Building Compiler-Student Friendship

Zhongxiu Liu and Tiffany Barnes

North Carolina State University
Raleigh, NC 27695

Abstract. Previous studies have shown that compilers positively influence students when they are designed to build connections with students. In this paper, I propose to study the use of a friendly compiler for young novice programmers. This study involves designing compiler messages that incorporate a friendship model. The goal is to make students view compiler as a friend, instead of as an error-picking authority. I hypothesize that a good compiler-student relationship will change students' attitude, self-efficacy and motivation towards programming, as well as change students compilation behaviors.

1 Introduction

The relationship between students and their peers or teachers influences many aspects of learning. Liem et al.[9] found that a good peer relationship has a positive effect on learning outcomes, mastery, and students' self-efficacy. More specifically, Bickmore and Picard[2] found that a safe communication climate, characterized by support, openness, trust and mutual respect, positively influences students' self-efficacy. In Will et al.[10] study, teachers' verbal immediacy was found to be positively associated with learning.

Traditional compilers are not usually designed with the goal of building a relationship with the users. This may hinder novice programmers. Kinnunen [7] found that novice programmers frequently reflected on their own incapability when encountered difficulties. Traditional compilers repeatedly tells students what they have done wrong, which could anguish the feeling of personal failure. Bosch et al.[3] found boredom as one of the most common emotions experienced by first-time programmer, and was negatively correlated with debugging and programming performance. I believe that traditional compilers contribute to boredom, because they do not convey the warmness, happiness or willingness to engage students in their conversations.

Previous research has shown promising results when programming environments are designed to build connections with students. Lee and Ko[8] designed a game where students used a gamified programming language. In the experimental group, the compiler was represented as a fallible and self-blaming robot character. The study found that students in the experimental group completed more tasks, and were more likely to state that they "want to help the robot succeed". Boyer[4] added a dialog-based tutor to a Java programming task. The tutor read the compilation messages, and conducted conversations with students

based on a corpus of real student-tutor dialogues. The study found that praise, reassurance, and dialogues with positive cognitive feedback increased students self-confidence with programming. However, these studies only evaluated designs that add to a specific programming environment and tasks. They did not develop a general purpose compiler or focus on adapting to more diverse programming activities.

2 Research Methodology

My research will focus on building a compiler-student friendship through the design of user-friendly compiler messages. I will investigate the following research questions: Will a friendly compiler change students' attitudes towards programming, such as self-efficacy, motivation and interest level? Are there differences in task completion, and compilation behaviors between groups? Moreover, as prior studies by Arroyo et al.[1] and by Evans and Waring[6] showed, females are more likely to be positively affected by affective and positive feedback. Thus I will also investigate whether a more affective compiler has larger affect on female students.

2.1 Compiler Design

Duck[5] defined relationship with friends as a list of provisions that we expect from friends.

- Provision1: Belonging and a sense of reliable alliance. The existence of a bond that can be trusted to be there for a partner when they need it.
- Provision2: Reassurance of worth and value, and an opportunity to help others.
- Provision3: Emotional integration and stability. Friendships provide necessary anchor points for opinions, beliefs and emotional responses.

Table 1 illustrates my approach to incorporate the three provisions into the friendly compiler design targeted for middle school students. For each provision, I explain how it can be interpreted in the context of programming, and give an example of a traditional compiler message in comparison to a friendly compiler message that incorporates the provision.

2.2 Experimental Design

To evaluate my design, i will use BlueJ, a free and open-source Java environment designed for novice programmers. Middle school students who have limited knowledge about programming will be randomly split into a control group, where they will use a traditional compiler, and an experimental group, where they will use the friendly compiler. Students will complete programming tasks. These groups will be assigned via balanced random assigment, but will be controlled for gender and incoming competence.

Table 1.

Interpretation	Traditional Compiler Message	Friendly Compiler Message
Convey the idea that the student and the compiler are in a team to solve programming tasks together	P.java:13:';'expected	I helped you find missing ';' on line 13. Lets work together and make this program runs.
Express to students that they are the compilers trustworthy friends, and the compiler needs students' help in order to compile	P.java:13: cannot resolve symbol	Sorry, I'm designed to be syntax sensitive. I need your help fixing a symbol I don't understand on line 13
a) respond to students success b) tell students that their mistakes are understandable c) show appreciation for the effort paid by the students	a)compilation successful b)';'expected (same errors happened several times before) c)';'expected (after several compilation errors in a row)	a)Great we made it! b)I found missing ';'. This is a common error even for expert programmers c)I found a missing ';'. I know debugging needs hard work. Thanks for all the effort you've been put to help me compile!

Before starting the programming activity, both groups will be given pre-questionnaires that ask about their programming experience, their opinions about programming and compilers, and their self-efficacy and motivation levels. During the programming activity, students from both groups are allowed to ask for help from observers, but each type of help will be recorded. BlueJ will be instrumented to log data at each compilation. The logged data will include the programming task's level, the time of the compilation, the corresponding source code, the compiler's output messages, and the wait-time before the student starts the next interaction with BlueJ. After the programming activity, students will be given post questionnaires include the same questions as the pre-questionnaires.

2.3 Evaluation

My evaluation will test the below hypothesis: a friendly compiler will:

- Improve students' self-efficacy and motivation in programing.
- Cause higher interest in and affection towards programming and the compiler?
- Help students persist through debugging. Thus, students will complete more tasks before giving up
- Cause difference in compilation behaviors. Students with friendly compiler will compile more frequently, and resume interactions with programming environment faster.
- Have stronger effect on female student for one or more of the above hypothesis

3 Conclusion

This study focuses on the re-design of a compiler that will build friendships between itself and novice programmers. A controlled study will be used to investigate how a friendly compiler affects students attitudes, self-efficacy and motivation towards programming, as well as their compilation behaviors.

I would like advice on the evaluation of students psychological attributes. My plan involves a questionnaire to collect students attitude toward programming and compiler, self-efficacy and motivation. I would like to be advised on the design of survey questions or alternative approaches that can help me collect information that best reflects students opinions.

Secondly, I would like to be advised on the design of my friendly compiler messages. The design of the compiler messages is highly subjective. What methodology should I use to create these messages to verify it well incorporates provisions goals, and well fits into the context of programming feedback?

References

1. I. Arroyo, B. P. Woolf, D. G. Cooper, W. Burleson, and K. Muldner. The impact of animated pedagogical agents on girls' and boys' emotions, attitudes, behaviors and learning. In *Proceedings of the eleventh IEEE International Conference on Advanced Learning Technologies*, pages 506–510, 2011.
2. T. W. Bickmore and R. W. Picard. Establishing and maintaining long-term human-computer relationships. *ACM Transactions on Computer-Human Interaction*, 12(2):293–327, 2005.
3. N. Bosch, S. DMello, and C. Mills. What emotions do novices experience during their first computer programming learning session? In *Proceedings of the sixteenth international Conference on Artificial Intelligence in Education*, pages 11–20, 2013.
4. K. E. Boyer, R. Phillips, M. D. Wallis, M. A. Vouk, and J. C. Lester. Investigating the role of motivation in computer science education through one-on-one tutoring. *Computer Science Education,*, 19(2):111136, 2009.
5. S. Duck. *Understanding Relationships.* Guilford Press, New York, USA, 1999.
6. C. Evans and M. Waring. Student teacher assessment feedback preferences: The influence of cognitive styles and gender. *Learning and Individual Differences*, 21(3):271280, 2011.
7. . S. B. Kinnunen, P. Experiencing programming assignments in cs1: the emotional toll. In *Proceedings of the Sixth international workshop on Computing education research*, pages 77–86, 2010.
8. M. J. Lee and A. J. Ko. Personifying programming tool feedback improves novice programmers' learning. In *Proceedings of the seventh international workshop on Computing education research*, pages 109–116, 2011.
9. A. D. Liem, S. Lau, and Y. Nie. The role of self-efficacy, task value, and achievement goals in predicting learning strategies, task disengagement, peer relationship, and achievement outcome. *Contemporary Educational Psychology*, 33(4):486–512, 2008.
10. P. L. Witt, L. R. Wheeless, and M. Allen. A meta analytical review of the relationship between teacher immediacy and student learning. *Communication Monographs*, 71(2):184–207, 2004.