All that glitters (in the lab) may not be gold (in the field)

Amruth N. Kumar

Ramapo College of New Jersey, Mahwah, USA
amruth@ramapo.edu

Abstract. AI-ED community has hewed to rigorous evaluation of software tutors and their features. Most of these evaluations were done *in-ovo* or *in-vivo*. Can the results of these evaluations be replicated in *in-natura* evaluations? In our experience, the evidence for such replication has been mixed. We propose that the features of tutors that are found to be effective *in-ovo*/*in-vivo* might need motivational supports to also be effective *in-natura*. We speculate that some features may not transfer to *in-natura* use even with supports. Recognition of these issues might bridge the gap between AI-ED community and educational community at large.

Keywords: In-ovo, in-vivo, in-natura, replication of results.

1 Introduction

Evaluation of software tutors may be carried out in one of three settings:

- *In-ovo*: Research subjects hand-picked for the evaluation use the software tutor in a laboratory setting, typically under tightly controlled conditions, and under the supervision of the researcher.
- *In-vivo*: Students enrolled in a course use the software tutor in the class room, typically under tightly controlled conditions and under the supervision of the researcher or course instructor.
- *In-natura*: Students enrolled in a course use the software tutors, typically after class, on their own time, and unsupervised.

These three types of evaluation are summarized in Table 1.

Type	Location	Subjects	Conditions	Supervised
In-ovo	Laboratory	Recruited	Controlled	Yes
In-vivo	Classroom	Students enrolled in a course	Controlled	Yes
In-natura	After-class		Not controlled	No

Table 1: Types of evaluation of software tutors

AI-ED community has reported frequently using *in-ovo* and *in-vivo* evaluations in its studies of the effectiveness of software tutors and their features. Researchers have strictly controlled the conditions of these studies – what a subject can do or not do during the study, whether the subject is exposed to any distractions during the study, etc. – so as to minimize the influence of extraneous factors.

However, in real-life, especially at baccalaureate level, software tutors are less used as in-class exercises than as after-class assignments or study aides. The reasons for such use are many, including: course instructors may not want to spend valuable class time using software tutors; and students may not have access to (sufficient numbers of) computers during class.

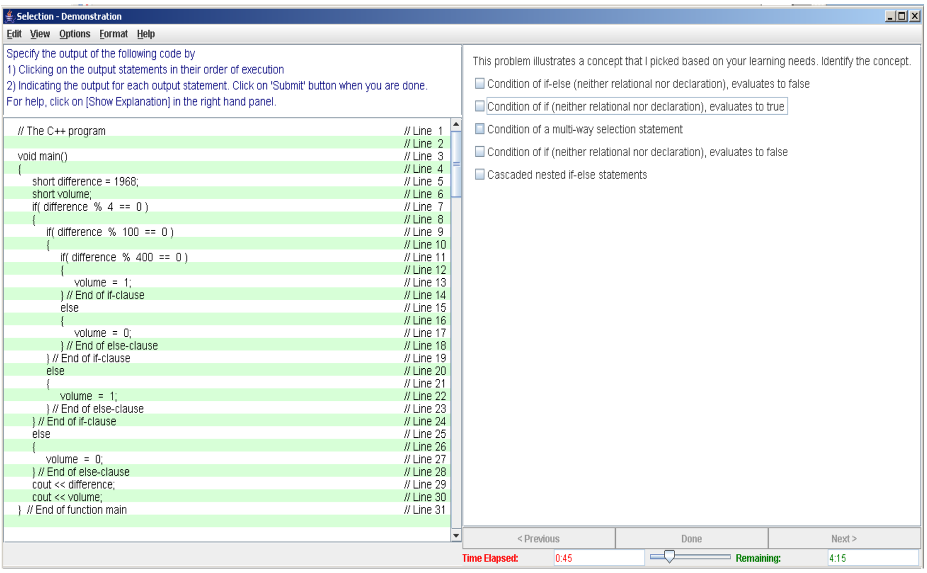
When software tutors are used for after-class assignments, mandatory or otherwise, issues of intrinsic and extrinsic motivation play a much larger role in their use and utility. For starters, the popular aphorism *If you build it, they will come* does not apply to software tutors – unless students are required to use a software tutor, they will not use it (in any significant numbers). This significantly drives down participation and may skew evaluation results because of the self-selected nature of subjects. When they do use it, extrinsic motivation often plays a larger role than intrinsic motivation – if they are awarded course grade proportional to how well they do on the software tutor, they are more likely to engage seriously with the tutor. On the other hand, if they are given credit simply for using the software tutor, they are likely to do the least amount of work possible to qualify for such credit.

Given these considerations, do the research results elicited under carefully controlled conditions *in-ovo* or *in*-vivo extend to *in-natura* use of software tutors? In other words, can results obtained *in-ovo* or *in-vivo* be replicated *in-natura*? Our experience has been mixed. We will present results from evaluations of two features – reflection and self-explanation - vouched for by the AI-ED community that did not pan out in our *in-natura* evaluations.

For our evaluations, we used software tutors for programming concepts, called problets (problets.org). These tutors are being used every semester by 50-60 schools, both undergraduate and high-school. Since problets are deployed over the web, students have access to the software tutors anytime, anywhere. Problets are set up to automatically administer pre-test-practice-post-test protocol every time they are used [5]. They have been continually used and evaluated *in-natura* since fall 2004.

2 Reflection

The benefits of post-practice reflection have been studied by several researchers (e.g., [3]). In problets, we introduced reflection in the form of a multiple-choice question presented after each problem. The question states "This problem illustrates a concept that I picked based on your learning needs. Identify the concept." The learner is provided five choices, each of which is a different concept in the domain. The learner must select the most appropriate concept on which the problem might be based, and cannot go on to the next problem until (s)/he correctly selects it. The problet records the number of unique concepts selected by the learner up to and including the most appropriate concept. See Figure 1 for a snapshot of the reflection question presented after the student has solved a problem on selection statements.



Specify the output of the following code by
1) Clicking on the output statements in their order of execution
2) Indicating the output for each output statement. Click on 'Submit' button when you are done.
For help, click on [Show Explanation] in the right hand panel.

```
// The C++ program                                  // Line  1
                                                    // Line  2
void main()                                         // Line  3
{                                                   // Line  4
    short difference = 1968;                        // Line  5
    short volume;                                   // Line  6
    if( difference  %  4  ==  0 )                   // Line  7
    {                                               // Line  8
        if( difference  %  100  ==  0 )             // Line  9
        {                                           // Line 10
            if( difference  %  400  ==  0 )         // Line 11
            {                                       // Line 12
                volume  =  1;                       // Line 13
            } // End of if-clause                   // Line 14
            else                                    // Line 15
            {                                       // Line 16
                volume  =  0;                       // Line 17
            } // End of else-clause                 // Line 18
        } // End of if-clause                       // Line 19
        else                                        // Line 20
        {                                           // Line 21
            volume  =  1;                           // Line 22
        } // End of else-clause                     // Line 23
    } // End of if-clause                           // Line 24
    else                                            // Line 25
    {                                               // Line 26
        volume  =  0;                               // Line 27
    } // End of else-clause                         // Line 28
    cout << difference;                             // Line 29
    cout << volume;                                 // Line 30
}  // End of function main                          // Line 31
```

This problem illustrates a concept that I picked based on your learning needs. Identify the concept.

□ Condition of if-else (neither relational nor declaration), evaluates to false
□ Condition of if (neither relational nor declaration), evaluates to true
□ Condition of a multi-way selection statement
□ Condition of if (neither relational nor declaration), evaluates to false
□ Cascaded nested if-else statements

Time Elapsed: 0:45 Remaining: 4:15

Figure 1: Selection tutor: Problem in the left panel; Reflection question in the right panel

We conducted several controlled evaluations of reflection [8] using selection and while loop tutors in 2006-07. Control group was never presented any reflection questions. Test group was presented a reflection question after each problem during pre-test, practice and post-test. If a student solved a problem incorrectly, the student was required to answer the subsequent reflection question correctly before going on to the next problem.

Practice was adaptive, and based on the student's performance on the pre-test. The entire protocol was limited to 30 minutes for control group and 33 minutes for test group. For analysis purposes, we considered only *practiced* concepts [5], i.e., concepts on which the student solved a problem incorrectly during pre-test, solved one or more problems during adaptive practice and also solved the post-test problem before running out of time.

Table 1 lists the score per problem on pre-test and post-test of all *practiced* concepts. No significant difference was found between control and test groups, indicating that the two groups were comparable. However, no significant difference was found in their pre-post improvement either, suggesting no differential effect of reflection on their learning. Please see [8] for additional details of the evaluation.

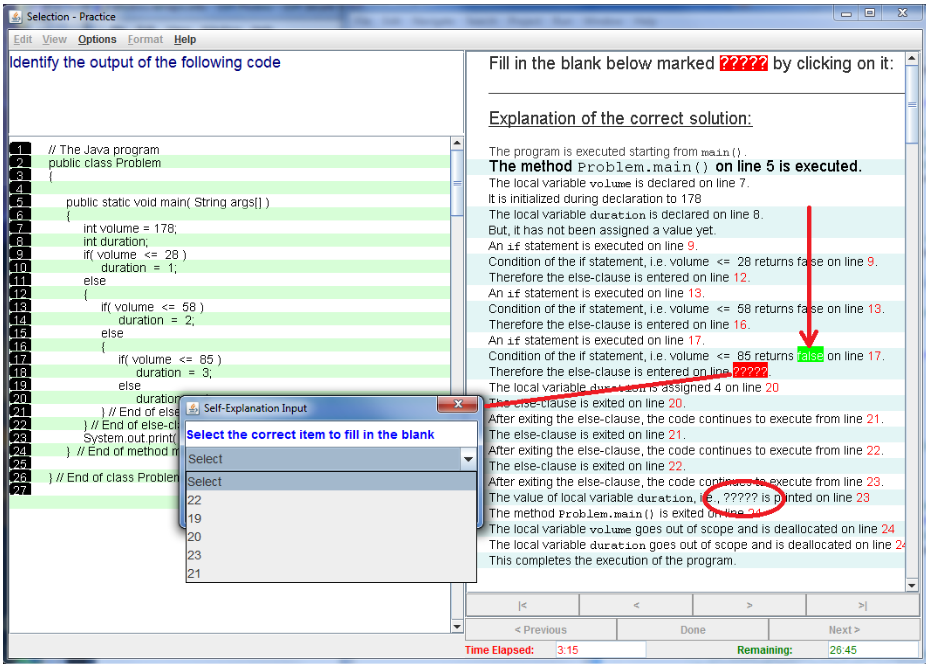
Score per problem	Pre-Test	Post-Test	Pre-post p
Control Group (Without Reflection) (N =89)			
Mean	0.118	0.736	< 0.001
Standard-Deviation	0.177	0.353	
Test Group (With Reflection) (N =152)			
Mean	0.144	0.787	< 0.001
Standard-Deviation	0.183	0.319	
Between groups p	0.283	0.266	

Table 1: Both the groups improved significantly from pre-test to post-test; the difference between the two groups was not significant on either the pre-test or the post-test

3 Self-Explanation

The effectiveness of providing self-explanation questions in worked examples has been well documented by AI-ED community (e.g., [1]).

Selection tutor was used for this study. When the student solves a problem incorrectly, the tutor presents feedback including step-by-step explanation of the correct execution of the program in the fashion of a fully worked-out example. Self-explanation questions were presented embedded in this step-by-step explanation, as shown in Figure 2. Each self-explanation question is a drop-down menu that deals with the semantics of the program, e.g., the value of a variable, the line to which control is transferred during execution, etc. The questions were independent of each other, but answering them required the student to closely read the step-by-step explanation/worked out example and understand the behavior of the program in question.



Identify the output of the following code

```
1  // The Java program
2  public class Problem
3  {
4
5      public static void main( String args[] )
6      {
7          int volume = 178;
8          int duration;
9          if( volume  <=  28 )
10             duration  =  1;
11         else
12         {
13             if( volume  <=  58 )
14                 duration  =  2;
15             else
16             {
17                 if( volume  <=  85 )
18                     duration  =  3;
19                 else
20                     duration
21             } // End of else
22         } // End of else-cl
23         System.out.print(
24     }  // End of method m
25
26 } // End of class Problem
27
```

Select the correct item to fill in the blank

Select

Select
22
19
20
23
21

Fill in the blank below marked ????? by clicking on it:

Explanation of the correct solution:

The program is executed starting from `main()`.
The method `Problem.main()` on line 5 is executed.
The local variable `volume` is declared on line 7.
It is initialized during declaration to 178
The local variable `duration` is declared on line 8.
But, it has not been assigned a value yet.
An `if` statement is executed on line 9.
Condition of the if statement, i.e. volume <= 28 returns false on line 9.
Therefore the else-clause is entered on line 12.
An `if` statement is executed on line 13.
Condition of the if statement, i.e. volume <= 58 returns false on line 13.
Therefore the else-clause is entered on line 16.
An `if` statement is executed on line 17.
Condition of the if statement, i.e. volume <= 85 returns false on line 17.
Therefore the else-clause is entered on line ?????.
The local variable `duration` is assigned 4 on line 20
The else-clause is exited on line 20.
After exiting the else-clause, the code continues to execute from line 21.
The else-clause is exited on line 21.
After exiting the else-clause, the code continues to execute from line 22.
The else-clause is exited on line 22.
After exiting the else-clause, the code continues to execute from line 23.
The value of local variable `duration`, i.e., ????? is printed on line 23
The method `Problem.main()` is exited on line 24
The local variable `volume` goes out of scope and is deallocated on line 24
The local variable `duration` goes out of scope and is deallocated on line 24
This completes the execution of the program.

Time Elapsed: 3:15 Remaining: 26:45

Figure 2. Snapshot of selection tutor with self-explanation questions displayed in the right panel

So as not to overwhelm the student, the tutor limited the number of self-explanation questions per problem to three. The student was allowed as many attempts as needed, but had to answer each self-explanation question correctly before proceeding to the next question, and had to answer all the self-explanation questions correctly before proceeding to the next problem. A version of the tutor was used for the control group that did not present any self-explanation questions. This version of the tutor allowed the learner to advance to the next problem as soon as it displayed step-by-step explanation of the current problem.

Controlled evaluation of selection tutor was conducted *in-natura* over three semesters: fall 2012-fall 2013 [4]. No significant difference was found in the average score per pre-test problem between control (N = 395) and test (N = 335) groups [F(1,729) = 1.018, p = 0.313]. So, the two groups were equivalent. The mean number of concepts practiced by control group was 1.62, and by test group was 1.78. However, since control group was allowed 30 minutes to practice with the tutor and test group was allowed 40 minutes, univariate analysis of the number of concepts practiced was conducted with self-explanation as the fixed factor and total time spent as the covariate. The difference between the two groups was found to be significant [F(2,597) = 62.207, p < 0.001]: accounting for the extra time allowed, control group practiced 1.72 ± 0.11 concepts whereas test group practiced 1.662 ± 0.12 concepts. Therefore, test group practiced significantly fewer concepts than control group. No significant difference was found between the two groups on the pre-post change in score on practiced concepts, suggesting no differential effect of self-explanation on learning. Please see [4] for additional details of the evaluation.

4 Discussion

In both the studies – on reflection and self-explanation – we have verified that our implementation is behaviorally similar to, if not the same as described in at least some of the literature on the topic published in the AI-ED community. Even if our interpretation of both reflection and self-explanation behaviors differs enough from those reported in literature to render our treatments ineffective, we would expect that the increased time-on-task due to these faux treatments would have still yielded some learning benefits.

Our evaluations cannot be faulted for inadequate participation – our evaluations have typically involved 200-300 students, which is an order of magnitude larger than the number of subjects reported in typical *in-ovo* and *in-vivo* evaluations.

We have used standard protocols for evaluation – controlled studies, pre-test-practice-post-test protocol and partial crossover design. We have used ANOVA for data analysis. In our studies, we have considered only *practiced* concepts – concepts on which students solved problems during all three stages of the protocol: pre-test, practice and post-test, so noise is not an issue in the analyzed data.

These practices have been effective - not all our evaluations have come up empty, e.g., we have found significant effect of providing error-flagging feedback on test performance (e.g., [6]), and significant stereotype threat (e.g., [7]).

An explanation for the lack of results might be the difference in student motivation in *in-ovo*/*in-vivo* versus *in-natura* evaluation. Apart from issues of extrinsic motivation mentioned earlier, it may also be argued that given the lack of supervision in *in-*

natura evaluation, students are less likely to experience *Hawthorne effect* [2]. So, the features of tutors that are found to be effective in *in-ovo*/*in-vivo* evaluations might need motivational support to also be effective in *in-natura* evaluations.

Then again, even with motivational support, students may resent having to perform tasks (such as answering questions on reflection) that they do not perceive as directly contributing to their assignment at hand, and may not participate in, or may not be amenable to benefiting from what they view as a chore. In other words, some features may not be transferable from the laboratory to the field regardless of the supports provided.

While we have focused on the transferability of evaluation results from lab/classroom to after-class setting, researchers have reported similar issues transferring results from the lab to the classroom, e.g., in a study of politeness in intelligent tutors [9], researchers reported finding weaker results when the study was conducted in a classroom rather than a laboratory. They speculated that grades, an extrinsic motivational factor, may be to blame. Furthermore, they wrote [9], "In the rough-and-tumble of the classroom, with its noise, question-asking, and social environment, students *may simply not concentrate as much on the feedback provided by the computer tutor*. The lab setting, on the other hand, is a quiet environment where subjects work on their own with few distractions, and certainly none from classmates and a teacher" (italics not in the original). The noise, distractions and lack of structure used to describe a classroom as compared to laboratory setting are the very same terms, magnified, that could be used to describe an after-class setting as compared to a classroom. In other words, when it comes to noise, distractions and lack of structure, laboratory and after-class setting are at opposite ends of a spectrum, with the classroom situated in between. That *students may not concentrate as much on the feedback provided by the tutor* may explain why reflection and self-explanation, both provided as part of feedback, failed to live up to expectation in our *in-natura* evaluations.

It appears that *in-natura* use of software tutors entails more than just large-scale/unsupervised deployment of *in-vivo* results and *in-vivo* use entails more than just live-classroom deployment of *in-ovo* results. Motivational supports may be needed to transition results from the laboratory to the field and some results found in the laboratory may fail to transfer to the field even with motivational supports. Treating *in-natura* use of software tutors as being distinct from *in-ovo/in-vivo* uses is reminiscent of the outgrowth of Chemical Engineering as a discipline of the field from Chemistry as a discipline of the laboratory. While Chemistry is the study of properties of materials, Chemical Engineering is the study of the production of materials on an industrial scale, albeit with its basics firmly rooted in Chemistry. In the early years, chemists refused to accept Chemical Engineering as anything more than Chemistry, and engineers refused to recognize Chemical Engineering as an engineering discipline [10], but not so any more. May be AI-ED community should treat *in-natura, in-vivo* and *in-ovo* as three independent, necessary and valuable stages in the evaluation of any treatment. May be, *in-natura* evaluation is what is needed for educational community at large (especially higher-education community) to recognize and incorporate the important pedagogical insights being offered by AI-ED community.

Acknowledgments. Partial support for this work was provided by the National Science Foundation under grants DUE-0817187 and DUE-1432190**.**

5 References

1. Conati, C. and VanLehn, K. (1999) Teaching meta-cognitive skills: implementation and evaluation of a tutoring system to guide self-explanation while learning from examples. Proc. AI-ED 99, 297-304.
2. Franke, R.H. and Kaul, J.D. The Hawthorne experiments: First statistical interpretation. *American Sociological Review*. Vol 43. 1978. 623-643.
3. Katz, S., O'Donnell, G., Kay, H. (2000) An Approach to Analyzing the Role and Structure of Reflective Dialogue. International Journal of Artificial Intelligence in Education, 11, 320-343.
4. Kumar, A.N. An Evaluation of Self-Explanation in a Programming Tutor. In Proc. Of ITS 2014, Hawaii, June 2014. 248-253.
5. Kumar, A.N. A Model for Deploying Software Tutors. IEEE 6th International Conference on Technology for Education (T4E). Amritapuri, India, 12/18-21/2014, 3-9.
6. Kumar, A.N. Limiting the Number of Revisions While Providing Error-Flagging Support During Tests. Proc. Intelligent Tutoring Systems (ITS 2012), LNCS 7315, Chania, Crete, 6/14-18/2012, 524-530.
7. Kumar, A.N. A Study of Stereotype Threat in Computer Science. Proceedings of Innovation and Technology in Computer Science Education (ITiCSE 2012). Haifa, Israel, 7/3-5/2012, 273-278.
8. Kumar, A.N. Promoting Reflection and its Effect on Learning in a Programming Tutor. Proceedings of 22nd International FLAIRS conference on Artificial Intelligence (FLAIRS 2009) Special Track on Intelligent Tutoring Systems, Sanibel Island, FL, May 19-21, 2009, 454-459.
9. McLaren, B.M., DeLeeuw, K.E., and Mayer, R.E. Polite web-based intelligent tutors: Can they improve learning in classrooms? *Computers and Education*. 56(3): 574-584, 2011.
10. Reynolds, Terry S. Engineering, Chemical, in Rothenberg, Marc, *History of Science in United States: An Encyclopedia*, New York City: Garland Publishing, 2001. ISBN 0-8153-0762-4