Towards an Evaluation Process around Active Learning based Methods

Camelia Şerban and Andreea Vescan
Faculty of Mathematics and Computer Science
Babeş-Bolyai University, Cluj-Napoca,România
{camelia, avescan}@cs.ubbcluj.ro

***Abstract*—This Research to Practice Full Paper proposes a novel incremental approach in teaching a Software Engineering related course, underpinned by active-learning methods. Teaching a Software Engineering related course for undergraduate students is a challenging task due to extremely frequent changes that appeared in programming paradigms and in software process development methodologies. In this ever-changing domain, which presents us with opportunities and challenges every single day, it might be difficult and almost impossible for teachers to predict what are those knowledge that will be of use to students, on the long term, maybe even for five years from now. This rapid growth in the software development evolution has led to a gap between how teachers do knowledge transfer to students and how students acquire these knowledge. To bridge this gap there is a need to change how we teach these subjects. Thus, the new theory of education suggests that learner should be in the center of the learning process and the instructors playing an advising and facilitating role. A shift in education theory to a more student-centered approach using active learning is recommended because this approach has its own role to make the students creative and competent in their study. In this respect, active learning's main drive is to put the responsibility of learning at the hands of the learners themselves and to delegate the role of facilitator to the teacher. In this paper we provide novel approaches in teaching an undergraduate Software Engineering related course at the Babeş-Bolyai University. Its contribution is threefold: firstly, we introduce a new learning process design which selects active learning methods intertwining in teaching this course; secondly, we aim to study the impact of applying active learning methods on students' grades over the three academic years. Thirdly, we investigate students perceptions, their feedback and learning experiences on the use of applying active learning methods. Also, the paper reviews challenges and constraints that we faced while trying to teach this course. The analysis results show the effectiveness of our approach. Also, our students have expressed a high level of satisfaction, and in a survey, they indicated that the skills they learned in the course are highly applicable to their careers, facilitating their interviews for the engagement within IT companies.**

***Index Terms*—Project-based learning, formative assessment, summative assessment, multiple choice exam, student experience.**

I. Introduction

Teaching a Software Engineering related course for undergraduate students is a challenging task due to extremely frequent changes that appeared in programming paradigms and in software process development methodologies. These changes are caused by rapidly development of technology and software systems evolution. In this ever-changing domain, which presents us with opportunities and challenges every single day, it might be difficult and almost impossible for teachers to predict what are those knowledge that will be of use to students, on the long term, maybe even for five years from now.

In the context mentioned above, the learning methods applied today, have to be linked to the development of students' certain skills that ensures their adaptation to whatever the next decade might bring. This becomes possible when the instructors are using the appropriate teaching and learning methodology and the learners are active and empowered in the teaching-learning process.

Thus, the new theory of education suggests that learner should be in the center of the learning process and the instructors playing an advising and facilitating role. A shift in education theory to a more student-centered approach using active learning [1]–[3] is recommended because this approach has its own role to make the students creative and competent in their study. In this respect, active learning's main drive is to put the responsibility of learning at the hands of the learners themselves and to delegate the role of facilitator to the teacher.

A student-centered approach encourages students to "discover" knowledge by themselves, working at their own individual speed or in groups in a minimally guided environment, with the lab instructor offering support, encouraging their imagination and creativity [2]. Also, these active learning methods should facilitate for the students to acquire knowledge and skills in communication, teamwork, active listening, moderation techniques, interactive presentations, oral and nonverbal communication, conflict management and related issues in a holistic way.

The current paper proposes a novel incremental approach in teaching a Software Engineering related course, underpinned by active-learning methods. This course is taught at our faculty within the Computer Science Curriculum for undergraduate students. We investigate course evolution over three academic years through the analysis of active learning methods used, students results and their surveys.

The contribution of this paper is threefold: *firstly*, we introduce a novel learning process design based on active learning methods that are intertwined in teaching this course. Students are encouraged to design their assessment throughout the entire learning process, being actively involved, and using a design thinking approach [4], [5]. *Secondly*, an empirical investigation is conducted, providing the course evolution over three academic years of teaching *Advance Programming Methods* course, at the Babeş-Bolyai University, in order to study the impact of applying active learning methods on students' outcomes. *Thirdly*, we aim to investigate the students perceptions, their feedback and learning experiences about applying active learning methods.

The rest of the paper is organized as follows. Section II describes the analyzed course emphasising its contents and objectives. Section III, presents the proposed active learning process design and outlines the research questions. Section IV encompasses the quantitative and qualitative analysis of the investigation done. Section VI states the results and outcome of our exploration regarding the impact of using active learning, outlining future developments.

II. Course Description

The *Advanced Programming Methods* – APM course, described in this paper, is the third course from a package of five introductory courses linked to Software Engineering domain, that are being taught at our faculty within the Computer Science Curriculum for all undergraduate students. Two out of these five courses, *Fundamental of Programming*–FP (semester 1) and *Object Oriented Programming*–OOP (semester 2) serve as prerequisite for the APM course. Figure 1 presents the package of Software Engineering courses, emphasising their timeline and the programming languages used.

A. Course Objectives

The main objective of the *Advanced Programming Methods* course, comprised in the course syllabus, states that "Students have to be able to develop small to medium applications using the main concepts and mechanisms defined by object orientation programming paradigm - OOP, together with design strategies expressed in terms of principles, heuristics and rules, and use/build well defined software architectures for these applications." The languages used are Java and C#.

Having in mind this objective, three important knowledge areas - **FDA** - have to be acquired by our students at APM course [6]:

1) The main **F**undamental concepts, principles and mechanisms defined by OOP paradigm - **F**.
2) **D**esign principles, heuristics and rules that act as strategies implied in designing the system - **D**.
3) Software **A**rchitectures that aim to define the main constituent elements of the system architecture and to establish rules to wire them together - **A**.

These three types of knowledge areas are introduced even from the FP course, then resume at OOP course, and then again reinforce at APM course in a cyclic learning approach. This means that, every concept from the FDA is introduced at FP course and it has to reach a maturity level at APM course, going through all the six levels of learning from Bloom's taxonomy [7], taxonomy later refined by [8]. A learner performing at a higher level of learning from the Bloom's taxonomy is expected to be able to perform also at the lower levels in the cognitive hierarchy.

We have applied this approach of cyclic learning defined by Bloom' taxonomy at FP, OOP and APM courses as illustrates Figure 2.

Having all the above reasoning into account, the students' *competences* comprised in the APM course syllabus can be summarized as follows:

- Ability to use the concepts, mechanism and principles of object oriented analysis and design.
- Good programming skills in Java and C# programming languages.
- Ability to apply design patterns in different contexts.
- Ability to build software projects with clear separations on architectural layers and by following the main phases in software applications development.

B. Course Content

The course meets formally three times a week: 2 hours are for lectures classes (9 lectures using Java programming language and 5 lectures are in C#), 2 hours are for seminar classes and 2 hours are for laboratory classes. The semester has 14 weeks of classes. Seminars are for problems solving from the course topics while laboratory classes are dedicated to incremental development of a software.The laboratory project evolves at the same time with the course: new topics are learnt, new project requirements, both functional and non-functional (design, architecture, quality attributes) are identified for students to implement them.

The main *topics* comprised in APM course syllabus are stated as follows:

- Introduction to Java platform: platform, language syntax, primitive data types, arrays, classes, interfaces, packages, enums, overriding, overloading, exceptions.
- Collections and Generic Types: anonymous classes, polymorphism, casting.
- IO, NIO: binary and character oriented streams, files, channels and buffers.
- Functional programming: lambda expressions, streams.
- GUI: Java FX components, event handling.
- XML: schema, documents.
- GUI (cont.):FXML, CSS. Metaprogramming.
- Concurrency.

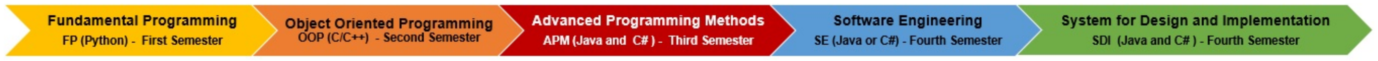

Fundamental Programming
FP (Python) - First Semester
Object Oriented Programming
OOP (C/C++) - Second Semester
Advanced Programming Methods
APM (Java and C#) - Third Semester
Software Engineering
SE (Java or C#) - Fourth Semester
System for Design and Implementation
SDI (Java and C#) - Fourth Semester

Fig. 1. Software Engineering Courses. Timeline.

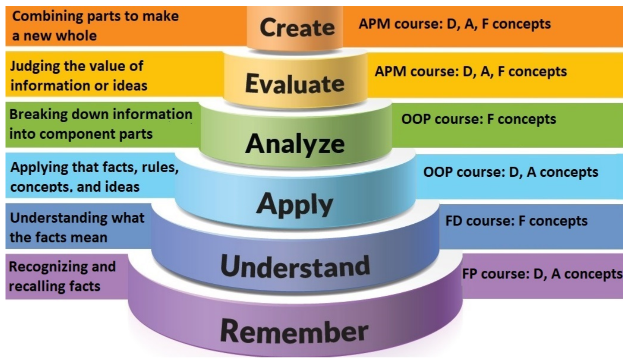

Combining parts to make a new whole
Create
APM course: D, A, F concepts
Judging the value of information or ideas
Evaluate
APM course: D, A, F concepts
Breaking down information into component parts
Analyze
OOP course: F concepts
Applying that facts, rules, concepts, and ideas
Apply
OOP course: D, A concepts
Understanding what the facts mean
Understand
FD course: F concepts
Recognizing and recalling facts
Remember
FP course: D, A concepts

Fig. 2. Cyclic learning methodology. Adapted from [9]

.

- Introduction in C# and .Net.
- Collections in C#.
- IO operations in C#.
- GUI in C#.
- LINQ.

C. Assessment

The design of how the assessment is done considers a wide range of activities comprising various perspectives and different granularity of the competences. The final grade at APM course is an aggregated measure, begin composed of three ingredients, each having a different weight. These ingredients are: i) the students' grades obtained at the project developed during the semester (LabPrj); ii) a written test (Written) during the examination period; ii) a practical test (Practic) during the examination period. This metric (formula) is defined by the following formula:

$$Average = w_1 \cdot LabPrj + w_2 \cdot Practic + w_3 \cdot Written$$

III. Proposed Active Learning Process Design

We recall here one of the goal of this paper. that of proposing an active learning process design at APM course. This has to comprises teaching activities that actively involves students in the real learning process [10]. When learners are active in the classroom, they are engaged in higher-order thinking (analysis, synthesis, evaluation) and in a variety of activities such as reading, discussing, writing, and problem solving [11]. Such classroom activities put the student at the center of the learning process enabling them to improve their critical thinking skills [12].

The proposed learning process design comprises student-centered learning methods that are defined through various activities during the lectures, seminars and laboratories classes. The assignments received by students at laboratories and seminars classes are needed to be accomplished during the semester, in-class or take home activities and they are assessed. During lectures classes small assignments are given to be solved on place.

In what follows we present the active learning methods embedded in the APM course design describing the way they are intertwined at lectures, seminars and laboratory classes.

A. Project Based Learning

Project-Based Learning–PBL is one of the main successful student-centered educational methods broadly used in computer science, information systems and engineering courses [13], [14], [15]. The first assignment for APM course, as part of the learning process, is to develop a project written in Java language and a smaller one in C#. The developed methodology is an iterative one [16], [17]. The project is individually implemented by the students, due to the fact that the APM course preceedes the Software Engineering course (see Figure 1. Even if the APM course incorporates Software Engineering base elements (ranging from: Software Life Cycle, Software Systems Analysis and Design, UML class diagrams, Iterative Software Development methodology, Software testing etc.), those required for team working are missing. There were discussions to evolve this task as being developed in teams, but the APM course syllabus includes also C# programming language and it would be too much.

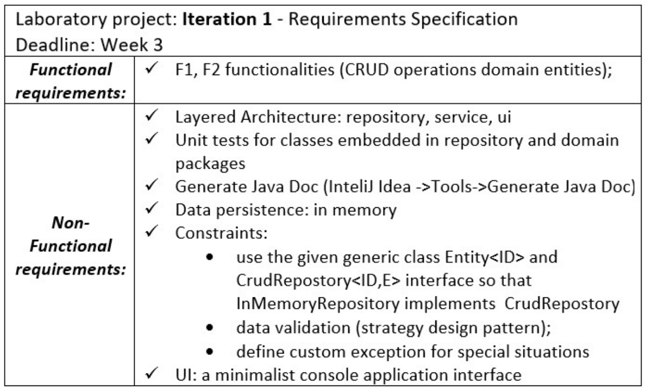
The proposed project is defined by course instructor. Choosing an application domain that is exciting for students and at the same time relevant for industry is not an easy task. Thus, every year students are invited to come with their own ideas, but they are more comfortable to accept a given project. Because of project synchronization with lectures and seminars classes, the course coordinator plans the iterations of the project, during the semester, in order to assure that the knowledge needed by students to implement a given iteration are already acquired at the lectures and seminars classes on that topic.

The students are assessed by lab coordinator for every iteration of the project, receiving also a valuable feedback. They are empowered to bring their own ideas for design and implementation, being encouraged to ask questions and to have debate sessions with lab instructors regarding the design constraints imposed, and also to discuss the solution they found to solve these issues. These types of activities involves active learning methods such as *inquiry-based learning method*, *collaborative based learning*, *peer learning* and also aim to acquire *soft skills* and *to link curriculum with life.*

Each iteration of the project is specified by two types of requirements : functional requirements and non-functional requirements (software design requirements and constraints). In Figure 3 we briefly presents the way of which project iterations are specified:

B. Extended Project - Workshop Based Learning

Another component of the learning process design offers the students the possibility to extend their project developed during the semester with some extra functionalities that overcome the difficulty level required for the course or with some topics (ex. authentication, paging repository, security, client-server) not covered in the course syllabus. It is a challenging task, the students have to improve their project to be as much similar as a real one project in terms of some quality attributes that are established by the course coordinator, like for instance, usability, security, maintainability, reusability, etc. The students that aim to extend their project have the benefit to skip the practical exam. Every student has to prepare a public presentation for the extended project that will be part of the organized workshop at the end of the semester. The event is



Laboratory project: **Iteration 1** - Requirements Specification Deadline: Week 3	
Functional requirements:	✓ F1, F2 functionalities (CRUD operations domain entities);
Non-Functional requirements:	✓ Layered Architecture: repository, service, ui ✓ Unit tests for classes embedded in repository and domain packages ✓ Generate Java Doc (InteliJ Idea ->Tools->Generate Java Doc) ✓ Data persistence: in memory ✓ Constraints: • use the given generic class Entity<ID> and CrudRepostory<ID,E> interface so that InMemoryRepository implements CrudRepostory • data validation (strategy design pattern); • define custom exception for special situations ✓ UI: a minimalist console application interface

Fig. 3. Iteration 1 - Requirements Specification

public, every student enrolled at APM course can be registered to this event if he/she wants.

The reasons that led to this extended project are based on the differences noticed by course instructor regarding the students knowledge: every year there are students that are already familiar with the course topics before they start, and this percent of students is equals to approximately 20. For these students, more difficult tasks are required to increase their motivation for the course. An important aspect here is that, not only these students accomplish the tasks from the extended project. This is also a challenge for every student enrolled at APM course.

C. Quizzes dataset - Inquiry Based Learning

Inquiry-based learning supposes that the learner explores a theme and chooses a topic for research, develops a plan of research and comes to conclusions, although an instructor is usually available to provide help and guidance when needed [2]. The third part of the learning process empowered the students to build their own test for the *written exam*. This is a multiple choice written test sustained in the exam session period, as part of their summative assessment. During the semester, each students has to come with at least one multiple choice question, based on a given course topic, (in Java or C#) to build a database containing quizzes for the written exam. These quizzes emphasise on understanding the course concepts and are discussed with the students at the seminar classes and they represents 10% of the written exam. Only difficult quizzes receive the maximum score assigned. For 3 points out of 10, the written exam contains identically some of the students questions, and for the others 7 points the students quizzes are decorated (modified) by course instructor.

D. Collaborative Based Learning

Collaborative learning is one of active learning methods which can facilitate learner's critical thinking. Peer interactions during collaborative learning can be helpful for the learner's development of critical thinking [2] [18]. A collaborative learning activity is designed at the seminar classes in order to assess the quizzes proposed by students as an assignment in order to build their written exam quizzes data set. The course instructor facilitates collaboration between students in order to find out for every quiz the following aspects:

- What are the concepts identified in the course support slides that are covered by that quiz?
- How many stars can be granted for given quiz? Justify.
- Are the concepts implied in the quiz used in a normal scenario or there is a tricky question?
- What things have you learnt with this quiz or what knowledge have you acquired?
- What could happen id the line of code "x" is replaced with ...?
- Other similar questions to previous ones.

Beside this activity of assessing students quizzes and having debate on this topic, the seminar classes follows a collaborative learning approach for all the course topics presented at lectures classes. The teacher coordinator of seminar proposes a problem, the students have to come with their own solution, or the students can solve the problem on a team of 3 or 4 students. They come at the seminar classes with their laptops and they can present the solution on the projector. Five minutes of discussions and debates regarding a proposed solution take place.

This type of activity is also applied during the lectures, lasting 5-10 minutes, in order to capture students attention. Also a quiz based test on Socrative was applied at APM leactures classes.

E. Research Questions

Having discussed the proposed approach of active learning process design at APM course, the next step is to validate it. In this respect, our goal is to assess what impact has the application of active learning methods on students' outcomes. Three research questions are formulated in what follows.

RQ1: Does active learning methods applied in course design significantly improve student outcomes?

RQ2: What is the students' perception on the use of *cyclic-learning* based approach?

RQ3: What is the students' perception on the use of *active learning* methods at the APM course?

IV. Results analysis

The course changes have been implemented starting from 2017. The current academic year, 2019-2020, represents the fourth iteration of the APM course, and it is still in progress. We omit this semester from the analysis of students grades because the exam session will follow in the next period. However, a self assessment test was applied for students from the current semester regarding their perception of the impact of applying cyclic learning approach at this course. These results are worth to be analyzed. At the same time, a valuable feedback, based on questionnaires, was received from the students in the current course iteration regarding their opinion on the use of active learning methods at a Software Engineering related course. These results are also analyzed.

A. The Impact of Active Learning Methods on Students' Grades at APM Course

An important goal throughout the course evolution by using active learning methods was to assure that the students' success rate in the class improves. Figure 4 shows box plots of the students' grades over three academic years. Analysing this figure it can be noticed that the students' average grades increased. In what follows, we will detail this analysis and we use the statistical methods to investigate if there are significant differences between students' averages of these three academic years.

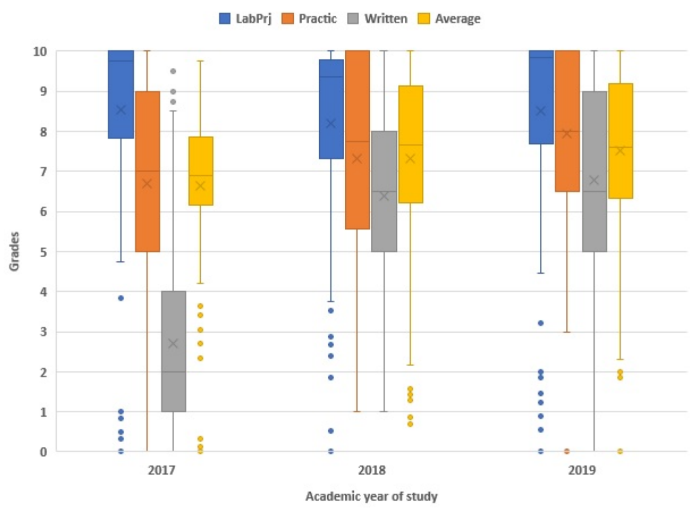


Fig. 4. Students' grades over the three academic years

A transversal and longitudinal analysis was made having into account different weights of the variables that define the ingredients of the final grade at APM subject (see Equation from Section II-C). The variation of these weights throughout the three iterations of the course, is based on course coordinator perception about the previous semester results, about the feedback received by course coordinator from students, in the previous iteration, and on lessons learnt by teaching this course.

1) Changes applied in Course Evolution: Over the three iteration of the course, the ingredients weights of the students' average at APM subject are given in Table I.

Academic year	LabPrj	Practic	Written
2017	0.4	0.4	0.2
2018	0.3	0.4	0.3
2019	0.25	0.25	0.5

TABLE I
FINAL GRADE FORMULA INGREDIENTS FOR APM COURSE.

In the first iteration of the course (2017) the weight of the written test at the final grade was 20%. The reason in establishing this value was based on the fact that the written exam contained some theoretical questions considered to assure students theoretical background acquisition. Turning back now to figure 4, in the first course iteration (2017) it can be noticed the disaster of Written test average value, less then three. The students reasoning was that it would have been needed too much effort to learn the theoretical concepts from the course support slides for a "reward" of 20% percents for the final grade. They argue that it is not important to learn all the theoretical aspects but the most important is to know to apply them to solve problems. The students were partially right, but checking their written test, the course coordinator had noticed severe flaws regarding students knowledge of fundamental concepts. For instance, there were students that encounter difficulties regarding the inheritance concept of object-oriented paradigm at APM course.

Thus, starting from the next year, 2018, the students had to build quizzes and discuss them with their colleagues, and present the quizzes to the lab instructor in order to assure the fact that theoretical background was appropriated by students. And, at the same time, the percentage of the Written variable from the final grade was increased to 30% in the next iteration. At the same time,the students had to obtained at least five grade at Written test in order to pass the exam.

Regarding the changes made from *the second iteration of the course*, changes that were applied started with the third iteration, we can summarize: i) the quizzes made by students in order to reinforce their theoretical background were collected by course coordinator and constituted a comprised quizzes pool that was shared with students and it was part of their written exam; because of this the Written weight of the final grade formula was 50% in the 2019 academic year; ii) the Practical test was sustained during the semester and together with the LabPrj project contributed to the final grade with another 50%, the Written test being the only one sustained in the Session Exam period in 2019; iii) started from the third iteration the lectures have been interactively presented embedding life coding, gamification and short quizzes.

2) Descriptive statistics: T-tests and Pearson Correlation are used to determine the impact of active learning methods on students grades. Firstly, we aim to determine if there was a difference between the APM course iterations over the three academic years regarding the grades obtained by students.

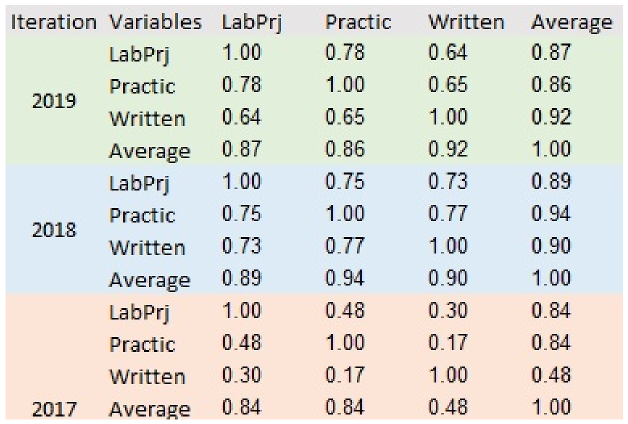
A T-test is applied in this respect. The results of the T-test, described in Table II, indicate that there is a highly *significant difference* between the students' averages of 2017 and 2018 academic years ($p = 0.001$) and also between the students' averages of 2017 and 2019 academic year ($p = 0.001$). *This means that there is a relevant impact on students' averages after introducing quizzes as students assignment during the semester, empowering the students to build their written exam and to learn the theoretical concepts by doing.*

Further analysis of the results obtained by students over the three iteration of the APM course is done using Pearson Correlation. In this respect, we aim to verify if the introducing of quizzes based assessment has positive influence on variables that accounted for the final grade: Practic exam and LabPrj project. Thus, we compute, for every iteration of the course the Pearson coefficient between these variables: LabPrj, Practic, Written, Average. Analyzing the Pearson's correlation matrices described in Figure 5 it can be noticed that in the 2017 academic year the correlation coefficients between these variable are low whereas for the 2018 and 2019 these coefficients are closer to value 1. This means that in the last two iteration of the course these values balanced meaning that there are a very small number of students that obtained a low grade at the written exam and a high score at the practical exam. This is a very important aspect due to the fact that students acquire a solid background knowledge for all the aspects implied in the assessment formula. They have good theoretical knowledge, this means a deep understanding of the course topics, and at the same time they have the ability to apply all these concepts in a practical context and the development skills needed to solve a real life problem.

Response for RQ1: The analysis described above and the findings presented reveals that there is a statistically significant impact on students' grades by applying active learning methods.

T-test description	Mean Difference	t	N	p-values
2017 - 2018 Grades	-0.67	-4.005	200	**0.001**
2017 - 2019 Grades	-0.76	-4.73	200	**0.001**
2018 - 2019 Grades	-0.1	-0.62	200	0.26

TABLE II
T-TEST: STUDENTS' RESULTS OVER THREE ACADEMIC YEARS



Iteration	Variables	LabPrj	Practic	Written	Average
2019	LabPrj	1.00	0.78	0.64	0.87
	Practic	0.78	1.00	0.65	0.86
	Written	0.64	0.65	1.00	0.92
	Average	0.87	0.86	0.92	1.00
2018	LabPrj	1.00	0.75	0.73	0.89
	Practic	0.75	1.00	0.77	0.94
	Written	0.73	0.77	1.00	0.90
	Average	0.89	0.94	0.90	1.00
	LabPrj	1.00	0.48	0.30	0.84
	Practic	0.48	1.00	0.17	0.84
	Written	0.30	0.17	1.00	0.48
2017	Average	0.84	0.84	0.48	1.00

Fig. 5. Pearson's correlation between final grade ingredients variables over the three academic years.

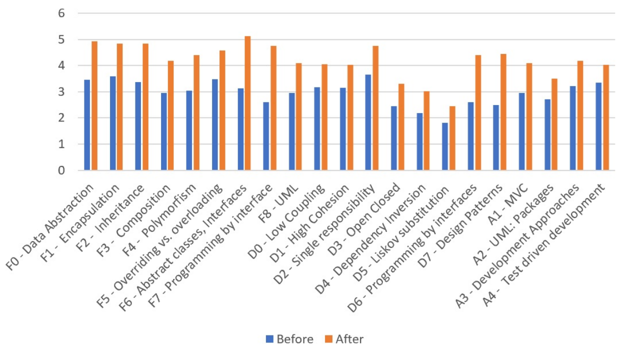


Fig. 6. Cyclic learning Self Assessment: Before and After APM course

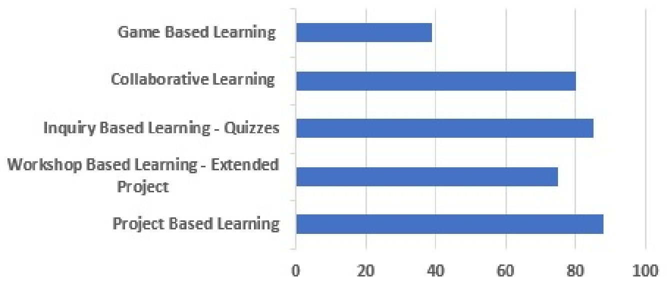


Fig. 7. The effectiveness of ALM on students' perception

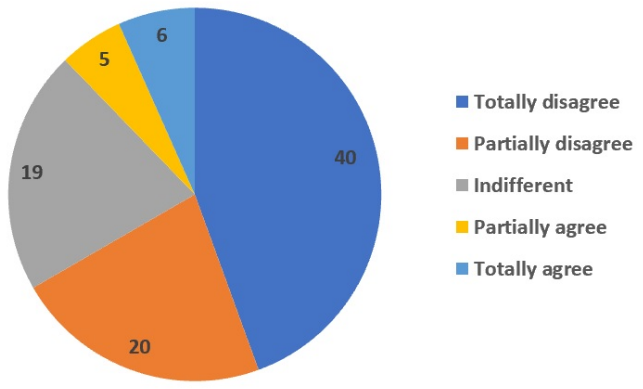


Fig. 8. Are traditional expository lectures sufficient for learning SE?

B. Students' feedback

At the end of the last semester, two questionnaires were applied to students in order to find out their perception on the use of active learning at the APM course. One of them was about the cyclic learning approach and the other one made reference to students' learning experience at APM course via active learning methods. In what follows we analyze the results of these questionnaires.

1) Cyclic learning self assessment: A study of students' self assessment regarding their perception of the efficiency of using cyclic-learning approach in teaching APM course was investigated at the last semester of the fourth course iteration, the results being presented in Figure 6.

Response for RQ2: The results showed reveals that there is a significant difference of students' levels of learning regarding the Bloom's Taxonomy between the beginning and the end of the semester, for all the concepts implied in the assessment.

2) Students' Perception on the use of Active Learning Methods: This section aims to respond to the third research question RQ3.

A questionnaire regarding the student's perception on the use of applying active learning methods at APM course was provided to the students. It contains several questions regarding the learning experience the students had at APM course.

Due to space restriction, we present here only a summary of this questionnaire. Analysing students response of the question "What are the most effective active learning methods that facilitating your knowledge acquisition?", Figure 7 reveals that Project Based Learning and Quizzes Based Learning are the most effective methods and in previous section we have also drawn the same conclusion that these two methods had the biggest impact on students grades. At the same time, students had to evaluate the statement: "I believe traditional expository lectures, with punctual evaluation methods (exams and specific assignments), are sufficient for learning SE", and their results shown in Figure 8 reveals that almost all students doubt that traditional methods could be enough to teach a Software Engineering related course.

Response for RQ3: The above results reveal applying active learning methods are effective in facilitating knowledge acquisition and that traditional methods could not be enough to teach software engineering concepts.

V. Related work

Several studies investigate how the active learning methods applied in Software Engineering related courses provide opportunities to learn about software development more efficiently. In what follows we lay out several such studies, each with different perspectives and tackled challenges.

The [19] study discusses issues and practices for improving the team project from the perspective of an introductory Software Engineering (SE) course. Rajlich [20] investigate challenges and constraints that they faced while trying to teach skills needed by developers on software projects, teaching practices of software change.

Other studies explore the evolution of courses related to SE, investigating the effects of changes on the students' grades and performed activities. For example, in paper [21] the authors present the evolution with the changes done over six semesters, at the end extracting a set of lessons regarding the class evolution. The [22] study conducted research regarding mistakes in UML diagrams by investigating student projects in a SE course. The authors introduced a catalogue of designing mistakes that increased the effectiveness of both teaching and learning of UML diagrams.The paper [23] evaluated the students' perception on the adoption of Project-Based Learning in SE education. There is a need of providing students with sufficient practical experiences for the development of competences expected for SE professional. The teaching-learning approach in [24] consisted in four main elements (inverted or flipped classroom, studio-based learning, real-client projects and deployment, large team and peer evaluation) applied to an undergraduate level course on SE.

In relation to existing teaching/learning strategies for Software engineering courses, our approach aims to achieve goals similar to [23] by also using Project-Based Learning. We also provide an analysis of how the discipline was taught during several years as in [21]. Our work is also similar to [24] on the grounds that we also considered various active learning methods.

VI. Conclusions and Future work

The new theory of education suggests that learner should be in the center of the learning process and the instructors playing an advising and facilitating role. A shift in education theory to a more student-centered approach using active learning is recommended.

The paper presents our experience to apply active learning methods at a Software Engineering related course, over three iterations. Changes were applied in order to overcome the limitations of the previous iterations. An empirical study was conducted to investigate the impact of active learning method on students' grades. Also, the study investigated the students perception on their knowledge acquired by adopting an active learning conduit. Statistical analysis was used to proof the relevance of changes appeared by applying an active learning approach. It is worth mentioning that the improvements of grades in the course evolution could have been influenced by other factors. Some of them that we can think of are: the class of 2018 might simply have been better than the class of 2017, the teachers may have learnt from the previous year, emphasising more strongly for students that lack of preparation lead to failing the course, and the weighting changed from one year to the next. Future work will also investigate these factors.

Our experience allowed us to distill a set of lessons learnt and to use them to further improve our course in the future. In this respect, we aim to register the difficulties encountered by students every year and use them in appropriate context as support for the students in the next iteration. At the same time an e-learning platform that can support us in this process in order to automate some trivial tasks, like for instance quizzes management, is one of the main extends of this project in near future.

References

[1] J. Rich, "An experimental study of differences in study habits and long-term retention rates between take-home and in-class examinations," *International Journal of University Teaching and Faculty Development*, vol. 2, no. 2, pp. 121–129, 2011.

[2] J. J. Rich, A. Colon, D. Mines, and K. Jivers, "Creating learner-centered assessment strategies for promoting greater student retention and class participation," *Frontiers in Psychology*, vol. 5, no. 1, pp. 1–3, 2014.

[3] S. Freeman, S. L. Eddy, M. McDonough, M. K. Smith, N. Okoroafor, H. Jordt, and M. P. Wenderoth, "Active learning increases student performance in science, engineering, and mathematics." pp. 50 – 62, 2014.

[4] *EDR and Design Thinking - Enabling Creative Pedagogies.* Vancouver, BC, Canada: Association for the Advancement of Computing in Education, 2016.

[5] T. Brown, *Harvard Business Review*, vol. 86, no. 6, pp. 84–92, 2008.

[6] C. Serban and H. Pop, "Software Quality Assessment Using a Fuzzy Clustering Approach," *Studia Universitas Babes-Bolyai, Seria Informatica*, vol. LIII, no. 2, pp. 27–38, 2008.

[7] B. S. Bloom, M. Engelhart, E. J. Furst, W. H. Hill, and D. R. Krathwohl, *Taxonomy of educational objectives: The classification of educational goals.* New York: David McKay Company., 1956.

[8] L. W. Anderson and D. Krathwohl, *A taxonomy for learning, teaching, and assessing: A revision of Bloom's Taxonomy of Educational Objectives.* New York: Longman, 2001.

[9] L. University, "Library teaching and learning," Accessed 2019. [Online]. Available: https://ltl.lincoln.ac.nz/teaching/preparing-to-teach/

[10] M. Prince, "Does active learning work? a review of research," *Journal of Engineering Education*, pp. 223–231, 2013.

[11] C. Bonwell and J. A. Eison, *Active Learning: Creating Excitement in the Classroom.* George Washington Univ. Washington DC, 1991.

[12] Y. F. Chan, G. K. Sidhu, N. Suthagar, L. Lee, and B. Yap, "Relationship of inquiry-based instruction on active learning in higher education," *Pertanika Journal of Social Science and Humanities*, pp. 55–72, 2016.

[13] M. Jazayeri, "Combining mastery learning with project-based learning in a first programming course: An experience report," in *Proceedings of the IEEE/ACM 37th IEEE International Conference on Software Engineering (ICSE)*, 2015, pp. 315–318.

[14] J. A. Macias, "Enhancing project-based learning in software engineering lab teaching through an e-portfolio approach," *IEEE Transactions on Education*, pp. 502–507, 2012.

[15] M. Marques, S. F. Ochoa, M. C. Bastarrica, and F. J. Gutierrez, "Enhancing the student learning experience in software engineering project courses." *IEEE Transactions on Education*, pp. 63–73, 2018.

[16] R. G. Fichman and S. A. Moses, "An incremental process for software implementation," *Sloan Management Review*, pp. 39–52, 1999.

[17] K. Beck and M. Beedle, "Agile alliance. manifesto for agile software development, 2001," Accessed 2020. [Online]. Available: http://agilemanifesto.org/

[18] K. Kim, P. Sharma, S. Land, and K. Furlong, *Effects of active learning on enhancing student critical thinking in an undergraduate general science course.*, 2013, pp. 223–235.

[19] D. C. C. Peixoto, V. A. Batista, R. F. Resende, and C. I. P. da Silva, "Learning from students' mistakes in software engineering courses," *2010 IEEE Frontiers in Education Conference (FIE)*, pp. 1–6, 2010.

[20] V. Rajlich, "Teaching developer skills in the first software engineering course," in *Proceedings of the 2013 International Conference on Software Engineering*, 2013, p. 1109–1116.

[21] D. Delgado, A. Velasco, J. Aponte, and A. Marcus, "Evolving a project-based software engineering course: A case study," in *2017 IEEE 30th Conference on Software Engineering Education and Training (CSEE T)*, Nov 2017, pp. 77–86.

[22] S. Chren, B. Buhnova, M. Macak, L. Daubner, and B. Rossi, "Mistakes in uml diagrams: Analysis of student projects in a software engineering course," in *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, May 2019, pp. 100–109.

[23] M. Souza, R. Moreira, and E. Figueiredo, "Students perception on the use of project-based learning in software engineering education," in *Proceedings of the XXXIII Brazilian Symposium on Software Engineering*, 2019, p. 537–546.

[24] A. Sureka, M. Gupta, D. Sarkar, and V. Chaudhary, "A case-study on teaching undergraduate-level software engineering course using inverted-classroom, large-group, real-client and studio-based instruction model," *CoRR*, vol. abs/1309.0714, 2013.