Nov 12th, 3:20 PM - 3:30 PM

Open Research for Robustifying Data-Centric Systems

Manuel RIGGER
National University of Singapore (NUS)

Jinsheng BA
National University of Singapore (NUS)

Wenjing DENG
National University of Singapore (NUS)

Rajdeep Singh HUNDAL
National University of Singapore (NUS)

Nathee JAYWAREE
National University of Singapore (NUS)

See next page for additional authors

Presenter Information

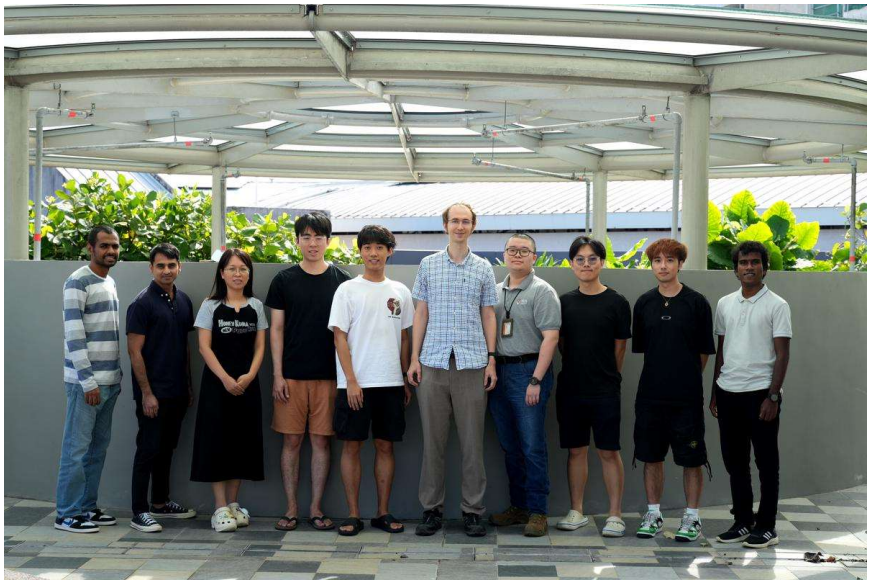
Manuel RIGGER, Jinsheng BA, Wenjing DENG, Rajdeep Singh HUNDAL, Nathee JAYWAREE, Yuancheng JIANG, Yuxi LING, Shuxin LI, Qiuyang MANG, Jovyn TAN, Ming Wei TAN, Chi ZHANG, Tongjun ZHANG, and Suyang ZHONG

Open Research for Robustifying Data-Centric Systems

Manuel Rigger

National University of Singapore

Lab Introduction

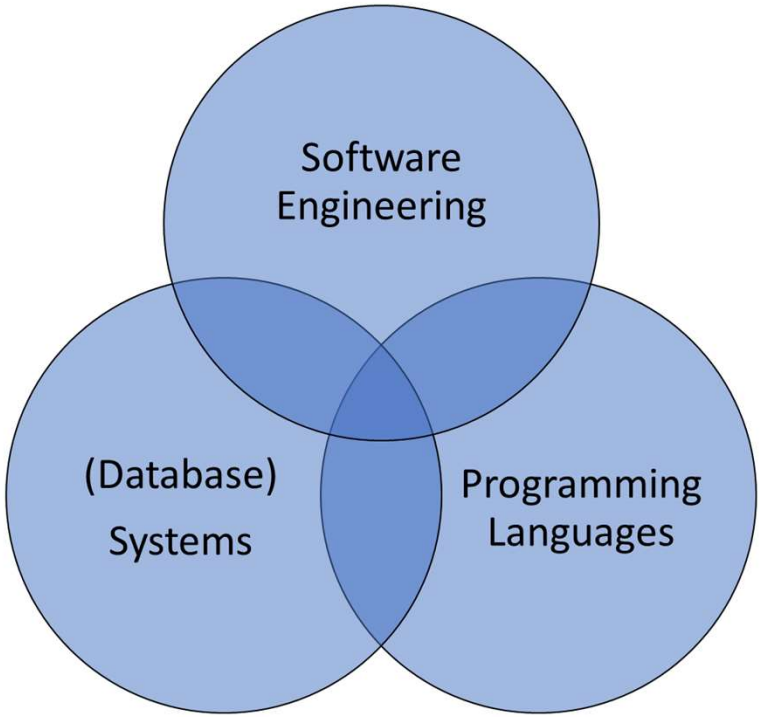


Trustworthy Engineering of Software Technologies Lab

https://nus-test.github.io/

Lab Introduction

Key disciplines



We work in applied computer science

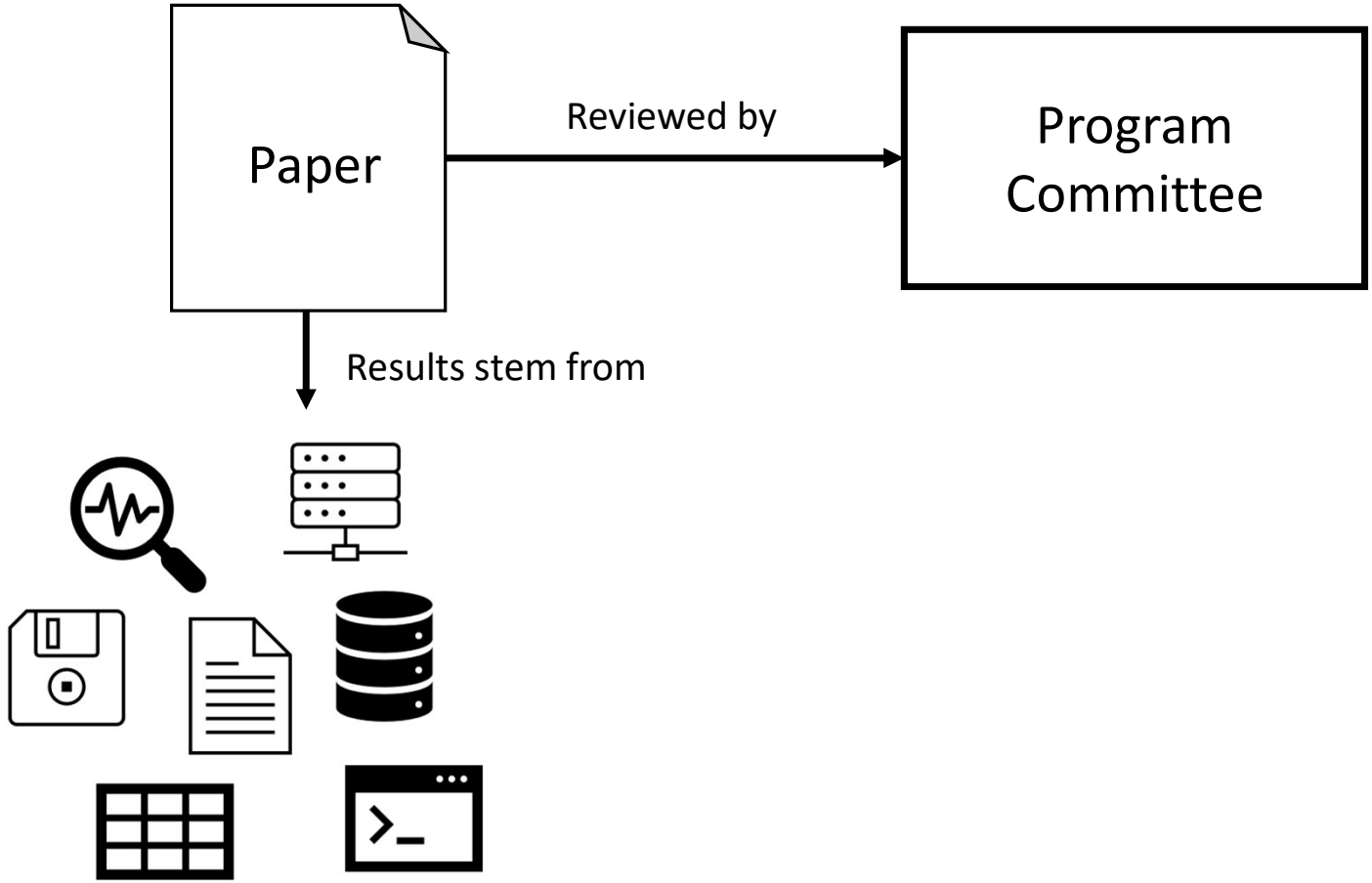
Research Methodology

Engineering research or design science: propose and evaluate technological artifacts, including algorithms, models, languages, methods, systems, tools, and other computer-based technologies

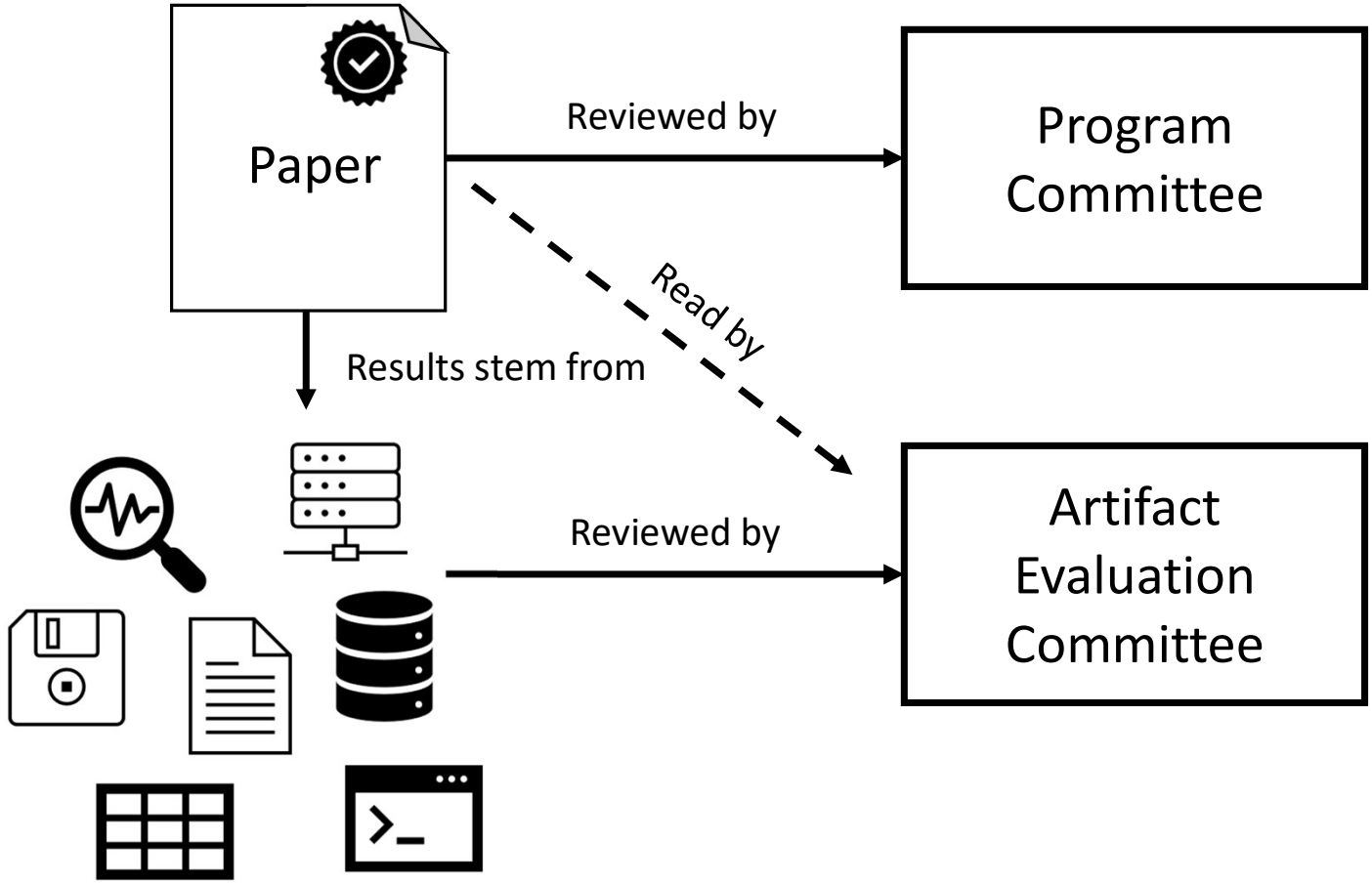
Research Methodology

Goal of talk: overview of the artifact evaluation process as well as how our lab goes beyond it

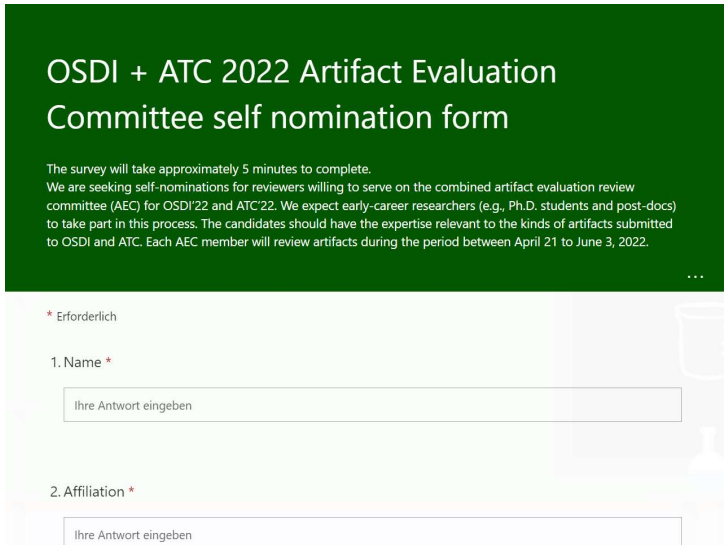
What's Artifact Evaluation (AE)?



What's Artifact Evaluation (AE)?



What's Artifact Evaluation (AE)?



Typically consists of junior members of the community (PhD students, postdocs, …)

Artifact Evaluation Committee

Why Artifact Evaluation?

- Validating reproducibility
- Validating reusability
- Fostering both

Community Expectations for Research Artifacts and Evaluation Processes

Ben Hermann
ben.hermann@upb.de
Heinz Nixdorf Institut
Universität Paderborn
Paderborn, Germany

Stefan Winter
sw@cs.tu-darmstadt.de
Dependable Systems and Software
Technische Universität Darmstadt
Darmstadt, Germany

Janet Siegmund
janet.siegmund@informatik.tu-chemnitz.de
Technische Universität Chemnitz
Chemnitz, Germany

ABSTRACT

Artifact evaluation has been introduced into the software engineering and programming languages research community with a pilot at ESEC/FSE 2011 and has since then enjoyed a healthy adoption throughout the conference landscape. In this qualitative study, we examine the expectations of the community toward research artifacts and their evaluation processes. We conducted a survey including all members of artifact evaluation committees of major conferences in the software engineering and programming language field since the first pilot and compared the answers to expectations set by calls for artifacts and reviewing guidelines. While we find that some expectations exceed the ones expressed in calls and reviewing guidelines, there is no consensus on quality thresholds for artifacts in general. We observe very specific quality expectations for specific artifact types for review and later usage, but also a lack of their communication in calls. We also find problematic inconsistencies in the terminology used to express artifact evaluation's most important purpose – *replicability*. We derive several actionable suggestions which can help to mature artifact evaluation in the inspected community and also to aid its introduction into other communities in computer science.

CCS CONCEPTS

• **General and reference**; • **Software and its engineering** → *Software libraries and repositories*; *Software verification and validation*;

1 INTRODUCTION

In 2016, a replicability crisis became public, when more than 1500 researchers revealed having trouble replicating previous research results [1]. This replicability crisis also reached the software engineering community, as it has embraced the importance of replication for knowledge building [3, 4, 15, 21, 22]. For example, Collberg and Proebsting could not obtain the relevant artifacts to conduct a replication, neither by contacting the authors, the authors' institution, and funding agency [7]. Also, Lung et al. describe their difficulties in conducting an exact replication, even when they were in direct contact with the authors [17]. Glanz et al. describe similar experiences when obtaining research artifacts for comparison and had to reimplement competing approaches in order to replicate results [10]. For the term *artifact*, we follow the definition provided by Méndez et al. [18], describing it as a self-contained work result with a context-specific purpose.

To improve the situation of missing or unusable artifacts, artifact evaluation has become a regular process for scientific conferences in the software engineering and programming language communities. It contributes to the larger trend towards open science in computer science. Since the first piloting of the process at ESEC/FSE 2011, many other conferences have included artifact evaluations as an additional step that authors of accepted papers may take. If their artifact is successfully evaluated the corresponding publication is marked with a *badge* [9, 11] indicating different levels by which the artifact is found to support the presented research results. Successfully evaluated artifacts are listed on the conference website

https://dl.acm.org/doi/10.1145/3368089.3409767

Why Artifact Evaluation?

- Validating reproducibility
- Validating reusability
- Fostering both

Different team can reproduce the results using the same experimental set-up

Community Expectations for Research Artifacts and Evaluation Processes

Ben Hermann
ben.hermann@upb.de
Heinz Nixdorf Institut
Universität Paderborn
Paderborn, Germany

Stefan Winter
sw@cs.tu-darmstadt.de
Dependable Systems and Software
Technische Universität Darmstadt
Darmstadt, Germany

Janet Siegmund
janet.siegmund@informatik.tu-chemnitz.de
Technische Universität Chemnitz
Chemnitz, Germany

ABSTRACT

Artifact evaluation has been introduced into the software engineering and programming languages research community with a pilot at ESEC/FSE 2011 and has since then enjoyed a healthy adoption throughout the conference landscape. In this qualitative study, we examine the expectations of the community toward research artifacts and their evaluation processes. We conducted a survey including all members of artifact evaluation committees of major conferences in the software engineering and programming language field since the first pilot and compared the answers to expectations set by calls for artifacts and reviewing guidelines. While we find that some expectations exceed the ones expressed in calls and reviewing guidelines, there is no consensus on quality thresholds for artifacts in general. We observe very specific quality expectations for specific artifact types for review and later usage, but also a lack of their communication in calls. We also find problematic inconsistencies in the terminology used to express artifact evaluation's most important purpose – *replicability*. We derive several actionable suggestions which can help to mature artifact evaluation in the inspected community and also to aid its introduction into other communities in computer science.

CCS CONCEPTS

• **General and reference**; • **Software and its engineering** → *Software libraries and repositories*; *Software verification and validation*;

1 INTRODUCTION

In 2016, a replicability crisis became public, when more than 1500 researchers revealed having trouble replicating previous research results [1]. This replicability crisis also reached the software engineering community, as it has embraced the importance of replication for knowledge building [3, 4, 15, 21, 22]. For example, Collberg and Proebsting could not obtain the relevant artifacts to conduct a replication, neither by contacting the authors, the authors' institution, and funding agency [7]. Also, Lung et al. describe their difficulties in conducting an exact replication, even when they were in direct contact with the authors [17]. Glanz et al. describe similar experiences when obtaining research artifacts for comparison and had to reimplement competing approaches in order to replicate results [10]. For the term *artifact*, we follow the definition provided by Méndez et al. [18], describing it as a self-contained work result with a context-specific purpose.

To improve the situation of missing or unusable artifacts, artifact evaluation has become a regular process for scientific conferences in the software engineering and programming language communities. It contributes to the larger trend towards open science in computer science. Since the first piloting of the process at ESEC/FSE 2011, many other conferences have included artifact evaluations as an additional step that authors of accepted papers may take. If their artifact is successfully evaluated the corresponding publication is marked with a *badge* [9, 11] indicating different levels by which the artifact is found to support the presented research results. Successfully evaluated artifacts are listed on the conference website

https://dl.acm.org/doi/10.1145/3368089.3409767

Why Artifact Evaluation?

- Validating reproducibility
- Validating reusability
- Fostering both

Other researchers can build on the artifact

Community Expectations for Research Artifacts and Evaluation Processes

Ben Hermann
ben.hermann@upb.de
Heinz Nixdorf Institut
Universität Paderborn
Paderborn, Germany

Stefan Winter
sw@cs.tu-darmstadt.de
Dependable Systems and Software
Technische Universität Darmstadt
Darmstadt, Germany

Janet Siegmund
janet.siegmund@informatik.tu-chemnitz.de
Technische Universität Chemnitz
Chemnitz, Germany

ABSTRACT

Artifact evaluation has been introduced into the software engineering and programming languages research community with a pilot at ESEC/FSE 2011 and has since then enjoyed a healthy adoption throughout the conference landscape. In this qualitative study, we examine the expectations of the community toward research artifacts and their evaluation processes. We conducted a survey including all members of artifact evaluation committees of major conferences in the software engineering and programming language field since the first pilot and compared the answers to expectations set by calls for artifacts and reviewing guidelines. While we find that some expectations exceed the ones expressed in calls and reviewing guidelines, there is no consensus on quality thresholds for artifacts in general. We observe very specific quality expectations for specific artifact types for review and later usage, but also a lack of their communication in calls. We also find problematic inconsistencies in the terminology used to express artifact evaluation's most important purpose – *replicability*. We derive several actionable suggestions which can help to mature artifact evaluation in the inspected community and also to aid its introduction into other communities in computer science.

CCS CONCEPTS

• **General and reference**; • **Software and its engineering** → *Software libraries and repositories*; *Software verification and validation*;

1 INTRODUCTION

In 2016, a replicability crisis became public, when more than 1500 researchers revealed having trouble replicating previous research results [1]. This replicability crisis also reached the software engineering community, as it has embraced the importance of replication for knowledge building [3, 4, 15, 21, 22]. For example, Collberg and Proebsting could not obtain the relevant artifacts to conduct a replication, neither by contacting the authors, the authors' institution, and funding agency [7]. Also, Lung et al. describe their difficulties in conducting an exact replication, even when they were in direct contact with the authors [17]. Glanz et al. describe similar experiences when obtaining research artifacts for comparison and had to reimplement competing approaches in order to replicate results [10]. For the term *artifact*, we follow the definition provided by Méndez et al. [18], describing it as a self-contained work result with a context-specific purpose.

To improve the situation of missing or unusable artifacts, artifact evaluation has become a regular process for scientific conferences in the software engineering and programming language communities. It contributes to the larger trend towards open science in computer science. Since the first piloting of the process at ESEC/FSE 2011, many other conferences have included artifact evaluations as an additional step that authors of accepted papers may take. If their artifact is successfully evaluated the corresponding publication is marked with a *badge* [9, 11] indicating different levels by which the artifact is found to support the presented research results. Successfully evaluated artifacts are listed on the conference website

https://dl.acm.org/doi/10.1145/3368089.3409767

Why Artifact Evaluation?

- Validating reproducibility
- Validating reusability
- Fostering both

Badges provide a motivation and recognition for authors

Community Expectations for Research Artifacts and Evaluation Processes

Ben Hermann
ben.hermann@upb.de
Heinz Nixdorf Institut
Universität Paderborn
Paderborn, Germany

Stefan Winter
sw@cs.tu-darmstadt.de
Dependable Systems and Software
Technische Universität Darmstadt
Darmstadt, Germany

Janet Siegmund
janet.siegmund@informatik.tu-chemnitz.de
Technische Universität Chemnitz
Chemnitz, Germany

ABSTRACT

Artifact evaluation has been introduced into the software engineering and programming languages research community with a pilot at ESEC/FSE 2011 and has since then enjoyed a healthy adoption throughout the conference landscape. In this qualitative study, we examine the expectations of the community toward research artifacts and their evaluation processes. We conducted a survey including all members of artifact evaluation committees of major conferences in the software engineering and programming language field since the first pilot and compared the answers to expectations set by calls for artifacts and reviewing guidelines. While we find that some expectations exceed the ones expressed in calls and reviewing guidelines, there is no consensus on quality thresholds for artifacts in general. We observe very specific quality expectations for specific artifact types for review and later usage, but also a lack of their communication in calls. We also find problematic inconsistencies in the terminology used to express artifact evaluation's most important purpose – *replicability*. We derive several actionable suggestions which can help to mature artifact evaluation in the inspected community and also to aid its introduction into other communities in computer science.

CCS CONCEPTS

• **General and reference**; • **Software and its engineering** → *Software libraries and repositories*; *Software verification and validation*;

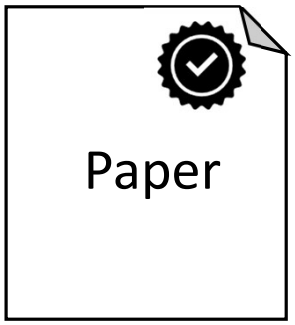
1 INTRODUCTION

In 2016, a replicability crisis became public, when more than 1500 researchers revealed having trouble replicating previous research results [1]. This replicability crisis also reached the software engineering community, as it has embraced the importance of replication for knowledge building [3, 4, 15, 21, 22]. For example, Collberg and Proebsting could not obtain the relevant artifacts to conduct a replication, neither by contacting the authors, the authors' institution, and funding agency [7]. Also, Lung et al. describe their difficulties in conducting an exact replication, even when they were in direct contact with the authors [17]. Glanz et al. describe similar experiences when obtaining research artifacts for comparison and had to reimplement competing approaches in order to replicate results [10]. For the term *artifact*, we follow the definition provided by Méndez et al. [18], describing it as a self-contained work result with a context-specific purpose.

To improve the situation of missing or unusable artifacts, artifact evaluation has become a regular process for scientific conferences in the software engineering and programming language communities. It contributes to the larger trend towards open science in computer science. Since the first piloting of the process at ESEC/FSE 2011, many other conferences have included artifact evaluations as an additional step that authors of accepted papers may take. If their artifact is successfully evaluated the corresponding publication is marked with a *badge* [9, 11] indicating different levels by which the artifact is found to support the presented research results. Successfully evaluated artifacts are listed on the conference website

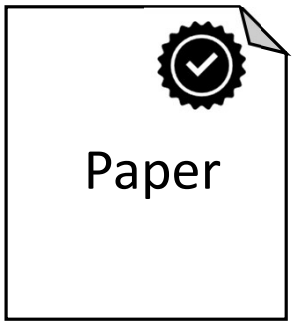
https://dl.acm.org/doi/10.1145/3368089.3409767

What Badges?



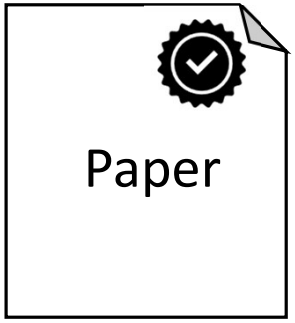
Different badge systems depending on the publisher

What Badges?



Different badge systems depending on the publisher

What Badges?



Different badge systems depending on the publisher

Experience of an Artifact Evaluation Chair

Artifact Evaluation Co-Chairing

- PLDI ’24
- PLDI ‘23
- OSDI ‘21
- ISSTA ‘21
- ECOOP ‘20
- ECOOP ‘19

~ 100 submit artifacts and Artifact Evaluation Committee members, reviewing around 3 artifacts each

PLDI Research Artifacts

About | Info for Reviewers | Call for Artifacts

Background

A paper consists of a constellation of artifacts that extend beyond the document itself: software, proofs, models, test suites, benchmarks, and so on. In some cases, the quality of these artifacts is as important as that of the document itself, yet most of our conferences offer no formal means to submit and evaluate anything but the paper.

Following a trend in our community over the past many years, PLDI 2024 includes an Artifact Evaluation process, which allows authors of accepted papers to optionally submit supporting artifacts. The goal of artifact evaluation is two-fold: to probe further into the claims and results presented in a paper, and to reward authors who take the trouble to create useful artifacts to accompany the work in their paper. Artifact evaluation is optional, but highly encouraged, and authors

nd enables authors to build on top of each other’s
semination of artifacts confers several direct and

artifact that is submitted, provided it meets the
ible to that ideal goal. However, even though
te in earnest and make our best attempt to follow

ge how well the submitted artifact conforms to the
e:

in principle reproduce the main results reported in the paper.

- *Completeness:* the artifact can in principle reproduce all the results that the paper reports, and should include everything (code, tools, 3rd party libraries, etc.) required to do so.
- *Documentation:* the artifact should be well documented so that generating the results is easy and transparent.
- *Ease of reuse:* the artifact provides everything needed to build on top of the original work, including source files together with a working build process that can recreate the binaries provided.

Note that artifacts will be evaluated with respect to the claims and presentation in the submitted version of the paper, not the camera-ready version.

Badges

The artifact evaluation committee evaluates each artifact for the awarding of one or two badges:

Functional: This is the basic “accepted” outcome for an artifact. An artifact can be awarded a functional badge if the artifact supports all claims made in the paper, possibly excluding some minor claims if there are very good reasons they cannot be supported. In the ideal case, an artifact with this designation includes all relevant code, dependencies, input data (e.g., benchmarks), and the artifact’s documentation is sufficient, in principle, for reviewers to reproduce the exact results described in the

Important Dates — AoE (UTC-12h)

Wed 24 Apr 2024
Author Notification

Wed 27 Mar 16:58 - Mon 15 Apr 12:00 2024
Response and Communication Period

Mon 11 Mar 2024
Artifact Submission Deadline

Mon 4 Mar 2024
Artifact Registration Deadline

Fri 22 Dec 2023
Reviewer Self-Nomination Deadline

Submission Link

https://pldi24ae.hotcrp.com/

Artifact Evaluation Committee

Manuel Rigger — Co-chair
National University of Singapore
Singapore

Ningning Xie — Co-chair
University of Toronto
Canada

Rui Dong
University of Michigan
United States

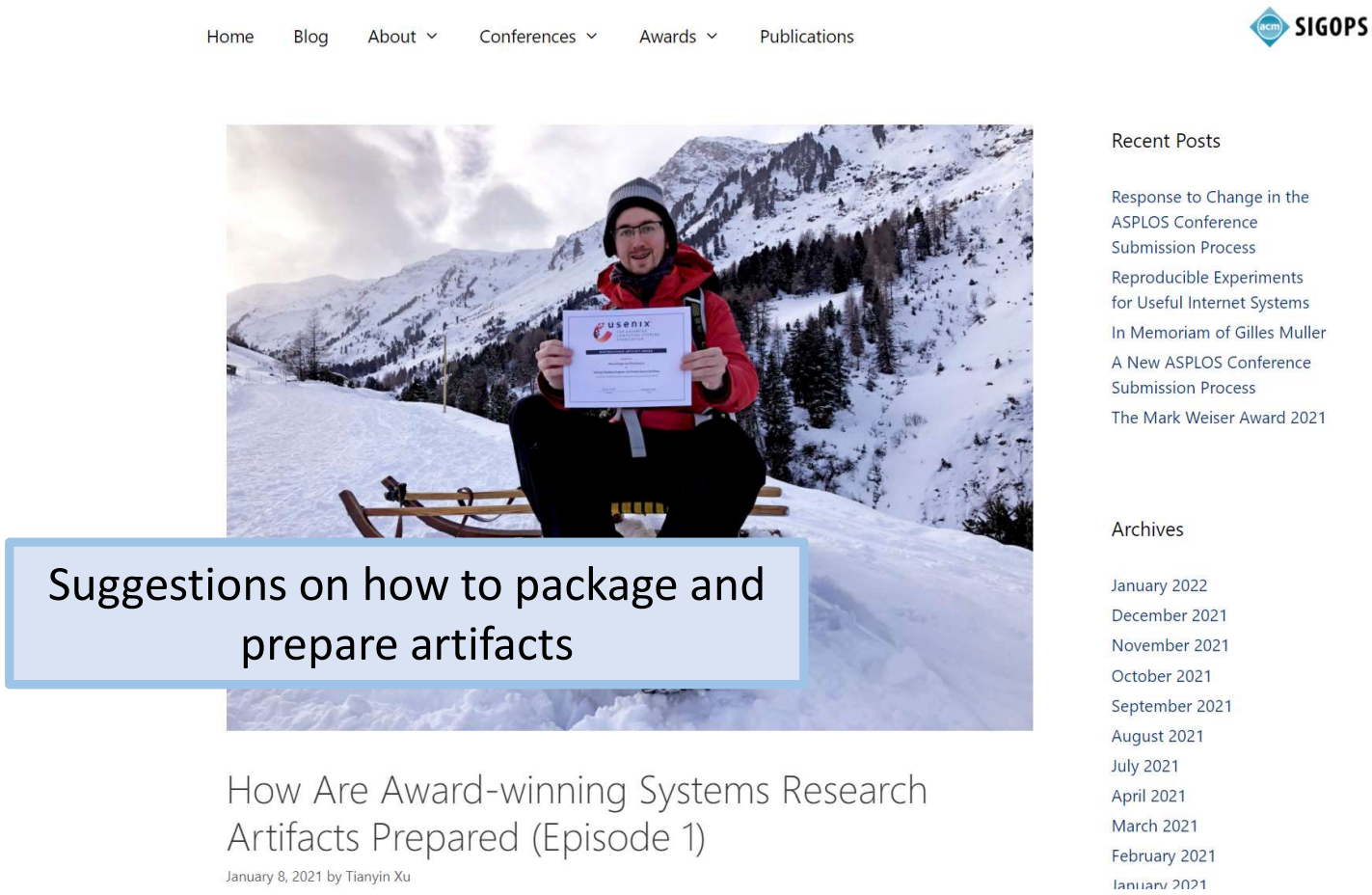
Song Liao
Clemson University
United States

Xiangzhe Xu
Purdue University
United States

https://pldi24.sigplan.org/track/pldi-2024-pldi-research-artifacts

Distinguished Artifact Award Winner

- ASPLOS ’22
- OSDI ‘20
- OOPSLA ‘20



Suggestions on how to package and prepare artifacts

https://www.sigops.org/2021/how-are-award-winning-systems-research-artifacts-prepared-part-1/

More Resources on Artifact Evaluation

README

The purpose of this page is to provide a resource collection about the Artifact Evaluation (AE) process that has been introduced for many publishing venues in Computer Science (CS) in order to promote and help researchers understand (and improve) the AE process.

General Resources

Process

- https://www.artifact-eval.org/: General information on the AE process, origins, and packaging guidelines.
- https://sysartifacts.github.io/: Information on the AE processes in system conferences, including calls, committees, and results.

Advocacy

- Artefact Review and Badging: Improving Confidence in our Experimental Results by Michel Steuwer
- The Real Software Crisis: Repeatability as a Core Value by Shriram Krishnamurthi and Jan Vitek

Artifact Types

- Proof Artifacts. Guidelines for Submission and Reviewing by Marianna Rapoport: Instructions on creating and reviewing proof artifacts.
- Checking machine-checked proofs by Assia Mahboubi: Instructions on how to review machine-checked proofs as well as suggestions for authors and organizers.

https://github.com/csartifacts/resources

A Retrospective Study of One Decade of Artifact Evaluations

Stefan Winter
LMU Munich
Munich, Germany
sw@stefan-winter.net

Christopher S. Timperley
Carnegie Mellon University
Pittsburgh, USA
ctimperley@cmu.edu

Ben Hermann
Technische Universität Dortmund
Dortmund, NRW, Germany
ben.hermann@cs.tu-dortmund.de

Jürgen Cito
TU Wien
Vienna, Austria
juergen.cito@tuwien.ac.at

Jonathan Bell
Northeastern University
Boston, MA, USA
j.bell@northeastern.edu

Michael Hilton
Carnegie Mellon University
Pittsburgh, PA, USA
mhilton@cmu.edu

Dirk Beyer
LMU Munich
Munich, Germany
dirk.beyer@sosy-lab.org

ABSTRACT

Most software-engineering research involves the development of a prototype, a proof of concept, or a measurement apparatus. Together with the data collected in the research process, they are collectively referred to as research artifacts and are subject to artifact evaluation (AE) at scientific conferences. Since its initiation in the software-engineering community at ESEC/FSE 2011, both the goals and the process of AE have evolved and today expectations towards AE are strongly linked with reproducible research results and reusable tools that other researchers can build their work on. However, to date little evidence has been provided that artifacts that have passed AE actually live up to these high expectations, i.e., to which degree AE processes contribute to AE's goals and whether the overhead they impose is justified.

We aim to fill this gap by providing an in-depth analysis of research artifacts from a decade of software engineering (SE) and programming languages (PL) conferences, based on which we reflect on the goals and mechanisms of AE in our community. In summary, our analyses (1) suggest that articles with artifacts do not generally have better visibility in the community, (2) provide evidence how evaluated and not evaluated artifacts differ with respect to different quality criteria, and (3) highlight opportunities for further improving AE processes.

KEYWORDS

Research artifacts, Artifact evaluation, Open science, Reproduction, Reuse, Long-term availability of software and data

ACM Reference Format:
Stefan Winter, Christopher S. Timperley, Ben Hermann, Jürgen Cito, Jonathan Bell, Michael Hilton, and Dirk Beyer. 2022. A Retrospective Study of One Decade of Artifact Evaluations. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '22), November 14–18, 2022, Singapore, Singapore.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3540250.3549172

CCS CONCEPTS

• **General and reference** → **Empirical studies**; • **Software and its engineering** → *Software post-development issues*; • **Information systems** → **Digital libraries and archives**.

1 INTRODUCTION

As reported in a 2016 Nature article, the scientific research community faces a "reproducibility crisis." 70 % of the 1 576 scientists surveyed by Nature (from various fields, including chemistry, physics, earth and environmental science, biology, and medicine) reported that they had tried and failed to reproduce another scientist's experiments [3]. Numerous conferences for computer science (including the software-engineering field) organize artifact evaluations with the goal to ensure reproducibility. Organizers assign badges based on peer review to recognize authors' efforts to make their tools and datasets available and reusable, and integrate these artifacts into publication processes. In the software community the artifact-evaluation process started at ESEC/FSE in 2011 [15] [1], and has now spread to become commonplace at most conferences in the area of software engineering and programming languages as well as other communities, including HCI, Communications, and Security.

As different communities have different requirements regarding research artifacts, artifact-evaluation organizers use different eval-

https://dl.acm.org/doi/pdf/10.1145/3540250.3549172

Community Expectations for Research Artifacts and Evaluation Processes

Ben Hermann
ben.hermann@upb.de
Heinz Nixdorf Institut
Universität Paderborn
Paderborn, Germany

Stefan Winter
sw@cs.tu-darmstadt.de
Dependable Systems and Software
Technische Universität Darmstadt
Darmstadt, Germany

Janet Siegmund
janet.siegmund@informatik.tu-chemnitz.de
Technische Universität Chemnitz
Chemnitz, Germany

ABSTRACT

Artifact evaluation has been introduced into the software engineering and programming languages research community with a pilot at ESEC/FSE 2011 and has since then enjoyed a healthy adoption throughout the conference landscape. In this qualitative study, we examine the expectations of the community toward research artifacts and their evaluation processes. We conducted a survey including all members of artifact evaluation committees of major conferences in the software engineering and programming language field since the first pilot and compared the answers to expectations set by calls for artifacts and reviewing guidelines. While we find that some expectations exceed the ones expressed in calls and reviewing guidelines, there is no consensus on quality thresholds for artifacts in general. We observe very specific quality expectations for specific artifact types for review and later usage, but also a lack of their communication in calls. We also find problematic inconsistencies in the terminology used to express artifact evaluation's most important purpose – *replicability*. We derive several actionable suggestions which can help to mature artifact evaluation in the inspected community and also to aid its introduction into other communities in computer science.

CCS CONCEPTS

• **General and reference**; • **Software and its engineering** → *Software libraries and repositories*; *Software verification and validation*;

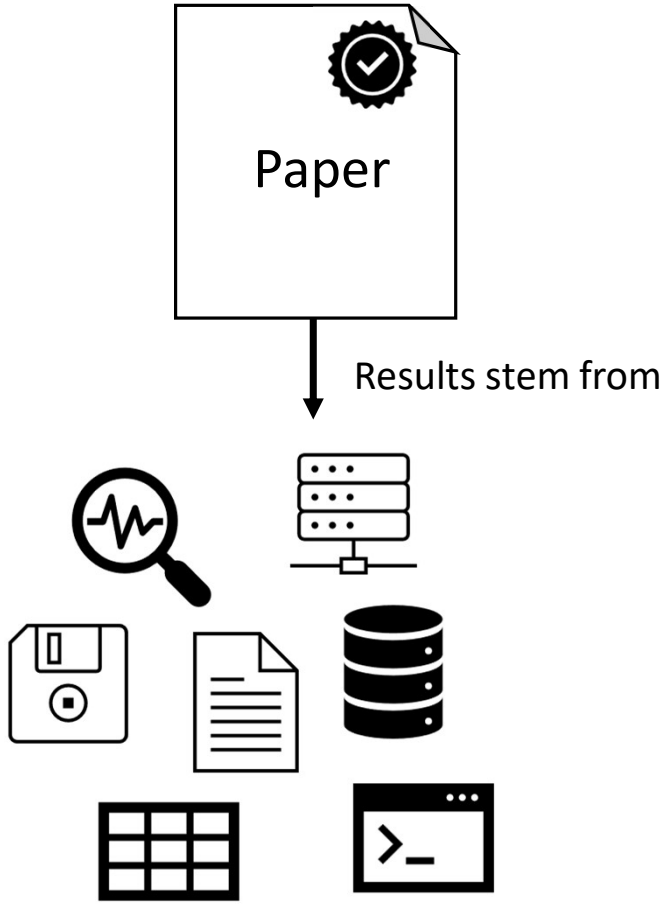
1 INTRODUCTION

In 2016, a replicability crisis became public, when more than 1500 researchers revealed having trouble replicating previous research results [1]. This replicability crisis also reached the software engineering community, as it has embraced the importance of replication for knowledge building [3, 4, 15, 21, 22]. For example, Collberg and Proebsting could not obtain the relevant artifacts to conduct a replication, neither by contacting the authors, the authors' institution, and funding agency [7]. Also, Lung et al. describe their difficulties in conducting an exact replication, even when they were in direct contact with the authors [17]. Glanz et al. describe similar experiences when obtaining research artifacts for comparison and had to reimplement competing approaches in order to replicate results [10]. For the term *artifact*, we follow the definition provided by Méndez et al. [18], describing it as a self-contained work result with a context-specific purpose.

To improve the situation of missing or unusable artifacts, artifact evaluation has become a regular process for scientific conferences in the software engineering and programming language communities. It contributes to the larger trend towards open science in computer science. Since the first piloting of the process at ESEC/FSE 2011, many other conferences have included artifact evaluations as an additional step that authors of accepted papers may take. If their artifact is successfully evaluated the corresponding publication is marked with a *badge* [9, 11] indicating different levels by which the artifact is found to support the presented research results. Successfully evaluated artifacts are listed on the conference website

https://dl.acm.org/doi/10.1145/3368089.3409767

Going Beyond Artifact Evaluation



In practice, most authors stop further building on or maintain their artifacts

Going Beyond Artifact Evaluation

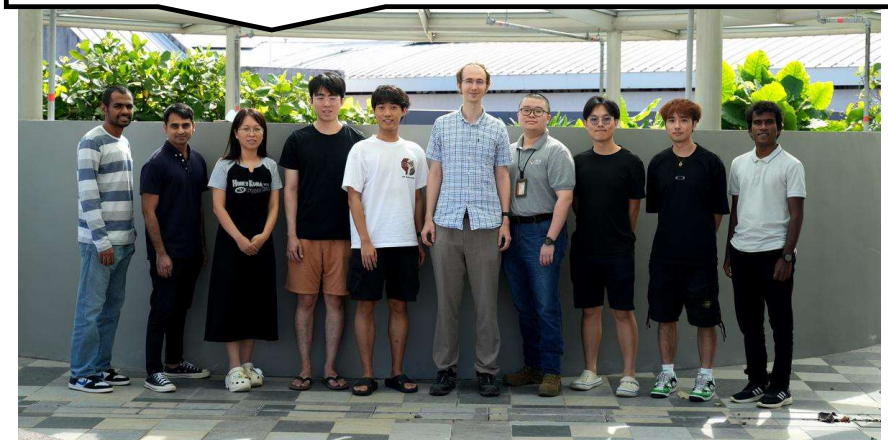
Not supported on the latest version of my system!

I would like to use it, but feature *X* is not supported

How can I extend it to achieve *Y*?

Lab

Besides doing research, we want to create (and maintain) impactful tools!



Trustworthy Engineering of Software Technologies Lab

https://nus-test.github.io/

SQLancer

https://github.com/sqlancer/sqlancer

Automated testing to find logic and performance bugs in database systems

www.sqlancer.com/

MIT license

Code of conduct

1.4k stars 265 forks 34 watching 17 Branches 4 Tags Activity

SQLancer: Approaches

ICSE '23

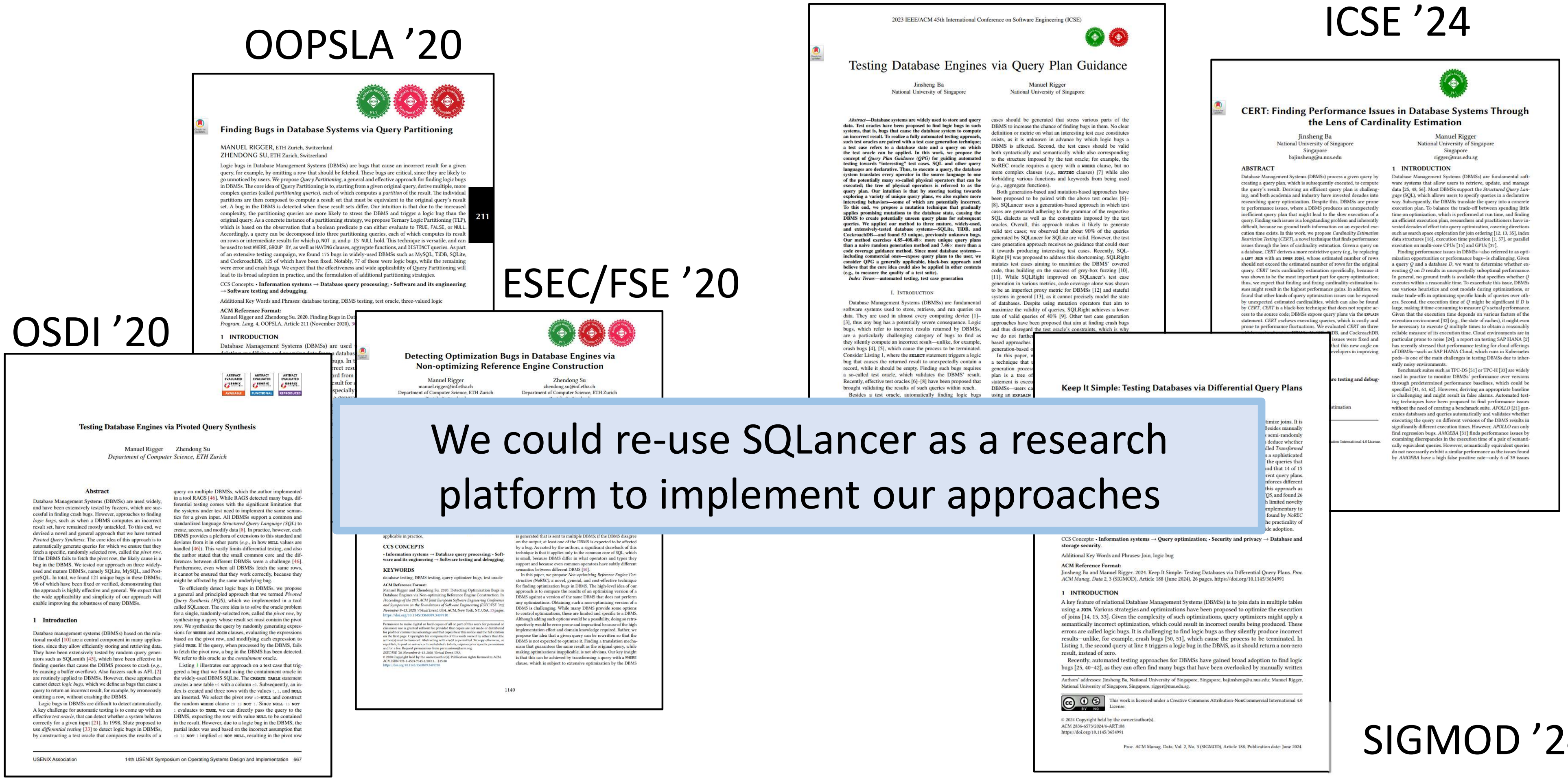
ICSE '24

OOPSLA '20

ESEC/FSE '20

OSDI '20

SIGMOD '24



We could re-use SQLancer as a research platform to implement our approaches

TLP and QPG As the State of the Art

Many other researchers used SQLancer to implement their approaches, to compare with in their evaluation, and for reproduction studies

A Comprehensive S
Techniques, Taxono

XIYUE GAO, ZHUANG
HUI LI*, Xidian University,
HUI ZHANG, KEWEI WEI, and KANKAN ZHAO, Inspur, China

Database Management System (DBMS) fuzzing is an automated testing technique aimed at detecting errors and vulnerabilities in

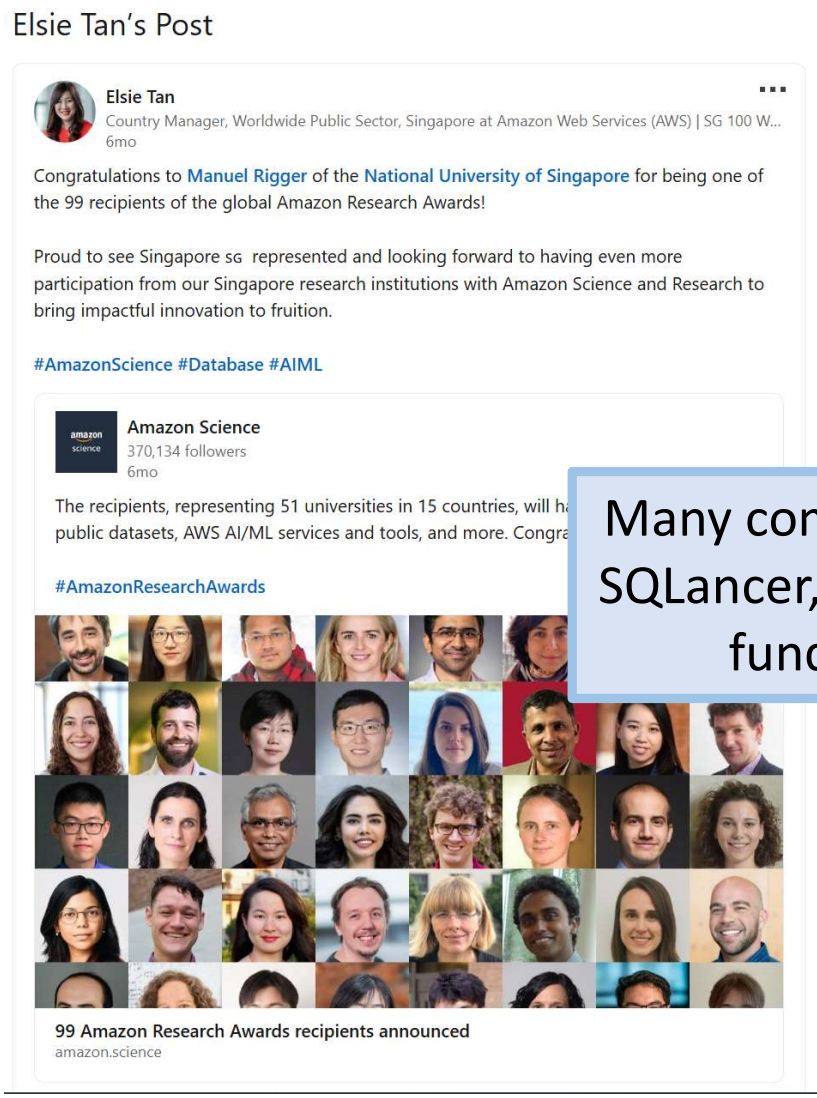
*"It can be observed **that TLP and QPG are quite exceptional**, as they quickly detected bugs in MySQL. They discovered 12 bugs in just 40 minutes, after which the system crashed (crash detected) […]"*

Based on this toolkit, we conduct a detailed experimental comparative analysis of existing methods and finally discuss future research directions.

CCS Concepts: • **Information systems** → **Database performance evaluation**; **Database performance evaluation**; • **Software and its engineering** → *Software testing and debugging*.

Additional Key Words and Phrases: Automated database testing, fuzzing, DBMS fuzzing, DBMS fuzz testing, Experimental comparison.

AWS



Many companies have adopted SQLancer, contributed to it, and funded our research

https://www.linkedin.com/feed/update/urn:li:activity:7190185979704414208/

New Problems

Users of SQLancer provided us with new research challenges

We first looked into a tool called SQLancer. […] It would **take a lot of work to properly integrate Vitess with SQLancer**, due to each DBMS tester in SQLancer essentially being written completely separately with similar logic. […] We decided to go for the low-hanging fruit and build our own random query generator.

https://planetscale.com/blog/summer-2023-fuzzing-vitess-at-planetscale

Scaling Automated Testing (Under Submission)

Toward Automated Database System Testing at Scale

Zhong Suyang
suyang@u.nus.edu
National University of Singapore
Singapore

Manuel Rigger
rigger@nus.edu.sg
National University of Singapore
Singapore

Abstract

Recently, various automated testing approaches have been proposed that have found hundreds of bugs in mature, widely-used Database Management Systems (DBMSs). At the heart of these approaches are so-called deep kinds of bugs, such as logic b to compute an incorrect result fo oracles require database and que account for the often significant d dialects of these systems. Since to implement such generators, many DBMS developers are unlikely to invest the time to adopt such automated testing approaches. In short, existing approaches fail to scale to the plethora of existing DBMSs. In this work, we present both a vision and a platform, SQLancer++, to apply automated DBMS testing approaches *at scale*. Our technical core contribution is a novel architecture for an *adaptive SQL statement generator*. This adaptive SQL generator generates SQL statements with various features, some of which might not be supported by the given DBMS, and then learns through interaction with the DBMS, which of these are understood by the DBMS. Thus, over time, the generator will generate mostly valid SQL statements. We evaluated SQLancer++ across 15

Applying database system testing at scale!

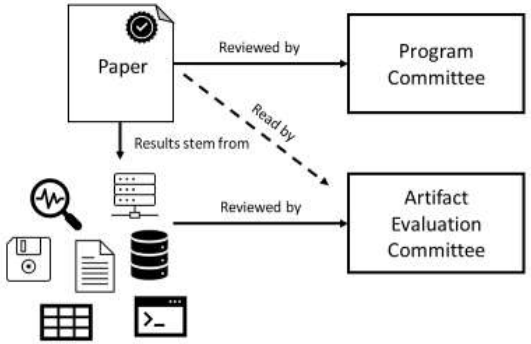
by mutating given seed inputs. More recently, automated testing approaches for DBMSs have been proposed that find so-called *logic bugs* [1, 15, 27, 30–32, 38, 42], which are bugs that cause a system to silently compute an incorrect result, to find. Many of these works mpute the correct result by eserving way and checking e. Overall, these approaches widely-known DBMSs such QL.

It would be ideal to apply the automated DBMS testing approaches to the hundreds, if not thousands, of DBMSs that exist. For example, a recent effort of documenting and classifying DBMS lists close to 1,000 existing database systems.[1] The market for DBMSs is significant, currently being 162.25 USD and growing at a compound annual growth rate (CAGR) of 15.2% [6], fueling the development of new DBMSs, as well as further development of existing ones. With the end of Moore's law, various trends that have set in posing new reliability challenges, such as the development of new, increasingly specialized DBMSs, often based on SQL and the relational model. In addition, existing DBMSs are becoming increasingly complex, by using accelerators [17, 23, 37], dis-

https://suyang.zone/

Summary

What's Artifact Evaluation (AE)?



Why Artifact Evaluation?

- Validating reproducibility
- Validating reusability
- Fostering both

Badges provide a motivation and recognition for authors


Community Expectations for Research Artifacts and Evaluation Processes

https://dl.acm.org/doi/10.1145/3368089.3409767

Lab

Besides doing research, we want to create (and maintain) impactful tools!

TEST

Trustworthy Engineering of Software Technologies Lab

https://nus-test.github.io/

SQLancer

SQLANCER