# Maintainability, Sustainability, Testing and Deployment of QE

Filippo Spiga[1,2] < filippo.spiga@quantum-espresso.org >

[1] Head of Research Software Engineering, Univ. of Cambridge
[2] Quantum ESPRESSO Foundation

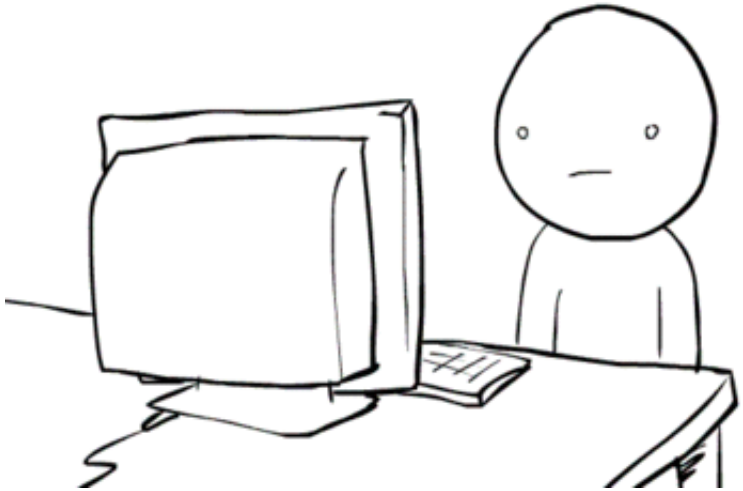

«*What I cannot compute, I do not understand.*» (adapted from Richard P. Feynman)

# Elevator pitch: Research Software Engineering

*"Software is a cornerstone of science. Without software, twenty-first century science would be impossible. Without better software, science cannot progress."*
-- SCIENCE CODE MANIFESTO

**BETTER SOFTWARE**
**BETTER RESEARCH**

# This happens… lot of times!



# Because everybody …

# The (invalid) argument


*Why bother about all of this, we always did without those and it was good enough…*


# WRONG

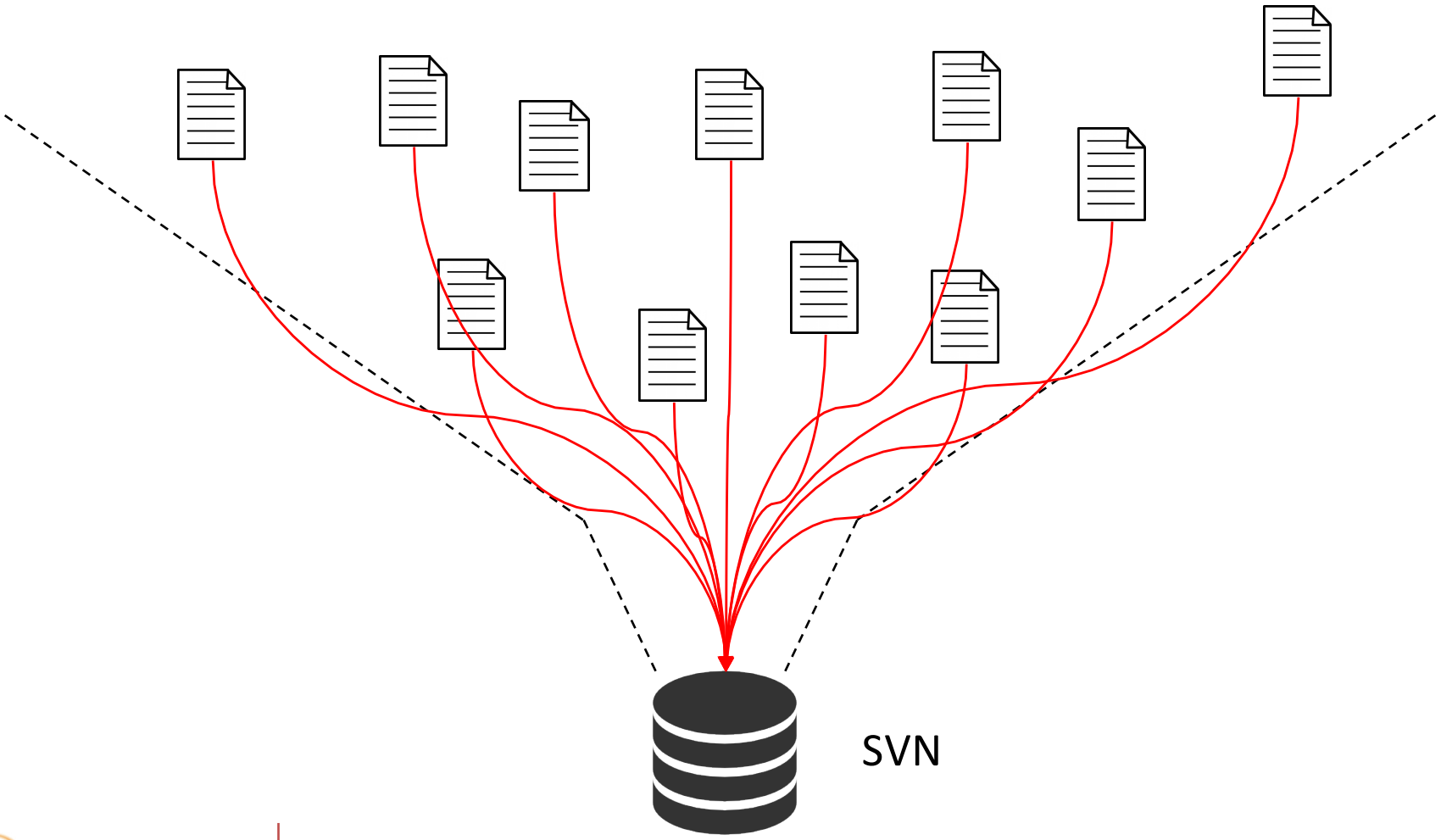

# 3 principles

- Versioning (SVN → SVN+GIT)
- Continuous Integration (BuildBot)
- Testing (test-code)

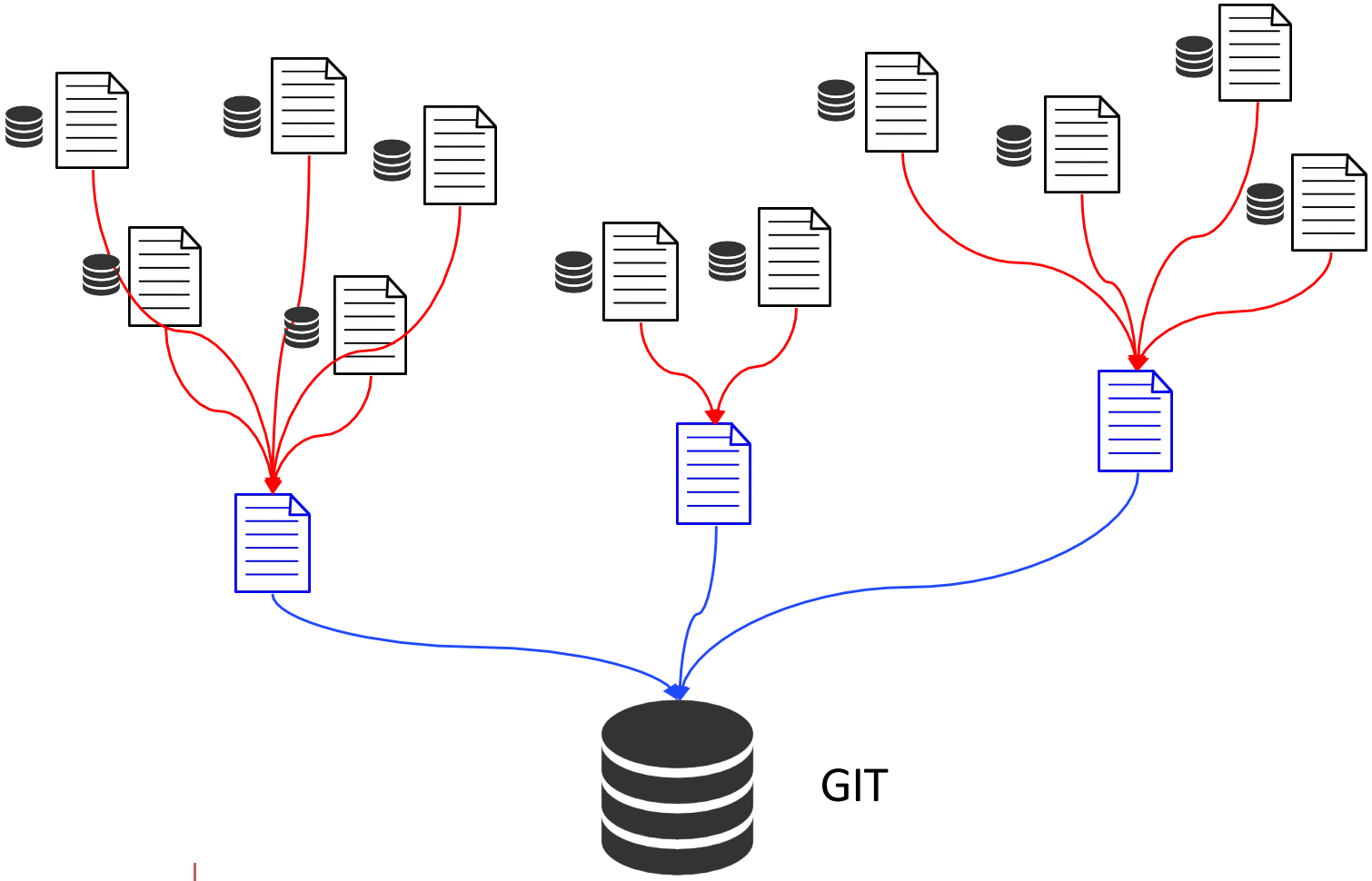
… applied to **Quantum ESPRESSO**.

Demo included, references included.

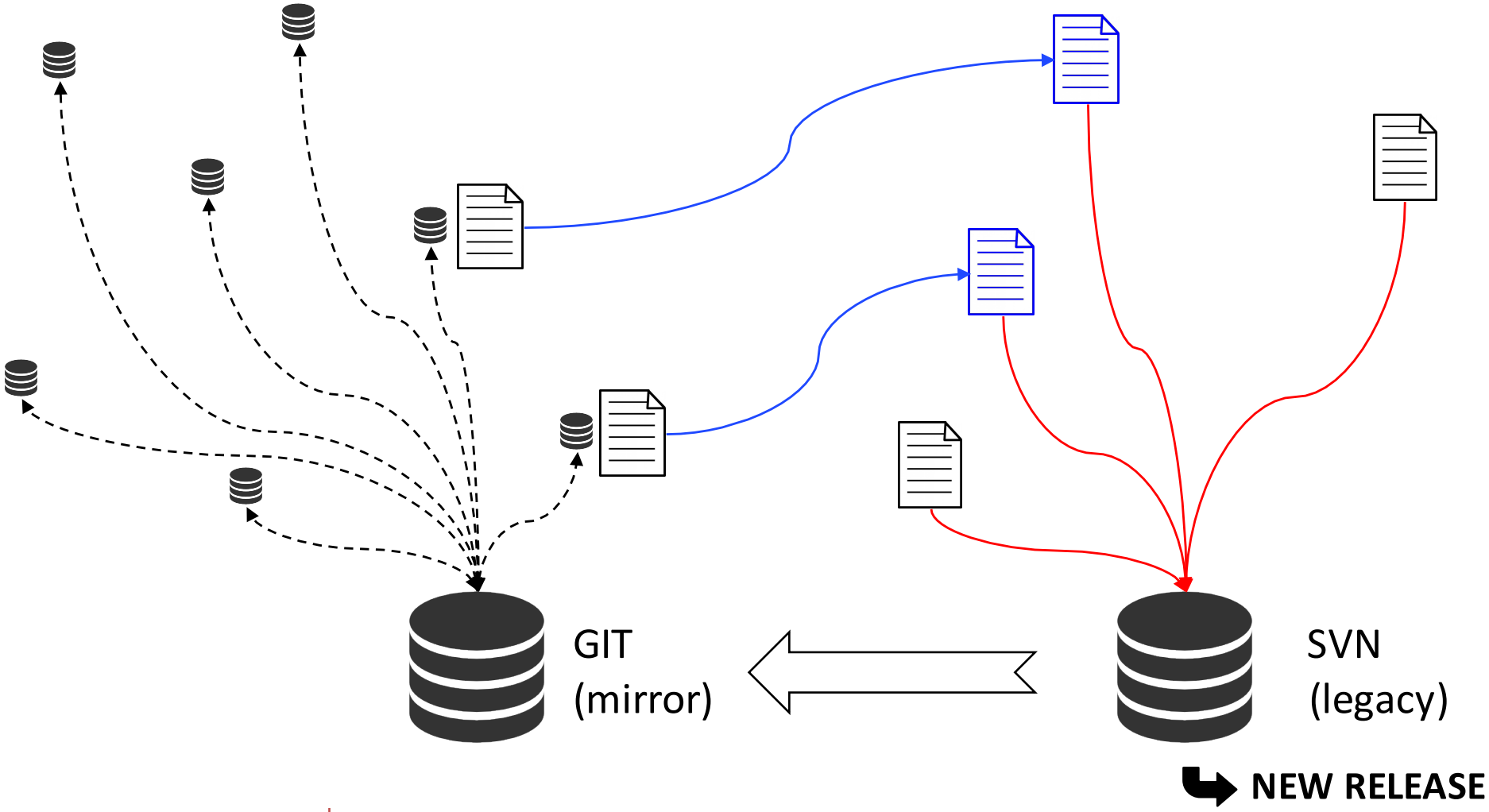
# Versioning (classic)



# Versioning (ideal)

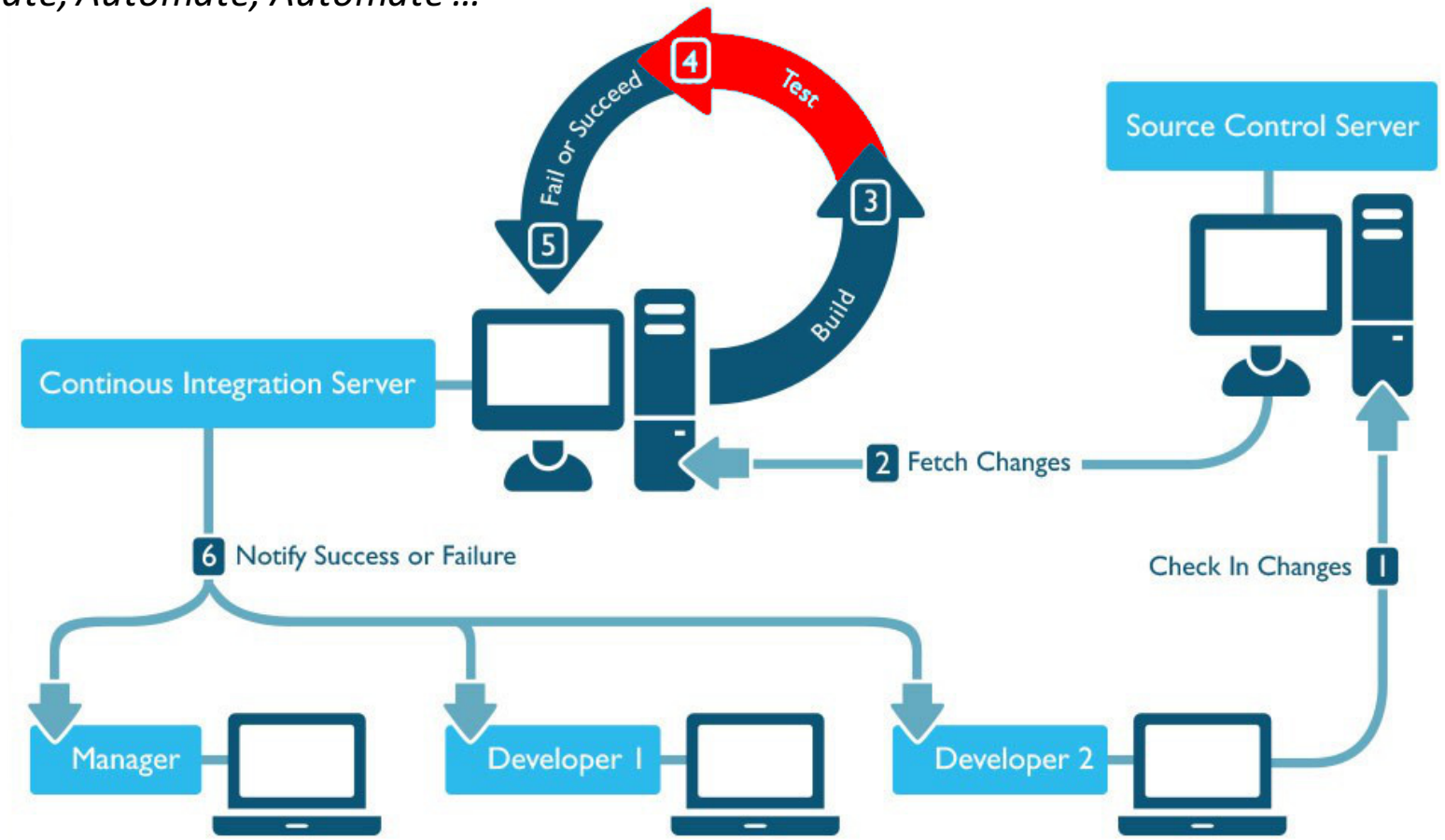


# Versioning (sustainable)



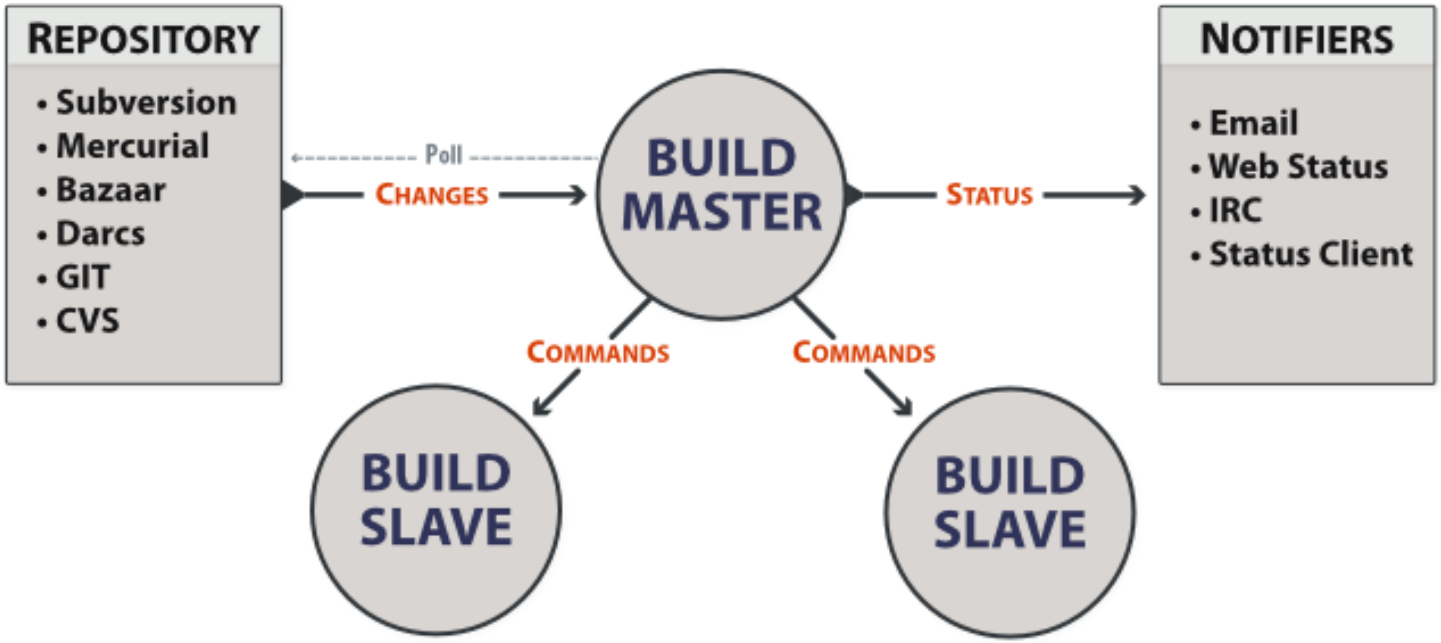
# Continuous Integration (C-I)

*Automate, Automate, Automate …*

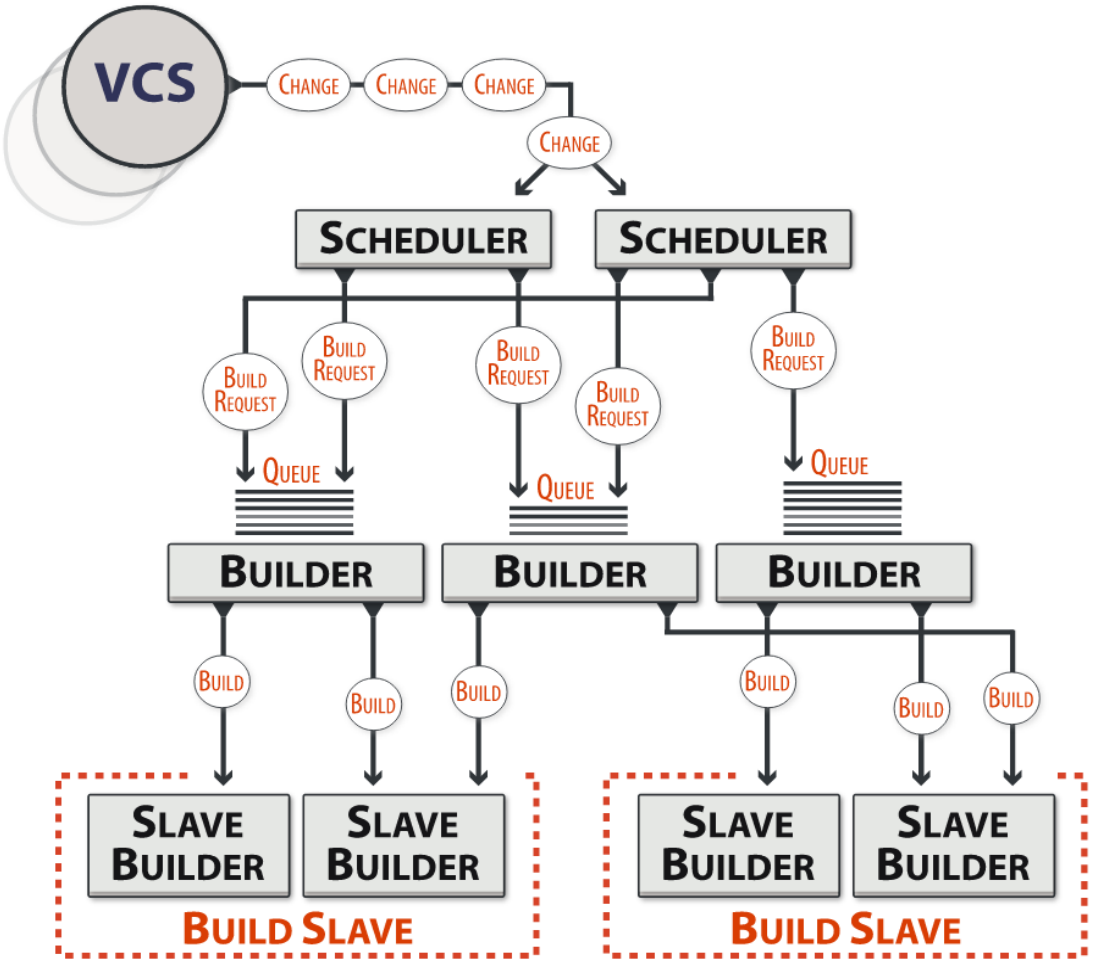


# BuildBot

Written in Python, very simple!



# BuildBot




*What buildbot does for me…*

- Upon branch update or at specific time interval, a **build** is created and test suite is run
- Developer is alerted (via UI or email) when a test fails, can submit fix, and re-launch test (even manually)
- Upload test results or compiled applications to an external server

# Deploy your BuildBot (slave)

```
mkdir -p $HOME/my_buildbot_slave
cd $HOME/my_buildbot_slave

virtualenv --no-site-packages buildbot_sandbox
source buildbot_sandbox/bin/activate

buildslave create-slave slave \
       <my-public-IP>:9989 my_slave_1 <password>
buildbot start my_slave_1
```

# DEMO

# test-code

- Project initiated by James Spencer (ICL) for testing looking at regression errors in scientific software.
- It runs a set of calculations, and compares the output data to that generated by a previous calculation (which is regarded to be "correct").
- Written in python, designed to be lightweight and highly portable.
- It can run a set of tests and check the calculated data is within a the desired tolerance of results contained in previous output (*data extraction* features)
- The programs to be tested can be run in serial and in parallel and tests can be run in either locally or submitted to a compute cluster.

# test-code - capabilities

- **compare** = compare set of test outputs from a previous testcode run against the benchmark outputs.
- **diff** = diff set of test outputs from a previous testcode run against the benchmark outputs.
- **make-benchmarks** = create a new set of benchmarks and update the *userconfig* file with the new benchmark id. Also runs the 'run' action.
- **recheck** = compare set of test outputs from a previous testcode run against benchmark outputs and rerun any failed tests.
- **run** = run a set of tests and compare against the benchmark outputs.
- **tidy** = remove files from previous testcode runs from the test directories.

# test-code - configuration

Two configuration files:

- **jobconfig** defines the tests to run
- **userconfig** defines a program to be tested

# DEMO

# Production Roadmap

Phase 1 -- April 2015, v5.3.1:

- QE-FORGE Continuous Integration goes live as service
- All “slaves” welcome (but we value diversity)
- Development branches covered if pushed on the QE-FORGE

(Possible) Phase 2 -- September2015, v5.x.x:

- Regular (how regular?) nightly binaries
- More packages within suite (PH, TDDFPT, …)
- More complex