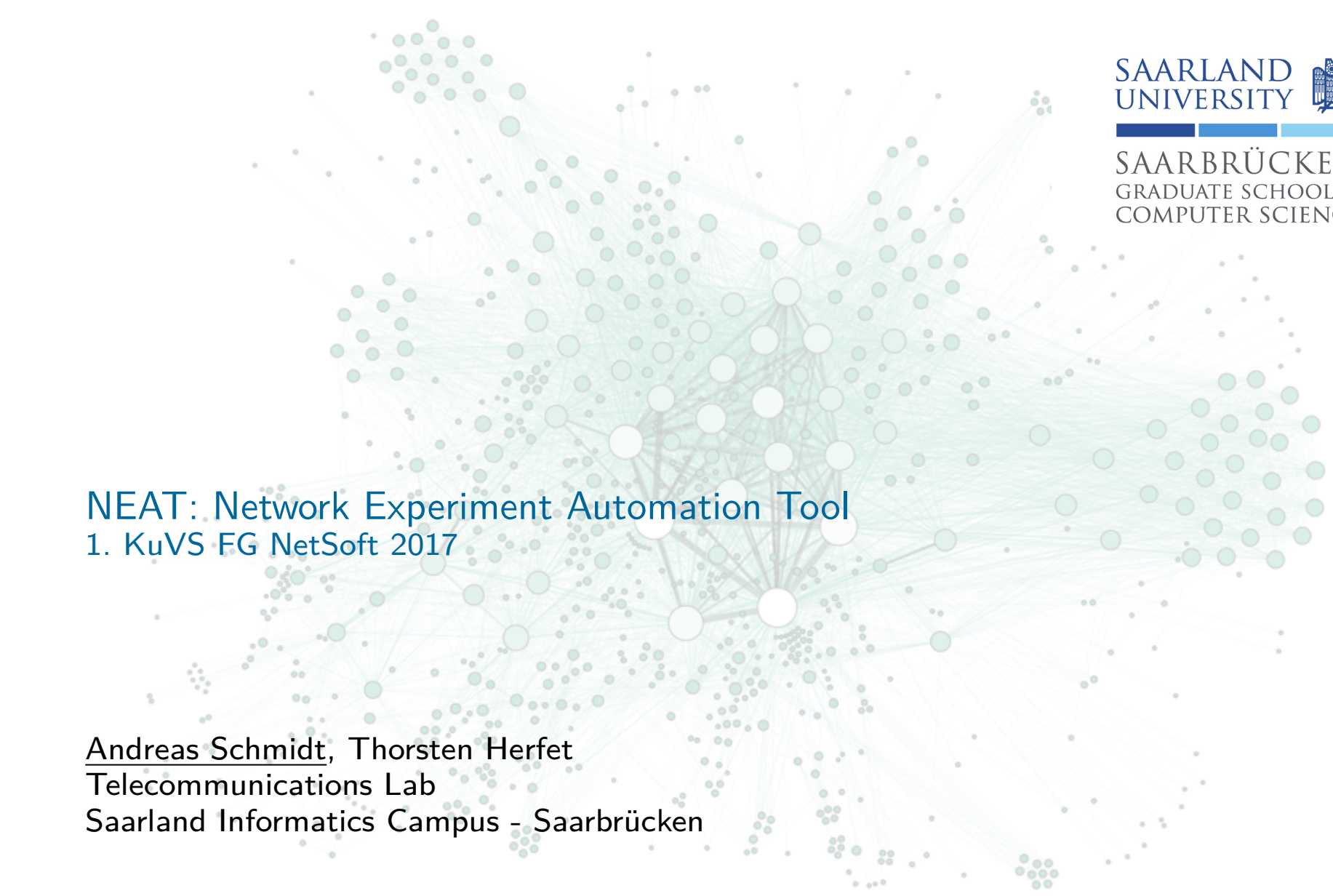


SAARLAND UNIVERSITY
SAARBRÜCKEN GRADUATE SCHOOL OF COMPUTER SCIENCE

NEAT: Network Experiment Automation Tool

1. KuVS FG NetSoft 2017

Andreas Schmidt, Thorsten Herfet
Telecommunications Lab
Saarland Informatics Campus - Saarbrücken

October 13, 2017

Motivation

Executing network experiments is crucial for **validating results**.

Motivation

Executing network experiments is crucial for **validating results**.

Current Challenges

- Experimentation process is **highly manual** and **seldomly reproducible**.
- In many cases, network simulations or purely virtual networks are used which **lack fidelity** and suffer from **resource limitations**.
- Peer reviews are **nearly impossible** (also due to lack of time, but thats a different story...).

Motivation

Executing network experiments is crucial for **validating results**.

Current Challenges

- Experimentation process is **highly manual** and **seldomly reproducible**.
- In many cases, network simulations or purely virtual networks are used which **lack fidelity** and suffer from **resource limitations**.
- Peer reviews are **nearly impossible** (also due to lack of time, but thats a different story...).

Networking experiments can be made more **reliable**, **automated** and **reproducible**.

Experimentation Rules

- §1: "For every result, keep track of how it was produced".
- §3: "Archive the exact versions of all external programs used".
- §4: "Version control all custom scripts".
- §5: "Record all intermediate results, when possible in standardized formats".

Experimentation Rules

- §1: "For every result, keep track of how it was produced".
- §3: "Archive the exact versions of all external programs used".
- §4: "Version control all custom scripts".
- §5: "Record all intermediate results, when possible in standardized formats".

Analysis & Collaboration Rules

- §2: "Avoid Manual Data Manipulation Steps".
- §7: "Always Store Raw Data behind Plots"
- §10: "Provide Public Access to Scripts, Runs, and Results"

[Sandve2013]: "Ten Simple Rules for Reproducible Computational Research"

An Abstract Network Experiment

Goal: Gather evidence that approach A provides Z in environments such as E.

Goal: Gather evidence that approach A provides Z in environments such as E.

Prepare

- Wire and configure topology: Nodes & Links (loss, delay, throughput).
- [Deploy SDN controller: Setup and connect nodes.]
- Add end-hosts: Connect and install applications.

An Abstract Network Experiment

Goal: Gather evidence that approach A provides Z in environments such as E.

Prepare

- Wire and configure topology: Nodes & Links (loss, delay, throughput).
- [Deploy SDN controller: Setup and connect nodes.]
- Add end-hosts: Connect and install applications.

Execute

- Start an application on a certain host.
- [Trigger an SDN function.]

Goal: Gather evidence that approach A provides Z in environments such as E.

Prepare

- Wire and configure topology: Nodes & Links (loss, delay, throughput).
- [Deploy SDN controller: Setup and connect nodes.]
- Add end-hosts: Connect and install applications.

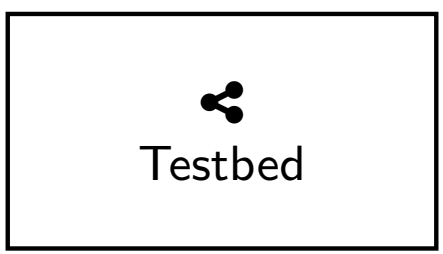
Execute

- Start an application on a certain host.
- [Trigger an SDN function.]

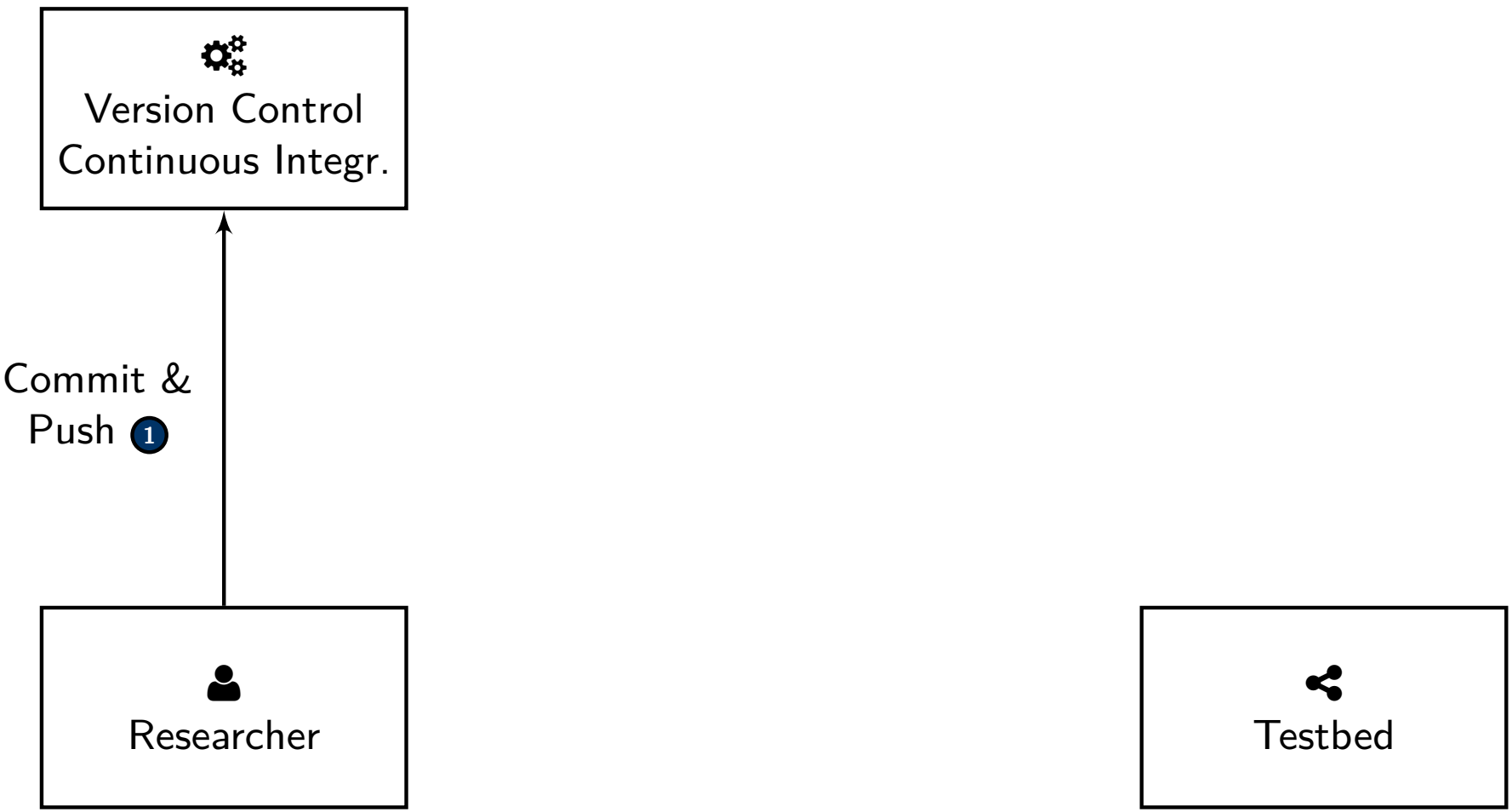
Cleanup

- Gather the execution data (`pcap`, `csv`, flow stats, logs, ...).
- Remove hosts and reset topology.

NEAT: In Action



NEAT: In Action



Goal: Fully **automate** and **document** the development process.

Goal: Fully **automate** and **document** the development process.

Benefits

- Check-in code and build artefacts automatically (§3,4).
- Every software component is associated and changes are tracked (§3,4).
- Inconsistencies with manual processes are avoided (§1).

Goal: Fully **automate** and **document** the development process.

Benefits

- Check-in code and build artefacts automatically (§3,4).
- Every software component is associated and changes are tracked (§3,4).
- Inconsistencies with manual processes are avoided (§1).

Software Solutions

- **GitLab (+CI)**: open source, self-hosted, ...
- Redmine + Jenkins: open source, self-hosted, ...
- GitHub + Travis: public, (enterprise versions exist)
- ...

NEAT: In Detail

Create A New Software Version

- `hobbes@dev-pc|~/ryu$ git commit -m "Tweak latency weights."`

```
[master 650b41e] Tweak latency weights.
    Date: Mon Sep 25 17:14:32 2017 +0200
    3 files changed, 118 insertions(+)
```

NEAT: In Detail

Create A New Software Version

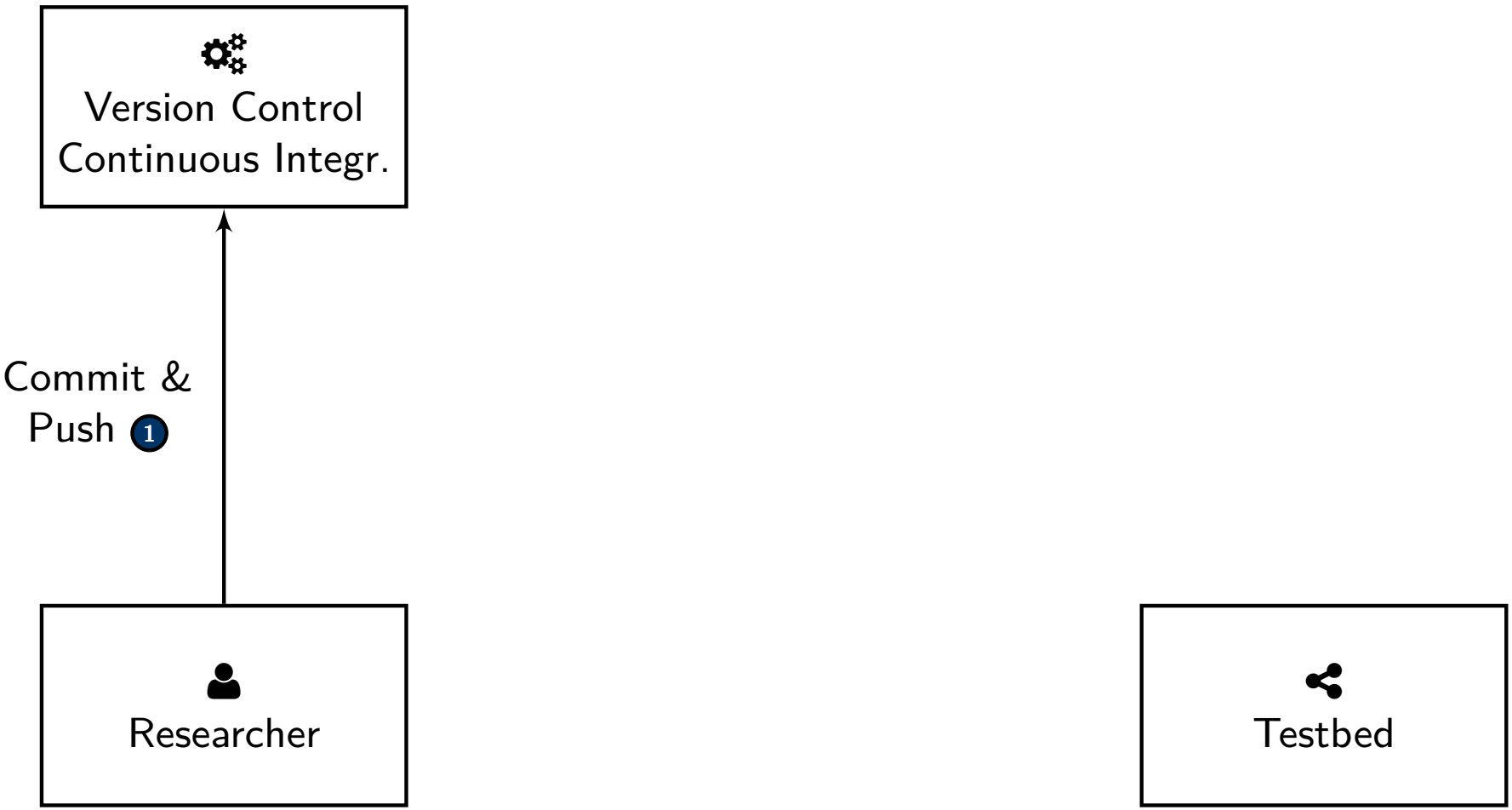
- `hobbes@dev-pc|~/ryu$ git commit -m "Tweak latency weights."`

```
[master 650b41e] Tweak latency weights.
    Date: Mon Sep 25 17:14:32 2017 +0200
    3 files changed, 118 insertions(+)
```

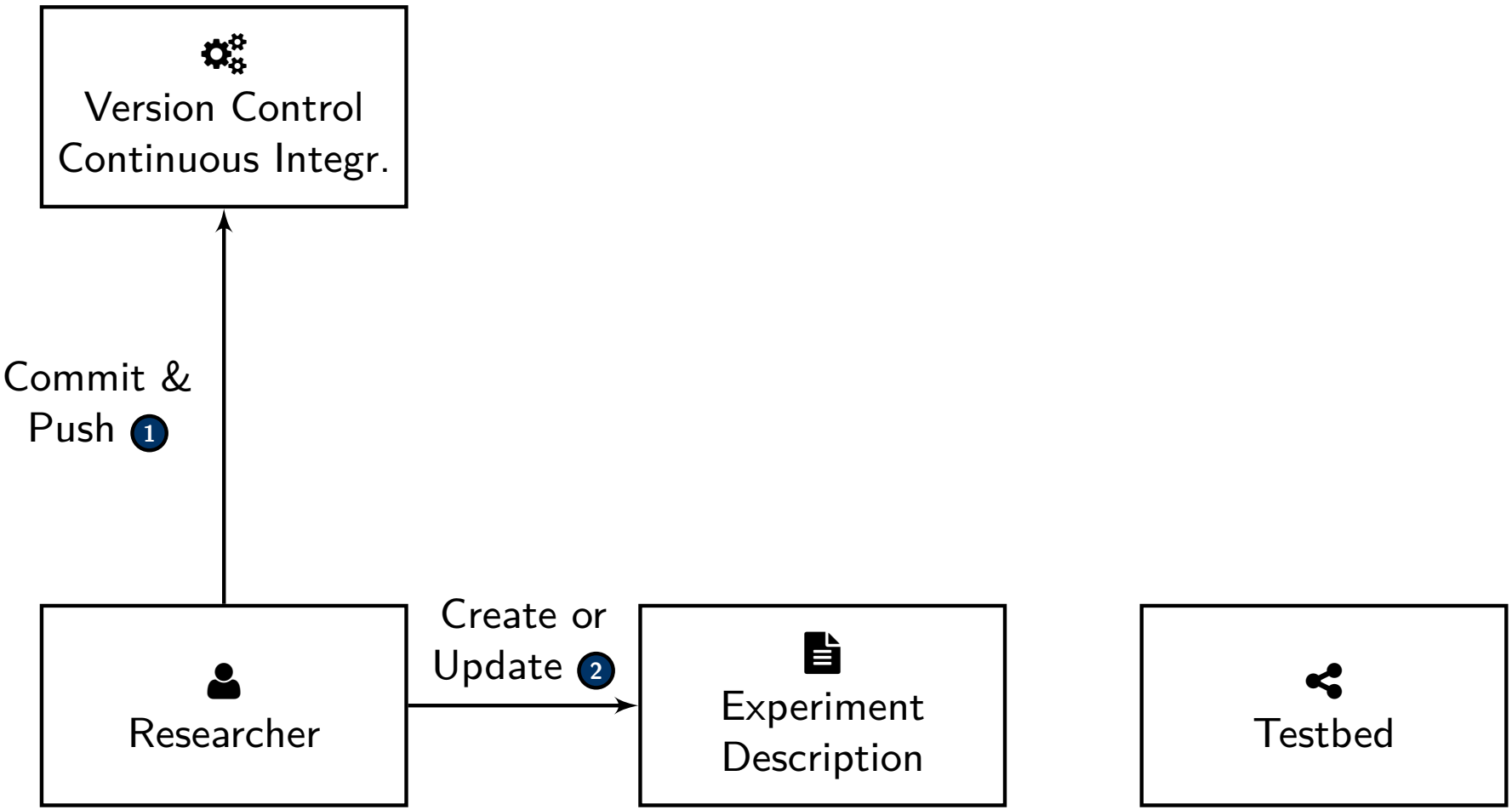
Create A New Software Artefact

- GitLab CI builds, compiles and creates the Docker image `ryu:650b41e`.
- `ryu:650b41e` is pushed to `registry.uds.on`.

NEAT: In Action



NEAT: In Action



Experiment Description

Goal: **Machine- and human-readable** description of all experiment parameters.

Experiment Description

Goal: **Machine- and human-readable** description of all experiment parameters.

Benefits

- Origins of results are thoroughly tracked (§1).
- Description can be shipped or checked into version control (§4).

Experiment Description

Goal: **Machine- and human-readable** description of all experiment parameters.

Benefits

- Origins of results are thoroughly tracked (§1).
- Description can be shipped or checked into version control (§4).

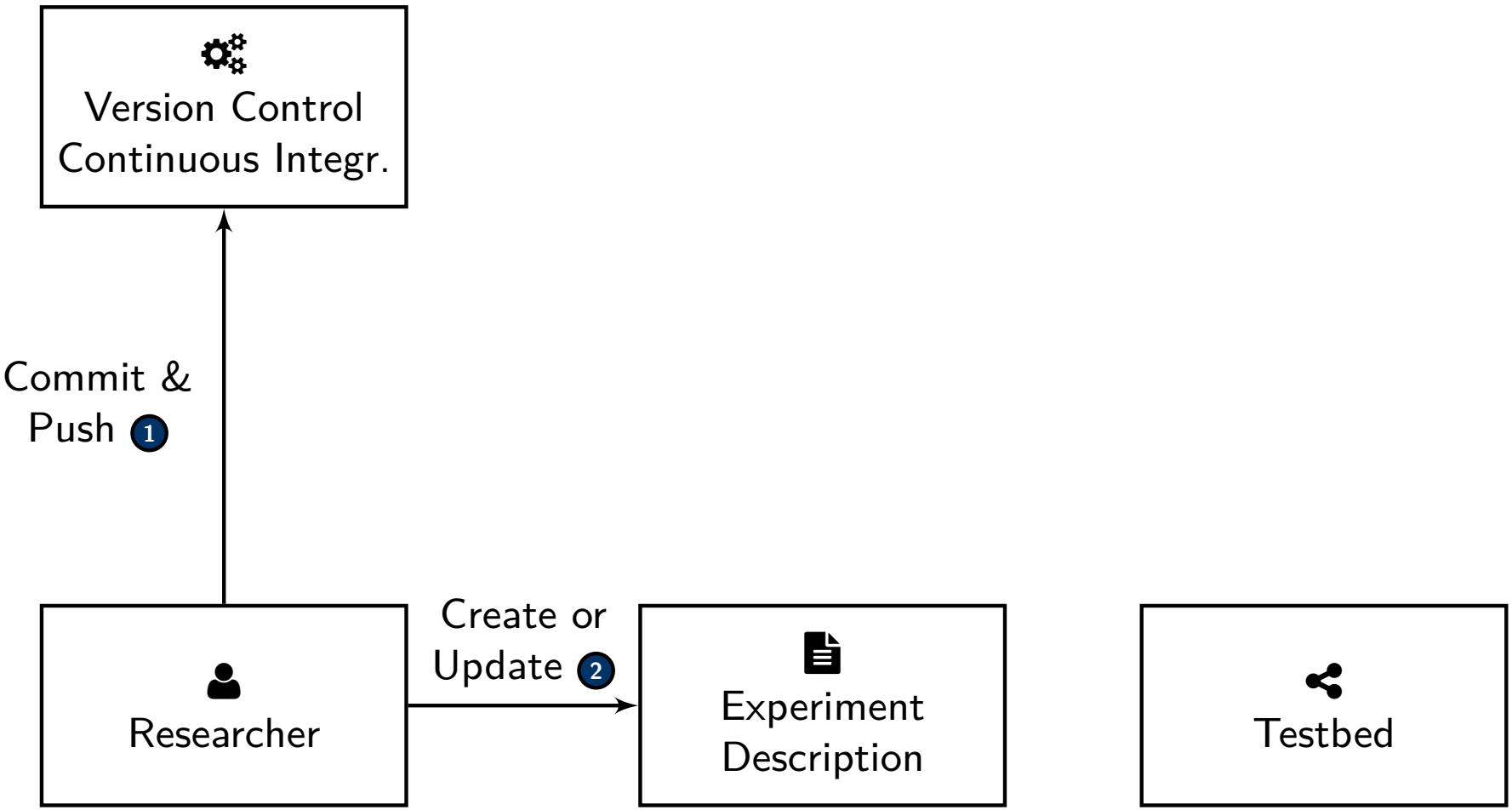
Formats

- **YAML**: many features, easy to read, ...
- JSON: least features, no comments, ...
- XML: most structure, highly verbose, hard to write
- ...

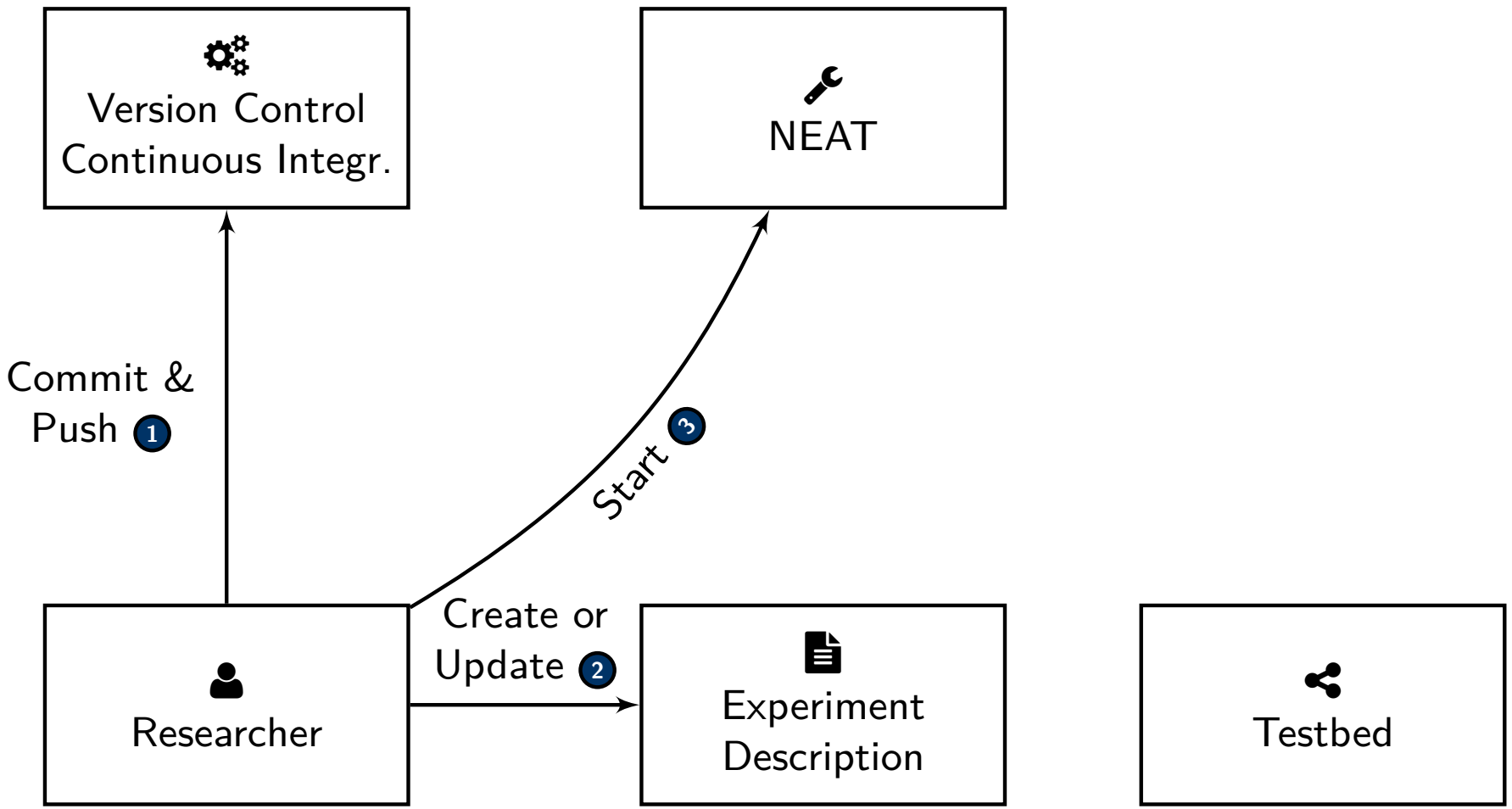
rtt_experiment3.yml

```
controller:
  minion: ctrl.uds.on
  image: registry.uds.on/LARN/ryu:650b41e
  args: --relaying=True stp
links:
  - minion: n1.uds.on,
      interfaces:
        eth0:
          bandwidth: 10Mbps,
          delay: 20ms
server:
  minion: h2.uds.on
  image: registry.uds.on/LARN/rtt:v0.7
  args: --server=True
  ip: 10.5.1.21/24,
  mac: 'AB:CD:EF:01:23:67',
  port: 8081
client: ...
```

NEAT: In Action



NEAT: In Action



Configuration Management (CM)

Goal: Bring the experimentation system to a **well-defined state**.

Goal: Bring the experimentation system to a **well-defined state**.

Benefits

- Left-overs are avoided (misconfigured links, ...).
- Many tools use configuration files that can be checked in (§1).

Goal: Bring the experimentation system to a **well-defined state**.

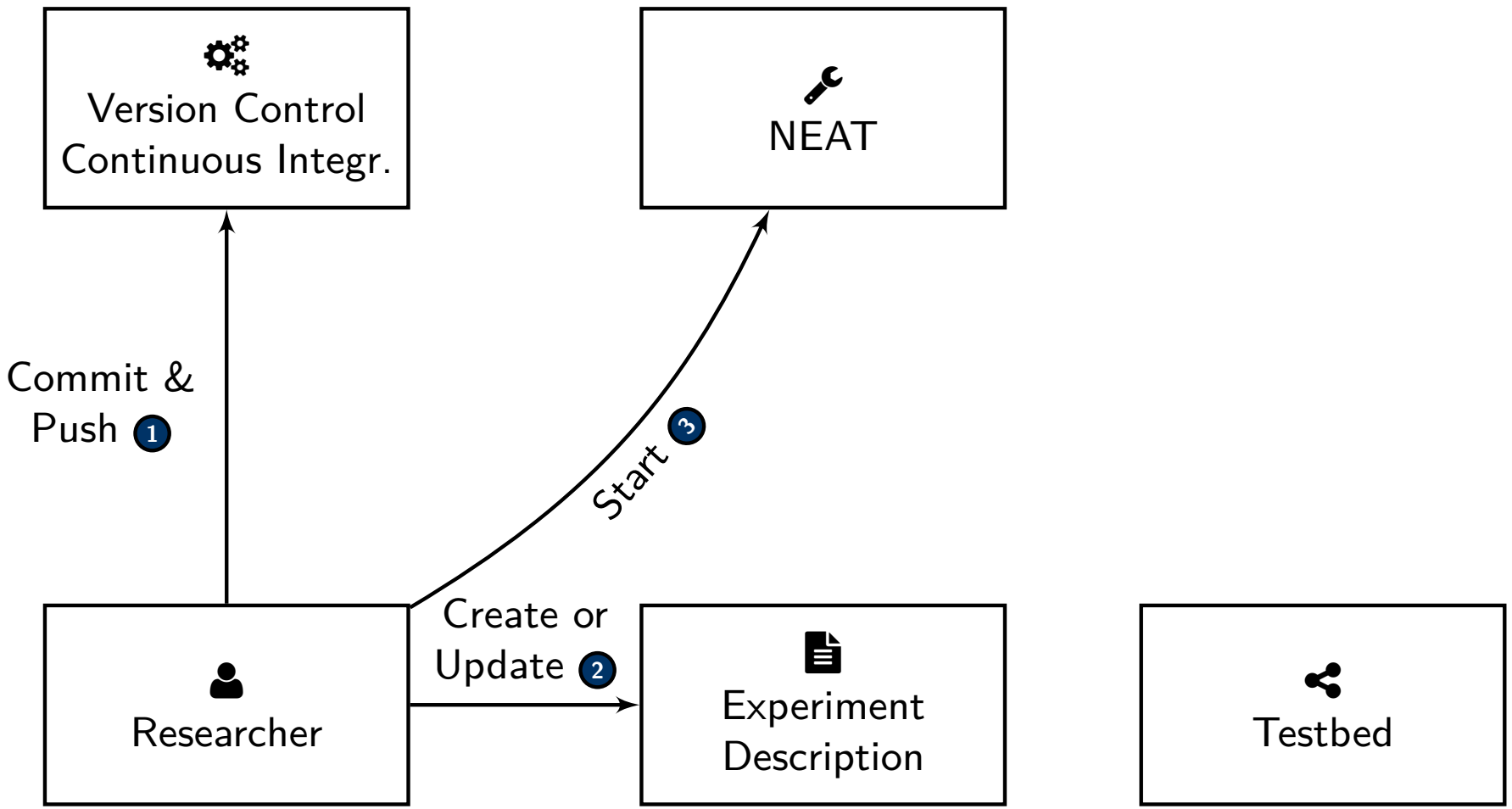
Benefits

- Left-overs are avoided (misconfigured links, ...).
- Many tools use configuration files that can be checked in (§1).

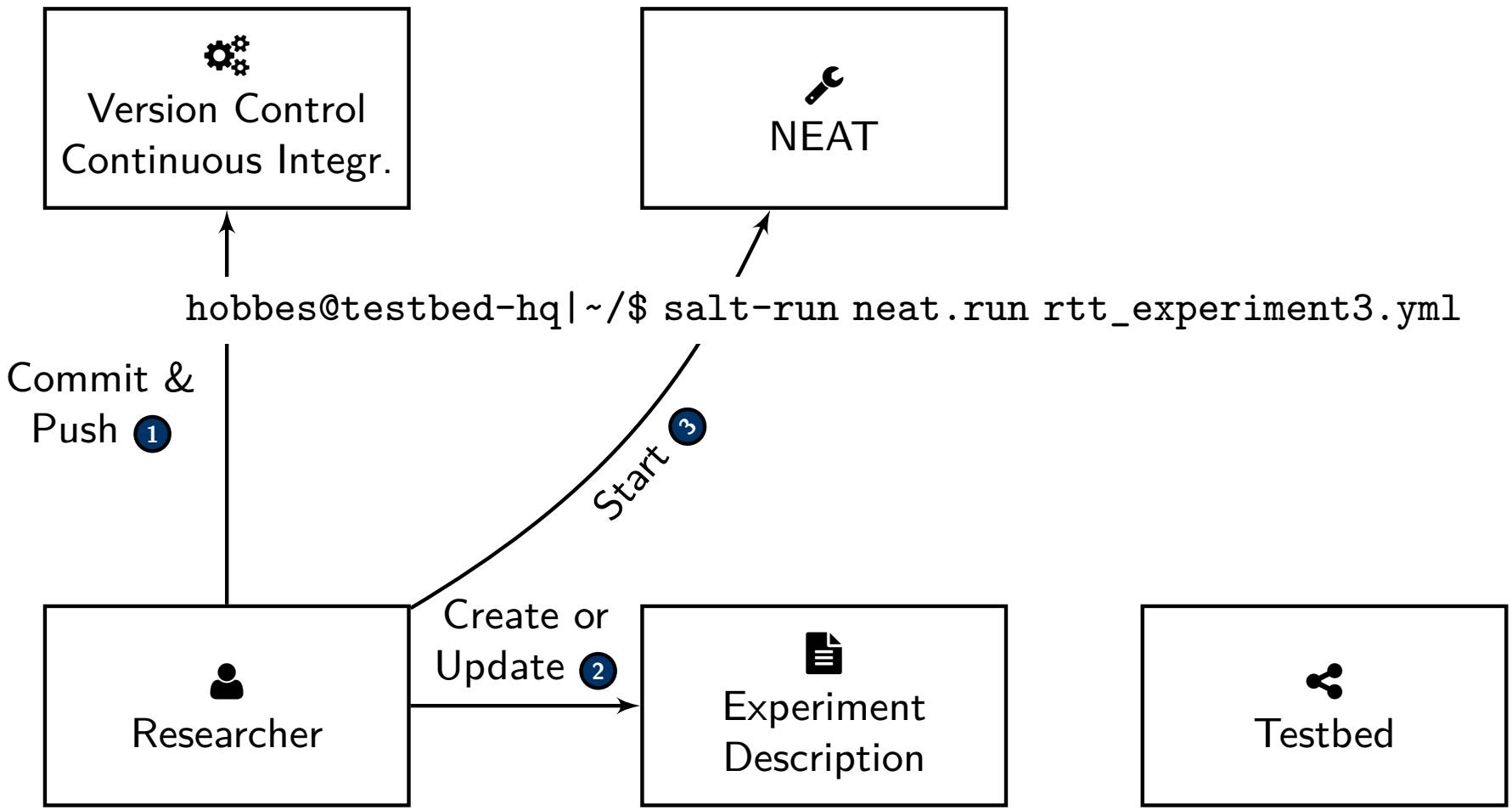
Software Solutions

- **SaltStack**: very consistent, good introspection
- Puppet: model- not code-driven, complex definitions
- Chef: no push, configurations in code (Ruby)
- Ansible: ssh-based, simple, inconsistent formats
- ...

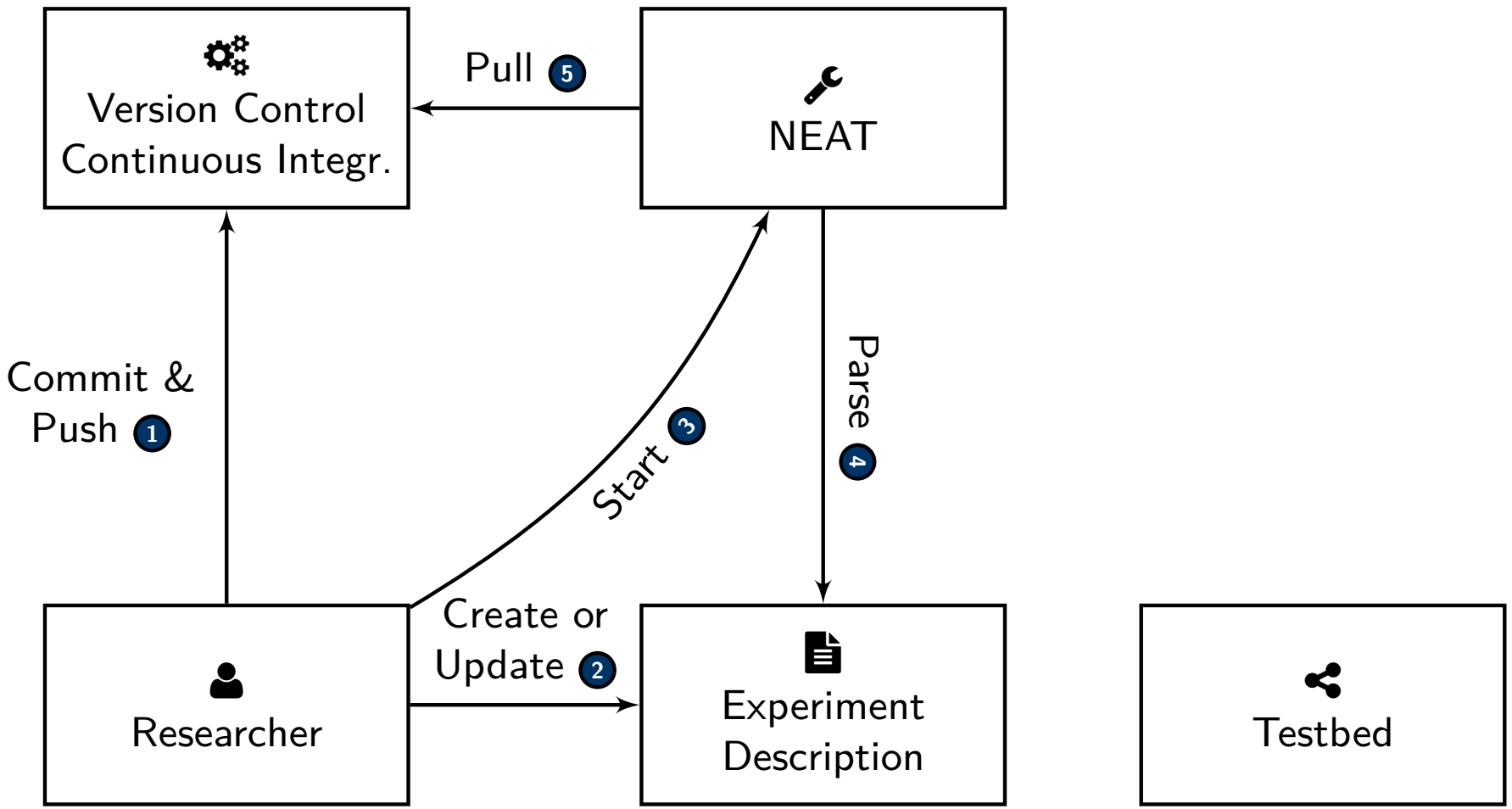
NEAT: In Action



NEAT: In Action



NEAT: In Action



Goal: **Do not hide** networking behaviour in proprietary hard-/software.

Goal: **Do not hide** networking behaviour in proprietary hard-/software.

Benefits

- Network behaviour is transparently defined by the SDN/NFV applications (§1).
- Network code is under version control (§3,4).

Goal: **Do not hide** networking behaviour in proprietary hard-/software.

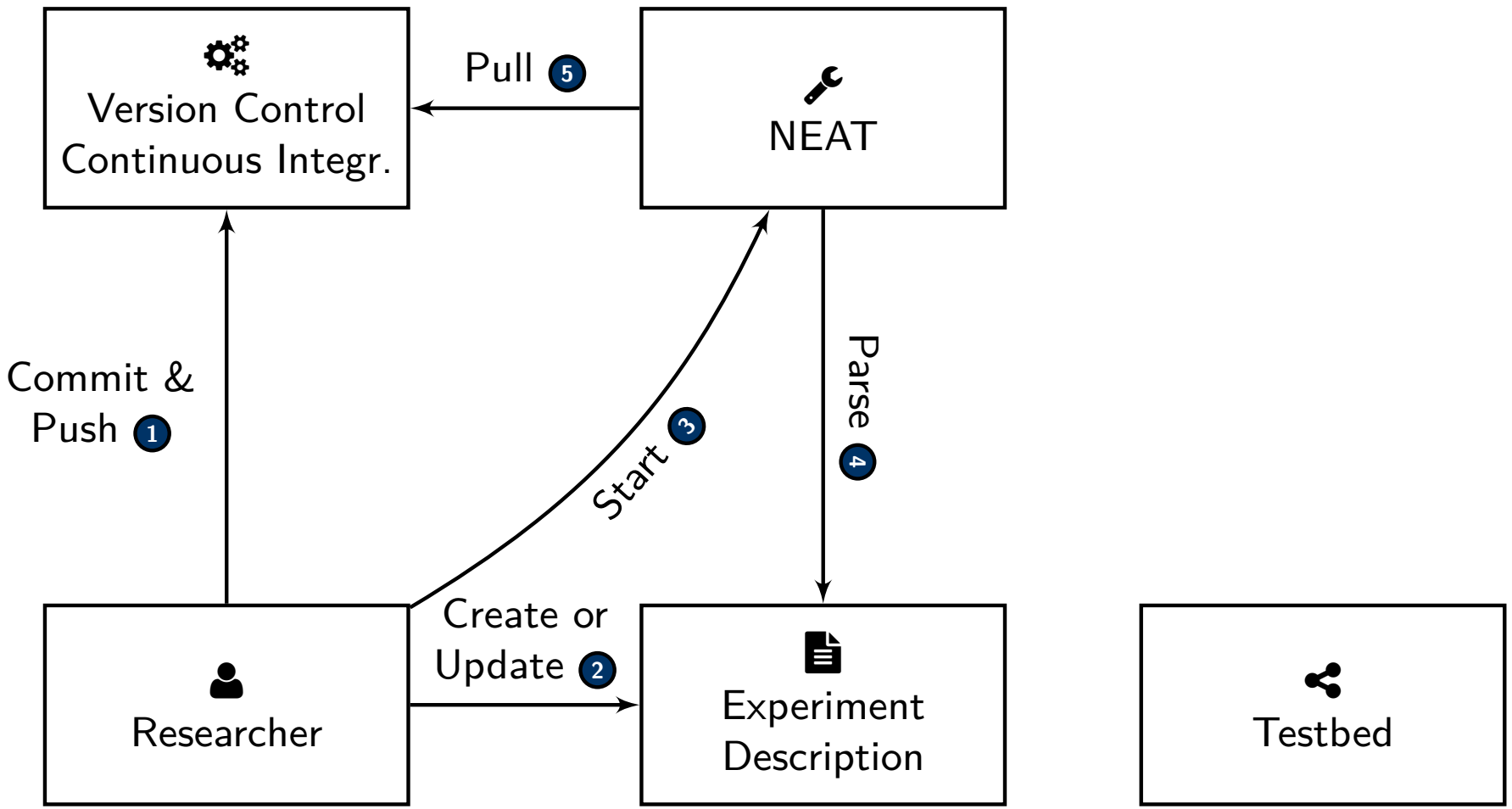
Benefits

- Network behaviour is transparently defined by the SDN/NFV applications (§1).
- Network code is under version control (§3,4).

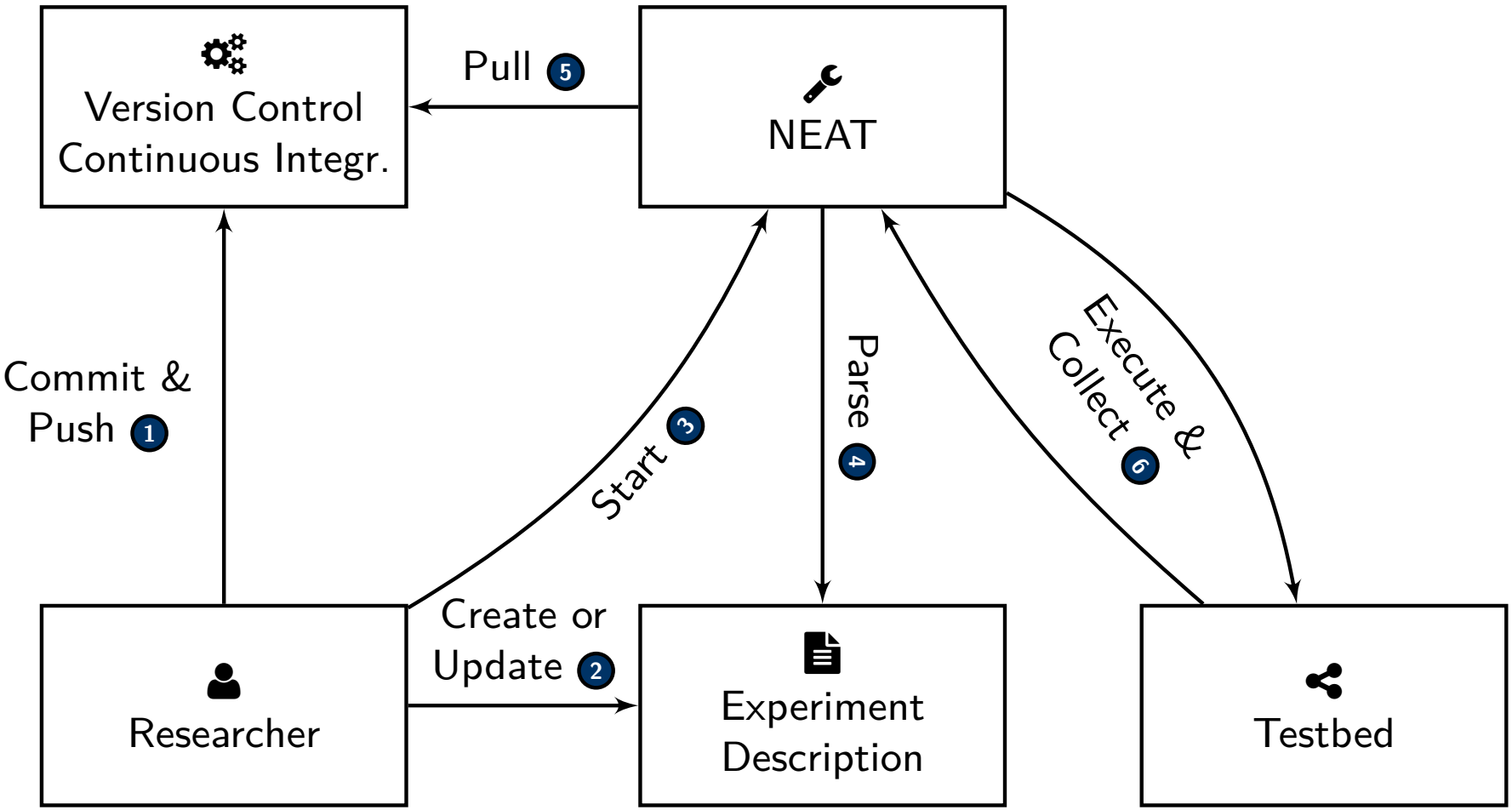
Deployment Solutions

- **Docker Containers**: specific software, and libraries in one confined image.
- **SaltStack States**: no virtualization, installed on the system.
- Virtual Machines: high system emulation overhead, highest flexibility.
- ...

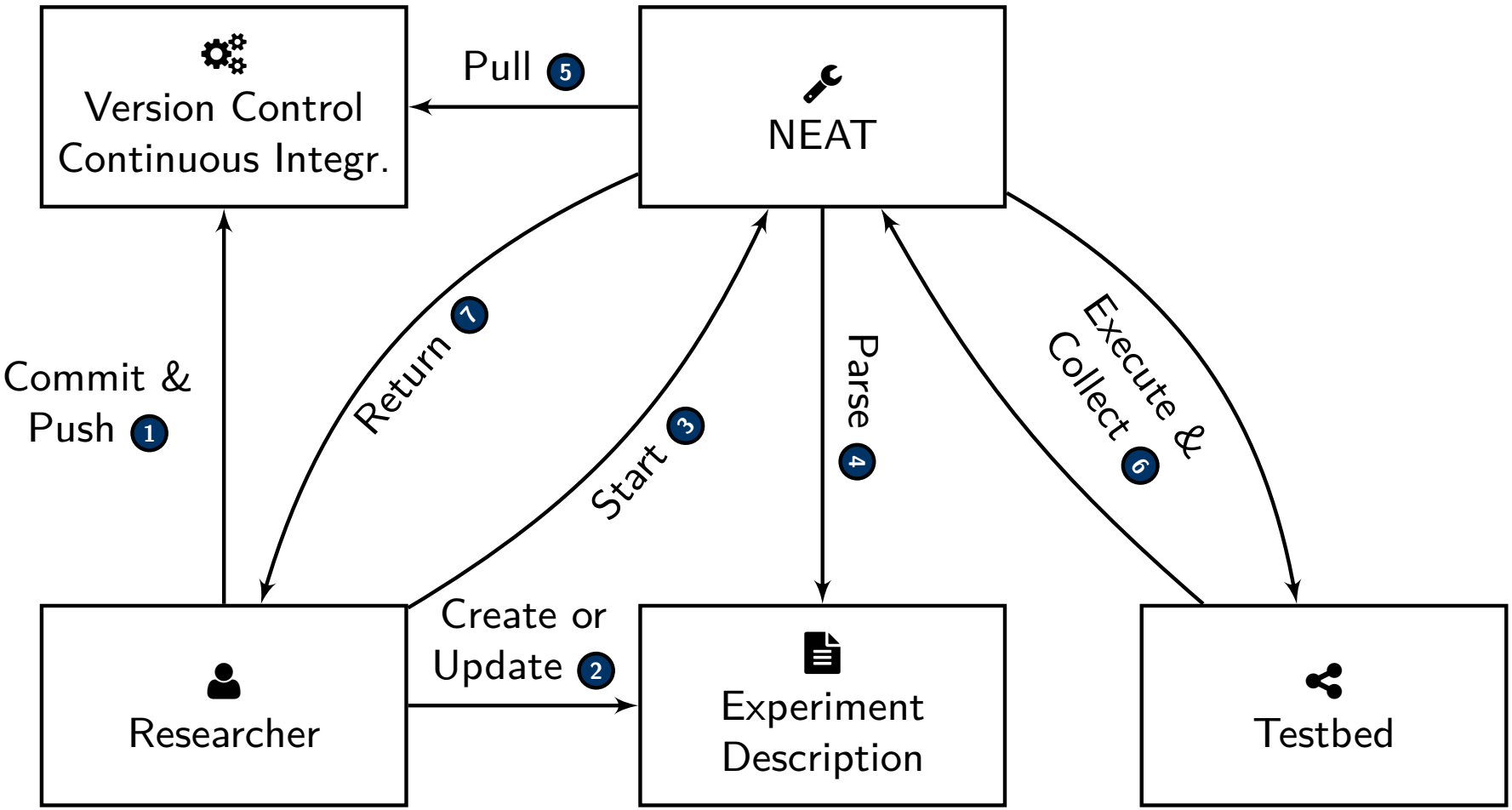
NEAT: In Action



NEAT: In Action



NEAT: In Action



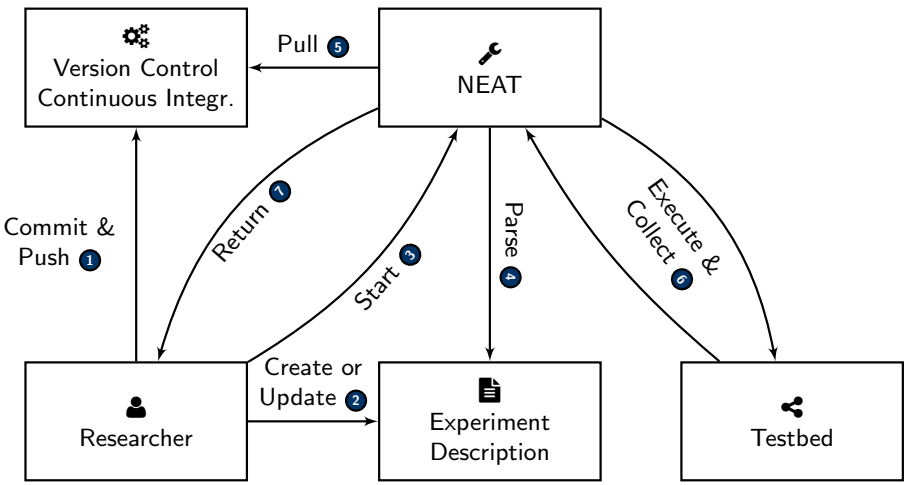
NEAT: Evaluation

- **hobbes@testbed-hq|~/$** `salt-run neat.run rtt_experiment3.yml`
  ```
  Experiment took 86.4s.
  Results are in rtt_experiment3_20171012_171829.xz.
  ```

NEAT: Evaluation

- **hobbes@testbed-hq|~/$** salt-run neat.run rtt_experiment3.yml
  Experiment took 86.4s.
  Results are in rtt_experiment3_20171012_171829.xz.
- **hobbes@testbed-hq|~/$** xz -l rtt_experiment3_20171012_171829.xz
  client.csv
  client.pcap
  rtt_experiment3.yml
  neat_log.txt
  controller.log
  server.csv
  server.pcap

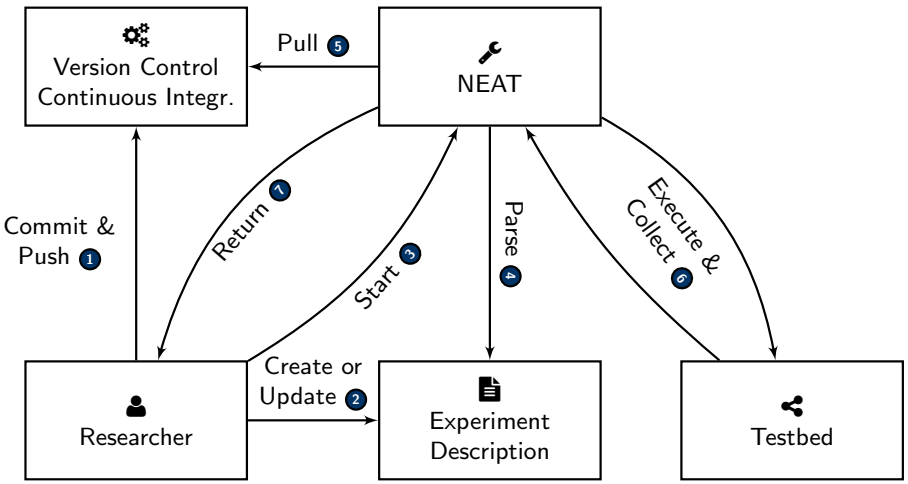
Conclusion



Summary

- Network experiments can be made more **reliable**, **automated** and **reproducible**.
- Using **open source** technologies, experiments can be thoroughly **defined** and **executed**.
- **NEAT** is our first prototype to implement such a **network experiment automation**.
- The code is available at `http://neat.larn.systems`.

Conclusion



Summary

- Network experiments can be made more **reliable**, **automated** and **reproducible**.
- Using **open source** technologies, experiments can be thoroughly **defined** and **executed**.
- **NEAT** is our first prototype to implement such a **network experiment automation**.
- The code is available at `http://neat.larn.systems`.

Thank you for your attention. Questions?

References

[ACM2016] ACM "Result and artifact review and badging." https://www.acm.org/publications/policies/artifact-review-badging. Accessed: 2017-07-04.

[Sandve2013] G. K. Sandve, A. Nekrutenko, J. Taylor, and E. Hovig, "Ten Simple Rules for Reproducible Computational Research" PLoS Computational Biology, vol. 9, no. 10, pp. 1–4, 2013.

[Docker] https://docker.io/

[GitLab] https://about.gitlab.com/

[SaltStack] https://saltstack.com/

[YAML] http://www.yaml.org/start.html