I enjoyed Robert Glass’ column “A Sad SAC Story about the State of the Practice” (July/Aug. 2005). You’re right on the mark with your complaint about “practice” journals and conferences. Let’s face it: The academic community needs to write papers, and they need venues to get the stuff published. So they created conferences and journals. ICSE is an academia fest, created by academics for academics. As a side benefit, they get to travel to exotic places. Fundamentally, there’s nothing wrong with that, except that they try to label these venues as covering practice. That’s just political window dressing and marketing.

In case you didn’t know, there’s a little gem of a conference: PNSQC (Pacific Northwest Software Quality Conference). It’s an annual event in Portland, Oregon, organized by a relentless group of volunteers. Attendance is 200 to 300 practitioners, and presentations are almost exclusively about real industry experience. Although many papers wouldn’t get published in IEEE Software since they don’t present earth-shattering new findings, they do tell the audience about experience from the trenches, what worked, and what didn’t. The review process is pretty relaxed, but again, it’s done by people from industry (including me). Check it out at www.pnsqc.org.

Wolfgang Strigel
President, QA Labs
strigel@qalabs.com

I enjoyed your column about the state of the practice. The amount of practical content in software magazines and books is poor from the practitioner’s perspective. One of the reasons I like IEEE Software is that I can, at least, read some interesting letters and columns from people who develop software for a living.

Like any financial advisor would say, you need to diversify your portfolio. I use the following mix of conferences to get a real sense of what’s going on in the real world:

■ Academic conferences are good for understanding research trends without any particular product in mind. You can engage with any leading expert in the field to get to the bottom of a particular problem.

■ Developer conferences are very useful to sense the level of adoption in the field. They come in two flavors: products and practices. In terms of products, every major software company holds an annual developers conference, such as OracleWorld, Sun’s JavaOne, and the IBM Technical Developer Conference. Besides the new product announcements, it’s interesting to see demos and talk with small companies that are building new software on top of a particular vendor platform. In terms of practices, the SD Best Practices Conference and Expo tends to have very good keynotes with lots of experience in the field.

■ Last but not least, user group meetings are formed around particular vendors or products. These are usually local events put together by a lot of volunteers. The good thing is that you meet the people in charge of live systems.

I’ve found more interesting practical insights in company conferences and user groups. The
As someone who’s recently “converted from practitioner to academic,” I agree with your view of the problem of learning about the state of the practice. Having done some consulting, I would say the gap between what’s done on average and what’s done by the top people is so huge that the average can’t even see across the chasm. However, I work on it, so that my students will help bridge it.

Nevertheless, there are some places where software practitioners meet. One large event to consider is ACM OOPSLA (Object-Oriented Planning, Systems, Languages, and Applications), even though the technical program is mostly academic.

If you’re more interested in learning from practitioners and from those who care to share their experience, try one of the PLoP conferences (Pattern Languages of Programming, www.hillside.net). I think the pattern community is one of the places where typical academia isn’t in the driving seat. Since patterns are always about old, proven stuff that has worked several times successfully, you wouldn’t find many inexperienced people around proclaiming they have invented a better wheel.

If you look over the Atlantic to Britain, two spring conferences are supported by and targeted at practitioners: SPA (Software Practice Advancement, formerly OT—Object Technology) and the ACCU Conference (Association of C and C++ Users; it’s not only about those languages, but the ACCU runs it). It would be great to meet you at one of these conferences.

Robert Glass’ column in the July/Aug. issue ends with the words “and no mechanism on the horizon to change that serious problem.”

In the article, you mainly blame the academics and seem to forget about the practitioners. Only they can report the state of the practice, but they usually don’t.

Omar Alonso
Senior product manager,
Oracle Corp.
oralonso@yahoo.com

Your column in the July/Aug. issue spoke right to my heart. I consider myself a software practitioner, but I’m familiar with the academic world too (although not in software). The reason you read more from theorists than from practitioners is quite simple: in the academic world, writing is as important as research. In the real world, delivering your product is probably sufficient.

Even if I often find interesting and stimulating articles in Computer and IEEE Software, they don’t really answer my needs. There’s a gap between popular magazines and academic papers that isn’t filled, at least not in Germany. But maybe we practitioners are to blame. Isn’t it up to us to publish what we think is important or helpful to our colleagues? Why not give us a platform in our own magazines? I hope your column will inspire both editors and writers.

Thomas Schäfer
Senior consultant, Siemens AG
thomas.schaefer@ieee.org

LETTERS

New from
The MIT Press

The
Reasoned Schemer
Daniel P. Friedman, William E. Byrd, and Oleg Kiselyov
Extends the functional language Scheme with logical constructs in order to help the functional programmer think logically and the logic programmer think functionally.
200 pp. $27.50 paper

Java Precisely
Second Edition
Peter Sestoft
A quick reference guide to Java 5.0 for students and Java programmers.
168 pp. $18.95 paper

Software Ecosystem
Understanding an Indispensable Technology and Industry
David G. Messerschmitt and Clemens Szyperski
“An encyclopedic and cutting-edge book. Bridging the technological and business spheres, it will be a terrific resource for anyone working in or studying the software industry.”
— Richard N. Langlois, University of Connecticut
432 pp., 49 illus. $22 paper

The MIT Press
http://mitpress.mit.edu
There seem to be several reasons for this: practitioners are busy getting their development done; there are usually no incentives in companies for publishing some paper—their job is development, not publishing; and, most important, it seems to be better to keep the details of their software products and their development secret. As I wrote in my 2000 report Characteristics of Computing and Informatics, “Computer programs are best protected by keeping the source program secret and leasing object programs.” I characterize this as “invisibility of software.”

By pure chance, Angela Jury mentions this problem on page 117, just three pages before your column, in her review of Code Reading: The Open Source Perspective. “The book also points out something obvious that I had missed. Until recently, it was difficult to find nontrivial code systems to read because they were all proprietary.”

It seems to me that software is different from other engineering disciplines’ more visible designs. You can easily read an architect’s design from the finished building, or you can buy a new car and analyze it.

Nevertheless, you could, in a future column, appeal to practitioners to tell their poor academic colleagues some facts about the state of the practice.

Jürgen Winkler
Professor, Institut für Informatik, Friedrich-Schiller-Universität
jwinkler@acm.org

It’s always interesting to read what Robert Glass writes, especially on the subject of practitioners versus academics. As a practitioner, I appreciate the distinction he makes between prescribing and describing. However, I don’t think the state of the practice is as sad as he describes in his latest column.

First of all, I think he’s too quick to dismiss books as a source of the state of the practice. Of course, it depends on what books you read, but virtually all books I read on software development are written by practitioners who are developing production software—for example, Joel on Software by Joel Spolsky (Apress, 2004), which is full of real-life examples; Lean Software Development by Mary and Tom Poppendieck (Addison-Wesley Professional, 2003), which is about their experience in using agile development approaches; or The Pragmatic Programmer (Addison-Wesley Professional, 1999) by Andy Hunt and Dave Thomas. All of these are fairly recent, as opposed to the 30-year-old The Mythical Man-Month (Addison-Wesley, 1975) that Glass mentions, and highly readable.

Second, he doesn’t even consider blogs. This is where I think you really get an up-to-date state of the practice. People who blog about software development often use their own experiences as a base, so you can get a good overview of what people actually do. As with books, magazines, and conferences, there are good ones and bad ones. A few good ones that I read regularly are Martin Fowler’s (www.martinfowler.com/bliki), Joel Spolsky’s (www.joelonsoftware.com; the book by Joel that I mentioned is a collection of essays from his Web site), and a whole host of blogs at www.artima.com.

So, I do think we can determine the state of the practice, but we have to look in the right places.

Henrik Warne
Senior system developer, i3 micro technology
henrik.warne@i3micro.com

Robert Glass responds:
I didn’t mean to say that it’s the state of the practice that’s sad—it’s the state of the description of the practice that’s sad. And I appreciate your suggestions of additional places to look.

I’m a technical lead for a corporation-wide project that tracks and reports the potential environmental impact of the company’s products. Before that, I was the technical lead for a division-level project for a factory management system that included a wide variety of technologies including machine controls, bar code technologies, database design, and user interfaces.

I think there are two reasons that academia has taken over control of the journals and conferences that you’d think would be more meaningful to practitioners.

The first reason is time. For the sake of argument, let’s assume that I’m typical. I work about 60 hours a week and have trouble getting everything done. I’m working on my MBA, so that takes another 10 to 15 hours a week. I’m also trying to keep up on the state of the art in software through reading a variety of magazines (including IEEE Software) and books, as well as learning new languages and technologies. To prevent my mind from getting stale, I also try to read non-technical material and magazines. And of course, my wife occasionally wants to know that I’m still alive and for me to spend some time with her. Fortunately(?), I don’t have children, so I only have general household chores to worry about.

The second reason, related to the first reason, is priorities. To put it bluntly, the corporation doesn’t reward me for taking any kind of leadership role in either journals or technical conferences. Taking such a leadership role isn’t always a priority due to work demands. There’s a research organization within the corporation, and they’re rewarded for that type of activity, but in many ways, they’re more like an academic organization than a pure industrial organization. I’ve presented some technical papers, both internally and externally, and received little recognition from management for them.

Based on that, I have to use my time where it will have the most impact, both personally and professionally. For my own personal satisfaction, I’d like to get more involved in some industry- or technology-wide activities, but it’s not seen as a high priority in my current growth path. Unless I change my career objectives or it’s seen as more important to the corporation, it’s unlikely that I’ll be getting more involved in the immediate future.

Gerald Sprague
Sr. staff software engineer, Motorola Corporate IT, PRSS
gerald.sprague@motorola.com