# Grounding the Scholarship of Teaching and Learning in Practice

JOSH TENENBERG
University of Washington, Tacoma
and
ROBERT MCCARTNEY
University of Connecticut

We invite submissions to ACM/JERIC from teaching practitioners who are our readers. Grounding these submissions in the lived practice of teaching; using data that is already collected to assess student learning; and reporting the data collection, analysis, and context of use accurately and honestly are key aspects for taking a scholarly approach to teaching and learning. We hope to see more such scholarly inquiries by computing educators in the pages of future issues.

Categories and Subject Descriptors: K.3.2 [**Computers and Education**]: Computer and Information Science Education—*Curriculum*; *Computer Science Education*

General Terms: Measurement, Reliability

Additional Key Words and Phrases: Scholarship of teaching and learning, practitioner knowledge

**ACM Reference Format:**
Tenenberg, J. and McCartney, R. 2008. Grounding the scholarship of teaching and learning in practice. ACM J. Educ. Resour. Comput. 8, 2, Article 4 (June 2008), 3 pages. DOI = 10.1145/1362787.1362788. http://doi.acm.org/10.1145/1362787.1362788.

We recently attended the ACM/SIGCSE Annual Symposium in Portland, Oregon. In this editorial, we would like to reiterate the themes that we discussed in the panel at SIGCSE'08 entitled "Publishing in Computing Education" [Tenenberg 2008].

The primary audience for which we are targeting ACM/JERIC consists of teaching practitioners in computing. We direct the following suggestions

Authors' addresses: J. Tenenberg, Institute of Technology, University of Washington, Tacoma, 1900 Commerce St., Tacoma, WA 98402; e-mail: jtenenbg@u.washington.edu; R. McCartney, Department of Computer Science and Engineering, University of Connecticut, Storrs, CT 06269-2155; e-mail: robert@engr.uconn.edu.

to those in this audience who are interested in publishing in ACM/ JERIC.

(1) *Interrogate your own practice.* Topics of interest and relevance to others will arise from the workaday world in which computing is taught. We believe that teachers in diverse settings across the globe are responding creatively to the lived conditions that they face. We invite our readers to interrogate their own practices to unearth and make explicit the innovations that they may have already normalized and take for granted.

(2) *Examine the data that you already collect.* In the normal course of instruction, students generate artifacts that provide evidence of learning: programs, documentation, designs, proofs, and textual descriptions. These serve as the raw material from which teachers assess learning and provide feedback to students. Looked at systematically, these artifacts can also be considered "data" that can provide insights into questions about learning.

(3) *Ask questions about learning that can be answered by the data at hand.* These questions will be shaped by the particular artifacts, but might include such things as "what are the errors that students are making?" "Are there places in a course where student performance diverges?" "Is there correspondence between different representations for the same thing?" and "Do students understand graphical encodings of program designs?" to name just a few. Note that there are a number of interesting questions that data at hand can answer that are not of the form "is X better than Y" (e.g., "Is pair programming better than individual programming?" "Is an objects-first approach better than an objects-late approach?") In analyzing the data collected, it is not uncommon to perform such actions as categorizing, counting, finding correspondences, and seeking patterns. The point that we wish to underscore is that the questions asked can be driven by the data that emerges from the practice of teaching.

(4) *Accurately and honestly report the methods of data collection and analysis.* This provides an audit trail so that others can replicate the same inquiry within their own setting. In addition, it improves the credibility given to the author by the reader. This reporting includes the number of students, the data collection instruments used, whether the data was mandatory or voluntary, the instructions given to students, the time that students had in which to complete their work, and any other details that might have an influence on the work that students produced. In addition, the basis for any inferences made (e.g., statistical tests, coding schemes) should also be disclosed.

(5) *Provide descriptive information about the institutional context and the student demographics.* A teaching intervention may "work" based on subtle interactions between aspects of context, chracteristics of students, and the particulars of the teaching intervention. What works in a resource-poor polytechnic with an open access admissions policy in New Zealand might not work within a selective liberal arts university in the eastern United States. Assumptions about culture, context, and students might be implicitly embedded within the teaching intervention; clever teachers often do

this without explicit awareness. Although it is the responsibility of the reader to determine if any particular intervention might be transferable to a new setting, this judgement is enhanced if the particulars of the originating context are disclosed.

In summary, we are inviting additional submissions from teaching practitioners who are our readers. Grounding these submissions in the lived practice of teaching, using data that is already collected to assess student learning, and reporting the data collection, analysis, and context of use accurately and honestly are key aspects for taking a scholarly approach to teaching and learning. We hope to see more such scholarly inquiries by computing educators in the pages of future issues.

In this issue, Victoria Sakhnini and Orit Hazzan present a study that inquires into the difficulties and heuristics that high school students face in dealing with abstract data types in a software design course. The authors used a "think aloud" method for data collection, where students verbalize their mental operations while solving an abstract data type problem. Hazzan's previously articulated framework for reducing abstraction served as a theoretical overlay to the data analysis; students made both deliberate and unintentional choices in order to reduce the level of abstraction of problems in order to make them more manageable.

In the article by Zendler and Spannagel, the authors are concerned with identifying the central concepts in computer science education. Rather than relying on either intuition or a top-down curriculum development approach, they took an empirical approach to this problem by surveying 37 computer science experts at the top universities in Germany. The surveys were based on the frequently mentioned concepts from the ACM Computing Classification System, and the experts were asked to rank these concepts according to specific criteria related to centrality within the discipline. Using a variety of statistical methods of analysis, the authors provide a catalog of concepts that are applicable in multiple ways across computing, can be taught at every intellectual level, have stood the test of time, and are connected to everyday language.

## REFERENCE

TENENBERG, J., FINCHER, S., IMPAGLIAZZO, J., AND JOYCE, D. 2008. Publishing in computer education. In *Proceedings of the 39th Technical Symposium on Computer Science Education (SIGCSE'08)*.

Received April 2008; accepted April 2008.