DOI:10.1145/2658987

Shriram Krishnamurthi and Jan Vitek

Viewpoint The Real Software Crisis: Repeatability as a Core Value

Sharing experiences running artifact evaluation committees for five major conferences.

WHERE IS THE software in programming language research? In our field, software artifacts play a central role: they are the embodiments of our ideas and contributions. Yet when we publish, we are evaluated on our ability to describe informally those artifacts in prose. Often, such prose gives only a partial, and sometimes overly rosy, view of the work. Especially so when the object of discourse is made up of tens of thousands of lines of code that interact in subtle ways with different parts of the software and hardware stack on which it is deployed. Over the years, our community's culture has evolved to value originality above everything else, and our main conferences and journals[a] deny software its rightful place.

Science advances faster when we can build on existing results, and when new ideas can easily be measured against the state of the art. This is exceedingly difficult in an environment that does not reward the production of reusable software artifacts. Our goal is to get to the point where any published idea that has been evaluated, measured, or benchmarked is accompanied by the artifact that embodies it. Just as formal results are increasingly expected to come with mechanized proofs, empirical results should come with code.

If a paper makes, or implies, claims that require software, those claims must be backed up.

Conversations about this topic inevitably get mired in discussions of *reproducibility*, which the act of creating a fresh system from first principles to duplicate an existing result under different experimental conditions. Reproducibility is an expensive undertaking and not something we are advocating. We are after repeatability, which is simply the act of checking the claims made in the paper, usually, but not only, by re-running a bundled software artifact. Repeatability is an inexpensive and easy test of a paper's artifacts, and clarifies the scientific contribution of the paper. We believe repeatability should become a standard feature of the dissemination of research results. Of course, not all results are repeatable, but many are.

Researchers cannot be expected to develop industrial-quality software. There will always be a difference between research prototypes and production software. It is therefore important to set the right standard. We argue the right measure is not some absolute notion of quality, but rather how the artifact stacks up against the expectations set by the paper. Also, clearly, not all papers need artifacts. Even in software conferences, some papers contain valuable theoretical results or profound observations that do not lend themselves to artifacts. These papers should, of course, remain welcome. But if a paper makes, or implies, claims that require software, those claims should be backed up. In short, a paper should not mislead readers: if an idea has not been evaluated this should be made clear, both so program committees can judge the paper on its actual merits, and to allow subsequent authors to get the credit of performing a rigorous empirical evaluation of the paper's ideas. Lastly, artifacts can include data sets, proofs and any other by-product of the research process.

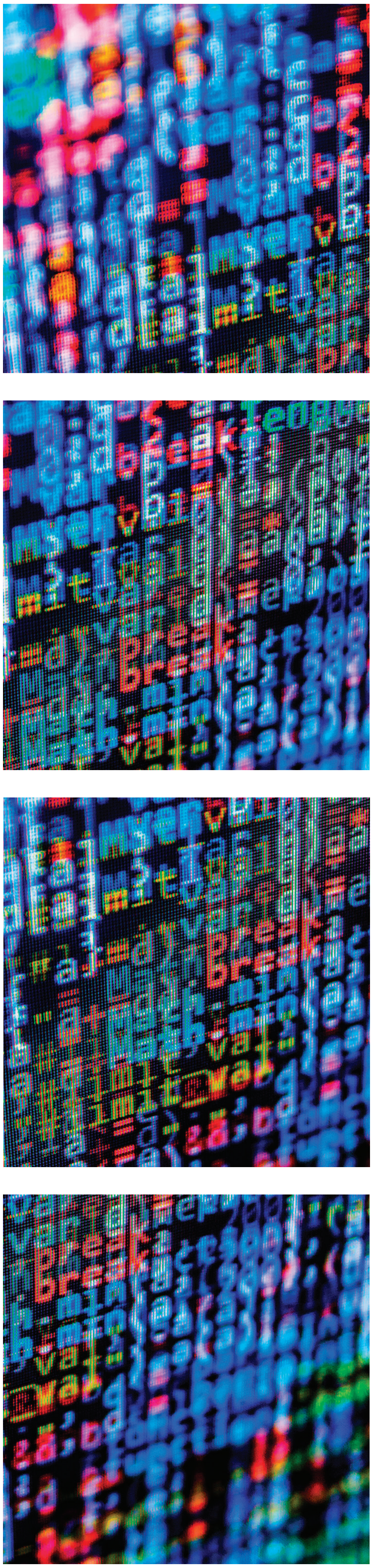
a Our central argument applies just as well, and perhaps even more strongly, to journals. However, we do not have experience creating an artifact evaluation process for journals; we also imagine that some journals might be concerned that their submission rate is sufficiently low that further obstacles would be unwelcome, though this is a weak argument for not performing a more thorough review.

The artifact evaluation process. Several ACM SIGPLAN conferences (OOPSLA, PLDI, and POPL) and closely related conferences (SAS, ECOOP, and ESEC/FSE) have begun an experiment intended to move in the direction outlined here. They have initiated an artifact evaluation process that allows authors of accepted papers to submit software as well as many kinds of non-software entities (such as data sets, test suites, and models) that might back up their results.[b] Since 2011 we have run, or helped with, six artifact evaluation committees (AECs). The results so far are encouraging. In 2011, the ESEC/FSE conference had 14 artifact submissions (for 34 accepted papers) and seven of those met or exceeded expectations. In 2013, at ECOOP, nine out of 13 artifacts were found to meet expectations. The same year, ESEC/FSE saw a big jump in artifact submission with 22 artifacts, of which 12 were validated. At SAS, 11 out of 23 papers had artifacts. The 2014 OOPSLA conference had 21 artifacts out of 50 accepted papers, and all but three were judged adequate. In 2014, all the preceding conferences had an artifact evaluation process.

What are the mechanics of artifact evaluation? The design of the first artifact evaluation process (conducted by the first author with Carlo Ghezzi[c]) involved discussions with leaders of the software engineering community, and met with more resistance than expected. There was concern that introducing artifact evaluation into the decision-making process would be an abrupt and significant cultural change. As a result, we erected a strict separation between paper acceptance and artifact evaluation in the simplest possible way: using a temporal barrier. Only accepted papers could be submitted for evaluation and their acceptance status was guaranteed to remain unchanged.

b For pragmatic and social reasons, artifact evaluation is limited to accepted papers. Integrating artifact evaluation with paper reviewing was felt to be risky, as the standards of what constitutes a valid artifact are still evolving. From a practical perspective, the effort of evaluating a large number of artifacts would overwhelm the committee. On average, an artifact takes a day and a half to evaluate by each of the three evaluators. The process would be difficult to scale to hundreds of submissions.

c http://cs.brown.edu/~sk/Memos/Conference-Artifact-Evaluation/



This was a necessary compromise to get the process approved at all. In time, it is conceivable that artifact evaluation will become a part of the evaluation of most scientific results.

Initially, we judged artifacts on a five-point scale, with crisp, declarative sentences (inspired by Identify the Champion,[d] which many evaluators are already familiar with) accompanying each level:

- The artifact greatly exceeds the expectations set by the paper.
- The artifact exceeds the expectations set by the paper.
- The artifact meets the expectations set by the paper.
- The artifact falls below the expectations set by the paper.
- The artifact greatly falls below the expectations set by the paper.

Over time we have come to think this is too fine-grained, and have settled for the simpler criterion of whether the artifact passes muster or not. Here, "expectations" is interpreted as the claims made in the paper. For instance, if a paper claimed the implementation of a new compiler for the Java programming language, it would be reasonable for the evaluators to expect the artifact would be able to process an arbitrary Java program; on the other hand, if the paper only claimed a subset of the language, say "all loop-free Java programs," then evaluators would have to lower their expectations accordingly.

In addition to "running" the artifact, the evaluators must read the paper and try to tweak provided inputs or create new ones, to test the limits of the artifact. The amount of effort to be invested is intended to be comparable to the time reviewers spend on evaluating a paper; in practice evaluators have reported spending between one and two days per artifact. Just like when reading a paper, the goal is not to render a definitive judgment but rather to provide a best-effort expert opinion.

Who should evaluate artifacts? Some have argued that evaluating artifacts is a job for the conference program committee itself. However, we believe this sits at odds with the reality of scientific reviewing. Due to high submission volumes, program

d http://scg.unibe.ch/download/champion/

committee members are in high demand. In addition, some of them are not always familiar with modern software tools and systems. We therefore think it best the AECs be populated by senior Ph.D. and postdoctoral researchers. This choice has several benefits. First, they are familiar with the technology needed to build and run artifacts. Second, in our experience, they respond with alacrity and write detailed reviews in a timely manner. Finally, and more subtly, we feel getting junior researchers involved in the process sends a message of its importance to those who will be future research leaders. One caution is that junior researchers can sometimes be overly eager at fault-finding, and their reviews may need moderation. This is why the AEC is chaired by senior researchers.

What are the benefits of artifact evaluation? The first benefit of the process is it sends a message that artifacts are valued and are an important part of the contribution of papers published in programming language conferences. Papers found to be at or above threshold get a little extra recognition, both in the proceedings and at the conference. They are marked with a special logo and distinguished in the conference proceedings. A handful of papers are selected for Distinguished Artifact Awards. Another benefit comes from the reviews themselves: several authors have confirmed the evaluators provided valuable feedback and even small bug fixes on the artifacts and on their packaging. At ECOOP 2013, for instance, some authors even claimed the artifact reviews were more useful than the reviews of the paper. For the scientific community at large, artifact evaluation encourages authors to produce reusable artifacts, which are the cornerstone of future research.

Artifact evaluation encourages authors to produce reasonable artifacts, which are the cornerstone of future research.

Should artifacts be published? While there are many good reasons for making the artifact available, there are also arguments against making artifacts public:

- The artifact may have been produced in a company and may therefore be regarded as proprietary.
- The data used in the paper's experiments may be proprietary or have high privacy needs.
- The artifact may depend on expensive or proprietary platforms that are difficult or impossible for anyone but the authors to access.
- By making the tools public, it becomes easy for others to continue that line of research, which reduces the payoff for the original researchers.

Reasonable people have come to opposite conclusions on each of these issues. In some cases, a different incentive structure might help. At any rate, it is clear that in some situations repeatability may be off limits; but these cases seem rare enough that they should not dominate the discussion.

In the long term, we would like to see evaluated artifacts be made public by mandate, as SAS 2013 did. Even as it remains optional, for authors who do wish to publish them, there remains the problem of how and where. ACM's digital library would be a natural host, and recent changes have made it possible for authors to deposit artifacts there without surrendering their copyright. Yet, the interface to the digital library is less than optimal; there are also problems with the current terms. We would prefer to use technologies that better support accessing artifacts. Furthermore, the digital library only hosts static artifacts; it would be worthwhile for it to consider combining forces with resources such as runmycode.org and researchcompendia.org.

We have come a long way. In our efforts to become more "scientific," we have moved away from papers that simply report on software projects to demanding that papers distill the novel contributions of these projects. In the process, however, we may have shifted too far, even as natural science itself has taken a lead on demanding repeatability, data sets, and public access to software; demands we recognize the need for and hence should have spearheaded. We should let the pendulum swing back to a happy medium between scientific contributions and software contributions, recognizing that ultimately, software is indeed the single most distinctive contribution our discipline has to make. So we should embrace it rather than act as if we are ashamed of it. While we report on one particular experiment in the area of programming language research, many other areas in computer science are looking at some of the same issues. References to other initiatives are included in the sidebar; also see http://www.artifact-eval.org

The ECML/PKDD'13 conference started an open science award process similar to the artifact evaluation process described here.[e] The SIGMOD conference evaluated repeatability from 2008 to 2011.[f,g] The ICERM workshop on reproducibility in computational and experimental mathematics produced a report that argues for a culture shift.[h] Journals such as *Biostatistics* are recognizing papers that are accompanied by artifacts.[i]

e http://www.ecmlpkdd2013.org/open-science-award/.
f Manegold, S. et al. Repeatability and Workability Evaluation of SIGMOD 2009. *SIGMOD Record*, September 2009.
g http://www.sigmod2011.org/calls_papers_sigmod_research_repeatability.shtml.
h hhttp://icerm.brown.edu/html/programs/topical/tw12_5_rcem/icerm_report.pdf.
i http://www.oxfordjournals.org/our_journals/biosts/for_authors/msprep_submission.html.

Shriram Krishnamurthi (sk@cs.brown.edu) is a professor of computer science at Brown University in Providence, RI.

Jan Vitek (j.vitek@neu.edu) is a professor of computer science at Northeastern University in Boston, MA.

The authors thank Andreas Zeller for taking the personal risk involved in initiating this Viewpoint. We thank Matthias Hauswirth for his enthusiasm and artwork, and Matthias, Steve Blackburn, and Camil Demetrescu for several good conversations. We thank our AEC co-chairs Eric Eide, Erik Ernst, and Carlo Ghezzi for their hard work. Most of all, we thank the AEC members for their diligent efforts, often above and beyond the call of duty, and the authors for giving the AEC members something to evaluate.