A Design Sprint based model for User Experience concern in project-based learning software development

1st Vinícius Gomes Ferreira
Federal Institute of Mato Grossso do Sul
Três Lagoas, MS, Brazil
vinicius.ferreira@ifms.edu.br

2nd Edna Dias Canedo
Department of Computer Science
University of Brasília (UnB)
Brasília, DF, Brazil
ednacanedo@unb.br

***Abstract*—This Innovative Practice Full Paper presents a model that includes ways to propose and construct software artifacts concerned with User Experience issues in Project-Based Learning (PBL) semesters long units. Project-Based Learning has been a trend in conducting teaching practices in modules related to software development. One of the cornerstones of Project-Based Learning is authenticity, a principle that guides the creation of software and development in the classroom by grounding it in a real-world context. However, the authenticity as addressed today in Project-Based Learning for software development, while recognizing that it is important to have quality in the end-user artifact, does not explicitly include User Experience (UX) in its scope. Creating software products that consider UX can have several advantages for students who create them and for a community that will benefit from these systems. This paper presents a proposal for a Project-Based Learning model that uses UX in the scope of projects. This model was built based on theoretical evidence found in a systematic literature review and practical evidence found in three exploratory case studies conducted during the development of this work. The purpose of the model is to increase the acceptance rate of the software developed by the students and help them in creating a portfolio for seeking their first job. The advantages and limitations of the model are discussed and reported. The results indicate that the model is suitable and can be used in any software development discipline and related areas.**

***Index Terms*—Design Sprint, User Experience, Project-Based Learning, active learning**

I. Introduction

One of the most effective ways to conduct educational training of new Information Technology (IT) professionals, especially concerning new trends and critical knowledge of the profession, is to teach them during their undergraduate degree through hands-on exercises that simulate the everyday situations that professionals typically encounter in the software industry [1]. In this context, research in practical education for software development proposes that constructivist teaching tools occupy this role [2], being Project-Based Learning (PBL) one of the most used for this purpose [3]–[5]. The idea behind PBL is the possibility for students to develop their knowledge through the construction of artifacts based on a real-world context that can be perceived and evaluated by others [6].

The pedagogical model of PBL considers authenticity, the foundation of the artifact to be developed in a real-world context, as one of its most important principles [7]. Among other things, an authentic teaching unit is one that is also concerned with the quality of artifact produced in the classroom [1]. In this way, it is reasonable to believe that for software development that takes place within Project-Based Learning, the user experience should be considered as an important aspect of authenticity. Comparing the artifacts developed by students with those produced by professionals in terms of their acceptance and usage, user experience (UX) should be considered as a key aspect of a successful software system. Thus, the software developed by them must be minimally comparable to competing software developed by professionals in the industry, obtaining a competitive advantage.

There are some pedagogically encouraging results of the PBL application reported in the literature [8], [9], including those related to software development [10] but there are not many papers that propose to study ways to explicitly include UX in the scope of PBL software development activities. The closest PBL activity to this goal is brainstorming for prototyping [11], without a focus on empathy exercises, as UX design models propose to be done [12].

This paper documents the process of creating a project-based learning pedagogical model for software development, in which the main objective is the inclusion of User Experience as an explicit concern in the development of the artifact and the search for the authenticity of the produced artifacts. The proposed model is based on the results found in the execution of a systematic literature review and on empirical evidence collected through exploratory case studies in the work we presented in [13] and [10].

This work is included in a project of creation and validation of the model that targets the inclusion of user experience (UX) concerns into software development project-based learning life cycle, but also has the following collateral results: (1) greater community and industry acceptance of software developed by the students; (2) the possibility of creating a project portfolio for students during their undergraduate degree; (3) mitigate the difficulties that students encounter in finding their first job that is a difficulty caused by the lack of proven experience in software projects developed; (4) increase student engagement in software development courses through additional motivation created by the authenticity of academic projects.

In this proposed model, we intend to propose a Design Sprint based brainstorming activity into proposal phase of Project-Based Learning (PBL) due to frequent use of it for achieving UX results in software development projects [14] and a shorter time used by this execution compared to other UX processes like Design Thinking [15], [16].

This paper is divided as follows: section I brings this introduction. Section II presents the theoretical foundation of Project-Based Learning and User Experience, which are important for understanding this work. Section III describes how the Systematic Literature Review and the exploratory case studies were configured as the research methods of this work. Section IV describes the model, its justifications, and its limitations. Finally, section V presents the conclusions and points out the next steps of the project that includes this work.

II. Background

A. Project-Based Learning (PBL)

Project-Based Learning (PBL) is an approach that uses projects as pedagogical tools [7], [8]. The centrality of the project is one of the main limits that determine whether or not a student-centered pedagogical approach can be called PBL [17]. The role of a teacher within a PBL unit should not be the one of a content exhibitor, but of a facilitator, tutor, or coach who will guide the student through scaffolding techniques [18].

The focus of a PBL is on the development of social, cognitive, and metacognitive skills through artifact-building tasks and not just immersing students in authentic professional practice environments [7], such as supervised internships. Artifact construction is therefore one of the main features of a PBL unit [7]. Observing the principles of authenticity, it is possible to say that an artifact produced in a PBL unit should be driven by real-world problems [8]. The possibility that this artifact is perceived, well evaluated, and used by real people is one of the main motivators for the special attention devoted to authenticity in constructionist teaching models, as well as the PBL [6], [7].

We did not find in the literature studies that perform an assessment of the potential benefits of Project-Based Learning in software development for the creation of real value to the community outside the classroom. Although the software projects developed from the PBL units can be put to use [19]–[21], these products are not concerned yet with the aspects related to the user experience that will use them. When projects add aspects of user experience, they have the potential to add value to both, the users, who will be able to receive low cost, good quality software, as for the students, who will have experience and portfolio to present themselves in search of their first job and consequently, have their motivation and engagement levels increased [1].

B. User Experience and its work processes

User Experience (UX) is a discipline that goes beyond preoccupation with the achievement of behavioral goals to consider what kinds of effects can be achieved on the user through their interaction with an interactive product [22]–[24]. UX directs its design efforts to create experiences that will make users enjoy the interaction and have their emotions, cognition, and motivation positively affected [24]. According to Hassenzahl [25], two dimensions are of concern for UX as an approach: the pragmatic quality dimension, composed by the utility of the product, its usability and its ability to provide functionality; and the dimension of hedonic quality, consisting of goals of being that are linked to emotion and satisfaction of use [25].

In User Experience, as shown in Figure 1, the experience is an event that can influence users by causing users to set expectations before using a particular product or service (Anticipated UX), creating user satisfaction while using a product or service (Momentary UX), making users have good memories after using the product or service (Episodic UX) or cumulatively affecting users and the next experiences that users will have with a product or service (Cumulative UX) [26].

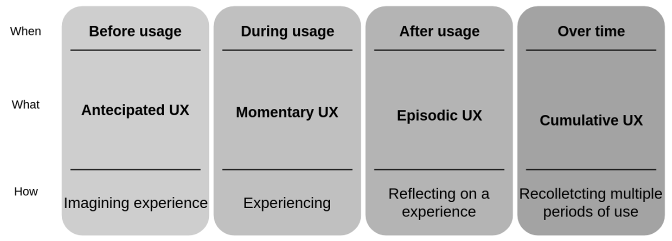

Fig. 1. Time intervals for UX. Adapted from Virpi et al. [26]

The theoretical basis of the UX is supported by concepts from areas such as cultural anthropology, engineering, journalism, business, psychology, and graphic design, besides the imported concepts of human-computer interaction (HCI) and usability, which are the closest disciplines of UX [12], [27]. In practical terms, a UX practitioner seeks to align the business goals of any venture (which usually revolve around economic motivations) with the needs of the user [12].

One of the beliefs of design and UX research is that developers of products and services need to learn gradually from their users through a continuous cycle of construction, measuring, and learning [28]. Some principles that help delimit the design of software artifact experience have emerged from this belief [12]. One of these principles is that the user experience is inevitable and wherever the user is and whatever they are doing, somehow they will be feeling an experience, whether good or bad. Thus, the experience of this user must be developed intentionally, while leaving it to be created by accident could be a reason for the abandonment or for the non-acceptance of that software artifact [12].

This same belief also applies to software artifacts developed in PBL units. Taking into account that if they are intended to add value to someone or the community outside the classroom, they will compete with products made by trained professionals who are viewed as more competent in software engineering activities. Software designed in a PBL unit using the user experience can give them an advantage and increase the chance of being well received and used. The authenticity of the PBL unit, recognized in the literature as a motivator for students [1], also addresses the usefulness of the artifact being developed [29] and, in a broader view, can also be about the satisfaction of using the software produced.

One of the processes used by UX designers for initial design development and requirements engineering assistance is Design Sprint, a pragmatic and well-structured instantiation of Design Thinking and User-Centered Design [15], [16]. According to Banfield et al. [16], Design Sprint is an exercise in five phases restricted by time to reduce the risks in the act of bringing a product to market. Similarly, software artifacts developed by students also suffer from risks of not adhering to the expectation of the users who demand them. Knapp et al. [15] presented Design Sprint as an alternative to brainstorming used in the ideation stages of innovation projects, as brainstorming is widely criticized for its lack of structure and inefficiency in mitigating the bias of the participants [30].

III. Methodology

In this work, we use both theoretical and empirical evidence to construct a model to be adopted in PBL units within the classroom. The theoretical evidence was collected from the results of a Systematic Literature Review (SLR) [10] and form a set of information that elucidates how software is developed in PBL units, how participant groups are divided, what software engineering processes are commonly used, how pedagogical teaching issues are handled, and what learning outcomes are reported for students. We present the results of the SLR in the work developed by [10]. The empirical evidence was collected through four exploratory case studies performed iteratively over three semesters: the first semester of 2018 (2018.1), the second semester of 2018 (2018.2), and the first semester of 2019 (2019.1). Figure 2 presents the steps adopted in defining the construction of the proposed model to be presented in Section IV.

A. Theoretical Evidence

A relevant concern for this work was how to maintain the software's authenticity (a property that defines the proximity of job did by students and job did by professionals [29]) its development after UX concerned proposal had been made. According to our previous work [10], it was possible to identify that making authentic software in a PBL unit involves multiple factors, among which it is possible to find educational support for students, the proper assembly of a software engineering process and the time available for the execution of this unit. The instructional forms of these PBL units use active approaches such as coaching, flipped classroom, peer teaching, and hands-on classes, but do not abandon traditional teacher-centered teaching that takes place through direct or on-demand instruction. Entire semesters are used in PBL units and division of classes over the weeks is required. In general, most of these classes are used for practical activities, after some kind of theoretical instruction on software engineering processes.

The importance of dividing students into small manageable groups with 2 to 5 students is also reinforced by what was found in the primary studies we reported in [13]. Also, the use of agile practices and methods is ostensibly adopted, especially the Scrum for project management and distributed development practices, because of difficulty in keeping students in the same place working on the project for 5 days a week. The advancement in learning rates reported in [13], indicates that making use of authentic learning in software development education increases academic performance of students and their perception of their learning.

B. Empirical Evidence

In conducting this research, three exploratory case studies were conducted over three semesters in 2018. In total, 34 students participated, 10 of them female and 24 male. The first case study took place in a Requirements Engineering discipline which has as a prerequisite the Software Engineering discipline, for which a 3-hour class and a pre and post-class time for preparation and validation, respectively, were reserved. It resulted in a prototype of a search app for internships with tutorials for improvement of focused skills related to most frequent internship (technical or socio-affective) issues. The second case study took place in Software Engineering discipline, which has as its prerequisite the Object-Oriented Analysis and Object-Oriented Programming disciplines, both taught to students who have passed the Algorithms discipline.

In this case study, we reserved 4 classes distributed in 3 weeks, all with 3 hours, and a pre-class moment for preparation. It resulted in a prototype of a web platform for validating paper documents based on the random choice of words in the document, content summarized by hash algorithms, and document linkage based on their timestamps. The third case study took place in an Agile Methods teaching project which was also reserved 4 lessons distributed over 3 weeks each with 3 hours, but where changes were made in the collection of exercises performed inside and outside the classroom. It resulted in a prototype of a low-cost system for identifying peoples and their vehicles to combat thefts. This system uses mobile phones, printed QR Codes, and log generating. All students were enrolled in Analysis Technology and Systems Development of the Federal Institute of Education, Science, and Technology of Goiás - Campus Formosa, which takes place in the morning and has 6 semesters. The application of the first case study was performed with students from the fifth semester, the second case was conducted with students from the fourth semester and the third case study was conducted with students from all semesters of the course [13], [10].

The purpose of the three case studies was to obtain answers to questions that had not been found in the literature. The absence of these questions is because there are few empirical

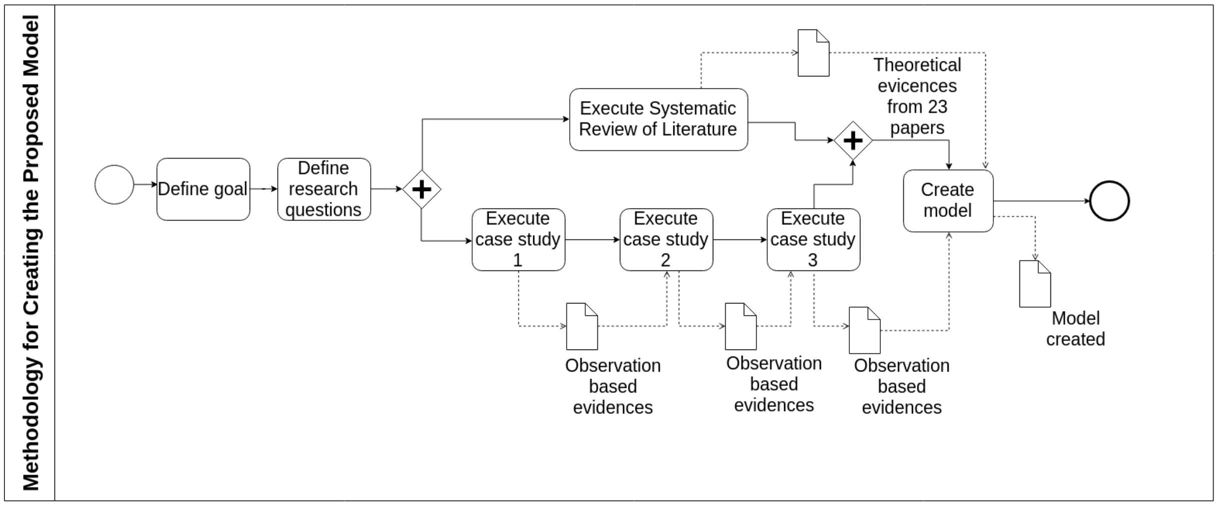
Methodology for Creating the Proposed Model
Define goal
Define research questions
Execute Systematic Review of Literature
Theoretical evicences from 23 papers
Execute case study 1
Execute case study 2
Execute case study 3
Create model
Observation based evidences
Observation based evidences
Observation based evidences
Model created

Fig. 2. Steps of Creating the Proposed Model

studies on Design Sprint inside the classroom and the lack of reports of practical considerations of its use. The application of the first case study showed that one of the biggest challenges of transforming Design Sprint into an active learning tool was related to time. The problems with time occurred in two dimensions: (1) Total time for Design Sprint's core activities is strictly defined for just one classroom meeting that has become impractical because of students being late for both the start of the class and return of the break. Therefore, it is considered very important to have extra time to take into account these delays and also the bursting time of activities and the possible need for a more detailed explanation of them; (2) there may be a need to reapply exercises that were not well performed by failures of the participants in understanding or by failures in facilitator conducting them.

The homogeneity of the technical background of the students ended up interfering with the end of the proposed solution. This was evident during the choice of aspects of the problem to be treated when most of the problems pointed out by the specialist were disregarded. In the first case study, it was also noted that students tend not to be silent in silent voting exercises, requiring the facilitator some kind of group management skills. It was also observed that the business specialist invited to the Design Sprint may omit information or be intimidated in his/her speech about the problem if he/she is left free to speak within the set time without any structure to guide him/her. Because of it, the final prototype addressed more issues pointed by students than issues pointed by the specialist in comparison with the prototype resulted in the other two case studies.

The format of the second case study was modified to mitigate the limitations observed in the first one. Therefore, its core was designed to be executed in just one class. Besides, a step of conducting semi-structured interviews with people outside Design Sprint has been added. People with a profile of a typical user of the processes that the students were trying to improve, and also of a technical profile who could point out to them the constraints that technology could offer in the interval between first and second classes. During this time the students were also encouraged to benchmark, looking for existing solutions that somehow already addressed the problem they were addressing. There was also a moment with exercises so that students could be inspired and unlock their creative flow before moving on to the final solution. As a way of countering exercise timeouts and providing the opportunity to redo those exercises that had not been well performed, a limit of two and a half hours was set for the classroom meetings.

In the second case study, one of the difficulties observed was related to the absenteeism of some students in certain classes, thus revealing the first trade-off to be considered when adapting Design Sprint to educational environments: the amount of time set aside for Design Sprint against the risk of student absenteeism in classes spread over a few weeks since that using just one class, students who started Design Sprint are more likely to be the ones who will finish it. However, dividing classes over the weeks allowed students to conduct semi-structured interviews that helped mitigate the bias of technical background homogeneity and avoid students explore issues pointed out by themselves, since, unlike the first case study, in this case study, everything the external community recommends was aggregated in the final solution.

A break for participants allowed that the time had not run out and there was more time to instruct the activities to be performed. However, prototyping and validation lacked better direction in the sense that, after students had done them, they could not accurately say what had worked and, especially, what had failed the test with users. This second case study also highlighted something that had not happened in the first, which was the participation of a student who isolated herself from the activities. Although there were efforts by the professor to encourage her to participate, she continued to remain isolated and this showed that not all students will be motivated to participate even if they are in the classroom to ensure attendance and assessment. Also, it is important to consider the availability of the business specialist for participation, who in this case study could only participate in the first class and prototype test.

To complement the performed case studies and intending to mitigate the faced problems [13], we used Design Sprint so that we could maintain the strengths, such as contact with the outside community representing typical user-profiles and technology experts, such as benchmarking existing solutions to the problem they are addressing, and the classroom activity ceiling with limited available time, which in this case has dropped to 2 hours and 10 minutes.

To combat absenteeism, the amount of exercise was wiped so that they would fit into three classes that would be performed in just two weeks. To make this possible, prototyping was carried out of the classroom, being performed between class 2 and class 3. The designed prototype began to focus mainly on the flow of user interaction and on the validation of assumptions that students create during class 2. To expedite the making of this prototype, as class 2 and class 3 took place in the same week, students started with ready-made wireframes, leaving aside concerns about the graphic design of the solution.

Some activities have been modified to better meet the needs of students in the classroom. For example, an exercise aimed at identifying the emotional states of the specialists in the first class was added, and an assumption and test exercise based on this emotional state to check whether or not the solution was successful during validation. Besides, assuming no solution is completely new, Design Sprint's problem has begun to be regarded as an improvement on existing processes, such as dissatisfaction with some legacy system or some unauthorized way of doing something so that it could be easier to draw parameters that indicate explicit improvements in the way users were already performing their activities.

Another change was the way How Might We (HMW) exercises should be handled, not just an exercise of inspiration and convergence, but a true creator of driving questions [11] that should be pursued during Design Sprint activities and answered with the validation result, thus delimiting the scope of the problem and the solution that would be made by the students. In this case study, absenteeism appeared again. Another important observation of this application was the need to well choose the problem to be solved, avoiding problems that probably involve the use of special hardware or the change of institutional policies. The problem addressed in this case study related to the safety of the campus where the students were studying, and invariably, it was necessary to address the problem of some malfunctioning turnstiles. Because of this, validation became very complicated and difficult to perform, extending far beyond the previously planned time and causing students to start missing classes and lose engagement at that time.

Although the three case studies performed had their problems the student-created solutions, namely: a redesign of the non-compulsory supervised internship process; a solution for validating the integrity of physical documents; and a people and property identification mechanism that uses smartphones at a very low cost; were well received by the community. The internship solution became a technology initiation project of a student and inspired a course completion project for another student. The solution regarding the validation of physical documents was highly praised at a public meeting and the proposal to use smartphones to create an identification system was taken to be discussed at higher levels. This result points to the potential use of Design Sprint by students to contribute to the academic community in which they are inserted.

IV. Proposed Model

At the end of the application of Design Sprint in the PBL units and having generated the solution, the students were divided into groups that worked on their projects. All proposed solutions were presented to the demanding community. Internal community issues are recommended because of the ease of integrating the experts and the motivational aspect of the students (case studies).

From the results obtained through the systematic literature review and the considerations made through the observation of exploratory case studies, it was possible to propose a PBL model for software development that addressed the user experience (UX) within its scope. Figure 2 presents the steps performed in the construction of the proposed model.

The built model has the following objective: *Make the software developed by students acceptable in comparison with software developed by professionals trained for the target audience whose profile is the same as that of the demanding community.* The research questions that guided the construction of the model, answered through case studies and systematic literature review were:

- **Case studies**
  - How does Design Sprint behave as an active learning tool in the classroom?
  - What are the likely benefits of Design Sprint as an active learning tool?
  - What were the benefits and challenges presented in adopting Design Sprint?
  - Which Design Sprint exercises have the best potential to mitigate student bias caused by the homogeneity of interests and diversity of knowledge?
- **Systematic Literature Review:**
  - What are the pedagogical benefits presented by PBL applied in software development disciplines?
  - In PBL units how are developed the software that is well accepted and evaluated by the demanding community?

Figure 3 presents a macro view of a PBL unit, whose ideation and prototyping step has been replaced by Design Sprint. The proposed model makes use of Bender's proposal for PBL [11], Design Sprint [15], [16] and eduScrum [31]. According to the theoretical evidence [10], it is recommended that a period of classes with direct instruction on software engineering principles should be provided to participating students in PBL units, especially about conducting user interviews and eliciting requirements that are very useful in the external activities that take place between classes 1 and 2 of the proposed model.

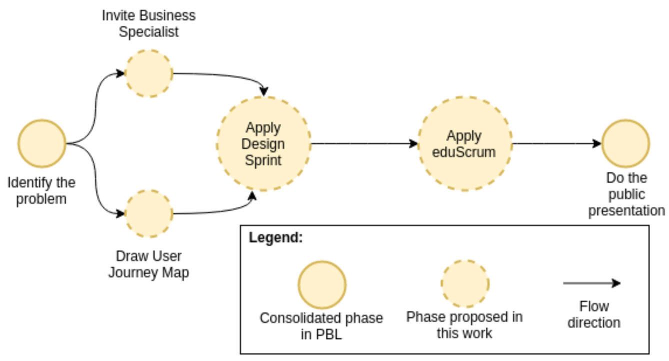
Invite Business Specialist
Identify the problem
Draw User Journey Map
Apply Design Sprint
Apply eduScrum
Do the public presentation
Legend:
Consolidated phase in PBL
Phase proposed in this work
Flow direction

Fig. 3. Proposal macro template

A. Identify the Problem

The first phase of the model happens with the collection of the problem to be solved (Figure 3), which is also an activity that happens outside the classroom. Although the model presented by Bender [11] does not treat PBL as a series of phases to follow, it makes clear that it is necessary to start from a problem that will later result in a driving issue [11]. In this model, we propose that the problem should be captured with a real stakeholder and not artificially created by the teacher of a subject for didactic purposes. It is described in the literature that external agents and a real problem of these agents have the power to motivate students in a unit of PBL, justifying the necessity for other than the professor to set the problem [7].

Because of the importance of valuing the class time of the PBL teaching unit, the capture of the problem must happen before the commencement of the project activities, preferably before the semester has begun. Without the pressure of time, the professor may be free to look for relevant problems and evaluate appropriate alternatives for both authenticity and complexity. The problem must be adjusted to the estimated capacity of the students in the class that will be a PBL participant, and it is strongly recommended that the problem does not involve solutions beyond software development. For example, choosing a problem that involves physical security from somewhere would lead to a project deemed inappropriate since it refers to the need for hardware inclusion, building structures, and change in institutional policies as part of a possible solution. We experience this difficulty during the execution of the case studies.

B. Invite Business Specialist

Even before PBL activities, the problem specialist must be invited (Figure 3). It is also necessary that the invitation is made in advance because if it is verified the unavailability of the specialist to be present in the class that will understand the problem, validation, and public presentation, it will be necessary to find another specialist with availability. Also, the invitation should be made in person, so that the professor can talk about the problem and draw the journey that the user has, and that includes the experience problems he/she is experiencing. A recommendation of the model is for the business specialist to be part of the community in which the students participate, such as being from the institution itself or the common life of the participants. If students are involved with the problem they are trying to solve, they are likely to be more engaged in solving activities, and it is easier for them to aggregate business domain knowledge and find people to interview for outside activities that take place between Design Sprint classes 1 and 2.

It is assumed that somehow the business expert does some unsatisfactory activity to reach a certain goal rather than imagining that the solution will be entirely new. Having this in mind makes it easier for the professor to discuss the potential problems to be addressed in the Design Sprint to the expert at the meeting that takes place outside of the classroom, as well as understanding how the expert already performs his/her tasks so that it is possible to identify the bad emotional states and set up the assumptions to be validated, as presented in Figure 4.

The expert should attend the first day of Design Sprint Application Detailing, where the problem will be contextualized, being responsible for the first lightning lecture in the convergence phase, as presented in Figure 4. The presentation of the expert includes the ”anchor” strategy described in Bender’s PBL model [11]. The anchor is an introduction to delivering basic information through an activity other than a lecture. It is used to communicate the driving issue while motivating students to participate in a PBL unit [11].

C. Draw User Journey Map

During the invitation to the expert, the teacher should seek to understand the sequence of tasks that the expert performs that address the experience problems he/she is experiencing and his/her ultimate goal as a user when performing these tasks. This task sequence is commonly referred to as a User Journey (Figure 3) by the UX community [12], [22]. As the expert describes his/her user journey, the professor should draw it so that it is not so detailed to make it too complex. This User Journey Map will be reproduced on a whiteboard or flip chart in the classroom while Design Sprint is running.

This phase is also important for the teacher to be able to verify that the problem to be solved is too complex for students or goes beyond the scope of software development since case studies have made it possible to identify problems in developing a non-software solution. If this is verified, the teacher may decide to choose another expert or choose another UX problem with the same expert. These adjustments are part of the process, as it is necessary to contextualize the problem with the scenario to which the PBL unit will be applied.

D. Apply Design Sprint

The Apply Design Sprint phase (Figure 3) takes place inside and outside the classroom to enable students to recognize the problem, study, and propose solutions, create and validate a prototype. The learning obtained and the prototype will be the basis for the software development in the later phases. Design Sprint consists of a series of creativity (divergence) and decision (convergence) exercises, guided by time limits, freedom of creation, and consensus. A script and execution details can be found in the repository [32].

Classes take place over two weeks. In the first week, the problem-related phases of divergence and convergence are performed, where students attend the expert lecture, map their emotional states on the User Journey Map, and write How Might We (HMW) questions that will represent the driving issues of the PBL. Between week one and week two, students make external interventions through interviews with typical users of the solution they are proposing and experts on specific aspects of technology (usually other professors of these students). Besides, in this interstice between weeks, students also search for ”competing solutions” that somehow already address the identified problem (benchmarking). The main purpose of these activities is to enable students to be inspired before the solution creation exercises and to mitigate possible biases caused by the homogeneity of their technical background.

The second and last week is composed of moments of inspiration, creativity, and assumptions test. The second week contains two classes, one for the presentation of what was learned in external interventions, design a large number of solutions and choose by consensus the features that will integrate the prototype and will be tested on the validation day, which takes place in the second class of this week. Figure 4 shows the details of the application of Design Sprint in PBL units.

After the validation, students should come together to review Design Sprint artifacts and, from the artifacts, extract the user stories that will compose the software backlog. Both user stories and Design Sprint learning will become inputs for software development during eduScrum.

E. Apply eduScrum

EduScrum is an education method based on the agile Scrum methodology [31]. It is a framework for training students where responsibility for the learning process is delegated from teachers to students [31]. EduScrum is adequate to the constructionist view, as it values student autonomy and peer collaboration, which preserves the attributes of authenticity present in current PBL models [31], thus appropriate to this proposed model. The choice to integrate eduScrum into this PBL model was due to the indication that Scrum is one of the most used processes in PBL units for software development that brought positive results [10].

The eduScrum used in this model (Figure 3) is divided into two Sprints, the first being Sprint for configuring the infrastructure required for artifact development and software architecture modeling, and the second Sprint for solution development. In the first Sprint, learning objectives are linked to software design, planning of project activities, and architectural decision making. Its time should only take up a quarter of the time available for PBL eduScrum. In the second Sprint, learning objectives orbit around software development and testing techniques, configuration management (versioning), quality assurance, tools usage, and teamwork. These techniques are also found in PBL units and have good results [10]. This phase will end with a second validation performed with the business expert, this time on the produced software. Its time will occupy the other three-quarters of the time available for eduScrum.

In the eduScrum of this model (Figure 3), students are divided into groups of 4 or 5 people. Both numbers of students per group are inside the of 2 to 5 for the group found in Systematic Literature Review and avoid overload of work for students. Each group has its own eduScrum Master, who has the responsibility of choosing the members of their eduScrum team, taking care of the Flip (a kind of Kanban board of the team by which they measure their progress), and ensure that eduScrum rules are adhered by the team they lead. The professor acts as the Product Owner of eduScrum, responsible for backlog with user stories, and defines the criteria for acceptance of each Sprint. The acceptance criteria for the second Sprint should be linked to the validation made with the specialist at the end of the second Sprint. Also, the professor is responsible for determining staff training needs, providing educational support as needed through any of the teaching strategies. Finally, the eduScrum team is responsible for achieving learning objectives, developing PBL artifacts, validate the artifacts with the professor and the specialist and agree on when the work will be completed (definition of done) and what conditions are needed to ensure a pleasant work (definition of fun) [31].

F. Do the Public Presentation

At this stage students publicly present their produced artifact. This is an important principle of constructionism and authenticity [6]. The format of the presentation should not be determined by the professor but chosen by the students themselves. The audience for this presentation may vary in size and technical background, but the business expert must be part of the audience. Thus, it is recommended that there are educational agents who can understand and value the pedagogical contributions of PBL present in the audience. In this way, the culture of learning by constructionist means can be reinforced, making other iterations of PBL better accepted in the academic community and more issues listed for further treatment within future PBL units.

G. Limitations and Threats for the Proposed Model Validation

The proposed model has some limitations, which means that it is not applicable in all situations, either because one of the case studies has shown it or because it was not tested under these conditions. The case studies have shown that the

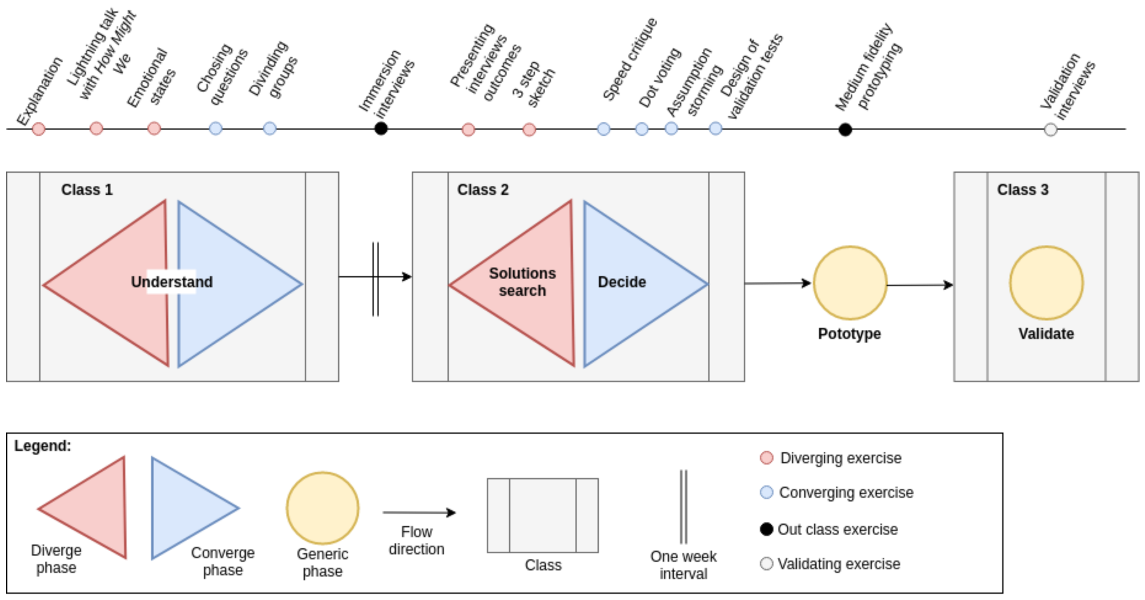

Fig. 4. Design Sprint Application Detailing

facilitator professor must have skills in group management, time management, ability to improvise, and skills related to motivation. It's important that professors that run this model feel comfortable with these skills. Also, doing work like this instead of traditional classes increases the teacher's workload, both in the classroom and outside, and needs some persistence. The fact that collecting and studying the problem and the flow of people who suffer from it is placed outside Design Sprint is somewhat costly for the professor, because it depends on prior organization and reorganization by the professor who may not find the best problem to address or the best business expert to participate in Design Sprint activities. Professors with high workload may want to decline to perform these activities. One way to address this issue is to organize materials before the subject semester starts.

Not all students will benefit from Design Sprint equally. There are students with introspective profiles who may refuse to participate in the activities and even skip classes to escape from group work and a potential exposure that activities at Design Sprint may bring. There is also the case of students who will be in the early years and will not have the necessary skills for creativity, prototyping, and, above all, software development and design. Design Sprint has not been tested with a class composed solely of first or second-semester students. It has also been tested only with students from an undergraduate degree focused on software development and there is not enough information to tell if it behaves well also with students who are from areas that do not directly involve software building. None of the case studies in this research had more than 13 students in a single application. Thus, the number of participants may also be a limitation.

It should also be noted that although Design Sprint was studied in this project and eduScrum is a recommended model for adopting Scrum in educational projects, there was no test to see how the two approaches work together. Besides, Design Sprint has not been tested yet under experimental conditions compared to brainstorming.

V. Final Considerations

This paper presented the steps taken in the creation of an alternative model to brainstorming to implementation of a PBL unit for the development of software. The purpose of the proposed model was to include a concern with user experience in software development during classroom activities, intending to continue a project that addresses the issue of the low acceptability of the software developed by the students and the lack of experience of students in finding their first job.

This work presented the empirical and theoretical foundations of the proposed model, obtained by exploratory case studies and a systematic literature review [10]. By applying the case studies it was possible to verify how Design Sprint behaves within an educational context. With the results found it was possible to identify how other PBL applications have achieved authenticity in their software development projects and gave insights about how to maintain the authenticity of the proposal during the software construction process of the PBL unit. This information, combined with theories and methods already consolidated in the literature, allowed us to build a PBL model that considers the principles of user experience and constructionist pedagogy in the creation of a solution proposal for software development and a way to construct it during the semester.

A. Future Works

As future work, the next steps of this research will be:

- Perform tests in experimental conditions to confront the model created with traditional brainstorming as recommended by Bender [11];
- Conduct case studies with the complete model (Design Sprint and eduScrum) in the creation of software that will be quantitatively and qualitatively evaluated against competing for software within the same solution scope to verify their acceptability and other user experience metrics;
- Investigate student motivation items using the model for motivation indices of a traditional PBL unit approach to software development.
- Accompany students who participated in applying the model compared to students who did not participate to verify how many of them got their first job in the development and software area and how long does it take.

References

[1] R. McDermott, M. Zarb, M. Daniels, A. Nylén, A. Pears, V. Isomöttönen, and M. Caspersen, "The authenticity of 'Authentic' assessment some faculty perceptions," *Proceedings - Frontiers in Education Conference, FIE*, vol. 2017-Octob, pp. 1–9, 2017.

[2] A. Ferreira and G. Filho, "Modelo de Ensino baseado nos Métodos Ágeis de Desenvolvimento de Software," Ph.D. dissertation, Universidade Federal do Rio de Janeiro, 2016.

[3] J. A. Maćias, "Enhancing project-based learning in software engineering lab teaching through an e-portfolio approach," *IEEE Transactions on Education*, vol. 55, no. 4, pp. 502–507, 2012.

[4] A. Jaime, J. M. Blanco, C. Domínguez, A. Sánchez, J. Heras, and I. Usandizaga, "Spiral and Project-Based Learning with Peer Assessment in a Computer Science Project Management Course," *Journal of Science Education and Technology*, vol. 25, no. 3, pp. 439–449, 2016.

[5] M. Jazayeri, "Combining Mastery Learning with Project-Based Learning in a First Programming Course: An Experience Report," *Proceedings - International Conference on Software Engineering*, vol. 2, pp. 315–318, 2015.

[6] S. Papert and I. Harel, "Situating constructionism," p. 14, 1991.

[7] P. C. Blumenfeld, E. Soloway, R. W. Marx, J. S. Krajcik, M. Guzdial, and A. Palincsar, "Motivating project-based learning: Sustaining the doing, supporting the learning," *Educational psychologist*, vol. 26, no. 3-4, pp. 369–398, 1991.

[8] J. W. Thomas, "A Review of Research on Project-Based Leaming," 2000, working paper. [Online]. Available: http://www.bie.org/index.php/site/RE/pbl_research/29

[9] B. Condliffe, J. Quint, M. G. Visher, M. R. Bangser, S. Drohojowska, L. Saco, and E. Nelson, "Project-Based Learning A Literature Review Working Paper Prepublication copy: Release," 2017, working paper.

[10] V. G. Ferreira and E. D. Canedo, "Design sprint in classroom: exploring new active learning tools for project-based learning approach," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 57, pp. 1 – 21, 2019. [Online]. Available: https://doi.org/10.1007/s12652-019-01285-3

[11] W. N. Bender, *Project-based learning: Differentiating instruction for the 21st century*. Thousand Oaks, California, USA: Corwin Press, 2012.

[12] E. Stull, *UX Fundamentals for Non-UX Professionals: User Experience Principles for Managers, Writers, Designers, and Developers*, 1st ed. Berkely, CA, USA: Apress, 2018.

[13] V. G. Ferreira and E. D. Canedo, "Using design sprint as a facilitator in active learning for students in the requirements engineering course: an experience report," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC 2019, Limassol, Cyprus, April 8-12, 2019*. Limassol, CY: ACM, 2019, pp. 1852–1859. [Online]. Available: https://doi.org/10.1145/3297280.3297463

[14] V. Control. (2019, apr) What is a design sprint and why is everyone talking about it? [Online]. Available: https://voltagecontrol.co/what-is-a-design-sprint-and-why-is-everyone-talking-about-it-9ed024b969d

[15] J. Knapp, J. Zeratsky, and B. Kowitz, *Sprint: How to solve big problems and test new ideas in just five days*. New York, NY, USA: Simon and Schuster, 2016.

[16] R. Banfield, C. T. Lombardo, and T. Wax, *Design Sprint: A Practical Guidebook for Building Great Digital Products*. "Sebastopol, CA": " O'Reilly Media, Inc.", 2015.

[17] A. Morgan, "Theoretical Aspects of Project- Based Leurning in Higher Education," *British Journal of Educational Technology*, vol. 14, no. 1, pp. 66–78, 1975.

[18] ——, "Theoretical aspects of project-based learning in higher education," *British Journal of Educational Technology*, vol. 14, no. 1, pp. 66–78, 1983.

[19] C. R. Rupakheti, M. Hays, S. Mohan, S. Chenoweth, and A. Stouder, "On a pursuit for perfecting an undergraduate requirements engineering course," *Journal of Systems and Software*, vol. 144, pp. 366–381, 2018. [Online]. Available: https://doi.org/10.1016/j.jss.2018.07.008

[20] V. Mahnič, "Student projects as a means of cooperation between academia and industry: Some experiences in the area of software engineering education," *World Transactions on Engineering and Technology Education*, vol. 15, no. 3, pp. 239–244, 2017.

[21] F. Llopis and F. G. Guerrero, "Introducing competitiveness and industry involvement as learning tools," *IEEE Global Engineering Education Conference, EDUCON*, vol. 2018-April, pp. 298–307, 2018.

[22] M. Hassenzahl and N. Tractinsky, "User experience - A research agenda," *Behaviour and Information Technology*, vol. 25, no. 2, pp. 91–97, 2006.

[23] E. L.-C. Law, V. Roto, M. Hassenzahl, A. P. Vermeeren, and J. Kort, "Understanding, scoping and defining user experience," in *Proceedings of the 27th international conference on Human factors in computing systems - CHI 09*. Boston, MA, US: ACM, 2009, pp. 719–728. [Online]. Available: http://dl.acm.org/citation.cfm?doid=1518701.1518813

[24] M. Hassenzahl, "Experience design: Technology for all the right reasons," *Synthesis Lectures on Human-Centered Informatics*, vol. 3, no. 1, pp. 1–95, Jan. 2010. [Online]. Available: https://doi.org/10.2200/s00261ed1v01y201003hci008

[25] ——, "User experience (UX): towards an experiential perspective on product quality," in *Proceedings of the 20th international conference on Association Francophone d'Interaction Homme-Machine, IHM '08, Metz, France - September 02 - 05, 2008*. Metz, France: ACM, 2008, pp. 11–15. [Online]. Available: https://doi.org/10.1145/1512714.1512717

[26] H. Virpi , Roto Effie, Law Arnold, Vermeeren Jettie, "User Experience White Paper All About UX," p. 12, 2011. [Online]. Available: http://www.allaboutux.org/uxwhitepaper

[27] J. Forlizzi and K. Battarbee, "Understanding experience in interactive systems," in *Proceedings of the Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques, Cambridge, MA, USA, August 1-4, 2004*. Cambridge, MA, USA: ACM, 2004, pp. 261–268. [Online]. Available: https://doi.org/10.1145/1013115.1013152

[28] D. Travis and P. Hodgson, *Think Like a UX Researcher*, 1st ed. Boca Raton, FL: CRC Press, 2019.

[29] F. M. Newmann and D. A. Archbald, *The nature of authentic academic achievement*. New York, NY, USA: State University of New York Press Albany, NY, 1992.

[30] M. M. Gobble, "The Persistence of Brainstorming," *Research-Technology Management*, vol. 57, no. 1, pp. 64–67, 2014.

[31] W. W. J. S. Arno Delhij, Rini Van Solingen, "O Guia EduScrum - As regras do jogo," p. 21, 2016. [Online]. Available: http://eduscrum.nl/en/file/CKFiles/O_guia_eduScrum.pdf

[32] V. G. Ferreira, "Design sprint pbl model," https://github.com/neogedom/design-sprint-pbl-model, 2020.