Evaluating the LEGO interface with experts

Athanasis Karoulis[1], Eleni Valsamidou[1], Savvas Demetriadis[1] and Olga Timcenko[2]
[1]Aristotle University of Thessaloniki, Greece
[karoulis, evalsami, sav63]@csd.auth.gr
[2]LEGO Innovation, LEGO Systems A/S, Billund, Denmark
olga.timcenko@europe.lego.com

Abstract. This work presents the results from a joint usability study concerning the LEGO programming environment, called "RoboLab". LEGO has already performed numerous evaluations concerning 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 the combination of their results. So, two expert-based methods have been applied, a Cognitive Graphical Walktrough and a Heuristic Evaluation, at the Department of Informatics of the Aristotle University. The results of the study unveiled some deficiencies of the interface and some limitations of the employed methods as well.

Keywords: Interface evaluation, usability, LEGO®, RoboLab®.

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. Design and construction process accepts the constructivist view of children development and learning, combined with flow theory, presented in works of Vygotsky (1936/1978), Piaget (1952), Papert (1980), and Csikszentmihalyi (1996).

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 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, and encourage them to learn.

However, when implemented to more complex environments, such as programming, only 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 when you explain something to a child, you prevent her from discovering it". But Edith Ackermann (2003) adds: "Yet at the same time, each time you fail to give wanted-for-help when needed, you prevent 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. And 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". So, one could argue that this is also one of the desired properties of educational software – to offer the rich environment for experimenting, providing in parallel enough and just-in-time information and guidance.

Description of the software

LEGO Company accepts this constructivist learning philosophy while developing both toys and educational materials for children. All LEGO products are designed with belief that:

- Children learn best by doing or making, and
- Learning should be an enjoyable, as well as an educational, experience.

Thus, in close cooperation of LEGO Company and MIT Media Lab, the LEGO Mindstorms RCX programmable brick was developed as a central part of a robotic construction set, and 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 3 motors and take input from up to 3 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 environment or after-school activities. 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 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 8 and up, LEGO was aiming to bring the best values of constructionist learning approach to children and their teachers.

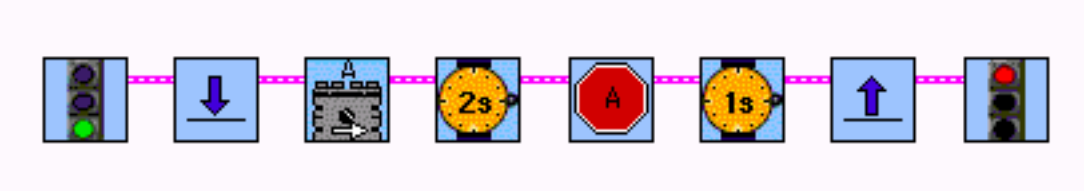


Figure 1. An example of a Mindstorm® program.

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

- Users first build their invention using the RCX and the LEGO elements included in the LEGO sets.

- Then they create a program for their invention using RoboLab
- The program has to be downloaded to the RCX using a special Infrared Transmitter.
- Their fully autonomous creation can now be tested in direct interaction with the environment, and eventually modified/improved, starting again from the first or second bullet point.

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

- Pilot is the basic environment where programs are built using a click-and choose interface.
- Inventor provides a more open-ended, icon-based environment.
- 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

Programming is a very abstract human activity, which requires training. In the case studied, all RoboLab programming environments are used for programming microprocessor-controlled LEGO robots build of LEGO bricks, together with motors and different sensors. Despite of success in sales, there are reports that some aspects of both Mindstorms and RoboLab are very difficult to grasp both for children and their instructors. So, an interesting question emerges here, 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 way of providing help for different functions could also need to be an object of research, as existing one, in spite of providing help in several levels of details, depending on users requests, proved to be not very clear for novice users.

Ideally, as a practical result, this should lead to suggestion of redesign of some parts of LEGO programming environment for teenage children. On more abstract level, the result should be some guidelines 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 applied methodologies as well as the evaluation procedure are described in the following.

Usability Evaluation

What exactly is «usability evaluation»? 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). 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 and 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, which in its turn supports our claim that usability affects learnability. Secondly, 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.

We distinguish two major evaluation categories: formative and summative evaluation (Scriven, 1976). The former is conducted during the design and construction phase, while the latter is conducted after the product has reached the end user. The results and conclusions of the former are used mainly for bug-fixing and improving the characteristics of the interface (detecting problems and shortcomings), while the results and conclusions of the latter are used to improve the interface as a whole and meet more user needs in a following upgrade. In more detail:

- During the formative phase, data are collected that allow designers to improve the programme/courseware/software, to ensure it achieves its full potential. The formative phase is intensive, with small numbers of users or testers or evaluators (depending on the chosen evaluation method), usually working in pairs or small groups, with frequent reports to the design team - an iterative design-test-redesign procedure, focusing on the materials design.
- The summative phase tests the success of the program, investigating the contextual conditions that achieve best results, and providing costing models of usage. The summative phase is extensive over time and place, large scale, with occasional reports, focusing on the implementation of the materials.

Going one step further we distinguish four main evaluation methods (Benyon et al, 1990):

- Analytic evaluation. It uses a formal or semi-formal description of the interface in order to predict users' performance in terms of the physical and cognitive operations that must be performed.

- Expert evaluation. Experts are asked to judge the system and identify the potential usability problems, taking the role of less experienced users.
- Empirical evaluation. Its purpose is to collect data concerning the user's behavior while using a system (observational evaluation), or involving the use of interviews or questionnaires with the purpose of eliciting users' subjective opinions and understanding of the interface (survey evaluation).
- Experimental evaluation. The evaluator can manipulate a number of factors associated with the interface design and study their effects on various aspects of users' performance.

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). Independently from the chosen evaluation method, every evaluation consists of three basic phases:

- A preparation phase during which we define the evaluation objective, selecting the appropriate method, selecting the evaluators (experts or users), arranging the questionnaire (if any), setting up the equipment etc.
- An evaluation phase where we conduct the evaluation procedure itself and
- A result interpretation phase during which the recorded data are elaborated and the results and conclusions are written and discussed.

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 & Rieman (1994) «you can't really tell how good or bad your interface is going to be without getting people to use it». This phrase expresses the broad belief that user testing is inevitable in order to assess an interface. Why then, don't we always simply apply empirical evaluations but explore other 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 disadvantageous issues, while, on the other hand, expert-based approaches have meanwhile matured enough to provide a good alternative.

The first main disadvantage of the empirical studies is the personal bias of the subjects. It is important to understand that test users can not 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; users (for various reasons) often can not 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 is a difficult 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 for us 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 this was what really happened. So they could not and did not tell him what feature of the situation really helped them solve the problem.

Lewis & Rieman (1994) give the three prerequisites for an empirical evaluation:

- People, real users, if possible, 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 the 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. They provide a crucial advantage which makes them more affordable compared to the empirical ones: it is in general easier and cheaper to find out experts eager to perform the evaluation than users. The main idea is that experts from different cognitive domains (certainly one from the domain of HCI and one from the cognitive domain under evaluation at least), are asked to judge the interface, each one from his own point of view. It is important that they are all 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 approach is that it can be implemented very early in the design cycle, even on paper mock-ups. Experts’ expertise allows them to understand the functionality of the system under construction, even if they lack 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 find out representative users, like the ones that will use the system in real situations. As Lewis & 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»? So, the first step for a user-centered interface design is that the design team observes how real users in their real environment and perform their work using the product under evaluation. User-based evaluation must also simulate the real user environment in a real working place (not in the laboratory), with representative users (not simply available users). This constitutes the second main disadvantage of the empirical evaluations, as it is difficult to find representative users to evaluate the system under real work conditions, which consequently means increased evaluation costs and a more difficult implementation of the evaluation session. Yet, this approach would be the optimal one for evaluating user interfaces designs, assuming one can recruit real users eager to evaluate an environment under construction. Based on our experience we claim that this goal is difficult to achieve, if at all.

On the other hand, expert-based evaluations overcome these limitations by replacing users with experts and although they may provide qualitatively poorer results, they do it with far lower costs and implementation difficulties, thus achieving a much higher performance/cost factor. So, we claim that an expert-based evaluation approach of an environment is more likely to be implemented in most cases. To conclude, Table 1 summarizes the main advantages and disadvantages of both approaches.

Table 1: Advantages and disadvantages of expert and user based evaluations

	Expert-based evaluation	***Usability testing***
Pro	• Cheap methods • Quick • Can early be applied in the design cycle • Easy to prepare and to conduct • Applicable at all stages • Can assess the severity of the problem • Good effectiveness/cost = efficiency factor	• Unveils problems of real users • Can pinpoint nearly all problems • Efficient, even in complex interfaces • Direct description of the problem • Absolutely necessary, if one looks for valid results
Contra	• Does not unveil all problems • Needs experienced evaluators • Difficulty in proposing solutions • Often loses the picture of the «whole» • Evaluators often forget or can not play at all the role of the user • HCI experts and domain experts are indispensable	• Expensive and difficult to materialize • Demands representative users • Difficulty to find the subjects • Users tend to be in confusion about the severity of the problems • Subjects can be biased • Can be applied after a certain level of product development and completion

The Cognitive Graphical Walkthrough

The Cognitive Graphical Jogthrough method (described in detail in Demetriades et al., 1999 and Karoulis et al., 2000) belongs to the expert-based evaluation methodologies. Its origin is in the 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 & Rhoades, 1992; Aedo et

al., 1996; Catenazzi et al., 1997). The main idea in the expert-based evaluation is to present the interface supported tasks to a group of four to six experts who will play the role of would-be-users and try 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 employ an appropriately structured questionnaire to record the evaluators' ratings. They are also characterized as "cognitive" to denote 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 completed 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 in the questionnaire for 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, however with the 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. This, however, limits the range of the application of the method. So, we tried to introduce the time factor to provide a means for recording increasing users' experience while working with 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 these 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

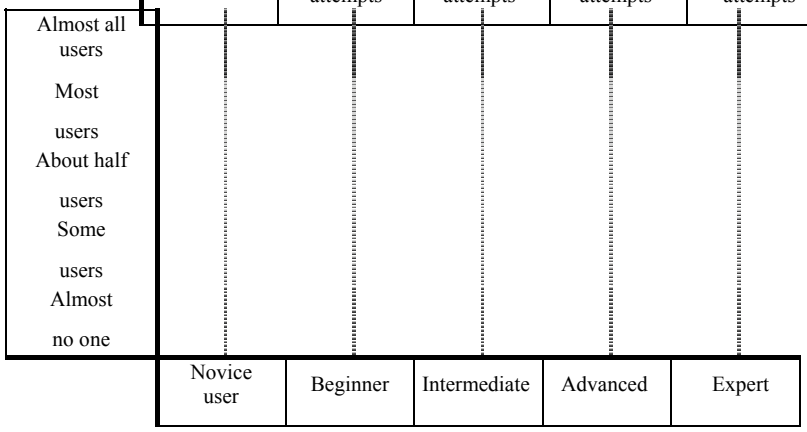
that the method is not only generally applicable, but also, under certain circumstances, it approaches the performance of empirical evaluation methods.

The core issue in the modified method of the CGW/CGJ is the different types of diagrams where the evaluators can note down their assessments. We suggest two main types of diagrams:

(A)

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

(B)



- A. The «digital» form of the diagram (utilizing boxes) and «countable» assessment (horizontal axis in attempts)
- B. The «analog» type of the diagram (utilizing lines) and «internal» assessment (the experience of the user is assessed within the same interface)

The differentiation of the diagrams refers mainly to their usability, as perceived by the evaluators. We were mainly concerned to find the more easy to use diagram form.

The Heuristic Evaluation

Maybe 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. These people can be users or experts in the particular domain. So we distinguish, as already explained, between user-based evaluations, known as "empirical evaluations" and expert based evaluations. However, at this point we have to make some clarifications about the notion of the user. Referring to the web we consider de facto that all involved persons are at the same time users, even if they deal with it as evaluators. "Real" user based evaluations assume that the users use the entity under consideration under conditions as

authentic as possible, while, simultaneously, observations are collected about the evaluation procedure. However, as already mentioned, in the evaluation case under consideration there are some criteria set, which have to be followed during the evaluation. In the web evaluation approach, where every evaluator performs on his/her own, these are sometimes assessed without real use of the entity, but the user or the evaluator usually utilizes the conceptual model, as described by Norman (1988) for the entity and the way it performs, simulates its performance in his mind and concludes for every criterion. Alternatively he may use the entity, not to produce real work, but in order to assess the application of the criteria. So we can argue that in any case it is about an expert based evaluation approach, even if users are involved, as long as they are concerned about answering according to the set criteria.

What can we evaluate in this way? Makrakis (1999) says everything has to do with:

- The design
- The organization
- The function
- The result of the entity under consideration

To evaluate the above he assumes as necessary:

(a) Defining the evaluation axis. They are the general questions set to be answered through the evaluation. They emerge:

- 1) From what we need to have evaluated and
- 2) What the evaluation method allows us.

These axis are the basic principles of the theoretical framework of the evaluation.

(b) Defining detailed criteria. They are the concrete questions, usually measurable variables (so they are components of the methodological framework) to assess the axis. However, a number of problems arise from this approach. It provides all the disadvantages of the expert-based evaluations (Karat et al., 1992; Nielsen, 1993; Karoulis et al., 2000). The axes and criteria list may become very long (Lewis & Rieman, 1994; Nielsen, 1993). For example, the full interface usability criteria list suggested by Smith & Mosier (1986) includes 944 criteria. The evaluators' expertise plays a major role. (Lewis & Rieman, 1994; Nielsen, 1993). We discuss this issue in detail later.

To handle these problems Jacob Nielsen and Rolf Molich started their research in 1988 and in 1990 they presented the “heuristic evaluation” (Nielsen & Molich, 1990). The basic point was the reduction of the set criteria to just a few, at the same time being broadly applicable and generally agreed; simultaneously augmenting the evaluators' expertise, and consequently their reliability. These “heuristic rules” or “heuristics” derived from studies, criteria lists, on 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 following heuristics, slightly modified and reorganized by us:

- Simple and natural dialog and aesthetic and minimalistic design. Dialogs should not contain information, which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.
- Visibility of the system status – provide feedback: The system should always keep users informed about what is going on, through appropriate feedback within reasonable time
- Speak the users' language: match between system and real world. The system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system oriented terms. Follow real world conventions, making information appear in a natural and logical order.
- Minimize the users' cognitive load: recognition rather than recall. Make objects, actions and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.
- Consistency and standards: Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.
- Flexibility and efficiency of use – provide shortcuts. Accelerators - unseen by the novice user - may often speed up the interaction for the expert user to such an extent that the system can cater for both inexperienced and experienced users. Allow users to tailor frequent actions
- Support users' control and freedom: Users often choose system functions by mistake and will need a clearly marked 'emergency exit' to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.
- Prevent from errors: Even better than good error messages is a careful design, which prevents a problem from occurring in the first place.
- Help users recognize, diagnose and recover from errors with constructive error messages. Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution
- Help and documentation: Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

In the heuristic evaluation we make two assumptions from the beginning (Lewis & Rieman, 1994), which have evolved from the observations of the application of the method:

- No distinct evaluator can find all the heuristically identifiable usability problems.
- Different evaluators find different problems.

The appropriate number of evaluators and their expertise are an issue of great importance. Researches up to now (Nielsen & Molich, 1990; Nielsen, 1992; Nielsen, 1993) have shown that:

- Simple or novice evaluators. They do not perform very well. We need 15 evaluators to find out 75% of the heuristically identifiable problems. These are problems that heuristic evaluation can point out. As already mentioned and for different reasons, there are problems that are overlooked using this kind of evaluation. The research has shown that 5 of these simple evaluators can pinpoint only 50% of the total problems.
- HCI experts (regular specialists). They perform significantly better: 3 to 5 of such evaluators can point out 75% of the heuristically identifiable problems and among them all major problems of the interface.
- Double experts (double specialists). These are HCI experts with additional expertise on the subject matter, e.g. educators for educational interfaces. The research has shown that 2-3 of them can point out the same percentage as the HCI experts.

The following figure by Nielsen (1992) summarizes these statements.

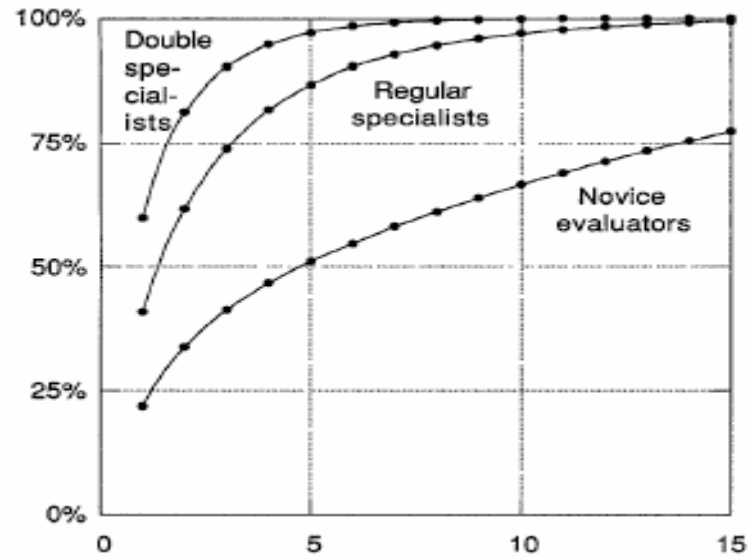


Figure 2: Expertise of the evaluators

It is obvious that there is no great difference between experts and double experts in seeking the involvement of the latter in the evaluation. However, there is a very distinct difference between experts and simple evaluators. As we can see in the figure, to point out 75% of the heuristically identifiable problems we need 15 simple evaluators, while 3 expert evaluators bring the same result.

The method refers mainly to traditional formative human-computer interface evaluation, yet a number of studies (Nielsen & Norman, 2000; Instone, 1997; Levi & Conrad, 1996) have proven its easy adaptability to the evaluation of web sites as well.

The study

Preparation of the evaluation session

The first step in the evaluation procedure of the CGW (Cognitive Graphical Walkthrough) is the decision on the tasks and the 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. Following procedure has been set here: the robot must go straight until it encounters an obstacle. Then, it has to go for a while backwards, turn (left or right) and move again forwards. Accordingly, this procedure has been analyzed in tasks as follows:

- Forwards
- Obstacle
- Backwards
- Sensor
- Turn
- Straight

Accordingly, the actions for every task have been defined. In this step, the programming environment, as described in the manual has been intensive considered. So, we have following final description of the tasks and actions:

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

Accordingly, a «note to the evaluators» was distributed, explaining the basics of the method. This has been 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.

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. In following, the session is described in detail.

Part I: Cognitive Graphical Walkthrough

The session was designed to be performed 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 witch a novice user could rely on. 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 the tasks already described. The set of questions was the one proposed in Karoulis et al. (2005):

- 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 on the questions, the evaluators used the already presented diagrams.

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. Their main argument direction was that the method aims to evaluate a walk-up-and-use interface, which is an interface designed to be easily used by any inexperienced user (Lewis et al., 1990; Lewis & 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. So, they considered that the initial contact of the user with the interface would not be successful, which is a prerequisite to apply the CGW method. The evaluators proposed the Heuristic Evaluation as a reasonable 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 proposed in Karoulis & Pombortsis (2002) slight modified heuristic list, as well as an «evaluator's notebook». As the evaluators were also experienced in this evaluation method, no further explanations were needed and the second part of the session begun.

Results

CGW results

There was from the beginning an extensive discussion on the direction of the arrow depicted on the motor icon, and in particular, in witch direction would the motor spin. The debate concerned the question 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 has to 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 aim 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. So, the robot came never in action and the question remained. The evaluators decided to seek for advice on this topic in the various help utilities of the environment, seeking thus to assess also other aspects of the «extended interface» of the RoboLab environment. However, this was done later with the heuristic approach and yielded no satisfactory results (in the on-line help or in the manual), as will be described later.

Further comments of the evaluators during this part of the session were as follows:

- Everywhere on the interface the hand-shaped cursor remains, thus confusing the user, as it is known that the hand-shaped cursor implies interaction with the system.
- By opening the Pilot level 4, a default program is presented to aid the user in his/her work. The evaluators considered this initial proposal as mediocre helpful. However, as users preferences vary strongly, they considered that some users would find it helpful, yet not the majority.
- The design of the icons does not imply their functionality, they are not intuitive.
- They proposed an interactive appearance of the RCX brick in a corner of the screen, depicting clearly its state. For example, if the users set the power of the motor on 2, the «wheel» would turn in the corresponding direction (this would also answer the former question concerning the spinning direction).

- Some colors (e.g. black letters on blue background) make reading difficult.
- There was a suggestion that the flow line could be vertical instead of horizontal. This could be combined with a palette containing the icons and a drag-n-drop facility. In addition, all modifiers should be operated via sliders, levels or buttons.
- Another concern was whether the notion of the «sensor» was familiar to this user group. If one should suggest so, then the profile of the «mean» user of this environment levels up significant, especially if one is talking of the Pilot environment.

Heuristic Evaluation results

The results from this part of the session are presented according to the used criteria. We emphasize that this part concerns the Inventor environment.

1. Simple and natural dialogue, and aesthetic and minimalist design

- The window "Front Panel", which appears at the top of the screen has no obvious use, confusing thus 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 himself.
- The graphic design was considered to be extremely poor for this user age. It could be a little fancier.
- Despite that 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!

2. Visibility of system status – provide feedback

- There were no notes on this heuristic.

3. Speak the users' language: match between system and the real world

- Some dialogs are difficult to understand, even by the evaluators.
- Some icons also.

4. Minimize the users' memory load: recognition rather than recall

- After enough practice, the interface here becomes transparent. The user can concentrate on his work with no problems, besides the programming task problems he will encounter.

5. Consistency and standards

- 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 for calling again 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, right click is not always context sensitive etc.
- Moreover, there is no consistency between Pilot and Inventor environments, and the user has to learn all over again.

6. Flexibility and efficiency of use – provide shortcuts

- Besides the aforementioned problem, concerning the sub palette, this heuristic is considered to work well in the interface.

7. Help users recognize, diagnose, and recover from errors with good error messages

- There were not comments on this heuristic; however there is in the former criteria a number of recorded comments concerning the messages and dialogs.

8. User control and freedom: clearly marked “exits”, undo and redo support.

- The already mentioned dialog boxes with no “cancel” choice.
- In general, this heuristic is supported satisfactorily only by the undo function. There are no clear exits.

9. Error prevention

- This heuristic was considered not to function properly. The evaluators believe that the user is not protected against wrong manipulations and actions.

10. Help and documentation

- There were not comments on this heuristic; however there is in the former criteria a number of recorded comments concerning the help facilities.

Statistics

This section concerns only the CGW part of the evaluation, as the Heuristic approach is a more qualitative one and can hardly undergo any statistical elaboration. In the subsequently presented statistics, the question 4 («closer to goal», yes / no) is not included, as it can also not be elaborated. However, where there were comments on it, they are presented.

Task 1: Forward

1.1. Power MOTOR A

According to the statistical elaboration, the evaluators consider this action to be problematic, due to fact of the spinning direction of the motor. The mean values and the standard deviation of the evaluators' opinions are presented in following diagram.

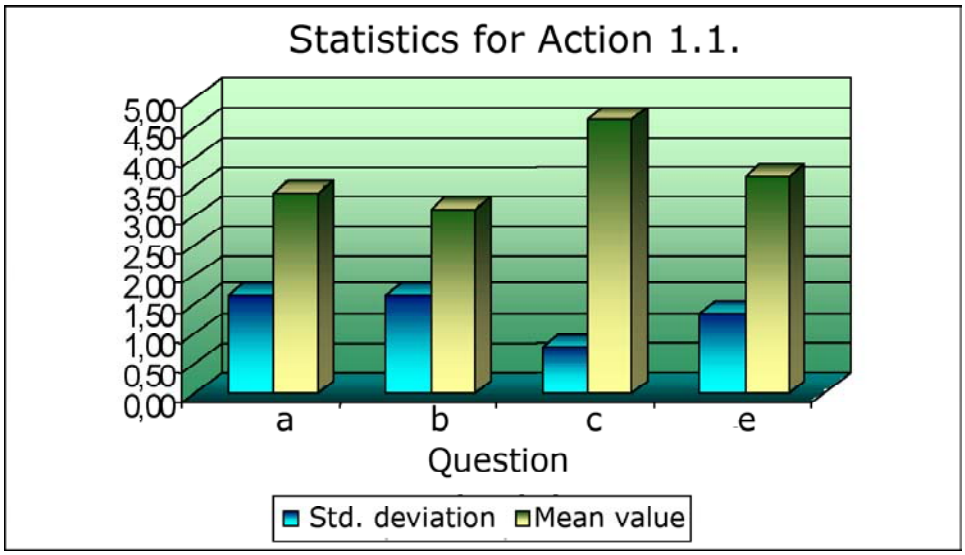


Figure 3. Mean Value and Standard Deviation for Action 1.1.

1.2. Power level for MOTOR A

This action is positive assessed and the evaluators believe there is no usability problem. The mean values and the standard deviation of the evaluators' opinions are presented in following diagram.

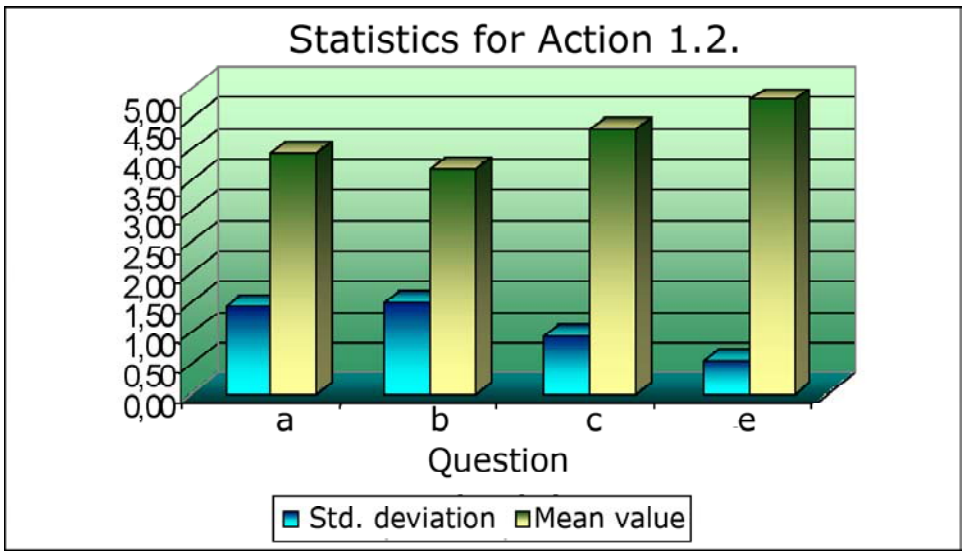


Figure 4. Mean Value and Standard Deviation for Action 1.2.

1.3. Power MOTOR C

For this action the evaluators agreed that it is not obvious that this actions brings the user closer to his goal. This fact bases on the long debate on the spinning of the motor, so they believe the user will not be

sure unless he downloads the program to the brick. For the rest of the questions, the mean values and the standard deviation of the evaluators' opinions are presented in following diagram.

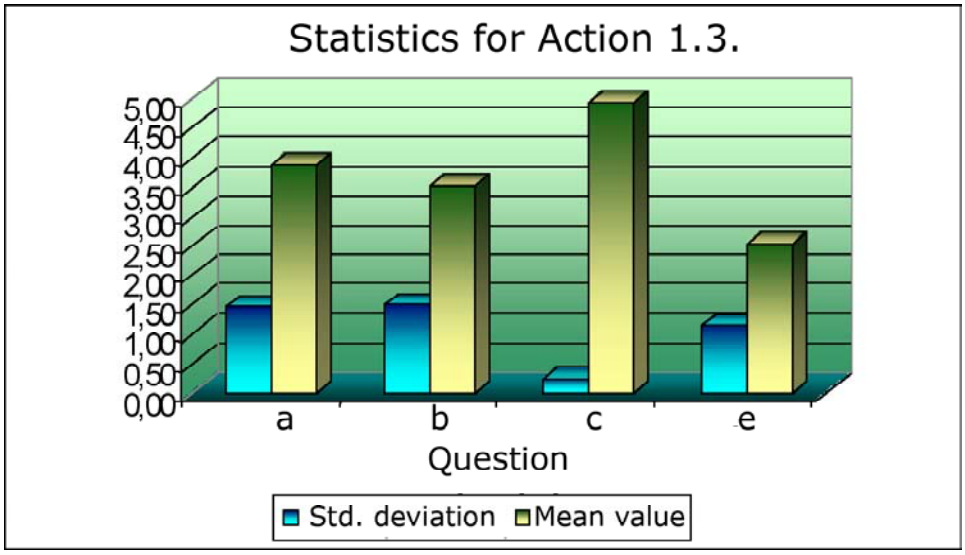


Figure 5. Mean Value and Standard Deviation for Action 1.3.

1.4. Power level for MOTOR C

This action is especial positive assessed by the evaluators: high mean values, with low standard deviation as well, which means consensus. The fact for that is that the evaluators believe the user to remember that the same action has been encountered before, so they are more likely to remember how to perform it. The mean values and the standard deviation of the evaluators' opinions are presented in following diagram.

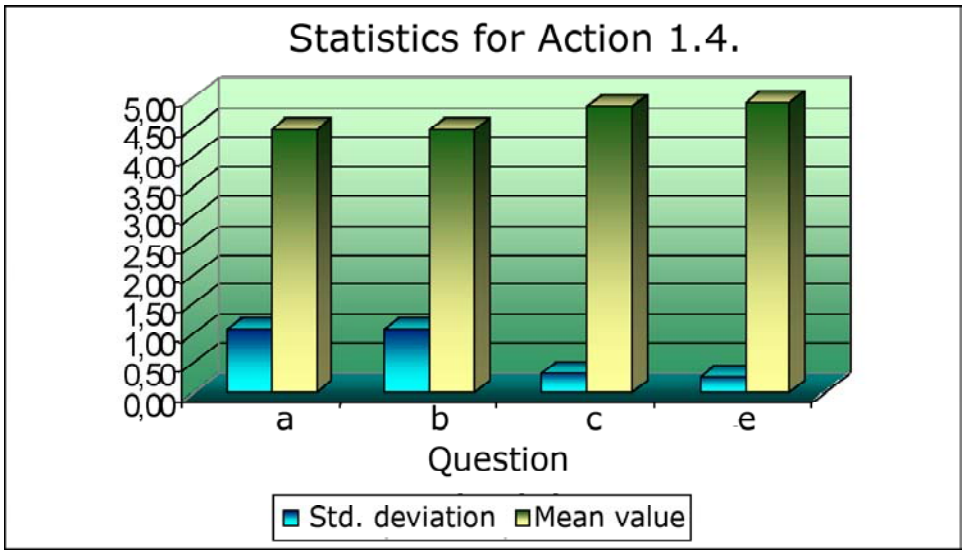


Figure 6. Mean Value and Standard Deviation for Action 1.4.

Task 2: Obstacle

2.1. Activation of the touch sensor

There is a usability problem. 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 does not clearly knows what exactly a «sensor» does, and which is the expected feedback. However, there was not consensus here. One evaluator gave very high grades for this action, stating that even a 12-year-old user knows roughly the use of a sensor and would perform this action without problem. In particular, following lines show the opinions of the evaluators.

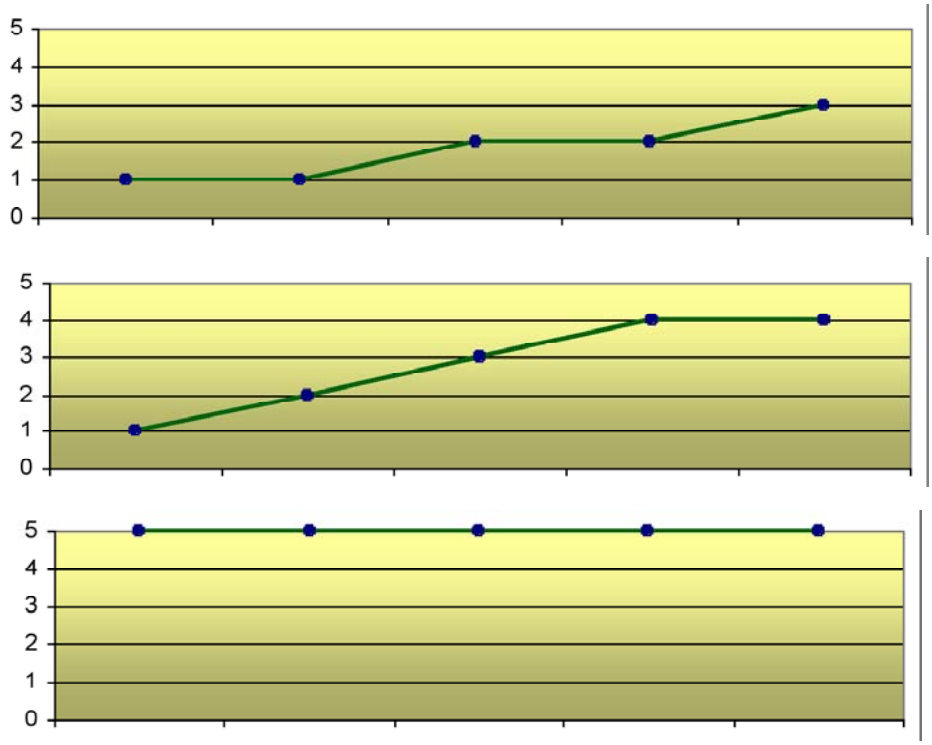


Figure 7. Ratings of the evaluators for action 2.1.

The mean values and the standard deviation of the evaluators' opinions are presented in following diagram.

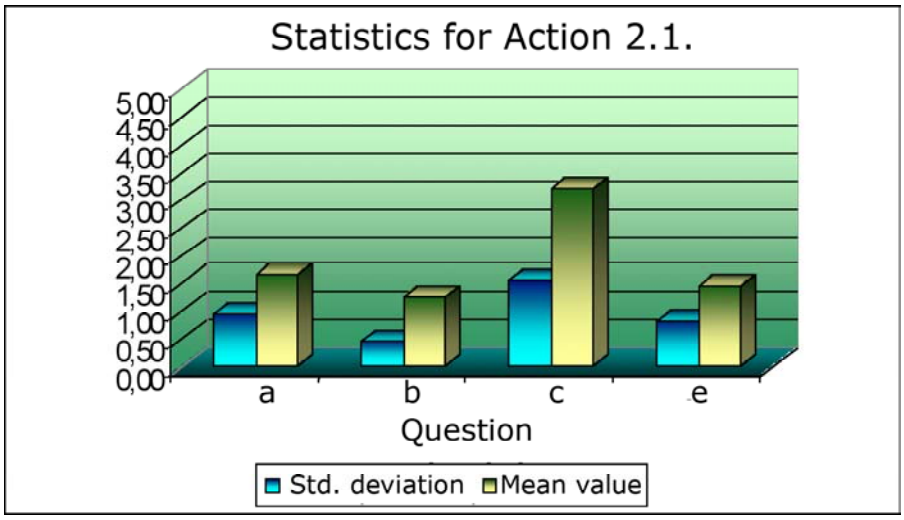


Figure 8. Mean Value and Standard Deviation for Action 2.1.

Discussion and Future Work

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

The RoboLab environment

Despite the many observations and comments from the evaluators, the RoboLab environment has been considered as 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. Under 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 in general more rigorous than real users. In other words, many of the problems stated by the evaluators will be in practice no problem at all, and vice versa, the users will encounter problems, which the experts did not perceive at all. This is an expected and known side effect, so the correct solution in this direction 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, in order to assist them in their tasks.

However, the first and second «feel and look» is very critical, especially for users of this age, so the stated usability problems augment their significance and should be treated accordingly.

The CGW and the Heuristic Evaluation

One unexpected result was the encountered inefficiency of the CGW to perform adequate 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 by the Heuristic approach in Investigator as such. The reason for the unwillingness of the evaluators to continue with the CGW could also rely on the enhanced difficulty and the 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 revealed in a popular interface, a fact that can assist its improvement. On the other hand, two expert-based methodologies have been tested and compared on the same software piece. So, we believe this study to be in line with the many reported studies by so far on the evaluation of educational interfaces and to contribute in this direction.

References

Ackermann, E. (2003). Hidden Drivers of Pedagogic Transactions: Teachers as Clinicians and Designers, Proc. 9th EuroLOGO Conference, August 2003, Porto, Portugal, pp. 29-37.

Aedo, I., Catenazzi, N. & Diaz, P. (1996) The Evaluation of a Hypermedia Learning Environment: The CESAR Experience. Journal of Educational Multimedia & Hypermedia, 5(1), 49-72

Benyon, D., Davies, G., Keller, L., and Rogers, Y. (1990) A Guide to Usability-Usability now! Milton Keynes: The Open University

Catenazzi, N., Aedo, I., Diaz, P. & Sommaruga, L. (1997) The Evaluation of Electronic Book Guidelines from two Practical Experiences. Journal of Educational Multimedia & Hypermedia, 1997, 6 (1), 91-114

Csikszentmihalyi, M. (1996). Creativity: Flow and the Psychology of Discovery and Invention, New York: Harper Perennial.

Demetriadis, S., Karoulis, A., and Pombortsis, A. (1998). Evaluation of Educational Simulation Interface using the Graphical Joghthrough Method: The Network Simulator Experience, ED-MEDIA & ED-TELECOM 98 Conference, Freiburg – Germany, 262-267.

Instone, K. (1997). Usability Heuristics For The Web. (http://www.webreview.com/1997/10_10/ strategists/10_10_97_2.shtml) , 17 Nov 02

ISO (1998) ISO 9241 - International Standarization Organization. Ergonomic requirements for office work with visual display terminals (VDT's), Part 10, Dialogue Principles.

Karat, C., Campbell, R. & Fiegel, T. (1992) Comparison of Emperical Testing and Walkthrough Methods in User Interface Evaluation, 1992. Proceedings of ACM CHI '92. Monterey, CA, May 3-7, 397-404, ACM publ.

Karoulis A., Demetriades S., and Pombortsis A. (2005). Cognitive Graphical Walkthrough: An Interface Evaluation Method. Encyclopedia of HCI. IDEA Group publ. (to appear)

Karoulis, A and Pombortsis, A. (2002). Heuristic Evaluation of Web-Based ODL Programs. In Claude Ghaoui (edt.) Usability Evaluation of On-Line Learning Programs. Hershey, PA (USA) & London (UK): Information Science, 89-109.

Karoulis, A., Demetriades, S., Pombortsis, A. (2000) The Cognitive Graphical Jogthrough – An Evaluation Method with Assessment Capabilities. Applied Informatics 2000 Conference Proceedings, 369-373. 14-17 Feb. 2000, Innsbruck, Austria. Anaheim, CA: IASTED/ACTA

Karoulis, A., and Pombortsis, A. (2000). Evaluating the Usability of Multimedia Educational Software for Use in the Classroom Using a «Combinatory Evaluation» Approach. Proc. of EDEN 4th Open Classroom Conference, 20-21 Nov 2000, Barcelona, Spain.

Levi, M.D., and Conrad, F.G. (1996). A Heuristic Evaluation of a World Wide Web Prototype, Interactions Magazine, July/August, Vol.III.4, pp. 50-61, ACM Publ. http://www.bls.gov/ore/htm_papers/ st960160.htm.

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. & Rieman, J. (1990). Testing a Walkthrough methodology for Theory-Based Design of Walk-Up-and-Use Interfaces. Proceedings of ACM CHI '90, Seattle, Washington. April 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. Washington, DC: APA Press, 181-194.

Makrakis, V. (1999) Evaluation of Open and Distance Learning Environments. In. Open and Distance Learning. Institutions and Functions. Volume A'. Hellenic Open University Patras, 1998 (in Greek)

Mindell, D., Beland, C., Wesley C., Clarke, D., Park, R., 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. (1992) Finding Usability Problems through Heuristic Evaluation. Proceedings of ACM CHI '92. Monterey, CA, May 3-7.

Nielsen, J. (1993) Usability Engineering. San Diego: Academic Press.

Nielsen, J. (1994). Heuristic evaluation. In Nielsen, J., and Mack, R.L. (Eds.), Usability Inspection Methods, New York, NY: John Wiley & Sons.

Nielsen, J., and Molich, R. (1990). Heuristic Evaluation of User Interfaces, Proc. of Computer-Human Interaction Conference (CHI), Seattle, WA, 1-5 April, 249-256

Nielsen, J. and Norman, D. (2000). Web-site Usability: Get the Right Answers From Testing http://www.useit.com 14/2/2000.

Norman, D.A. (1988). The Psychology of Everyday Things. New York: Basic Books.

Papert, Seymour (1980): Mindstorms:Children Computers and Powerful Ideas, New York: Basic Books

Piaget, J. (1952). The Origins of Intelligence in Children. New York: International University Press.

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, 741-773.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S., Carey, T., (1994). Human-Computer Interaction, Reading, Mass: Addison-Wesley.

Reeves, T.C. (1993) Evaluating Technology-Based Learning. In Piskurich, G.M. (Edt.) The ASTD Handbook of Instructional Technology. McGraw-Hill, New York,15.1-15.32.

Resnick, M., Berg, R., Eisenberg, M. & 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. Proc. of CHI-95 conf, May 7-11, 387-388

Rowley, D. & Rhoades, D. (1992). The Cognitive Jogthrough: A Fast-Paced User Interface Evaluation Procedure. Proceedings of ACM CHI '92, Monterey, California, May 3-7, 389-395.

Scriven, M. (1976). The methodology of evaluation. In Tyler, R. (edt.) Perspectives of Curriculum Evaluation, Rand McNally, Chicago.

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

Vygotsky, L.S., (1936/1978). Mind and society: The development of higher mental processes. Cambridge, MA: Harvard University Press

Wharton, C., Rieman, J., Lewis, C. and Polson, P. (1994). The Cognitive Walkthrough: A Practitioner's Guide. In Nielsen, J. and Mack, R.L. (edts) Usability Inspection Methods. New York: John Wiley & Sons, 105-140.