

Evaluating the LEGO–RoboLab Interface with Experts

ATHANASIS KAROULIS

Aristotle University of Thessaloniki , Thessaloniki, Greece

This work presents the results from a joint usability study concerning the LEGO programming environment "RoboLab". LEGO has already performed numerous evaluations on various usability aspects, aimed to enhance the overall usability and utility of the product. This study focuses on a more academic parameter; namely, the combination of different evaluation methods on the same software piece, as well as on their combined results. Therefore, two expert-based methods have been applied at the Department of Informatics of Aristotle University: a cognitive graphical walkthrough and a heuristic evaluation. The results of the study unveiled some deficiencies of the interface and some limitations of the employed methods as well.

Categories and Subject Descriptors: H.5.2 [**Information Interfaces and Presentation**]: User Interfaces -- *Evaluation/methodology*; K.3.1 [**Computers and Education**]: Computer Uses in Education; D.2.6 [**Software Engineering**]: Programming Environments; K.8.0 [**Personal Computing**]: General -- *Games*

General Terms: Design, Human Factors, Measurement

Additional Key Words and Phrases: Interface evaluation, usability, cognitive graphical walkthrough, heuristic evaluation.

INTRODUCTION

This study concerns the usability evaluation of the RoboLab environment, the programming environment for the LEGO Mindstorms RCX hardware.

LEGO is a company that manufactures construction toys for children of all ages. The design and construction process accepts the constructivist view of child development and learning, combined with flow theory, presented in works of Vygotsky [1936;1978], Piaget [1952], and Papert [1980].

Constructivism, pioneered by Piaget, states that knowledge should not be simply transmitted from teacher to student, but actively constructed by the mind of the student, as noted in Mindel et al. [2000]. "Learning is an active process in which people actively construct knowledge from their experiences in the world," as Resnick et al. [1997] state. Seymour Papert, co-founder of the MIT Media Lab, extends it to what he has termed the "constructionist" approach to learning [Papert 1980]. Constructivism adds the idea that people construct new knowledge with particular effectiveness when they are engaged in building projects that are personally meaningful. Students construct their own knowledge effectively while building creations that interest and excite them, thus encouraging them to learn.

Author's address: Department of Informatics, Aristotle University of Thessaloniki, PO Box 888 – 54.124 Thessaloniki, Greece; email: karoulis@csd.auth.gr

Permission to make digital/hard copy of part of this work for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication, and its date of appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee. Permission may be requested from the Publications Dept., ACM, Inc., 1515 Broadway, New York, NY 10036, USA, fax:+1(212) 869-0481, permissions@acm.org

© 2005 ACM 1544-3574/05/1000-ART5A \$5.00

However, when implemented in more complex environments such as programming, even trial-and-error and experimenting seem to be insufficient for the majority of the children. It is interesting to compare sayings about constructivism from two researchers:

Piaget [1952] said: "Each time you explain something to a child, you prevent him/her from discovering it." But Edith Ackermann [2003] contends: "Yet at the same time, each time you fail to give wanted-for-help when needed, you prevent him/her from getting more deeply involved." She adds: "The purpose of a good mentor is to decide how much freedom and guidance makes for a nurturing and yet challenging learning experience. In addition, different people need different kinds of feedback, at different times, in different situations. A good clinician, like a good teacher, is someone who masters the art of providing the 'right' amount of elbowroom in each singular case." Hence, one could argue that this is also one of the desired properties of educational software – to offer a rich environment for experimenting, simultaneously providing enough and just-in-time information and guidance.

DESCRIPTION OF THE SOFTWARE

The LEGO Company accepts this constructivist learning philosophy while developing both toys and educational materials for children. Thus, in close cooperation, the LEGO Company and the MIT Media Lab developed the LEGO Mindstorms RCX programmable brick as a central part of a robotic construction set, which was put on the market in 1998. The "brain" of LEGO robots is a programmable brick, RCX. The RCX brick is a programmable microcomputer that can control up to three motors and take input from up to three sensors when it executes a program made on a personal computer. LEGO offered two environments for programming RCX brick: Mindstorms, mainly for individual home use, and RoboLab, mainly for collaborative use in classroom environments or after-school activities [Timcenko 2005]. LEGO hobbyists, students, and teachers throughout the world developed several other ways to program RCX, using specifically tailored languages like NotQuiteC, or general-purpose languages like C++ and Java.

This study will consider only the RoboLab programming environment. Based on LabVIEW™, from National Instruments, Texas USA, the RoboLab Software uses an icon-based, diagram-building environment to write programs that control the RCX.

The main idea in developing RoboLab was to empower elementary school children (as young as possible) to do some programming and engineering activities otherwise not graspable for them, thus boosting their interest in technology and natural sciences. With RoboLab's customized user interface designed for student users ages eight and up, LEGO was aiming to bring the best values of the constructionist learning approach to children and their teachers.

The workflow (an example depicted in Figure 1) while building and programming robots using LEGO bricks and the RoboLab programming environment consists of:

- Users first building their invention using the RCX and the LEGO elements included in the LEGO sets.
- Then creating a program for their invention using RoboLab
- Downloading the program to the RCX using a special infrared transmitter.
- Testing their fully autonomous creation in direct interaction with the environment, and eventually modifying and improving it, starting again from the first or second bullet point.

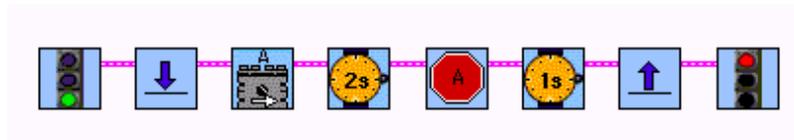


Fig. 1. An example of a Mindstorm program.

RoboLab encourages this incremental-loop learning style, as it has progressive programming phases that allow the programming level to match the students' knowledge and skills. Thus, it is subdivided into following graduations:

1. *Pilot* is the basic environment where programs are built using a click-and-choose interface.
2. *Inventor* provides a more open-ended, icon-based environment.
3. *RoboLab Investigator* uses *Pilot* and *Inventor* programming to incorporate data collection into projects.

Training Missions are also included in RoboLab environment. They are interactive audio and video tutorials for students and teachers to become more familiar with RoboLab programming. In addition, a detailed teacher's guide is available, in a form of a book or a PDF file on a CD.

MOTIVATION OF THE STUDY

Programming is a very abstract human activity that requires training. In the case studied, all RoboLab programming environments are used for programming microprocessor-controlled LEGO robots built of LEGO bricks, together with motors and different sensors. Despite success in sales, there are reports that some aspects of both Mindstorms and RoboLab are very difficult to grasp for both children and their instructors. Therefore, an interesting question emerges, as one would like to understand the source of these difficulties and try to improve or redesign the existing programming environments. The ideal process includes not just redesign of icons, but questions representations of as many programming environment elements as possible – providing inspirational materials, guidance, error reporting, etc. Efficient ways of providing help for different functions might also need to be an object of research, as the existing one (in spite of providing help in several levels of details depending on users' requests) proved not very clear for novice users.

Ideally, as a practical result this should lead to suggestion of redesign of some parts of the LEGO programming environment for teenage children. On a more abstract level, the result should provide some guidelines on how to design certain aspects of programming environments for teenagers.

As a first step in this direction, a usability evaluation session has been organized and performed at the Multimedia Laboratory of the Department of Informatics of the Aristotle University of Thessaloniki. The used methodologies as well as the evaluation procedure are described subsequently.

USABILITY EVALUATION

What exactly is a "usability evaluation?" A usability or interface evaluation of a software system is a procedure intended to identify and propose solutions for usability problems caused by the specific software design. The term "evaluation" generally refers to the process of "gathering data about the usability of a design or product by a specified group of

users for a particular activity within a specified environment or work context” [Preece et al. 1994, p.602]. The main goal of an interface evaluation is, as already stated, to discover usability problems. A usability problem may be defined as “anything that interferes with user’s ability to efficiently and effectively complete tasks” [Karat et al. 1992]. An evaluation of user interface design is of special importance in the overall software evaluation plan, for two major reasons: Firstly, because it concerns exactly that part of the software product which enables users to communicate their instructions to the machine. Evaluation should verify that the interface design delivers a friendly, intuitive, transparent yet powerful environment to end-users for the accomplishment of their goals, which in our case is the acquisition of knowledge through the interaction with the instructional environment. Secondly, it is paramount because evaluation of the user interface should be carried out at the right time; early enough to offer designers the chance of getting valuable feedback about their design ideas and possibly proceed to interface redesign while all important interface characteristics have been designed and are included for evaluation.

The choice of a particular method depends on the stage of development of the interface, the extent and type of users’ involvement, the kind of data expected, external limitations such as time constraints, cost and availability of equipment, and so forth [Aedo et al. 1996].

EXPERT-BASED VS. USER-BASED EVALUATIONS

The most applied methodologies are the expert-based and the empirical (user-based) evaluation. Expert evaluation is a relatively cheap and efficient formative evaluation method applied even on system prototypes or design specifications up to the almost ready-to-ship product. The main idea is to present the tasks supported by the interface to an interdisciplinary group of experts who will take the part of would-be users and try to identify possible deficiencies in the interface design. However, according to Lewis and Rieman [1994] “you can't really tell how good or bad your interface is going to be without getting people to use it.” This expresses the broad belief that user testing is inevitable in order to assess an interface. Why do we then use not only empirical evaluations, but other research approaches as well? As we may see further on, the efficiency of these methods is strongly diminished by the required resources and by some other disadvantages, while on the other hand, expert-based approaches have matured enough to provide a good alternative.

The first main disadvantage of empirical studies is the personal bias of the subjects. It's important to understand that test users can't tell you everything you might like to know, and that some of what they will tell you is useless. This is not done on purpose; for different reasons users often cannot give any reasonable explanation for what happened, or why they acted in a certain way. Psychologists have done some interesting studies on these points.

Maier [1931] had people try to solve the problem of tying together two strings that hung down from the ceiling too far apart to be grabbed at the same time. One solution was to tie some kind of weight to one of the strings, set it swinging, grab the other string, and then wait for the swinging string to come close enough to reach. It's a hard problem, and few people came up with this or any other solution. Sometimes, when people were working Maier would “accidentally” brush against one of the strings and set it in motion. The data showed that when he did this, people were much more likely to find the solution. The point of interest is what these people said when Maier asked them how they solved the problem. They did NOT say, “When you brushed against the string that gave me the idea of making the string swing and solving the problem that way,” even though Maier knew that's what really happened. Therefore, they could not and did not tell him what feature of the situation really helped them solve the problem.

Lewis and Rieman [1994] give the three prerequisites for an empirical evaluation:

- People: if possible real users in real circumstances
- Some tasks for them to perform, and
- Some version of the system to work with

At this point, we already have another obstacle regarding empirical approaches: All these prerequisites are required simultaneously.

On the other hand, according to Reeves [1993], expert-based evaluations are perhaps the most applied evaluation strategy. Why is this? They provide a crucial advantage which makes them more affordable compared to the empirical ones: in general, it is easier and cheaper to find experts rather than users who are eager to perform the evaluation. The main idea is that experts from different cognitive domains - in particular, at least one from the domain of HCI and at least one from the cognitive domain under evaluation - are asked to judge the interface, each from his own point of view. It is important that all are experienced, so they can see the interface through the eyes of the user and reveal problems and deficiencies of the interface. One strong advantage of the method is that it can be applied very early in the design cycle, even on paper mock-ups. The expert's mastery allows him to understand the functionality of the system under construction, even if he lacks the whole picture of the product. A first look at the basic characteristics would be sufficient for an expert. On the other hand, user-based evaluations can be applied only after the product has reached a certain level of completion.

Another important issue about empirical evaluations is to finding representative users, like the ones that will use the system in real life situations. As Lewis and Rieman [1994] emphasize, "if you can't get any representative users to be test users, why do you think you'll get them as real users?" Therefore, we mention here the statement of the aforementioned researchers that the first step for a user-centered interface design is for the design team to visit the real users in their real environment and write down the real tasks they're performing in doing their work. User-based evaluation must simulate the real user environment in a real workplace (not in the laboratory), with representative users (not available users). This constitutes the second main disadvantage of empirical evaluations, as it is difficult to find representative users to evaluate the system under real work conditions, which consequently means augmented evaluation costs and a more difficult performance of the evaluation session. Yet this approach would be optimal for third generation ODL environments, supposing one can pinpoint real ODL students eager to evaluate an environment under construction. Based on our experience, we claim that this goal is difficult to achieve, if possible at all.

On the other hand, expert-based evaluations overcome these limitations by replacing users with experts, providing qualitatively poorer results, but with far less cost and difficulty; namely, a much better performance/cost factor. Therefore, we claim that an expert-based evaluation of an ODL environment is more likely to succeed in most cases.

THE COGNITIVE GRAPHICAL WALKTHROUGH

The *cognitive graphical jogthrough* method (described in detail in Karoulis et al. [2000]) belongs to the expert-based evaluation methodologies. Its origin is in C. Lewis and P. Polson's work, where the initial *cognitive walkthrough* was presented [Polson et al. 1992; Wharton et al. 1994], and in the improved version of the *cognitive jogthrough* [Rowley and Rhoades 1992; Aedo et al. 1996; Catenazzi et al. 1997]. The main idea in the expert-based evaluation is to present interface supported tasks to a group of four to six experts who will

play the role of would-be-users in order to identify any possible deficiencies in the interface design. In order to assess the interface, a set of tasks has to be defined which characterizes the method as "task-based." Every task consists of a number of actions which complete the task. The methods utilize an appropriately structured questionnaire to record the evaluators' ratings. They are also characterized as "cognitive" to emphasize that the focus is on the cognitive dimension of the user-interface interaction and special care should be given to understand the tasks in terms of user-defined goals and not just as actions on the interface (click, drag, etc.).

The evaluation procedure itself takes place as follows: A presenter describes the user's goal that has to be achieved by using the task. Then he/she presents the first action of the first task. The evaluators are trying to:

- i) Pinpoint possible problems and
- ii) Assess the percentage of users who would possibly encounter problems, according to the questions in the questionnaire.

When the first action is finished, the presenter presents the second one, and so on, until the whole task has been evaluated. Then the presenter introduces the second task, following the same steps. This iteration lasts until all tasks have been evaluated. The questions stated via the questionnaire to the evaluators are as follows:

- a) How many users will think this action is available?
- b) How many users will think this action is appropriate?
- c) How many users will know how to perform the action?
- d) Is the system response obvious? YES NO
- e) How many users will think that the system reaction brings them closer to their goal?

These questions are based on the CE+ *theory of exploratory learning* by Lewis and Polson [Polson et al. 1992; Rieman et al. 1995]. In Appendix A, there is a sample page of the evaluators' questionnaire, but with modified phrasing of the questions derived after these studies.

THE DIAGRAMS

The basic idea in modifying the walk- and jogthrough methods was that both of them focus on novice or casual users who encounter the interface for the first time. However, this limits the range of the method's application. Therefore, we tried to introduce the time factor in the form of the augmentation of the user's experience while working in the interface. This was achieved through the embodiment of diagrams where the evaluators record their estimations. The processing of the diagrams produces curves, one for each evaluator, so that the diagrams graphically represent the intuition and the learning curve of the interface. Educational environments were mainly chosen for the application of the modified method, as they are particularly demanding in many aspects. Moreover, the results have shown that the method is not only generally applicable, but also, under certain circumstances, it approaches the performance of empirical evaluation methods.

The core issue in the modified method of the cgw/cgj is the different types of diagrams wherein the evaluators can note down their assessments regarding the augmentation of the user's experience during his/her work in the interface. The following is a typical cgj diagram:

Almost all users					
Most users					
About half users					
Some users					
Almost no one					
	1 st attempt	Few (2-3) attempts	Some (4-6) attempts	More (7-8) attempts	Many (>8) attempts

Fig. 2. A typical cgj diagram.

THE HEURISTIC EVALUATION

Perhaps the most frequently encountered evaluation method of any entity is the provision of a list of criteria relative to this entity, followed by questioning in order to express peoples' opinion. However, a number of problems arise from this approach:

1. It provides all the disadvantages of the expert-based evaluations [Karat et al. 1992; Nielsen 1993; Karoulis et al. 2000].
2. The criteria list may become very long [Lewis and Rieman 1994; Nielsen 1993]. For example, the full interface usability criteria list suggested by Smith and Mosier [1986] includes 944 criteria.
3. The evaluators' expertise plays a major role [Lewis and Rieman 1994; Nielsen 1993].

To handle these problems, Jacob Nielsen and Rolf Molich started their research in 1988, and in 1990, they presented the "heuristic evaluation" [Nielsen and Molich 1990]. The basic point was the reduction of the set criteria to just a few, being both broadly applicable and generally agreed upon, while simultaneously augmenting the evaluators' expertise, and, consequently, their reliability. These "heuristic rules" or "heuristics" were derived from studies, criteria lists, field observations and prior experience of the domain.

The core point to evaluate in the initial approach is the usability of the interface. Based on the ISO principles about usability [ISO 1998], Nielsen [1994] stated the following heuristics, slightly modified and reorganized:

1. Simple and natural dialog and aesthetic and minimalist design.
2. Visibility of the system status – provide feedback.
3. Speak the users' language: match between system and real world.
4. Minimize the users' cognitive load: recognition rather than recall.
5. Consistency and standards.
6. Flexibility and efficiency of use – provide shortcuts.
7. Support users' control and freedom.
8. Prevent errors.
9. Help users recognize, diagnose and recover from errors with constructive error messages.
10. Help and documentation.

THE STUDY

Preparation of the evaluation session

The first step in the evaluation procedure of the cgw (cognitive graphical walkthrough) is

the definition of the tasks and pending actions, which will be evaluated. In order to do this, an initial program must be defined, which the user has to construct on his/her computer by using the RoboLab environment. The following procedure has been set here: the robot must go straight until it encounters an obstacle. Then, it has to go backwards for a while, turn (left or right), and move forward again. Accordingly, this procedure has been analyzed in tasks and actions, as shown in following Table 1.

Accordingly, a "note to the evaluators" was distributed, explaining the basics of the method. This was done just because it had to be done. The participating evaluators were all experienced experts on the HCI domain, two of them were the modifiers of the used method (cgw), and there was no extra need for explanations. Finally the software was set up and the session begun.

Table I. Tasks and Actions Defined for the CGJ Session

1. Forwards
1.1. Power MOTOR A
1.2. Power level for MOTOR A
1.3. Power MOTOR C (same direction with MOTOR A)
1.4. Power level for MOTOR C
2. Obstacle
2.1. Activation of the touch sensor
3. Backwards
3.1. Power MOTOR A (in opposite direction)
3.2. Power level for MOTOR A
3.3. Power MOTOR C (in opposite direction)
3.4. Power level for MOTOR C
4. Away from the obstacle
4.1. Release of the touch sensor
5. Turn
5.1. Power MOTOR C (direction as in task1)
5.2. Power level for MOTOR C
5.3. Setting time (2 sec)
6. Forwards
6.1. Power MOTOR A (as in task1)
6.2. Power level for MOTOR A

THE SESSION

Three evaluators took part. All were HCI experts; two of them were double experts, as they had an educational record of over 20 years each. The session lasted more than 3 hours, yet not all prescribed actions could be evaluated. The following session is described in detail.

Part I: cognitive graphical walkthrough

The session was designed to perform at Pilot level 4 and Inventor level 2.

Initially, the interface to be evaluated was introduced. Two of the evaluators were unaware of it. At first, all evaluators agreed that it is about a hard-to-understand interface, especially for novice users. The main argument was that there was no prior experience from other comparative interfaces on which a novice user could rely. In addition, the used icons and metaphors were considered not obvious at first sight.

At this point, the session started with the elaboration of the predefined tasks and

actions. The procedure followed as already described, however, the set of questions was the ultimately elaborated and proposed set in Karoulis et al. [2005a]:

- How many users will think that this action is **available**, namely that the system can do what the user wants, and simultaneously affords the mode for it to be done?
- How many users will consider this action, and not some other, to be **appropriate** for the intended goal?
- How many users will know how to **perform** this action?
- Is the system response **obvious**?
- How many users will consider that the system response brings them **closer to their goal**?

To answer these questions, the evaluators used the already presented diagrams, one for each question.

At action 1.2, the evaluators started to discuss (and agreed) that the used method of the cgw did not perform well in the particular interface. The main argument in this direction was that the method aims to evaluate interfaces which are designed for the first contact, such as walk-up-and-use interfaces [Lewis et al. 1990; Lewis and Rieman 1994]. However, in the particular interface it was for many actions debatable whether the novice user could even perceive that the specified action was present at first sight. Therefore, they considered that the initial contact of the user with the interface would not be successful, a prerequisite to apply the cgw method. The evaluators proposed the heuristic evaluation as a logical substitute to continue the session. Despite this remark, the session continued with the cgw for a while. However, after the evaluation of task 2, the evaluators agreed to stop and continue with the heuristic method, and they proposed to continue with the Inventor level 2.

Part II: heuristic evaluation

During a short break, the pending documents were printed. Every evaluator received the slightly modified heuristic list proposed in Karoulis and Pombortsis [2002], as well as an "evaluator's notebook." As the evaluators were also experienced in this method, no further explanations were needed and the second part of the session began.

RESULTS

cgw results

There was from the beginning an extensive discussion on the direction of the arrow depicted on the motor icon (the third icon in Figure 1), and, in particular, in which direction the motor would spin. The debate concerned the question of whether both arrows of motors A and C had to face in the same direction or in opposite directions, in order for the robot to move forward. It must be noted here that the "trial and error" method could well be employed here, namely to put the existing robot in motion and see what would happen. However, all evaluators disagreed with this approach, as the focus of the evaluation is to assess the usability of the software and grade the *intuitiveness* and *transparency* of the interface, two notions of paramount importance as regards to usability. Of course, such a trial-and-error method would also help the user to answer this question, yet the usability problem would remain. Therefore, the robot never came into action and the question remained. The evaluators decided to seek advice on this topic in the various help utilities of the environment, seeking thus to assess other aspects of the "extended interface" of the RoboLab environment, as well. However, this was done later with the heuristic approach and yielded no satisfactory results, neither in the on-line help nor in the manual.

heuristic evaluation results

The results from this part of the session were grouped firstly according to the used criteria. A number of usability problems have been recorded and classified, however, there were also fruitful debates for the correct classification of every problem. The final usability problem list showed seven problems for the first heuristic, while for the rest of the heuristics the evaluators noted far fewer problems, from 0 to 3. The evaluators considered as most important the violations at the first and the fifth heuristic.

In more detail, the first usability heuristic argues for "simple and natural dialogue, and aesthetic and minimalist design." However, the evaluators pinpointed following problems:

- The window "Front Panel," which appears at the top of the screen has no obvious use, thus confusing the user.
- The help windows are not easy to use. For example, they are editable, which makes obviously no sense for a help window!
- The system does not inform the user on the available functionality in any way; the user must instead discover it for himself.
- The graphic design was considered to be extremely poor for this user age. It could be a little fancier.
- Though the user cannot get lost, due to the limited depth the window titles do not help from a navigational perspective.
- Where the dialog "Save changes?" appears, it includes no "cancel" choice, or it is dimmed.
- If the user wants to delete an icon, a dialog appears asking him if he wants to save the changes he is going to discard!
- As regards the fifth usability heuristic ("consistency and standards"), the evaluators pinpointed that:
- In the functions palette, when the user clicks on the three bottom icons, a sub palette appears, with no intuitive manipulation schema. It shrinks, and in order to recall the mother palette, the user has to click on the up arrow. The evaluators consider that this arrow does not imply its functionality, e.g., to unfold again the shrunk palette, one must click on the up arrow.
- There are not some standards followed, in general. There is no pop-up help, the cursor is hand shaped throughout the whole interface (even if the particular area is not clickable), right click is not always context sensitive, etc.
- Moreover, there is no consistency between Pilot and Inventor environments, and the user has to learn them all over again.

Statistics

The statistical elaboration was the next step. This sector concerns only the egw part of the evaluation, as the heuristic approach is a quantitative one and can hardly undergo any statistical elaboration.

For every question of the questionnaire, the mean value and the standard deviation has been calculated. The results showed a certain consensus of the evaluators for most of the questions. There was only one exception: the activation of the touch sensor of the RCX motor when encountering an obstacle. Herein, all evaluators gave negative grades (low mean values). They also stated that the user will not believe he is coming closer to his goal, as the user neither clearly knows what exactly a "sensor" does, nor what the expected feedback is. However, there was no consensus. One evaluator gave very high grades for

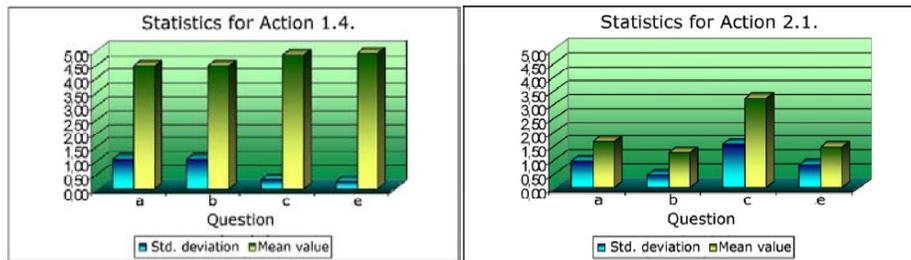


Fig. 2. Comparison of a “normal” to a “debatable” assessment.

this action, stating that even a 12-year-old user knows roughly the use of a sensor and would perform this action without problem.

This debate is visible in following Figure 2. The reference action (Action 1.4.) graphically shows both high mean values, which means that the evaluators considered this action as “non-problematic,” and low standard deviations, indicating a unanimity. By contrast, Action 2.1. shows low mean values and higher standard deviations. Therefore, they “didn’t agree that there wasn’t a problem,” a well-known debate phenomenon in usability evaluations which always alarms system designers to immediately confront this aspect.

DISCUSSION AND FUTURE WORK

The evaluation concerning the LEGO RoboLab interface was performed on two distinct levels. One was the evaluation of the environment per se, and the second was the application of two expert-based evaluation methodologies on the same software piece.

The RoboLab environment

Despite the many observations and comments from the evaluators, the RoboLab environment has been considered successful. The aim of every evaluation is to discover as many deficiencies and shortcomings of the considered interface as possible, in order to assist the redesign and improvement process. In this point of view, this session has tried to unveil all *possible* problems. However, many studies (such as Karoulis and Pombortsis [2000]) show that expert evaluators are generally more rigorous than real users. In other words, many of the problems cited by the evaluators will be in actual practice no problem at all, and vice versa; the users will also encounter problems which the experts did not perceive at all. This is an expected and known side effect, so the correct solution in this regard seems to be the combination of the expert-based session with an empirical session (user-based) to elicit valuable results.

Another aspect (which is hardly depicted in the diagrams) is that the evaluators considered the interface to provide a steep learning line; in other words, they believe the users will encounter the stated usability problems at the first or second interaction with the particular action. After this initial phase, the estimation of the evaluators was that the RoboLab environment would become transparent enough to its users so as to assist them in their tasks.

However, the first and second “feel and look” is very critical, especially for users of this age, so that the stated usability problems are magnified in significance, and should be treated accordingly.

The cgw and the heuristic evaluation

One unexpected result was the encountered inefficiency of the cgw to perform adequately

in this study. The explanation is not yet clear; it is believed to be in the enhanced difficulty of some actions during the first contact with the interface. On the other hand, the heuristic evaluation did not provide significantly different results. Indeed, the same aspects pinpointed in Pilot as usability problems by the cgw were also pinpointed as such by the heuristic approach in Investigator. The unwillingness of the evaluators to continue with the cgw could also be attributable to the enhanced difficulty and slower pace of this method. However, this point remains open for future investigation.

CONCLUSION

In conclusion, this evaluation was very fruitful. A number of usability problems were unveiled in a popular interface, which can assist in its improvement. On the other hand, two expert based methodologies have been tested and compared on the same software piece, and have been proven to perform satisfactorily. Clear indicators that both methods performed satisfactorily are the resulting 28 major and minor usability problems that have been recorded. In addition, the methods consumed very few resources, thus proving their good cost/performance factor. In an attempt to generalize this result, one could argue that the expert-based methodology constitutes a potential candidate for any educational evaluation where there are resource restrictions. However, one must keep in mind that the combination of the expert-based approaches with empirical ones always provides the best results, as already stated. Expert evaluators tend to over-estimate minor usability problems and to neglect important aspects (such as language usage), while users tend to pinpoint critical problems (such as mislabeling), and to under-estimate technical maturity. Therefore, a “combinatory evaluation” as proposed in Karoulis and Pombortsis [2000] always provides the most accurate and reliable results.

ACKNOWLEDGMENT

The author wishes to thank LEGO and the evaluator team for their help during this study. A special thank you must go to Olga Timcenko of LEGO systems for her valuable assistance during the whole procedure, as well as to the EU-funded Network of Excellence “Kaleidoscope,” Jointly Executed Research Project (JERP): “Interaction between learner’s internal and external representations in multimedia environments,” which provided the framework for this study.

REFERENCES

- ACKERMANN, E. 2003. Hidden drivers of pedagogic transactions: Teachers as clinicians and designers. In *Proceedings of the 9th EuroLOGO Conference* (Porto, Portugal, Aug. 2003), 29-37.
- AEDO, I., CATENAZZI, N., AND DIAZ, P. 1996. The evaluation of a hypermedia learning environment: The CESAR experience. *Journal of Educational Multimedia & Hypermedia* 5,1 (1996), 49-72.
- CATENAZZI, N., AEDO, I., DIAZ, P., AND SOMMARUGA, L. 1997. The evaluation of electronic book guidelines from two practical experiences. *Journal of Educational Multimedia & Hypermedia* 6,1 (1997), 91-114.
- ISO (1998) ISO 9241. International Standardization Organization. Ergonomic requirements for office work with visual display terminals (VDT's), Part 10, Dialogue Principles.
- KARAT, C., CAMPBELL, R., AND FIEGEL, T. 1992. Comparison of empirical testing and walkthrough methods in user interface evaluation. In *Proceedings of the ACM CHI '92 Conference* (Monterey, CA, May 3-7). ACM, New York, 397-404.
- KAROULIS, A., DEMETRIADES, S., AND POMBORTSIS, A. 2000. The cognitive graphical jogthrough – An evaluation method with assessment capabilities. In *Proceedings of the Applied Informatics 2000 Conference* (Innsbruck, Austria, Feb. 14-17). IASTED/ACTA, Anaheim, CA, 369-373.
- KAROULIS, A., AND POMBORTSIS, A. 2000. Evaluating the usability of multimedia educational software for use in the classroom using a “combinatory evaluation” approach. In *Proceedings of the EDEN 4th Open Classroom Conference* (Barcelona, Spain, Nov. 20-21).
- KAROULIS, A., AND POMBORTSIS, A. 2002. Heuristic evaluation of web-based ODL programs. In *Usability Evaluation of On-Line Learning Programs*, Claude Ghaoui (ed.). Information Science, Hershey, PA, 89-109.
- KAROULIS, A., DEMETRIADES, S., AND POMBORTSIS, A. 2005a. Cognitive graphical walkthrough: An interface evaluation method. In *Encyclopedia of HCI*. IDEA Group (to appear).

- LEWIS, C., AND RIEMAN, J. 1994. Task-centered user interface design - A practical introduction. Retrieved Sept.9, 2003 from <ftp://ftp.cs.colorado.edu/pub/cs/distribs/clewis/HCI-Design-Book/>
- LEWIS, C., POLSON, P., WHARTON, C., AND RIEMAN, J. 1990. Testing a walkthrough methodology for theory-based design of walk-up-and-use interfaces. In *Proceedings of the ACM CHI '90 Conference*, (Seattle, WA., Apr. 1-5), 235-242.
- MAIER, N.R.F. 1931. Reasoning in humans: II. The solution of a problem and its appearance in consciousness. *Journal of Comparative Psychology* 12 (1931), 181-194.
- MINDELL, D., BELAND, C., WESLEY, C., CLARKE, D., PARK, R., AND TRUPIANO, M. 2000. LEGO mindstorms, the structure of an engineering (r)evolution, 6.933J Structure of Engineering Revolutions, <http://web.mit.edu/6.933/www/Fall2000/LegoMindstorms.pdf>
- NIELSEN, J. 1993. *Usability Engineering*. Academic Press, San Diego, CA.
- NIELSEN, J. 1994. Heuristic evaluation. In *Usability Inspection Methods*, J. Nielsen and R.L. Mack (eds.), John Wiley & Sons, New York.
- NIELSEN, J., AND MOLICH, R. 1990. Heuristic evaluation of user interfaces, In *Proceedings of the Computer-Human Interaction Conference* (Seattle, WA, Apr. 1-5), 249-256.
- PAPERT, S. 1980. *Mindstorms: Children, Computers and Powerful Ideas*. Basic Books, New York.
- PIAGET, J. 1952. *The Origins of Intelligence in Children*. International University Press, New York.
- POLSON, P.G., LEWIS, C., RIEMAN, J., AND WARTON, C. 1992. Cognitive walkthroughs: A method for theory-based evaluation of user interfaces. *International Journal of Man-Machine Studies* 36 (1992), 741-773.
- PREECE, J., ROGERS, Y., SHARP, H., BENYON, D., HOLLAND, S., AND CAREY, T. 1994. *Human-Computer Interaction*. Addison-Wesley, Reading, MA.
- REEVES, T.C. 1993. Evaluating technology-based learning. In *The ASTD Handbook of Instructional Technology*, G.M. Piskurich (ed). McGraw-Hill, New York. 15.1-15.32.
- RESNICK, M., BERG, R., EISENBERG, M., AND TURKLE, S. 1997. Beyond black boxes: Bringing transparency and aesthetics back to scientific instruments. <http://el.www.media.mit.edu/groups/el/papers/mres/blackbox/proposal.html>.
- RIEMAN, J., FRANZKE, M., AND REDMILES, D. 1995. Usability evaluation with the cognitive walkthrough. In *Proceedings of the CHI-95 Conference* (May 7-11), ACM, New York, 387-388.
- ROWLEY, D. AND RHOADES, D. 1992. The cognitive jogthrough: A fast-paced user interface evaluation procedure. In *Proceedings of the ACM CHI '92 Conference* (Monterey, CA, May 3-7), 389-395.
- SMITH, S. AND MOSIER, J. 1986. Design guidelines for designing user interface software. The MITRE Corp. <ftp://ftp.cis.ohio-state.edu/pub/hci/Guidelines>.
- TIMCENKO, O. 2005. Understanding iconic representations of complex programming structures in graphical programming language for tweens and teenagers. In *Interaction between learner's internal and external representations in multimedia environments*, res.REP., Kaleidoscope NoE JEIRP, 39-44.
- VYGOTSKY, L.S. 1936/1978. *Mind and Society: The Development of Higher Mental Processes*. Harvard University Press, Cambridge, MA.
- WHARTON, C., RIEMAN, J., LEWIS, C., AND POLSON, P. 1994. The cognitive walkthrough: A practitioner's guide. In *Usability Inspection Methods*, J. Nielsen and R.L. Mack (eds). John Wiley & Sons, New York, 105-140.

Received September 2005; revised January 2006; accepted February 2006