Automated Testing
and
Release Engineering

@briancolfer
bcolfer@apple.com

I will focus on automating tests for a continuous pipe line.
Tell you a story about how I delivered a better experience for application and quality developers.
Keep test runs small and focussed. Many small test runs are better than a few big ones.

Introduction to release engineering and testing

Create tools and integration that takes source artifacts to deploy target artifacts. The critical component that adds value to continuous delivery is testing.

Test Automation and Continuous Delivery

- Types of tests
- Types of test results
- Automated test framework examples

Types of tests: Unit tests, development integration tests, Test case integration tests, Acceptance tests
Types of test results: Pass, Fail, Error, Info
Automated test framework examples: TestNG, Cucumber, Spock, JUnit, JMeter

A layered approach to automated tests and release engineering

- Each stage provides more confidence
- Only test enough for the next stage in the release pipeline
- Treat the tests and continuous delivery pipeline as part of the value to the product.

How to organize the running of tests

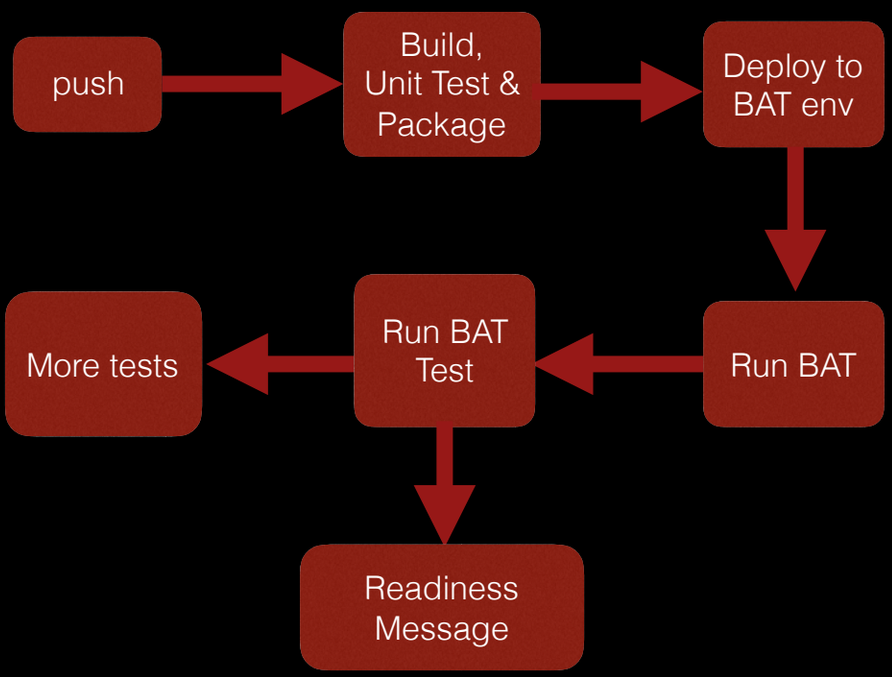
- Break large test suites to small test runs
- Give test feedback that is obvious on how to reproduce the results
- Provide circuit breakers to the release pipeline

Each test run should provide a specific level or area of confidence
Report not just pass or fail
Circuit breakers for services that are not critical but important … if SonarQube is down

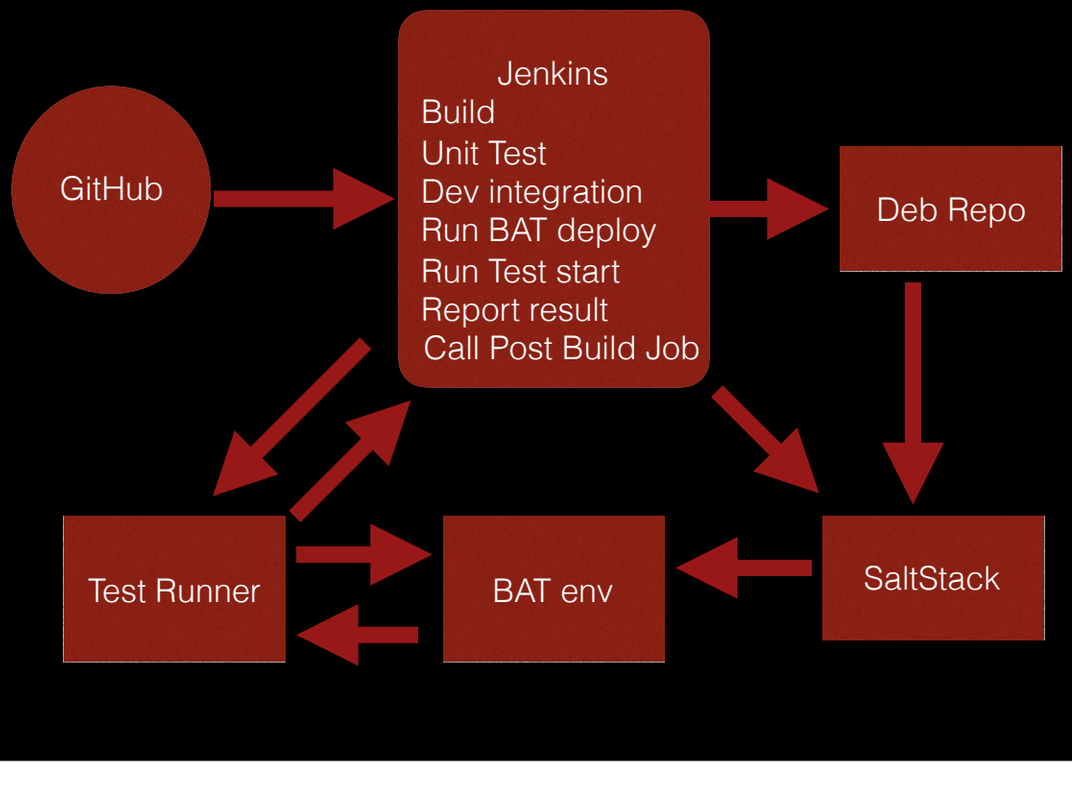
Summary

- Have many small steps in the CD pipeline
- Test code is part of the product
- Optimize the CD pipeline to provide confidence

Process Flow



Systems


GitHub
Jenkins
Build
Unit Test
Dev integration
Run BAT deploy
Run Test start
Report result
Call Post Build Job
Deb Repo
Test Runner
BAT env
SaltStack

How to contact me, Brian Colfer:

@briancolfer

bcolfer@apple.com