Challenge Based Learning applied to Mobile Software Development Teaching

Fabio Vinicius Binder
PPGIa
PUCPR
Curitiba, Brazil
fabio.binder@pucpr.br

Mark Nichols
Challenge Institute
USA
mark@challengeinstitute.org

Sheila Reinehr
PPGIa
PUCPR
Curitiba, Brazil
sheila.reinehr@pucpr.br

Andreia Malucelli
PPGIa
PUCPR
Curitiba, Brazil
malu@ppgia.pucpr.br

***Abstract*—The growth of the mobile device market has generated a demand for specific applications and the consequent need for labor training to develop them. Such demand has as a direct consequence, a growing need for training of application developers. This work presents the use of the active learning methodology Challenge Based Learning (CBL) for the teaching of software development for mobile devices. The use of CBL for little more than a year in four classes of a free course with 110 students is presented. Some difficulties appeared and were solved with proposed actions. The results are obtained from objective and subjective evaluations and analyzed through thematic networks. Among the positive results CBL helps in understanding the problems to be solved and in its solution and has motivated students more than other methods. CBL showed to be an interesting active learning methodology for teaching Mobile Software Development and promising to be applied in other areas.**

Keywords— Challenge Based Learning, Programming Teaching, Active Learning, Software Development, Mobile Devices

I. Introduction

The mobile device market is growing at an accelerated rate, creating a strong demand for personal and business applications [8]. Mobile development is a relatively new process, typically having a lower lifecycle than desktop/laptop or web-based software. Its distribution is usually done by pre-installing it in some device or by downloading it from an online store [5].

Due to the growth in smartphone sales [16], applications are becoming an important part of the portfolio of enterprise and mission critical systems [8]. This demand has a direct consequence: the growing need for solid training of mobile application developers (Apps), which offers not only basic programming education, but also software development and production for publishing.

Several traditional teaching methodologies are currently being used to teach software development, however, some features of the Challenge Based Learning (CBL), described in this paper, indicate that it is more suitable for this type of training. Thus, in order to minimize the difficulties encountered in this type of environment and with traditional teaching and learning methods, and contribute to the solid training of developers, this research presents the application of the active learning methodology CBL, for the teaching of software development for mobile devices.

The motivation to use non-traditional methodologies comes from the observation and experimentation of student behavior in traditional classes. There is a relationship between the type of stimulus received and the engagement of students during a lesson. This is evidenced by classroom experience where students usually prefer to engage in practical activities than listen [13]. Non-traditional teaching methodologies have been used for several years in programming courses with mixed results, mainly the Problem Based Learning (PBL) [7], the Flipped Classroom (FC) [13] or both simultaneously [4].

The project had the following premises: (i) the activities should be oriented by teachers of the area of computation, with skills and competences for the development of applications of this nature and developed by students of technical, undergraduate, specialization, master's and doctorate courses; (ii) students should develop innovative applications; and (iii) all applications developed should be the intellectual property of the students themselves.

The environment in which the project was inserted is considered complex due to the large number of people and risk factors such as: (i) use of an unknown teaching method by teachers and without any literature regarding its application in teaching software development; (ii) more than 100 students involved; (iii) heterogeneity of students in technical courses, undergraduate, specialization, master's and doctorate; 10 teachers, of which 5 were full-time instructors; just over 30 external project proponents; management of 4 support teams in the areas of usability, interface, illustration, 3D modeling, animation, sound, soundtracks, innovation and entrepreneurship, which provided resources and services; 800 hours of course; development of innovative apps with professional appeal, that is, applications ready to be published.

No similar cases involving numbers of this magnitude were found in the literature, but there were studies with similar objectives involving smaller scope and fewer resources. One of these studies [14], which had positive results, offered a course

DOI 10.1109/CSEET.2017.19

for the development of apps, but it was only six weeks of course, there was the prerequisite programming knowledge in the Java language, a traditional teaching method was applied (lectures and practical classes), there was no support of other areas and the project to be developed was the same to all teams.

This paper is organized as follows: Section 2 introduces the CBL method. In Section 3 the research method, which is based on the action research, is presented. Section 4 details the action research. Section 5 presents the results and discussion and Section 6 concludes the paper.

II. CBL Method

CBL's active learning method emerged as a byproduct of a large-scale collaborative initiative that began in 2008, called the Apple Classroom of Tomorrow - Today, whose main goal was to identify the key principles for designing educational environments in the 21st century [1]. It can be defined as a motivational, collaborative and multidisciplinary approach that encourages the use of common technologies for knowledge acquisition and real-world problem solving.

A. CBL Overview

The guiding principles of the method are [10]: an environment for investigative reflection on teaching and learning; flexible framework with multiple entry points; non-proprietary scalable model; focus on global challenges with local solutions; authentic connection between academic disciplines and real-world experience; a framework for developing 21st century skills; a process that places the student as responsible for his / her learning; it requires students to develop and deploy solutions in a real-world environment.

Fig. 1. CBL Phases [9].

The CBL method has a set of phases: Engage, Investigate and Act (Fig. 1), whose main objective is to aid students to solve a global critical problem through local actions. Some activities may be carried out during all phases: evaluation, documentation and reflection actions. Within the phases there are steps that may be executed sequentially, but it is possible to start the process in any of the first four steps. The steps are [9]: Big idea, Essential Question, Challenges, Guiding Questions, Guiding Activities and Resources, Analysis, Solution Development, Implementation and Evaluation. These phases may be constantly described, monitored and analyzed based on three optional aspects: Informative Assessment, Documentation and Publishing, Reflection and Dialogue.

B. CBL Research

CBL has recently been used with relative success to teach various topics: composition of materials for aircraft manufacturing [17], english as a mother tongue [18], biochemistry [21], programming [19] and nursing [20].

The flexible and open character of the CBL generates a need for integration with other methods and techniques. This integration can be done in any of its phases and, even though it is not mandatory, is encouraged by the creators of the method [9]. Successful results were reported in the integration with Design Thinking [18] to facilitate the generation of ideas during the first phases and with SCRUM [19] to provide a form of project management during the Stage of the solution.

In addition to formal methods, punctual changes are possible, such as rearranging its initial phases and targeting the study format [21]. In the present study a number of specific changes were made, such as extensive research of similar works, specific subject workshops, reinforcement classes, among others.

C. CBL x Other Active Learning Methods

One of the main differences between the CBL and other active methodologies is the objective of the study: it focuses on the acquisition of certain skills by developing a solution to a real problem identified and proposed by the student. The contents that must be studied arise from the needs of the proposed challenge: content that will not be useful for the challenge will not be studied.

In other methodologies, the teacher usually proposes the content (FC) [6] or problems associated with a specific content (Problem Based Learning - PBL and Project Based Learning - PjBL) [12] followed by practical activities that involve such content, that is, while in CBL the practical activity (challenge) defines the content that must be studied, in other methodologies the topics of study are defined first and the practical activities are related to this content. Considering that in courses that use CBL, students and teachers design the challenges and ideally the learning experience, it is common that different students study different topics to solve their challenges.

Table I presents a comparison between the characteristics of the CBL and other methods (PBL, PjBL and FC). The criterion for choosing the features of the table was based on those most relevant to the teaching of software development according to the observations made by the researcher in the case of CBL and in the analysis of the studies of active methodologies applied in programming courses for the other methods.

TABLE I. COMPARISON BETWEEN CBL X OTHER ACTIVE METHODS

Features/Methods	CBL	PBL	PjBL	FC
Previous Study				x
Predefined content		x	x	x
Predefined Challenge		x	x	x
Self paced learning	x	x	x	x
Collaborative Learning	x	x	x	x
Autonomy	x	x	x	x
Critical Thinking	x	x	x	x
Problem Solving	x	x	x	x
Student Feedback	x	x	x	x
Evaluation Review	x	x	x	x
Challenges defined by students and teachers	x			
Specific Environment	x			
Solution implementation	x			
Frequent self reflection	x			

Of all the features identified, seven of them are common to all methods and represent a reaction to what is usually criticized in traditional teaching methods: autonomy, collaboration, critical thinking, problem solving, student's pace of learning, feedback and revision.

It is interesting to note that, with the exception of the item related to student's pace of learning, the others are some of the characteristics necessary for the work of a software developer. It is possible to observe that the teaching methods are aligned with several skills that software developers must have in order to accomplish their tasks, however, some aspects are unique of the CBL. With regard to software development teaching none of these characteristics would be essential for student training, but all have advantages in terms of student motivation and experience.

The fact that the student can choose its subject and challenge is an additional motivation factor, as it encourages the freedom and, especially personal projects that many students have when starting a course.

The specific environment is composed of flexible classroom, laboratories with islands of teamwork, spaces for individual reflection and projection of computer screens on high-quality televisions. It is a differentiated, pleasant, high quality and highly motivating environment.

Frequent self-reflection enables the software developer to find better ways to solve certain problems. The analysis of finished projects is a common practice in some development niches, notably in the production of digital games, as can be easily identified by reading the major magazines in the area and the presentation sessions of major game conferences such as Game Developers Conference (GDA) in the USA.

In addition, the need to implement and publish the developed solution forces the student to worry about aspects related to the production of professional software that are not usually taken into consideration in common courses. Among these aspects, it is possible to mention: software quality, usability and interface and maintenance after the publication of the software.

This need of solution implementation is presented by CBL in a free and abstract way, without detailing for specific situations. In a programming course, this deployment is quite challenging, involving several aspects that beginners may have difficulty assimilating.

III. RESEARCH METHOD

The scenario of the research was a project of technological innovation that aimed at the production of applications for mobile devices within a university environment. Fig. 2 presents the approach defined for its organization, composed of three phases, described below.

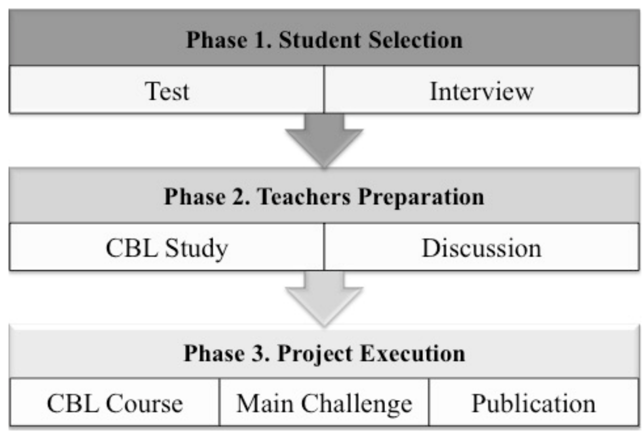

Fig. 2. Research Phases

A. Students Selection

At this stage tests and interviews were applied. The initial premise was that students should have good basic programming skills and/or logical reasoning. So the test was divided into programming and logical reasoning sections. The content was balanced to prevent more advanced students from having any advantage, eliminating possible talents not yet so experienced. The test was composed by logical problems and programming questions. To eliminate guessing there was no objective questions, like multiple choice or true/false. Partial grades were awarded in order to better evaluate candidates thus avoiding the elimination of potential good students. The total number of candidates registered in this stage was 398 and the top 150 were selected according to the overall grade obtained in the test.

The interview aimed to know a little more about the profile of the student and was held in person with all 150 students. We selected 110 students for the first class. The students were from several different levels of courses, including technical education, graduation, specialization, masters and doctorate.

Most of these students were from courses with some programming education and only three students (2.7%) were from courses not related in any way to computer programming.

B. Teachers' Preparation

The goal was to train teachers in CBL. The preparation was composed of two parts: in-depth study of the method and discussion of its application for the teaching of software development. Teachers conducted a previous CBL study [9] then they took a two-day course. It should be noted that CBL has never been used in a continuous and relevant way in teaching software development.

The second phase was composed of discussions among teachers to identify best practices for applying the CBL. Although it was possible to use the CBL without any kind of lecture given by teachers, it was decided to continue with lectures for two reasons: the importance of the teacher's role in promoting interest in students, even when active learning methods are used [22], and the need of differentiated forms of teaching for the student to attain complex levels of thought and commitment [23].

C. Project Execution

At the beginning of the implementation, some challenges were identified, related to the complexity of the environment as students' difficulties with active methodologies, notably the CBL; lack of specific technical skills such as development for mobile devices; design and execution of complex graphical interfaces; use of sound resources, illustration, animation and 3D modeling; difficulty of students to identify unpublished applications and relevant problems; absence of prior experience with software development processes.

These challenges were used to define a plan for the execution of the action-research [3] as shown in Fig. 3. The action research, described in the next section, started when students had already been selected, teachers had already been trained and initial difficulties had already been mapped.

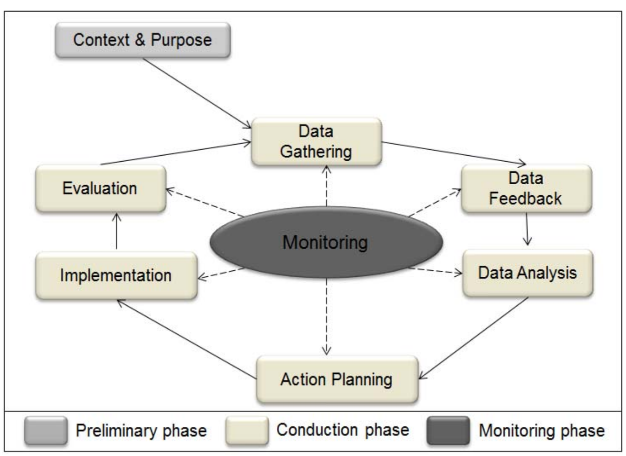

Fig. 3. Action-Research lifecycle, according to [3].

IV. Action Research

The **Action Planning** resulted in a need to change the project execution activity, which had originally been proposed as being composed by CBL course, Main Challenge and Publication (Fig. 2 – Phase 3). The change was motivated by students' difficulties to move directly from the course to the Main Challenge. It was observed the difficulty of the students in developing programs that demanded nontrivial solutions.

The resulting change (Fig. 4) was the division of the Project Execution activity into Initial Challenge, Mini Challenges, Main Challenge, and a transversal Support task. The publication of the developed product was incorporated into the Main Challenge activity, because it was not a separate activity. The Support task represents all the activities necessary for the production of software that do not involve programming: interface and usability, 2D and 3D art, animation, sound and mentoring for the creation of startups.

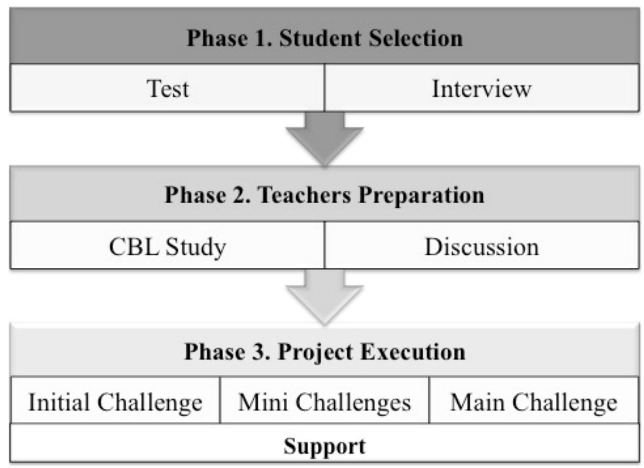

Fig. 4. Post-Adjustment Phases

During the **Implementation** phase of the action research it was observed that several improvements were necessary for the teaching of software development. Some of them were made during the Project Execution activity itself, such as the integration of a software process model with the CBL. In this step, CBL was applied in three different ways, and evaluations were made to improve the method in the next application.

- **Initial Challenge**: CBL's first application consisted of a quick lesson on features of relevant and successful Apps and an explanation of the CBL, followed by the Initial Challenge. This challenge was included because it offers significant choices to the students about what and how to study, this being one of the ways to increase engagement and interest in the classroom [15]. It consisted in defining an idea of an application and creating it's concept, generating a presentation video of how the application would work if it was already developed. The students met in groups of three or four participants and had three hours to complete the challenge. At the end of the activity each team presented their work to all students. Some of the works presented by the groups were: how to find a free parking spaces, identify nearby hospitals according to the needs of the user, an augmented reality game, etc. The students then recorded a two-minute self-reflection audio which consisted of completing the following phrases: I learned ...; I wish I had learned ...; I still need to learn ...

- **Mini Challenges**: students should develop a relevant and ready-to-publish App in two weeks. The students chose their teams composed of two (2) to four (4) students, which resulted in 39 teams. At the end of the two weeks 12 projects out of 39 were finalized. Out of these 12 projects only two (2) were published in the Mobile Application Store and used by the target audience: a puzzle game and an app to help the environmental police to reach difficult to find locations. Individual reflections showed three main problems: short time, inexperience in the development of a complete product (for students who did not have previous experience) and inadequacy of CBL for software development (for students who already had experience). This result was the basis for the creation of workshops (classes of up to 15 hours for specific topics) and integration of the CBL with a software development process. The workshops covered areas that the students couldn't learn by themselves due mainly to time constraints or lack of the specific basic knowledge of that area: version control, project management, game design, backend software development, web services programming and notions of augmented reality.

- **Main Challenge**: Due to the need to publish an App at the end of the course, it was decided that students should actively participate in a software development process. Since many students were newcomers to the programming area and had never developed software professionally, a hybrid method was chosen that integrate traditional features and techniques like waterfall and agile, sometimes called "AgileFall". The objectives of this approach were to get students to establish a feasible project scope for the time and resources available; give students visibility of the progress and possible delays of their projects; detect delays and student performance problems as quickly as possible; provide a view of the progress of course projects from the management point of view, facilitating course decision making and course corrections in the current and subsequent years. This method was used in the main challenge in all 60 proposed projects. At the end of the course 54 projects were finalized and presented, of which 41 were approved for publication. The students delivered the documentation, fonts and Apps installed on their own devices, in addition to a final reflection video with a free theme. Some examples of finished apps were: a game to help deaf children to read, a fitness helper, a Go player using machine learning, a speed reader that identified the most relevant phrases of a text, a hotel reservation app target to people in a hurry, a framework to turn smartphones into game controls, an educational tour into space exploration and a neuronavigtor.

V. Results and Discussion

Due to the practical nature of the course and also to the evidence that the use of active methodologies affect more directly the students' behavior and attitudes than their performance in standardized tests [11], it was decided not to make a traditional assessment. Thus, the evaluation was carried out through questionnaires applied to students and semi-structured interviews with the teachers involved in the project and other teachers who were applying the CBL in similar environments.

A. Students' Evaluation

Of the 110 students originally selected, 84 finished the course and 78 of them answered the questionnaire. The class consisted of 57.7% of novice students, those with less than one (1) year programming experience and 15.4% veterans with more than three (3) years of programming experience. Table II provides a summary of the survey. Positive answers vary between the questions but were usually strongly agree, agree, totally agree or partially agree.

TABLE II. Survey Summary

Questions	Positive Answers
How many times did you use the method? (>3)	71,8%
Has the method helped you understand the challenge?	74,4%
Has the method helped you to work out a solution?	70,5%
Has the method helped you find an innovative solution?	48,7%
Degree of motivation of the method?	70,5%
Do you want to use the method in other situations?	76,9%
Has the guiding questions, activities, and resources helped you develop your Apps?	82,1%

Students were required to use CBL only three times during the course. On its own initiative, 71.8% of the students used it more often (four or more times), indicating that there was an interest on the part of the majority in applying the method. This result presents the **motivational factor**, where 70.5% of the students felt motivated to solve the problem using CBL, answering that they strongly agree or agree with the question "Has the method helped you to work out a solution?".

In its initial phases, the CBL has as one of the objectives to **aid in the understanding of the problem or challenge**. Most students (74.4%) chose the options "strongly agree" or "agree", indicating that the method helped them in this regard. In addition, 39.7% of students chose the partially or totally Agree options to represent the degree to which the method helped them solve the problem.

The **effectiveness of the method in constructing a solution** was considered positive (partially or totally agree) for 70.5% of the students, and 82.1% considered that the guiding questions, activities and resources helped in the development of the App. This result was expected, since most students had no prior experience in software development and naturally would have difficulty developing an App without any sort of systematized help.

Still regarding the development of a solution, 48.7% of the students considered that the method helped (partially or totally agree) to **find an innovative solution**. This result is in line with the opinion of the evaluators' bank, which ranked half of the projects presented as innovative and with great market potential or relevance.

Students' acceptance of the method can be considered positive, because 76.9% answered "certainly" or "probably" that they may use it in other situations and 87.2% believe that it can contribute to the teaching of content other than programming and development of software.

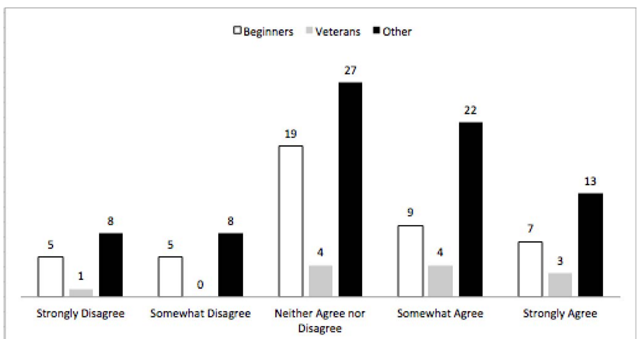

Fig. 5. Effectiveness of CBL on Improving Programming Skills.

Other positive results relate to the **improvement in programming skills** (Fig. 5) and to the **motivational factor** of using the method. 58.3% of students with programming experience (more than 3 years) considered that the method contributed to the improvement of programming skills while only 8.3% responded negatively. Taking into account only novice students (less than a year of programming experience), 35.6% agreed Partially or Totally with the question and 22.2% disagreed Partially or Totally.

In the space for **free comments**, 26.9% of the students wrote some kind of consideration about the CBL. What stood out the most was the need to integrate it into a method of software development and the fact that CBL helps in understanding the problems to be solved and in its solution.

B. Teachers' Evaluation

Interviews were conducted with 23 teachers from eight Brazilian universities where similar iOS courses using CBL were taking place. The interviews were recorded, transcribed, reviewed and analyzed quantitatively and qualitatively.

The qualitative analysis was done using thematic networks [2] with the aim of systematizing and identifying common elements. The profile of teachers was based on the time of experience with software development, teaching software development and teaching in general; use of traditional teaching methods; and knowledge of software development methods. Teachers were chosen who had worked with CBL for at least six months and had applied it at least twice in programming courses.

The teachers involved had extensive experience in software development (mean age of 14 years, standard deviation of 7.7, median 15) and a reasonable teaching experience, both programming, with a mean of 7.1 years (standard deviation 6 and median 5), and in general education, with a mean of 9 years (standard deviation 6.3 and median 8). The traditional teaching techniques (previous to the use of CBL) used in the classroom by these teachers were lectures and practices with 100% positive answers and guided tutorial with 50% positive responses. Guided tutorial was defined as an activity in which a programming problem is proposed and the teacher solves it together with the students, programming it directly in the appropriate tool and showing the entire development and thinking process.

The professional knowledge of software development methods was thus divided into 56.5% of teachers with some experience in traditional methods, 47.8% in iterative and incremental methods, and 73.9% in agile methods. This question allowed multiple answers.

The results of the analysis corroborated the general opinion of students, because 95.6% of teachers agreed in whole or in part to the statement: "The CBL has motivated students more than other methods that you know." Those who agreed partially stated that there was a greater motivation, but did not know whether this motivation was due to the CBL or the novelty aspect of using a different teaching methodology. Possibly or certainly, these options were chosen by 95.6% of the teachers when asked if they will use the CBL for other programming classes, while 91.3% intend to use the method to teach other contents not related to programming.

The use of CBL for teaching software development was addressed in the question "Is the quality of software developed by students using CBL better than the quality of software developed using other methods that you have experience?". In this question we explained to the teachers that the term quality refers to all of the following characteristics: software ready for publication, relevancy, no critical errors and no usability problems. Unlike the other issues, the result was not significantly positive. 65.2% of the teachers totally or partially agreed, and the others chose the "not agree or disagree" option. The justification of the teachers was that they had not worked enough with other methods to support a positive opinion.

The interview consisted mainly of multiple choice questions, however, interviewees could make additional comments whenever they wished. The subjective part of the interview is composed by these comments plus the open questions that dealt with the changes were made in each stage of the CBL and also the general opinion about the CBL. The analysis of thematic networks generated the main theme "Proposed Changes", composed of five sub-themes: positive features, negative features; inconclusive features; suggestions of other methods and improvement suggestions. The relationships between these themes are presented in the Fig. 6 Thematic Network of positive, negative and inconclusive features. and Fig. 7 Thematic Network of Improvement Suggestions and Other Methods.

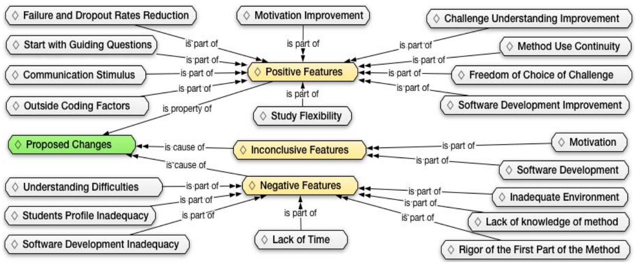

Fig. 6. Thematic Network of positive, negative and inconclusive features.

As can be observed in Fig. 6, the application of CBL had several positive aspects, being the most cited: intention to continue using the method in other teaching situations; improving students' motivation to study and carry out proposed activities; encouraging student autonomy; and improving students' understanding of the problems to be solved. An interesting finding of the teachers was that the flexibility of studying proposed by the method (not having a pre-defined syllabus) promoted the development of some skills in students, such as autonomy.

For the execution of the project, students have the freedom to choose the challenge, starting point of the method, and to use different forms of learning (courses, books, monitoring, internet searches) to acquire the necessary knowledge to solve the problem. There is also a need to worry about other factors that go beyond coding, such as usability, economic feasibility, etc. Another positive point was the incentive to communicate, because the method is eminently collaborative. Students seek out experienced teachers and students to assist in problem solving throughout the project's development. According to one of the interviewees, these aspects contributed to the reduction of both course evasion and failing.

Although the method was considered by all respondents to be successful, some negative points were identified, the most cited being the inadequacy of the method for certain student profiles. It was perceived by the teachers that talked about profile inadequacy that the students' inadequate profile has in common three characteristics: lack of commitment, lack of theoretical and technical base, as well as resistance to autonomy. This same perception was identified by the researcher in the reflections of the students and in the observation of their behavior during the execution of the proposed activities.

Another problem concerns the teaching environment. The CBL requires an environment with certain essential characteristics for proper application, like a good Internet connection, individual and collaborative spaces and flexibility to change tables and chairs. The lack of this environment makes it difficult to use. It was also pointed out the difficulty of understanding the method on the part of the students and the lack of familiarity of the teachers with the application of the method.

The CBL adequacy aspect for software development was inconclusive in the interviews, since negative, positive and neutral opinions appeared. The method was not specifically created for teaching software development and was considered as a general purpose educational framework, so it was somewhat unexpected that 13% of respondents reported that the method is in itself adequate for teaching software development. Based on the analysis of the profile of the interviewees, it was possible to identify that 33% of the teachers who opined in this way, considered the CBL to be suitable for development only in the requirements survey stage, which is perfectly natural since the goal of one of its phases is precisely to improve the understanding of the problem. The remaining 66% were not from the computing area or similar, thus having a different view on software development.

Fig. 7 presents another dimension captured in the interviews and concerns suggestions for improvements and other similar methods. Although the number of citations of these improvements was relatively small (13%) and its impact on CBL is restricted mainly to one-off changes, this shows that opportunities and needs for changes have been identified.

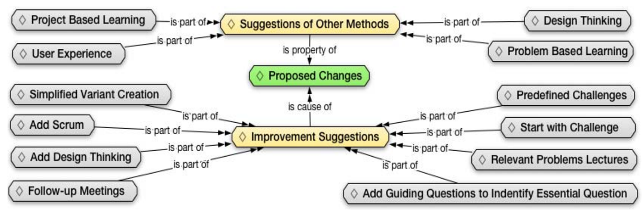

Fig. 7. Thematic Network of Improvement Suggestions and Other Methods.

The main suggestions can be grouped into two categories: timely, for those where the change occurred only in one phase of the CBL and global, for changes that span more than one phase. The most cited points were the use of guiding questions to define the essential question, thus offering a predefined course for the student; and start the application of the method in the challenge phase, usually when the student already has the idea of what will do, thus eliminating the first two phases, big idea and essential question.

The global ones were suggested in order to enrich the CBL in the solution stage. In the suggestions of other methods with some similarity, the teachers cited Project Based Learning and Problem Based Learning. The reason was that the CBL appears to be an evolution of such methods, presenting some expressive innovations, such as freedom of choice by the student, flexibility of the study and integration into the teaching environment.

VI. CONCLUSION

This paper presented the application of CBL for the teaching of software development for mobile devices. It is interesting to highlight that according to one of the interviewees, some aspects of CBL contributed to the reduction of both course evasion and failing, what is a relevant aspect of teaching.

The main problems observed during the application of CBL were difficulties in monitoring the CBL method and in the construction of applications due to the lack of previous experience with software development, and the need to choose students with a differentiated profile. The actions taken to solve the problems were the inclusion of presentations of current problems in several distinct areas and of external projects, both by researchers and entrepreneurs with the goal of aiding students to choose the final challenge; the inclusion of regular lectures and reinforcement for cases where the individual study was not enough; and changes in the selection process to select students with a profile appropriate to the requirements of the CBL (active participation, engagement, resourcefulness) and the program (part time, app publishing).

The main findings of this research regarding the use of CBL in software development are: overall the CBL framework is effective for teaching and learning mobile app development, and is worthy of continued research; faculty need more training and ongoing support implementing CBL; CBL can evolve beyond a teaching framework provided that it is merged with a software development process; CBL needs to be better integrated with a software development framework to be used in the context presented in this research; that students arriving with less background need more scaffolding; further research is needed to identify the characteristics of successful participants; and the framework needs to be continually updated and contextualized to meet the needs of the students and faculty, including the addition of other methods.

In future classes it will be necessary to combine other methods to identify relevant problems. The implementation of the solution is not formally defined in CBL by design, and it is mandatory to include some method of software development, especially for students who do not have previous experience.

References

[1] Apple. (2008). Apple Classrooms of Tomorrow—Today Learning in the 21st Century.

[2] Attridge-Stirling, J. (2001). Thematic networks: an analytic tool for qualitative research. *Qualitative Research*, *1*(3), 385–405. https://doi.org/10.1177/146879410100100307

[3] Coughlan, P., & Coghlan, D. (2002). Action research for operations management. *International Journal of Operations & Production Management*, *22*(2), 220–240. https://doi.org/10.1108/01443570210417515

[4] de Oliveira Fassbinder, A. G., Botelho, T. G., Martins, R. J., & Barbosa, E. F. (2015). Applying flipped classroom and problem-based learning in a CS1 course. In *2015 IEEE Frontiers in Education Conference (FIE)* (pp. 1–7). IEEE. https://doi.org/10.1109/FIE.2015.7344223

[5] Flora, H. K., & Chande, S. V. (2013). A Review and Analysis on Mobile Application Development Processes using Agile Methodologies. *International Journal of Research in Computer Scienc*, *3*(4), 8–18. https://doi.org/10.7815/ijorcs.34.2013.068

[6] Gilboy, M. B., Heinerichs, S., & Pazzaglia, G. (2015). Enhancing student engagement using the flipped classroom. *Journal of Nutrition Education and Behavior*, *47*(1), 109–114. https://doi.org/10.1016/j.jneb.2014.08.008

[7] Kinnunen, P., & Malmi, L. (2005). Problems in Problem-Based Learning–Experiences, Analysis and Lessons Learned on an Introductory Programming Course. *Informatics in Education-An International Journal*, *4*(2), 193–214. Retrieved from http://www.ceeol.com/aspx/issuedetails.aspx?issueid=31a3afb2-d1d0-4c59-9692-0b5fb5be23a1&articleId=49c3dfc7-475b-4e17-b39b-6ca554648471

[8] Lewis, G. A., Nagappan, N., Gray, J., Rosenblum, D., Muccini, H., & Shihab, E. (2013). Report of the 2013 ICSE 1st international workshop on engineering mobile-enabled systems (MOBS 2013). *ACM SIGSOFT Software Engineering Notes*, *38*(5), 55. https://doi.org/10.1145/2507288.2507327

[9] Nichols, M., Cator, K., Torres, M. & Henderson, D. (2016). Challenge Based Learning User Guide. Redwood City, CA. Digital Promise.

[10] Nichols, M. H., & Cator, K. (2008). Challenge Based Learning White Paper. *Apple Inc.*

[11] Prince, M. (2004). Does Active Learning Work ? A Review of the Research. *Journal of Engineering Education*, *93*(3), 223–231. https://doi.org/10.1002/j.2168-9830.2004.tb00809.x

[12] Robinson, J. K. (2013). Project-based learning: Improving student engagement and performance in the laboratory. *Analytical and Bioanalytical Chemistry*, *405*(1), 7–13. https://doi.org/10.1007/s00216-012-6473-x

[13] Rosiene, C. P., & Rosiene, J. A. (2015). Flipping a Programming Course: the Good, the Bad, and the Ugly. In *Frontiers in Education Conference (Fie), 2015* (pp. 803–805).

[14] Scharff, C., & Verma, R. (2010). Scrum to support mobile application development projects in a just-in-time learning context. In *Proceedings of the 2010 ICSE Workshop on Cooperative and Human Aspects of Software Engineering - CHASE '10* (pp. 25–31). New York, New York, USA: ACM Press. https://doi.org/10.1145/1833310.1833315

[15] Schraw, G., Flowerday, T., & Lehman, S. (2001) Increasing situational interest in the classroom. In: *Educational Psychology Review*, 2001, p. 211-224.

[16] Smartphone sales worldwide 2007-2015. (n.d.). Retrieved February 2, 2017, from https://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/

[17] O'Mahony, T. K., Vye, N. J., Bransford, J. D., Sanders, E. A., Stevens, R., Stephens, R. D., … Soleiman, M. K. (2012). A Comparison of Lecture-Based and Challenge-Based Learning in a *Workplace Setting: Course Designs, Patterns of Interactivity, and Learning Outcomes*. *Journal of the Learning Sciences*, *21*(1), 182–206. https://doi.org/10.1080/10508406.2011.611775

[18] Marin, C., Hargis, J., & Cavanaugh, C. (2013). iPad learning ecosystem: Developing challenge-based learning using design thinking. *Turkish Online Journal of Distance Education*, *14*(2), 22–34

[19] Santos, A. R., Fernandes, P., Sales, A., & Nichols, M. (2013). Combining Challenge-Based Learning and Scrum Framework for Mobile Application Development. *ACM Conference on Innovation and Technology in Computer Science Education*, 189–194. https://doi.org/10.1145/2729094.2742602

[20] Cheng, W. L. S. (2016). Application of Challenge-Based Learning in nursing education. *Nurse Education Today*. https://doi.org/10.1016/j.nedt.2016.05.026

[21] Gabriel, S. E. (2014). A modified challenge-based learning approach in a capstone course to improve student satisfaction and engagement. *Journal of Microbiology & Biology Education*, *15*(2), 316–8. https://doi.org/10.1128/jmbe.v15i2.742

[22] Rotgans, J. I., & Schmidt, H. G. (2011). The role of teachers in facilitating situational interest in an active-learning classroom. *Teaching and Teacher Education*, *27*(1), 37–42. https://doi.org/10.1016/j.tate.2010.06.025

[23] Berbel, N. A. N. (2011). As metodologias ativas e a promoção da autonomia de estudantes. *Semina: Ciências Sociais E Humanas*, *32*(1), 25–40. https://doi.org/10.5433/1679-0383.2011v32n1p25