# Teaching software maintenance with ludic techniques supported by Robotics

Hellen Christine Seródio Thomazinho, Alexandre L´Erario, José Augusto Fabri
Universidade Tecnológica Federal do Paraná - UTFPR
PPGI - Programa de Pós Graduação em Informática
Cornélio Procópio – PR - Brasil
{hellen.serodio}@gmail.com, {alerario, fabri}@utfpr.edu.br

***Abstract* - This article demonstrates the application of playful techniques with the use of robotics in teaching software maintenance. In the area of Computing, there are several initiatives that propose the use of robots as an object of learning, having as main function to increase students 'motivation and commitment in the development of the taught discipline, as well as to stimulate students' creativity due to their dynamic nature, interactive and even playful. Four experiments were applied to validate the effectiveness of the technique presented in this work, in two undergraduate courses with the purpose of measuring the learning ability of the team to apply processes and activities related to software maintenance. The results indicated that the use of play techniques associated with robotics in the teaching process was relevant for the motivation, satisfaction and consolidation of concepts related to software maintenance.**

***Keywords - software maintenance, Lejos, Mindstorms.***

## I. INTRODUCTION

The software development lifecycle consists of a series of phases: planning, analysis and specification of requirements, design, implementation, testing, delivery and deployment, operation and maintenance. Upon completion of its development, the software is delivered to the customer and regardless of application domain, size and complexity, will continue to evolve over time. In the scope of software changes occur when errors are corrected, when there is adaptation to a new environment, when the client requests new improvements or new functions and when the application goes through a reengineering process to provide benefits in a modern context.

Sommerville [1] points out that the success of the operations of several organizations is directly linked to the proper functioning of the systems. It may be the activity that most consumes the organizations' efforts and resources, studies estimate that it represents about 80% of the total budget [2]. It is, therefore, an economically strategic activity during the life of any software. And for this reason it is salutary to include software maintenance knowledge in training IT professionals.

Within this context, this work aims to present the application of ludic techniques with the use of robotics in the teaching of software maintenance.

In the area of Computing, there are several initiatives that propose the use of robots as an object of learning [3], [4]. These objects have as main function to increase students 'motivation and commitment in the development of the taught discipline, as well as to stimulate the students' creativity due to their dynamic, interactive and even playful nature.

To validate the effectiveness of the technique presented in this work, two experiments were applied in two undergraduate courses (Software Engineering and Technology in Analysis and Systems Development) of UTFPR-CP. The purpose of the experiments was to measure the learning ability of teams to apply processes and activities related to software maintenance. In the experiments, the students solved exercises related to the maintenance of embedded software, whose purpose was to make the robot Mindstorms perform some activities.

The results indicated that the use of play techniques associated with robotics in the teaching process was relevant for the motivation, satisfaction and consolidation of concepts related to software maintenance.

The structure of this text includes a brief theoretical foundation, presented in section II. In section III the methodology applied to perform the experiments, in section IV the analysis of the results and in section V is presented the conclusion, followed by the references.

## II. THEORETICAL FOUNDATION

This session presents the theoretical basis used in order to provide a theoretical basis on the main concepts used in this work.

### *A. Software maintenance*

The software maintenance activity is characterized by the modification of a software product already delivered to the client, to correct any errors, performance

978-1-5090-5920-1/17/$31.00 ©2017 IEEE

improvements, or any attribute, or to adapt the product to a modified environment. Software maintenance can be defined as a part of the software development cycle [1][5]. In fact, it is not uncommon in a software organization to spend from 60% to 70% of all software maintenance resources [6]. Sommerville [1] points out that there are three different types of software maintenance: maintenance to repair defects in the software, maintenance to adapt the software to a different operating environment and maintenance to add functionality. A fourth category of maintenance is presented by some authors, such as Pressman.

- **Corrective maintenance**: Corrective maintenance refers to modifications required by actual errors in a software product [5].
- **Preventive Maintenance**: They seek to identify previously possible sources of problems in the software and to correct them in advance [5].
- **Adaptive Maintenance**: refers to adapting the software to its external environment, including modifications to implement new system interface requirements, new system requirements, or new hardware requirements [5].
- **Perfective Maintenance:** they aim to add new functionality features to the software, usually due to user requests [5].

### *B. Robotics an object of the learning*

Robotics as a technical discipline has emerged very frequently in school curricula. According to Leska [7], and Sullivan [8] there are indications that robotics has been an efficient learning object for teaching. Following are some works of different natures that represent the state of the art in the use of robotics in education.

Lew [3]state that the use of robots helps students better understand fundamental concepts and increases enthusiasm in teaching and learning activity. Vallim[4] uses robots and says that this is an appropriate approach to introducing engineering concepts and provides professional skills.

Brandt and Colton [9] used Minsdstorms to teach programming, mechanics and control to first year students of Mechanical Engineering at Brigham University. The objective of this work was to present the detection of an interface. The authors conclude that Mindstorms is a versatile platform that is widely accepted by students. With their use, the authors improved mechanics learning, sensor calibration, programming language and physics principles.

Fabri[10] applied Mindstorms to the teaching and learning of software processes and project management. To validate the effectiveness of the work, they carried out 8 experiments including university courses and companies in the software production sector. The experimental results proved that the process transfer of knowledge in processes and software management using Mindstorms led to greater consolidation, motivation, satisfaction of those involved.

However, it was not found any work that directs the use of robots specifically in teaching software maintenance.

## III. RESEARCH METHODOLOGY

The method used for the development of this work was the experimental one, because the experiment represents the best example of scientific research. According to Wohlin [11], the experimental research consists in determining a study object, selecting the variables that would be able to influence it, defining the forms of control and observations of the effects that the variable produces on the object.

The experiments were performed based on an execution plan divided into the following stages: Environment Definition, Subject Definition, Sample Definition and Experiment Execution. The research question, outlined by the work, that provided the mapping of the experimental method was:

- *It is possible the teaching of Software Maintenance, through playful techniques in robotics?*

### *A. Setting the Environment*

The experiments were carried out in an academic environment, the first one was carried out in the discipline of Software Project Management existing in the master's program of the Universidade Tecnológica Federal do Paraná (UTFPR) - Campus Cornélio Procópio - PR, whose research line is Software Engineering. And the second experiment in the discipline of Software Engineering of the undergraduate course in Technology in Analysis and Systems Development, on the same campus.

Students were randomly selected from the undergraduate and master's program to perform the experiments. Each experiment was followed without interference during its execution. The groups were divided as follows:

**Graduation:**

1-Experiment executed with group of 2 students (GROUP 1);

2-Experiment executed with group of 4 students (GROUP 2);

**Master's Program:**

1-Experiment executed with group with total of 4 students (GROUP 3);

2 -Experiment executed with 2 groups of total of 10 students (GROUP 4);

Each participant had access to a characterization form. This form aims at minimally identifying the professional who will participate in the environment. The form used in this

experiment can be obtained through the https://goo.gl/VlZtSP.

### B. Definition of subjects

In the master's degree the number of participants was 14 students in the 1st Experiment and 6 undergraduate students participated in the 2nd Experiment, divided by group.

### C. Sample definition

It is the amount of subjects who participated in the experiment. This information is entirely linked to the number of students. A total of 20 students participated in the initial experiment.

### D. Experiment Execution

The execution of the experiment aims to characterize the steps that the researcher will follow to carry out the experiment. The steps characterized for carrying out these experiments are:

1. Bring all the students together in a classroom;
2. Request that each professional fill in the Characterization form, containing Name, Participating Group, Age, course belonging, time of experience in Software Engineering, development in Java and Far.
3. Deliver the written consent form for signature, available at https://goo.gl/usWevJ.
4. Present to the participants the concepts of software maintenance and how the lego Mindstorms robot, presentation available at https://goo.gl/fDGKnG.
5. Ask the class to be divided into groups, and perform the environment setup, available at https://goo.gl/BmK7XB
6. After the environment has been configured, the groups should download the first version of the project available at: (https://github.com/alerario/LejosCircuit).
7. Each group received a maintenance service order, and this order contained the robot traveling a route more quickly
8. Provide for each group of participants the robot mindstorms for the execution of software maintenance for the purpose of making the robot perform a path correctly.
9. Define an estimated time to carry out the proposed activity.
10. Each group must deliver the source code for correction, make available in the repository.
11. After performing the maintenance and execution of the proposed activities, it will be necessary to fill in another questionnaire to evaluate the use of Mindstorms in software maintenance. This form will be available through https://goo.gl/RBZQKD.

The proposal for this experiment was to solve a bug in the path of the LEGO Mindstorms robot. It was a corrective maintenance so that the robot could execute a course identifying colors and following the final straight of a track. In addition to developing this maintenance, teams should consider software analysis and maintenance activities. One of them was to identify through a work order the problem to be solved, based on the documentation available. The other was running tests after the change in source code.

It is important to note that students did not have information about LEGO Mindstorms technology. Given this fact, one of the activities of this experiment was precisely to map this information to the performance analysis in the indicated problem. Figure 1 will illustrate the path to be performed by the robot in the experiments performed.

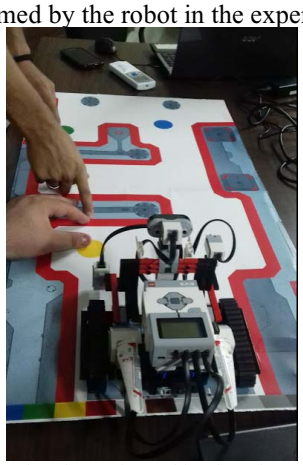

Fig. 1 - Path to be performed by the robot

The questionnaire applied after corrective maintenance of the software is composed of 21 questions, according to the table below:

TABLE I – QUESTIONS

| | |
|---|---|
| Q1 | The realization of maintenance was a simple task? |
| Q2 | Do you consider that your team had the ability to perform maintenance? |
| Q3 | You had ability to relate / meet the user? |
| Q4 | The allocated time was sufficient to perform maintenance required? |
| Q5 | You had to make a training with a colleague? |
| Q6 | In your group a person has been set to perform acceptance / review of maintenance performed? |
| Q7 | Do you think it is important for someone with the role to perform / accept maintenance? |
| Q8 | There was the identification of the problem to be solved? |
| Q9 | Did you consider it important to know the product before performing maintenance? |
| Q10 | Was the user knowledge necessary to maintain the product? |
| Q11 | Did you consider it important to know the product business rule before performing the maintenance? |
| Q12 | Did you have difficulty setting up the required maintenance development environment? |
| Q13 | Was the time to start maintenance development adequate? |
| Q14 | Within the stipulated deadline, do you agree that you |

| | |
|---|---|
| | did not spend much time getting to know the product? |
| Q15 | You have performed the procedures (processes) maintenance as stipulated? |
| Q16 | Did you consider it important to know the software architecture before performing maintenance? |
| Q17 | Did you consider it important to know the libraries used by the product before performing the maintenance? |
| Q18 | Did you consider it important to know the business rules of the product before performing the maintenance? |
| Q19 | Did you consider it important to know the IDE before performing the maintenance? |
| Q20 | Was it possible to identify the problem through the Maintenance Request (Service Order)? |
| Q21 | At some point did you need to contact the user who opened the Service Order for clarification? |

## IV. ANALYSIS OF RESULTS

After the execution of the experimental plan, the authors collected 20 questionnaires, which were tabulated in two worksheets so that their results could be analyzed. The first worksheet characterizes the answers of all the questionnaires by the graduation course. The second worksheet, characterizes the achievement of the percentage obtained in each response.

In possession of the spreadsheets, the authors prepared 4 graphs. Spreadsheets and charts can be obtained through the link https://goo.gl/9r4soK.

The options for answers to the questions that make up the questionnaire are characterized by the Likert scale [13] (1 - Strongly Disagree, 2 - Partially Disagree, 3 - Indifferent, 4 - Partially Agree and 5 - Strongly Agree).

### *A. Results of the two experiments applied graduation.*

At the undergraduate level, two experiments were carried out in the last module of the course, with a total of 6 students. The first experiment contained 2 students, presented in the work as Group1 and the second experiment 4 students, presented in the work as Group2. Both had the same experience in Software Engineering (Less than 1 year) demonstrated in the graph of Fig. 2.

Analyzing the chart of the Fig. 3 it is possible to notice that the Q1 to Q21 legends of the X axis represent the questions answered in the software maintenance questionnaire. In the Y axis are presented the total percentage answered by all participating students, represented with the options of answers based on the Likert scale.

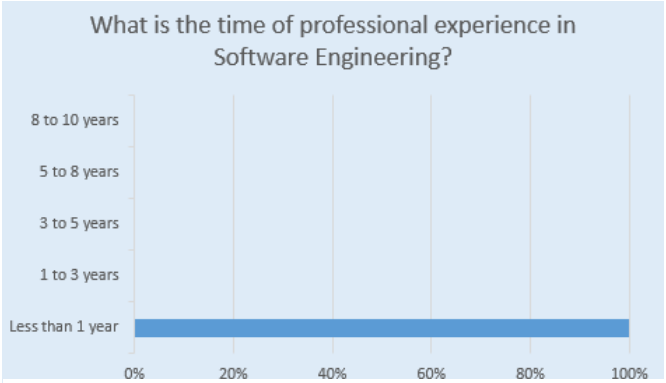

Fig. 2 - Experience in Software Engineering

In the analysis of group 1, it is concluded that question 1 (Q1), all participating students fully agree that maintenance was a simple task, in (Q2) all participating students fully agree that the team had the ability to perform Maintenance and ability to service the user (Q3). In relation to the time being sufficient to carry out the activities (Q4), all agreed fully that it was sufficient. In question (Q5), all disagreed fully that there was no need to conduct training with the colleague for informed maintenance. Regarding (Q6), 50% disagree fully that no person was defined to perform maintenance acceptance / review, and 50% dislike this definition. Regarding this role (Q7), 50% agree partially and 50% find it indifferent to have someone to play this role. All students have fully agreed that the problem has been identified (Q8) and everyone agrees fully that it is important to know the product to perform maintenance (Q9). Regarding the user's knowledge that it was necessary to maintain the product (Q10), 50% fully agree and 50% disagree partially. In regards to consider important to know the business rule of the product, before performing maintenance (Q11), all agreed fully. Regarding the difficulty in setting the environment to perform the requested maintenance (Q12), 50% fully agree and 50% found it indifferent. About the time to start the development of maintenance have been adequate (Q13), 50% fully agree and 50% find it indifferent. All agreed fully that they did not spend much time getting to know the product (Q14) and fully agreed that they performed the maintenance processes as stipulated (Q15). Regarding the importance of knowing the software architecture before performing maintenance (Q16), 50% fully agree and 50% partially agree and 50% fully agree and 50% disregard knowledge of the libraries used by the product, before performing maintenance \(Q17). About knowledge of product business rules, before performing maintenance (Q18), 50% partially agree and 50% fully agree. Regarding the importance of knowing IDE before performing maintenance (Q19) 50% fully agree and 50% disagree partially. Regarding problem identification by order of service (Q20), 50% partially agree and 50% fully agree, and students reported in 50% partially agreed that they had to contact the user who opened the service order (Q21) And 50% disagreed partially that they did not need to contact.

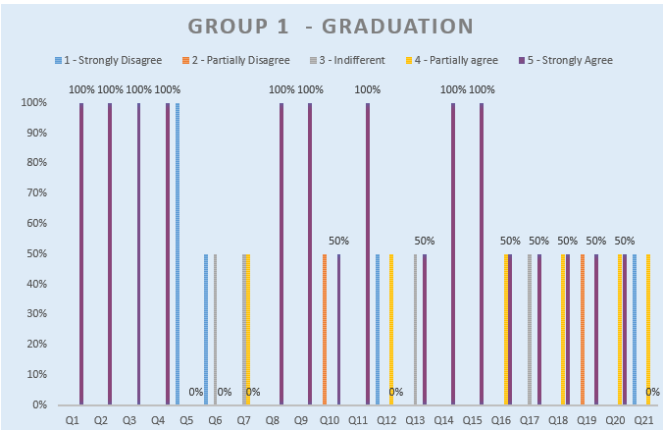

Fig. 3 - Group 1 Graduation

In the analysis of group 2, it is concluded that question 1 (Q1), 50% of the participating students fully agree that maintenance was a simple task and 50% disagree partially, in (Q2) 75% of the students fully agree that the staff had ability to perform maintenance 25% disagree partially. Regarding the ability to serve the user (Q3) and 75% fully agree and 25% partially agree. About enough time to carry out the activities (Q4), all agreed fully that was enough. Regarding the need to have a training with a colleague (Q5), 25% disagree fully, 25% disagree partially, 25% find it indifferent and 25% partially agree. Regarding (Q6), 50% strongly disagree that a person was not defined to perform maintenance acceptance / review, 25% find this definition indifferent, and 25% disagree partially. Regarding this role (Q7), 75% agree fully and 25% partially agree to have someone to play this role. All students fully agreed that the problem was identified (Q8) and 75% fully agree that it is important to know the product to perform maintenance and 25% agree partially (Q9). Regarding the user's knowledge that it was necessary to maintain the product (Q10), 50% fully agree and 25% agree partially and 25% find it indifferent. Regarding to consider it important to know the product business rule, before performing maintenance (Q11), 50% fully agreed, 25% partially agreed and 25% disagreed partially. Regarding the difficulty in setting the environment to perform the requested maintenance (Q12), 50% partially agreed, 25% fully agreed and 25% partially disagreed. About the time to start maintenance development was adequate (Q13), 75% fully agreed and 25% found it indifferent. Regarding the time taken to know the product (Q14), 75% fully agreed and 25% disagreed partially. 75% fully agreed that they performed maintenance procedures as stipulated (Q15) and 25% partially agreed. Regarding the importance of knowing software architecture before performing maintenance (Q16), 25% fully agreed, 25% partially agreed, 25% found it indifferent and 25% disagreed partially and 50% disagreed partially, 25% fully agree and 25% agreed Knowledge of the libraries used by the product, before performing maintenance (Q17). About knowledge of product business rules, before performing maintenance (Q18), 75% partially agree and 25% fully agree and 25% disagreed partially. Regarding the importance of knowing the IDE before performing maintenance (Q19) 50% fully agree, 25% disagree partially. On the identification of the problem through the service order (Q20), all agreed fully that it was possible, and the students reported in 75% disagreed fully and that they needed to contact the user who opened the service order (Q21) and 25% They did not need to get in touch.

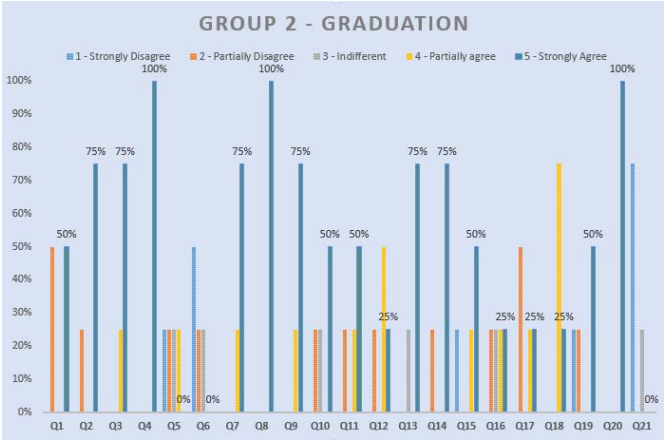

Fig. 4 - Group 2 Graduation

*B. Results of the two experiments applied to the master.*

In the master program two experiments were carried out, with a total of 14 students. The first experiment contained 4 students, defined in the work as Group 3 and the second experiment 10 students, defined in the work as Group 4. Both had the time of different experiences, as shown below:

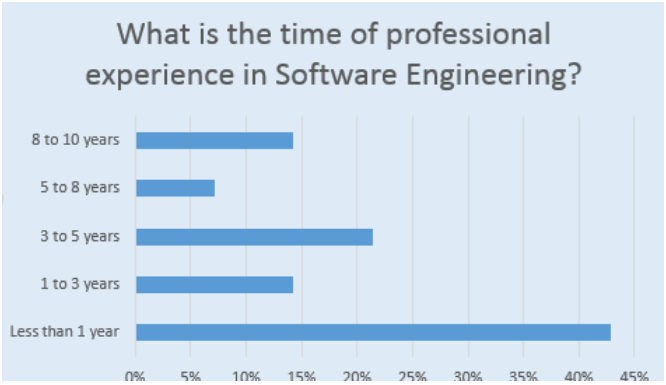

Fig. 5 - Experience in Software Engineering

In the analysis of group 3, it was concluded that question 1 (Q1), 50% of the participating students partially disagreed that maintenance was a simple task, 25% fully agreed and 25% partially agreed. In (Q2) 50% of the participating students fully agreed that the team had the ability to perform maintenance and 50% agreed partially. Regarding the ability to serve the user (Q3), 50% partially agreed, 25% fully agreed and 25% found it indifferent. As to the time being sufficient to carry out the activities (Q4), 50% fully agreed that it was enough and 50% partially agreed. In question (Q5), 50% fully agreed, 25% partially agreed, and 25% disagreed fully that there was no need to conduct training with the colleague for informed maintenance. Regarding (Q6), 50% find it indifferent that no person was defined to perform maintenance acceptance / review, and 25% agree partially and 25% disagree fully with this definition. Regarding this role (Q7), 75% agreed partially and 25% find it indifferent to have someone to play this role. All students fully agreed that the problem was identified (Q8)

and 80% fully agree that it is important to know the product to perform maintenance (Q9) and 20% disagree partially. Regarding user knowledge having been required for product maintenance (Q10), 75% fully agreed and 25% partially agreed. Regarding to consider it important to know the product business rule, before performing maintenance (Q11), 75% fully agreed and 25% partially agreed. Regarding the difficulty in setting the environment to perform the requested maintenance (Q12), 50% fully agreed and 50% partially agreed. Over time to start maintenance development was adequate (Q13), 75% fully agreed and 25% partially agreed. Regarding the time to know the product (Q14), 75% fully agreed that it did not spend much time and 25% partially agreed and 75% fully agreed that they performed maintenance procedures as stipulated (Q15) and 25% partially agreed. Regarding the importance of knowing the architecture of the software before performing maintenance (Q16), 75% fully agree and 25% agree partially and 75% fully agree and 25% defer knowledge of the libraries used by the product before performing maintenance (Q17). About knowledge of product business rules, before performing maintenance (Q18), 75% fully agree and 25% partially agreed. Regarding the importance of knowing IDE before carrying out maintenance (Q19) 50% fully agree, 25% partially agree and 25% disagree fully. Regarding problem identification by order of service (Q20), 50% fully agree, 25% partially agree and 25% disagree fully, and the students reported in 50% partially agreed that they had to contact the user who opened the order (Q21) and 25% strongly disagreed that they did not need to contact and 25% found it indifferent.

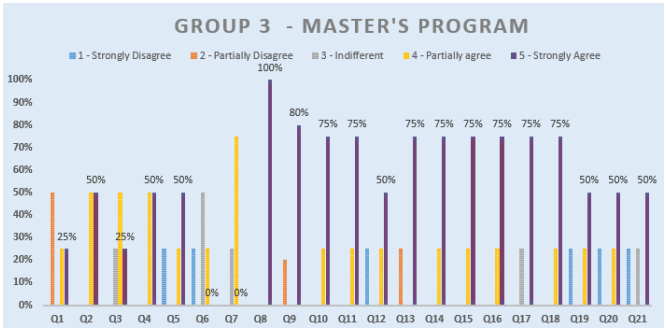

Fig. 6 - Group 3 Master's program

In the analysis of group 4, it was concluded that question 1 (Q1), 40% of the participating students fully agreed that maintenance was a simple task, 30% partially agreed, 20% disagreed fully and 10% disagreed partially. In the (Q2) 70% of the participating students fully agreed that the team had the ability to perform the maintenance and 20% partially agreed and 10% disagreed partially. Regarding the ability to serve the user (Q3), 80% fully agreed, 10% partially agreed and 10% disagreed fully. As to the time being sufficient to carry out the activities (Q4), 60% fully agreed that it was sufficient, 20% partially agreed, 10% disagreed partially and 10% disagreed fully. In question (Q5), 40% disagreed fully, 30% fully agreed, 20% partially agreed, and 10% found it indifferent that there was no need to conduct training with the colleague for informed maintenance. Regarding (Q6), 30% fully agree that a person was defined to perform the acceptance / review of maintenance performed, 30% disagree partially, 30% disagree fully with this definition and 10% find it indifferent. Regarding this role (Q7), 80% agreed fully, 10% partially agree, and 10% disagree partially with having someone to play this role. As to whether the problem was solved (Q8), 90% of the students agreed fully and 10% agreed partially. Regarding the importance of knowing the product to perform maintenance (Q9), 80% fully agree, 10% find it indifferent and 10% disagree partially. Regarding the user's knowledge that it was necessary to maintain the product (Q10), 60% fully agreed, 20% found it indifferent, 10% partially agreed and 10% disagreed partially. Regarding to consider it important to know the product business rule, before performing maintenance (Q11), 70% fully agreed, 20% partially agree, and 10% disagree partially. Regarding the difficulty in setting the environment to perform the requested maintenance (Q12), 50% partially agree, 30% fully agree and 20% disagree fully. About the time to start the development of maintenance have been adequate (Q13), 50% fully agree, 40% indifferent and 10% disagree fully. Regarding the time to know the product (Q14), 80% fully agree that it did not spend much time, 10% partially agreed and 10% indifferent. Regarding maintenance processes performed as stipulated (Q15), 60% fully agree, 20% partially agreed, 10% disagree partially and 10% disagree fully. Regarding the importance of knowing software architecture before performing maintenance (Q16), 50% fully agree, 20% find it indifferent, 20% disagree partially and 10% partially agree. Regarding the knowledge of the libraries used by the product, before completing maintenance (Q17), 50% fully agree, 20% disagree partially, 10% agree fairly, 10% disregard, and 10% disagree fully. About knowledge of product business rules, before performing maintenance (Q18), 80% fully agree and 20% partially agreed. Regarding the importance of knowing the IDE before carrying out maintenance (Q19), 60% fully agree, 20% disagree partially and 20% do not fully agree. Regarding problem identification through service order (Q20), 70% fully agree, 20% partially agree and 70% disagree completely that they had to contact the user who opened the service request (Q21) 20 % Fully agree, 10% indifferent.

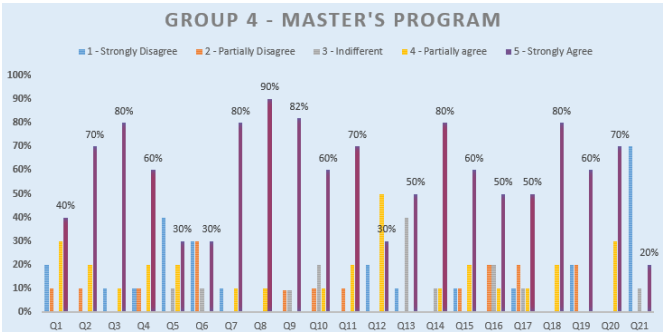

Fig. 7 - Group 4 Master's Program

After the analysis of individual form per group / experiment, the analysis by class (Graduation and Master), with the average of answers based on Likert, where it is possible to be observed through the table II, that the majority of the answers were positive after the Execution of the experiment.

TABLE II - ANALYSIS OF ANSWERS BY CLASS

| | Graduation | Master's Program |
|---|---|---|
| **Q1** | 5 | 2 |
| **Q2** | 5 | 5 |
| **Q3** | 5 | 5 |
| **Q4** | 5 | 5 |
| **Q5** | 1 | 5 |
| **Q6** | 1 | 3 |
| **Q7** | 4 | 5 |
| **Q8** | 5 | 5 |
| **Q9** | 5 | 5 |
| **Q10** | 5 | 5 |
| **Q11** | 5 | 5 |
| **Q12** | 4 | 4 |
| **Q13** | 5 | 5 |
| **Q14** | 5 | 5 |
| **Q15** | 5 | 5 |
| **Q16** | 4 | 5 |
| **Q17** | 5 | 5 |
| **Q18** | 4 | 5 |
| **Q19** | 5 | 5 |
| **Q20** | 5 | 5 |
| **Q21** | 1 | 1 |

In this experiment the LEGO Mindstorm robot proved to be an object of teaching and learning relevant in the teaching of software maintenance, and in this case also application, flexible and practical. In the literature, mindstorms is applied in the teaching and learning of computer programming, in the process of teaching and learning and in the management of software projects [10] and also being motivating in the teaching of software maintenance.The robot has easy-to-use (physical and logical) interfaces and students have quick and fun demonstration of results. These qualities encouraged the teams to perform maintenance, and when analyzing the answers applied to each question, it was possible to verify that the use of such an approach generated motivation and interest, being these essential factors for the success of teaching any subject.

## V. CONCLUSIONS

Software maintenance activity is poorly understood and easily confused with software development activity. The success of the operations of several organizations is directly linked to the proper functioning of the systems. The lack of knowledge of professionals in models and approaches that help in the process of maintenance of software in organizations, makes many systems have characteristics of legacy systems, therefore, it is necessary to ensure the availability of systems and the efficiency of service requests of software maintenance.

The present work, presented the results of 4 experiments applied in 2 courses (Undergraduate and Master's), the purpose of the experiments was to measure the learning ability of the teams in applying processes and activities related to software maintenance. In the experiments, the students solved exercises related to the maintenance of embedded software, whose purpose was to make the robot Mindstorms perform some activities.

The experiments presented the results described in section IV. In analyzing the results of the four experiments shown in Charts 1, 2, 3 and 4 and according to the table of questions (Table I), it is possible to conclude in both experiments that:

• The realization of maintenance was a simple task, teams had skills to perform the required changes and sufficient skills to meet the request of users.

• The time allotted for maintenance was sufficient, however much the students had to configure the environment to perform the proposed task.

• At the undergraduate level, students did not need to carry out a training with the colleague to understand the task, unlike the master's program, where it was possible to observe that many had experiences in software development, but had to train somebody else in the maintenance process.

• It was the first time they had contact with the product, even though it was a simple matter, they were afraid to update the main product in the repository. While agreeing that it is interesting to do the review on the product, they did not make anyone available for this at this stage.

• All agreed that it was possible to identify the problem through the service order, that is, the level of knowledge of the user who opened the service order was sufficient for the clarity of the proposed problem.

• Everyone agreed that knowledge of the business rule is necessary to perform the maintenance to be performed on the software.

• Both classes had difficulties in configuring the environment, that is, the user necessarily needs some knowledge in the programming language to perform any maintenance, that is, how the configuration will be performed without knowing the programming language.

• The students agreed with a whole to know the product, more performed the maintenance and did not worry about the other features of the system, just focusing on what was proposed.

In general, the teaching of software maintenance is linked to the activities below and was pointed out in this study as a positive means to follow, as shown in Fig.8.

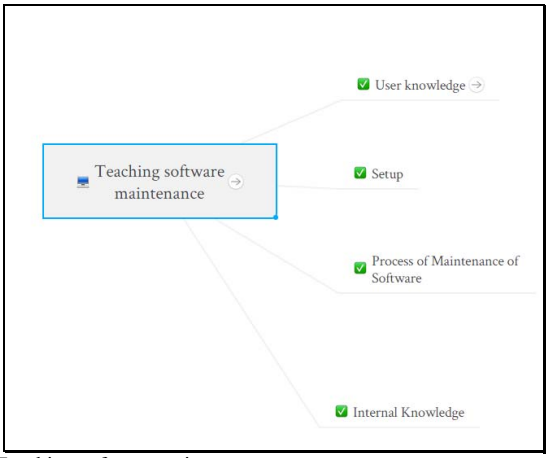

Fig. 8 - Teaching software maintenance

- User Knowledge: Knowledge of the user in the software acquired;
- Setup: Configure the environment for performing maintenance.
- Process of Maintenance of software: Planning of tasks and activities, documentation, management of demands, control of deliveries.
- Internal Knowledge: Know the business rules of the product, programming language, libraries, IDE.

The scope of this work was limited to application, although there are indications that the results are positive, the experiments will be replicated to confirm the truth of the facts.

## VI. FUTURE WORKS

As a proposal of future work, it is intended to perform more experiments and perform the maintenance with the same game environment in a more complex software and perform this comparison of results. It is also intended to improve the software and compare the experience with simple software without the robot.

## REFERENCES

[1] I. Sommerville, *Software engineering*. 2007.

[2] S. L. Pfleeger e J. M. Atlee, *Software Engineering: Theory and Practice*, 4º ed. 2004.

[3] M. W. Lew, T. B. Horton, e M. S. Sherriff, "Using Lego Mindstorms NXT and LeJOS in an advanced software engineering course", in *Software Engineering Education Conference, Proceedings*, 2010, p. 121–128.

[4] M. B. R. Vallim, J. M. Farines, e J. E. R. Cury, "Practicing engineering in a freshman introductory course", *IEEE Trans. Educ.*, vol. 49, nº 1, p. 74–79, 2006.

[5] I. Standard, *INTERNATIONAL STANDARD ISO / IEC Software Engineering — Software Life*, 2º ed, vol. 2006. 2006.

[6] R. S. Pressman, *Software Engineering A Practitioner's Approach 7th Ed - Roger S. Pressman*. 2009.

[7] C. Leska, "Introducing undergraduates to programming using robots in the general education curriculum", *ACM SIGCSE Bull.*, vol. 36, nº 3, p. 263, 2004.

[8] A. Sullivan e M. U. Bers, "Robotics in the early childhood classroom: learning outcomes from an 8-week robotics curriculum in pre-kindergarten through second grade", *Int. J. Technol. Des. Educ.*, vol. 26, nº 1, p. 3–20, 2016.

[9] A. M. Brandt e M. B. Colton, "Toys in the classroom: LEGO MindStorms as an educational haptics platform", in *Symposium on Haptics Interfaces for Virtual Environment and Teleoperator Systems 2008 - Proceedings, Haptics*, 2008, p. 389–395.

[10] J. A. J. A. Fabri, A. L'Erario, R. H. C. Palacios, e W. Godoy, "Applying mindstorm in teaching and learning process and software project management", *2015 IEEE Front. Educ. Conf.*, p. 1–8, 2015.

[11] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, e A. Wesslén, *Experimentation in software engineering*, vol. 9783642290442. 2012.

[13] R. Likert, "A technique for the measurement of attitudes", *Arch. Psychol.*, vol. 22 140, p. 55, 1932.