education review // reseñas educativas
a multi-lingual journal of book reviews

editors: gustavo e. fischman / melissa cast-brede / gene v glass

Supported by the Mary Lou Fulton Teachers College, Arizona State University

October 13, 2014 ISSN 1094-5296

Kadijevich, D. M., Angeli, C., & Schulte, C. (Eds.) (2013). *Improving computer science education*. New York, NY: Routledge.

Pp. xii+168 ISBN 978-0415645379

Reviewed by *Jose Victor Lineros*
University of North Texas
United States

Pedagogical books about computer science are generally underrepresented. A contributing cause, effectively highlighted in the book *Improving Computer Science Education,* is the profession's relative newness. Mark Guzdial, in the foreword, smartly points out that while the National Council for Teachers of Mathematics was founded almost 100 years ago, the comparable Computer Science Teachers Association has only existed since 2005. This relative nascence has undoubtedly contributed to more in-house and intra-institutional teaching practices than other professions. The relatively smaller number of instructors and students has also accentuated this difference through cloistered mindsets and isolated teaching practices. Another difficulty – also shared with other disciplines – has been finding a way to apply the sometimes dense instructional research findings into actionable teaching practices.

Directly addressing this gap, the book contains various applications of current learning pedagogies through a wide array of contributions from various authors. The use of different authors diversifies the learning approaches in order to expose the reader to different research findings and their varied classroom applicability. Throughout the book, tangible examples and strategies are presented to optimize classroom success. Although disparate authors are utilized, an attempt is made to present the material in an integrated fashion to maximize the impact of the content.

Organizationally, the book is divided into three main sections: 1 – *Improving Learning*, 2 – *Methodological Perspectives*, and 3 – *Improving Teaching*. Incrementally, the book strives to gradually shift the focus from students, to different learning pedagogies, and finally to the instructor. At each step, helpful tips and ideas are presented along with graphs and charts which illustrate the relevant concepts. Good coverage is provided of the different computer science disciplines, and an effective balance is struck between theoretical and application-related approaches.

In the first section, *Improving Learning*, various authors offer coverage of what is required to improve computer science pedagogy. Here, the book attempts to address the learning side of the pedagogical models through three chief areas of study: *Text Comprehension in Computer Science Education*, *Learning About Spreadsheets*, and *Personalizing Learning About Databases.* Each area examines the supporting research and associated learning approaches with a strong focus on effective teaching. Through this approach, doctrinal research findings are presented along with specific recommendations for improving student learning.

Surprising – for some readers – will be the book's inclusion of *Text Comprehension in Computer Science Education*; this shock coming only because so many educational texts ignore it. In a technology-agog society, frequent omission of this most basic teaching tool is not unusual. In the book, text comprehension is not only covered, but effectively analyzed for learning relevance. Particularly insightful is the excellent coverage of the role of high versus low cohesion texts, and how they interrelate with high versus low knowledge students, respectively. Also interesting is the summary of the educational implications of ALMA (Adaptive Learning Models from texts and Activities) and its ability to tailor material to the student's relevant knowledge level.

Learning About Spreadsheets is noteworthy not only for its content, but once again, for its very inclusion. Its presence in a computer science book is revelatory, and a welcome change from material that simply ignores this significant component of computing. As proof, the book offers a 2009 study by Deloitte that found 70% of companies relying on spreadsheets for some critical business computations. Even for veteran practitioners, the content presented on different types of spreadsheet errors and their frequency is highly practicable. Its clear delineation of the strategies used to detect and prevent them includes: 1) having a separate input section, 2) barring hardcoded numbers from any formulas, 3) maintaining consistent row and column formulas, and 4) breaking complex formulas into a series of intermediate steps. Learning these, the students cannot avert all spreadsheet errors, but a significant step is taken towards prevention.

In *Personalizing Learning About Databases*, the concept of the discipline being divided into base hard- and soft-ware relevant knowledge, and the complementary procedural steps required to maximize their use, is nicely balanced. The use of modeling and abstraction, to instill in the learner an awareness of both, is covered at length. Other inclusions involve the use of predicative and functional structures to teach databases, and the tailoring of each to engage the learner. Generally, learners who respond best to networks of relations and structures prefer the former; while those who relate to a more functional model opt for the latter. This adjustment of the teaching material to the needs of the specific student; through either presentation of both to facilitate choice, or through initial assessment to customize a specific delivery, is a consistent theme in the book. Also presented is a competence matrix for database education that allows the instructor – and learner – to assess where the current skillset level resides, and what tasks can consolidate existing skills. This type of concrete tool is useful and a key component to the tailoring of database education to a specific student.

The second main component of the book, *Methodological Perspectives*, addresses the aspects of how to improve teaching and learning in computer science. The three areas of coverage are *Programming Visualization*, *Unplugging Computer Science*, and *Assessment of Students' Programs*. Each provides guidance and supplies relevant research in the field. The text's aim is to answer the question, how might computer science students best acquire new skillsets? – especially when coupled to specific information technologies.

Programming Visualization covers a wide array of topics related to using visual tools to promote enhanced understanding. These tools serve students by attempting to create representations that experientially relate to the student's background, thus enhancing retention and prospective use. Incorporating some of the history of these tools, the text includes an early 1980s program called Prolog that inaugurated some of these concepts. Prolog allowed students, in lieu of giving specific computer commands, to visually define the required relationship between the input and output. This had the obvious advantage of muting the student's needs for specific computational processes; however this was frequently achieved at the expense of deeper algorithmic understanding. While not ideal, these tools, and others like them (Scratch, Jeliot, Java, etc.) could establish a firm grounding in basic programming. The tension, as aptly illustrated in the chapter, is the balance between deeper immersion in programming methodology, and the simpler visual tools that can accelerate learning. Ultimately, the research correctly points to the importance of the teacher in the process, which regulates and represents an important complement to the overall pedagogy.

Unplugging Computer Science is a playful respite away from the incorrect assumption that technology teaching has to involve computers. Often, instructors forget that many of the concepts required of computer students really involve models and examples available in the physical world. A great example is provided through the use of playing cards to teach binary systems. Illustrating an example of Vygotsky's zone of proximal development, previous student experience with playing cards is used to enhance experiential learning. In addition to binary numbers, more complex concepts such as even/odd parity bit checking are demonstrated under the auspices of a simple magic trick. All of these greatly increase student enjoyment and decontextualize computer science so the core concepts can be revealed. The book correctly identifies that in younger students who are easily distracted by the very presence of the computer, this approach can yield much higher productivity. Using weights and scales to sort and organize data, students can also be introduced to ordinals and factorial concepts in a non-intrusive manner. Perhaps most useful is the chapter's link to the "unplugged website" that hosts many of these creative applications and the learning pedagogies they support.

Assessment of Students' Programs examines the ever-difficult development of grading heuristics in programming assignments. The two main methods presented are the automatic and manual methods. The automatic method uses a computer program to gauge the effectiveness and efficiency of the student's programming assignment. This assessment is then used by the instructor to assign a grade. Manual has the instructor personally evaluating the program and then judgmentally choosing to rely on an external program – or not – to assign a grade. The book elaborates well the pros and cons of each approach. The propagation and use of effective grading rubrics, when evaluating work, is constantly communicated with an emphasis on core integrated principles. The foci for student programming rubrics is placed on understandability, identification of incorrect solutions, identification of specific errors, demonstration of program flexibility, and tolerance for minor formatting errors. Collectively, these are presented as an effective foundation for quality grading and most importantly, student improvement.

Improving Teaching represents the final section of the book and perhaps its most important. In it, each of the topics presented, *Traditions in School Computing Education*, *Applying Standards to Computer Science Education*, and *Teaching Spreadsheets through a TPCK* (Technological Pedagogical Content Knowledge) *Perspective*, illustrate the best practices available to instructors. Based on research, they help the reader improve their understanding, planning, and management of computer science education. Throughout, the emphasis is on supplying the instructor with the necessary background and examples to effectively teach concepts and applicable skills.

In *Traditions in School Computing Education*, the historical basis for much of modern technology teaching is provided. From its roots in mathematics, to its splintering into sub-branches (computer, software, and electrical engineering), the profession's growth and scaling into new areas is demonstrated. Also included is the scientific community's sometimes hesitant admission of technology as an equal in scientific education. Although it dates back to the 1940s, distinction as a separate scientific discipline only occurred in the 1970s. The book effectively shows how this delay reflected as much a wariness of what the National Science Foundation thought of computers, as it did inherent dissonance about their importance. This origin is then used to construct how education in the discipline has been stunted by less pedagogical research and subsequent application of findings.

To remediate this, new approaches such as problem-based learning and progressive inquiry are provided to illustrate new possibilities. Effective hybrid approaches based on flexible project-based learning, and contextually-relevant pedagogies which utilize constructionism, are also presented. The overarching message of this chapter is that the availability of these pedagogies – especially when contrasted with the profession's early history – reflects great change and opportunity. Prospectively, this opportunity represents the unprecedented incorporation of quality research findings into daily practice.

Applying Standards to Computer Science Education involves the potential application of formalized standards within the profession. The development of achievement matrices is promoted with the intent of guiding teachers and students towards more comprehensive goal setting. Much of the chapter is dedicated to an examination of Germany's application of standards through its GI (*Gesellschaft fur Informatick*) standards. In them, explicit content and process expectations were promulgated. Developed collectively, many instructors were canvassed on how to create the resulting formal documentation. Unsurprisingly, the authors stress that while general expectations were easy to develop, explicit conclusions for defining the actual content proved more elusive. Overall, promotion of professional standards is presented in this chapter as capable of elevating the quality of teaching, and thus improving the definition of the key success criteria. Whether formal standards contribute to this success, and how to objectively measure them, remains the crux of problem.

The last section, *Teaching Spreadsheets through a TPCK Perspective* presents empirical study findings that were intended to instruct pre-service teachers on how to optimize Excel use. The intent was to help them integrate spreadsheet concepts into a deeper and more comprehensive immersion into the content. This integration was facilitated by the now familiar, TPCK learning architecture; where teaching domains are divided into five categories: 1) pedagogy, 2) content, 3) context, 4) learners, and lastly, 5) information and communication technology. In all categories, what is most relevant is the degree of domain knowledge the instructor brings into learning environment prior to student engagement.

The empirical findings of the study were quite interesting. In it, students were asked to design and develop a lesson comprised of five Excel learning activities. The activities were delineated to use Excel as a tool to develop useful models, feedback, calculations, data organization, and creative hypertext stories. Despite the relative inexperience of the students, their performances on all five tasks proved exceptionally high. Students reported that the contextualization of the spreadsheet concepts made them easier to learn, and therefore superior to simply presenting technical features. In this, the chapter strongly stressed the role of technology mapping to help design assignments that holistically engaged students. For instructors, this mapping can increase complexity because it involves not only software-related Excel topics such as navigation, input, formulas, and reports, but also cognitively-challenging problems like those found in the study. However, this fuller utilization of the teachers' TPCK domains and the students' experiential engagements with the content, highlights the key takeaway for readers in the book's final chapter; specifically, that greater student engagement leads to superior results.

In conclusion, *Improving Computer Science Education* attempts to amalgamate various authors' ideas across nine chapters to give educators and educational researchers improved pedagogical insights. The emphasis on learner-focused heuristics is an oft-repeated, and admirable theme throughout the book. Various viewpoints echoed this and are well worth the reader's attention. The book's strength lies in its ability to amass these varied viewpoints while delivering a cogent view of the research. While the volume is fairly successful in this endeavor, "common voice" is difficult to achieve across fourteen authors and nine different topics. The two chapters on spreadsheets did not contradict each other, but seemed to beg for incorporation into one. At times, the text becomes very dense and requires a strong background in educational theory and research history in order to decipher the complex findings. That said, the book is successful in its core goal: to elevate the level of understanding in the field of computer science pedagogy. Whether an instructor or a researcher approaches the book from an augmentative perspective, or as an introductory text for new concepts, the book serves both functions admirably. As new research on technology-related pedagogies continues to evolve, additional entries from these and other authors will strengthen this emerging canon.

About the Reviewer

Jose Victor Lineros
University of North Texas
United States
jose.lineros@unt.edu

education review // reseñas educativas
a multi-lingual journal of book reviews

editors: gustavo e. fischman / melissa cast-brede / gene v glass

Supported by the Mary Lou Fulton Teachers College, Arizona State University

Education Review/Reseñas Educativas/Resenhas Educativas is supported by the edXchange initiative's Scholarly Communicatiosn Group at the Mary Lou Fulton Teachers College, Arizona State University. Copyright is retained by the first or sole author, who grants right of first publication to the *Education Review*. Readers are free to copy, display, and distribute this article, as long as the work is attributed to the author(s) and ***Education Review*****,** it is distributed for non-commercial purposes only, and no alteration or transformation is made in the work. More details of this Creative Commons license are available at http://creativecommons.org/licenses/by-nc-sa/3.0/. All other uses must be approved by the author(s) or ***Education Review***. ***Education Review*** is published by the Scholarly Communications Group of the Mary Lou Fulton Teachers College, Arizona State University.

Please contribute reviews at http://www.edrev.info/contribute.html.

Connect with *Education Review* on Facebook (https://www.facebook.com/pages/Education-Review/178358222192644) and on Twitter @EducReview