Chen, W. et al. (Eds.) (2017). Proceedings of the 25th International Conference on Computers in Education.
New Zealand: Asia-Pacific Society for Computers in Education

Applying Pedagogy to the Design of Software for Helping Students Learn Equation Solving

Daphne ROBSON*
Ara Institute of Canterbury, New Zealand
*daphne.robson@ara.ac.nz

Abstract: In this paper, we describe the development of software designed to help students learn strategies for equation solving. Instructional design principles were followed and two cycles are described. In the first cycle of analysis, design, development, implementation, and evaluation, prototype software was completed. In the second cycle, after analysis of the prototype, Equations2go was designed, developed and implemented ready for evaluation. Pedagogical principles are fully embedded into the design of Equations2go and are the focus of this paper. Equations2go is now available free of charge.

Keywords: Educational software, pedagogy, instructional design, equation solving, flexible thinking.

1. Introduction

For many years, the potential for computers to be used to help students learn has been seen but it is now well recognised that any use of computers or software for learning must be based on sound pedagogical principles. This paper focuses on the design of software for students undertaking study in applied fields such as Computing, Engineering, and Science who need to learn how to use and manipulate mathematical formulae and equations relevant to their field. The software development followed the ADDIE instructional design model with its five components: Analysis, Design, Development, Implementation, and Evaluation.

In the first phase of the ADDIE cycle, the learning needs of adult students with a variety of backgrounds and prior mathematical knowledge were analysed and relevant pedagogies identified. Software was designed and developed using a rapid prototyping principle in which a simplified version of the main software design was developed. This prototype software was then implemented and evaluated to investigate its impact on equation solving strategies of students. (Robson, 2006).

In the second phase, results of the first phase were analysed and the software redesigned, redeveloped and implemented ready for evaluation. The focus of this paper is to describe the relationship between pedagogical principles and the design of the software, Equations2go (Robson & Wratt, 2017).

2. Prototype Design

When the prototype software was designed, relevant pedagogical principles were considered and then applied to the design of features of the software. In this section, each pedagogical principle is described followed by the software feature that it led to.

2.1 Emphasis on Strategies

Researchers have observed that many mathematics teachers focus on teaching procedures, but the type of thinking needed to plan a strategy or set of actions to solve an equation is an important part of learning to think mathematically and to solve problems. Star and Madnani (2004) described this type of higher level thinking as ‘flexible thinking’.

When students use the software, they are encouraged to learn to think more flexibly without being distracted by algebraic details as students make decisions about what action to take and the software carries out the procedure needed for that step.

2.2 Multiple Strategies

Learning activities which help students think with the flexibility needed for equations solving include finding and comparing multiple strategies (Star & Seifert, 2006) and thinking about which strategy is best (Star & Madnani, 2004). The ability to use and select from multiple strategies is widely associated with mathematical expertise (Pegg & Tall, 2002; Jonassen, 2000).

To further encourage students to think with flexibility, the prototype software accepts several different strategies for each equation and each strategy is displayed visually with its own stepping stones pattern.

2.3 Informative Feedback

Much research has found that feedback makes a valuable contribution to learning as it provides communication to students after they have performed a learning task (Hattie, 2012). Most feedback is intended to either acknowledge a correct response or provide information about an incorrect response allowing students to learn from both their successes and their errors. It is generally agreed that feedback improves learning if it encourages students to think actively (e.g., Lyster & Ranta, 1997). It is relevant to consider a study of feedback for algebra. Nguyen-Xuan, Nicaud and Gelis (1997) tested different types of feedback and found that for algebra, it should be short, include consequences of errors, give enough information for students to see why their response was incorrect, but allow them to work out the next step themselves.

The prototype software provides feedback at each step that refers to the action that the student chose rather than just explaining what to do. In other words, the feedback relates to their choice of strategy. When a student chooses a correct action, the software provides positive feedback, explains what progress has been made, and displays a working step for the action along with the results of the action. When a student chooses an action that does not progress towards the solution, information about the goal for that step is provided and students can then try again.

2.4 Visual Interface

Research on whether graphics contribute to learning is varied but any graphics must be integral to the topic and the learning, and should be included for a specific reason (Rieber, 1994). Graphics can allow students to use both visual and verbal channels. By providing visual images, it can help students observe patterns and spot relationships (Pfitzner, Hobbs & Powers, 2001). This in turn can help students develop their understanding of relationships, including those in algebra (Hewitt, 2012).

In the prototype software, graphics were used to provide stepping stone metaphors of “take one step at a time” and “leave no stone unturned” with the latter reflecting the students’ search for different strategies. Other visual metaphors are provided by different paths along the stones representing different strategies, feedback flags or signposts providing guidance, and the light in the tree turning on to reward successful choices. The direction of each successful step uses a metaphor of opposite directions for inverse operations. Students were also able to request that a score be displayed. The interface design of a partially completed equation is shown in Figure 1.

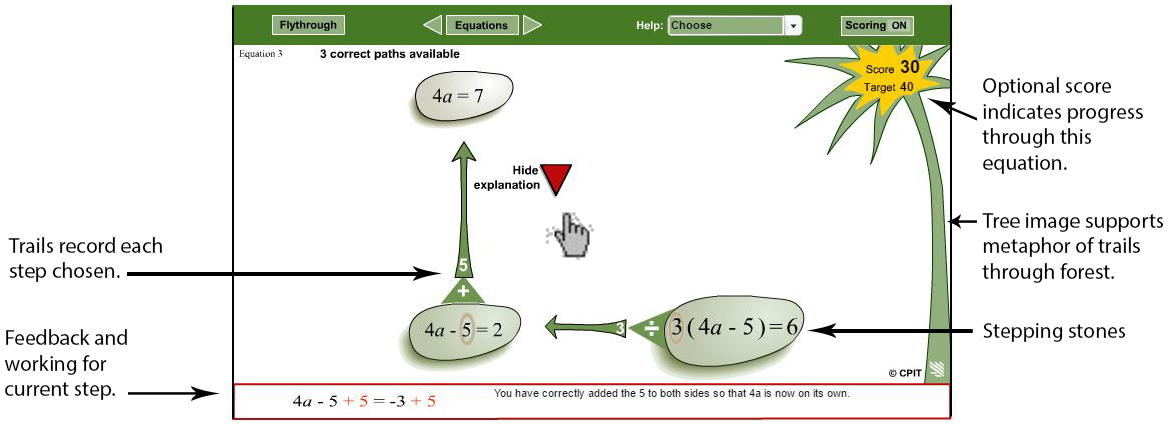

Figure 1. Partially solved equation in prototype software

3. Analysis of the Prototype Design

Research involving tertiary students using the prototype software (Robson, Abell & Boustead, 2012) included pre and post-tests, surveys, and analysis of electronic data showing which choices students made. Results relevant to each pedagogical principle follow.

3.1 Emphasis on Strategies

Emphasising strategies rather than the details of procedures was very successful for students when the equation was in their "Zone of Proximal Development" as described by Vygotsky (1978). In this zone, students are unable to solve a problem on their own but are able to solve it with guidance. All students who were unable to solve a particular type of equation in the pre-test were able to solve a similar equation in the software and half of these were also able to successfully apply their learning to the post-test and solve this type of equation on their own.

3.2 Multiple Strategies

The majority of students found searching for multiple solution strategies helpful, but the students who made the most use of this feature were those who were already able to solve equations in the pre-test. Some students found multiple strategies confusing, and may have been cognitively overloaded by several strategies.

3.3 Informative Feedback

In the prototype, a quick tip was displayed for each action and students could click to request further feedback. This was well used with logged data showing that students requested the extra feedback an average of three times per minute. Furthermore, students' survey comments described how useful they found the explanations. The reason for this two stage display of feedback was to keep the main screen as simple as possible (Neilsen, 1993) while allowing students to actively request more information when they needed it (Mason & Bruning, 2001). The majority of students liked the feedback provided by the score as it allowed them to see their progress, but others found it discouraging.

3.4 Visual Interface

The data collected showed no evidence that the visual metaphors contributed to learning. This result supports Reiber's (1994) assertion that any graphics must be integral to the topic, and suggests that the visual metaphors for abstract principles were not a strong enough reason to use graphics. Although in the trials, students were asked to explore multiple strategies, the software did not make this clear. A need was identified to redesign the interface so that students would be encouraged to explore multiple strategies and hence have the opportunity to increase their understanding.

4. Redesign of the Software

As the emphasis on strategies in the prototype was so successful in enabling students to solve equations they couldn't solve on their own, this was still the main principle on which the redesign was based.

To encourage more competent students to learn from exploring multiple strategies, but without confusing beginning students, a system of stars was included in the design that recorded the number of 'important' strategies found. Important strategies were considered by the author, in consultation with her colleagues, to add to students' understanding. Each equation began with one or more outlines of stars and these were filled in when each important strategy was found. Other strategies were also accepted by the software and students received other positive feedback for these.

As students requested the extra feedback so often in the prototype and reported favourably on its usefulness for learning, it was decided to display it after every action. To continue to keep the main screen as simple as possible a clean simple interface was designed, without the graphics that had not appeared to contribute to students' learning. The redesign of the interface is shown in Figure 2.

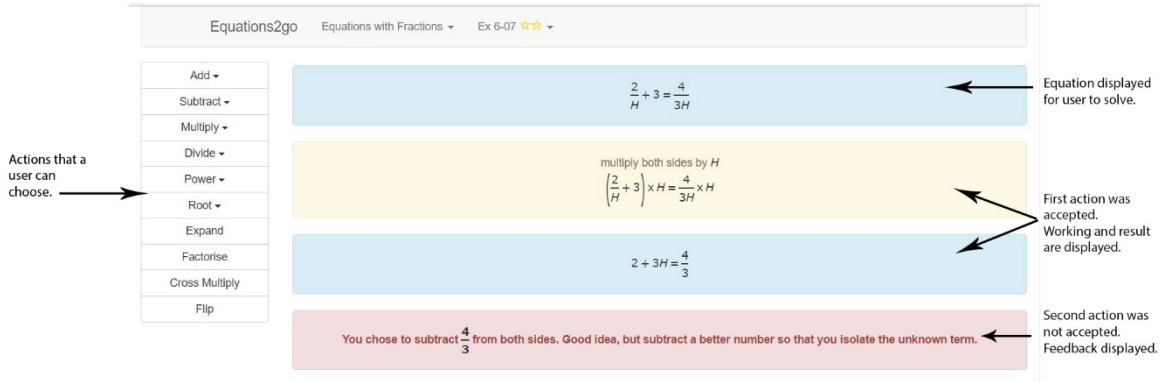

Figure 2. Partially solved equation in Equations2go

5. Development and Implementation of Redesigned software

Equations2go was developed as a web application that can be run on computers and mobile devices using a browser. There are two parts to the software: the web application code written by Matthew Wratt using NPM, Webpack and React, and the code for each equation written by the author of this paper using a custom DSL (domain-specific language). The DSL is parsed and compiled into the web application.

The software was first used with students at Ara Institute of Canterbury during the first semester of 2017 and is now freely available at http://equations2go.ara.ac.nz. It was originally designed for adult students but it is expected that school students studying algebra will also find it useful. It is hoped that in return for using Equations2go free of charge that users will use the feedback button to provide useful comments that the author can use to improve the software.

6. Evaluation of Redesigned Software

For the evaluation part of this second phase of the ADDIE cycle, We will initially involve tertiary students at Ara Institute of Canterbury who are studying mathematics as part of their preparation for studying technical qualifications in engineering, science, medical imaging or computing. As in the first phase (Robson, Abell & Boustead, 2012), the participants will answer a pre-test and a post-test before and after using the software. This will allow us to see any impact of the software on students' ability to solve a variety of types of equations. Participants will also complete a survey which will give us more information about how the software impacted on students' learning and will provide us with information for improving the software.

7. Conclusion

This paper describes a case of using pedagogy as the basis of a software design which was then developed using the instructional design principles of cycles of analysis, design, development, implementation and evaluation. In the first cycle, each pedagogical principle is described along with the software feature that it led to. The prototype was developed and evaluated by students who were studying equation solving. In the second cycle, the results of the evaluation were analysed and used to inform the design of the main version of Equations2go. With development and implementation of the redesigned software complete, we are ready to evaluate it and thus complete the second cycle. As it is important that educational software is designed using pedagogical principles, the contribution of this paper is the description of the strong relationship between pedagogical principles and software features.

Acknowledgements

Many thanks to the software developer, Matthew Wratt, who so willingly developed the main code for Equations2go and who consistently thought of many creative and innovative solutions to my many requests. I particularly appreciate the way he designed and wrote the custom DSL so that I have full control over creating the equations and determining the actions and feedback at each step.

References

Hattie, J. (2012). Visible learning for teachers: Maximizing impact on learning NY: Routledge.

Hewitt, D. (2012). Imagery as a tool to assist the teaching of algebra. In R. Sutherland & J. Mason (Eds.), Exploiting Mental Imagery with Computers in Mathematics Education (pp. 277-290): Springer Science & Business Media.

Jonassen, D. H. (2000). Toward a design theory of problem solving. Educational Technology: Research & Development, 48(4), 63-85.

Lyster, R., & Ranta, L. (1997). Corrective feedback and learner uptake: Negotiation of form in communicative classrooms. Studies in Second Language Acquisition, 19(1), 37-66.

Mason, B., & Bruning, R. (2001). Providing feedback in computer-based instruction: What the research tells us. CLASS Research Report No. 9. Center for Instructional Innovation, University of Nebraska-Lincoln Retrieved May 5, 2017, from https://www.researchgate.net/publication/247291218_Providing_Feedback_in_Computer-based_Instruction_What_the_Research_Tells_Us

Nguyen-Xuan, A., Nicaud, J.-F., & Gelis, J.-M. (1997). Effect of feedback on learning to match algebraic rules to expressions with an intelligent learning environment. Journal of Computers in Mathematics and Science Teaching, 16(2/3), 291-321.

Nielsen, J. (1993). Usability Engineering. Boston, MA: Morgan Kaufmann.

Pegg, J., & Tall, D. (2002). Fundamental cycles in learning algebra: An analysis. Paper presented at the 26th conference of the International Group for the Psychology of Mathematics Education, Norwich, UK.

Pfitzner, D., Hobbs, V., & Powers, D. (2001). A unified taxonomic framework for information visualization. Paper presented at the 2nd Australian Institute of Computer Ethics Conference, Canberra.

Rieber, L. P. (1994, 16-20 February 1994). Visualization as an aid to problem-solving: Examples from history. Paper presented at the Selected Research and Development Presentations at the 1994 National Convention of the Association for Educational Communications and Technology, Nashville.

Robson, D., & Wratt, M. (Producer). (2017). Equations2go. Retrieved 3 Aug, 2017, from http://equations2go.ara.ac.nz

Robson, D. E., Abell, W., & Boustead, T. (2012). Encouraging Students to Think Strategically when Learning to Solve Linear Equations. International Journal for Mathematics Teaching and Learning (May 2012).

Robson, D. E. (2006). Development of educational software for learning equation solving. (M Appl. Sc.), Lincoln University. Retrieved 3 Aug, 2017 from http://researcharchive.lincoln.ac.nz/handle/10182/2061

Star, J., & Seifert, C. (2006). The development of flexibility in equation solving. Contemporary Educational Psychology, 31, 280-300.

Star, J. R., & Madnani, J. K. (2004). Which way is the "best"?: Students conceptions of optimal strategies for solving equations. Paper presented at the twenty-sixth annual meeting of the North American chapter of the International Group for the Psychology of Mathematics Education.

Vygotsky, L. S. (1978). Zone of proximal development: A new approach. In M. Cole, V. John-Steiner, S. Scribner, & E. Souberman (Eds.), *Mind in society: The development of higher psychological processes* (pp. 84-91). Cambridge, MA: Harvard University Press.