Popper: Practical Reproducible Evaluation of Systems

Ivo Jimenez, Michael Sevilla, Noah Watkins, Andrea Arpaci-Dusseau, Remzi Arpaci-Dusseau, Jay Lofstead, Carlos Maltzhan, Kathryn Mohror

CROSS CENTER FOR RESEARCH IN OPEN SOURCE SOFTWARE
UC SANTA CRUZ
Sandia National Laboratories
Baskin Engineering

Reproducibility as a DevOps Problem

- Independently validating experimental results is challenging.
- Recreating experimental setup is often difficult to impossible.
- Software engineers deal with reproducibility all the time:
  - Bug **A** can be reproduced in version **X** on platform **Y** using input **Z**.
- Shared (cloud) computing and storage services readily available.
- **Manage an academic article as a software project!**

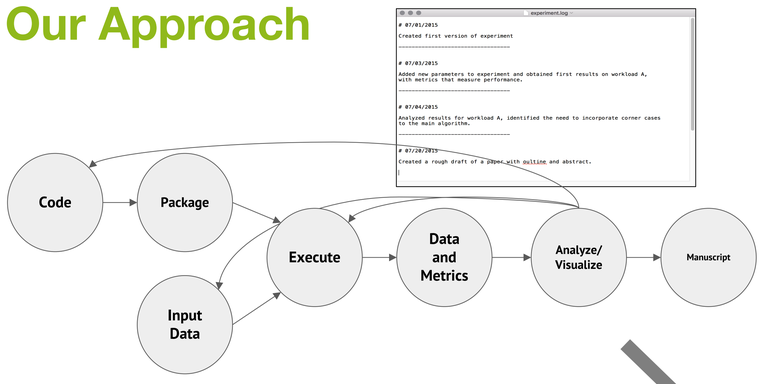
Import Existing Projects

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

$ popper search
-- available templates ---------------
ceph-rados   proteustm  mpi-comm    adam     sirius    comd-openmp
cloverleaf   gassyfs    zlog        bww      unum-py   cuddn-deeplrn
spark-stand  torpor     malacology  genevo   mantle    rita-idx
hadoop-yarn  kubsched   alg-encycl  macrob   dadvisor  obfuscdata

$ popper add gassyfs
-- Added gassyfs experiment to mypaper-repo
```

Project Structure

```
mypaper-repo
| README.md
| .git/
| .popper.yml
| pipelines
|   |-- gassyfs
|   |   |-- README.md
|   |   |-- ansible/
|   |   |   |-- setup.yml
|   |   |   |-- vars.yml
|   |   |-- geni/
|   |   |   |-- request.py
|   |   |-- results/
|   |   |   |-- figure.png
|   |   |   |-- postprocess.py
|   |   |   |-- output.csv
|   |   |-- run.sh
|   |   |-- setup.sh
|   |   |-- teardown.sh
|   |    -- validate.sh
| paper
|   |-- build.sh
|   |-- figures/
|   |-- paper.tex
|    -- references.bib
```

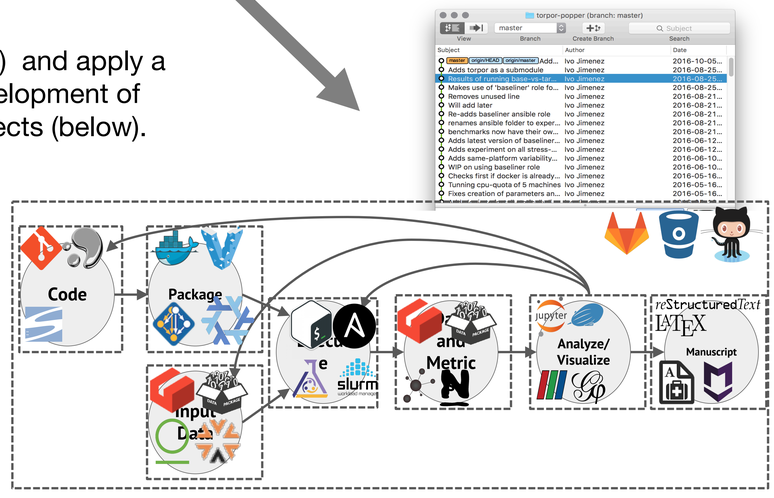
Our Approach



Popper: Take a common generic experimentation workflow (above) and apply a *DevOps* practice used in the development of open source software (OSS) projects (below).

The Convention:

1. Pick a DevOps tool for each stage of the scientific experimentation workflow.
2. Put all associated scripts (experiment and manuscript) in version control, in order to provide a self-contained repository.
3. Document changes as experiment evolves, in the form of version control commits.



Scaffolding

```
$ popper init exp1
-- Initialized exp1 pipeline.

$ ls -l pipelines/exp1/
total 20K
-rw-r----- 1 ivo ivo README.md
-rwxr-x--- 1 ivo ivo run.sh
-rwxr-x--- 1 ivo ivo setup.sh
-rwxr-x--- 1 ivo ivo teardown.sh
-rwxr-x--- 1 ivo ivo validate.sh
```

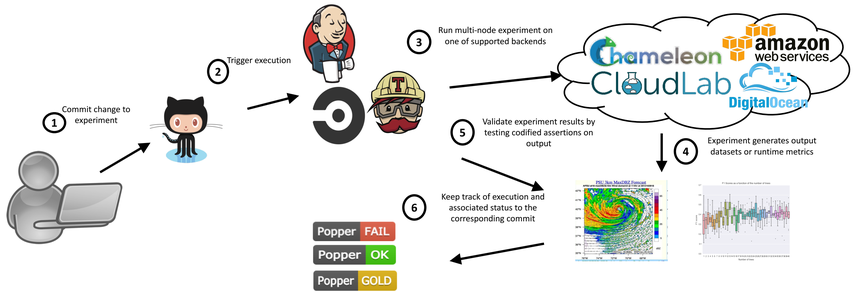
Validation

```
$ popper check exp1

Popper check started

Stage: setup.sh .....
Stage: run.sh ................
Stage: teardown.sh ..

Popper check finished

Status: SUCCESS
```

Codified Validations:

```
WHEN
    NOT network_saturated AND num_nodes=*
EXPECT
    system_throughput >= (baseline_throughput * 0.9)
```

Popper
falsifiable.us



Popper Compliance

Tools: Generate referenceable assets (associate unique IDs to binaries, data, configuration and infrastructure state); usable from scripts/CLI and capable of acting upon IDs.
Experiment: Provide all necessary assets in a single repository (self-contained), including experiment code, orchestration logic, data dependencies, results and validation criteria.
Article: Provide full text and figures of article, as well as all popper-compliant experiments.

Benefits and Challenges

Pros:
- Experiments can be falsifiable with minimal re-execution effort.
- Facilitates collaboration by following the OSS model for sharing.
- Investing time in DevOps skills quickly pays off.
- The convention complements many existing efforts.

Challenges:
- Steep learning curve of DevOps practices and tools/frameworks.
- Big cultural change; new experimentation paradigm.