Artifact Evaluation: Is it a *Real* Incentive?

Bruce R. Childers
School of Computing and Information
University of Pittsburgh
Pittsburgh PA 15260
Email: childers@cs.pitt.edu

Panos K. Chrysanthis
School of Computing and Information
University of Pittsburgh
Pittsburgh PA 15260
Email: panos@cs.pitt.edu

I. Introduction

It is well accepted that we learn hard lessons when implementing and re-evaluating systems, yet it is also acknowledged that science faces a *crisis in reproducibility*. A remarkable study, reported in Nature [3], showed that 70% of researchers could not faithfully reproduce another study's results. Experimental computer science (CS) is far from immune to this crisis, although it should be easier for CS than other sciences, given the emphasis on encapsulating experimental artifacts, such as source code, data sets, workflows, configuration parameters, etc. Collberg and Proebsting report that only 32.3% of computer systems experiments could be reproduced [4].

Fortunately, there is growing recognition of the challenge in CS. Early on at VLDB 2007, there was a panel on "Performance Evaluation and Experimental Assessment" that debated the challenges in adopting reproducibility as a review criterion and promoted the idea that software, experiments and analyses papers should be treated equally with those offering new solutions. Recently, several conferences and journals have enabled evaluating and gaining access to the software and artifacts behind published results. Perhaps these early efforts were inspiration to other CS communities, which have encouraged many other conferences to consider similar ideas.

The question is whether the emerging emphasis on artifacts, in particular scientific software, is having a *real* incentive in computer systems research, or is it just another *fad*? While this question can certainly be asked in a much broader context, across different science communities, we examine the question fpr computer systems research, where a specific type of artifact review, Artifact Evaluation [5] (AE, artifact-eval.org) has gained traction in ACM and IEEE conferences. Our study is only in its earliest stages. With this paper, we aim to give the science software community preliminary insight into what we are learning, and seek their assistance in furthering the study, particularly to broaden it.

II. Artifact Evaluation

AE is a process to incentivize and reward authors for doing a great job in conducting experiments with robust software and data artifacts. It has been used by more than a dozen conferences, mostly for software and the experiments that use the software, since its inception in 2011. Author participation rates hover around 40%. The goal is to encourage authors to offer access to their artifacts and experiments to propel the community as a whole to do a better job. The process also rewards authors that are *already* going to great lengths to build robust software and carefully package it. As such, AE is about *positive incentives* to nudge (needle?) researchers to improving experimental methods, providing open access to their artifacts, and working together to leverage and more directly and fairly build upon and compare with one another's work.

AE is an optional, post-acceptance process, which is conducted independently of paper review and by an artifact evaluation committee (AEC) separate from the program committee. Upon paper acceptance, authors are given the option of providing their artifacts to the AEC, which checks that results/conclusions of a paper are *consistent* with the artifacts. This "consistency check" may be interpreted differently by different communities, and even from one AEC to another, in the same way that views about acceptable papers vary. Nevertheless, the check usually involves asking reviewers to actively use author-provided artifacts to repeat a portion of experiments from the paper. A paper that passes is rewarded with a badge to distinguish it. ACM has adopted a set of badges for different award degrees [2], which are put in a paper's PDF and the ACM Digital Library. Authors often proudly display the badge on their conference talks, web sites, CVs, etc. There may also be tangible rewards (e.g., financial, special paper sessions, extra proceeding pages) for especially meritious authors.

As an incentivization process, AE smartly sets aside many impediments that have stymied past attempts at artifact review. First, there is no mandate that authors release artifacts as open source to avoid objections about proprietary ownership and preserving competitive advantage. Second, review is done independently of the program committee to avoid placing more burden on overworked PCs. Third, the AEC is composed of senior graduate students and post-docs, who are often most experienced with the latest software and experimental methods and have motivation to participate for networking, learning about review practices, and learning how to create their own artifacts for review. Although the AEC may operate anonymously, discussion between reviewers and authors is often necessary and helpful to improve author materials. The review is typically confidential to allow as many authors to participate as possible, including ones from industry, which may have restrictions on their software. Fourth, paper acceptance decisions are made *prior* to artifact review to reduce workload (i.e.,

978-1-5386-2686-3/17 $31.00 © 2017 IEEE
DOI 10.1109/eScience.2017.79

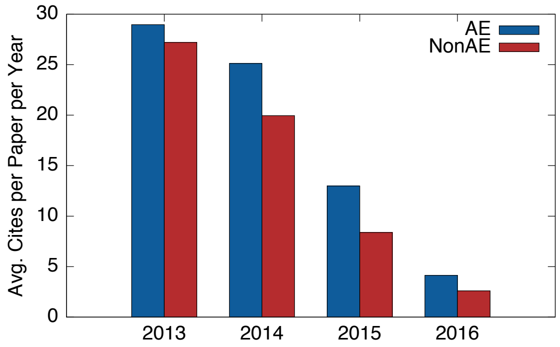

Fig. 1. Average citation counts of AE and non-AE papers.

review artifacts of accepted papers) on an optional basis (to let those that want to participate to do so, without penalizing those that do not want to participate). Finally, artifact evaluation is not coupled with paper acceptance to avoid an implicit penalty for authors that do not want to participate, or can not participate. AE is about the incentive of doing the “right thing” with an eye looking toward encouraging authors to make their experiments and artifacts available.

III. Our Study

To answer whether is AE an incentive for authors to provide their software artifacts, we used *citation count* as a proxy metric. For a conference that adopts AE, this metric sets up an experimental group of papers that successfully went through AE and a control group that either were not successful or did not participate. Citation count permits answering our question quantitatively, albeit indirectly. Of course, this metric is imprecise with many sources of bias. k evertheless, it can show trends, and indicate, at least in a preliminary way, whether AE can promote improved practices.

We did our best to identify all conferences that use the AE process, and then we systematically went through every conference program to find papers that successfully went through AE. We used Google Scholar to get citation counts for the papers[1]. Figure 1 shows the summary data for the average number of citations per paper for AE and non-AE papers for conferences in 2013 to 2016. We did not include 2011 and 2012 because only 1 conference used AE in 2011 (when it was first introduced), and no conference used it in 2012.

I(. The Ak swer

Figure 1 illustrates that AE does seem to have an effect. At least, the citation counts for the experimental group (AE papers) is higher on average than the control group. It is very important to note: There may not be a direct correlation; e.g., perhaps authors that participated in AE for whatever reason have a tendency to be more active and visible in the community, already have a history and a desire to release software, may have a history of producing high quality and innovative outcomes, etc. These biases may lead to the higher citation counts, and further and deeper study will be necessary to understand and correct for possible bias.

Table I shows citation counts on a per year basis for three conferences which regularly use AE. In these conferences, there is a trend that AE papers receive more citations. Interestingly, the table shows that older AE papers tend to collect even more citations than non-AE papers of equivalent age. For instance, in 2013, the AE papers for ECOOP had an average of 22.25 citations per paper, while the non-AE papers had 15.67. PLDI had a big spike for 2014 (60.83 vs. 26.54). However, this spike is due to one particularly influential paper in 2014 that successfully went through AE [1].

TABLE I
A(erage citatiok s per AE ak d k ok -AE per year.

Year	ECOOP		OOPSLA		PLDI	
	AE	k on-AE	AE	k on-AE	AE	k on-AE
2013	22.25	15.67	22.50	28.06	k /A	k /A
2014	11.67	11.44	13.35	12.86	60.83	26.54
2015	7.92	5.47	7.56	7.52	15.04	11.97
2016	2.50	1.00	1.00	1.34	4.55	4.33

While we can not draw a firm cause-and-effect conclusion yet, there is quantitative evidence that AE does influence, or at least, it is correlated with citation count. There are also qualitative indicators that AE and similar processes are creating incentive. The ACM, for instance, recently adopted guidelines to badge papers and their artifacts/experiments. There are community groups working on policies, rights management, guidelines, and many other aspects, which will eventually be codified into professional association practices. Perhaps the biggest carrot (or, is it a *stick*?) are mandates and signals from funders. For instance, k IH is making a large investment in the Data Commons for fair, accessible, interoperable and reusable artifacts. Likewise, the k SF has mandates for data management, open access, and even a recent Dear Colleague Letter to encourage reproducibility studies in CS. The EU has seen similar emphasis. Although it is early, the quantitative and qualitative indicators suggest that artifact evaluation is having influence and one potentially powerful incentive for producing better software and other artifacts!

Referek ces

[1] S. Arzt, S. Rasthofer, C. Fritz, E. Bodden, A. Bartel, J. Klein, Y. Le Traon, D. Octeau, and P. McDaniel. Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps. In *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI ’14, pages 259–269, k ew York, k Y, USA, 2014. ACM.

[2] Result and artifact review and badging. http://www.acm.org/publications/policies/artifact-review-badging, 2016. [Online; accessed 18-Aug-2016].

[3] M. Baker. 1,500 scientists lift the lid on reproducibility. *Nature*, 533:452–454, May 2016.

[4] C. Collberg and T. A. Proebsting. Repeatability in computer systems research. *Commun. ACM*, 59(3):62–69, Feb. 2016.

[5] S. Krishnamurthi and J. (itek. The real software crisis: Repeatability as a core value. *Commun. ACM*, 58(3):34–36, Feb. 2015.

[1] This data was collected in Spring 2017 by Anuradha Kulkarni and Seth M. Stayer as part of their degree projects.

This material is based upon work supported by the k ational Science Foundation under grants ACI-1535232, Ck S-1305220 and CBET-1609120.