Web-based Multiplatform Development Environment for Educational Robotics

Sarah Thomaz[1] and Luiz Marcos Garcia[2]

***Abstract*— We created an online development environment, specifically for educational robotics applications, in which it is possible to register programming languages associated with different robotic platforms, so that the user program a robot using the registered language or the language R-Educ, developed for teaching programming. To validate the platform, we conducted a series of tests divided into six steps to verify that the complete cycle was satisfied - registering a language, program in R-Educ, compile to the registered language, compile to machine code and send the code for the robot.**

I. Introduction

For the development of educational robotics classes the following resources must be used in the classroom [1]:

- A robotic kit, consisting of a programmable logic controller, sensors, motors and parts for building a mechanical structure for the prototype robotic;
- A development environment;
- A computer capable of using the chosen environment, and
- An environment conducive to the development of a particular activity.

With the spread of educational robotics, several studies have been conducted in order to produce materials for educational robotics classes, to create new possibilities to the students. It is in this scenario that a search for a more versatile hardware is used in learning environments that make use of educational robotics. Linked to the use of different hardware is the use of softwares to program or control them.

The development environments accompanying educational robotics kits, mostly have different programming languages and different ways of programming: graphical or textual. The graphical programming interfaces are twofold: or approaching the maximum of the direct control of the hardware parts or try to abstract the details of hardware from the fitting of graphic content. The textual programming interface have, mostly, a programming language based on a given language, usually English, with keywords and command sequences differentiated.

For a program to be compiled and sent to a robot, it is often necessary to install specific software on a computer, limiting such operations to specialist users [2]. In some cases it is necessary that the user has system administration privileges to install and/or run software. Is also often required that the user have computer with high computing power or particular hardware specifications.

Due to the non-uniformity found in the forms of programming robots, the problems to install specific software on a given computer and the benefits that educational robotics has in the teaching and learning process, we developed a programming environment that allows greater flexibility in the use of hardware through a web development environment, allowing users to edit, compile and run programs using only a web browser. With the proposed solution, any device that has a web browser, can be used as development station to create programs for robotic prototypes.

The idea of this work is to create an environment where it is possible to program in multiple languages. The programming languages for robotics can be registered using a registration form, by an experienced user in the language, and the environment is configured to compile the code into machine language and send this code to the robot. From this work we make it possible for the user to program in a simpler programming language, called R-Educ [3] and send this program to any registered robot.

The development environment is based on the educational software R-Educ [4], which has programming environments in levels, in which students use different programming environments depending on their level of maturity and knowledge, in a total of five levels. The problem of this software is that it is used to program a particular robot. We created two of the five original levels, allowing the user to program different types of robots.

To validate the platform, we conducted a series of tests divided into six steps to assess the functioning of the platform records of languages and integrated development.

A. Web development environments

There are programming environments that enable that any computer or device with web access via a browser can be used to program robots without the need to install specific software. The Mindstorms Internet Control Environment (MICE) [5] is in this category, for being an environment for remote programming of robots. The physical robot in this case is not with the user, but located next to a server.

This web development environment, aims to solve the problem that not every student has the ability to have acquire a robot Lego Mindstorms NXT [6], with approximate cost of U.S.$350.00. This environment allows the user to program Lego robots remotely and watch the results via a webcam.

There are other studies in the literature that resemble this first, such as [7], [8], [9] but runs a bit out of the scope

[1]Sarah Thomaz is with Graduate Program in Electrical and Computing Engineering, Federal University of Rio Grande do Norte, Natal, Brazil sarahthom4z@gmail.com

[2]Luiz Marcos Garcia is with the Department of Computer Engineering and Automation, Federal University of Rio Grande do Norte, Natal, Brazil lmarcos@dca.ufrn.br

of this work, since they allow only remote programming of robots.

Another programming environment is the ADWN [10], a web development environment for robot programming, designed specifically for the robot N-Bot. It is an open source solution developed for mobile devices and traditional computers that dispenses the installation and configuration of any program and can be accessed by any device that has access to a web browser connected to the Internet. Came up with the proposal to be a simple and intuitive platform for beginners and to be a powerful and flexible tool for experienced users.

None of the program environments founds allows the user to program multiple robots. The environment developed in this work has such functionality, depending only on the registration of robotic prototypes.

II. Web environment developed

A. Design of the environment

The first step in the development of this work was design the environment. To this end, we conducted an analysis of the features that the environment should have to achieve the desired goals. Among the established features are: registration of programming languages, modification of the language R-Educ, environment for programming robots in any registered language, compilation of programs via server and sending the program to the robot via website. The following list summarize the actions performed:

1) Didatic restructuring language R-Educ;
2) Functional restructuring language R-Educ;
3) Restructuring the translator of the language R-Educ;
4) Creating a database for registration of programming languages;
5) Creating a web environment for registration of programming languages;
6) Creating an environment for web programming;
7) Defining a communication mechanism computer - robot.

B. R-Educ language

To work with robots, machines with specific operation, was necessary to create another programming language specification. This new specification has a specific purpose which is the programming of mobile robots. For such it became necessary a set of specific commands that allow the user to make the robot move as desired, send a light signal, etc.

Electronic equipment, in general, have large differences between their settings and can be applied to very different tasks, leading manufacturers of each robotics electronic device the development of their own language, or the use of a particular common language associated with specific libraries.

Motivated by the diversity of hardware and programming languages, we developed in this work an environment that configure any hardware with their distinct languages. It contains an universal language for any type of hardware for robotics. The language chosen for this is the programming language R-Educ [4].

inicio	fim
tarefa	se
senao	entao
enquanto	farei
repita	para
teste	numero
texto	booleano

TABLE I
RESERVED WORDS IN LANGUAGE R-EDUC.

The language R-Educ was initially developed in C++, making reading data via XML file content and designed to program robots like Lego RCX, Lego NXT [6] or H-Educ [11].

To use this language in this work was necessary to make a re-implementation of the language translator. We take the need for re-structuring required and implemented some improvements in the language, giving a greater computational power to it, correcting some errors found and introducing some concepts used in programming languages.

Table I shows the reserved words of the language R-Educ. The language has a few reserved words, which shows its high level of abstraction. Despite being based on a simple Portuguese, the language allows the teaching of algorithmic structures such as program start, program end, flow controls, conditional statements, calls to specific functions, etc.

One of the main modifications of the existing language R-Educ is that it was inserted symbols as '{' and '}' to begin and end flow control commands, replacing wordas as: endwhile, endif, endfor. The use of these symbols was based on the fact that we have seen in surveys with students that they have some degree of difficulty to assimilate this symbols so common in programming languages while migrating from R-Educ language to a language with a lowest level of abstraction.

C. Language registration

To define the necessary data to register a language, we analyzed the main components of programming languages. To this end, we conducted a selection among the main languages used to program robots or not.

In the preliminary study, we track if the language require headers and footers. Then evaluate what types of data were present in each language, such as string, integer, booleans, among other. We analyzed which were used the most and how variables are declared and modified.

We also evaluated how is performed the writing of the flow control structures and which are present in most languages. We also verified the declaration of methods and how these methods were called, and the declaration of the main function of the program. Another important data is regarding to logical operators and where they differ.

From this analysis we made a registration form for languages that allow us to translate a program written on the language R-Educ into a program written on the language

registered. At this stage of development we found that the user responsible for the registration of languages or responsible for the production of a register document must be a programmer with specific knowledge of this language or have programming skills sufficient to study the API language - Application Programming Interface - and extract information from it.

To make the register of native programming languages of different types of robots, we developed a web environment using specific APIs of Java EE. The developed environment runs online, allowing a greater spread of the tool.

We seek to leave the environment as explanatory as possible so that there are no errors in the registry. At the beginning of the registration form it is provided an explanatory text giving the user all the information required for the success of the registration. We also present examples of codes of all the information requested in R-Educ and in another language.

The set of data to be entered in the registration form languages are distributed as follows:

- Name and Description: information on the language, as name, robot to which it is associated and its description;
- Compilation and Dubmission: information about system calls for compilation and submission of the program and extension of files generated by the compiler;
- Header and Footer: codes that must be entered in the header and footer of all programs generated by the translator;
- Declaration of Functions: information on the main function and how methos are declared;
- Data Types: information on how data types are declared in the language;
- Operators: logical and relational operators of the language;
- Flow Control Structures: information about how flow control structures are written in the language;
- Functions: information about specific functions of the language, and how they are going to be called in R-Educ language.

D. Translation and compilation

It was necessary to create a translator for the configurable language R-Educ, generating the universalization of the language. Figure **??** shows that a language code generated in the language R-Educ, after processed by the translator designed, can be converted into a code in a language for robotics, such as NXC, Lejos, CCS, NQC or any other language registered.

In the first part of the translation is performed lexical analysis of the code written in R-Educ, in which is made a separation and store of language tokens. Then is called a mapping function that returns to the translator all data registered by the user in a set of words separated by the appearance of specific reserved words used during registration.

Finally, syntactic analysis is performed. It is made a check of the correctness of the sentences written by the user. Each sentence presented correctly is automatically translated into a code in the selected language. If the parsing is performed

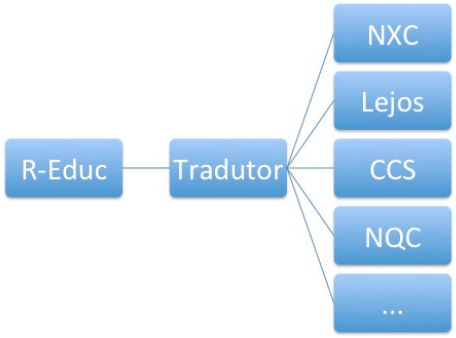

Fig. 1. Inductance of oscillation winding on amorphous magnetic core versus DC bias magnetic field

successfully a call is made to the compiler provided in the registration to complete the compilation process.

E. Communication and sending

The proposed configurable development environment implemented in this work not only provide the user with a development environment where you can program in various languages and also perform a translation between registered languages and R-Educ, but an environment that, in addition, allows the user to establish communications with various robotic platforms. Any device with access to a web browser - computers, laptops, tablets, etc. - can access the environment, which is connected to the Internet through a server. After this, the users can compile their program and, if they have resources capable of establish communication with the hardware, they can send a program to the robot.

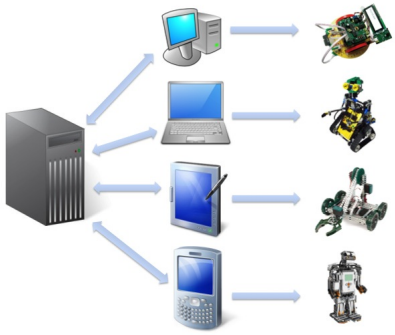

Fig. 2. Connection Diagram.

Figure 2 shows the connection diagram of the system. In this we can see that there is a server environment where the processing is done, the server establishes communication with computing devices, such as PCs, tablet, smartphone or laptop, via a web browser. These devices can latter communicate with robotic devices that have compatibility through data cable or via a wireless communication mechanism.

Sending data to the robotic device or execute the generated code is done via system calls provided by the user when registering a language. This system call is made via Java Applet.

For the execution of the applet the user must have the Java Virtual Machine installed and sign the digital certificate required for the applet to access and perform operations on the local computer. We emphasize that the communication between the computing device and the robotic device is required, and depending on the robotic platform used and the form of communication between them, it might be necessary a hardware compatibility, which would require the user to install specific drivers.

F. Development Environment

To perform programming in R-Educ and other languages registered by user, we developed a web environment using specific APIs of Java EE. The developed environment runs online, allowing a greater spread of the tool. Since the writing and compiling the code can be accomplished by any device with access to a web browser, all the computing capacity is only required to the server. Furthermore, the fact that the tool is web-based makes it unnecessary the installation of auxiliary programs.

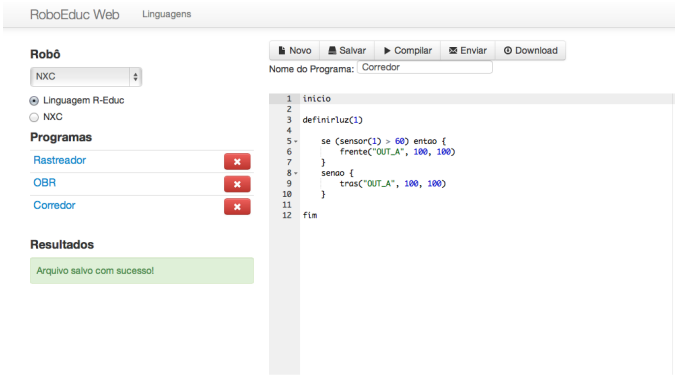

Fig. 3. Development environment.

The main screen of the environment is shown in Figure 3. The page has resources to save, compile, download and upload code in R-Educ and/or in the language previously registered, in addition to send a program to a robot. The left side has a list of the programs created for that language on the server, which can be selected for editing by the user.

The environment has on the left side an area for the results, where the users are informed if their code was successfully saved or deleted, if the file was successfully compiled, and if it was not, the environment reports the description of the error and in what line it is. In addition, if the program is successfully compiled, a program is generated in the language registered viewable by the user, so it is possible to make comparative studies between the written and generated code.

Still on the left side is found a list of robots that can be selected and changed by the user during programming without the need to reload the environment. If you perform the change of robot, the language associated with this will then be used in the translation and compilation of the final code.

Another feature of this programming environment is that if the language has been registered with errors, the code generated by the user will be successfully compiled in R-Educ, but the user will be informed that there was an error registering the language, since it was found an error in the final build.

III. EXPERIMENTS AND RESULTS

The experiments conducted were divided into six steps, involving register of language and the use of the programming environment and its functions. These steps evaluated the use of the tool at all stages of its development and use by experienced and inexperienced users in programming languages applied to robotics.

A. First step

The first stage of tests was carried out so that the development environment had its first finalized version. At this stage, we found that the programming language most used and known by the research group of laboratory NatalNet was the language NXC to program Lego Mindstorms NXT robots. Thus, we believed that this was the first language that should be registered and tested, since that if the tool only worked for that language, it would be a great contribution.

The registration of languages was performed by an experienced user who has specific knowledge already working and ministering robotics classes with it since the year 2008. This user evaluated the environment and reported that the registration form was not providing enough data so that the registration was carried out satisfactorily, suggested that there might be a check at the end of the registration requested that the words have been used correctly.

Despite impairments, the register of language NXC to program Lego Mindstorms NXT robot was successful. Were included in this register functions essential for the use of robots to participate in school robotics competitions.

B. Second step

The second step was performed in a training for participation in the Brazilian Robotics Olympiad 2013, the focus of this stage of the test was to evaluate the operation of the development environment.

The training was attended by 25 people which can be divided into three groups:

- Group 1: Students of elementary school with an average age of 13 years and knowledge of robotics;
- Group 2: Students of higher education in the field of technology;
- Group 3: Students in higher education with specific knowledge of robotics.

The first training followed the basic steps of a class on educational robotics. At first we presented the test of elementary education of the regional stage of the Brazilian Robotics Olympiad - OBR 2013, rescue robots. A lecture was held, leading participants to reflect on how their robots should be and what would be the best strategy for their programming. Then they were challenged to build a robot

that was able to accomplish the challenge. At the end, the development environment was presented and the forms of programming available to date (NXC programming language and programming in R-Educ), were introduced to the participants.

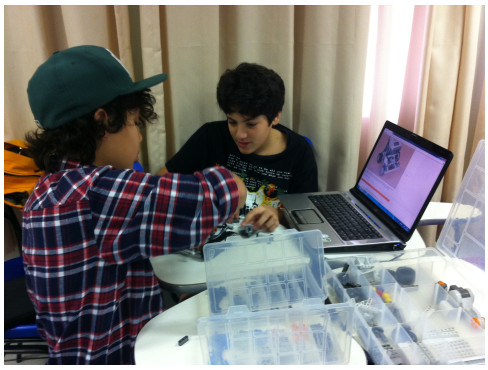

Fig. 4. Second step: assembly robot using the Lego Mindstorms NXT kit

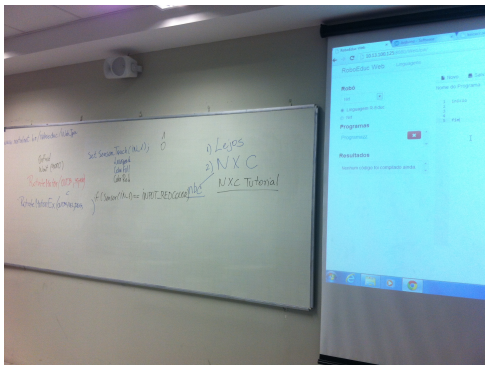

Fig. 5. Second step: presentation of the programming environment

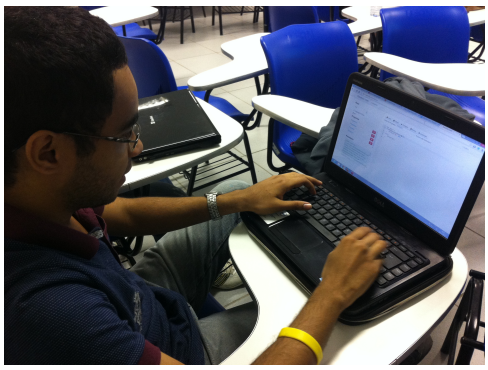

Fig. 6. Second step: programming using the web environment

At the end of the presentation of the programming environment, users have found interesting the fact that they could exchange between programs generated in NXC and R-Educ with just one click. In addition, a subgroup of Group 1 reported that they had experience in Java, and questioned whether it would be possible to program in Lejos, a Java-based language used to program Lego robots. Another student who participated in a research project, reported working with Arduino and questioned whether it would be possible to use the environment to program such module.

Was then initiated the programming of robots. Some of the members of Group 1 chose to use the language R-Educ, being a Portuguese language and easy to understand, while another part of the Group 1 decided to wait for that language Lejos be registered on the environment. Participants in Groups 2 and 3 were urged to program in NXC, since this is a language based on C, similar to what they do use in their academic work. One of the users in Group 2 asked about the possibility of sending the program to the robot via bluetooth.

Figures 4, 5 and 6 presents the second stage of testing. Figure 4 shows users in Group 1 performing assembly, Figure 5 presents step presentation of the environment and Figure 6 a user group 2 performing programming using the environment.

During testing, some users have encountered problems with the system due to instability of the Internet. In addition, the system does not yet have a user registration module, and all who were programming the environment had access to other programs. We evaluate the need to include more functions in the language NXC for the realization of the challenge. All participants were able to use the environment.

C. Third step

After the first and second stages of testing, we conducted an assessment of requirements, considering criticism and requests submitted by the participants of the previous steps.

The programming environment was evaluated according to the observations of the participants in the first stage of testing. We inserted options to download the code generated for the local computer and prepare the system to send programs using any communication port of the local computer.

According to the first stage of testing, we conducted a reformulation of the environment for registring languages, inserting more explanatory texts and restructuring its graphical interface to make it more intuitive. We also implemented a system to perform the scanning of text entered in each field, checking and reporting errors if any required words have not been entered.

D. Fourth step

In the fourth step of the test it was required that an experienced user programmer studied some programming languages for robotics and prepare a guide with data required for registration.

The languages chosen were NXC, registered and used in the previous steps, Lejos, requested by participants of Group 1 of the second stage, and CCS, used to program the robot H-Educ, developed by this same laboratory.

NXC is a language based on C that has to be compiled using an executable compiler that does not need to be installed on the server. If the program is going to be send to the robot via bluetooth the user must know a send code, but if the program is going to be sent using the compiler, this executable file must be send to the local machine to complete this operation.

Lejos is a language based on Java that has to be compiled using a compiler that necessarily needs to be installed on the server. In addition, to send the program to the robot, its necessary to install the compiler on the local computing device. The codes translated from R-Educ to Lejos should form classes with the same name of the saved file, and this treatment is provided by the translator, without the need of extra fields in the registration form.

CCS is a language used for programming PIC microcontrollers brand. We include your registration in this work since it is the language used to program robots H-Educ [11]. For the operation of the H-Educ additional code should be used. Thus, the generation of this language file must contain a header with all the functions and facilities of pins needed. This language allows us to see how much the language R-Educ decreases the programmer's job.

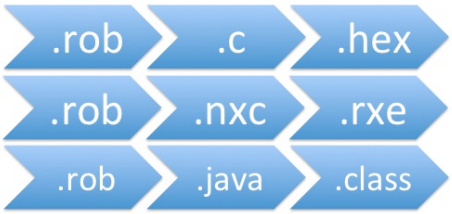

Fig. 7. Sequence of file generation.

Figure 7 shows the sequence of files generated for each of the languages mentioned. The first line of the figure represents the sequence of files generated for the language CCS. To solicit the compilation with a code written correctly on R-Educ, a code with the extension '.c' will be generated, that will be processed by the CCS compiler and then will generate a '.hex' file. The second and third row represent the same sequence generation for the languages NXC and Lejos respectively.

The experienced user generated register guides for each of these languages as well as selected files that should be sent to the server to provide the compilation and sending.

E. Fifth step

In the fifth stage of testing we selected a group to make the register of languages NXC, Lejos and CCS using the registration guides developed in the previous step. This group can be divided into three: the first, corresponding to 23% of the total, with no programming knowledge, the second, corresponding to 64% of the total, with basic knowledge in programming, and the third, corresponding to 13% of the total, with specific expertise in programming languages for robotics.

Were performed a total of 17 entries of languages, and the registration was done successfully for 70% of the cases, most of the mistakes were caused by forgetting key elements of the code syntax, not being made to compile completely because of this error.

Figure 8 present a graph that shows the evaluation of the test group participants regarding the necessary knowledge of

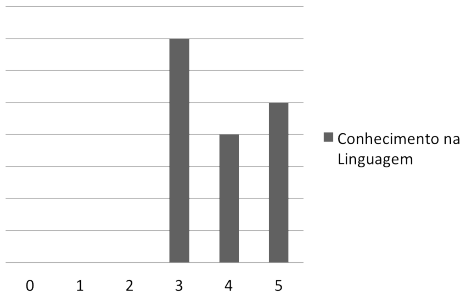

Fig. 8. Knowledge needed to register

the language by the user to perform this registration. The level of knowledge is represented in a range of 0-5 where 0 represents no knowledge and 5 is a large knowledge. As we can see, most participants indicated that it is necessary at least one intermediate knowledge of the language so that the registration could be done.

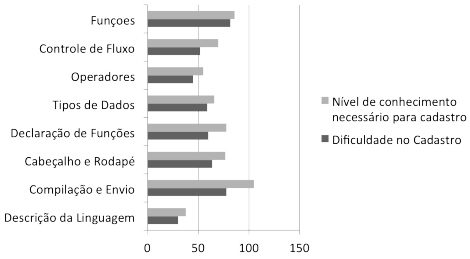

Fig. 9. Graph comparative: difficulty of records and knowledge needed.

The graph shown in Figure 9, presents a comparative evaluation of the level of knowledge required for the registration of each item required and the difficulty of registration of each. Plot data were obtained from a survey of participants who assess the degree of difficulty and level of knowledge required to register each item individually.

Through the data in Figure 9 we can see that the item that was judged as requiring a higher level of knowledge of the language was the compilation and sending, since the user needs to know the system calls to build.

F. Sixth step

The sixth and final stage of testing was performed immediately after the fifth, and the group of participants were asked to test the operation of the language that was registered.

For the test environment, participants had to enter the programming module after performing the registration of a language and use a test program provided in the guide. After entering their code, the participants had to select their robot from among the list of languages associated with robots registered, compile the code in R-Educ, then open the generated code in the language extension registered and perform the compilation to check if the code has been generated correctly. Then, if the communication with the robot was possible, the sending should be performed. This

was not always possible, due to technical limitations of the participant device.

At this stage, some of the participants who registered the language with some kind of error were alerted of this error while trying to compile a program. With registration errors, the code in R-Educ is successfully translated to another language, but the full build is not satisfied.

The graph shown in Figure 10 presents the evaluation of the participants in the test group as the complexity related to some features of the development environment, such as: selection of robots, exchange languages, compiling and sending the program. Each of these items was evaluated on a scale from 0 to 5 where 0 is very easy and 5 is very difficult.

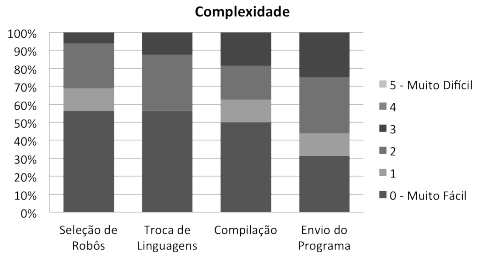

Fig. 10. Complexity attributed to features of the programming environment

We can see from the data obtained and presented that all participants judged the development environment as easy to use.

In Algorithm 2, 3 and 4, we present the translated code for each of the languages registered by participants - NXC, Lejos and CCS respectively - from a basic code, including the main routine and one function, written in R-Educ, presented in Algorithm 1.

```
inicio
frente(A,100,360)
fim
```

Algorithm 1: Code written in R-Educ

The code written in R-Educ presents only the basic data for the compilation. Note that in Algorithm 2 the primary function is created, and includes a motion function.

```
task main(){
        RotateMotor(OUT_A, 100,  360);
}
```

Algorithm 2: Code translated from R-Educ to NXC.

The translated code for Lejos, unlike the translated code into the language NXC, has the inclusion of libraries. These inclusions were within header, and are inserted into all codes translated into this language. We can also observe that for the codes written in Lejos should have a class with the same name of the saved file. This treatment is realized from the use of the word ”programname” in the header.

```
import lejos.nxt.Button;
import lejos.nxt.LCD;
import lejos.nxt.Motor;
import lejos.util.Delay;
public class nomedoprograma {
    public static void main (String[] args) {
                Motor.A.setSpeed(100);
                Motor.A.rotate(360);
    }
}
```

Algorithm 3: Code translated from R-Educ to Lejos

The code translated into the language CCS, presented in Algorithm 4, had part of its header omitted. Note that before the main function some definitions of functions and libraries inclusions are added, in order to reduce the complexity of use for users.

IV. Rating environments

Upon completion of testing, we conducted a survey with participants requesting that they assess on a scale 0-5, where 5 is very good and 0 is very bad, the registration and development environments. The data obtained are presented in the graph in Figure 11.

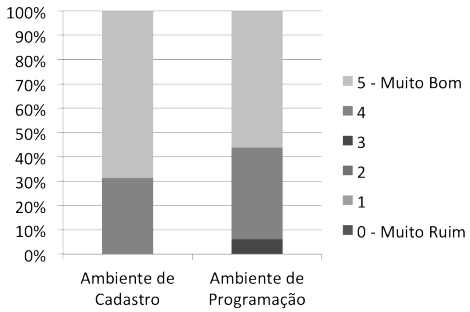

Fig. 11. Reting developed environments

As we can see in the graph of Figure 11, an average of 60% of the participants judged the environment as being very good. One participant stressed that the environment register could be used by anyone who has basic knowledge of programming.

V. Conclusion

The learning environment of educational robotics, when used, helps students in developing creativity, investigative thinking, logical reasoning, coordination, also contributing to structuring skills to scientific research and solving concrete problems.

The design and development of this web configurable development environment for applications in educational robotics was designed to be used as a development environment for students from 8 years to college students, or even anyone interested in learning about robotics.

We consider the proposal that the project should allow the registration of programming languages for robotics, which could easily be used by experienced users on this specific

```
#include <SanUSB.h>
#include <lcd.c>

#FUSES NOWDT
#FUSES PUT
#FUSES NOPROTECT
#FUSES NOBROWNOUT
#FUSES NOLVP
#FUSES NOCPD
#use delay(clock=20000000)
#use rs232 (baud = 1200, xmit = PIN_C6, rcv = PIN_C7)

#define motor1pinA          PIN_B0
#define motor1pinB          PIN_B1
#define motor2pinA          PIN_B2
#define motor2pinB          PIN_B3
#define motor3pinA          PIN_B5
#define motor3pinB          PIN_B6
#define motor4pinA          PIN_B7
#define motor4pinB          PIN_E2
#define pwm1                PIN_C2
#define pwm2                PIN_B1
#define sensor1             0
#define sensor2             1
#define sensor3             2
#define sensor4             3
#define sensor5             5
#define sensor6             6
#define TrigOn1             PIN_A5
#define TrigOn2             PIN_D3
#define echo1         PIN_E0
#define echo2         PIN_E1
#define CLOCKS_PER_SECOND   1000

#define A 1

int16 info, time;

void initPWM(){...}
void initAD(){...}
int16 leituraAD(int sensor, char tipo){...}
void Motor(short int motor, char direcao, short int potencia){...}
char rxUsart(){...}
void flushUsart(){...}
void initTimer(){...}
void initUltrasonico(){...}

enum boolean{
    true = 1, false = 0
};

typedef  enum boolean  bool;

int main () {

        initPWM();
        initAD();
        lcd_init();
        initUltrasonico();

        Motor(A,'H',100);
        delay_ms(360);

        return 0;

}
```

Algorithm 4: Translated code from R-Educ to CCS

language and also lay users on this specific knowledge, that would receive information from experienced user, and then register successfully a new language and make use of it.

Performing an analysis from these primitives, we found that the environment should have joined a number of attributes and explanatory guides that would help in the registration of programming languages. Also developed the programming environment so that the users could have a better use of available resources in a robotics class, thus contributing to their learning and increasing creativity.

We show, through charts, that the environment meets the requirements and desired functionality. It is possible to make use of this configurable development platform for educational robotics teachers, so they would have no more need for change their classes when acquire new hardware. Besides that, it allows experienced users to program directly in the language registered.

REFERENCES

[1] O. R. Neves Junior, Desenvolvimento da fluencia tecnologica em programa educacional de robotica educacional, Masters Dissertation, Federal University of Santa Catarina, Brazil, 2011.

[2] R. V. Aroca, R. Gardiman and L. M. G. Goncalves, Web-based robot integrated development environment and control architecture, Latin American Robotics Symposium, Brazil, 2012.

[3] R. P. Barros, S. Thomaz, A. Aglae, S. Azevedo, A. Burlamaqui and L. M. Goncalves, Roboeduc: Um software para programacao em niveis, in WorkShop de Informatica na Educacao - WIE, Brazil, 2010.

[4] R. P. Barros, Evolucao, avaliacao e validacao do software roboeduc, Masters Dissertation, Federal University of Rio Grande do Norte, Brazil, 2011.

[5] A. Garrett and D. Thornton, A web-based programming environment for lego mindstorms robots, in Proceedings of the 43rd annual Southeast regional conference - Volume 2, New York, NY, USA, 2005.

[6] LEGO, Mindstorms NXT, Available in: ¡http://mindstorms.lego.com¿. Access in: June 25, 2013.

[7] I. R. Belousov, R. Chellali and G. Clapworthy, Virtual reality tools for internet robotics, in ICRA, 2001.

[8] S. Mootien, R. T. F. Ah King and H. C. S. Rughooputh, A web-based interface for the gryphon robot, em International Journal of Electrical Engineering Education, 2006.

[9] D. Lopez, R. Cedazo, F.M. Sanchez and J.M. Sebastian, Ciclope robot: Web-based system to remote program an embedded real-time system, em Industrial Eletronics IEEE Transaction, 2009.

[10] R. V. Aroca, R. P. Barros, A. Burlamaqui and L. M. G. Goncalves, Um robo por aluno: uma realidade possivel, in Workshop de Robotica Educacional - WRE 2012', Brazil, 2012.

[11] S. T. L. Sa, H-educ: Um hardware de baixo custo para a robotica educacional, Graduation Conclusion, Federal University of Rio Grande do Norte, Brazil, 2011.

[12] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.