Extreme Scaffolding in the Teaching and Learning of Programming Languages

Dan-Adrian German
Indiana University School of Informatics and Computing, Bloomington, IN, USA
dgerman@indiana.edu

Abstract: We report on the design and implementation of an e-learning framework that can be used to produce tutoring modules with an arbitrary degree of scaffolding. Each module teaches by problem-solving; instant feedback is built in the scaffolding and provides the learner with the reinforcement (negative or positive) that conditions learning. Although the tutoring system guides the learner's actions, it does not restrict them excessively and significant number of degrees of freedom remain. This encourages the learners (as they prioritize their problem-solving steps) to become active constructors of knowledge through direct experimentation. By analogy with the principles of extreme (or agile) programming, we call the adjustable scaffolding technique used in our system "extreme" because it too can be set up in micro-increments and tunes the learners into the same behavioral pattern as they analyze, design, test, integrate and deploy their problem-solving steps.

Keywords: extreme scaffolding, agile instruction.

1. Introduction

Since most learning, at least in its early stages, is observational, well-chosen examples are a necessary (though not sufficient) condition for effective teaching. To be a successful learner one needs to practice, and internalize those very examples. But students aren't always self-regulated learners. They may not be aware what they do or do not understand. They sometimes think they get it when they really don't. Learning to program well requires a lot of determined practice and without evidence of learning there can be no legitimate claim of teaching. No matter how wonderfully crafted the examples presented in a lecture, it is not uncommon for students to come and say: "I understand everything when I watch you develop a program in class, but when I go and try to write a program by myself I stare at the blank computer screen and don't know where to start." This often leads to panic and frustration. Just giving examples, no matter how good, is not enough. For teaching to be effective the learner needs to assume initiative and responsibility. In this paper we present an e-learning framework that can assist in producing tutoring modules with an arbitrary degree of scaffolding. The tutoring modules generated by our framework facilitate the necessary transfer of initiative and responsibility from teacher to students.

2. Framework

We will introduce the framework through an example. We start from its English-language description:

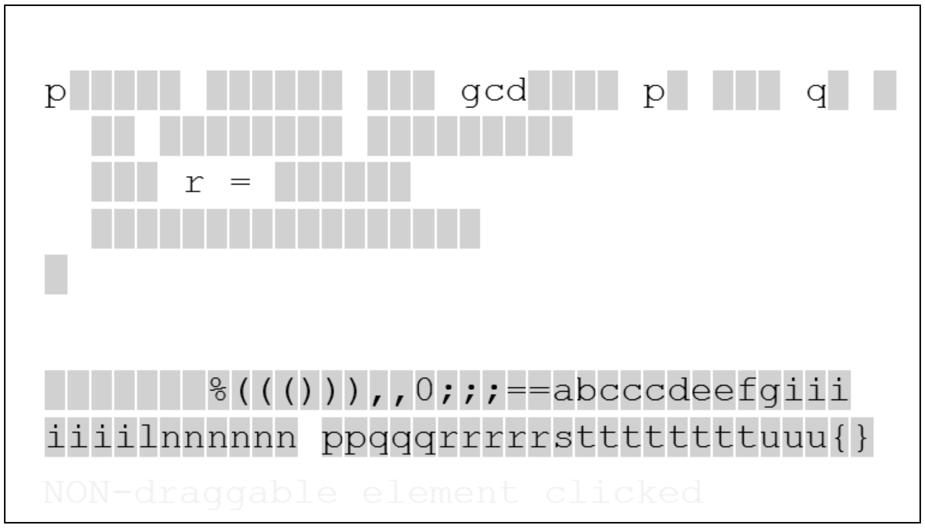
> Compute the greatest common-divisor of two nonnegative integers *p* and *q* as follows: If *q* is 0, the answer is *p*. If not, divide *p* by *q* and take the remainder *r*. The answer is the greatest common divisor of *q* and *r*.

This can be a problem already discussed in class or a problem that the students should now be able to solve with little or no difficulty. In preparing a tutoring module for this program the instructor starts from an annotated model solution (see Figure 1, code in Java).

```
`p`u`b`l`i`c `s`t`a`t`i`c `i`n`t gcd`(`i`n`t p`, `i`n`t q`) `{
  `i`f `(`q` `=`=` `0`) `r`e`t`u`r`n` `p`;
  `i`n`t r = `p` `%` `q`;
  `r`e`t`u`r`n` `g`c`d`(`q`,` `r`)`;
`}
```

Figure 1: Solution for the greatest common divisor problem, marked for tutoring module generation.

In this example the annotations consist of backquotes. The conversion program produces a tutoring module as shown in Figure 2. Marked characters are extracted and listed in lexicographic order at the bottom of the screen. The student's task is to reconstruct the program given a set of constraints (timed test, maximum number of allowable mistakes, etc). Characters listed at the bottom can be dragged with the mouse and dropped anywhere on the screen; they maintain their color until placed in the right location, when they turn into white (or the background color) as an indication that they belong there. In Figure 2 one of the characters (you can see it's missing from the bottom) has already been placed correctly; obviously, there are two other locations where it would've worked just as well.



```
p                     gcd       p         q
  
      r = 
  
 

      %((())),,0;;;==abcccdeefgiii
iiiilnnnnnn ppqqqrrrrrstttttttttuuu{}
NON-draggable element clicked
```

Figure 2: Tutoring module for the template shown in Figure 1, ready for student interaction.

3. Model

The tutoring modules produced by our framework resemble CLOZE exercises (Taylor 1953) of Gestalt theory descent; traditional CLOZE exercises are frequently used as after-reading activities for English as a Second Language learners in the mainstream classroom (Gibbons 2002). The interactive nature of our templates fully situates them within the scope of the Cognitive Apprenticeship model (Collins 1991). The generating framework could then be said to provide concrete implementations of the Extreme Apprenticeship method (Vihavainen 2011). In working with a tutoring module students (a) work by doing, (b) receive continuous feedback and (c) work at their own pace (no compromise, unless the module is set up for a test). We can easily distinguish in our tutoring modules the three phases of the Extreme Apprenticeship method: modeling, scaffolding and fading. The annotated solution represents the model, the annotation indicates the scaffolding and the fading is built-in (see the first letter in the screen shot in Figure 2). Notice that the automatic fading leaves behind a zone of proximal development (Vygotsky 1978) that is constantly evolving and entirely determined by the specific steps that had been taken by the learner.

4. Data

In his book (Willingham 2010) cognitive scientist Daniel Willingham states a series of principles with far-reaching implications for teaching: the brain is not designed for thinking it is actually designed to save us from having to think too often (nevertheless we find successful thinking pleasurable, so we seek out opportunities to think but we are very selective in doing so); thinking skills and knowledge are bound together (factual knowledge must precede skill); memory is the residue of thought (students will not remember your lecture, they will remember their thoughts during your lecture); understanding is remembering in disguise; practices enables further learning, etc. These principles seemed to back us up, but to measure the impact of a tutoring module we set up and conducted the following experiment.

A group of 59 students had their weekly labs on Thursday and Friday. The labs were structured and conducted in a traditional manner (examples presented by lab instructors, combined with individual time spent on computer by the students). The lab assignment was to produce a short program due Sunday night. Late submissions were acceptable (with no penalty) until Monday evening. On Monday at 2:30pm, in lecture, a pop quiz was given in class asking the students to write the shortest program that matched the specifications of the lab assignment that was due that evening (or the night before). The duration of the pop quiz was 15 minutes, the tests were collected and archived as pre-test data. A tutoring module for the solution to the pop quiz was then presented in lecture (this was the students' first exposure to the tool) and posted on-line. Tuesday morning a message was sent to the students with the URL of the tutoring module demonstrated in class and students were asked to work with the template once just so we can ask them if they liked it or not. A reminder was sent on Wednesday morning. There was no other implicit or explicit incentive to review/study the program other than through the requested interaction with the interactive template (the tutoring module presented on Monday in class). On Wednesday at 2:30pm in lecture we repeated the quiz from Monday, collected the tests after 15 minutes and archived them as post-test data. Both quizzes were closed book. After the second quiz we asked a few open-ended questions (which is what students had expected in the first place) such as: did you work with the interactive template as we asked you to, did you have a positive experience with the tutoring module (if so, what was the single most useful aspect, if not why not), did you do better than last time or not, and so on. The answers to these questions were anonymous, they were collected and archived with the post-test data.

Quizzes were then graded and grouped into categories as shown in Table 1.

Table 1: Pre- and post-test results.

Categories	Monday lecture		Wednesday lecture	
Wrong or no idea	27	45%	9	15%
Some idea, in the right direction	19	32%	21	35%
Recognizable (but unfinished) solution	11	18%	10	17%
Complete, correct solution	2	3%	19	32%

5. Conclusions

Significant improvement can be seen in the table: on Monday 45% of the students are hopelessly lost. Wednesday, after being asked to work with the tutoring module once, a larger percentage (49%) are able to produce a recognizable solution (with two thirds of these students actually providing a complete, correct solution). We don't know how much time students actually spent playing with the template, and what else they did besides that. We didn't grade, review or otherwise bring up the lab assignment in e-mail, blogs, on-line forums, individual and/or office hours communication with the students during this time (from Monday to Wednesday). The motivation presented to the students for completing a tutoring module was that they were going to provide usability feedback to us on the tool.

In written, open-ended anonymous comments 8% (5 students) indicated they didn't work with the template for whatever reasons (didn't have time, confused about location on the web or their browser didn't seem to work, intended to but forgot etc.) Of the remaining 54 students 76% (41 students) indicated that they had found it useful in some respect and enjoyed interacting with it. Half of these cited the ability to actively create by guided exploration as the single most useful instructional aspect. The remaining half was evenly split between the built-in interactivity and the actual scaffolding per se. There is evidence that working with a tutoring module builds endurance: "I had to look at [it] over and over to figure it out," wrote one student. "It forced me to spend time [on task]" wrote another. And as they learn to better monitor their own understanding students also tend to become more confident: "[i]t made me slow down [and] made me think about each part [...] the things I could complete I was proud of and the things I [thought I] couldn't turned out to be easy to figure out eventually."

6. Directions for Future Work

The system can already support variable-length scaffolding and can easily record/replay the steps taken by the user. Better database support, adaptive-testing (Wainer 2000) and team-based learning (TBL) capabilities along with alternative interface support (free typing) is being added. Experiments have been designed (but not yet carried out) to determine the most suitable application for individual and group work. The ability to record learners' steps solves the individual accountability problem in a TBL setting.

7. Acknowledgments

I would like to thank Joan Middendorf (former Associate Director of Campus Instructional Consulting on the Bloomington campus) for expert advice and guidance; Suzanne Menzel and John F. Duncan from the Division U Collegium for Excellence in Undergraduate Teaching for their constructive criticism and comments on an earlier draft; and Kathryn Propst from the Center of Innovative Teaching and Learning for her very efficient steering of the "New Technologies New Pedagogies" group meetings.

References

Collins, A., Brown, J.S., and Holum, A. (1991) "Cognitive apprenticeship: making thinking visible", *American Educator*, Winter, pp. 38–46.

Gibbons, P. (2002) *Scaffolding Language, Scaffolding Learning—Teaching Second Language Learners in the Mainstream Classroom*, Heinemann, Portsmouth, NH.

Taylor, W.L. (1953) "Cloze procedure: A new tool for measuring readability." *Journalism Quarterly*, Vol. 30, pp. 415–433.

Vihavainen, A., Paksula, M. and Luukkainen, M. (2011) "Extreme Apprenticeship Method in Teaching Programming for Beginners", *SIGCSE '11: Proceedings of the 42nd ACM technical symposium on Computer science education*, ACM, New York.

Vygotsky, L.S. (1978) "Interaction Between Learning and Development", *Mind in Society: The Development of Higher Psychological Processes*, Harvard University Press, Cambridge, MA

Wainer, H., and Mislevy, R.J. (2000). "Item response theory, calibration, and estimation." In Wainer, H. (Ed.) *Computerized Adaptive Testing: A Primer.* Lawrence Erlbaum Associates, Mahwah, NJ.

Willingham, D. T. (2010) *Why Don't Students Like School: A Cognitive Scientist Answers Questions About How the Mind Works and What It Means for the Classroom*, Jossey-Bass.