Stefano Cozzini

CRS – OGS Udine – 27 Novembre 2019

Introduction to Continuous Integration

Agenda

- Introducing the need of continuous integration approach
- Practices of Continuous Integration
- Example/Demo for GEOTOP code

The challenge

- As the project grows, complexity grows:
  - Physical code size
  - Dependencies on libraries
  - Number of developers
  - Package versions
  - Different computer architectures
  - Different compilers

  ...etc..

The challenge

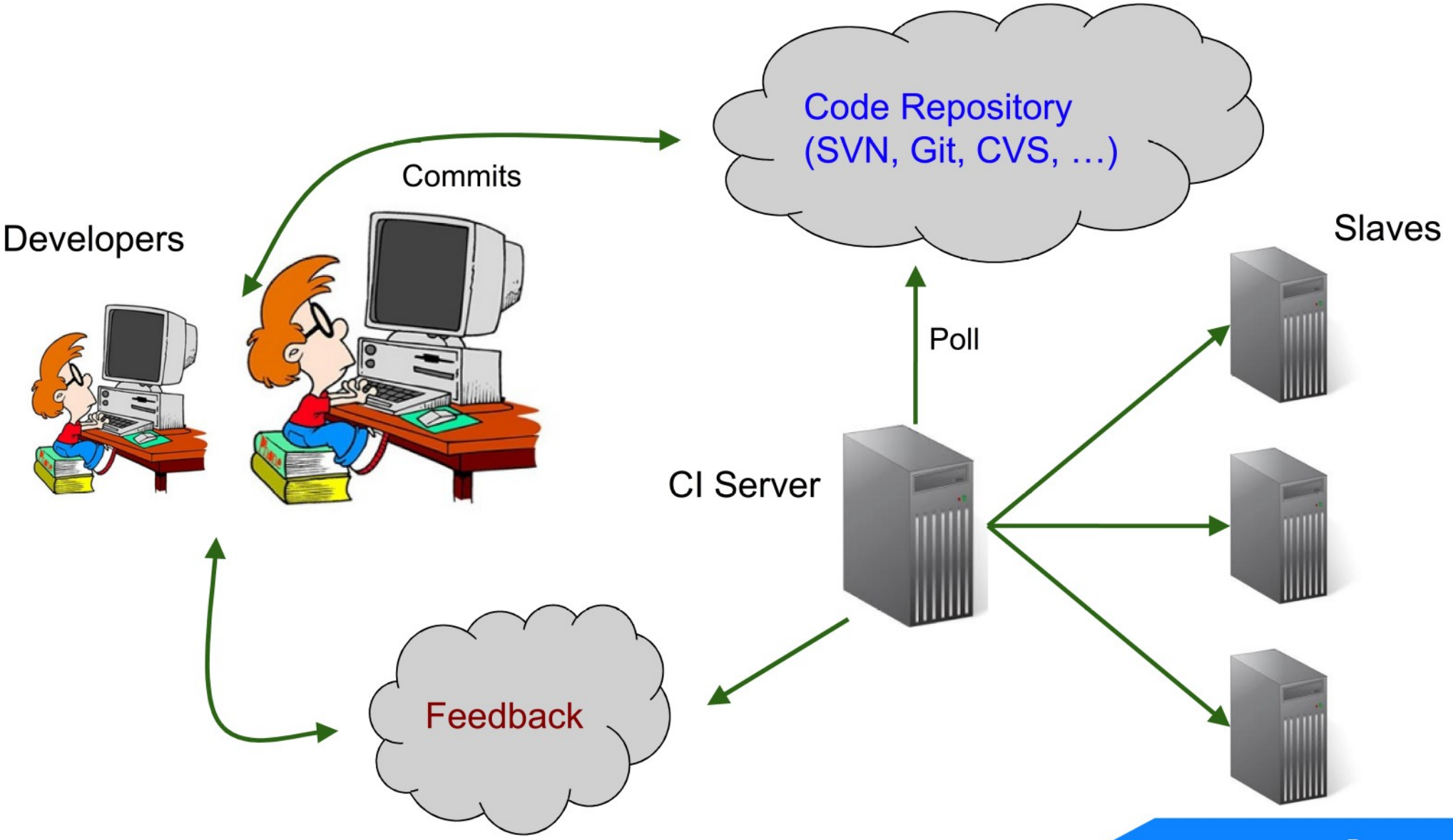
- How do we handle increasing code-base sizes?
- How do we handle an increasing number of developers?
- How can developers interact with each other?
- How do we build across multiple platforms?
- How do we build multiple versions?
- How can we make sure we don't break things!

Continuous Integration (CI) - What?

“. . . a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily—leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible.”

- Martin Fowler

Components;


Code Repository
(SVN, Git, CVS, …)
Commits
Developers
Slaves
Poll
CI Server
Feedback

CI - Why?

- Early/rapid feedback!
  - Do all components/projects compile?
  - Coding standards?
  - Are tests successful?
  - Performance requirements?
  - Problems archiving or deploying?
- Better project visibility
  - Possible to notice trends
  - What features are needed/being added

Why? (cntd)

- Insures clean environments
- Manual tasks automated
  - Speedup of working software turnover
- No large integration steps
  - Much less likely to break something
- A full working/deployable version at ANY POINT IN TIME
- Complete documentation of who did what

CI - How?

In order for CI to work, individual developers should:

- Commit frequently
  - Many small commits
- Run local build first (if possible)
  - Huge code repos may make this difficult
  - Only commit working code
- Fix broken builds immediately
- Write automated tests

CI - General Concepts

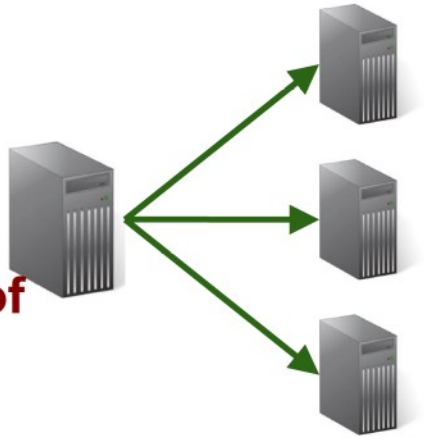
- Hopefully you're convinced that CI is useful
- What do we need to CI?
  - Hardware
    - CI Server + CI Slaves
  - Software
    - What do we build?
    - When do we build?
    - How do we build?
    - How do we test?
    - How do we get feedback?

CI General Concepts: Master - slave

- Everything on 1 server
  - Possible, not ideal unless for special projects
- Distributed builds
  - faster, safer, scalable, multiple platforms, more expensive

CI Server:

- **Poll version control repo**
- **Perform actions on scheduled basis**
- **Send emails**
- **Host web accessible dashboard**
- **Store and display history of builds**



CI Slaves:

- **Pure workhorse**
- **Various architectures**
- **Reduces build time**
- **Communication through built in ssh-client (or similar)**
- **Receives action from server**
- **Carries out action**

CI - Who?

Who should CI?

- Your code that builds your thesis?
  - Probably not
- A project with more than 2 developers?
  - Yes
- An open source project:
  - Mandatory !

CI General Concepts: Testing

- Absolutely crucial for CI!
- Several forms of testing:
- UnitTests
  - First step of testing
  - Can be run on all commits
  - During build
  - Or after build
- Integration tests
  - Does my new code/changes break the rest of the build?
  - Probably after build

Testing (2)

- Custom tests
  - Often much more lengthy, performed less often
  - Software validation
    - Errors? Segfaults?
  - Physics validation
    - Correct output?
  - Possibly a dedicated slave
- All should be automized!
- Performance profiling is also important

Continous Integration

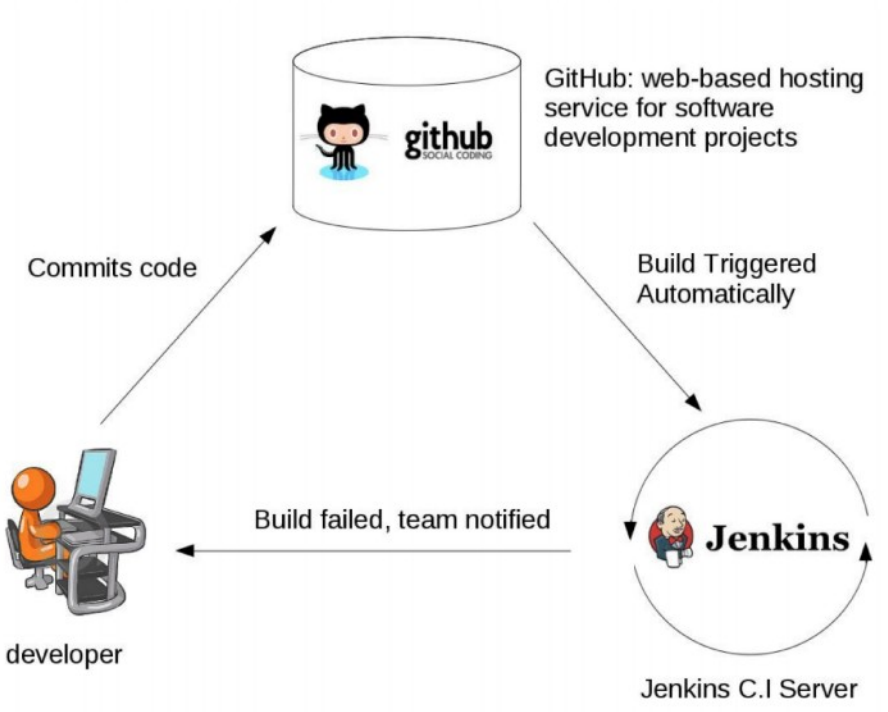
- To practice CI, every developer on a project must integrate their work daily with every other developer
- Everyone on the team needs to practice this for it to work
- Continuous integration is not a tool, but a technique
- But there are many open source tools to help practice CI

Continuos Integration tools/ services

- Jenkins
- Travis CI (integrated with github)
- Buildbot
- And many others..

Continuous Integration at work:

- Developers commit code to a shared repository on a regular basis.
- Version control system is being monitored.
- When a commit is detected, a build will be triggered automatically.
- If the build is not green, developers will be notified immediately.



```
Subject: buildbot failure in ICTP BuildBot on
regcm-conf=debug|mod=openmpi:1.4.3:intel:2011
Date: Tue, 21 Feb 2012 22:20:39 +0100
From: buildmaster@gforge.ictp.it
To: amessina@ictp.it, ggiulian@ictp.it
The Buildbot has detected a failed build on builder
regcm-conf=debug|mod=openmpi:1.4.3:intel:2011 while building ICTP
BuildBot.
Full details are available at:
 http://buildbot.ictp.it/builders/regcm-conf%3Ddebug%7Cmod%3Dopenmpi
%3A1.4.3%3Aintel%3A2011/builds/8
Buildbot URL: http://buildbot.ictp.it/
Buildslave for this Build: argo-buildbot01
Build Reason: The web-page 'force build' button was pressed by '':
Build Source Stamp: 2750
Blamelist:
BUILD FAILED: failed compile
sincerely,
 -The Buildbot
```

Continuous Integration is also a mindset

- Fixing broken builds should be treated as a high priority issue for all team members.
- The deployment process should be automated, with no manual steps involved.

  All team members should focus on contributing to high-quality tests because the confidentiality of the CI process highly depends on the quality of the tests.

Continuos integration

- Enhance portability:
  - Different Build slaves for different HW/SW architectures
- Enhance/check numerical stability:
  - Different build slaves for different compiler options
- Enhance your own ”...ility”

People thought...

- Before
  - It can't work (here).
  - Doing it won't make much difference.
- After
  - Yes we do that – how could you live without it?

Practices of CI

- Maintain a single source repository
  - Put everything required for a build
- Automate the build
  - Build and launch in a single command
  - Need a virgin machine

Practices of CI (cont.)

- Make your build self-testing
  - Testing can catch a lot of bugs – enough to be useful
- Everyone commits every day
  - Conflicts that stay undetected for weeks can be very hard to resolve

Practices of CI (cont.)

- Every commit should build the mainline on an integration machine
  - Why: things still go wrong
    - Developers forget to run build before commit
    - Environmental differences
  - Manual build & CI system

Practices of CI (cont.)

- Keep the build fast
  - The whole point of CI is to provide rapid feedback
  - 10 minutes build
- Test in a clone of the production environment
  - Real tests without mocks

Practices of CI (cont.)

- Make it easy for anyone to get the latest executable
  - Nightly build
- Everyone can see what's happening
  - Travis CI demo
- Automate deployment
  - Also needs automated rollback

Benefits of CI

- Reduce risk
  - Integration is a long and unpredictable process
- There's no long integration anymore
- Easier to find bugs and remove them
  - Quality of your test suits

Travis CI for GEOTOP

- Continuous integration:
  - Test gcc compilation on the travis server
  - Test all unit tests prepared
  - Run some regression tests on any change…
  -