2006-2144: DESIGNING EFFECTIVE EDUCATIONAL SOFTWARE: INVOLVING CHILDREN IN THE DESIGN PROCESS

Barbara Moskal, Colorado School of Mines
Department of Mathematical and Computer Sciences Colorado School of Mines, Golden, CO 80401

Leanne Hirshfield, Tufts University
Department of Computer Science Tufts University, Medford, MA 02155

Designing Effective Educational Software: Involving Children in the Design Process

Abstract

According to proponents of educational software, one manner in which to improve student learning is to provide students with personalized tutors through the use of educational software. However, without the authoritative involvement of a teacher, many students are not motivated to learn material presented via computer. The challenge to educational software designers is to create environments that motivate students to think reflectively about content, encouraging them to invest time and energy in the learning process. One manner in which to accomplish this goal may be to include student ideas when developing software. This paper presents the results of a research investigation that examined the inclusion of middle school students in the process of designing educational software. Eight middle school students participated in a focus group discussion, during which time they generated ideas for teaching fractional content. Based on their input, an educational game was developed. Sixty-three middle school students who had not participated in the focus group were then randomly assigned to either treatment or control group. The treatment group worked with the software that was developed based on the ideas of the middle school focus group; the control group worked with software that was developed based on the ideas of adult software designers. Both games had nearly identical fraction content and the differences between the two games stemmed primarily from the ideas produced by the student designers. Results suggest that students working with the game based on middle school students' ideas had a greater increase in fractional knowledge as measured by a content assessment than did those that worked on the game developed by adult designers.

I. Introduction

Many believe that mathematical educational software can act as a personalized tutor for each student, supplementing the classroom instruction provided by the teacher. This permits teachers to continue with the class schedule while enabling students to develop a deeper understanding of prior material. Although some advocates believe that software can act as a personalized tutor regardless of the skill that is being learned, many researchers and educators disagree.[1,2] Most educational software focuses on drill and practice, an approach that may be successful for teaching low level skills such as memorizing facts, but that fails to promote the learning of conceptual, higher level information.[1] Examples of low level content are memorizing the multiplication tables or the process for adding or subtracting single digit numbers. Low level problems do not require reflective thought on the part of the student. In order for educational software to be truly effective as a personalized tutor, students need to think reflectively while working with the software. This study is concerned with student learning at the *reflective level*, where students need to dispense cognitive energy in order to reflect on and solve the problem at hand. Indeed, learning conceptual information requires a higher investment of mental energy than learning low level mathematical facts.[3]

Research suggests that educational games can provide an environment that motivates students to think reflectively about the academic content presented.[3,4] A well designed educational game can cause the user to enter *flow*, which is defined as "a condition of deep nearly meditative involvement ."[5] When students enter *flow*, their social and self esteem concerns are temporarily halted, and they invest their energy in the content that is required to win the game.

While much research suggests that educational games are a great way to capture the interest of students, designing an enjoyable and engaging game is difficult.[4] Many software designers do not adequately understand their target audience, and consequently, they design programs that fail to motivate their users.[6] The type of learning environments that are motivating to students differ for students at various ages. For example, the software that motivates a 2nd grade student to learn the material may be labeled *too immature* by a 4th grade student. Middle school students are at a very challenging time in life when social pressures are high and self-esteem is low.[7] They need an engaging software environment that does not come across as childish. Without proper motivation, students will not dispense the reflective thought necessary to learn the content. Contributing further to this problem is the practice of limiting children's involvement in the design process to critiquing final products.[8] This feedback often focuses on aesthetics of the interface rather than the design itself.

In an attempt to create software that better matched users' needs, Robert Reimann and Alan Cooper created Goal Directed Design (GDD). GGD requires that the designer understand the target users' goals and motivations before creating software. This is accomplished through extensive literature reviews and interviews with both members of the target audience and content experts. This information is then used to guide the designer in the creation of the desired software.[6] Because of these efforts to better understand the user, GDD is more effective than many traditional design methods in meeting users' needs.

GDD cautions against including users in the process of generating ideas for designing the software. Since most users have little knowledge of the design process, their ideas with respect to software design are assumed to be of little use. This strategy of excluding the user from the design process appears to be effective when adults are the target audience. In this situation, adults are designing software for other adults. However, when the target audience is children, adult designers may have difficulty understanding the desires of a child.

The designers of KidPad,[8] a software environment that replaces traditional windows with an interactive zooming interface, recognized the problem of excluding children from the design process. Instead of using children as passive critiquers of their completed product, they decided to actively involve children in the design process. The KidPad designers recognized that children are experts at being kids (a time in life that many software designers have long since forgotten) and they felt that children can use their un-biased, expressive imaginations to come up with innovative ideas for software design. The final design for KidPad was strongly influenced by the children's ideas.

KidPad was not an instructional program, but rather an educational tool. In other words, KidPad was not designed to tutor students in a particular content area. However, the success of KidPad and its appeal to children raises an important question for the designers of educational software. How can children contribute to the design of educational software?

II. Research Goal

This research examines a new technique in the design of educational software, which we call Child Directed Design (CDD). As part of this research effort, children were included as active participants in the design of educational software. The hypothesis of this research is that educational software that is created using CDD will be more motivating to students than software created through traditional design methods. This increased motivation will result in greater reflective thought and thus, more in-depth learning. As a result, students working with programs developed through CDD will learn more content than those working with programs developed through traditional software design techniques.

III. Design of Programs

As part of this research, two programs were designed to teach 6th grade students fractional content. Program 1 was created using GDD. Program 2 was designed using CDD, which involves detailed focus group sessions where children actively participate in the design of the target software. Both programs addressed the same content. The students that participated in the design focus groups were drawn from a low performing school located near Denver, Colorado.

A. Design Phase of Program 1

In this investigation, the researchers guided a group of undergraduate students at the Colorado School of Mines in the creation of an educational software environment to teach the addition, subtraction and reduction of fractions to middle school students [9]. Using GDD, the undergraduate students conducted a literature review concerning fractional pedagogy, interviewed their target audience (6th grade students) and interviewed domain experts (middle school mathematics teachers). During the interviews, the 6th grade students stated that they preferred games that included competition. Additionally, all students, male and female, indicated that basketball was the most popular sport at their school. Based on this information, the undergraduate designers created the educational basketball game called *Score*.

Figure 1 displays the main screen for the *Score* applet. Students are given either a visual or numeric representation of a fraction (or an operation with fractions) at the top of the screen. They must reduce the fraction or complete the required operation and click on all the buttons in the 4 x 4 grid that are equivalent to their solution. A timer gives students 30 seconds to find all solutions in the grid. The designers chose to include a timer because of the students' indication that they found competition to be a motivational force.

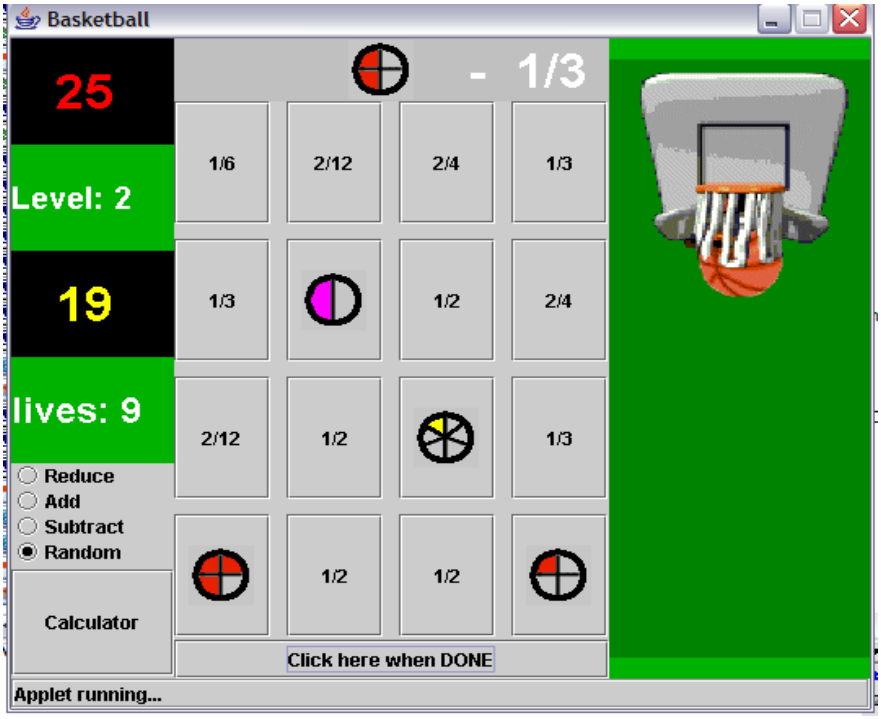

Figure 1: The main screen of *Score*.

After making the selections, the player pushes the *Click here when DONE* button. If all the appropriate solutions are identified, 2 points are awarded and the basketball is thrown through the hoop. If the student has not found all solutions, or has selected incorrect solutions, a prompt appears to try again. After 3 incorrect attempts, the student loses a life and a help screen appears. The designers included the help screen as a way for the program to adapt to each student. The level increases with each correct solution and the questions increase in difficulty when the student completes 3 levels. For greater detail regarding the design of *Score*, see Miller.[10]

B. Design Phase of Program 2

A group of eight sixth grade students participated in an hour long semi-structured interview or focus group. Half the group was female. Students were told that the researchers were going to make a computer program to teach the addition, subtraction, and reduction of fractions. They were asked: "If you could use a magic wand to create this program, what would it look like?" Without prompting by the researchers, the students immediately assumed that the software would be a game. A lengthy discussion then followed in which students shared their ideas for a fun, motivating educational game.

The boys and girls in the group tended to differ in their ideas for the ideal game. Boys liked a more 'shoot-em-up' violent game and girls preferred games with strategy, role-playing, and mystery. The researchers asked the students to discuss ways that both genders would be happy. The children's unanimous solution was to have a plot-driven game. The remainder of the discussion focused on students' ideas for such a game. Students discussed a number of different plots, including kidnapping scenarios, various adventures and world travel.

Choosing one's own character emerged as an important design consideration. In order to feel connected to the game and plot, students wanted to have a choice of characters. One boy mentioned the idea of having a timer in the game. Interestingly, the other seven students disagreed, indicating that they found timers to be frustrating.

Many of the students' ideas reflected their experiences with popular video games. They talked repeatedly about the importance of "awesome graphics". The students felt that the researchers should create an educational video game on the scale of Zelda or Grand Theft Auto. Although an educational game on this scale was not possible, it was obvious to the researchers that the students did not have any pre-conceived notions about the software's limitations, allowing them to think ambitiously. When asked about the availability of a help screen, the students insisted that they did not want mandatory help pop ups. They wanted to have control over whether or not help was provided.

The software resulting from the focus group session was a plot driven game called *Nomathia*. The first screen of the game allows the player to choose between several characters (with an equal number of male and female characters) to use throughout the game. After selecting a character, a movie shows a young child asking for help. The child's twin sister has been kidnapped by the Riddler (a character that is similar to the Riddler character from the Batman comic books). The Riddler has left a string of riddles which must be answered in order to find the kidnapped child. The riddles are all mathematical, and coming from the land of *Nomathia*, the child needs help to answer the riddles. By using the arrows on the keyboard, students navigate their character through a world looking for question marks. Figure 2 shows one of the game screens in *Nomathia*.

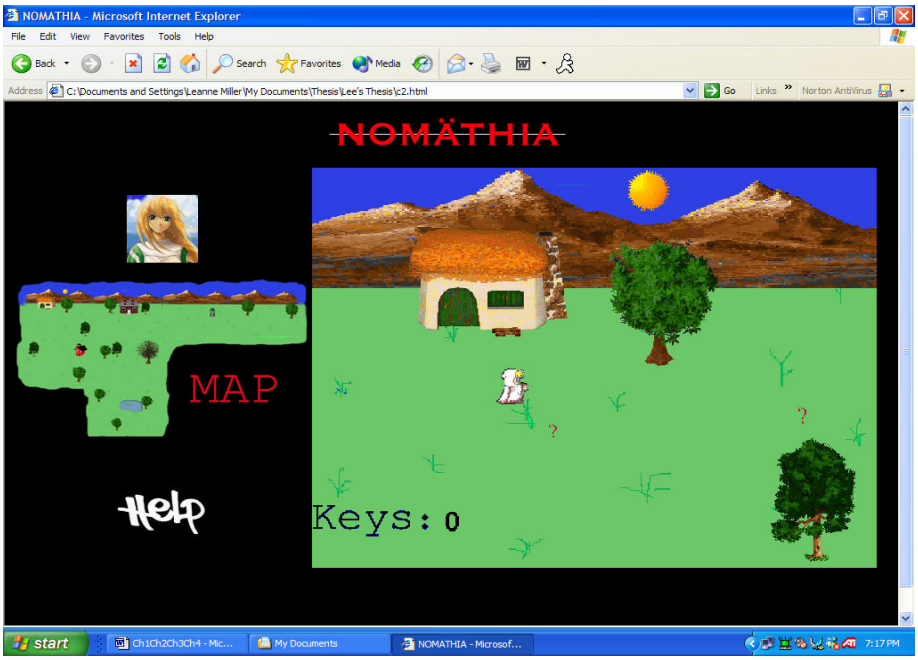

Figure 2: Starting world scene in *Nomathia*.

There is a picture of the chosen character and a map of the entire world to the left of the screen. As in the popular game Zelda, the character uses the map as a guide to walk from one screen to

another throughout the quest. When the character runs into a question mark he or she is taken to a screen that contains a mathematical riddle. An example of a riddle screen is shown in Figure 3. The riddle is a question involving the addition, subtraction, or reduction of visual and numeric fractions. In order to keep the fractional content the same between the two programs, *Nomathia* draws randomly from a subset of the questions available in *Score*.

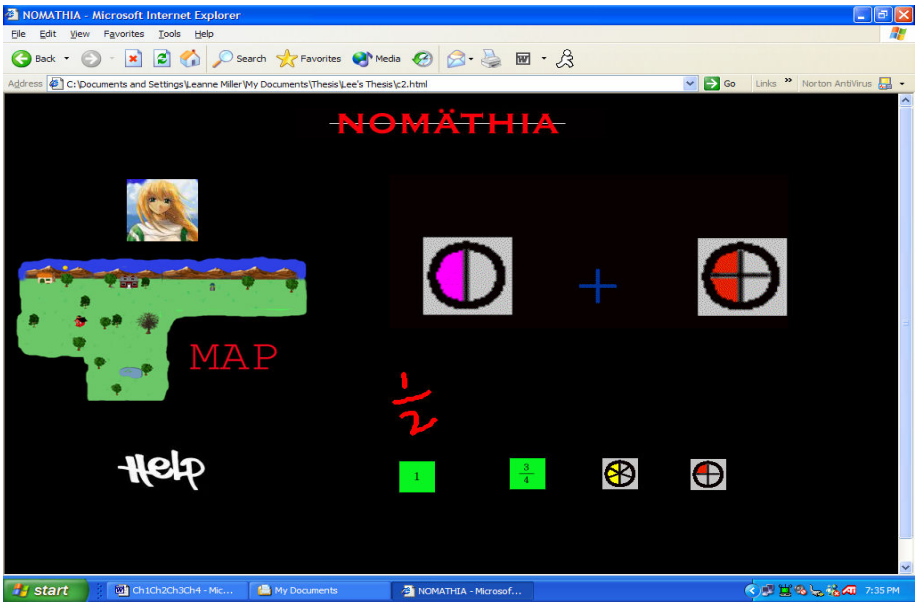

Figure 3: The riddle screen in *Score*.

The players select the button corresponding to the correct solution. Each time the students answer a question correctly they are shown a movie where the Riddler gives them a key. After earning 15 keys, the students can open the 15 doors leading to the kidnapped sister and a movie shows the happy reunion between the two siblings from *Nomathia*. If the students answer incorrectly, they are returned to the main game screen. After answering incorrectly three times in a row, the students are returned to the beginning of the game and any keys they have earned are destroyed. As with *Score*, the questions increase in difficulty as the game progresses. The questions are easiest when the student has 0-4 keys, moderate when he has 5-9 keys, and difficult when he has 10-14 keys. At any time the student can press the voluntary help button to view a help pop-up explaining reduction, addition, and subtraction of fractions.

C. Design Comparison

It is important to note the similarities and differences between *Score* and *Nomathia*. Since the goal of this study is to examine the effect that CDD has on the resulting educational software, precautions were made to ensure that any differences in learning between the two programs could be attributed to the student design choices. For this reason, the fraction content between the two programs was kept nearly identical. Both games used the same fractional values and the same levels of difficulty as the game progressed.

Also, the help screens on both programs had nearly the same text. The only difference was that *Nomathia's* help screen included pictures, which was a request made during the focus group session. A notable difference between the games is the role of the help screen. *Nomathia* has a

voluntary help screen and *Score* has a help screen that pops up after the user gets three answers wrong in a row. The child designers insisted on a voluntary help screen in *Nomathia*, while the goal directed designers felt that their users would benefit from an involuntary pop up after three consecutive wrong answers.

Another difference between the programs is the absence of a timer in *Nomathia*. The designers of *Score* added the timer to increase competition and keep students highly motivated, while the child designers requested that no timer be used in *Nomathia*. Another difference is that *Nomathia* is a plot driven game and *Score* is a high-score sports game. This was also a design choice of the child designers, who wanted a plot driven game as opposed to a sports game. The child designers also stressed their desire to have a choice in characters in their target educational game. The designers of *Score* did not include characters in their basketball design. Lastly, a considerable difference worth mentioning is the graphics in both games. Although neither game is on the level of today's popular XBox and PlayStation games, *Nomathia* does contain more realistic graphics than *Score*. *Nomathia* also contains more video game qualities than *Score*. Examples are the movies that play throughout the game, the ability to control characters with the keyboard, sound throughout the game, and the realistic characters and landscapes chosen for the game. All of these differences stemmed from the two design processes. Since GDD instructs designers not to involve users in the design process, specific software design issues such as the graphics of the game, the help screen, the timer, and the importance of characters would not emerge in the GDD interviews.

IV. Experiment

In order to empirically investigate the results of the educational software on student learning, the software was tested on 63 sixth grade students from the same middle school in which the original interviews took place. Students involved in the design of the educational games were not included in this group. Students were randomly placed into two groups. The first group worked with *Score*, and the second group worked with *Nomathia*. Each group spent two 50 minute class periods working with their respective software. Released questions from the *Adston Mathematics Skills Series: Diagnostic Instrument in Common Fractions*, a validated fraction test, were used to determine students' understanding of fractions before and after using each educational game. After working with their respective program, students also filled out a survey asking them whether or not they liked certain design choices included in their game.

V. Results

Two methods were used to statistically compare treatment and control students' performance on the content assessment: t-tests and Analysis of Covariance (ANCOVA). The design survey was examined using qualitative methods.

A. T-test

The t-test analysis began with the calculation of difference scores between the pre and post test for students in the two groups. Next, the average increase for each group was determined. The students who played *Score* had an average increase of 0.5% from pre to post test, while students

that played *Nomathia* had an average increase of 5.9%. These differences in scores were compared and found to be statistically significantly different with $\underline{p} = .03$. In other words, the students that played *Nomathia* displayed greater increases in their fractional knowledge than did the students that played *Score*.

B. ANCOVA

The scores on the pre and post tests were also compared using ANCOVA, which is commonly used to analyze differences between pre and post test scores.[11] ANCOVA adjusts post test scores of both groups based on any initial differences between the two groups on the pre test. A program in Mathematica was used to analyze the data with ANCOVA. The program produced an F-value of 7.591 with 60 degrees of freedom. These results were found to be statistically significant with $\underline{p} = .01$. Based on the results from ANCOVA, students who worked with *Nomathia* on average had significantly higher adjusted post-test mean scores than did students that worked with *Score*.

C. Design Survey

As was previously discussed, students also completed a survey that addressed their reaction to specific design choices in their respective game. Table 1 displays the results of that survey.

Table 1: The percentage of students answering *Yes*, when asked if they enjoyed a specific design choice within each game.

	Nomathia Design Agreement	Score Design Agreement
Nomathia: I liked saving the kidnapped sister *Score*: I liked the basketball in the game	97%	91%
Nomathia: I liked choosing my character *Score*: I liked the bonus level	97%	75%
Nomathia: I liked the Riddler character *Score*: I liked the timer	68%	53%
Nomathia: The help was useful *Score*: The help was useful	65%	50%

While no quantitative comparisons can be made between the design choices within each program, it is interesting to note that students working with *Nomathia* appeared to respond more positively to the design choices than did the students working with *Score*. Only 53% of *Score* players liked the timer and only 50% of these students found the help pop ups to be useful. These were two design choices that the child designers selected not to include in *Nomathia*. Also, 97% of students enjoyed the kidnapping scenario and the character selection, design choices made specifically by the child designers.

D. Limitations

One notable limitation in this investigation is that *Score* allows students to choose between addition, subtraction, reduction, and random (a random question involving the three operations

listed previously) and *Nomathia* randomly selects these questions. To overcome this discrepancy, researchers overseeing the students working with *Score* instructed students to select the random button so that the questions would be like those presented in *Nomathia*. Also, when researchers compared the questions involving only reduction on the fraction exam between the two groups, *Nomathia* students still performed significantly better than their *Score* playing counterparts. The *Nomathia* players showed an average increase of 6.17% on these questions, while students working with *Score* had an average increase of .48%. The t-test resulted in a p value of .05, which is significant at the alpha = .05 level. This suggests that students using *Nomathia* learned reduction better than those working with *Score*. Therefore, even if the *Score* players did primarily choose reduction questions, *Nomathia* was more effective at teaching these fractional skills.

Another limitation involves the resources available to create *Nomathia*. The child designers made it clear that they wanted a game with graphics and believability on the level of a professional video game. Monetary and personnel resources were not available to create such a game in this investigation.

VII. Conclusions

One of the most important findings to come from this investigation is that children working with *Nomathia* displayed significantly greater increases in their fractional knowledge than did children working with *Score*. Students within each group spent the same amount of time working with each program, they were randomly assigned to work with *Score* and *Nomathia,* and the fractional content presented in the program was kept nearly identical. Therefore, the design of the program appears to be the dominant factor contributing to this difference.

The designers of *Score* followed the steps in GDD, and therefore, they did not ask students questions specific to the actual software design. This may have resulted in their design choices being less desirable to the user than those made in *Nomathia*. Since the gap in motivations and goals between adult software developers and middle school aged children is so large, perhaps design techniques that stress 'knowing ones users' are not sufficient when designing for children. CDD appears to be preferable to the methods of GDD when working with middle school students. Further research is necessary to determine whether this conclusion can be generalized across different educational settings, content and age groups.

Acknowledgements

We would like to thank Dr. Cyndi Rader, Rossie Parkhurst, Que Nguyen and Agata Dean for their help throughout this research investigation. We would also like to thank BJ Buchmann for supporting this research at his school. Finally we are grateful to Elaine Pugh and Sharon Peters, who agreed to have their mathematics students participate in the study.

Bibliography

[1] Caftori, Netiva. *Educational Effectiveness of Computer Software.* T.H.E. Journal Feature. August 1994.

[2] Ringstaff, Cathy. *The Learning Return on Our Educational Technology Investment.* WestEd. 2002.

[3] Klawe, Maria. *When Does The Use Of Computer Games And Other Interactive Multimedia Software Help Students Learn Mathematics?* EGEMS. 1998.

[4] Sedighian, Kamran. Challenge Driven Learning: A Model for Children's Multimedia Mathematics Learning Environments. EGEMS. University of British Columbia. 1997.

[5] M. Csikszentmihalyi. *Flow=The Psychology of Optimal Experience.* New York: Harper and Row. 1990.

[6] Cooper, Alan & Reimann, Robert. *About Face 2.0 : The Essentials of Interaction Design.* Wiley Publishing. Indianapolis, IN. 2003.

[7] *Helping Your Child Through Early Adolescence.* U.S. Department of Education. [Online]: <http://www.ed.gov/parents/academic/help/adolescence/adolescence.pdf> [23 May 2005]

[8] Hourcade, Juan Pablo, Bederson, Benmaimin, Druin, Allison, Taxen, Gustav. *KidPad: a Collaborative Storytelling Tool for Children. Extended Abstracts of Human Factors in Computing Systems*. CHI 2002.

[9] Parkhurst et al. CSM #2: *GK-12 Learning Partnerships. Final Implementation Document*. 2004.

[10] Miller, Leanne. *Educational Software Considerations to meet the Various Needs of Middle School Mathematics Students*. MS Thesis. Colorado School of Mines. 2005.

[11] Gay, L.R. *Educational Research: Competencies for Analysis and Application*. Macmillan Publishing Co. New York, NY. 1987.