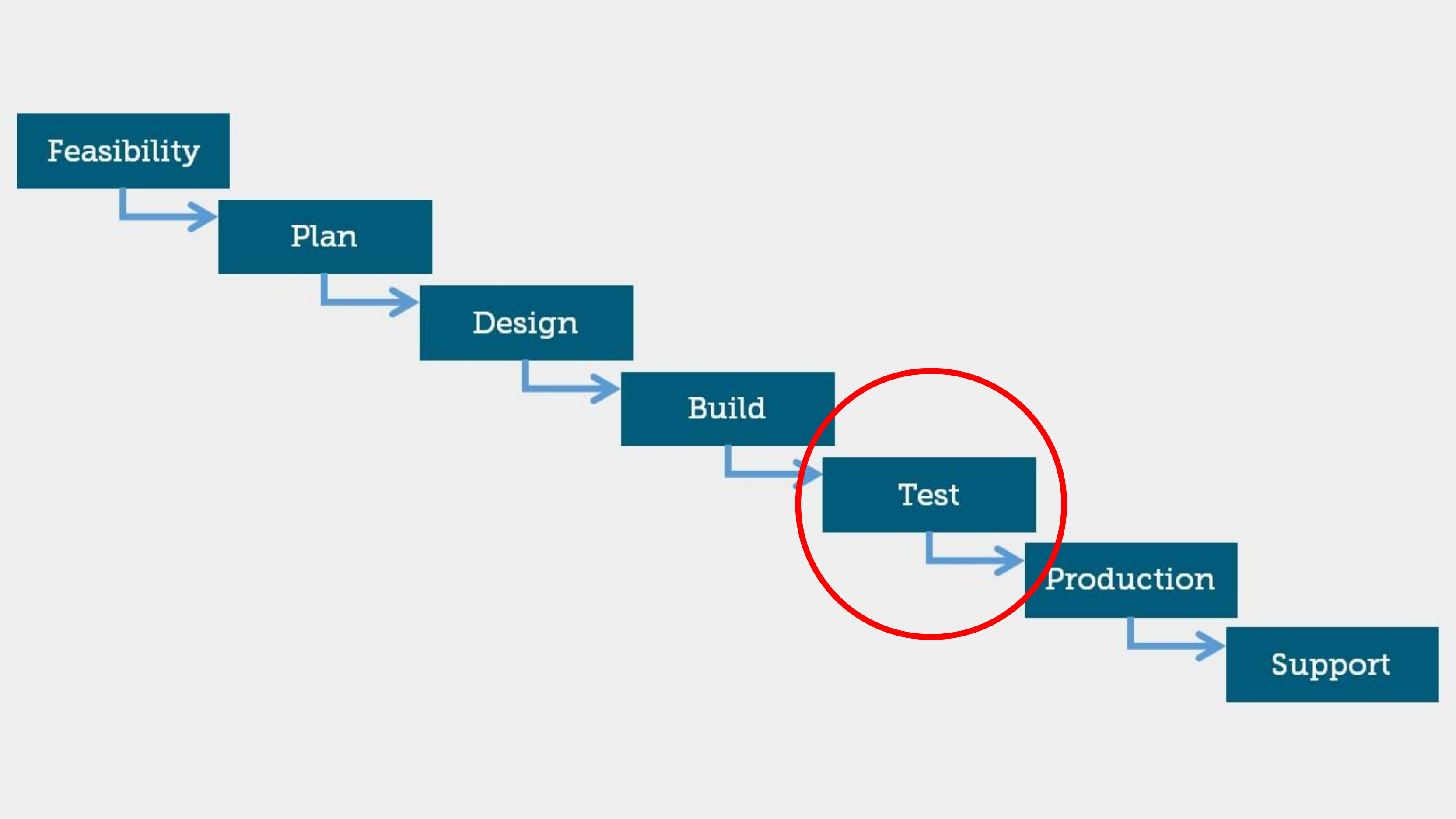
Following the Waterfall Methodology



Continuous Integration

Traditional “Waterfall” Testing Practices

Testing occurs as a project phase towards the end of project

- Lots of lead time for test planning, test case generation, test environment and infrastructure setup

Test cases don’t change often (tied to requirements)

- Cost of creating test cases is borne once
- Few changes to system once it is specified and designed

Tests are executed periodically

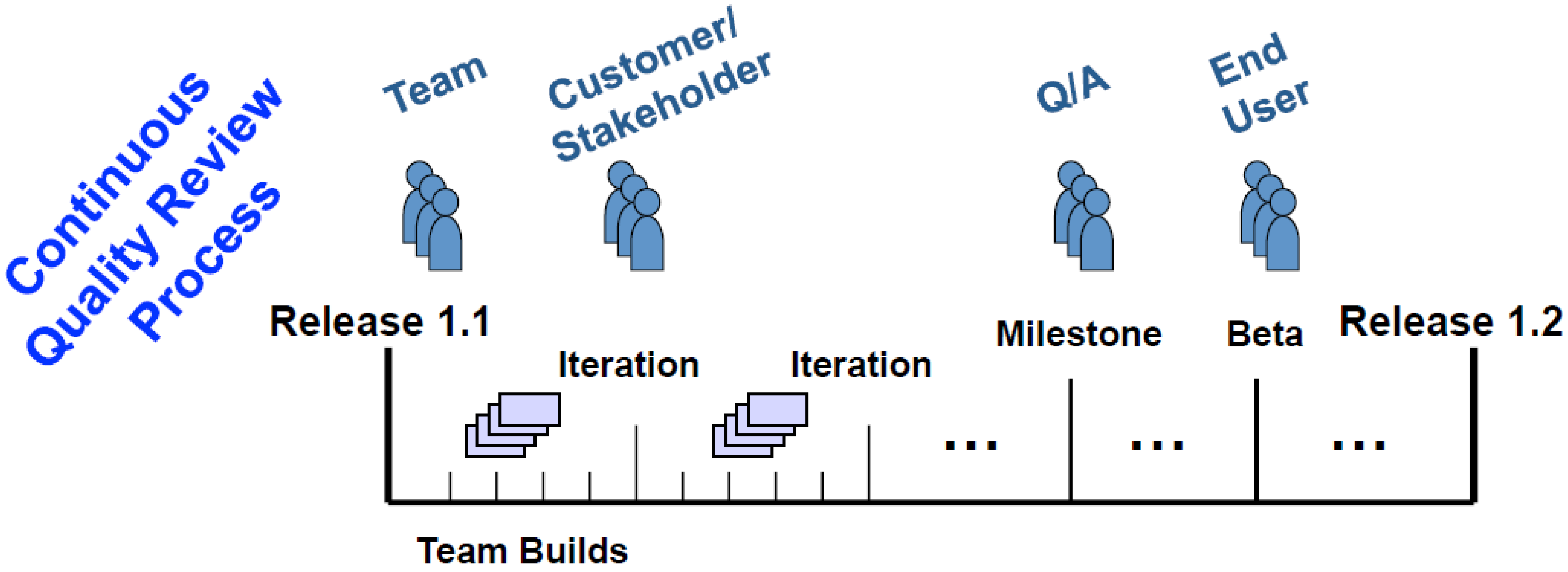
- Initial round of testing to ensure system meet requirements
- Regression testing after any significant change to ensure nothing broke

Continuous Integration

Following AGILE Practices

- Development is continuous
- No separate phase for testing
- Develop and test continuously, feature by feature
- Features change, new features come up
- Testing must adapt to changes

Continuous Integration



Continuous Integration

“Continuous Integration”

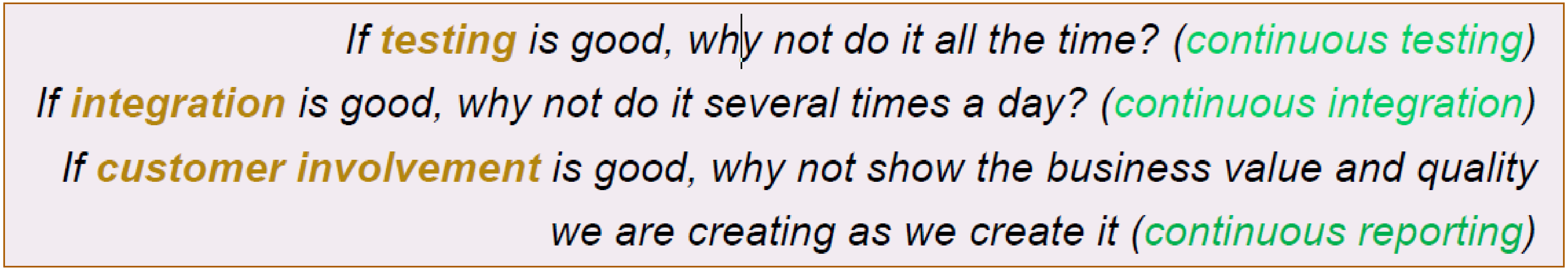
- Follows the Agile Approach
- Definition:
  - Integrate & build the system several times a day
  - Integrate every time a task is completed
  - Let’s you know every day the status of the full system
- Continuous integration and relentless testing go hand-in-hand.
- By keeping the system integrated at all times, you increase the chance of catching defects early and improving the quality and timeliness of your product.
- Continuous integration helps everyone see what is going on in the system at all times.

Continuous Integration

“Continuous Integration is 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

https://en.wikipedia.org/wiki/Martin_Fowler



*If **testing** is good, why not do it all the time? (continuous testing)*

*If **integration** is good, why not do it several times a day? (continuous integration)*

*If **customer involvement** is good, why not show the business value and quality we are creating as we create it (continuous reporting)*

Continuous Integration

Why do Continuous Integration?

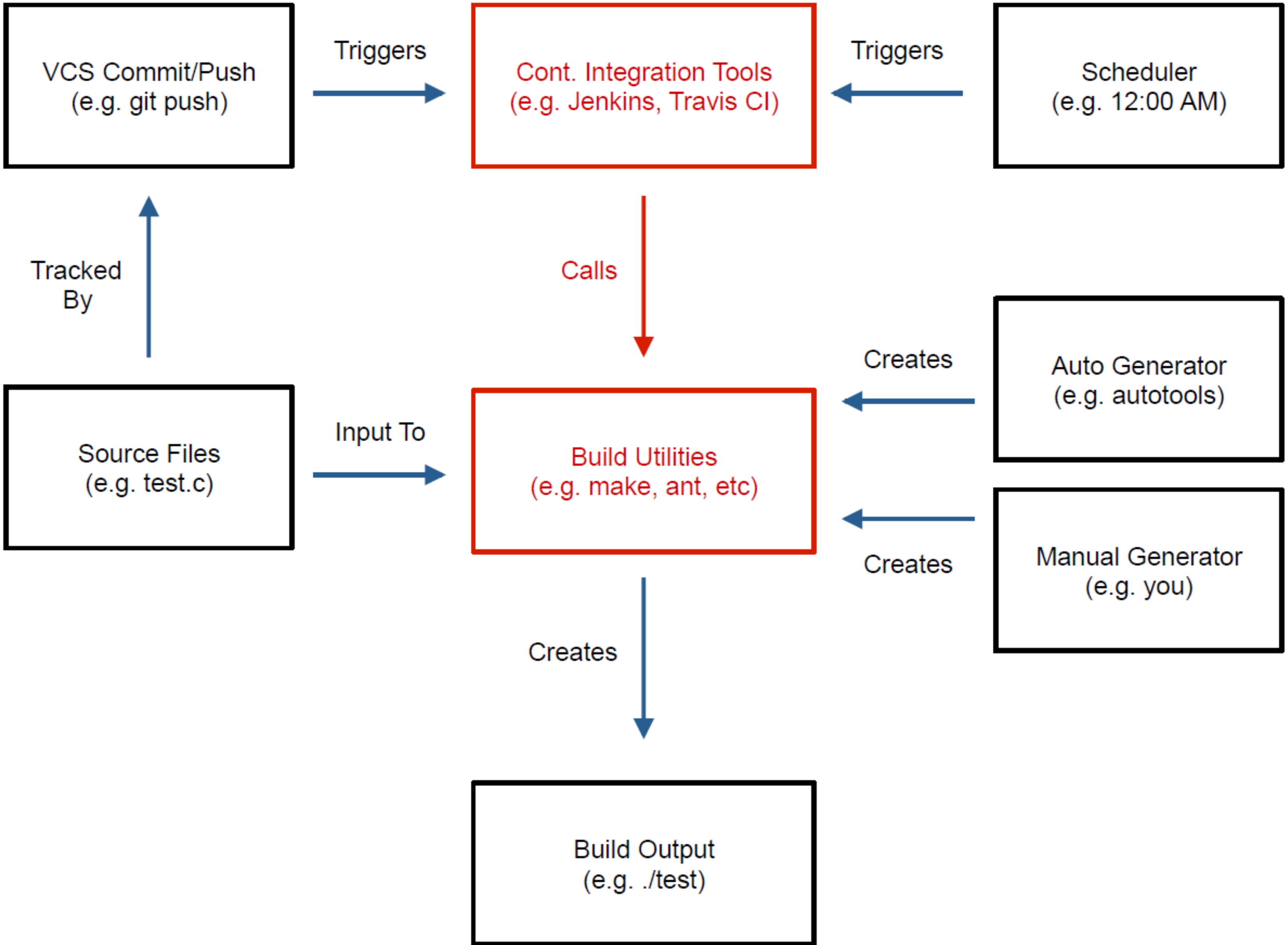
- Speed Up Each Feature/Sprint
- Automate Deployment
- Guarantee that Necessary Tests Are Being Run
- Track Build and Test Status
- Immediate Bug Detection
- Yields more Bug-Free Code
- A “Deployable” system at any given point
- Record of the history/evolution of the project

Continuous Integration

At regular intervals (ideally at every commit), the system is:

- Integrated
  - All changes up until that point are combined into the project
- Built
  - The code is compiled into an executable or package
- Tested
  - Automated test suites are run
- Archived
  - Versioned and stored so it can be distributed as is, if desired
- Deployed
  - Loaded to an environment where the developers can interact with it

Continuous Integration

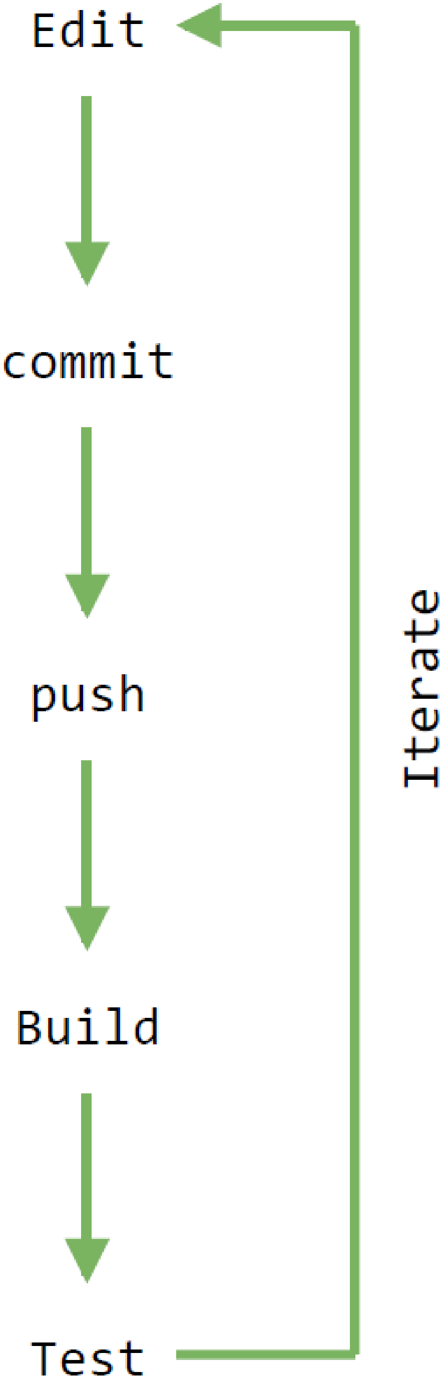


Continuous Integration

Fowler’s 10 Best Practices for CI

1. Maintain a Single Source Repository
2. Automate the Build
3. Make your Build Self-testing
4. Everyone Commits Everyday
5. Every Commit should Build the Mainline on an Integration Machine
6. Keep the Build Fast
7. Test in a Clone of the Production Environment
8. Make it easy for Anyone to get the Latest Executable
9. Everyone can see what’s Happening
10. Automate Deployment

Continuous Integration



Edit

commit

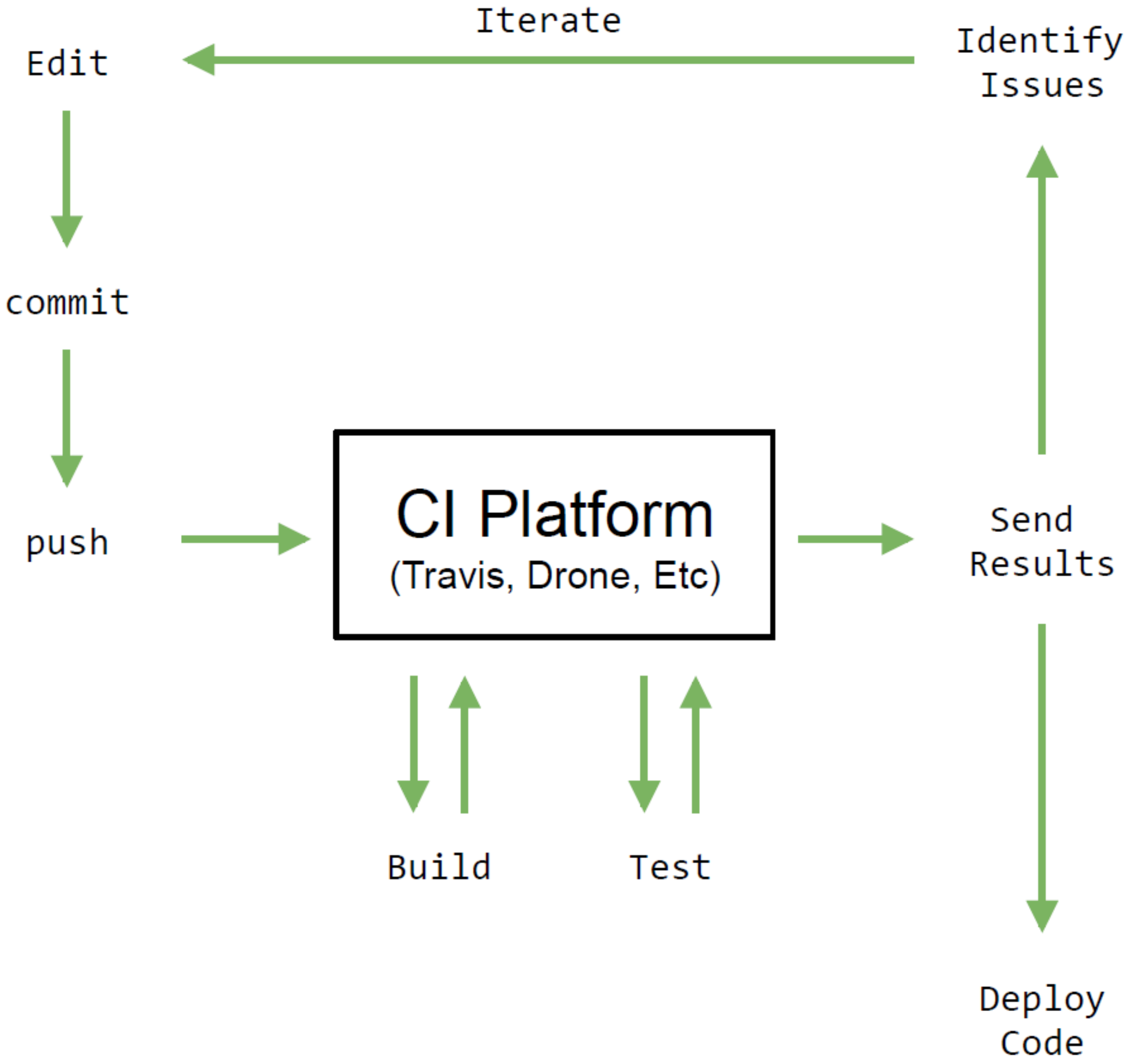
push

Build

Test

Iterate

Continuous Integration



Continuous Integration

drone.io: https://drone.io/

Public Projects: Free

Private Projects: Paid

Concurrent Builds: 1

VCS: GitHub, Bitbucket, Google Code

Travis CI: https://travis-ci.org/

Public Projects: Free

Private Projects: Paid

Concurrent Builds: A Few

VCS: GitHub

Upcoming Stuff

Lab 12 – Continuous Integration

Thursday & Friday this week

Milestone 5 due this week (Week 14) -- Unit Testing

1. Create a document that describes how at least three features within your finished product will be tested. The test plan should include specific test cases (user acceptance test cases) that describe the data and the user activity that will be executed in order to verify proper functionality of the feature.
2. Provide a link to the tool you use to automate testing, OR explain how to run your automated test cases OR schedule time with the TAs to demonstrate your automated tests. Provide a copy of the output showing the results of the automated test cases running.

Upcoming Weeks

Milestone 6 due Week 16

- Ten Minute Team Presentation
- Sign Up for Time Slot on Moodle
- 15 Teams
- Extra Credit (40, +5) for Presenting During Lecture Slots (up to 8 team presentations)
- All Members Present
- All Members Present

Upcoming Weeks

Milestone 6: Presentation Content

- Team Identity
- Title of Project
- Names/Introductions of Team Members
- Overview of Project –
  - What does the software do for whom?
  - How does it work?
- Tools List
  - Name of tool, logo, purpose
  - Rating for each tool
    - Score 1 – 5: 1= worthless, 5=great
  - Overview of Methodology Followed

Upcoming Weeks

Milestone 6: Presentation Content

- Challenges you Encountered
- How you Overcame them
- Key Lessons Learned
  - About tools
  - About methods
  - About software development

- Hints:
  - Everyone talks
  - A picture is worth 1000 words

Upcoming Weeks

Next Week (Week 15):

- Brief Lecture on system documentation
- Review for 2nd mid-term exam (during lab next week.)
  - Timed (60 minutes)
  - One attempt