Popper: Practical Reproducible Evaluation of Systems Research

Ivo Jimenez[u], Michael Sevilla[u], Noah Watkins[u], Carlos Maltzahn[u], Jay Lofstead[s], Kathryn Mohror[l], Remzi Arpaci-Dusseau[w] and Andrea Arpaci-Dusseau[w]

[u]*UC Santa Cruz* [s]*Sandia National Labs* [l]*Lawrence Livermore National Labs* [w]*UW Madison*

Independently validating experimental results in the field of computer systems research is a challenging task. Recreating an environment that resembles the one where an experiment was originally executed is a time-consuming endeavour. In this WIP, we present Popper [1], a convention (or protocol) for conducting experiments following a DevOps [2] approach that allows researchers to automate the re-execution and validation of an experiment.

Over the last decade software engineering and systems administration communities (also referred to as DevOps) have developed sophisticated techniques and strategies to ensure "software reproducibility", i.e. the reproducibility of software artifacts and their behavior using versioning, dependency management, containerization, orchestration, monitoring, testing and documentation. The key idea behind the Popper Convention is to manage every experiment in computation and data exploration as a software project, using tools and services that are readily available now and enjoy wide popularity. By doing so, scientific explorations become reproducible with the same convenience, efficiency, and scalability as software reproducibility while fully leveraging continuing improvements to these tools and services. Rather than mandating a particular set of tools, the convention only expects components of an experiment to be scripted (see Fig. 1). There are two main goals for Popper:

1. It should be usable in as many research projects as possible, regardless of their domain.
2. It should abstract underlying technologies without requiring a strict set of tools, making it possible to apply it on multiple toolchains.

We say that an experiment is Popper-compliant (or that it has been "Popperized") if all of the following are available, either directly or by reference, in one single source code repository: experiment code, experiment orchestration code, reference to data dependencies, parametrization of experiment, validation criteria and results. In other words, a Popper repository contains all the dependencies for one or more experiments, optionally including a manuscript (article or tech report) that documents them.

We maintain a list of "Popperized" experiments at `http://github.com/systemslab/popper`. We also provide a CLI tool for researchers to bootstrap a project that follows the convention, as well as a wiki with guides and examples. Projects that follow the convention can make use of our `http://falsifiable.us` service to automatically validate an experiment.

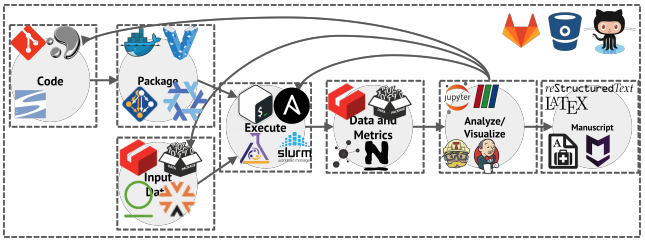

Figure 1: A generic experimentation workflow viewed through a DevOps looking glass. The logos correspond to commonly used tools from the DevOps toolkit. Scripts corresponding to each stage are stored in a version control repository, whose commit log resembles a lab notebook .

Listing 1 Interacting with the Popper-CLI tool.

```
$ cd mypaper-repo
$ popper init
-- Initialized Popper repo

$ popper experiment list
-- available templates ---------------
ceph-rados    proteustm  mpip
cloverleaf    gassyfs    zlog
spark-bench   torpor     malacology

$ popper add torpor myexp
-- Added torpor experiment to mypaper-repo

$ popper check myexp
-- SUCCESS - myexp is Popper-compliant
```

Bibliography

[1] I. Jimenez, M. Sevilla, N. Watkins, C. Maltzahn, J. Lofstead, K. Mohror, R. Arpaci-Dusseau, and A. Arpaci-Dusseau, *Popper: Making Reproducible Systems Performance Evaluation Practical*, UC Santa Cruz, SOE-16-10, 2016.

[2] M. Httermann, *DevOps for Developers*, 2012.