# Recherche Reproductible

## Module 3: Workflows

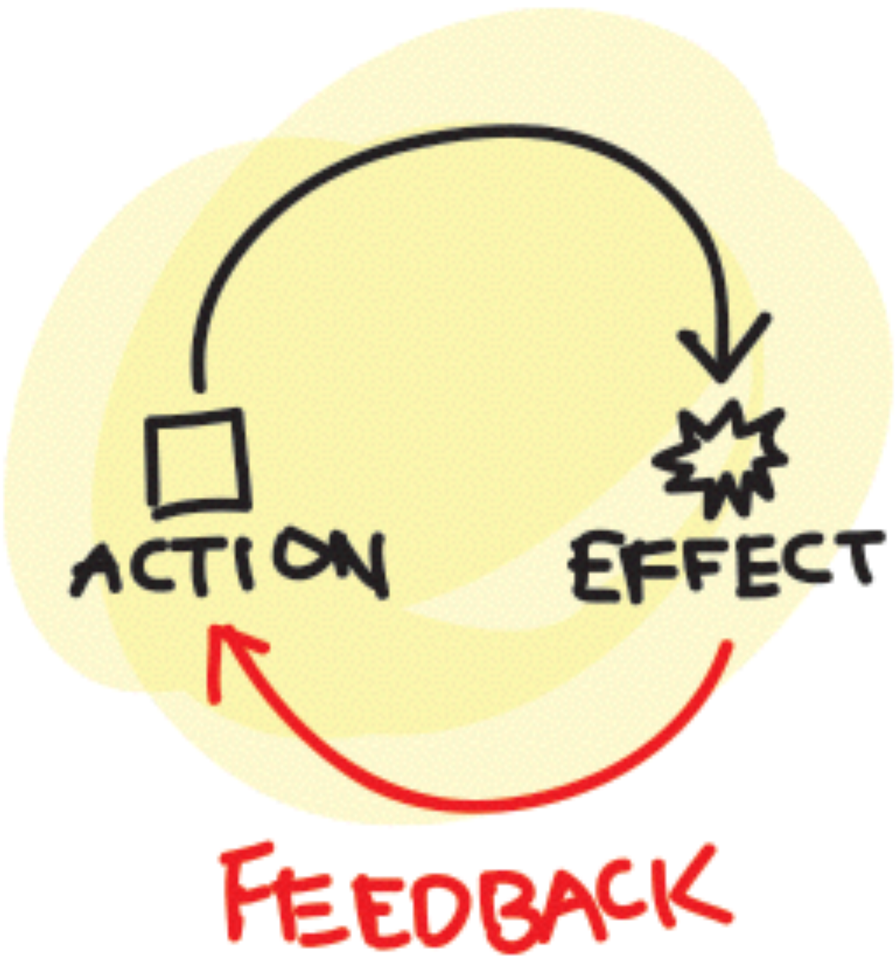
### Continuous Integration 101

by Guille Polito
@GuillePolito

CRIStAL Centre de Recherche en Informatique, Signal et Automatique de Lille

Université de Lille 1 SCIENCES ET TECHNOLOGIES

# Feedback loops

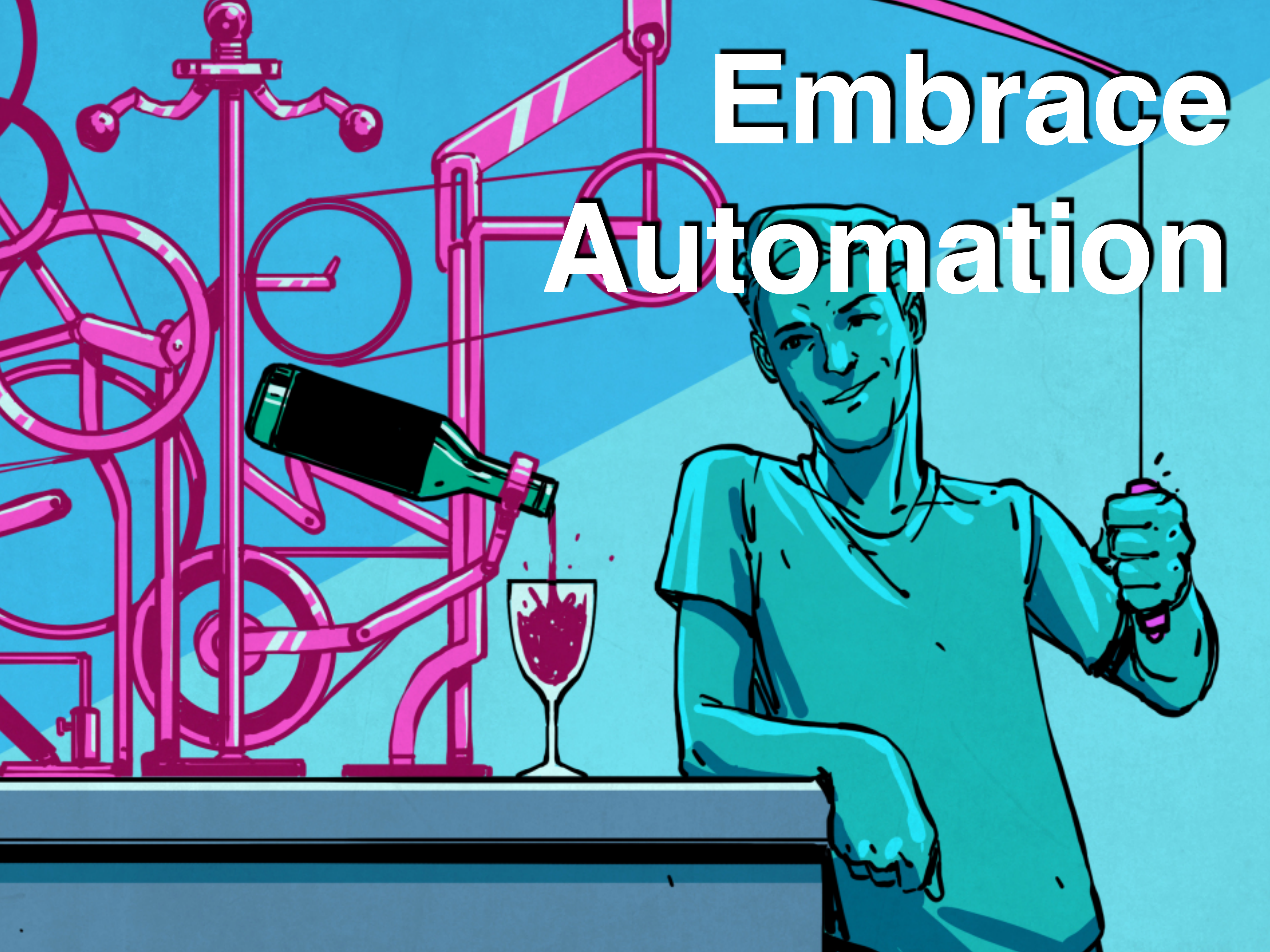


CONTINUOUS
IMPROVEMENT

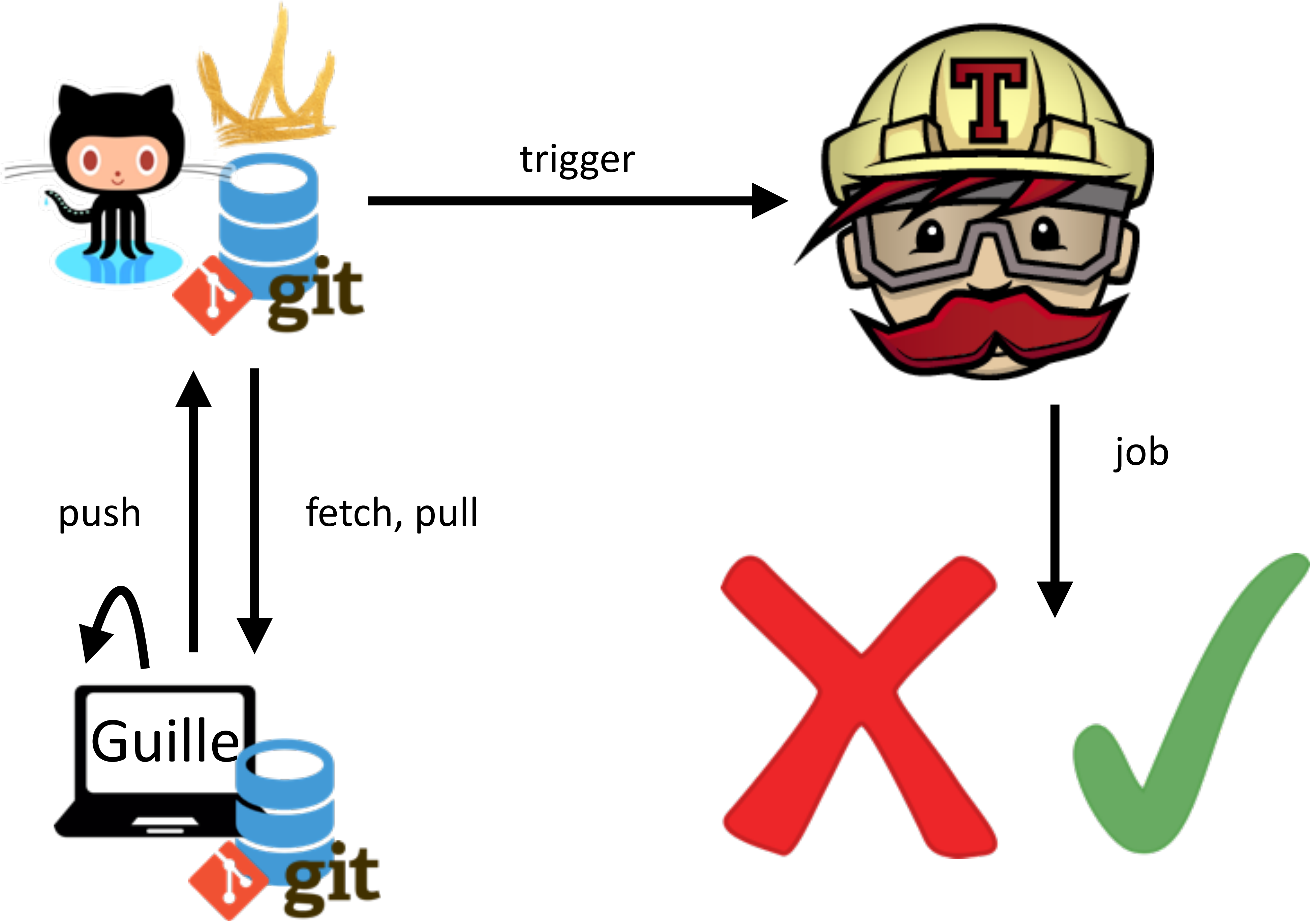
# Continuous Integration

- Get feedback fast!
  - Is it working?
  - How is its quality?
  - Does it follow the project rules?

# Embrace Automation



# CI dissected



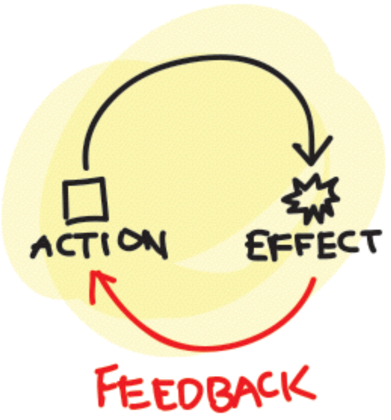
trigger

push

fetch, pull

Guille

job



# Demo 1:
# Automated Testing

https://travis-ci.org/guillep/chasqui

# Demo 2:
# In Multiple platforms

https://travis-ci.org/pharo-rdbms/garage

# Demo 3:
# Jenkins Pipelines

https://ci.inria.fr/pharo-ci-jenkins2/job/Test pending pull request and branch Pipeline/

# Demo 4:
# Building Documentation

https://travis-ci.org/SquareBracketAssociates/Booklet-AReflectiveKernel

# Demo 5: Validating Documentation

https://travis-ci.org/CRIStAL-PADR/python-paper-template

# Demo 6: Scientific Papers

https://travis-ci.org/CRIStAL-Sigma/latex-travis-test

# Other possible usages

- Continuous benchmarking
  - monitor performance improvements;
  - detect regressions
- Code quality rules
- Generate code
- Continuous delivery and deployment

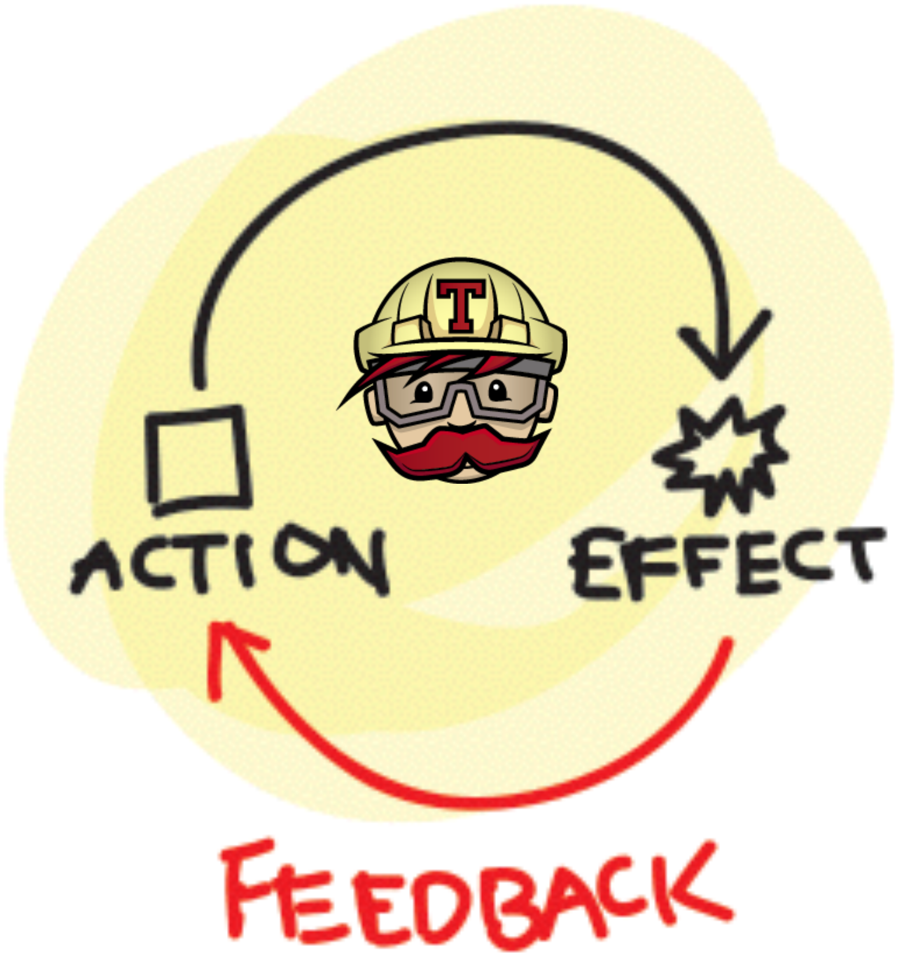
# When to use CI?

- If you have common manual tasks
  - e.g., compilation, testing, building archives for deployment…

- If you don’t have them, automate them
  - i.e., write tests, make your system reproducible!

- Travis
  - OSX is slow
  - For windows we would need appveyor…
  - Less freedom to design our process
- Jenkins: we have full control (and responsibility) on it

# Conclusion



productivity