A Practical Approach to Teaching Requirements Engineering in Computing Programs

Anderson dos Santos Guerra
Department of Exact and Technological Sciences
Federal University of Amapá
Macapá, Amapá, Brazil, email:
and.guerra@outlook.com

Julio Cezar Costa Furtado
Department of Exact and Technological Sciences
Federal University of Amapá
Macapá, Amapá, Brazil, email:
furtado@unifap.br

***Abstract*—This paper aims to contribute to the teaching of Requirements Engineering with a proposal of teaching approach that makes use of student-focused strategies for the development of skills expected by the industry. For this, the work presents the methodology used to create the approach through the identification of the skills expected by a requirement engineer. The results obtained through an experiment are presented, and it shows that the students were more motivated to research and to learn.**

Keywords-Software Engineering; Requirements Engineering; Software Engineering Education; Teaching Methodology.

I. INTRODUCTION

Based on study done by Menon et al. [11], although requirements engineering is an area of great importance and helps to avoid failure in systems development, its teaching still does not reach the expected performance of the industry. This occurs often because traditional teaching is used instead of dynamic teaching that prioritizes group activities and the use of creativity to solve problems.

Several researches [10][13][19] reports on the existing lack of qualified professionals to work in the software development industry, and one of the important factors that influences the quality of the professionals is education. This may indicate that the shortage of qualified professionals may be related to the required skills of a requirements engineer not being properly developed during education, making it difficult for the software industry to achieve the skilled workforce. A newly graduated professional will only effectively obtain the necessary knowledge when acting in the market. In this context, the objective of this work is to present a teaching approach to requirements engineering for computer courses that can prepare participants to understand how requirements engineering works in a real environment. For this, the Capability Maturity Model Integration for Development (CMMI-DEV) model [15] was used, which, among its good practices, offers recommendations for the Requirements Engineering process. Through the CMMI-DEV was created a list of skills expected by a professional in the area of requirements engineering, through this list were developed activities and dynamics used in the methodology for teaching Requirements Engineering.

In addition to this introduction, the article is structured as follows: Section 2 presents a review of the definitions of software engineering, requirements engineering and the education landscape of these two items; Section 3 talks about the teaching approach created, from the methodology used to elaborate to the activities used; Section 4 shows the survey created and presented to the students at the end of the course; in Section 5 the analysis of the answers obtained in the survey is made; Section 6 concludes the article.

II. BIBLIOGRAPHIC REVIEW

This session discusses a literature review of the concepts used to create the teaching approach, starting with Requirements Engineering and followed by Software Engineering Education.

A. Requirements Engineering

The area of Requirements Engineering has a great importance in the development of software because it offers a series of concepts that formalize an adequate elicitation and validation of requirements, ensuring that a certain system can adequately satisfy the needs of the client, reducing so the margin for errors, as well as costs that could be unnecessary and saving time [16].

One of the problems encountered in relation to requirements engineering is the communication barrier between developers and clients [18]. It is necessary to move away from traditional teaching and use more didactic means for teaching requirements engineering, such as Role Playing, which improve the use of communication in projects. The ACM / IEEE [1] states that the curriculum in the area of Software Engineering requires that the classroom training goes beyond the expository.

Since communication between stakeholder and developers may be flawed by several factors, such as the very difference of technical knowledge between the two parties, another problem that occurs is the flaw in the requirements documentation process, research [11] shows that this problem is caused by the lack of preparation of the professionals in the moment of action in activities involving the engineering of requirements.

B. Software Engineering Education

In order to find an opinion of professionals in Software Engineering in relation to an area that is approached in the courses of Computer Science, a research was made and there is a lack of attention to some topics of Software Engineering during a class [19], and these topics are taught insufficiently, highlighting the following: project

 ISBN: 978-1-61208-752-8

management, software quality assurance, requirements management, and requirements development.

Professionals in the field learn more about these topics at work than in the period of academic training [9]. Regarding the teaching strategy, the curriculum in the area of Software Engineering [1] emphasizes that there is a need to go beyond the expository classroom format. It is important to consider the variation of teaching and learning techniques.

The quality of the software engineering professionals is directly related to the quality of the education they have had, although there are other factors that may contribute to this [13]. The most common approaches to teaching software engineering include lectures, lab classes, with a greater focus on the teacher, as shown in the Table 1 [14].

TABLE I. COMPARISON BETWEEN TEACHER-FOCUSED CLASSES AND STUDENT-FOCUSED CLASSES

Characteristics	Teacher-focused	Student-focused
Teacher's role	Main provider of information; Specialist; Academic performance evaluator.	Facilitator; provides information to help in understanding the information.
Learning climate	Individualistic.	Collective; Focus on group cohesion.
Guidance	Based on the teacher's experience and knowledge.	Based on students' experience and knowledge.
Study program	Defined by the teacher.	Negotiated between teacher and students.
Teaching Objective	Defined by the teacher; default result.	Defined by the students; Different results for each student.
Knowledge Acquisition	Focus on acquisition; Focus on memorization.	Focus on the use and absorption of knowledge focusing on real problems.
Teaching methods	Didactic, Great participation of the teacher.	Methods involving student participation (dynamic techniques)
Focus on education	Educação individual.	Collective education
Evaluation	Performed by Teacher, Traditional Use of Tests and Grades	Students are also responsible for assessing

Research done in [14] states that there are teacher-centered approaches to teaching and student-centered approaches, each with its own peculiarities. The more expository the class, the greater its tendency to be focused on the teacher. However, the more dynamism and practicality in the class, the greater the focus on the student

III. AN APPROACH TO TEACHING REQUIREMENTS ENGINEERING

This session explains how the teaching approach was created, from defining the methodology that was used to the practices that were used throughout the approach.

A. Methodology

In order to define the content to be taught by the approach, it was necessary to first identify which competences are expected by a professional working in the area, so that the teaching approach can focus on the acquisition and improvement of these skills by participating students. The CMMI-DEV model was used as a reference, since its processes are used as a basis in several areas [15].

The CMMI-DEV model contains a set of guides that covers content for the development of products and services. CMMI for Development includes practices that encompass process management, project management, systems engineering, hardware engineering, software engineering, and other processes used to assist in the development and maintenance of products [15].

A total of 10 competencies were identified for a professional to be able to act in the area of ER in a manner satisfactory to the industry. In CMMI-DEV, Requirements Engineering is seen in the Requirements Development process (in level 3 of the model) and in the Requirements Management area (in level 2 of the model), in Table 2 it is possible to observe the skills and abilities.

TABLE II. EXPECTED SKILLS OF A REQUIREMENT ENGINEERING PROFESSIONAL, BASED ON CMMI-DEV

Skills	CMMI-DEV
Elicit stakeholder needs and expectations for all phases of the product life cycle	RD SG 1 SP 1.1
Turn stakeholder needs and expectations into customer requirements	RD SG 1 SP 1.2
Establish and maintain product requirements, which are based on customer requirements.	RD SG 2 SP 2.1
Identify interface requirements	RD SG 2 SP 2.3
Establish and maintain a definition of required features and quality attributes.	RD SG 3 SP 3.2
Analyze requirements to ensure they are necessary and sufficient	RD SG 3 SP 3.3
Validate requirements to ensure that the resulting product will perform as intended in the end user environment	RD SG 3 SP 3.5
Develop understanding of requirements through who provides the requirement	REQM SP 1.1
Get commitment to requirements from project participants	REQM SP 1.2
Manage changes in requirements throughout the project	REQM SP 1.3
Maintain bidirectional tracking on requirements	REQM SP 1.4
Ensure project plans stay in line with requirements	REQM SP 1.5

The study from [13] was used as a basis for choosing the methods and resources used in this methodology, with the aim of developing the joint adoption of these items, through an iterative cycle that attends the different learning profiles. The teaching model created by Portela [13] is supported by the learning cycle of Kolb [8] and the iterative teaching methodology proposed by Gary et al. [4]. Based on this, the model is aligned primarily in the reading of articles and reports of experiences, practical case discussions, use of simulators and games, besides the execution of projects and reflection by the student about the content that was learned and the exercises that were performed. The course was designed to be run over a semester as an optional topic in the Computer Science course for students who had already taken the basic discipline of Software Engineering, so that they

 ISBN: 978-1-61208-752-8

already had the necessary basics. The discipline had a workload of 60h, used by means of 4h weekly.

Table 3 lists the planning of the discipline, where it is possible to observe: the programmatic content, attending the competences planned for the discipline; the teaching strategy, established according to the level of learning that has been estimated for the topic and the expected outcomes; the expected results, what the student can do after the study of the unit; and the level of learning, using a terminology based on Bloom's [20] taxonomy, which consists of knowledge, understanding and application, in which: Knowing means remembering the material that was previously taught; Understanding refers to understanding the information and meaning of the material that has been taught; and Apply, indicates knowing how to use the material learned in new situations.

The first two topics in the discipline agenda, introduction to requirements engineering and requirements discovery, are based primarily on the CMMI-DEV Requirements Development process, while the third topic is mainly based on CMMI-DEV Requirements Management process.

TABLE III. DISCIPLINE AGENDA

Content	Teaching Strategy	Expected results
1. Introduction to Requirements Engineering		
1.1. Course Presentation	Card dynamics for product creation.	The student should be aware of the yearnings, stress and difficulty of project execution when requirements are incorrectly collected.
1.2. The Importance of Requirements Engineering	Reading and class discussion of the support material.	The student should be able to know the importance of the Requirements Engineer for software product quality.
	Game: Quantum Software	
1.3. The Requirements Engineering Process	Reading and class discussion of the support material.	The student should know the steps, roles and activities involved in the requirements engineering process.
	Game: A Ilha dos Requisitos	
2. Requirements Discovery		
2.1. Activities involved	Reading and class discussion of the support material.	The student should be able to understand the relationships between the activities developed in the requirements discovery.
	Requirements Discovery Dynamics	
2.2. Main difficulties	Reading and class discussion of the support material.	The student must understand the main difficulties of requirements discovery and be able to develop ways to mitigate these problems.
	Seminar on the difficulties encountered in the previous dynamics	
2.3. Techniques	Reading and class discussion of the support material.	The student should be able to understand the various requirements discovery techniques.
	GameMaker Dynamics	
3. Specification and Documentation		
3.1. Requirement Types	Reading and class discussion of the support material.	The student must know and identify the types of requirements.
3.2. Ways to document requirements	Reading and class discussion of the support material.	The student should be able to understand the various requirements documentation techniques.
	GameMaker Dynamics Continuation	
4. Final project		
4.1. Practical project	Customer Interview Dynamics	The student must be able to apply the knowledge gained throughout the course and perform the entire requirements engineering process (with teacher supervision)

B. Practices Used

1) Card Dynamics: The dynamics of the cards for product creation are as follows: the class is divided into groups, each group must create the same product based on the requirements requested by the client (the teacher). Throughout the class the client adds or removes a requirement, forcing the groups to make the necessary adjustments to the prototype. In this way, it is possible to provide participants with an initial view of the volatility of the requirements, of the difficulties caused by poor collection of these requirements and the consequences of this throughout a project. The dynamics of the cards were designed to be executed in 45 minutes, with the level of learning classified in knowing;

2) Game: Software Quantum: Quantum Software [7] is a game created to run on the web and simulates the requirements engineering process. It was created with the intention of offering a model that is simple and easy to learn in a few minutes, but at the same time has the ability to convey the main ideas contained in Requirements Engineering. The game does not necessarily cover all the dynamic aspects of software projects, but focuses on the tension between developing a system correctly and developing the right system for the customer [7]. The dynamics with the use of Quantum Software was designed to be executed in 45 minutes, with the level of learning classified in knowing.

3) Game: Ilha dos Requisitos: The game Ilha dos Requisitos [17] has a story in which the protagonist suffers an accident on a trip and ends up on an unknown island. The main objective of the game is to help the protagonist escape from the island along with members of the Nerds tribe before the island is destroyed by a volcano. Throughout the game are presented seven challenges that help the player to better understand the concepts and the requirements engineering process. Throughout the challenges the player can receive immediate feedback of his actions as tips to

 ISBN: 978-1-61208-752-8

remember concepts that are related to requirements engineering and results that were obtained at the end of each challenge. A feature of the game is that it seeks to engage the participant in a situation analogous to situations that could be solved through practices related to Requirements Engineering. The challenges are Requirements Engineering Process: the player must order the phases of the requirements engineering process; Validation of requirements: it is necessary to take the requirements to the client so that they are validated before the actual execution of the project; Role of the requirements analyst: the player must correctly identify the skills of a requirements analyst; Analysis of the problem: it is necessary to distinguish between the problems and their possible solutions; Requirements specification: the player must perform the classification of requirements between functional or non-functional; Classification of requirements: it is necessary to classify the requirements presented as functional or non-functional; Requirements Management: The player must order the process of changing the listed requirements and identify the activities that are part of the requirements management. The dynamics with the Game Ilha dos Requisitos was designed to be executed in 30 minutes, with the level of learning classified in knowing.

4) GameMaker Dynamics: GameMaker is a proprietary tool used for game development. The use of GameMaker is based on the teaching of Software Engineering through Game Design [6], which has a fun factor that can engage students to participate more actively in teaching. To perform the activity, the students were instructed so that the classroom teacher is the client and the students would be part of a development team, the client requested requirements to be implemented in the product (the game that was being created in the GameMaker tool) , throughout the activity the participants were able to experiment with a collection of requirements and its implementation in the product under development, as well as to experience the difficulties caused by communication failures and consequently in the collection of requirements. The dynamics with the use of GameMaker was designed to be executed in two 50-minute classes, with the level of learning ranked in understanding

5) Practical project: In the Practical Project, the participants created "companies" that were supposed to interview an external customer to create a product, with weekly meetings and prototype presentations to verify that the project was in line with the client's needs. the requirements engineering process and all the good practices that have been learned throughout the course. The client was the coordinating secretary of the Computer Science course where the main problem that the groups needed to solve was the systematization of the selective process of an extension project. At the end of the project, the groups presented the final software for the client, developed from the requirements, which made the choice of what best served their needs and was put to use by the extension project. The practical project was planned to run over four weeks, with the apprenticeship level.

IV. RESULTS

For the evaluation of the teaching methodology it was used a survey, the target population is characterized by students of computer courses that have subjects related to Software Engineering, the specific group attends the fourth semester of Computer Science in a public institution of teaching. Regarding the design of data collection, it can be considered crosscutting, since the participants inform data related to their past experiences.

The survey used quantitative and qualitative data about the students who participated in the experiment, regarding their individual information and preferences. Thus, for the data collection, a questionnaire was used consisting of objective and subjective questions.

Responses were received from 22 students, all in the undergraduate degree in Computer Science at the Federal University of Amapá, the participants were in the fourth semester of the course and already had a base due to the discipline of Software Engineering that had previously been studied.

A. The Survey Questions

The study aims to answer two research questions:

- **Research Question 1 (RQ1):** Was a teaching approach used during the course appropriate to the relevance of content and teaching methods?
- **Research Question 2 (RQ2):** What are the strengths and weaknesses of the teaching approach used?

Based on the references cited in the previous section, the survey questions were defined. Table 4 shows the questions created for the participants of the experiment in order to assess whether the teaching approach used during the course was adequate in relation to the relevance of the content and teaching methods.

TABLE IV. RESEARCH QUESTION 1

Questions	Answer Options
RQ1.1. The content covered by the course was sufficient to understand how Requirements Engineering works in an organization.	Answer performed by the likert scale: • Strongly disagree • Partially Disagree • Neutral • Partially agree • Totally agree
RQ1.2. The approach chosen for the discipline had a good integration of theory and practice.	
RQ1.3. The dynamics / practices were performed in a timely manner.	
RQ1.4. The dynamics / practices had an appropriate level of complexity.	
RQ1.5. The dynamics / practices developed did not restrict students' creativity to think about their own solutions.	
RQ1.6. The dynamics / practices	

made the learning process fun and challenging	
RQ1.7. Throughout the course, the teaching approach kept me motivated to learn	
Formulation: Frequency of distribution of each group variable RQ1	

V. RESULT AND ANALYSIS

In this section, there will be presented the data obtained in the survey that was answered by the academic participants of the discipline proposal.

Regarding the adequacy of the approach to software engineering education (RQ1), relative to RQ1.1 (The content covered by the discipline was sufficient to understand how Requirements Engineering works in an organization), participants responded with a 63.7% approval rate. As to the verification if the approach used in the discipline had a good integration of theory with practice (RQ1.2), 86.3% of the students who participated in the approach responded positively. Taking into account RQ1.3 (The dynamics / practices were carried out in a timely manner), the result was a positive response of 77.3%.

As for RQ1.4 (Dynamics / practices had an adequate level of complexity), 77.3% of respondents gave a positive answer to the question. Regarding RQ1.5 (The dynamics / practices developed did not restrict the students' creativity to think about their own solutions), the question had 63.7% positive answers. Considering RQ1.6 (The dynamics / practices made the learning process fun and challenging), the question is 86.4% approved. Based on RQ1.7 (Throughout the course, the teaching approach kept me motivated to learn), the question had a total of 81.9% positivity

A. *About teaching approaches*

In terms of teaching, 77.3% of respondents said they feel motivated to learn more about Software Engineering content. Regarding the teaching approach, 81.9% of the participants state in full or in part that they felt motivated to learn more about requirements engineering due to the dynamics used in class, it is possible to observe the response graph in Figure 1.

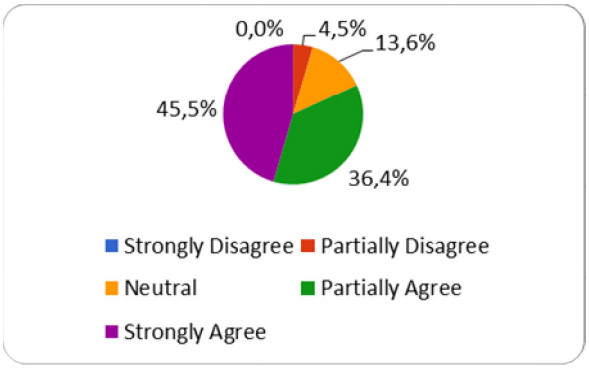

Figure 1. I am motivated to learn more about requirements engineering

This shows that the dynamics adopted throughout the course motivate students to learn more about the content. Added to this, another important fact is that 86.4% of students fully or partially agree that the dynamics / practices have made the learning process fun and challenging, as shown in Figure 2 with the response data.

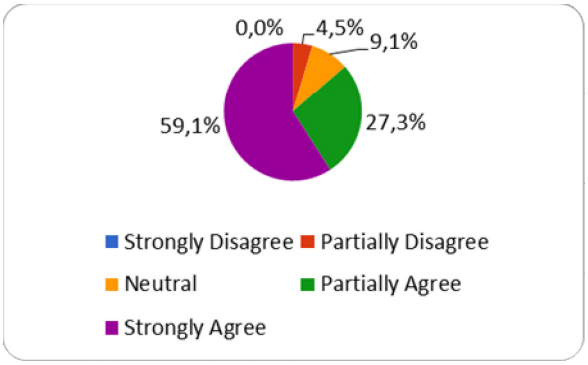

Figure 2. The dynamics / practices made the learning process fun and challenging

From these results, it was possible to show that students prefer more practical teaching approaches that motivate them to practice the content of software engineering, besides the incentive through practices, it is also noticed that through this approach the students can fix a concepts in this area, in addition to fostering learning and participation, as seen in the results of teaching strategies highlighted by [3][13][14]. The dynamics along with the hands-on activities reflect the interest of students who are more motivated to learn more about the subject. This reveals that the Requirements Engineering discipline is much more understood from practical situations than from the conventional teaching mode with only theoretical classes.

Still analyzing the data concerning the teaching approach chosen for the course, it is possible to observe in Figure 3 that 86.3% of the participants totally or partially agreed that the approach chosen for the course had a good integration of theory and practice. This further corroborates how more practical approaches are most effective in the students' learning process in the software engineering discipline.

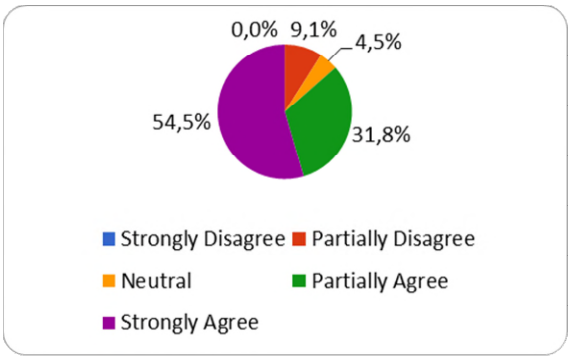

Figure 3. The approach chosen for the discipline had a good integration of theory and practice.

 ISBN: 978-1-61208-752-8

Turning to the analysis of the data obtained through the survey, it is noteworthy that 78.3% of the course participants totally or partially agree that the content taught in the course was relevant, the answer graph can be seen in Figure 4. This is explained by the fact that students, through requirements engineering dynamics and practices, have really understood the real relevance of content. Because they had to work in a reality-simulating environment, they were able to see how crucial this phase is and what it has on the entire development of software, with students better understanding requirements engineering concepts in a number of ways. practical classes significantly lower than theoretical classes, a result similar to that obtained by [3][13], in the teaching strategy that involves the use of recreational activities and games in general.

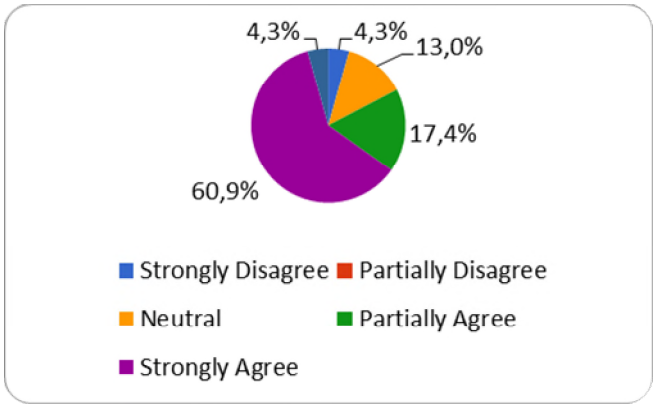

Figure 4. The content taught in the course was relevant.

From this deeper understanding of the content, participants were more aware of how requirements engineering is important in an organizational setting. This was evident when participants were asked if the content covered by the course was sufficient to understand how Requirements Engineering works in an organization, and the answer was that 63.7% of participants fully or partially agreed with this statement while only a total of 9 % disagree with the statement in some way, the remaining 27.3% goes to those who were neutral about the statement, as shown in Figure 5, showing that as much as the experiment used practices that resembled the operation of software engineering In one organization, a considerable number of participants could not confirm this similarity as they did not have the experience of working in the area. Although not such a high percentage, it is still a satisfactory percentage given that understanding how an organizational environment works is not a simple task and also takes some time.

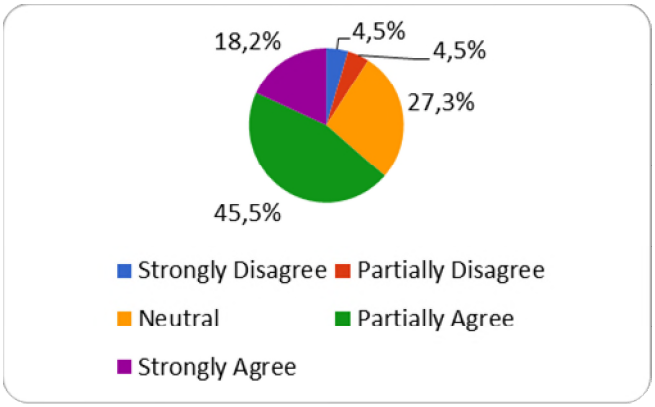

Figure 5. The content covered by the course was sufficient to understand how Requirements Engineering works in an organization.

Figure 6 demonstrates participants' satisfaction by comparing two statements: (i) I am motivated to learn more about requirements engineering; (ii) the dynamics / practices have made the learning process fun and challenging. It can be seen that most students considered the activities appropriate and were motivated to delve deeper into the requirements engineering area.

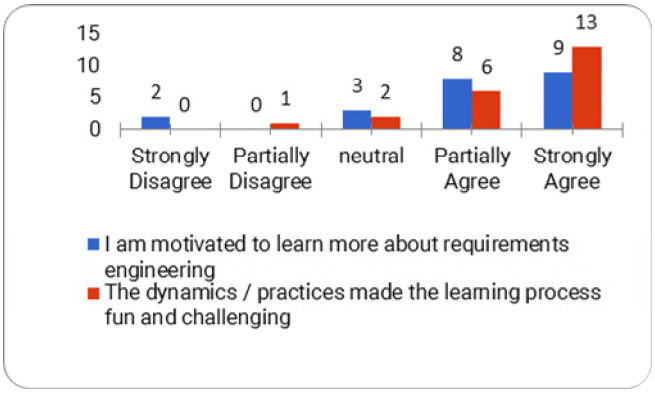

Figure 6. Relationship between learning motivation and learning process to be fun

The main finding with regard to the negative points was not about the approach itself, but about the short time given to the practical activities and theory, given that the experiment lasted one semester. When asked if the dynamics / practices were performed in a timely manner, 77.3% agreed totally or partially with this finding. Even with the short course period being short, the pass rate is still quite satisfactory. Figure 7 shows the response data.

 ISBN: 978-1-61208-752-8

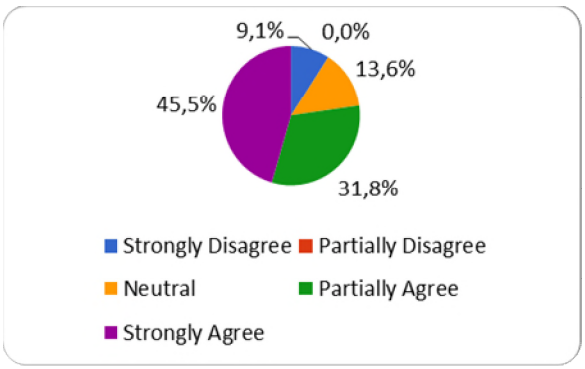

Figure 7. The dynamics / practices were performed in a timely manner.

Another point analyzed by the survey regarding the teaching approach shows us that 77.3% of the participants totally or partially agreed that the dynamics / practices had an adequate level of complexity, this is an important factor, since most students already had had some contact with the subject, mainly through the software engineering discipline, in Figure 8 it is possible to observe the answers given by the students.

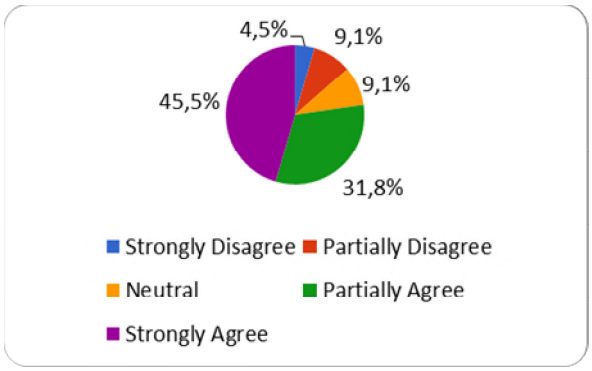

Figure 8. The dynamics / practices had an appropriate level of complexity.

Based on the survey, 63.7% of the participants of this experiment totally or partially agreed that the dynamics / practices developed did not restrict students' creativity to think about their own solutions, which was another very positive point of the approach adopted, as participants could make use of their own ideas for solving some data problems, data relating to this issue can be viewed in Figure 9.

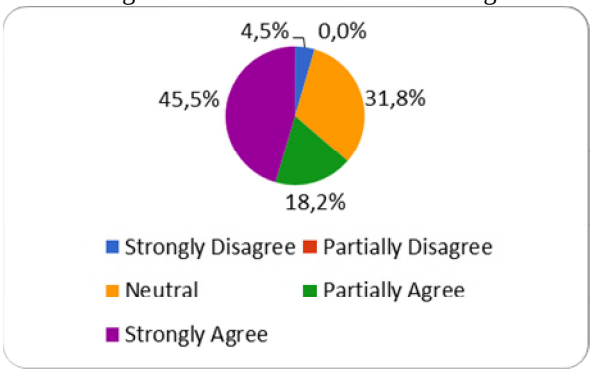

Figure 9. The dynamics / practices developed did not restrict students' creativity to think about their own solutions.

B. *About teaching approaches*

One of the main things about conducting an experiment is knowing how valid its results are. The priority order of the validation types is performed through the objectives that the experiment has. The order of importance of validity applied is: internal, external, construction and conclusion.

1) Internal Validity: Internal validity is used to define whether the relationship between the treatment itself and the result obtained is causal rather than the result of a factor that has not been measured or cannot be controlled. In this experiment, some threats to internal validity can be described as the effect of maturation or even instrumentation. Regarding maturation, it may have occurred through studies and learning activities that were not planned in the scope of the experiment, through the participant's own initiative to search for more content. To alleviate this factor, the students were instructed not to look for questions that were not within the planned activities for the experiment. However there is no way to confirm that the instructions have been properly followed.

2) External Validity: Through external evaluation, conditions are defined that limit the ability to generalize the results obtained to other populations in other contexts. The context of the experiment was academic, so the overall results should be limited by the academic context. Another limiting factor in this regard is that the experiment was conducted with a small sample of participants and, to date, has not been replicated in other populations, groups or universities. The teaching approach used tried to insert the student into the practical application of the concepts learned, however the practical activities were based on simplified real scenarios, so there is no guarantee that the skills obtained will be reflected in a real development environment.

3) Construction Validity: Construction validity considers the relationship between theory and observation, that is, it verifies whether the treatment reflects the cause satisfactorily and the result reflects the effect satisfactorily. During the evaluation of the construction validity, human factors and aspects relevant to the design of the experiment should be highlighted. Thus, it is not possible to state that the participants learned to use this new knowledge in a different context from the one used in the experiment.

4) Validity of Completion: The validity of completion is the ability to reach a correct conclusion about the relationship between treatment and the outcome of the experiment. During the evaluation of this quality it is necessary to take into account concepts such as: the reliability of the measurements; the reliability of the treatment implementation and the choice of the statistical test. Even when grouping all the data, the sample remains small, making it impossible to demonstrate any valid statistical relationship.

 ISBN: 978-1-61208-752-8

VI. CONCLUSION

This research aimed to analyze the strengths of a student-centered approach to teaching Requirements Engineering. Through a list of competencies obtained using the CMMI-DEV, an approach was created that seeks to achieve the competencies listed satisfactorily, making use of practical activities, dynamics, games and projects, so that at the end of the discipline, the student get a sense of how the requirements engineering process works in an organization.

According to the data obtained by the survey, it was shown that the approach taken in the short course tends to be effective. In addition to the theory, practical aspects of the discipline were approached, allowing the participants a great understanding and a better contact with the Requirements Engineering area. There is an interest of the students in practical and dynamic activities [3][13]. This behavior can also be verified through the results of the survey applied after the short course, because besides the interactions, the participants also felt motivated to study more the topics covered.

Although the results of the approach are significant, further research is needed to assess the actual learning gain that has been achieved by the proposal compared to other models used for teaching Requirements Engineering.

REFERENCES

[1] ACM/IEEE, Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science, New York, EUA, 2013.

[2] J. Carver, L. Jaccheri, S. Morasca, and Shull, F. “Issues in Using Students in Empirical Studies in Software Engineering Education”, Proceedings of the 9th Interna-tional Software Metrics Symposium, IEEE, 2003.

[3] J. C. Furtado and S. R. B. Oliveira, Evaluating Students’ Perception of their Learn-ing in a Student-Centered Software Engineering Course – A Experimental Study, In: 13th International Conference on Software Technologies, Porto, Portugal, 2018.

[4] K. Gary, T. Lindquist, S. Bansal, and A. Ghazarian, “A Project Spine for Software Engineering Curricular Design”, Proceedings of 26th Conference on Software Engineering Education and Training, 2013.

[5] C. Ghezzi and D. Mandrioli, “The challenges of Software Engineering Education”, Proceedings of the 27th international conference on Software engineering, St. Louis, MO, USA, pages 637-638, May, 2005.

[6] K. Claypool and M. Claypool., “Teaching software engineering through game design”, In Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education (ITiCSE '05). ACM, New York, NY, USA, 123-127, 2005.

[7] E. Knauss, K. Schneider, and K. Stapel, “A Game for Taking Requirements Engineer-ing More Seriously”. In Proceedings of the 2008 Third International Workshop on Multimedia and Enjoyable Requirements Engineering - Beyond Mere Descriptions and with More Fun and Games (MERE '08). IEEE Computer Society, Washington, DC, USA, 22-26, 2008.

[8] D. Kolb. “Experiential Learning: Experience as the Source of Learning and De-velopment”, NJ: Prendice-Hall, 1984

[9] T. C. Lethbridge, “What Knowledge is Important to a Software Professional”, Journal IEEE Computer Society Press Los Alamitos, CA, USA, pages 44-50, Volume 33 Issue 5, May, 2000.

[10] T. C. Lethbridge, J. Diaz-Herrera, R. Leblanc, and J. Thompson, “Improving software practice through education: Challenges and future trends”, Proceedings of the Conference Future of Software Engineering, Minneapolis, 2007.

[11] R. N. Menon, R. B. Ahmad, and S. S. Salim, “Problems in requirement Engineering education: a survey”, Proceedings of the 8th International Conference on Frontiers of Information Technology, 2010.

[12] P. Naur, B. Randel, “Software Engineering: Report of a Conference Sponsored by the NATO Science Committee, Garmisch, Germany, 7-11 Oct, 1968, Brussels, Scientific Affairs Division, NATO”, 1969.

[13] C. Portela, “Um modelo iterativo para o ensino de engenharia de software baseado em abordagens focadas no aluno e práticas de capacitação da indústria”, Centro de Informática, Universidade Federal de Pernambuco, Recife, 2017.

[14] R. Prikladnicki, A. Albuquerque, C. G. Wangenheim, and R. Cabral, “Ensino de Engenharia de Software: Desafios, Estratégias de Ensino e Lições Aprendidas”, 2009.

[15] SEI, CMMI® for Development, version 1.3, CMU/SEI-2010-TR-033 ESC-TR-2010-033. Software Engineering Institute-SEI, Carnegie Mellon University: 561, 2010.

[16] I. Sommerville, Engenharia de Software, 9ª ed., Pearson Education do Brasil, 2011.

[17] M. Thiry, A. Zoucas, and R. Gonçalves, “Promovendo a Aprendizagem de Engenharia de Requisitos de Software Através de um Jogo Educativo”. Simpósio Brasileiro de Informática na Educação, Brasil, 2010.

[18] D. Zowgui and S. Paryanim, “Teaching requirements engineering through role playing: lessons learnt”, Proceedings. 11th IEEE International Requirements Engineering Conference, Monterey Bay, CA, USA, 2003.

[19] C. G. Wangenheim and D. A. Silva, “Qual conhecimento de engenharia de software é importante para um profissional de software?”, 2009, Fórum de Educação em Engenharia de Software, Fortaleza.

[20] B. S. Bloom, Taxonomy of Educational Objectives: The Classification of Educational Goals, Handbook I, Cognitive Domain: Longmans, 1956.

 ISBN: 978-1-61208-752-8