

The Marriage of Research and Practice

Warren Harrison

When the IEEE Computer Society announced my appointment as the new editor in chief of *IEEE Software* last year, I received congratulations from many of my friends and colleagues in both academia and industry. Not surprisingly, many academicians urged me to refocus the magazine to publish papers with more research emphasis. However, I have a duty to consider this advice both within the context of *IEEE Software*'s mission to "build the community of leading software practitioners" (emphasis mine) and the differences between a magazine (which is what *IEEE Software* is) and a transaction (which *IEEE Software* is not).



Last fall, my predecessor, Steve McConnell, published a *From the Editor* column entitled "How to Write a Good Technical Article" (Sept./Oct. 2002), in which he described the difference between a magazine and a transaction. In short, a transaction is expected to publish archival research results whereas a magazine publishes articles (and columns) of contemporary interest to practitioners. According to *Webster's Revised Unabridged Dictionary*, a practitioner is "one who is engaged in the actual use or exercise of any art or profession," while research is "diligent inquiry or examination in seeking facts or principles."

Universal appeal

I honestly do not believe that "research" and "articles of contemporary interest to practitioners" are mutually exclusive, but

neither are they synonymous. To paraphrase Abraham Lincoln:

Some types of research are of interest to all Software readers, and all types of research are of interest to some Software readers, but not all types of research are of interest to all Software readers.

The key to success for *IEEE Software*, or any other publication trying to disseminate research results to practitioners, is to identify research that is of interest to "all *Software* readers."

What research might have such universal appeal? I first pondered this question in 1987 when, as a brand new PhD, I became frustrated with the lack of acceptance of software measurement in industry. To better understand industry's needs as well as gain credibility for my ideas, I helped launch a small start-up called SET Laboratories that produced a popular software metrics analyzer. I was fortunate to have an opportunity to play every role in the company at one point or another, from programmer to tester to help desk operator.

Obviously, to capture our market, we had to understand our users and their needs very well. For the next five years, I undertook an immersion course in the needs of software practitioners—aided by guides such as Steve Wilkinson of Ashton-Tate, Scott Cherf of Tandem, George Stark of MITRE, Mike Perdue and Bob Birss of Sun Microsystems, Linda Rosenberg of NASA, and Carl Seddio of Kodak, to name just a few. These individuals had a huge influence on my current views and perspectives. During this time, I think I began to understand how research could help the prac-

DEPARTMENT EDITORS

Bookshelf: Warren Keuffel, wkeuffel@computer.org

Construction: Andy Hunt and Dave Thomas, Pragmatic Programmers, (andy, dave)@pragmaticprogrammer.com

Design: Martin Fowler, ThoughtWorks, fowler@acm.org

Loyal Opposition: Robert Glass, Computing Trends, rglass@indiana.edu

Manager: Don Reifer, Reifer Consultants, d.reifer@ieee.org

Quality Time: Nancy Eickelmann, Motorola, nancy.eickelmann@motorola.com, and Jane Huffman Hayes, Univ. of Kentucky, hayes@cs.uky.edu

Requirements: Suzanne Robertson, Atlantic Systems Guild, suzanne@systemsguild.com

STAFF

Senior Lead Editor

Dale C. Strok
dstrok@computer.org

Group Managing Editor
Crystal Chweh

Associate Editors
Shani Murray and Dennis Taylor

Assistant Editor
Denise Kano

Editorial Assistants
Rebecca Deuel and Joan Hong

Magazine Assistant
Pauline Hosillos, software@computer.org

Art Director Cover Illustration
Toni Van Buskirk **Dirk Hagner**

Technical Illustrator Production Assistant
Alex Torres **Monette Velasco**

Production Artists
Carmen Flores-Garvey and Larry Bauer

Executive Director
David Hennage

Publisher
Angela Burgess

Assistant Publisher
Dick Price

Membership/Circulation Marketing Manager
Georgann Carter

Advertising Assistant
Debbie Sims

CONTRIBUTING EDITORS

Candace English, Anne Lear, Christine Miller, Keri Schreiner, and Margaret Weatherford

Editorial: All submissions are subject to editing for clarity, style, and space. Unless otherwise stated, bylined articles and departments, as well as product and service descriptions, reflect the author's or firm's opinion. Inclusion in *IEEE Software* does not necessarily constitute endorsement by the IEEE or the IEEE Computer Society.

To Submit: Access the IEEE Computer Society's Web-based system, Manuscript Central, at <http://cs-ieee.manuscriptcentral.com/index.html>. Be sure to select the right manuscript type when submitting. Articles must be original and not exceed 5,400 words including figures and tables, which count for 200 words each.

itioner as well as why the work I had been doing was not.

Communicate, communicate

The first great truth I realized while sitting at the feet of these masters is that they were getting along quite well without my help, thank you. In other words, as the researcher, it was my job to communicate the significance of my research with respect to their jobs, which was to develop software. Software practitioners are extremely busy people, and they simply have neither the time nor motivation to figure out on their own how someone's research can apply to them.

Having served as editor in chief of two research-oriented journals before joining *IEEE Software*, I think I see this mistake more often than almost any other. This is especially true in empirical studies. For instance, you might find a piece of research that proudly proclaims that, after a rigorous statistical evaluation of a dozen companies (a statistical evaluation that you have been forced to wade through to get to the punch line, by the way), programmers prefer garlic two-to-one over pepperoni on their pizza. While perhaps interesting, this research has just put the burden of figuring out what to do with this valuable nugget of wisdom on the practitioner's shoulders.

The practitioner, who is doing well to even find the time to read the article, is not only *not* going to reduce the results to practice, but will likely make a mental note to stay away from papers in the future with titles like "A Four-factor, Log-linear Correlational Analysis of Patterns of Pizza Consumption among Multinational Software Developers." Rather, if the research results are significant to software practitioners, it is the researcher's job to explain how, and if they aren't, most practitioners really don't have the time to hear about them. A better approach would be for the researcher to propose specific actions that practitioners can take now that we know programmers like garlic—such as reducing the number of two-person cubicles—and use the study's results to substantiate the proposal.

No half-baked ideas, please

Another common mistake made by researchers trying to write for practitioners is the lack of conclusive results. I often come across research summaries that end with the phrase "... and of course these results must be confirmed through future studies, but we feel the approach has great promise ...". Such papers are great fun in a research-oriented conference. It is the academic equivalent of saying, "Come on in, the water's fine." However, to the software practitioner, it gives the impression of a half-baked idea.

Practitioners read publications like *IEEE Software* to learn about techniques they can use to make their jobs easier. Only extraordinarily trusting individuals are interested in adopting methods or techniques that still "... must be confirmed through future studies," no matter how promising they appear. As Frank Bartels and Ed Jaymes used to say, "We will serve no wine before it's time."

When researchers decide it is time to expose their ideas to practitioners, at least they should be truly confident in their idea. Tentative results belong in conferences, not a practitioner-oriented magazine. Only after additional studies have confirmed the hypothesis are the results really useful to practitioners.

Simply being confident in an idea isn't in itself adequate to convince a practitioner to adopt—or even care about—a technique, tool, or other piece of research, although it certainly helps. Practitioners also want to see easily understandable evidence that the approach being described is better than the alternatives. This might entail comparative case studies in which a technique is applied in practice and then compared to earlier projects where the technique was not used. Note the emphasis on "easily understandable." You should not need a statistics book to be able to tell if the two outcomes differ significantly—if you do, the benefits are probably far too subtle to be truly useful to the practitioner. If the evidence is not easily understandable, the researcher's first job is to work on a way to make it so.

Address the practitioner's reality

There is always the researcher who tries to morph reality to fit his or her study. One of my favorites is the argument that studies of student projects generalize to industrial projects because student projects are made up of code and industrial projects are made up of lots of code. Using student projects to test out ideas is fine, and if you discover universal truth, it might be generalizable. However, before trying to publish such work for practitioners, you should refer back to the paragraph that starts with "Another common mistake"

How do I actually use this?

Finally, there is the issue of operationalization. If a technique is going to actually be useful to practitioners, it must ultimately be reduced to practice. Although page limitations might prevent enough detail from being published in the article itself, if a researcher wants to attract interest from practitioners, detailed instructions on how the technique could be reduced to practice must be available somewhere.

Maurice Halstead's *Elements of Software Science* and the work that surrounded it in the 1970s and 1980s should be a study in the pitfalls of de-

finitions and operationalizations. Although Halstead published good rules for applying *Software Science* to Fortran, later attempts by others to extend the technique to languages such as C and C++ resulted in a montage of studies, each of which applied its own definition of the *Software Science* rules (for instance, is a function call an operator or an operand?). Many practitioners threw up their hands in disgust while researchers squabbled among themselves. The World Wide Web has provided easy ways to augment research reports by making tools, manuals, and so on available to readers who wish to explore the work further.

We welcome your feedback

In short, practitioners can be interested in research when it has clear applications to their work, has well-founded, understandable credibility, and can be operationalized. If you are a researcher and can write this kind of article, I enthusiastically invite you to submit it to *IEEE Software*. If you want to bounce ideas off me or have your own thoughts about the problem of coupling research and practice, please write to me at warren.harrison@computer.org. ☺

New Editors

After over three years editing the Quality Time column, Jeff Voas has passed the baton. Serving as both an associate editor in chief and a column editor required a huge amount of time and energy, and we are grateful for his contribution. Not to worry, Jeff will remain an AEIC for *Software*. We now have a pair of new Quality Time editors to introduce this issue.

Nancy Eickelmann is a Distinguished Member of the Technical Staff at Motorola Labs. She is a Six Sigma Black Belt and leader of three research initiatives: the Motorola software and system test process modeling and simulation research initiative, the Motorola ODC initiative, and a four-year NASA initiative for risk-based analysis to minimize the cost of poor quality. She received her PhD and MS in computer science from the University of California, Irvine, and an MBA in information and decision sciences and a BS in finance from San Diego State University.

Jane Huffman Hayes is an assistant professor of software engineering in the Computer Science Department of the University of Kentucky. Previously, she was a corporate vice president and operation manager for Science Applications International Corp., where she managed a 230-person, US\$35 million business unit. She received her MS in computer science from the University of Southern Mississippi and her PhD in information technology from George Mason University.

EDITOR IN CHIEF:

Warren Harrison

10662 Los Vaqueros Circle
Los Alamitos, CA 90720-1314
warren.harrison@computer.org

EDITOR IN CHIEF EMERITUS:

Steve McConnell, Construx Software

ASSOCIATE EDITORS IN CHIEF

Design: Maarten Boasson, Quaerendo Invenietis
boasson@quaerendo.com
Construction: Terry Bollinger, Mitre Corp.
terry@mitre.org
Requirements: Christof Ebert, Alcatel
christof.ebert@alcatel.com
Management: Ann Miller, University of Missouri, Rolla
millera@ece.UMR.edu
Quality: Jeffrey Voas, Cigital
jvoas@cigital.com
Experience Reports: Wolfgang Strigel,
Software Productivity Center; strigel@spc.ca

EDITORIAL BOARD

Don Bagert, Rose-Hulman Institute of Technology
Nancy Eickelmann, Motorola
Richard Fairley, Oregon Graduate Institute
Martin Fowler, ThoughtWorks
Robert Glass, Computing Trends
Jane Huffman Hayes, University of Kentucky
Andy Hunt, Pragmatic Programmers
Warren Keuffel, independent consultant
Brian Lawrence, Coyote Valley Software
Karen Mackey, Cisco Systems
Deependra Moitra, Lucent Technologies, India
Don Reifer, Reifer Consultants
Suzanne Robertson, Atlantic Systems Guild
Dave Thomas, Pragmatic Programmers

INDUSTRY ADVISORY BOARD

Robert Cochran, Catalyst Software (chair)
Dave Aucsmith, Microsoft
Annie Kuntzmann-Combelles, Q-Labs
David Dorenbos, Motorola Labs
Enrique Draier, PSINet
Eric Horvitz, Microsoft Research
David Hsiao, Cisco Systems
Takaya Ishida, Mitsubishi Electric Corp.
Dehua Ju, ASTI Shanghai
Donna Kaspersen, Science Applications International
Pavle Knaflic, Hermes SoftLab
Wojtek Kozaczynski, Rational Software Corp.
Tomoo Matsubara, Matsubara Consulting
Masao Matsumoto, Univ. of Tsukuba
Dorothy McKinney, Lockheed Martin Space Systems
Stephen Mellor, Project Technology
Susan Mickel, AgileTV
Dave Moore, Vulcan Northwest
Melissa Murphy, Sandia National Laboratories
Kiyoh Nakamura, Fujitsu
Grant Rule, Software Measurement Services
Girish Seshagiri, Advanced Information Services
Chandra Shekaran, Microsoft
Martyn Thomas, Praxis
Rob Thomsett, The Thomsett Company
John Vu, The Boeing Company
Simon Wright, Integrated Chipware
Tsuneo Yamaura, Hitachi Software Engineering

MAGAZINE OPERATIONS COMMITTEE

Jean Bacon (chair), Thomas J. Bergin, Pradip Bose, Doris L. Carver, George Cybenko, John C. Dill, Frank E. Ferrante, Robert Fillman, Forouzan Golshani, Rajesh Gupta, Warren Harrison, Mahadev Satyanarayanan, Nigel Shadbolt, Francis Sullivan

PUBLICATIONS BOARD

Ranga Kasturi (chair), Jean Bacon, Laxmi Bhuyan, Mark Christensen, Thomas Keefe, Steven L. Tanimoto, Anand Tripathi