IEEE SOFTWARE ACCEPTS less than 25 percent of the articles submitted for consideration, and I’m keenly aware that all of those submissions—whether eventually accepted or rejected—entail many hours of effort on the part of authors, reviewers, and magazine staff.

In addition to being selective, IEEE Software is also a somewhat unique venue. Our mandate is to be the authority on translating software theory into practice—meaning that while we’re interested in rigorous and well-tested research results, those results also need to be explained in a way that can reach our intended reader (the reflective software practitioner) and help her understand something important about the software profession. For this reason, we prioritize writing with an accessible style and relatively tight word limits. Good advice for juggling these constraints can be found in articles written by my predecessors as editor in chief, Steve McConnell and Hakan Erdogmus. To add to what they’ve written, in this article, I’d like to focus on a special type of submission: the experience report.

I’m an advocate of experience reports because I’m a firm believer in just about any approach that stands a chance of improving the communication between software research and practice. Here at IEEE Software, we receive fewer experience reports than other types of submissions, and this is understandable. I realize how tough it can be, as a professional developer, to get the time to reflect on an experience and write about it, and I appreciate those who do so. Although it can be difficult, the effort to produce an experience report is almost always rewarding and helps the author reflect on the true causes for success and failure amid all the noise and pressure of day-to-day deadlines. Well-written experience reports can be among the most compelling pieces that we publish in IEEE Software.

In this article, I’d like to take the time to reflect, myself, on what we are looking for in experience reports and provide some guidance that can help authors.

What Is It, and What Does It Do?

An increasing number of conferences and periodicals in software engineering are featuring experience reports. From a quick and admittedly subjective perusal of the author guidelines, however, the calls for experience reports often seem to suffer from the lack of a clear definition of exactly what is being sought. This is a danger because—especially in research-focused venues—without a clear definition, experience reports are often perceived as the place to send work that won’t be accepted in normal technical tracks.

But experience reports are an important type of article in their own right—not just technical pieces that didn’t quite make the bar. Experience reports should provide a benefit that more “traditional” research studies cannot: this is a bit of an oversimplification, but let’s call this benefit “depth”—that is, a more detailed and nuanced understanding of what happened in a single environment (or single project). Experience reports, in describing a single environment, can only describe what happened...
to the authors; they don’t provide sufficient data to argue that if other teams follow the same approach, they can confidently expect the same outcome. To make up for this lack, a good experience report provides enough of a narrative to discuss with confidence why a certain result was seen.

IEEE Software is interested in publishing experience reports for a number of reasons. In my mind, the most important is that they help keep research grounded. Our field has self-organized in such a way that many software researchers aren’t familiar with the contemporary experience of working in a software development environment, and sharing that vision can help keep research focused on compelling problems and help produce results capable of operating under reasonable constraints.

Software professionals can also benefit from hearing about what development is like in other contexts. None of us have the time or opportunity to experience all types of environments, and many of us can find some benefit in looking at practices in other types of organizations. A developer in Silicon Valley, for instance, might find some value in looking at practices in other types of environments, and many of us can find some benefit in looking at practices in other types of organizations. A developer in Silicon Valley, for instance, might find some value in looking at practices in other types of environments, and many of us can find some benefit in looking at practices in other types of organizations.

Other reasons for valuing experience reports are that they can often provide the most practical advice to practitioners. I often respond with more interest and curiosity to someone telling me a story (“Oh, look, someone who is doing similar work to mine swears by this tool—I think I’d better give it a closer
look”) than to reams of data (“This tool vendor claims to have reduced the amount of effort needed for the job and claims that 9 out of 10 customers are highly satisfied”). I assume that other humans are motivated in similar ways.

Finally, experience reports can provide fast feedback to the community on new technologies or approaches being advocated. Long before anyone can have enough data to start to consider statistically significant effects, we may be able to share success (or failure) stories from individual projects. These should be taken with the appropriate caveats, of course, just like any study. But if the results of the experience report are compelling, they can help readers understand whether this is an area worth expending time and effort on.

Why do I mention all of this? Mainly so that prospective authors can use this as fodder for their own article reviews prior to submission. The single most important thing that any author can do as part of a self-critique is to think of the reader. Will an experience report help a reader keep up with what she needs to know to be effective in the software profession?

Tell Me a Story
Potential authors who ask for feedback from me on abstracts of planned papers will almost always get a response structured around the following set of questions.

Environment
Is it clear what type of environment your story takes place in? Other readers would like to benefit from your insights, but they need to have a good sense of how likely your findings are to translate to their projects. If you’re building a website app and I’m building embedded satellite software, I might find your story thought-provoking, but I might approach the idea of applying the same techniques in my work more carefully.

Focus
Is it clear what you did? What method, tool, or practice did you apply? In short, what is your story about? If a reader finds your experience compelling and is willing to try it out in his own work, would he know what to do or where to get more info? Above all, keep focus. Don’t describe everything you did on the project. Be ruthless in down-selecting to just those facts that support the coherent story you’re trying to tell. When choosing the focus of that story, keep in mind that IEEE Software has a broad coverage area, and we’re interested in methods and tools related to the nuts and bolts of software development as well as management and human factors issues.

Results
What were the results of what you did—and how do you know that those results were caused by the method, tool, or practice you’re advocating? Our reviewers are looking for reports that describe a concrete result. If you’re telling me a story that revolves around applying a new approach (let’s say an automated tool that attempts to detect hidden technical debt items), you have to tell me the end of the story. How well did the tool work? Was the project a success—and was that success traceable back to the tool in any meaningful way?

When it comes to describing results, there are other issues to con-
sider. How do you know that your results really mean? And how would a reader have confidence that your story can be trusted? We don’t expect experience reports to have reams of hard, quantitative data, but there are other ways of addressing this issue. When appropriate, these might include subjective forms of evidence such as feedback from key stakeholders or management—in this case, the more specific the author can be, the more convincing the story tends to be. Direct quotes can be helpful in this regard. Comparison to prior projects is always useful, as a way to show what has changed as a result of the new approach. Often, what the author’s organization is willing to do on the basis of the results speaks volumes. If the results are convincing enough to impact day-to-day practices across other projects, then they’re probably compelling enough for readers to pay attention to.

Also, when describing results, authors shouldn’t claim to have found a silver bullet. Readers appreciate a careful weighing of pros and cons and it’s very rare indeed to be able to see progress on one dimension without trade-offs on others. Truly great experience reports are those that look at multiple types of impact—say, a tool’s impact on the eternal triangle of project cost, quality, and schedule. If a tool really helps improve the delivered quality of a product, what does a project have to give up for that result—a substantial amount of extra effort? An impact on the schedule? And how about one-time costs like investments in training?

Where to Go from Here

I welcome experience reports submitted through the usual channels. But if all of the above constraints seem daunting, don’t despair. Our Insights department, helmed by Linda Rising, was established especially to help—in fact, the one on page 9 of this issue focuses on stories. Proposals to Insights are reviewed by Linda and her distinguished advisory board and, if accepted, shepherding is provided. Please see Linda’s inaugural column for much more helpful information and guidance.

If I could boil all of this guidance down to a simple test, it would be this: Is there more to an article than just a description of, “we did this” or “we built this”? Is there a meaningful principle exemplified through an experience report that readers will care about, be intrigued by, and possibly think of applying themselves? Linda has compared a good experience report to a project retrospective: “We not only want teams to look back and say what happened, but we also want analysis.”

I couldn’t put it better than that. And, like Linda, I remain excited by the idea of hearing more reflection and analysis from the ambitious software development projects going on throughout the industry today—with results we can all learn from together.

References

FORREST SHULL is a division director at the Fraunhofer Center for Experimental Software Engineering in Maryland, a nonprofit research and tech transfer organization, where he leads the Measurement and Knowledge Management Division. He’s also an adjunct professor at the University of Maryland College Park and editor in chief of IEEE Software. Contact him at fshull@computer.org.