized in groups provided with infrastructure for easily sharing information contents with the entire class through large screens.

Although the aforementioned PJBL approaches implemented in the three institutions (IDI/NTNU, IIE/NTNU and NORD) and have substantial differences; the expected knowledge, skills and competences from the IT students are similar. In particular, it is expected from the students that they master the following technical skills:

- Define, model and break down complex engineering problems, including choosing relevant models and methods, and carrying out calculations and solutions independently and critically,
- Develop comprehensive solutions to engineering problems, including the ability to develop solutions in an interdisciplinary context, and carry out an independent engineering research and development project under academic supervision,
- Be able to renew and adapt professionally, including the development of professional competence on his/her own initiative.

In addition to the aforementioned technical skills, the students are expected to master the following general competences:

- Understand the role of the engineer in a comprehensive societal perspective, have insight into ethical requirements and consideration of sustainable development, be able to analyze ethical problems connected to engineering work, and contribute to innovation and entrepreneurship.
- Ability to disseminate, communicate and cooperate inter-disciplinarily on engineering problems and solutions to specialists and the general public.
- Understand possibilities and limitations when using information and communication technology, including juridical and societal aspects.
- Ability to lead and motivate co-workers, including having an international perspective on his/her profession, and develop ability to international orientation and collaboration.

It is compulsory for all students to define a problem description and establish a project to solve a respective problem, and there should be a strong real-world element to ensure that the experience is realistic. Although emphasis is placed on knowledge and skills related to software and development, the PJBL approaches enable a student to demonstrate the following qualities, as defined from the ACM/IEEE task force (ACM/IEEE-CS 2015):

- Show mastery of professional knowledge
- Demonstrate an understanding as well as being able to apply technical knowledge
- Work both individually and as part of a team
- Demonstrate an understanding of negotiation, effective work habits, leadership, and good communication
- Design appropriate solutions in one or more application domains
- Find acceptable compromises within the limitations of cost, time, knowledge, existing systems, and organizations.
- Appreciate the necessity of continuing professional development.

## CLASSIFICATION OF PROJECT-BASED LEARNING

As mentioned, although PJBL has been an effective teaching strategy; there are many substantial differences/needs and many facets across disciplines, educational levels and contexts. There are different ways to implement PJBL and many design decisions to be considered from the instructor in order to set up an appropriate plan for the expected learning goals and objectives PJBL approach. It is often difficult for instructors to make informed decisions about the design, since the alternatives and the consequences of different choices are not always clear. The instructors should develop an outline that explains the PJBL essential elements, expectations, assessment and so forth. Although the outline can take various forms, it should clarify a set of minimum elements. Within the Norwegian higher IT education, we have also encountered different PJBL approaches, which many times lead our students to master different set of skills and competences.

Drawing from our experience at NTNU (Jaccheri and Sindre, 2007) in the Experts in Teams (EiT) course; students have to work in teams where each team has to deliver a product and process report, counting 100% towards the grade. The product report must present and discuss the interdisciplinary problem solution and the scientific methods that have been used to come to the solution. The process report must describe the team process, e.g., how the team cooperated, roles of different team members, whether there were any significant events during the process (e.g., conflicts, how these were solved), and how these can be related to group process theory. The instructor describes a fairly open-ended theme for that village. Each student team may then invent their own project assignment, and set their own milestones, as long as they stay within the given thematic area and end up delivering the required reports. This openness of the assignment is supposed to foster student creativity and a strong sense of ownership of the conceived project. Many EiT courses within NTNU do imply some background knowledge as a precondition while others may be harder to accommodate. Students are not assigned to different EiT courses randomly; students themselves carry the responsibility for convening, meeting agenda, and leading the meetings. At the meeting, shared problems are discussed to agree on actions to be taken. Some meetings are reserved for technical or team process presentations by the students in accordance with a milestone plan that the students have worked out themselves at the beginning of the semester. By working in EiT where each team member has different competences as well as different perspectives, the students will develop attitudes and interdisciplinary teamwork skills. Through this process the students will be exposed to the challenge in interdisciplinary communication, learn to operate within an interdisciplinary environment, learn to understand the interaction between each member of the team, and learn how this interaction affects them.

Another example is the 15 ECTS credits PJBL course on NTNU's 3-year study program on informatics (Krogstie and Divitini, 2009). This is an undergraduate project course taking 50% of students' workload in the last (6th) semester of the bachelor program. Students work in teams to develop software for "real clients". Deliverables include a software product, a project report in several versions and an oral presentation. Teams consist of 3–5 students; with an overarching learning goal to "get experience with software development in a team". The students get one grade for the whole group and product, process and presentation count towards the final grade.

Instructors of PJBL in IT education have encountered many problems towards the design of their course (Börstler and Hilburn, 2015). In this work we attempt to gather the different alternatives into one space so that they can be used as a pedagogical aid for teaching PJBL courses in IT education. We do this by focusing on the aspects of PJBL that through our own experience with PJBL over many years and in several learning institutions have emerged as most critical to course success.

Drawing from our experiences with the aforementioned courses, forming the teams of the project is one of the most important elements. Issues related to the interdisciplinarity, experience, skills, personalities, and availability while creating teams are crucial. Another essential aspect is the outcome and deliverable of the PJBL, which can be a presentation, project report, or even an artefact (or any combination). Guidelines for the process may vary from one PJBL course to another. For instance, instructors can help students to create a common language by sharing course material and lectures, setting up face-to-face meetings, joint presentations, and electronic media and even scheduling regular meetings to ensure the progress and communication. Another important aspect is related to the theoretical pedagogical framework used, which can be, for instance, peer tutoring, flipped classroom or any other active learning framework. Other issues we found important during designing and putting into practise our PJBL in IT subjects include the grading (what counts and whose grading is taking place) as well as the limits of the PJBL course (the project being bounded by a single course or institute). The latter issue is very important given the need in IT education to introduce our students "to strong real-world element . . . to ensure that the experience is realistic" [ACM/IEEE-CS 2015].

In an attempt to model our experience, and drawing from the aforementioned PJBL courses at NTNU and NORD; in this article, we provide a classification of different ways in which PJBL can be applied (Figure 1). This will allow instructors to make informed decisions about the alternative design decisions. We consider the following characteristics of project-based learning:

1. *Teaching context:* A project can be the entire course, or be a smaller part of a course which also includes more traditional teaching like lectures covering a textbook.
2. *Range of implementation:* A project can range from one to many courses, and then either courses taught in the same term, or courses taught in subsequent terms.
3. *Learning context:* A project takes place after students have learned relevant theory about necessary technology and methods (theory first approach), or the instructor introduces the theory within a project framework (theory-project approach), or the instructor introduces the theory entirely after the project (project first approach).
4. *Institutional context:* A project can be implemented locally to one organization (typically the university), or involve other organizations (e.g. cross institutional); furthermore projects might also involve other stakeholders (e.g. enterprises, municipality).
5. *Personnel composition:* A project can focus on individual team projects, or even larger constellations with large teams composed of smaller teams. Also, teams could be homogeneous (consisting of same course/field students) or heterogeneous.
6. *Assessment:* Projects typically use Pass/Fail or a more granular grading scheme. In a team project, the policy could either be to give the same grade to all team members,

though possibly with exceptions for extreme cases of non-contribution; or grades could be individual despite the deliverable being a team effort. Individual grades require explicit information on who has done what, based on close teacher tracking or student self-reporting about individual contributions to the team effort, thus being more demanding than giving a joint grade.

7. *Project Variety:* Same project for every student/team, or unique, personalized, or even self-selected by the students in each team.
8. *Degrees of Freedom of the Process:* Project process can range from a well-specified (e.g. what methods to be used, steps to be undertaken and deadlines) to a more flexible one, where the only thing that matters is the final deliverable.
9. *Degrees of Freedom in the Deliverables:* Projects' deliverables can range from very strict and well-defined, where instructors have specified in much detail what problem is to be solved and what should be delivered; to more flexible ones, where each team is completely free to decide what to develop as long as it relates to the learning goals.

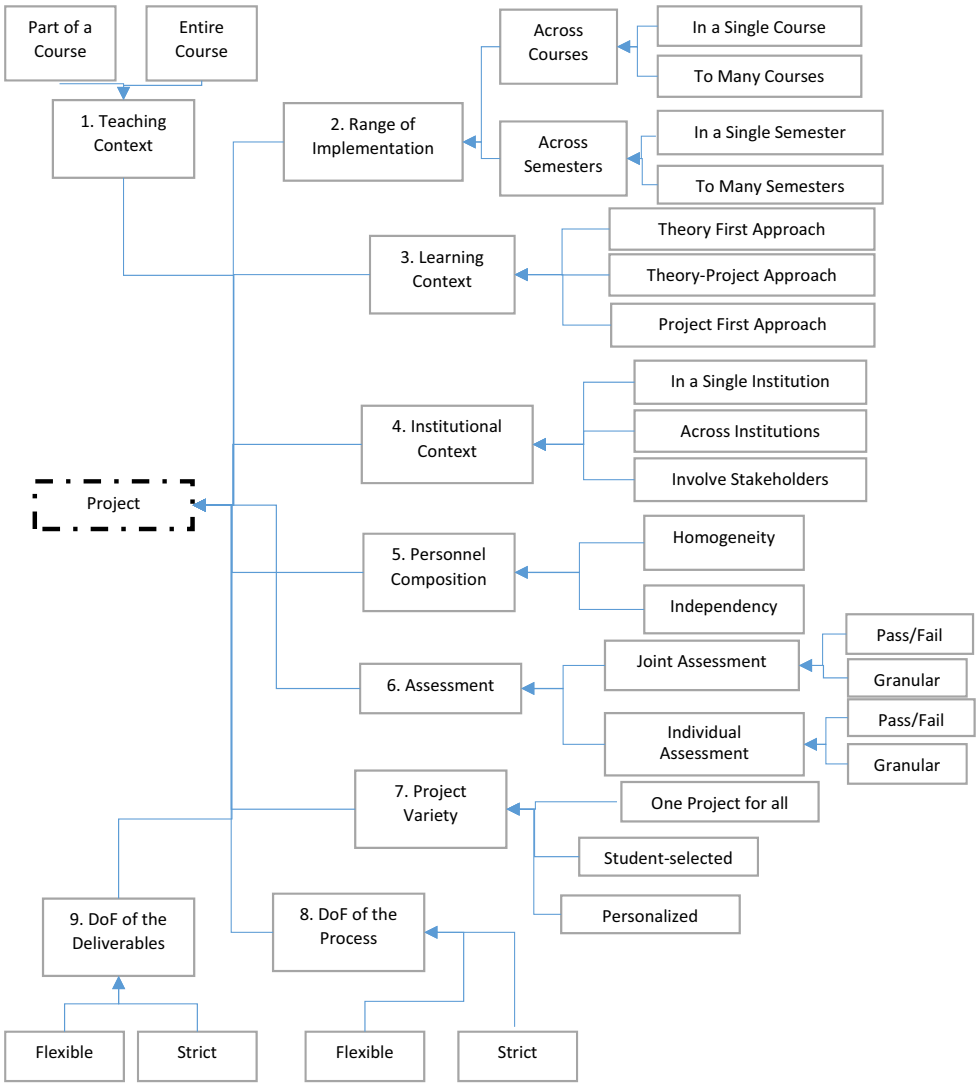


**Fig. 1.** Classification of different ways in which project learning can be applied

Following the characteristics as outlined in Figure 1 (e.g. "C1" for Characteristic 1) we provide more detailed explanation as well as illustrative examples. We start by considering the degree of freedom for the deliverable (C9) and process (C8). If a project assignment needs to ensure coverage of certain topical learning goals, this works in favour of setting some strict guidelines for what must be delivered (i.e., low degree of freedom for the deliverable [C9]), and how the work should progress and proceed (i.e., low degree of freedom for process (C8)). With a high degree of freedom there is a risk that some project teams might end up doing work that does not quite address the learning goals, or that they divide work among themselves so that some team members are exposed to some topics and others to other topics, whereas the idea was that all should at attain certain competencies. Little freedom will also make it easier for teaching staff to supervise and assist the project teams, as they might have a clear idea about typical obstacles the project teams will meet as well as the stages of the project work.

Another aspect that impacts on how easy it is for staff to supervise and help, is the project variety (C7). Low variety of projects (e.g., same project assignment to all teams) makes help and supervision easier, as partial solutions can be supplied to students if some project teams have trouble with progress. On the other hand, projects with a high degree of freedom and different / self-selected projects for each team may be better if the key learning goals pursued are about creativity and innovation and discovery of customer / market needs, rather than covering specific technical competencies. Project variety furthermore reduces the threat of plagiarism between project teams. Hence, considering the degrees of freedom in the deliverables as well as the process, it is important to consider what are the learning goals and the expected competencies.

Two other critical ingredients of a project assignment are range of implementation (C2) as well as the composition of the teams (C5). A large project, which takes an entire course, or possibly spans courses and institutions, involving a heterogeneous group of students, will have advantages in increased realism – as in later work-life students must expect to face complex projects and collaborate with experts from different fields, not just with people with a similar background as themselves. The more heterogeneous the project team, the bigger the collaboration challenge, so with ambitious learning goals regarding collaboration, this kind of project has an advantage. However, increased size and complexity of a project also increases project risk, e.g., that the students might not be able to successfully deliver what is expected of them. Sometimes, a failed project might be an interesting learning experience, but in other cases it might just demotivate students. Also, if the students are supposed to be taking other courses in parallel with the project course (which is often the case) their learning progress in these other courses may suffer if they spend much more than the expected time on the project. Hence, complex projects, even if motivating and successful, might challenge collaboration between students and be a source of conflict with other teachers.

Having students form the teams, as opposed to having the staff decide the team composition, is another recurring issue with effects on project processes and results. One model applied in one of our courses is to allow students to create their own project team as well as choose their own project task (prioritizing from a list of offered, different tasks – giving a relatively high degree of freedom regarding the deliverable [C9]). This leaves the teaching

staff to do some match-making to ensure everyone gets a team and a task of reasonable preference, as well as combining students and projects such that at least one or two students in the team have reasonable technical competence/experience to match the specific requirements of the project. One result of this arrangement is that the project course gets several teams of motivated students, often at the same level of (high) ambition, who already know each other. This reduces the time needed for them to become effective teams (in practice skipping early phases of group formation) and often ensures good project results (typically in the form of working software) (see Krogstie [2008] for an example). On the other hand, this way of organizing tends to harm the diversity within the teams while also creating some "high risk" teams including one or more "loners". All in all, it does not necessarily lead to more learning, considering the class in total.

Assessment (C6) can happen in an individual or a joint manner. Joint grade (i.e., same grade to all team members) is an advantage for the staff workload, as it is much more cumbersome for staff to collect and evaluate the extra information needed to grade individually. Especially in projects where background and support for other tasks might have been important for the group's progress without having clear contributions in the report, individual grades may easily end up being misleading. Outsourcing the individual component of the grade to student peer review is also risky, as this might be vulnerable to intrigues and collusion. If students need to spend a lot of time documenting who wrote what in the report, who made which suggestions in meetings, etc., this will "steal" working hours from making better deliverables, so they might achieve less in the project. A joint grade may also be better for collaboration within the project, i.e., team-members helping each other, since there is always an incentive for everybody to provide such help, while with purely individual grades, the team members might be wary of contributing to other parts of the report than "their own". Hence, the joint grade might also be more natural for realism – in industry, the customer's satisfaction will depend on the quality of the overall deliverable, not on who did what in the vendor company. The main disadvantage with joint grades is that some students might opt to work little in the project, thus causing extra load on the others, or that some students are too weak to make any substantial contribution.

As for the Learning Context (C3), the "traditional" theory-first approach has the advantage that students may progress more quickly with the project if they already know the theory. However, a challenge could be that a considerable fraction of the students have not learnt the theory very well, even if they have passed the theory course which is a prerequisite for the project course. For instance, a student who got D or E in preceding courses in Programming and Databases might not really be able to contribute much in a project to make a useful application running on top of a database. A challenge with the theory-first approach could be that first the theory course feels demotivating to some students because it lacks a project to demonstrate what the knowledge can be used for, with the result that they learn the topic poorly, and thereafter their learning outcome from the subsequent project becomes limited, too, because their skills level was not really adequate to undertake the project. If such problems are common, the theory-project approach might be better, as it provides more scaffolding for students who are weak at the outset of the project, and also gives a more motivating context for learning the theory. The project-first approach is believed to be quite rare, as students would have an even bigger lack of necessary knowl-

edge or skill than if having D or E from a prerequisite course. However, the project-first approach could be envisioned in cases where the project requirements are fairly easy to meet, though not necessarily in the optimal way. Hence, students could succeed in the project, yet later be shown that they could have solved the project much more elegantly if they had known the theory from the next course. This could be useful if the next course contains theory which is hard to explain and motivate without big example cases, where a preceding project would then ensure not only the availability of such examples, but also that the students were thoroughly familiar with the examples and thus more easily able to grasp the motivation for the new theory.

## DISCUSSION

Modern IT projects have moved away from traditional models where a single collocated team was building, from scratch, a piece of software for a known client according to well-defined requirements and process (Mead, 2009). IT projects are ubiquitous and typically just a part of a complex system that imposes complex inter-operability requirements based on changing platforms and technologies (Börstler and Hilburn, 2015). To prepare IT graduates for professional careers, their education must provide them with the expected real-life professional development. There is a large body of knowledge on issues and challenges pertaining to PJBL in IT education (Crnkovic et al. 2012; Wikstrand and Borstler 2006; Borstler and Hilburn, 2015). Furthermore, there are also accreditation regulations and international curriculum initiatives, such as ABET [2015] and ACM/IEEE-CS [2015], that drive educational institutions to integrate the teaching of nontechnical skills into their IT curricula.

PJBL has been found to have great potential for introducing realistic experiences to IT students (ACM/IEEE-CS 2015). However, the implementation needs to be carefully considered along many parameters in order to be effective. Given the diversity of IT education contexts (e.g., years of studies, engineers or not) and the lack of instructors' awareness of the alternatives; the development of an appropriate and effective PJBL course is becoming more difficult.

The aim of this article is to provide a classification of different design decisions in which project learning can be implemented, and allow instructors to make informed decisions during the preparation of their PJBL approach. Drawing from our experience and the different scenarios and implementations found in the literature (Railsback, 2002) we provide some insights as well as ideas which might help teachers, educational policymakers, and curriculum designers.

Assessment seems to be one of the most important aspects for students' work and listing the formal requirements for achieving certain grades influences the way students work and prioritize tasks (Vasilevskaya et al., 2015). As a consequence, the grading schema definitely guides students' PJBL. The expected deliverable (which in most of the cases is connected to the grade) is also considered one of the cornerstones of PJBL courses. This is due to the fact that the project deliverable is associated with the grade, the expected workload of the course, the formation of the team and in many times the level of realism (e.g., real product for real clients).

Another important characteristic from our category is the institutional context. For instance, when the instructor involves stakeholders from the IT industry, the students get numerous advantages: a wider range of learning resources than those found in a classroom environment, a built-in support system, motivation, and more (Meiszner et al., 2008).

Assessment as well as different pedagogical approaches (related also to the DoF of the process) have been found to play an important role in the success of PJBL in IT education. There are various models to assess PJBL in IT education; the diversity of these models varies. For instance, PJBL can be assessed from the IT industry, the problem owner or the instructor. Looking at the recent special issue in team PJBL in IT education, published from the ACM Transactions in Computing Education (Börstler and Hilburn, 2015), assessment represents one of the most complicated elements in the area, and the design, implementation, and analysis of assessment models, especially for large teams; is one of the most challenging but also important areas for future work.

Complicating the challenge of how to organize PJBL, is that that different aspects (such as the ones outlined in this paper) are not independent, A choice along one dimension typically has a positive or detrimental effect on another aspect. For instance, group composition is relevant to learning (Wilson, Goodman and Cronin 2007). Diversity in teams holds both learning opportunities and challenges. Teamwork quality – which has been found to be very important in PJBL in IT education (Börstler and Hilburn, 2015) – has a mediating effect on the impact of group diversity (Curseu and Pluut, 2013). Thus, in practice, to gain the most out of the learning potential of heterogeneous teams, it is required that high quality teamwork be given necessary attention and resources.

The insights presented in this section might help educators to reflect upon our experience, make informed decisions and illustrate the relevance of the aspects of project-based learning presented in Figure 1. However, we see a need for a more rigorous framework for PJBL to help educators design project-based learning effectively. To underpin such a framework, there is a need for empirical data. We recommend that large-scale investigations be made based on the systematic implementation and comparison of different design decisions. This might stimulate future research addressing the following additional research questions:

- Which theory could represent a suitable normative starting point for large-scale investigations in PJBL?
- How could a suitable framework be constructed? For this, we would need a structure model for tests based on multifactorial item response models.
- How could a test of these design decisions be constructed? Which items and contexts are suitable? How can the items be validated?
- How should the setting of the tests be designed? Which influence factors could distort the measurements?

## CONCLUSIONS

In summary, the many variables in PJBL can produce wide variations in quality and in the educational objectives that can be achieved in IT education. A classification is proposed to

facilitate an awareness of these differences and help teachers choose a PJBL method most appropriate for their students and courses. The classification and the discussion developed in this article help shed light on different ways in which project learning can be taken beyond single educational context, and better support IT professionals' development.

Despite the apparent different forms of PJBL, there are several topics that are addressed by any PJBL strategy. In particular, PJBL improves creativity and teamwork competences, which in turn allow students to create more original projects and obtain better academic results. Well defined organizational issues, knowledge preconditions and learning goals allow students to deliver complete projects of high level. Carefully selected supportive materials, technologies and methods provide flexibility and independency to the students, which in turn allow them to be more creative. Another benefit of PJBL in IT education is the deeper understanding of the role that computer tools can play in our lives, especially in supporting creativity.

Despite the fact that there have been several and important developments in PJBL in IT education during the last years; there is still much that can be done in order to better understand the benefits and challenges of different PJBL approaches in IT education. The contribution of this article might foster this by providing a framework for exchanging experiences among universities and assisting national stakeholders and educators on their future decisions and plans. In this endeavor, the identified types of PJBL could be used as a common ground and allow researchers and educators to better understand, compare and improve the value of their PJBL initiatives. Future research is expected to empirically analyse the effectiveness of the different PJBL types as well as combinations of them on students' outcome, attitudes, learning style and creativity.

## REFERENCES

ABET (2015). *Accreditation Criteria and Supporting Docs.* Retrieved February 22, 2016, from http://www.abet.org/ accreditation/accreditation-criteria/.

ACM/IEEE-CS (2013). *Joint Task Force on Computing Curricula*. 2013. ACM/IEEE Computing Curricula 2003 Final Report. http://www.acm.org/education/CS2013-final-report.pdf.

ACM/IEEE-CS (2015). *Joint Task Force on Computing Curricula: Association for Computing Machinery and IEEE Computer Society. Software Engineering 2014: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering.*

Andersen, R., Conradi, R., Krogstie, J., Sindre, G., & Sølvberg, A. (1994). Project courses at the NTH: 20 years of experience. *Proc. of the Software Engineering Education*, Springer Vol. 750, pp. 177–188.

Barron, B. J., Schwartz, D. L., Vye, N. J., Moore, A., Petrosino, A., Zech, L., & Bransford, J. D. (1998). Doing with understanding: Lessons from research on problem-and project-based learning. *Journal of the Learning Sciences, 7*(3/4), pp. 271–311.

Binkley, M., Erstad, O., Herman, J., Raizen, S., Ripley, M., Miller-Ricci, M., & Rumble, M. (2012). Defining twenty-first century skills. In P. Griffin, B. McGaw, & E. Care (Eds.), *Assessment and Teaching of 21st century Skills* (pp. 17–66). Springer Netherlands.

Blumenfeld, P. C., Soloway, E., Marx, R. W., Krajcik, J. S., Guzdial, M., & Palincsar, A. (1991). Motivating project-based learning: Sustaining the doing, supporting the learning. *Educational Psychologist, 26*(3–4), pp. 369–398.

Bureau of Labor Statistics (2014–2015). *Employment Projections 2010–2020*, available at http://www.bls.gov/emp/

Börstler, J., & Hilburn, T. B. (2015). Team Projects in Computing Education. *ACM Transactions on Computing Education* (TOCE), 15(4), 16.

Cameron, B. H. (2014). Enterprise integration: An experiential learning model. *Innovative Practices in Teaching Information Sciences and Technology* (pp. 143–155): Springer.

Clayton, Kaylene L., von Hellens, Liisa A., & Nielsen, Sue H. (2009). *Gender Stereotypes Prevail in ICT: A Research Review.* ACM.

Clear, T., Claxton, G., Thompson, S., & Fincher, S. (2012). Cooperative and work-integrated education in information technology. In R. K. Coll & K. E. Zegward (Eds.), *International Handbook for Cooperative and Work Integrated Education: International Perspectives of Theory, Research and Practice* (2nd Edition) (pp. 141–151). Hamilton, New Zealand: University of Waikato.

Crnkovic, I., Bosnic, I., & Zagat. M. (2012). Ten tips to succeed in global software engineering education. In *Proceedings of the 34th International Conference on Software Engineering*, 1225–1234.

Curseu, P. L., & Pluut, H. (n.d.). "Student Groups as Learning Entities: The Effect of Group Diversity and Teamwork Quality on Groups' Cognitive Complexity." *Studies in Higher Education, 38*(1), 87–103.

Giannakos, M. N., Jaccheri, L., & Proto, R. (2013), Teaching computer science to young children through creativity: Lessons learned from the case of Norway. *Proc. of the 3rd Computer Science Education Research Conference* (CSERC '13), ACM Press, pp. 103–111.

Giannakos, M. N., Pappas, I. O., Jaccheri, L., & Sampson, D. G. (2016). Understanding student retention in computer science education: The role of environment, gains, barriers and usefulness. *Education and Information Technologies*, 1–18. DOI: http://dx.doi.org/10.1007/s10639-016-9538-1.

Giannakos, M.N., Aalberg, T., Divitini, M., Jaccheri, L., Mikalef, P., Pappas, I., & Sindre, G. (2017). Identifying dropout factors in information technology education: A case study. In *Global Engineering Education Conference* (EDUCON), 2017, (pp. 1187-1194), IEEE.

Grover, S., Pea, R., & Cooper, S. (2014). Remedying misperceptions of computer science among middle school students. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (pp. 343–348). ACM.

Hewner, M., & Guzdial, M. (2008). Attitudes about computing in postsecondary graduates. In *Proc. of the International Computing Education Research Conference* (pp. 71–78). ACM.

Hogue, A., Kapralos, B., & Desjardins, F. (2011). The role of project-based learning in IT: A case study in a game development and entrepreneurship program. *Interactive Technology and Smart Education, 8*(2), 120–134.

Hogue, A., Percival, J., El-Khatib, K., & Hayes, G. (2015). *Reflection on integrative project based learning in business and information technology programs*. Higher Education in Transformation Conference, Dublin, Ireland, 2015, pp.132–144.

ICT Skills Action Plan (2014–2018). "*Government, education and industry working together to make Ireland a global leader in ICT talent*", retrieved 19 Feb. by: http://cork.etb.ie/wp-content/uploads/sites/20/2014/08/14042014-ICT_Skills_Action_Plan-Publication.pdf

IKT-kompetanse. (2014). *Dimensjonering av avansert IKT-kompetanse*: https://www.regjeringen.no/nb/dokumenter/Dimensjonering-av-avansert-IKT-kompetanse/id762445/

Jaccheri, L., & Sindre, G. (2007), Software engineering students meet interdisciplinary project work and art. *Proc. of the 11th International Conference of Information Visualization* (IV'07). IEEE, pp. 925–934.

Krogstie, B. (2008). *Power through Brokering. OSS Participation in SE Projects*. IEEE Computer Society.

Krogstie, B. R., & Divitini, M. (2009). Shared timeline and individual experience: Supporting retrospective reflection in student software engineering teams. *Proc. of the Software Engineering Education and Training* (CSEET'09). IEEE, pp. 85–92.

Mead, N. R. (2009). Software engineering education: How far we've come and how far we have to go. *Journal of Systems and Software, 82*(4), 571–575.

Meiszner, A., Glott, R., & Sowe. S. (2008). Free/Libre open source software (FLOSS) communities as an example of successful open participatory learning ecosystems. *European Journal for the Informatics Profession*, UPGRADE. IX. 3, 62–68

Mills, J. E., & Treagust, D. F. (2003). Engineering education—Is problem-based or project-based learning the answer? *Australasian Journal of Engineering Education, 3*(2). url: http://www.aaee.com.au/journal/2003/mills_treagust03.pdf

Pappas, I., Giannakos, M., & Jaccheri, L. (2016). Investigating factors influencing students' intention to dropout computer science studies. In *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education* (ITiCSE '16). ACM, New York, NY, USA, 198–203.

Pirker, J., Riffnaller-Schiefer, M., & Gütl, C. (2014). Motivational active learning: Engaging university students in computer science education. In *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education* (pp. 297–302). ACM.

Railsback, J. (2002). *Project-based instruction: Creating excitement for learning*. Portland, OR, Northwest Regional Educational Laboratory.

Roberts, E. S. (2011). Meeting the challenges of rising enrollments. *ACM Inroads, 2*(3), 4–6.

Romeike, R. (2008). What's my challenge? The forgotten part of problem solving in computer science education. *Proc. of the Informatics Education-Supporting Computational Thinking* Springer Vol. 5090, pp. 122–133.

Rosson, M. B., Carroll, J. M., & Sinha, H. (2011). Orientation of undergraduates toward careers in the Computer and Information Sciences: Gender, self-efficacy and social support. *ACM Transactions on Computing Education (TOCE), 11*(3), 14.

Sindre, G. (2005), Teaching oral communication techniques in RE by student-student role play: Initial experiences. *Proc. of the Software Engineering Education & Training* (CSEET '05), IEEE, pp. 85–92.

Sindre, G., Line, S., & Valvåg, O. V. (2003b), Positive experiences with an open project assignment in an introductory programming course. *Proc. of the 25th International Conference on Software Engineering* (ICSE '03), IEEE, pp. 608–613.

Sindre, G., Moody, D. L., Brasethvik, T., & Solvberg, A. (2003c). Introducing peer review in an IS analysis course. *Journal of Information Systems Education, 14*(1), pp. 101–120.

Sindre, G., Stålhane, T., Brataas, G., & Conradi, R. (2003a). The cross-course software engineering project at the NTNU: four years of experience. In *Software Engineering Education and Training* (CSEE&T '03). IEEE, pp. 251–258.

Stefanou, C., Stolk, J. D., Prince, M., Chen, J. C., & Lord, S. M. (2013), Self-regulation and autonomy in problem-and project-based learning environments. *Active Learning in Higher Education, 14*(2), pp. 109–122.

Tai, R., Qi Liu, C., Maltese, A.V., & Fan, X. (2006). Planning early for careers in science. *Science. 312*(5777), 1143–1144.

Taub, R., Ben-Ari, M., & Armoni, M. (2009). The effect of CS unplugged on middle-school students' views of CS. *ACM SIGCSE Bulletin, 41*(3), 99–103.

Vasilevskaya, M., Broman, D., & Sandahl, K. (2015). Assessing large-project courses: Model, activities, and lessons learned. *ACM Transactions on Computing Education* (TOCE), 15(4), 20.

Wang, Y. Y., Lin, T. C., & Tsay, C. (2016). Encouraging IS developers to learn business skills: an examination of the MARS model. *IT & People, 29*(2), 381–418.

Wikstrand, G., & Börstler, J. (2006). Success factors for team project courses. In *Proceedings of the 19th Conference on Software Engineering Education and Training,* 95–102.

Wilson, Jeanne M., Goodman, Paul S., & Cronin, M. A. (2002). "Group learning." *Academy of Management Review, 32*(4), 1041–59.

Yu, P. L., Fang, S. C., & Wang, Y. L. (2016). Improving IT professionals' job skills development: The use of management styles and individual cultural value orientation. *Asia Pacific Management Review, 21*(2), 63–73.