Contents lists available at Science-Gate

International Journal of Advanced and Applied Sciences

Journal homepage: http://www.science-gate.com/IJAAS.html

Implementing problem-based learning in the software engineering course

Azida Zainol *, Wafa Sulaiman Almukadi

College of Computer Science and Engineering, University of Jeddah, Jeddah, Saudi Arabia

ARTICLE INFO

Article history:
Received 14 April 2020
Received in revised form
13 July 2020
Accepted 13 July 2020

Keywords:
Problem-based learning
Software engineering
Software engineering education

ABSTRACT

The student learning process using real-life contextualization and soft skills are the factors that every student should acquire after completing a Software Engineering course. Problem-based learning (PBL) that promotes learning by doing is used to be implemented in the Software Engineering course in order to solve these issues. A study is conducted with the intention to investigate the introduction of implementing PBL in the Software Engineering course at the Faculty of Computer Science and Engineering, University of Jeddah. Thus, this paper aims to report the findings of introducing PBL to stimulate students' learning and skills. The PBL was implemented in the Software Engineering course as an ongoing teaching method for Fall Semester 2019/2020, and after completing this course, a set of the questionnaire is distributed to all students to get their perceptions of using PBL. The results indicated that PBL stimulates the students' learning by allowing them to be responsible for their learning, become self-directed learners and given the students the ability to be responsible for their own learning in a simulated environment, working with fellow students to achieve an outcome. These results are comparable with previous works on PBL in the Software Engineering course, and PBL has been emphasized as a new teaching paradigm in an ongoing review of software engineering undergraduate program by IEEE/ACM. Therefore, the result of this study is significant as it highlights the positive conclusion of PBL, and this led to the theoretical of Software Engineering education in reducing the gap between theory and practice in curriculum development.

1. Introduction

Software engineering is a discipline, which has a broad definition with many competence areas. It is an application-oriented focus that incorporates theories and methodologies from many different disciplines within engineering, science, and economics (Sørensen et al., 2018). In Guide to the Software Engineering Body of Knowledge (Bourque and Fairley, 2014) stated that software engineering is composed of more than 17 knowledge areas and related disciplines and that each of these represents a discipline in itself. These areas employ techniques of math, science, engineering, and design and are required for strong analytical and problem-solving skills as well as communication and interpersonal skills (Sedelmaier and Landes, 2014). Thus, software engineering is a combination of technical knowledge and soft skills abilities that every software engineering graduates have to engage during their study for professional development in the industry.

The main issue in teaching software engineering courses is for the lecturers to blend the technical knowledge and soft skills abilities during their lecturer. In addition, software engineering lecturers are faced with the teaching of concepts, which sometimes are not easy for inexperienced students to understand. It is essential to equip the students with adequate software engineering knowledge and soft skills to satisfy the industry demand. Currently, the lecturers have their own approach in conducting this course, and this is according to the previous study that there is no official consensus on how to teach Software Engineering course, as each institution adopts their own methods based on the experience of its professors (Marques et al., 2014). Most of the lecturers are practicing traditional approach in software engineering course by using expository lectures, exams, and complimentary assignments are mostly used by lectures because of the difficulty in changing the instructional process (Sancho-thomas et al., 2009; Marques et al., 2014)

* Corresponding Author.
Email Address: azzainol@uj.edu.sa (A. Zainol)
https://doi.org/10.21833/ijaas.2020.12.002
Corresponding author's ORCID profile:
https://orcid.org/0000-0001-7572-0449

that discouraging students in learning (Delgado et al., 2017; Souza et al., 2019). These teacher-centered educational methods do not support the practical development of competences and have limited learning efficiency (Shuto et al., 2016; Delgado et al., 2017).

It is important to note that students always consider software engineering courses as difficult to understand via theoretical approach because the course is based on tangible aspects that require the students to visualize and experience the software development from the beginning. Although it is possible to carry out the laboratories to practice the hand out exercises, still these classes are unable to provide students with real-life problems, decision making, and situations during software development phases. Thus, there is a huge gap left by the academic approach that should be given attention.

Hence, in order to overcome these situations, a student-centered approach using the Problem Based Learning (PBL) is suitable and recommended as it allows the development of competences, require motivation from students, students play an active role in the learning process and allow the students to participate in solving life situation (Richardson and Delaney, 2009; Brodie et al., 2008; Sørensen et al., 2018). PBL is an approach for reducing the gap between theory and practice when it is based on real problems, and it is able to provide students with the chance to propose the solution strategy that contributes to understanding the real-life problem and solution (Corrêa and Martins, 2016). In addition, PBL also allows the students to develop soft skills such as teamwork, self-assessment, problem-solving, leadership, and negotiation (Deep et al., 2019; Jabarullah and Hussain, 2019).

This study aims to investigate the implementation of a software engineering course at the Faculty of Computer Science and Engineering, the University of Jeddah, using PBL methodology from the students' perspective in the learning process. This is the first time PBL is introduced to the Software Engineering course at this faculty. Software Engineering is a programming course that is offered to fourth-year students for the computer science program. In order to prepare the Computer Science graduates to satisfy the industry demand, it is essential for the delivery of software engineering courses to reduce the gap between theory and practice using the PBL, as well as to equip the students with technical knowledge and soft skills. Hence, this study purposely conducted to answer the following research questions:

- Does PBL stimulate the students learning?
- Does PBL develop students' skills?

This paper is structured by explaining the PBL and its components, describing the methodology for this study, presents the result analysis and discussion, and concludes this study with a conclusion section.

2. Background review

This section describes the background review of PBL in general and specifically reviews the PBL in Software Engineering education at the university.

2.1. Problem-based learning (PBL)

PBL was first coined in the late 1960s at the new medical school at McMaster University (Richardson and Delaney, 2009; Graaff and Kolmos, 2007; Shuto et al., 2016; Sørensen et al., 2018) that aimed to equip students with clinical problem-solving and lifelong learning skills (Hung et al., 2008). It is an instructional methodology used by a student to learn through problem-solving technique by a "learn by doing" basis that required students to use their ability to think, in a gradual way, during the acquisition of knowledge related real-life problems (Hung et al., 2008; Graaff and Kolmos, 2007; Souza et al., 2019).

The PBL has the following characteristics (Delaney and Mitchell, 2002; Hung et al., 2008; Richardson and Delaney, 2009; Graaff and Kolmos, 2007):

- It is problem-oriented. The students enable learning of new context and skills by resolving the problems. The construction of knowledge is provided by problem-solving and is applied back to problem-solving. There is no hierarchical list of topics, such as in a conventional learning methodology.
- Students determine their pace of learning.
- PBL is self-directed. Students generate their own learning issues and processes, individually or collaboratively, through self and peer assessment.
- It is self-reflective. Students manage their understanding and are able to adjust their learning strategies.
- Tutors are only facilitators and never disseminators of knowledge.
- PBL is unique that involves collaboration among students, stresses the development of problem-solving skills within the context of professional practice, promotes effective reasoning and self-directed learning, and is aimed at increasing motivation for life-long learning.

Consequently, PBL is an active learning process whereby learning is no longer a standard process, but it transforms into a personalized process. Thus, studies (Tandogan and Orhan, 2007; Hung et al., 2008) stated that PBL is based on learning that:

- Knowledge is not transferred from professors to students. Learning is enhanced through social interaction and is facilitated when students are exposed to real-life situations.
- To understand every phenomenon, students need to have multiple perspectives related to this phenomenon.

- Knowledge is related to relevant contexts. In order for information to become knowledge, it is necessary to activate existing cognitive concepts and structures on the subject matter and to enable students to elaborate and assign new meanings.

2.2. PBL in software engineering education

Software Engineering was introduced 50 years ago with the intention to solve the software crisis (Naur and Randell, 1968; Randell, 1996; Mahoney, 2004). Since then, software engineering became a noteworthy discipline as the significance and importance of software to modern society has increased tremendously, software engineering education has been introduced to the university (Garousi et al., 2020; Maguire et al., 2019). The software engineering courses have been taught in the university in order to equip the graduate with the knowledge and skills of software development.

Teaching software engineering in the university is difficult as software systems are often complex and shifting, and the development of such systems can be extremely challenging. Thus, equipping software engineering graduates with the skills and abilities needed by the industry to meet these challenges is very important to the university (Mishra et al., 2007; Brodie et al., 2008). In addition, the previous studies have shown that there is a wide gap between software industry needs and education for prospective software engineers (Beckman et al., 1997; Garousi et al., 2020). In 1955, the industry perceived universities as not delivering computing science graduates that satisfied their requirements (Mengel, 1995), and in 2016, the university in the United Kingdom industry still perceives universities as not delivering computing science graduates that satisfy their requirements (Shadbolt, 2016). The reason behind this gap is not obvious, and a study suggested that this problem is related to curriculum content, delivery, or it could be both (Maguire et al., 2019).

Hence, ACM/IEEE introduced curriculum design that emphasis on supplementing students with adequate practical experiences, for the development of competences expected for Software Engineer professionals (Ardis et al., 2014). Software Engineers are required to have technical knowledge (Ghezzi and Mandrioli, 2005) and supplementary with soft skills abilities for software engineering practices such as leadership, teamwork, decision-making, negotiation, and self-reflection (Souza et al., 2019). Nevertheless, the development of these soft skills abilities is less supported in Software Engineering programs (Marques et al., 2014).

Many researchers in the literature have presented the use of PBL for improving teaching and learning in Software Engineering courses (Delaney and Mitchell, 2002; Delaney et al., 2003; Richardson and Delaney, 2009; Sørensen et al., 2018; Souza et al., 2019). A study (Brodie et al., 2008) has shown the integration of PBL into the software engineering major provides the students with many of the required foundation skills, including working in multidisciplinary teams, solving complex problems, communication and teamwork and provides an excellent framework in which to situate the technical content demanded of the course.

Furthermore, a study (Delaney et al., 2003) has presented PBL in the Software Engineering course that required students to work as a team to investigate and study relevant topics such as project management, software engineering, databases, network security, and client-server programming. This study has shown the success of small group PBL in developing software engineering skills, students express their enjoyment of the course despite the amount of work involved, and found themselves better prepared for their subsequent industrial work placements.

A recent study (Deep et al., 2019) was conducted to identify and determine the effects of PBL in improving and developing soft skills among undergraduates has revealed that PBL has a significant effect on improving students' soft skills and enhancing group learning, including overcoming communication conflicts.

Software engineering is not all about programming; yet, it is an applied and professional concern that has different roles that are filled by computing graduates, each of which implies different course content (Maguire et al., 2019). Thus, using PBL as a methodology in implementing a software engineering course prepares the students with adequate knowledge and soft skills in order to satisfy the industry demand.

3. Methodology

This section introduces the main aspects of the implementation of the PBL in the Software Engineering course by characterization of the study, a description of the aspects of data collection, as well as the analysis of results.

3.1. Characterization of the study

In this study, a software engineering course has been selected, and there are 24 students enrolled in this course for the first semester 2019/2020. This course is offered to fourth-year students from the Bachelor in Computer Science program. The students are grouped into four members in a group. The selection of the team members is given to the students to select their own group members. Every group is given a project that aims to develop an application using object-oriented analysis and design using Unified Modeling Language (UML). The software development involves requirements elicitation and analysis, design, implementation, and testing phase. Every phase has the artifacts that every group should deliver, and continuous assessment is implemented to ensure every group is on track for the software development process.

3.2. PBL applied to software engineering course

The PBL applied to the Software Engineering course is following a methodology adapted from (Ribeiro, 2008) that has a sequence of steps based on problems, as shown in Fig. 1.

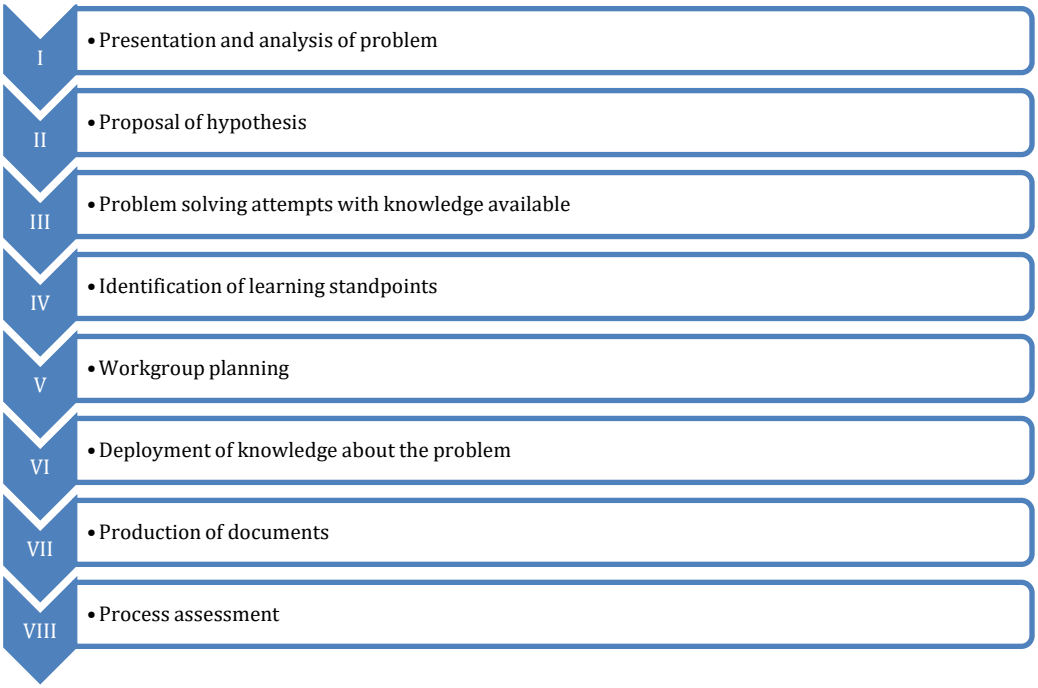

Fig. 1: The PBL methodology (Ribeiro, 2008)

The steps are designed as follows:

- **Step I:** Presentation and analysis of the problem: The first step of the methodology is the introduction of a specific problem presented by the tutor (teacher or professor) and analyzed by groups of students. The proposed projects are presented and elaborated on to the students, obviously. The projects are having the same size from a different domain and have been allocated to the group randomly.
- **Step II:** Proposal of hypothesis: After the presentation and problem analysis, the students, assisted by the tutor, discuss and raise possible causes of the problem and potential solutions for it. These hypotheses were related, for instance, identification of potential users, types of applications, domain, and others.
- **Step III:** Problem-solving attempts with knowledge available: After the hypothesis is clearly defined, students tried to provide answers intuitively and predict possible results for the solution of the problem based on their previous knowledge. These answers were, in general, related to the requirements identification obtained in Step 2.
- **Step IV:** Identification of learning standpoints: In this step, the professor already had the opportunity to present all the concepts of the course that had been covered by the students so far or which they had previous knowledge about. These concepts were in general related to the requirements elicitation, requirements analysis using UML, use case diagram, activity diagrams, and others.
- **Step V:** Workgroup planning: In this step, students planned the group work based on the problem to be solved and on the new knowledge acquired. In this software development, the following activities were carried out: requirements elicitation, analysis, design, implementation, and testing.
- **Step VI:** Deployment of knowledge about the problem: The students apply the knowledge acquired during problem-solving as many times as required in order to achieve the deliverables at the end of every phase, as shown in Table 1.

Table 1: Phases and deliverables

Phases	Artifacts
Requirements elicitation	List of requirements
Analysis	Use case diagram Use case specification Activity diagrams Sequence diagrams Collaboration diagrams
Design	Sequence diagrams (Design perspective) Collaborations diagrams (Design perspective) Class diagram Package diagram
Implementation	Component diagram Deployment diagram Prototype
Testing	Software test plan Software test report

- **Step VII:** Production of documents: The final step of this problem-solving cycle is related to the production of a document, as in Table 1, describing all the proposed solutions for the problem, which

will be presented to the tutor and all the other groups.

- **Step VIII:** Process assessment: In this step, the professor of the course carried out the formal presentation where each student in the group has to present their work, and assessments are based on the group work.

3.3. Data collection

After PBL is successfully applied in the Software Engineering course, a survey using questionnaire was distributed among all students that seek to explore student attitudes towards their participation in current PBL education in understanding the activities involve in every software development phases. The questions in the questionnaire are adopted from Mossuto (2009) in order to collect the students learning and skills. The survey was distributed using the Google form to collect the students' perceptions about implementing PBL in the Software Engineering course. The data collected are analyzed statistically using a descriptive statistic to show the data distribution and identify associations among variables.

4. Result and discussion

This section describes the data analysis to how the result of this study and its discussion. We have conducted an online survey, particularly to understand the students' opinions after completing the Software Engineering course using PBL. The first section in the survey is the demographic profile that is used to understand the students' background who is taking the Software Engineering course. Fig. 2 shows the number of students in their current semester, and most of the students are in the 7th semester. Fig. 3 illustrates the age of the students, and most of the students are 21 years old. Fig. 4 presents the students' experience in using PBL during their learning journey, and the majority of the students with 79% claimed this is their first time using PBL in the learning process.

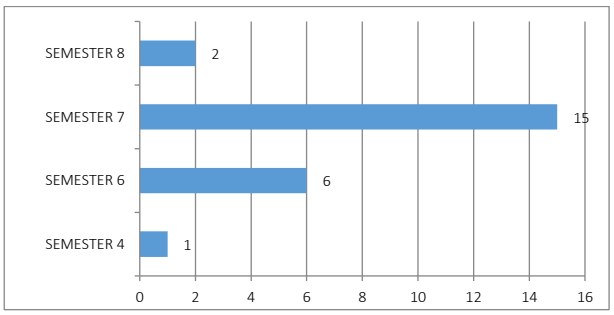

Fig. 2: The number of students in their current semester

The second section of the survey is describing students' opinions, perceptions and awareness of PBL in the Software Engineering course using the Likert scale ranging from 1 to 5; 1 represents strongly agree, 2 represents agree, 3 represents neutral, 4 represents disagree, and 5 represents strongly disagree. Table 2 shows the descriptive analysis of the result.

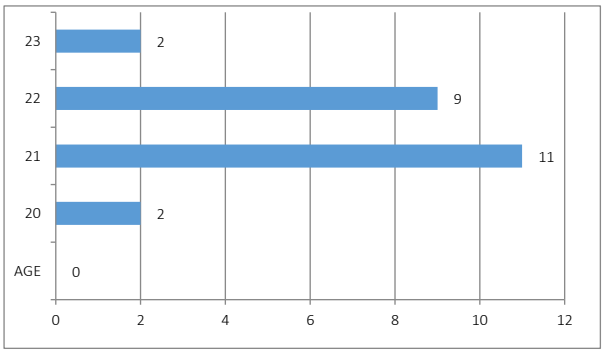

Fig. 3: The number of students based on age

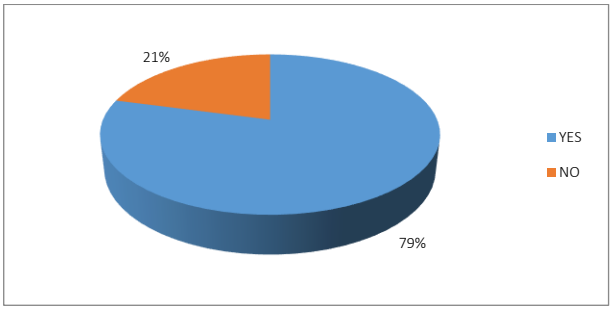

Fig. 4: The number of responses who are the first time using PBL in their learning process

Table 2 shows that the data are distributed towards strongly agree and agree, and the number of neutral scales also significant. It can be seen that most of the students are agree that PBL stimulates their learning by allowing them to be responsible for their learning and become self-directed learners. Software Engineering students should be competent in all areas of software development phases by using real-world assessment because it can measure the students' understanding of how to develop a real-world application and deal with the real situation. As stated in the literature, PBL is learning by doing indicated by the data gathered in Table 2 has given the students the ability to be responsible for their own learning in a simulated environment, working with fellow students to achieve an outcome. In addition, the result also revealed that students obtained the required knowledge to better understand group dynamics, deal with group issues, and it stimulated their learning because of its capacity to engage them. This finding is similar to a study by Souza et al. (2019) revealed that using PBL has highly motivated students to stimulate interactions with stakeholders and team members. Thus, for benchmarking purposes, the PBL implemented in our Software Engineering course has a similar finding as other previous works.

This study also showed the finding that skills in collaboration, communication, and research gained through problem-based learning projects enabled students to engage themselves in finding solutions to each problem discover during the software development process. The finding is supported by the studies (Delaney and Mitchell, 2002; Richardson

and Delaney, 2009; Sørensen et al., 2018) that indicated a similar finding in implementing PBL.

Table 2: The descriptive result

No	Perception	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
1	For problem-based learning to work effectively, problems (in your given case study) set need to be realistic and current.	1	17	4	1	1
2	For problem-based learning to work effectively, resources for self-directed research need to be easily available and relevant to the problem at hand.	9	8	6	1	0
3	The problem-based learning process stimulated my learning.	6	13	5	0	0
4	Problem-based learning stimulated my ability to work better for the group and myself.	8	10	6	0	0
5	Problem-based learning gave me the ability to voice my ideas effectively to the group.	4	11	8	1	0
6	Problem-based learning gave me the ability to build group interaction that enhanced my learning.	6	8	9	1	0
7	Problem-based learning gave me the ability to become aware of my limitations and to address them.	8	11	5	0	0
8	Problem-based learning gave me the ability to identify my ethical and moral obligations to other group members.	9	11	4	0	0
9	Problem-based learning gave me the ability to structure my learning based on the problem.	3	16	4	1	0
10	Problem-based learning gave me the ability to respect others within the group.	7	12	4	1	0
11	Problem-based learning gave me and the group a balanced workload.	6	12	5	1	0
12	The problem-based learning process stimulated my information processing skills.	6	12	5	1	0
13	Problem-based learning gave me the ability to enrich my learning by working in small groups.	8	9	6	1	0
14	Problem-based learning makes me responsible for my own learning.	6	12	5	1	0
15	Problem-based learning gave me the ability to understand the ethics involved in the decision-making	8	11	5	0	0
16	Problem-based learning gave me the ability to communicate effectively, both one on one and within a group environment.	8	9	6	1	0
17	Problem-based learning gave me the ability to become a self-directed learner.	7	12	5	0	0

The result of this study has revealed that implementing PBL in Software Engineering course given students to work individually and teamwork to deliver software artifacts, enrich learning in a small group indicate the knowledge sharing, effective communication skills and ability to respect others members which show the attitude. These results are corresponding with ACM/IEEE curriculum guideline for undergraduate degree programs in Software Engineering that stated Software Engineering students (Bourque and Fairley, 2014):

- Should have skills to work individually as well as on teams to develop and deliver quality software artifacts.
- Teamwork and professional practice play a vital role in Software Engineering.
- The importance of professional practice is concerned with the knowledge, skills, and attitudes that software engineers must possess to practice Software Engineering in a profession.

Consequently, the results of this study have highlighted and answered the research questions using PBL in the Software Engineering course has stimulated the students learning. Using PBL during the software development process has encouraged the students to develop collaboration teamwork, effective communication, and intrapersonal skills, which are required as a professional software engineer.

5. Conclusion

This study discussed the application of PBL in the learning process applied to the demands of real-life software development related to an undergraduate Software Engineering course. The students who participated in this study are focused on software development phases that begin with requirements elicitation, requirements analysis, software design, implementation, and testing. The results observed by the professor, according to the reports and to the follow-ups of students during activities related to the project, using PBL in Software Engineering allowed students to develop the following skills:

- Identify the relevant aspects of the problem being studied, carrying out important discussions within the context of the project.
- Acquire knowledge in the software development life cycle.
- Build a complete knowledge base in order to support the construction and management of problems due to software development.
- Develop logic reasoning, including analysis and synthesis.
- Carry out a critical evaluation of information concerning the problem.
- Develop the ability to communicate with group members and the public (in terms of clients).

The result of this study is significant to show that implementing PBL in Software Engineering course at

Faculty of Computer Science and Engineering, University of Jeddah highlighted positive and promising conclusion that PBL stimulates learning process and develop the students' soft skills. Therefore, this led to the theoretical of Software Engineering education in reducing the gap between theory and practice in curriculum development. This study contributes to the delivery methods in Software Engineering curriculum development in the Software Engineering education domain.

Compliance with ethical standards

Conflict of interest

The authors declare that they have no conflict of interest.

References

Ardis M, Budgen D, Hislop GW, Offutt J, Sebern M, and Visser W (2015). SE 2014: Curriculum guidelines for undergraduate degree programs in software engineering. Computer, 48: 106-109. https://doi.org/10.1109/MC.2015.345

Beckman K, Coulter N, Khajenoori S, and Mead NR (1997). Collaborations: Closing the industry-academia gap. IEEE Software, 14(6): 49-57. https://doi.org/10.1109/52.636668

Bourque P and Fairley RE (2014). Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0. IEEE Computer Society Press, Washington, USA.

Brodie L, Zhou H, and Gibbons A (2008). Steps in developing an advanced software engineering course using problem based learning. Engineering Education, 3(1): 2-12. https://doi.org/10.11120/ened.2008.03010002

Corrêa AGD and Martins VF (2016). Methodology applied problem-based learning in teaching HCI: A case study in usability evaluation of an online course. In: Fonseca D (Ed.), Handbook of research on applied e-learning in engineering and architecture education: 159-177. IGI Global, Pennsylvania, USA. https://doi.org/10.4018/978-1-4666-8803-2.ch008

Deep S, Salleh BM, and Othman H (2019). Improving the soft skills of engineering undergraduates in Malaysia through problem-based approaches and e-learning applications. Higher Education, Skills and Work-Based Learning, 9(4): 662-676. https://doi.org/10.1108/HESWBL-07-2018-0072

Delaney D and Mitchell GG (2002). PBL applied to software engineering group projects. In the International Conference on Information and Communication in Education, Government of Extremadura, Municipality of Mérida, Spain: 1093-1098.

Delaney JD, Mitchell G, and Delaney S (2003). Software engineering meets problem-based learning. The Engineers Journal, 57(6): 57-59.

Delgado D, Velasco A, Aponte J, and Marcus A (2017). Evolving a project-based software engineering course: A case study. In the IEEE 30th Conference on Software Engineering Education and Training, IEEE, Savannah, USA: 77-86. https://doi.org/10.1109/CSEET.2017.22

Garousi V, Giray G, Tuzun E, Catal C, and Felderer M (2020). Closing the gap between software engineering education and industrial needs. IEEE Software, 37(2): 68-77. https://doi.org/10.1109/MS.2018.2880823

Ghezzi C and Mandrioli D (2005). The challenges of software engineering education. In the International Conference on Software Engineering, Springer, St. Louis, USA: 115-127. https://doi.org/10.1145/1062455.1062578

Graaff E and Kolmos A (2007). History of problem-based and project-based learning. In: Graaff E and Kolmos A (Eds.), Management of change: 1-8. Brill Sense Publishers, Rotterdam, Netherlands. https://doi.org/10.1163/9789087900922_002

Hung W, Jonassen DH, and Liu R (2008). Problem-based learning. In: Jonassen D, Spector MJ, Driscoll M, Merrill MD, van Merrienboer J, and Driscoll MP (Eds.), Handbook of research on educational communications and technology: 485-506. Routledge, Abingdon, UK.

Jabarullah NH and Hussain HI (2019). The effectiveness of problem-based learning in technical and vocational education in Malaysia. Education+Training, 61(5): 552-567. https://doi.org/10.1108/ET-06-2018-0129

Maguire J, Draper S, and Cutts Q (2019). What do we do when we teach software engineering? In the 1st UK and Ireland Computing Education Research Conference, Association for Computing Machinery, Canterbury, UK: 1-7. https://doi.org/10.1145/3351287.3351295

Mahoney MS (2004). Finding a history for software engineering. IEEE Annals of the History of Computing, 26(1): 8-19. https://doi.org/10.1109/MAHC.2004.1278847

Marques MR, Quispe A, and Ochoa SF (2014). A systematic mapping study on practical approaches to teaching software engineering. In the IEEE Frontiers in Education Conference (FIE) Proceedings, IEEE, Madrid, Spain: 1-8. https://doi.org/10.1109/FIE.2014.7044277 **PMCid:PMC3945822**

Mengel ME (1995). Present and projected computer manpower needs in business and industry. In the Conference on Training Personnel for the Computing Machine Field, Wayne University Press, Detroit, USA, 1: 4-9.

Mishra A, Ercil Cagiltay N, and Kilic O (2007). Software engineering education: Some important dimensions. European Journal of Engineering Education, 32(3): 349-361. https://doi.org/10.1080/03043790701278607

Mossuto M (2009). Problem-based learning: Student engagement, learning and contextualized problem-solving. Occasional Paper, National Centre for Vocational Education Research Ltd., Adelaide, Australia.

Naur P and Randell B (1968). Software engineering: Report on a conference. In the NATO Science Committee Conference, Garmisch, Germany.

Randell B (1996). The 1968/69 Nato software engineering reports. In the Dagstuhl-Seminar 9635: History of Software Engineering, Wadern, Germany: 37-41.

Ribeiro LRDC (2008). Aprendizagem baseada em problemas (PBL) na educação em engenharia. Revista de Ensino de Engenharia, 27(2): 23-32. https://doi.org/10.7476/9788576002970

Richardson I and Delaney Y (2009). Problem based learning in the software engineering classroom. In the 22nd Conference on Software Engineering Education and Training, IEEE, Hyderabad, India: 174-181. https://doi.org/10.1109/CSEET.2009.34

Sancho-Thomas P, Fuentes-Fernández R, and Fernández-Manjón B (2009). Learning teamwork skills in university programming courses. Computers and Education, 53(2): 517-531. https://doi.org/10.1016/j.compedu.2009.03.010

Sedelmaier Y and Landes D (2014). Software engineering body of skills (SWEBOS). In the IEEE Global Engineering Education Conference, IEEE, Istanbul, Turkey: 395-401. https://doi.org/10.1109/EDUCON.2014.6826125

Shadbolt N (2016). Shadbolt review of computer sciences degree accreditation and graduate employability. Innovation and

Skills and Higher Education Funding Council for England, London, UK.

Shuto M, Washizaki H, Kakehi K, Fukazawa Y, Yamato S, and Okubo M (2016). Learning effectiveness of team discussions in various software engineering education courses. In the IEEE 29th International Conference on Software Engineering Education and Training, IEEE, Dallas, USA: 227-231. https://doi.org/10.1109/CSEET.2016.31 **PMid:24980145**

Sørensen L, Falch M, and Skouby KE (2018). Report on problem based learning for software engineering. In the Workshop on Ph.D. Software Engineering Education: Challenges, Trends, and Programs (SWEPHD'18).

Souza M, Moreira R, and Figueiredo E (2019). Students perception on the use of project-based learning in software engineering education. In the 33rd Brazilian Symposium on Software Engineering, Association for Computing Machinery, Salvador, Brazil: 537-546. https://doi.org/10.1145/3350768.3352457 **PMid:30407608**

Tandogan RO and Orhan A (2007). The effects of problem-based active learning in science education on students' academic achievement, attitude and concept learning. Online Submission, 3(1): 71-81. https://doi.org/10.12973/ejmste/75375