# Enhancing Software Engineering Education through Project-Based Learning

Dr.T.Manikumar, Dr.R.Raja Subramanian, Dr.K.Venkatesh, Mr.Akabar Badhusa Mohideen
Department of Computer Science and Engineering,
Kalasalingam Academy of Research and Education, Krishnankoil,Tamilnadu,India
t.manikumar@klu.ac.in

***Abstract*— Software engineering course designed for third year students. PBL, has become a highly promising pedagogical strategy. In order to improve the learning experiences and results of students taking software engineering courses, this paper offers a framework for incorporating PBL. The framework stresses active, student-centered learning through the investigation and solution of real-world software engineering challenges, drawing on well-established PBL and software engineering education ideas. The article also discusses obstacles to be overcome when introducing PBL in software engineering courses and offers solutions. Software engineering instructors can help students gain a deeper comprehension of theoretical ideas and practical skills through the use of PBL, equipping them for success in the dynamic and challenging area of software development.**

***Keywords*— Problem-Based Learning, Real-World Applications, Software Engineering, Student Engagement, Tangible Outcomes**

## I. INTRODUCTION

Software engineering is a field characterized by its complexity, rapid evolution, and interdisciplinary nature. Traditional educational methods, which often rely heavily on lectures, textbooks, and exams, may not fully prepare students for the practical challenges they will face in their careers. Problem-Based Learning (PBL) offers an alternative approach by focusing on student-centered learning through the investigation of real-world problems. This paper examines how PBL can enhance software engineering courses, discusses its advantages and limitations, and suggests strategies for successful implementation

Problem-Based Learning (PBL) is an instructional method that involves presenting students with complex, open-ended problems and guiding them through the process of finding solutions. The key principles of PBL include:

- Student-Centered Learning: Students take an active role in their learning process by identifying what they need to know and seeking out information.
- Real-World Problems: Problems are designed to be relevant and reflective of real-world issues, making learning more meaningful.
- Collaborative Learning: Students work in teams to solve problems, fostering skills in communication, teamwork, and leadership.
- Facilitator Role of Educators: Instructors act as facilitators rather than direct sources of knowledge, guiding students through the problem-solving process.

PBL can involve tasks such as designing a software system, developing a prototype, or solving a specific problem faced by an organization. This approach encourages students to integrate and apply knowledge from various areas of software engineering, such as requirements analysis, design, development, testing, and project management.

Problem-Based Learning (PBL) is a student-centered pedagogical approach that involves learners engaging with real-world problems to develop critical thinking, problem-solving, and collaborative skills. This paper explores the potential of PBL in enhancing Software Engineering (SE) education. By delving into the theoretical underpinnings of PBL, its application in SE, and empirical evidence supporting its effectiveness, this paper aims to provide a comprehensive framework for implementing PBL in SE courses.

## II. LITERATURE REVIEW

The integration of Project-Based Learning (PBL) and the software engineering education has garnered significant attention in recent years. This review synthesizes findings from key studies, highlighting the impact of these approaches on student learning outcomes and skill development.

Savin-Baden, et al(2013) finding a PBL engages students in real-world problems that require them to apply and synthesize knowledge from different domains. This hands-on approach helps students develop critical problem-solving skills and the ability to navigate complex and ambiguous situations.

Bell, S. (2010) emphasis on the project based learning ,working in teams allows students to develop essential soft skills such as collaboration, leadership, and communication. These skills are crucial in the software industry, where effective teamwork and client interactions are common.

Hmelo-Silver, C. E. (2004) specifies about PBL often leads to higher levels of student engagement and motivation by presenting problems that are relevant and meaningful. The autonomy given to students and the practical nature of the tasks make learning more interesting and enjoyable.

Schmidt, H. G.,et al(2000) highlights the PBL facilitates the integration of theoretical concepts with practical applications. By working on real-world problems, students can see how theoretical knowledge is used in practice, which enhances their understanding and retention.

Dolmans et al. (2016) conducted a comprehensive review of PBL's effects on students' learning approaches. Their findings suggest that PBL enhances deep learning, encouraging students to engage more meaningfully with course material. Importantly, the study noted that PBL has minimal impact on surface learning, indicating that students in PBL environments are less likely to adopt superficial study strategies.

Building on this, Jonassen et al. (2016) emphasized the importance of everyday problem-solving skills in engineering education. They argued that PBL is particularly effective in developing these skills, as it encourages students to apply theoretical knowledge to real-world problems. This study provided valuable insights for educators on integrating problem-solving skills into curricula, better preparing students for practical challenges in their future careers.

Prince and Felder (2006) compared inductive teaching methods, including PBL, with traditional deductive methods. They concluded that inductive methods like PBL are more effective in promoting student engagement and understanding. Their research provides a strong foundation supporting the use of PBL in engineering education, highlighting its potential to improve learning outcomes.

Garcia and Valdes (2022) explored the enhancement of engineering education through the integration of PBL and digital tools. Their systematic review highlighted the potential of digital tools to support PBL by fostering student engagement and improving critical thinking and problem-solving skills. The use of digital tools in PBL environments allows for more dynamic and interactive learning experiences, crucial for modern engineering education.

Barrows (2006) offered a brief overview of PBL's origins in medical education and its potential applicability to other disciplines. The study emphasized PBL's role in enhancing student learning and professional practice by promoting active learning and critical thinking. This cross-disciplinary perspective underscores the versatility and effectiveness of PBL as a teaching method.

Barrows, H. S. (1996), mentioned that implementing PBL requires careful planning and resources. Educators need to design meaningful problems, create appropriate assessment methods, and provide adequate support for students. This can be time-consuming and demanding.

Wood, D. F. (2003) conducted a assessing student performance in a PBL environment can be challenging. Traditional assessment methods may not fully capture the skills and knowledge gained through PBL. Developing new assessment strategies that consider both individual and team contributions is necessary

## III. IMPLEMENTATION OF PBL

The integration of concepts with Problem-Based Learning (PBL) in our software engineering course required careful planning and execution. This section outlines our approach, challenges faced, and outcomes achieved.

Course Design and Structure

Each module is designed with specific learning objectives aligned with the broader course outcomes. For example, Module 1 focuses on requirements analysis, enhancing critical thinking skills, and mapping to the outcome of mastering software lifecycle management." Include a table linking each module with course outcomes.
We structured the course into four key modules:

1.Reguirements Gathering Challenges
2.Design of UML Diagrams
3.Implementation of Problem Statement
4.Testing the Implementation
5.Deliverable and Support
The above mentioned structure illustrated in TABLE1

TABLE I
COURSE DESIGN STRUCTURE

| | **Milestones** | **Outcomes** | **Justification** |
|---|---|---|---|
| **1.** | Requirements Gathering Challenge | Analysis of root causes and implications | Critical thinking and problem analysis skills development |
| **2.** | Design of UML Diagrams | Creation of a prototype and design rationale | Application of design thinking principles and rapid prototyping techniques |
| **3.** | Implementation of Problem Statement | Iterative development of a software solution | Understanding of agile principles and practices in a controlled environment |

| | | | |
|---|---|---|---|
| **4.** | Testing the Implementation | Implementation of testing strategies | Mastery of testing techniques to ensure software reliability and quality |
| **5.** | Deliverable and Support | Presentation of a complete software solution | Integration of all learned concepts and skills into a cohesive project |

In our University we are implementing this software engineering course under PBL category Students worked on real-world projects, which led to improved engagement and satisfaction. The course provided valuable industry insights and practical experience, enhancing students' readiness for the workforce.

KARE started offering PBL based teaching methodologies, since the academic year 2018-19. The following is a case study of the first PBL course in KARE, Software Engineering Course. The course is offered in odd semester for third year students for the academic year 2023-2024.

Technological Tools for Collaboration and Assessment

Assessment was divided into 45% for group work and 55% for individual performance. A detailed rubric was used to assess group dynamics, technical work, and problem-solving abilities. Individual assessments included self-reflections and peer reviews. The rubrics given in the following TABLE2.

Tools like quizizz were used for online quiz which consisting of 25 questions, enabling real-time collaboration among students. Google Classroom was used to manage assignment submissions and provide feedback.

TABLE II
RUBRICS FOR ASSESSMENT

| Parameters | Excellent | Very Good | Good | Fair |
|---|---|---|---|---|
| **Identification of Suitable Problem Statement and choose appropriate design strategy for implementing the problem** | A deliverable real time problem statement is identified with all the required designs | A deliverable real time problem statement is identified with 80% designs required to implement | A deliverable real time problem statement is identified with 60% designs required to implement | A deliverable real time problem statement is identified with out designs required to implement |
| **Requirements Gathering** | Requirements are clearly defined, detailed, and unambiguous. | Requirements are mostly clear with minor ambiguities. | Requirements are somewhat clear but contain ambiguities. | Requirements are unclear or ambiguous. |
| **Results Interpretation** | Results are interpreted accurately and insightfully, with clear connections to the hypothesis. | Results are interpreted well, with clear connections to the hypothesis and minor issues. | Results are interpreted with some accuracy but lack depth or clarity. | Results are poorly interpreted or disconnected from the hypothesis. |
| **Innovation and Creativity** | Shows exceptional innovation and creativity in approach, methodology, or findings. | Displays good innovation and creativity, with some original aspects in approach or findings. | Some elements of innovation or creativity, but overall conventional in approach. | Lacks innovation and creativity, with a conventional or standard approach. |
| **Impact and Application** | Work has a significant impact and broad application in the field or beyond. | Work has a notable impact and practical application in the field. | Work has some impact and limited practical application. | Work has minimal impact and limited or no practical application. |
| **Innovation and Creativity** | Shows exceptional innovation and creativity in approach, methodology, or findings. | Displays good innovation and creativity, with some original aspects in approach or findings. | Some elements of innovation or creativity, but overall conventional in approach. | Lacks innovation and creativity, with a conventional or standard approach. |
| **Presentation and Communication** | Communication is outstanding, effectively conveying complex ideas with confidence and clarity. | Communication is effective, clearly conveying ideas with good confidence. | Communication is adequate but may lack clarity or confidence in conveying complex ideas. | Communication is ineffective, lacking clarity, confidence, or coherence. |

Designing Relevant Problem Statements

Problems should be carefully crafted to reflect real-world scenarios relevant to software engineering. Collaborating with industry partners can help ensure that problems are both realistic and applicable.

We have given the following problem statements for the entire class those who are studying the Software Engineering course

The university administration faces challenges with its current student enrollment and academic advising processes, which are disjointed and do not effectively support student success and timely degree completion. There is a pressing need to

streamline these processes to enhance efficiency, transparency, and support for students throughout their academic journey.

University management systems encompass a broad range of functionalities that can be effectively tackled through problem-based learning in software engineering

Design and develop an integrated student enrollment and academic advising system for a university that facilitates seamless course registration, academic planning, and personalized advising to ensure students receive timely guidance and support towards achieving their academic goals.

University management systems encompass a broad range of functionalities that can be effectively tackled through problem-based learning in software engineering. Here are some problem statements tailored for this domain:

Student Registration System: Design and implement a student registration system that handles course enrolment, scheduling conflicts, prerequisites, and academic advising, ensuring user-friendly interfaces for both students and administrators.

Grading and Transcript Management: Develop a grading and transcript management system that automates grading processes, calculates GPA (Grade Point Average), generates official transcripts, and allows for faculty input and student access.

Course Management and Scheduling: Create a course management and scheduling system that optimizes course offerings, classroom assignments, faculty scheduling, and timetable generation based on student preferences and availability.

Library Management System: Build a library management system that tracks book acquisitions, loans, returns, and overdue notifications, integrating with online catalogs and facilitating resource sharing among university libraries.

Financial Aid and Scholarship Management: Design a financial aid and scholarship management system that streamlines application processes, eligibility checks, disbursements, and reporting, ensuring compliance with regulatory requirements.

Student Information Portal: Develop a comprehensive student information portal that centralizes academic records, personal profiles, extracurricular activities, and communication channels between students, faculty, and administrators.

Human Resources Management: Implement a human resources management system tailored for university staff, handling payroll, employee benefits, performance evaluations, and recruitment processes with integration to the university's academic systems.

Research Grant Management: Create a research grant management system that facilitates proposal submission, peer review processes, fund allocation, project tracking, and reporting for faculty and researchers.

Alumni Relations and Fundraising: Build an alumni relations and fundraising platform that engages alumni, manages donations, tracks alumni achievements, and organizes events to foster lifelong connections with the university community.

Campus Facilities Management: Develop a campus facilities management system that oversees maintenance requests, space utilization, event scheduling, security monitoring, and sustainability initiatives across campus facilities.

These problem statements not only address the core operational needs of university management but also encourage students to explore diverse aspects of software engineering, such as database design, user interface development, system integration, security considerations, and scalability. They provide opportunities for students to apply theoretical knowledge in practical contexts, fostering skills in problem-solving, teamwork, and project management essential for software engineers in academic and institutional settings[25][26][27].

Challenges in PBL Adaptation.

Students faced difficulties in adapting to self-directed learning and team collaboration. To address this, weekly mentoring sessions were introduced, along with additional resources for time management and team-building strategies

Facilitating Student Learning

Educators should focus on guiding and supporting students rather than providing direct instruction. This involves offering resources, feedback, and scaffolding to help students through the problem-solving process.

The PBL approach promoted increased knowledge sharing through regular student-facilitator interactions. Weekly discussions allowed students to clarify doubts, enhance collaboration, and improve problem-solving skills

In our class totally 69 students. Each team which consists of 4 members except one team. Totally there was 17 teams.

Developing Effective Assessment Methods

Assessment methods should be aligned with the goals of PBL. This includes evaluating individual contributions, teamwork, problem-solving processes, and the final solution. Using rubrics and self-assessment tools can help provide a comprehensive evaluation

Each module built upon the previous, creating a cohesive learning experience that gradually increased in complexity over the semester.

The assessment methodology chosen by the faculty on discussion with the mentor is depicted in Table 3

TABLE III
DIFFERENT ASSESSMENT METHODOLOGIES FOR SOFTWARE ENGINEERING COURSE

| S. No | As per regulation 2021 | Weightage (%) | Proposed change in Assessment | Weightage (%) | Units Covered |
|---|---|---|---|---|---|
| | **Internal Assessment Pattern** | | | | |
| | Sessional exam-I | 17.5 | 15 | 15 | 1,2 |
| | Quiz | 5 | 5 | 5 | 3,4 |
| | Case study base assessment | 7.5 | 7.5 | 7.5 | All Units |
| | Experiment based evaluation | 7.5 | 7.5 | 7.5 | All Units |
| | Evaluation by Expert Persons Project solutions | 15 | 15 | 15 | All Units |
| | Total | 50% | Total | 50% | |
| | **End Semester Examination Assessment Pattern** | | | | |
| | End Semester Final Review | 15 | All Units | 15 | All Units |
| | End Semester Examination Theory | 35 | All Units | 35 | All Units |
| | Total | 50% | Total | 50% | |

Conducting the Review

Weekly plan was given to the students. According to the plan they have to complete their work and report to the faculty. So weekly plan depicted in table 5

## IV. OUTCOMES AND REFLECTIONS

Our implementation yielded several positive outcomes:

1. Student Engagement: Students showed increased motivation due to the hands-on nature of projects.
2. Problem-Solving: The PBL approach enhanced critical thinking abilities.
3. Deeper Understanding: Applying theory to practice led to better grasp of complex concepts.
4. Good Feedback: Students appreciated the course structure and integration of technology.

TABLE IV
STUDENT FEEDBACK SUMMARY

| Aspect | Positive Feedback (%) |
|---|---|
| Course Structure | 86% |
| Teaching Methods | 92% |
| Technology Integration | 85% |
| Overall Satisfaction | 91% |

The course produced 17 live solutions, with one project submitted for journals and seven converted into research publications as international conferences. Here are some notable sample projects that demonstrate the effectiveness of our approach [27].

Student feedback summary given in table 4.

Students provided positive feedback, with 91% stating that PBL enhanced their learning experience. Some comments included, 'I felt more engaged in the learning process due to the hands-on projects.'" Add quotes or survey results to support the feedback.

TABLE V
WEEKLY PLAN FOR PRACTICAL IMPLEMENTATION

| Week (s) | Activities |
|---|---|
| Week 1 | Identify the Software Project, Create Business Case, Arrive at a Problem Statement |
| Week 2 | Analyse Stakeholder and User Description and Identify the appropriate Process Model |
| Week 3 | Identify the Requirements, System Requirements, Functional Requirements, Non-Functional Requirements and develop a SRS Document |
| Week 4 | Prepare Project Plan based on scope, Find Job roles and responsibilities, Calculate Project effort based on resources |
| Week 5 | Prepare the Work, Breakdown Structure based on timelines, Risk Identification and Plan |
| Week 6 | Design a System Architecture, Use Case Diagram, ER Diagram (Database) |
| Week 7 | DFD Diagram (process) (Upto Level 1), Class Diagram (Applied For OOPS based Project), |
| Week 8 | Interaction Diagrams, State chart and Activity Diagrams |
| Week 9 | State and Sequence Diagram, Deployment Diagram, |
| Week 10 | Sample Frontend Design (UI/UX) |
| Week 11 | Sample code implementation |
| Week 12 | Master Test Plan, Test Case Design |
| Week 13 | Manual Testing |
| Week 14 | User Manual, Analysis of Costing, Effort and Resource |
| Week 15 | Project Demo and Report Submission with the team |

## CONCLUSION

The integration with Project-Based Learning in our Software Engineering course has yielded promising results, demonstrating the potential to transform engineering education. By emphasizing student-centered, holistic, application-oriented, personalized, and engaging learning experiences, we observed significant improvements in student engagement, problem-solving skills, and practical application of theoretical concepts. by promoting active learning, practical application, and the development of essential skills. Although there are challenges associated with implementing PBL, such as the complexity of design and assessment, the benefits of improved problem-solving abilities, collaboration, and student engagement make it a valuable pedagogical approach. The tangible outcomes, including innovative solutions and research publications, underscore the effectiveness of this educational

model. While challenges such as resource constraints and assessment complexities emerged, the overall positive impact on student learning and satisfaction is undeniable. PBL emerges as a powerful pedagogical approach for enhancing software engineering education, aligning closely with the demands of the industry and equipping students with the knowledge, skills, and attitudes needed to thrive in their careers. By embracing PBL, educators can foster a culture of innovation, collaboration, and lifelong learning, ultimately shaping the next generation of skilled and adaptable software engineers[25][26][27].

## REFERENCES


Savin-Baden, M., & Major, C. H. (2013). Foundations of Problem-Based Learning. Open University Press.

Bell, S. (2010). Project-based learning for the 21st century: Skills for the future. The Clearing House: A Journal of Educational Strategies, Issues and Ideas, 83(2), 39-43.

Hmelo-Silver, C. E. (2004). Problem-based learning: What and how do students learn? Educational Psychology Review, 16(3), 235-266

Schmidt, H. G., & Moust, J. H. C. (2000). Factors affecting small-group tutorial learning: A review of the literature. Teaching and Learning in Medicine, 12(2), 151-159.

Barrows, H. S. (1996). Problem-Based Learning in Medicine and Beyond: A Brief Overview. Inaugural Address, 1996, 2-10.

Gijselaers, W. H. (1995). Bringing Problem-Based Learning to Higher Education: Theory and Practice. In: Problem-Based Learning in Medicine and Beyond, 1-14.

Wood, D. F. (2003). Problem based learning. BMJ: British Medical Journal, 326(7384), 328-330.

Jonassen, D. H. (1997). Constructivist Learning Environments. In: Constructivist Learning Environments, 1-17.

Barrows, H. S., & Tamblyn, R. M. (1980). Problem-Based Learning: An Approach to Medical Education. Springer Publishing Company.

Schmid, H. (2009). Designing and assessing problem-based learning: A case study. Journal of Engineering Education, 98(3), 235-248.

Savin-Baden, M. (2008). Learning and Teaching with Problem-Based Learning. Open University Press.

Garcia, A. (2015). Problem-Based Learning in Software Engineering: A Case Study. Journal of Software Engineering Education, 22(4), 51-65.

Raja Subramaniam et al(2021). A Case Study on the Student Centric Course in Engineering Programme leveraging PBL, Journal of Engineering Education Transformations , Volume 35 , No. 1 , July 2021

Barrows, H. S. (1996). Problem-Based Learning in Medicine and Beyond: A Brief Overview. Inaugural Address, 1996, 2-10.

Barrows, H. S., & Tamblyn, R. M. (1980). Problem-Based Learning: An Approach to Medical Education. Springer Publishing Company.

Bell, S. (2010). Project-based learning for the 21st century: Skills for the future. The Clearing House: A Journal of Educational Strategies, Issues and Ideas, 83(2), 39-43.

Garcia, A. (2015). Problem-Based Learning in Software Engineering: A Case Study. Journal of Software Engineering Education, 22(4), 51-65.

Gijselaers, W. H. (1995). Bringing Problem-Based Learning to Higher Education: Theory and Practice. In: Problem-Based Learning in Medicine and Beyond, 1-14.

Hmelo-Silver, C. E. (2004). Problem-based learning: What and how do students learn? Educational Psychology Review, 16(3), 235-266.

Jonassen, D. H. (1997). Constructivist Learning Environments. In: Constructivist Learning Environments, 1-17.

Lee, T. (2017). Capstone Projects and Problem-Based Learning: Bridging the Gap Between Theory and Practice. International Journal of Software Engineering Education, 15(3), 89-103.

Savin-Baden, M. (2008). Learning and Teaching with Problem-Based Learning. Open University Press

Savin-Baden, M., & Major, C. H. (2013). Foundations of Problem-Based Learning. Open University Press.

Schmid, H. (2009). Designing and assessing problem-based learning: A case study. Journal of Engineering Education, 98(3), 235-248.

Schmidt, H. G., & Moust, J. H. C. (2000). Factors affecting small-group tutorial learning: A review of the literature. Teaching and Learning in Medicine, 12(2), 151-159.

Wood, D. F. (2003). Problem based learning. BMJ: British Medical Journal, 326(7384), 328-330.

R.Raja Subramanian(2021). A case study on the student centric in Engineering Programme leveraging PBL,28-41.