Graphical Programming Environments for Educational Robots: Open Roberta - Yet another One?

Beate Jost, Markus Ketterl, Reinhard Budde, Thorsten Leimbach
Fraunhofer IAIS
Schloss Birlinghoven
53754 Sankt Augustin, Germany
{Beate.Jost, Markus.Ketterl, Reinhard.Budde, Thorsten.Leimbach}
@iais.fraunhofer.de

Abstract

In recent years, an increasing number of school children is beginning to learn about robotics in the classroom in order to stir their interest in STEM professions. Teachers rely on simple educational robots and intuitive programming environments and graphical programming environments have become a frequent starting point for young robotics newbies. However, currently available tools do often not sufficiently support teachers and students in the classroom. In this study, we evaluate programming environments for educational robots; our results point to the need of lowering the complexity of tools as well as of incorporating combinations of web and cloud technologies, embedded systems and communication concepts into these environments. The technical part of this work presents Open Roberta – an open source based addition to commercial educational robot environments that addresses these needs.

1 Introduction

The **S**cience, **T**echnology, **E**ngineering and **M**ath (STEM) domain is growing rapidly in our working world but children leaving school today are rarely interested in technical professions. There are many different attempts to overcome this barrier. One increasingly used approach is learning with educational robots in classrooms. Researchers and developers have done a competent job since the first inception of classroom turtle programming back in the 1970's [16], [7]. Kelleher and Pausch [10] have already worked out that Educational robots are highly motivating for boys and girls hence keeping this motivation on a high level for children and for teachers is essential. Manifold educational robotic system have been invented or enhanced during the last years. But did they really take into consideration what teachers, the main multipliers, really need - especially in the classroom?

Nowadays interdisciplinary teaching is the way teachers promote the school children's technical competence. These teachers are not necessarily math, physics or computer science experts. Quite often they are responsible for different school subjects or are primary school teachers. Certainly some of the interested teachers already know how to interact successfully with different (education) robotic platforms, programming languages and corresponding coding environments but there is a significant amount of teachers without basic knowledge.

With the Roberta - Learning with Robots initiative Fraunhofer IAIS has developed and implemented a growing concept to assist lectures and teachers directly in the classroom with workshops and practical re-usable tutorials back in 2002. These hands-on programming and guiding materials are growing on a daily basis and can be applied directly in a curriculum. Roberta especially targets the lack of engineers in general but with a focus on female engineers in Germany and other European countries by raising children's interest in technical professions [5]. With Roberta, annually over 35.000 pupils participate in the program and more then 250 teachers are becoming certified instructors each year. These teacher trainings are being evaluated regularly in order to improve the program and adjust to changing demands (see table 1).

V. good	Good	Satisfying	Not Satisfying	Total
67,94%	29,56%	2,50%	0%	100%
462	201	17	0	680

Table 1. Results of the Roberta teacher training evaluation (from 2008-2013) [11]

978-1-4799-4311-1/14 $31.00 © 2014 IEEE
DOI 10.1109/ISM.2014.24

Roberta and the developed programs rely on programmable robots (mainly LEGO hardware) and additional technology which can be used by pre-school pupils up to expert college students. The basis is to use off-the-shelf robot kits and give children and students a target-oriented use of computers without neglecting the playful element. Within the Roberta network a special concept has been developed. It starts with supervised gender specific robot courses and personal training workshops for teachers until a certain level of expertise is reached. The ideal completion of this effort is the establishment and founding of an regional Roberta center (RobertaRegionalCenter) which is able to keep up with further workshops and mentoring (see [12] for more information). Figure 1 depicts an overview of founded regional centers in Europe.

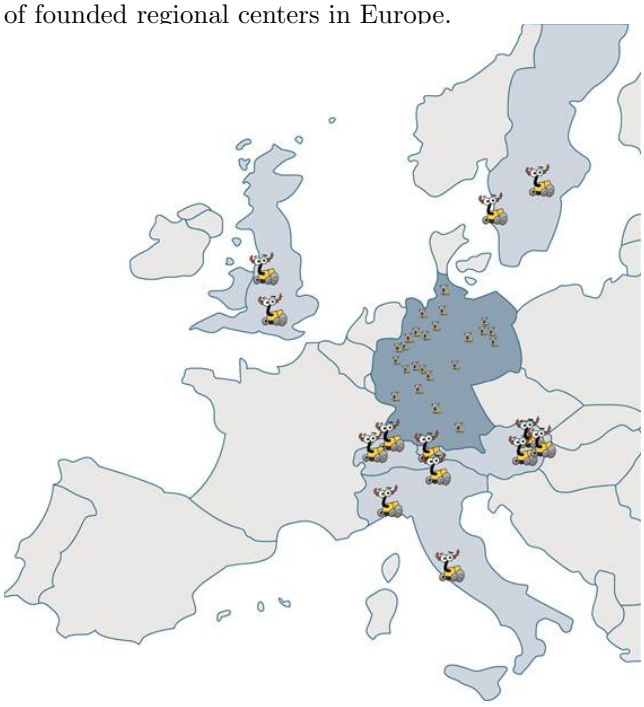

Figure 1. Roberta regional centers in Germany and Europe

The remaining course of the paper is organized as follows: The next chapter presents related work and a brief characterization of benefits of graphical programming with a special focus on classroom teaching. System commonalities, merits and limitations are being addressed in section 3. A focal point is the evaluation of graphical programming environments. Particularly the process of required preparation work and final execution workflows which enable user to run their programs on actual robot hardware is of interest in this study. Furthermore, are available graphical programming environments and available workflows capable of satisfying the needs of secondary school level children and their teachers on a daily basis? The practical part of this work (section 4) addresses this identified benefits and system limitations that have been identified and proposes an online based graphical programming environment with real roboter hardware connection for teachers and learners. This system can be used in todays web browsers without additional plug-ins or other installations. The presented Open Roberta platform allows its users to attach their own robot hardware to a computer or tablet device and directly download and run the generated online programs. Finally section 5 briefly summarizes the implementation so far and provides an outlook of intended future work.

2 Related work and existing graphical programming environments for educational robots

Psychologists and mathematicians Seymour Papert mentions in his work back in 1981 [16] that children and young people need to dominate the computer and not vice versa. Children may not be trained to become an answering machine. Papert further points out, that kids must be in a position to easily experiment with computers and invent new things. The idea of Papert, scientific thinking and acting through independent research with the wealth to promote each other linking perception and action in the physical space, continued and expanded at the MIT in the late 1990s mainly due to work by Mitchel Resnick, Robbie Berg and Michael Eisenberg [19].

Powers at al. [17] provided a broad overview about robot programming systems in 2006. They divided these systems into automatic programming and manual programming environments which can be text-based and/or graphically driven. This basic classification for graphical programming includes also tools for interacting with educational robots. The study lacks a deeper insight into possible system couplings and setup information regarding the connection of real robots, simulations or program deployments.

Other work targets the programming task itself as being presented in work from Kelleher and Pausch [10] in 2005. The authors provide a detailed taxonomy and note, that graphical programming environments belong to the group of so-called *alternatives to typing programs* without further information. Major, Kyriacou and Brereton's report [13] from 2012 provides a pure literature review of teaching novice programming using robots. Although this is a very detailed survey with an almost entire list of programming tools the authors did not make a distinction between text-based and graphical programming environments. Moreover pure

scientific and/or statistical information seems to be less important for practical users like teachers, parents or other instructors in the educational domain.

For most of the children and some teachers in the classrooms robot programming is the very first contact with computer languages and real programming. As learning a textual programming language can be really stressful graphical pendants are frequently chosen, especially in the context of educational robotics.Graphical programming languages provide advantages for beginners compared to text based code interpretations. One can summarize positive attributes such as a better user experience [4], an easier entrance to programming paradigms [9, 17] and they are helpful to prevent typical syntactical errors due to the fact that the syntax of the statements is already implemented into the visual shapes of the statements [10, 21].

O'Hara and Kay [15] (2003) have not evaluated graphical programming environments but educational robotics. They highlight the importance of easily available tools as a very important element for realizing undergraduate classroom projects. And still, ten years later, the setup, hardware communication and installation roundtrip is not classroom ready as frequently criticized by Roberta network participants. Some of these limitations are being further addressed in the next section, followed by a brief systems in use overview.

3 Merits and common limitations of available systems

Teachers and instructors are oftentimes left alone with technical problems in the classrooms. They are quite overwhelmed by technical preparation tasks like system installations for manifold platforms, network/device setups, firewall restrictions and/or huge data downloads before they even can start teaching. Technical resources are limited or oftentimes completely missing at schools. Moreover there is a mixture of old/new computer systems, operation systems and available tools and devices. Many schools have started to focus on using tablet devices to reduce the administrative overhead. But this technical changeover does not play well with the current status of available educational robotic environments.

Several different environments are being offered in the context today. In most cases they belong to or are designed for a specific robotic system. To choose the right one out of them for a specific school or learning task is not easy. A lot of questions have to be answered before a decision for a system can be made. Like for example: What computer systems do we use and maintain? Is the software available for our operating systems? Who can do the installation for one or more classrooms? What do we have to pay? How long will this system be supported? How do we select the right target robot hardware (e.g. costs, stability)? How do we interact with the robots, taking administration restrictions at our school into account?

In summary one can say that it is important to lower the installation complexity in general and to provide generic open tools that can be used across devices and allow an interaction with different robot platforms. Currently this demand can't be fully satisfied by available tools, frameworks and development kits as we highlight in the next subsection.

3.1 Commonly used graphical robot programming environments

Based on our experience we've selected some of the most frequently used educational robot ecosystems for an elaboration; taking into account the previously mentioned criteria. Particularly we are taking a closer look at the systems:
Enchanting [1], EV3[1] [22], GRAPE [6], miniBloq [2], Modkit [14], ROBO pro [8], NXT-G[1] [23], RobotC[2] [20], RoboMind [18], Ardublock [3].
Table 2 depicts different features of the systems under evaluation. All of them provide integrated development environments (IDEs) for multiple computer platforms and operation systems and also present graphical programming alternatives in addition to a pure code based representation to users. Certain robot hardware can be connected and user generated programs can be downloaded from the IDE to the robot device. Some of the IDEs provide a web interface like Modkit and RoboMind but extra system specific software needs to be installed for a robot communication. All, except two products (Enchanting and Minibloq) are fee-based, users need to pay for at least the educational version. In general each of the IDEs is designed for a certain robot system or hardware family. ROBOTC, Modkit, and Minibloq support at least two robot platforms. EV3 supports its preceding robot system NXT partially at the moment. The ideal system candidate needs to include the following features: Visual graphic programming support, fully web based with possibilities to run the infrastructure remotely or on local school servers, connections to robots should be easy (e.g. WLAN, Bluetooth) and it should not be restricted to a certain vendor, the technical complexity like compilation and program preparation should run in the background, price sensitive (or free) in

[1]LEGO® MINDSTORMS® Software
[2]for LEGO® MINDSTORMS® and MINDSTORMS with TETRIX™

IDE	Platforms: Windows	Linux	Mac OS X	web	Costs	Robot
NXT-G	×	-	×	-	(yes)[1]	1[2]
EV3	×	-	×	-	(yes)[1]	2
Enchanting	×	×[3]	×	-	no[4]	1
RobotC	×	-	-	-	yes	2
ROBO pro	×	-	-	-	yes	1
Modkit	×	-	×	(×)[5]	yes	2
GRAPE	×	-	-	-	yes	1
miniBloq	×	×[6]	-	-	no[4]	2
RoboMind	×	×	×	(×)[5]	yes	1
Ardublock	×	×	×	-	no[4]	1

Table 2. IDE comparison of portability, costs and number of supported robots

[1] only the educational version is fee based
[2] third-party sensors and motors are supported
[3] experimental version
[4] open source
[5] needs further platform depending software for the robot ↔ application communication
[6] preliminary version

order to allow schools to easily join, test and participate as individual or as a group (school class). None of the available products currently fulfill this criteria without technology getting in the way.

This is why we are proposing a fully online, web based alternative for visual educational robot programming which allows to easily use the browser for programming without manual installation and preparation with an easy connection with different robot hardware. The compilation process and program preparation is intended to run on local or cloud based servers and is not limited to a certain computer platform or system software.

4 Open Roberta - An open source online programming environment for teaching and learning with flexible integration of robot hardware

Parts of the Roberta workshop material is currently based upon existing proprietary LEGO Mindstorms IDE software solutions which are available only for Windows- and Mac computers. Roberta teachers are frequently faced with the previously mentioned roadblocks and limitations. In order to solve this recurring issues we have started an open source network initiative with the intention to make the programming of robots easier and more fun for all participants. A major goal of the Open Roberta project is to work on a software platform dissemination for a scalable, education-friendly, cloud-based robot development kit.

4.1 From desktop to tablet towards an easy robot program deployment

We have a deep interest in software portability. Portability stands for reusability in schools e.g. robots have been acquired in one year and computers for the classroom in another year. Can the robots still be programmed with the newly purchased tablet pc's? And what about the other way round? Cross-platform IDE's are well suited in this case, but none of the given programming environments are really cross-platformed. Modkit and RoboMind have a web interface but the real work (compiling the program) still has to be done on the computer and therefore special drivers and software is necessary. "Cloud programming" or programming via a web interface on a server would make administration redundant: no installation effort, updates are installed automatically, available also for tablets.

4.2 Technical perspective and goals

The age of the students we are mainly trying to address is between 10 and 16. The teacher may have a computer science background but it is much more likely that his or her main experience is natural science or mathematics. The preparation time for a course is limited, the hardware heterogenous, professional IT support is almost never available and the number of students in a course is usually large. This determines our goals for a programming environment for educational robots.

The intended project will be fully based on open source technology. The development process shall be open and flexible which includes involving extern parties like interested partners within the Roberta network (e.g. also universities) or other interested individuals. An environment must have almost no setup, no configuration, it has to be available for almost all platforms/operating systems and must be easy to use, because only very well known user interface concepts are used. We believe, that in this case using a browser is the best decision to abstract from the underlying operating system.

The browser's user interface is well known to almost all teachers and students, including its limitations. Furthermore it is probably the only solutions that allows

the use of the diverse tablets. Thus a client server design is appropriate which is based on: The design of a server back end, that relies on Ajax and REST. We favor Java and its many robust well known frameworks for this job. The use of a Javascript enabled browser and adopt some well engineered frameworks for the fronted architecture. Of course we cannot avoid a basic robot setup, because we need a communication facility between server and robot, but – compared to work station setups – this should be easy and straightforward.

After having decided to use a browser-based client, the communication between robot and client is the major problem, as the client is executed in a "sandbox". Any local solution, be it Bluetooth, NFC, direct USB connection is not feasible without heavy configuration and administrator privileges.The only practical solution to this problem is, that both the browser client (using Ajax calls) and the robot (acting as a HTTP client) are connected to the same server using Wifi. In most cases this will be based on a Wifi network that is connected to the internet. Then any public server hosting the backend software can arrange the connection between robot and browser client.

A simple and sufficiently safe solution would be a token, that can be used to pair the robot and a browser client. The token could be generated on the robot and retyped by the user into the browser window. This establishes the connection. Such a connection exposes user names, passwords, programs and information about teachers and pupils to the public. In the straight past this would have been advertised as a kind of cloud server and cloud storage. But many schools, teachers and students have concerns w.r.t. to privacy. They should not be forced to expose their data when using the programming environment. Therefore it should be possible to use any local Wifi network to which a computer and a robot can connect to. A local server must be deployable on any -even small- notebook or PC in this network. A minimal solution consisting out of a smartphone establishing a hot spot, a notebook on which the server is deployed and (the same or more) notebooks or tablets executing the browser client should be usable without any restriction.

Actually we expect that a Java >= 7 installation on a computer is the only prerequisite for deploying the back end part of our programming environment. The client assumes no installation - beside a Javascript enabled browser - at all. To attract rookies it makes sense to offer a pure graphical programming language. We have chosen to use blockly for that purpose, as many other projects did. But it would be a bad idea to limit ourselves to that kind of program representation. Offering only one perspective to understand software development would hide the comprehensive possibilities of computer science. Furthermore, if problems (and solutions) become bigger and bigger, graphical programming evolves to a tough job very often. Thus we plan to offer more perspectives of programming especially a round trip between graphical and textual representation for the user. The programming environment must support different code generators, too, Java being the first. The generated code has to be human readable, thus the many levels of abstraction in system development can be experienced by the pupils. Last, but not least, execution of program on a robot as well as on a simulation engine should be possible.

It is almost impossible to foresee the way educational software will be used in the future (this is true for other kinds of software as well). It is very likely, that people benefit from the evolution of the software into many very distinct directions. We don't expect to have the resources to achieve that on our own. Therefore our programming environment has to be open source. Furthermore it should not confuse developers enhancing it. We want to achieve that by applying well-known standards (rules as design by contract, tools as sonar) and well-known frameworks (as jetty, jersey, jaxb2, hibernate, antlr4). The code base should be as small as possible. The NIH (not invented here) syndrome must be avoided. Using existing frameworks is preferred to re-inventing solutions. We hope to have tackled most of our goals in the first version of our programming environment.

5 Conclusion and outlook

The presented Open Roberta project will have a huge impact for the existing Roberta network. The intention is to solve many of the frequently occurring issues reported by our certified educational robotic teachers (Roberta teacher) in the actual classrooms. We've exemplified merits and common limitations of available visual programming development environments with a special focus on technical obstacles. This overview and analysis is the baseline for the development of the open source visual online programming environments for educational robots as presented in this work. The further development of Open Roberta is going to focus on three main traces. A further extension of the visual programming language including adaptation of additional robot sets. A collaborative programming environment for complete school classes and/or individuals. The extension of available tutorials and hands-on practical guides for both teachers and kids with an integration of blended learning concepts, tools and online infrastructure.

Acknowledgements

Besides the members of the Open Roberta community we would like to thank Google for their generous project support and advice.

References

[1] Enchanting. Available online: http://enchanting.robotclub.ab.ca/tiki-index.php, accessed 08-22-2014.

[2] Minibloq, 2014. Available online: http://blog.minibloq.org/, accessed 08-25-2014.

[3] Ardublock. Ardublock a Graphical Programming Language for Arduino, 2014. Available online: http://blog.ardublock.com/, accessed 08-15-2014.

[4] T. Booth and S. Stumpf. End-User Experiences of Visual and Textual Programming Environments for Arduino. In Y. Dittrich, M. Burnett, A. Mø rch, and D. Redmiles, editors, *End-User Development*, pages 25–39. Springer Berlin Heidelberg, 2013.

[5] A. Bredenfeld and T. Leimbach. The Roberta Initiative. Proceedings of the 2nd International Conference on SIMULATION, MODELING and PROGRAMMING for AUTONOMOUS ROBOTS (SIMPAR 2010), pages 1–2, Darmstadt, 2010.

[6] S. Enderle. The qfix Robot Kits. In D. Gottscheber, Achim and Enderle, Stefan and Obdrzalek, editor, *Research and Education in Robotics — EUROBOT 2008*, pages 84–95. Springer Berlin Heidelberg, 2009.

[7] W. Feurzeig. Educational Technology at BBN. *Annals of the History of Computing, IEEE*, 28:18–31, 2006.

[8] Fischertechnik GmbH. ROBO pro, 2014. Available online: http://www.fischertechnik.de/, accessed 08-15-2014.

[9] Z. Istenes and A. Pásztor. Using Innovative Devices in the Education of IT from Primary to University Level. In *Sixteenth International Electrotechnical and Computer Science Conference, ERK*, volume 24, page 26, 2007.

[10] C. Kelleher and R. Pausch. Lowering the Barriers to Programming : A Taxonomy of Programming Environments and Languages for Novice Programmers. *ACM Computing Surveys (CSUR)*, 37(2):83–137, 2005.

[11] T. Leimbach, B. Jost, U. Petersen, J. Börding, and S. Härtig. *Roberta-Grundlagenband EV3*. Fraunhofer Verlag, Stuttgart, Germany, 2014.

[12] T. Leimbach and U. Petersen. *Roberta-Goes-EU: Zielgruppensensible Robotikkurse als Erfolgsmodell in Europa*, pages 183–194. Gender und Diversity in den Ingenieurswissenschaften und der Informatik. UniversitätsVerlagWebler, Bielefeld, 2008.

[13] L. Major, T. Kyriacou, and O. Brereton. Systematic literature review: teaching novices programming using robots. *IET Software*, 6(6):502–513, 2012.

[14] Modkit LLC. Modkit for VEX, 2014. Available online: http://www.modkit.com/vex, accessed 08-25-2014.

[15] K. J. O'Hara and J. S. Kay. Investigating open source software and educational robotics. *Journal of Computing Sciences in Colleges*, 18:8–16, 2003.

[16] S. Papert. *Mindstorms: Children, computers, and powerful ideas*. Basic Books, New York, NY, January 1981.

[17] K. Powers, S. Cooper, K. J. Goldman, M. Carlisle, M. Mcnally, and P. Viera. Tools for Teaching Introductory Programming : What Works ? *ACM SIGCSE Bulletin*, 38:560–561, 2006.

[18] Research Kitchen. RoboMind, 2014. Available online: http://www.robomind.net/en/index.html, accessed 08-26-2014.

[19] M. Resnick. *Turtles, termites, and traffic jams - explorations in massively parallel microworlds.* MIT Press, 1998.

[20] Robomatter, Inc. ROBOTC for LEGO MINDSTORMS, 2014. Available online: http://www.robotc.net/download/lego/, accessed 08-25-2014.

[21] E. Rosenbaum, E. Eastmond, and D. Mellis. Empowering programmability for tangibles. In *TEI '10: Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction*, pages 357–360, New York, New York, USA, 2010. ACM Press.

[22] The LEGO Group. LEGO® MINDSTORMS® Education EV3-Software, 2014. Available online: http://education.lego.com/de-de/lego-education-product-database/mindstorms-ev3/2000045-lego-mindstorms-education-ev3-software-single-user, accessed 08-25-2014.

[23] The LEGO Group. LEGO® MINDSTORMS® Education NXT Software 2.1.6 (inkl. Messwerterfassung), 2014. Available online: http://education.lego.com/de-de/lego-education-product-database/mindstorms/2000080-lego-mindstorms-education-nxt-software-v-2-0, accessed 08-25-2014.