Response to "Comments on 'Formal Methods Application: An Empirical Tale of Software Development'"

Ann E. Kelley Sobel and Michael R. Clarkson

Abstract—We respond to criticism by D. Berry and W. Tichy of our paper that appeared in the March 2002 issue of *IEEE Transactions on Software Engineering*. Many of the supposed faults they identify in our experiment are a result of a misunderstanding on their part, while others are inherent aspects of an educational experiment. We present counterarguments that explain why our experiment is valid.

1 Introduction

We appreciate the opportunity to elaborate upon the contributions presented in our paper appearing in the March 2002 issue of *TSE* [12] (hereafter, "the *TSE* paper"). The paper describes one evaluation given during a three-year educational experiment. The experiment addressed whether the knowledge and skills acquired by learning and applying formal methods during an undergraduate student's core curriculum would increase the level of the student's complex problem solving skills [11], [12]. The experiment used a control group that did not receive instruction in formal methods. To our knowledge, this was the first work to address empirically the benefits of formal methods using a controlled experiment. It was also the first longitudinal study of teaching formal methods in an undergraduate curriculum.

One of the most important results in the *TSE* paper was that, on a particular programming assignment, 100 percent of the experimental group's programs passed all test cases, whereas only 45.5 percent of the control group's programs passed all test cases. This is statistically significant at the $p = .05$ level.[1] The conclusion drawn in the paper is that the use of formal methods caused the increase in correctness for the experimental group. The paper infers that this supports the common belief of formal analysis advocates that the use of formal analysis during software development produces "better" programs [7], [6], [10].

The note [1] from Berry and Tichy (hereafter, "BT") critiques our experiment. Before examining their specific criticisms, there is a fundamental misunderstanding in most of BT's comments on our experiment that we must correct. They often treat the experiment reported in the *TSE* paper as a laboratory experiment testing whether formal analysis produces programs that are more correct. However, it was actually one part of a larger educational experiment testing whether instruction in formal methods causes students' complex problem solving skills to increase. These differences—performing the experiment in a laboratory environment versus an educational environment, whether the experiment was isolated or part of a larger experiment,[2] and whether the experiment addressed software development or curriculum design—impact nearly every issue raised by BT. We address this in more detail in Section 2 through a discussion of our experimental design.

Despite the misperception just discussed, BT are correct that any experiment must refute possible threats to validity. We argue in Section 3 that most of the specific threats identified by BT are implausible or fallacious. Section 4 discusses the related work cited by BT and Section 5 concludes.

2 Nature of the Experiment

The primary research question addressed by the experiment was "What are the effects of teaching formal methods to undergraduate software engineering students?" Throughout the experiment, the null hypothesis "Instruction in formal methods causes no change in the students' skill in solving complex problems" was tested. The independent variable implied by this hypothesis is instruction in formal methods, hence the use of a control group that did not receive this instruction. BT suggest an alternate design that involves teaching both groups formal methods [1, Section 3.1], but this would not be useful for testing our desired hypothesis. It would only be useful if the particular experiment reported in the *TSE* paper, which was one part of the whole three-year experiment, were to be isolated and used as part of an experiment addressing software development rather than curriculum design.

The dependent variable implied by our null hypothesis is skill in solving complex problems. The experiment used the ACT as a pretest measure of the dependent variable. Several posttests were administered throughout the experiment. One posttest reported in the *TSE* paper was the number of test cases passed in a particular programming assignment. This is reasonable since programming is a natural example of complex problem solving. This posttest, along with two others described in [11], provided evidence that the null hypothesis should be rejected.

The experimental group was self-assigned in this experiment, which means that the students were allowed to choose whether to join the control or experimental group. Instead of self-assignment, ideal laboratory experiments use random assignment to ensure internal validity. However, this experiment was not laboratory research, but rather field research:

> In field research, we cannot always assign participants randomly; rather, they must be taken as intact groups. Educational experiments, for example, are sometimes restricted to using one intact class for one group of participants and another class for another group of participants [3, p. 241].

This is exactly what we faced: It was impossible to assign students randomly to a particular three-year curriculum track. Thus, the self-assignment was mandatory. We see no way of overcoming this problem in any traditional academic setting. Since full experimental control (in this case, randomization) could not be achieved, our design is by definition a quasi-experiment [2, p. 34].

BT characterize our design as a Non-Equivalent Posttest-Only (NEPO) design, which is classified as faulty in [3]. NEPO, a design for laboratory experiments, uses a control group and an experimental group where the population is not randomly assigned into the two groups. There are two reasons why this can be considered faulty. The first is a fault in the design. Random assignment is a standard feature of laboratory experiments [2]; omitting it creates

1. Based on the Mann-Whitney test, as suggested in [1] and confirmed by our own calculations. There were six programs from the experimental group and 11 from the control group available for testing [12, Section 6].

2. This misperception is even reflected in a misquote by BT of the *TSE* paper. In [1, Section 3.4] they have written "this experiment," whereas the *TSE* paper reads "this particular experiment." The word "particular" is crucial because it places the posttest measurement within the context of the three-year experiment.

- *A.E.K. Sobel is with the Department of Computer Science and Systems Analysis, Miami University. E-mail: sobelae@muohio.edu.*
- *M.R. Clarkson is with the Department of Computer Science, Cornell University. E-mail: clarkson@cs.cornell.edu.*

Manuscript received 4 Mar. 2003; accepted 4 Mar. 2003.
Recommended for acceptance by D. Rosenblum.
For information on obtaining reprints of this article, please send e-mail to: tse@computer.org, and reference IEEECS Log Number 118384.

nonequivalent groups that cause threats to internal validity. The second is a fault on the part of the experimenter. By choosing to equate the two groups in ways other than randomization, the experimenter has failed to meet the standards of laboratory experiments.

Based on the discussion above, we must reject BT's claims about the faultiness of our design. First, NEPO is a design for laboratory experiments, whereas our experiment was a quasi-experiment. Nonequivalent groups are a standard feature of quasi-experimental designs [4]. To claim that this aspect of our design is faulty is to denounce a large body of other quasi-experiments as faulty. Second, we did not choose to omit random assignment; there simply was no possibility to include it. Thus we used the next best technique, which is matching on relevant variables [3, p. 238].

An accurate characterization of our design as a quasi-experiment can be found in [4, p. 520]. Like all quasi-experimental designs, the threats to validity that result from using nonequivalent groups are present. Despite this, the design can be "quite strong" under specific circumstances [4, p. 523]. Though we were unable to achieve all these circumstances, one that we did use was a matched control group. Another element that strengthens this design is the use of proxy pretests (pretests that cannot be directly scaled to posttests) [4, p. 521]. In our design, the ACT scores served this purpose. It would have been desirable to have a direct comparison available as a pretest; however, it is unclear how to design a pretest that would be meaningful yet inarguably comparable to the programming assignment posttest.

Regardless of the label or strength of the design used in the experiment, the real issue is establishing a causal inference by excluding alternative hypotheses. Design only facilitates causal inference; it does not guarantee it: "the structure of a quasi-experiment does not by itself determine the quality of a causal inference" [4, p. 523]. In other words, excluding alternate hypotheses is the most important task, whether it is done through the design of the experiment or by external means. In Section 3, we respond to the alternative hypotheses noted by BT and argue that most of them are implausible or fallacious.

3 Alternative Hypotheses

BT's note suggests several alternative hypotheses to explain the results of the *TSE* paper. These include motivation, exposure, learning style, skills, lack of control, Hawthorne and novelty effects, and measurement errors. Of these, we hold that only motivation and novelty effects are plausible confounds.

3.1 Extraneous Variables

Motivation. BT claim that the experimental group demonstrated greater motivation than the control group by volunteering for a more difficult curriculum, taking extra courses, and undertaking a full verification exercise [1, Section 3.1]. The last of these is partially plausible since the team that performed the full formal analysis did demonstrate higher motivation. However, no strong evidence exists to suggest that the entire experimental group was highly motivated. To the contrary, the full verification team accounts for less than 25 percent of the population of the experimental group. Furthermore, the curriculum was not advertised as being more difficult, even if it did emerge to be.[3] Nor did the experimental group take extra courses, as described below. Nonetheless, it would be advisable for future curriculum experiments to include controls for motivation.

Exposure. It is not the case that the experimental group received "significantly more instruction," as BT claim [1, Section 3.1]. In fact, the experimental group received marginal extra instruction in programming—only one additional coding assignment to the point of the project described in the *TSE* paper. The additional instruction that the group received was in axiomatic and algebraic semantics. The two groups were instructed equally in all other aspects of programming. The students also incurred no additional classes beyond what were required for the major; the two additional formal analysis classes were permitted to count toward six of the 12 elective hours that are required for the major. We therefore find exposure to be implausible as an alternative hypothesis.

Learning style. From the results of a learning style survey administered at the end of the three-year curriculum to the experimental group, BT infer a possible alternative hypothesis: "subjects in the experimental group were better learners or harder workers" [1, Section 3.1]. There are two problems with this inference. First, the survey does not measure the students' learning styles before or at the midpoint of the experiment, which is when the project analyzed in the *TSE* paper occurred. Of course, hindsight reveals that it would have been prudent to administer the survey as both a pretest and posttest to both groups. Second, the connection between having a particular learning style and being a better learner is apparently supposition on the part of BT and may even be fallacious. Thus, although we cannot exclude this hypothesis, we cannot find any reason to support it.

Skill. A standardized test, the ACT, was used as a proxy pretest for the purpose of matching in our experiment. It revealed no statistical difference between the two groups of students [11], [12]. Despite this, BT claim that the experimental group may still have included more talented students [1, Section 3.1]. They make this claim on the basis of a GRE exam and a hypothesis by Berry about the nature of those who apply formal methods. The GRE exam (consisting of questions from the "general" section of the Computer Science GRE) was given at the end of the three-year curriculum and revealed that the experimental group performed over 30 percent better than the control group. This result is further evidence that the null hypothesis of our experiment should be rejected, i.e., that instruction in formal methods increased the complex problem solving skills of the experimental group. The test is not a measure of the initial skill of the two groups, thus it was not mentioned in the *TSE* paper, which focused on only the first three semesters of the curriculum.

BT claim that "it is unlikely that exposure to formal methods alone improved general analytic skills by over thirty percent." Admittedly, it is a surprisingly large increase. A plausible explanation is that in applying formal analysis, the experimental group gained much more practice in logical reasoning (through first-order logic in particular). But, if one rejects this explanation, as have BT, there are two alternate explanations. The first is that the ACT was unable to differentiate the two groups. The ACT does measure other skills in addition to problem solving, but it is also sufficiently accurate that it is a widely used test instrument for differentiating students in college admissions. Thus, it is unlikely that the ACT would introduce exactly the right perturbation to make the problem solving skills of the two groups appear identical if they were actually different.

The second explanation is that a maturation interaction occurred in which the experimental group's skill would have naturally developed more rapidly than the control group's, even without the treatment effect. Christensen states that evidence of this could be obtained by comparing the variance of the pretest and posttest scores: If variability increases between the tests, a maturation interaction may have occurred [3, second ed., p. 206]. The Levene test can be used for this comparison. It tests the null hypothesis that the variances remain equal from pretest to posttest. The median-based form of it is well suited to data that may be nonnormal, such as ours [8]. When applied to our data, it fails to

3. Our goal, after all, was to have as many students in the experimental group as possible, not to scare them away!

reject the null hypothesis for $p = .05$. Moreover, it continues to fail to reject it for $p \leq .4$. This provides strong evidence that the variability did not increase between tests, and therefore that the "talent" of the two groups was indeed equivalent. Thus, we find BT's claim about the GRE exam to be fallacious.

As for Berry's hypothesis [1, Section 3.1], all freshmen students were presented with this alternative course track during their first programming class. To claim that these students chose this experimental course track because they "were more adept than those in the control group at getting to the heart of the problem, abstracting from extraneous detail, and carefully organizing their whole approach to solving the problem" clearly misrepresents the intellectual maturity that a freshman possesses. As the instructor of the first formal analysis class attests in personal communication, the students in the experimental group had little knowledge of what formal methods were, let alone that they were related to abstraction. Although we can see how this hypothesis might be applicable in other settings, we find it implausible in the context of first-year undergraduate students.

3.2 Presence of Controls

BT object that the control group's design behavior was not sufficiently controlled to be comparable to the experimental group's behavior [1, Section 3.3]. The behavior of the two groups was as follows. First, the formal group submitted formal specifications. Then, both groups implemented their solutions, performing informal design in the process. Some students from the experimental group, but none from the control group, elected to submit design documentation.

Since neither group was required to submit design documentation, BT take the absence of documentation from the control group as evidence that at least some of the control teams performed little or no design before starting to program. They generalize from this and conclude that too little is known about the design behavior of the control group for it to serve its purpose as a comparison group.

Even though we did not collect physical evidence of the design behavior of both groups, absence of proof is not proof of absence. Informal analysis must have been performed given the context of this team programming assignment: It occurred in an object-oriented design (OOD) course and, in this course, standard informal design techniques are taught, assigned, and their use stressed. Also, as with any team assignment, some coordination of programming activities among the team members must take place. This necessity supports the existence of a design.

The degree to which informal analysis was applied, based on examination of source code, is described in the *TSE* paper and appeared to vary similarly in both groups. Thus, the design behavior from the control group does not differ from the experimental group: Both groups show examples of poor or nonexistent designs and examples of good designs. Even the formal specifications exhibited a range of quality. From this, we conclude that we do know enough about the behavior of both groups to claim that the control group was indeed controlled.

Furthermore, control was maintained over many extraneous variables that are much more important to a curriculum study. For each class that overlapped between the control and experimental curricula, the two groups took that class during the same semester. Both sections (control and experimental) of the class were taught by the same professor, used the same textbooks and lectures, and were given the same programming assignments and exams. All this required surprisingly intense effort.

3.3 Hawthorne and Novelty Effects

The term *Hawthorne effect* refers to a subject's performance in an experiment being affected by knowledge that he or she is participating in the experiment. In particular, BT suggest [1, Section 3.2] that, since the experimental group knew the experimenters had an expectation of higher performance from it, the students in it did indeed perform better. We find two flaws in this argument. First, one of the most recent and authoritative research handbooks does not support the general existence of Hawthorne effects:

> Despite the widespread discussion of treatment confounds presumed to result from wanting to give data that will please researchers—which we suspect is a result of discussions of the Hawthorne effect—there is neither widespread evidence of the Hawthorne effect in field experiments (see...[4]) nor... [4, p 508].

Second, knowledge of a hypothesis does not imply motivation or ability to comply with it [4, p. 508]. Consider that our groups were presumably very motivated to perform well on the ACT, yet it showed they had the same ability. Thus, even though our experimental group knew it was participating in an experiment, it is unlikely it had the ability to perform better. In light of these points, we find Hawthorne effects to be implausible in our experiment.

Novelty effects are similar to Hawthorne effects. They refer to outcomes resulting from the perceived new or unusual nature of the treatment. In our experiment, the experimental group did initially perceive instruction in formal methods as new and unusual and, thus, a novelty effect may have played some role in the outcome. But, at the point in time that the *TSE* paper describes, one and a half years (a significant amount of time for an undergraduate) into the experiment, the novelty may have diminished. A questionnaire would have helped to determine whether this was the case. Unfortunately, novelty effects are an inherent aspect of any educational experiment addressing the injection of new material into a curriculum.

3.4 Measurement Errors

The *TSE* paper reports functional correctness on a set of six test cases. BT argue [1, Section 3.4] that the experimental group could have performed better than the control group on this set entirely because of the choice of test cases. Instead, they suggest that automated test case generation and unit testing should have been used [1, Section 3.4]. However, these were infeasible in the case of the project described in the *TSE* paper. One reason for this is that the students were not required to implement a standard interface that could be used for testing. Future experiments should design assignments with this in mind. Another reason is that we did not have the resources to perform such sophisticated testing. It was also beyond our resources to require students to work in a programming environment with automatic capture of events or be monitored by a proctor in a laboratory (both suggested in BT [1, Section 3.3]).

As for the six test cases actually used, four of them exercised the most basic control paths through the system, while the remaining two provided very basic tests of efficiency and liveness. Programs that could not pass these cases would be unlikely to pass the multitude of cases suggested by BT, making automated testing of dubious value.

4 Related Work

BT's note criticizes the *TSE* paper for not citing a paper by Pfleeger and Hatton [9]. The Pfleeger and Hatton study reports on experiences with the application of formal methods in an industrial setting where professionals developed safety-critical software. In contrast, the *TSE* paper reports on an experience during the learning and application of formal methods in an

4. Here, we omit a list of citations.

instructional setting where undergraduate students developed a classroom programming assignment. Since we have never tried to generalize the results of our curriculum study to industrial software development, the difference in environment, personnel, and product make [9] no more relevant to the *TSE* paper than the many other uncited industrial applications of formal methods (but, for a survey of these, see [5]).

5 Conclusion

Experimental design is indeed tricky, which led Berry and Tichy to claim that our work exhibited sufficient problems to call for a redesign of the experiment. We have shown above that they misunderstood some aspects of our design, notably whether it was a laboratory experiment or an educational experiment, whether it was isolated or part of a larger experiment, and whether it addressed software development or curriculum design. Many of their alternative hypotheses were implausible or fallacious. The remaining issues they identify are not so severe as to invalidate our experiment. Our research design was sound and our controls were appropriate, though we do not deny that hindsight reveals improvements that could have been made. Perhaps additional tests, which could have definitively excluded other alternative hypotheses, should have been administered. If so, we hope our work has brought these to light. But realistically, only so many tests can be performed. As Christensen emphasizes:

> The point is that we can never be certain that complete control has been effected in the experiment. All we can do is increase the probability that we have attained the desired control of the confounding extraneous variables that would be sources of rival hypotheses [3, p. 228].

We remain convinced that, for the first attempt at a three-year educational experiment in formal methods, there are few ways in which our experiment could have been improved without either the benefit of hindsight or greater resources. We agree with Berry and Tichy that our discipline is in need of additional controlled experiments, particularly since we support making curriculum changes based on experimental data rather than the usual ad hoc approaches. Thus, we hope that those who strive to continue our work will establish even more definitive results thanks to this dialog.

Acknowledgments

The authors would like to thank David Gries, Andrew Myers, Nathaniel Nystrom, Alton Sanders, and Fred Schneider for discussions on the issues raised in this note.

References

[1] D.M. Berry and W.F. Tichy, "Comments on 'Formal Methods Application: An Empirical Tale of Software Development,'" *IEEE Trans. Software Eng.*, vol. 29, no. 6, pp. ???-???, 2003.
[2] D.T. Campbell and J.C. Stanley, *Experimental and Quasi-Experimental Designs for Research.* Rand McNally, 1963.
[3] L.B. Christensen, *Experimental Methodology,* eighth ed. Allyn and Bacon, 2002.
[4] T.D. Cook, D.T. Campbell, and L. Peracchio, "Quasi Experimentation," *Handbook of Industrial and Organizational Psychology,* M.D. Dunnette and L. M. Hough, eds., vol. 1, pp. 491-576, 1998.
[5] D. Craigen, S. Gerhart, and T. Ralson, "An International Survey of Industrial Applications of Formal Methods," Technical Report GCR 93/626, NIST, 1993.
[6] E.W. Dijkstra, "On the Cruelty of Really Teaching Computer Science," *Comm. ACM* vol. 32, no. 12, pp. 1398-1404 Dec 1989.
[7] D. Gries, "Teaching Calculation and Discrimination: A More Effective Curriculum," *Comm. ACM,* vol. 34, no. 3, pp. 44-55, Mar. 1991.
[8] NIST, *NIST/SEMATECH e-Handbook of Statistical Methods,* Jan. 2003, http://www.itl.nist.gov/div898/handbook.
[9] S.L. Pfleeger and L. Hatton, "Investigating the Influence of Formal Methods," *Computer,* vol. 30, no. 2, pp. 33-43, Feb. 1997.
[10] H. Saiedian, "An Invitation to Formal Methods," *Computer* vol. 29, no. 4, pp. 16-30, Apr. 1996.
[11] A.E.K. Sobel, "Empirical Results of a Software Engineering Curriculum Incorporating Formal Methods," *ACM SIGCSE Inroads,* vol. 32, no. 1, pp. 157-161, Mar. 2000.
[12] A.E.K. Sobel and M.R. Clarkson, "Formal Methods Application: An Empirical Tale of Software Development," *IEEE Trans. Software Eng.*, vol. 28, no. 3, pp. 308-320, Mar. 2002.