

# Continuous Delivery

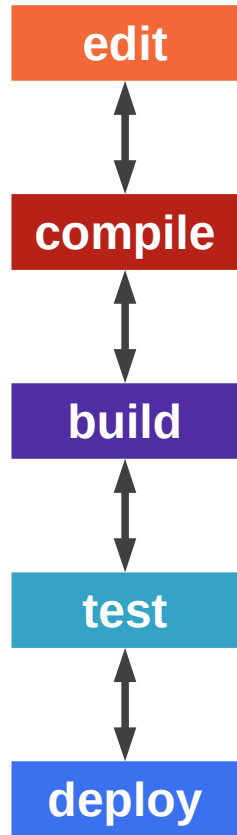
**Prof. Dr. Dirk Riehle**

**Friedrich-Alexander University Erlangen-Nürnberg**

**AMOS F02**

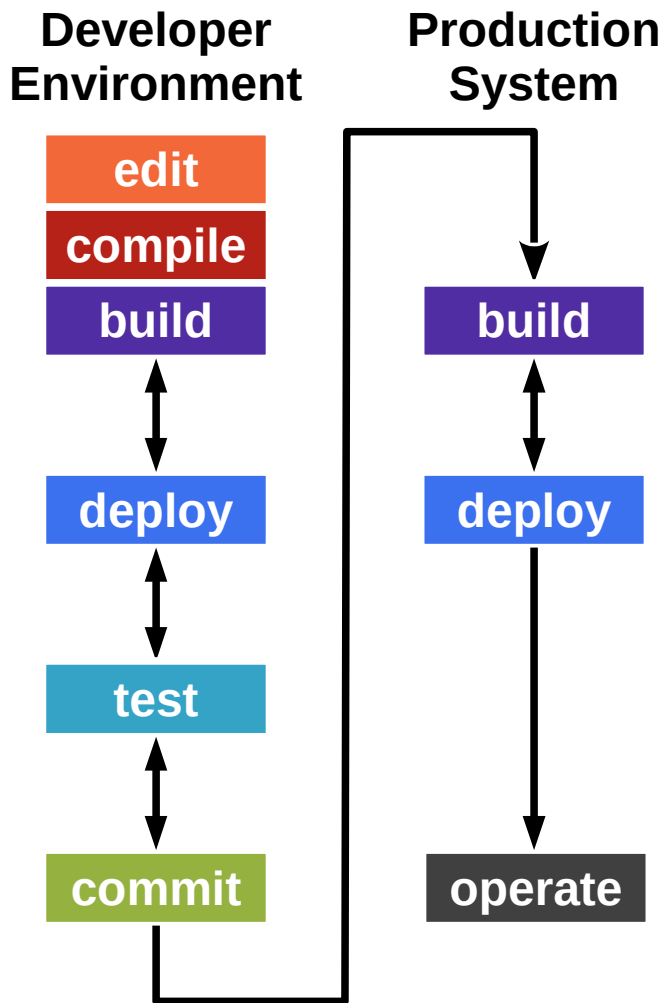
Licensed under CC BY 4.0 International

# Beginner's Development Cycle



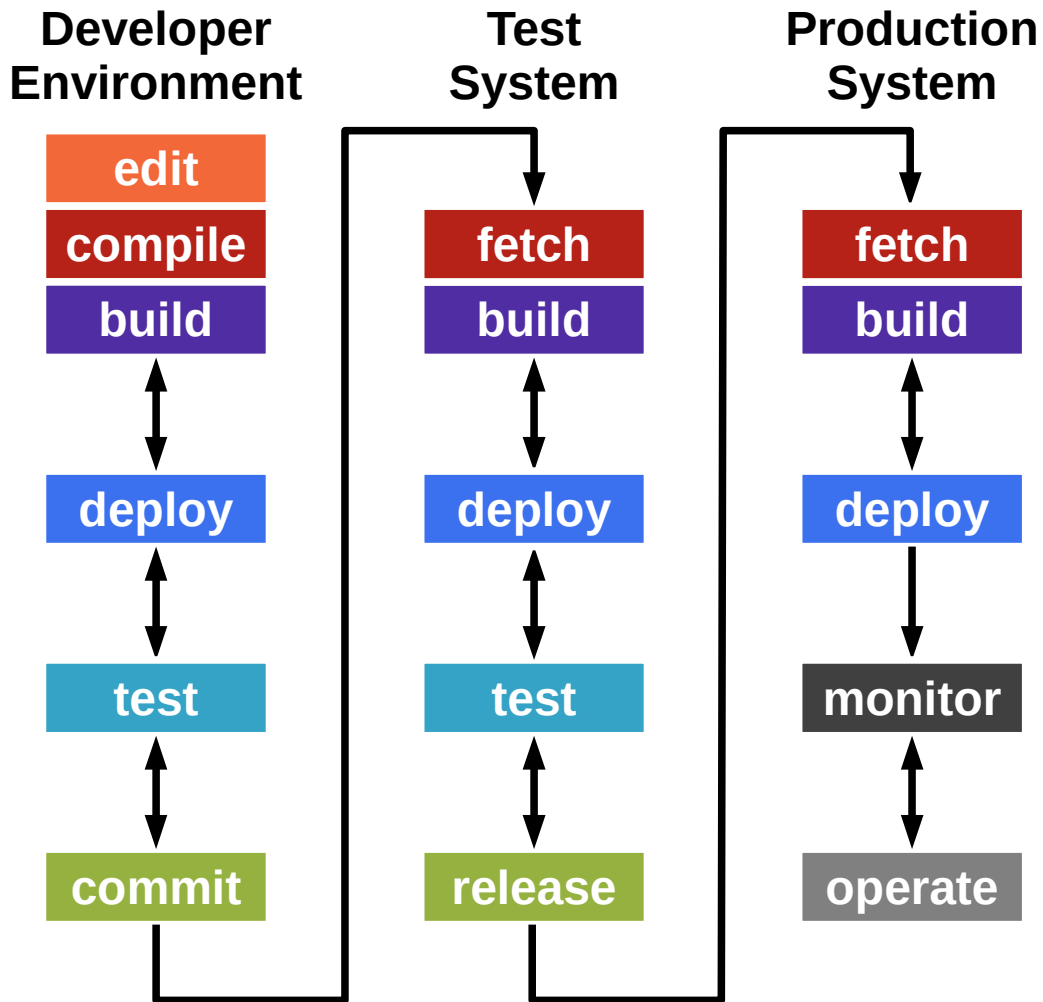
- Developer
  - Edit: Implements new feature
    - Iterates over the code until it looks right
  - Compile: Compiles the code
    - Iterates over the code until it compiles (no syntax error)
  - Build: Puts classes, build path together
    - Packages jar, if any, by hand
  - Test: Tests the program
    - Keeps going until “behavior looks right” i.e. no bugs
  - Deploy: Puts code into production
    - If a student, submits homework

# Practitioner's Development Cycle



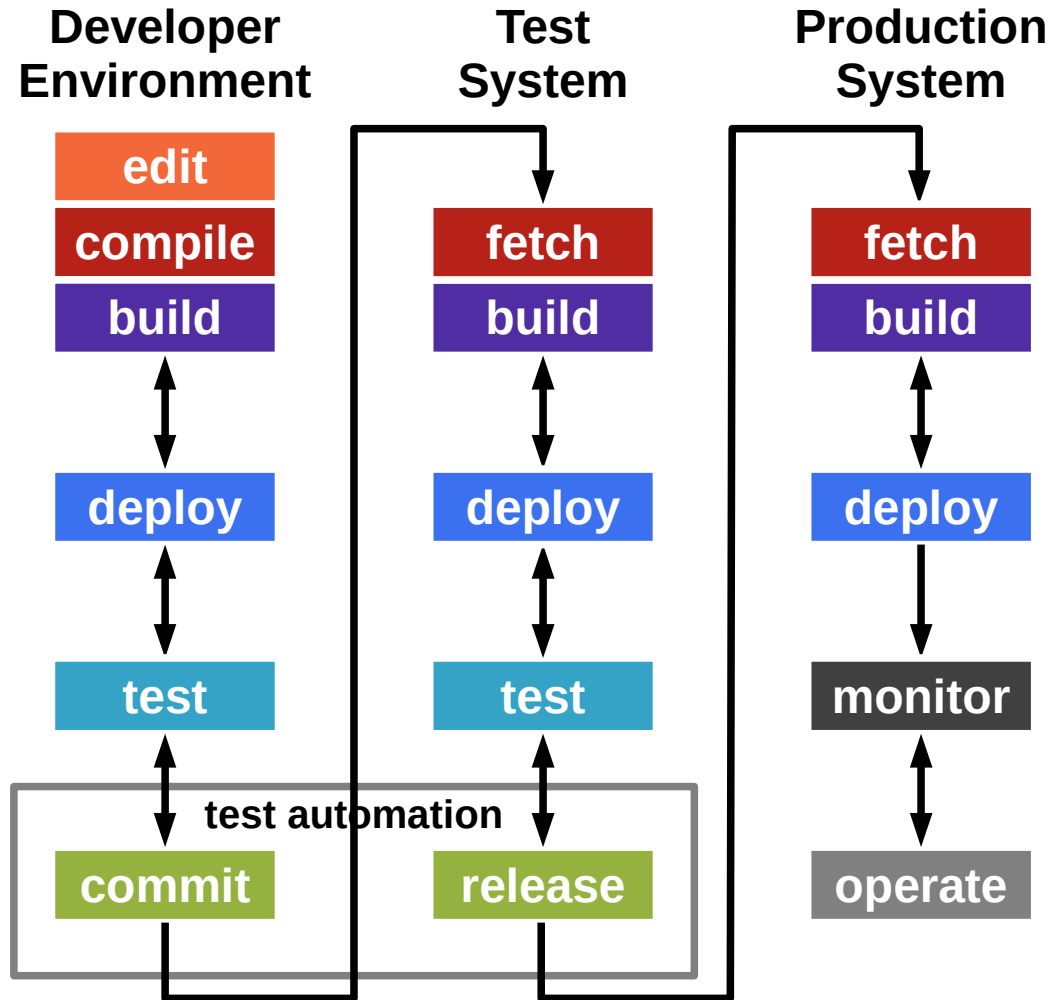
- Developer
  - Finishes development work
    - Uses local edit-compile-test cycle
    - Commits to indicate finishing
  - Builds for production environment
    - Could possibly use local build
  - Deploys to production system

# Professional Development Cycle



- QA engineer
  - Fetches code
  - Builds full system
    - Deploys in test system
  - Tests full system
    - Automated and by-hand
      - Component tests
      - Acceptance
      - Integration tests
  - Deploys full system
  - Operates system

# Continuous Integration Development Cycle



- Release (QA) engineer
  - Fetches code
  - Builds full system
    - Deploys in test system
  - Tests full system
    - Automated and by-hand
      - Component tests
      - Acceptance
      - Integration tests
  - Deploys full system
  - Operates system

# Test Automation

- Test automation ...
  - automatically carries out all available tests
    - Component tests (unit tests)
    - Acceptance tests (functional tests)
    - Integration tests and system tests
  - provides feedback to development and QA

# Continuous Integration

- Continuous integration (CI) is a code integration process
  - Upon trigger (commit to official repository)
    - the system under construction is fetched, built, deployed, and tested
    - in a fully automated way (no human intervention)
  - Feedback upon system status is provided to both
    - developers and
    - managers
- The purpose of continuous integration is to
  - always know where you are standing with respect to the project
  - ideally improve quality such that you can deploy at any time
- Continuous integration requires test-driven development

# Continuous Integration and Lava Lamps

In the early days, lava lamps were used to signal whether the project could be deployed to production or not.





# Continuous Integration Dashboards



**Continuous integration quickly evolved to build and test status dashboards hung on office walls for everyone to see.**

# Example Dashboard Build #437

[Jenkins](#) » [fsahoy](#) » [#437](#)

ENABLE AUTO REFRESH

[Back to Project](#)

[Status](#)

[Changes](#)

[Console Output \[raw\]](#)

[View Build Information](#)


[Polling Log](#)

[Test Result](#)


[See Fingerprints](#)

[Previous Build](#)


[Next Build](#)


 **Build #437 (Jul 27, 2011 12:40:18 PM)**

Started 6 mo 28 days ago  
Took [1 min 7 sec](#)

 Changes

1. version bump to 1.1-SNAPSHOT, added ignored UpdateLogbook selenium test ([detail](#) / [hgwweb](#))

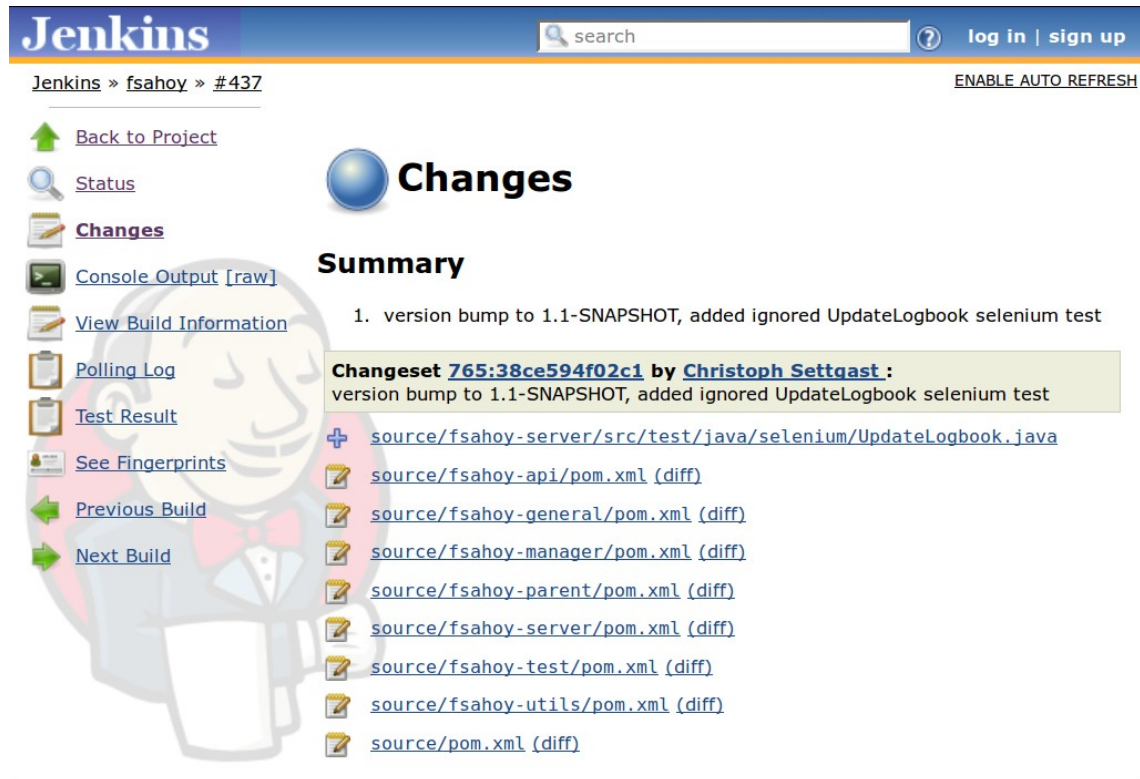
 Started by an SCM change

 [Test Result](#) (no failures)

**Module Builds**

|                                            |          |
|--------------------------------------------|----------|
| <a href="#">FreeSeasAhoy Project</a>       | 0.94 sec |
| <a href="#">FreeSeasAhoy Api</a>           | 5.4 sec  |
| <a href="#">FreeSeasAhoy General</a>       | 3.8 sec  |
| fsahoy-main (didn't run)                   |          |
| <a href="#">FreeSeasAhoy Manager</a>       | 1.3 sec  |
| <a href="#">FreeSeasAhoy MetaProject</a>   | 0.11 sec |
| <a href="#">FreeSeasAhoy Test Fixtures</a> | 1.2 sec  |
| <a href="#">FreeSeasAhoy Utilities</a>     | 3.3 sec  |

# Example Dashboard Changes #437



**Jenkins**  [log in](#) | [sign up](#)

Jenkins » [fsahoy](#) » [#437](#) [ENABLE AUTO REFRESH](#)

[Back to Project](#) [Status](#) **Changes** [Console Output \[raw\]](#) [View Build Information](#) [Polling Log](#) [Test Result](#) [See Fingerprints](#) [Previous Build](#) [Next Build](#)

## Changes

### Summary

- version bump to 1.1-SNAPSHOT, added ignored UpdateLogbook selenium test

**Changeset 765:38ce594f02c1 by Christoph Settgaest:**  
version bump to 1.1-SNAPSHOT, added ignored UpdateLogbook selenium test

- [source/fsahoy-server/src/test/java/selenium/UpdateLogbook.java](#)
- [source/fsahoy-api/pom.xml \(diff\)](#)
- [source/fsahoy-general/pom.xml \(diff\)](#)
- [source/fsahoy-manager/pom.xml \(diff\)](#)
- [source/fsahoy-parent/pom.xml \(diff\)](#)
- [source/fsahoy-server/pom.xml \(diff\)](#)
- [source/fsahoy-test/pom.xml \(diff\)](#)
- [source/fsahoy-utils/pom.xml \(diff\)](#)
- [source/pom.xml \(diff\)](#)

Page generated: Feb 20, 2012 1:37:09 PM [Jenkins ver. 1.417](#)

# Example Dashboard Test Results #437

**Jenkins**[?](#) [log in](#) | [sign up](#)

Jenkins » fsahoy » #437 » Test ResultENABLE AUTO REFRESH

[Back to Project](#)  
[Status](#)  
[Changes](#)  
[Console Output \[raw\]](#)  
[View Build Information](#)  
[Polling Log](#)  
**[Test Result](#)**  
[See Fingerprints](#)  
[Previous Build](#)  
[Next Build](#)

## Test Result

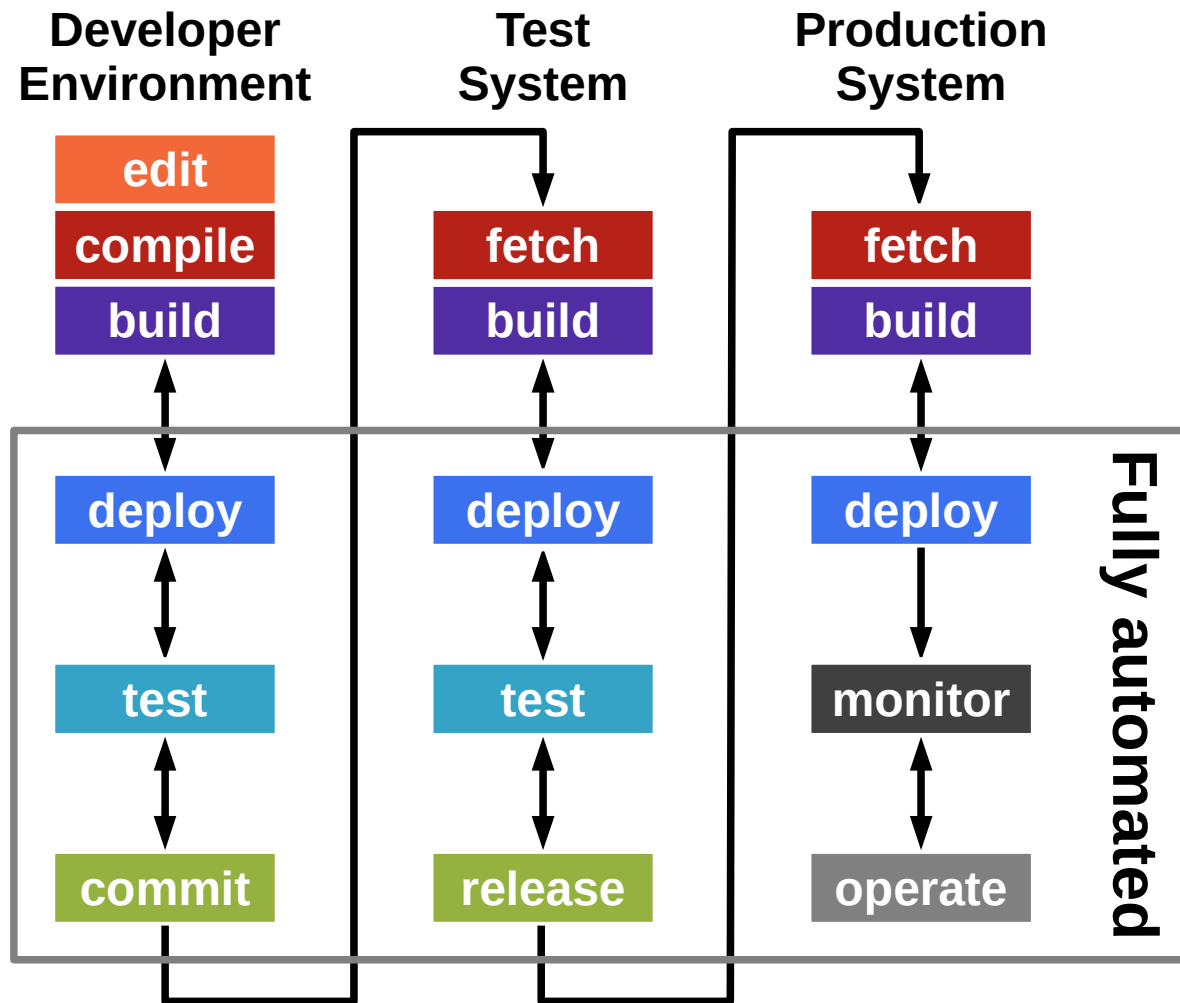
0 failures (±0) , 3 skipped (±0)

107 tests (±0)

| Module                                          | Fail | (diff) | Total | (diff) |
|-------------------------------------------------|------|--------|-------|--------|
| <a href="#">de.fau.cs.fsahoy:fsahoy-api</a>     | 0    |        | 2     |        |
| <a href="#">de.fau.cs.fsahoy:fsahoy-general</a> | 0    |        | 71    |        |
| <a href="#">de.fau.cs.fsahoy:fsahoy-server</a>  | 0    |        | 20    |        |
| <a href="#">de.fau.cs.fsahoy:fsahoy-utils</a>   | 0    |        | 14    |        |

Page generated: Feb 20, 2012 2:15:28 PM [Jenkins ver. 1.417](#)

# Continuous Deployment Development Cycle



- Fully automated
  - Compile and build
  - Deployment
    - To test environment
    - To production
  - Test execution
- Partially automated
  - System monitoring
  - Automated rollback
- Human decisions
  - Commit decision
- No release decision

# Continuous Delivery

- Continuous delivery is a delivery process
  - Upon trigger (commit to official repository)
    - the system is integrated, tested, and **deployed to production**
    - in a fully automated way (no human intervention)
  - A poorly functioning system may be rolled back
    - Requires monitoring and rollback facility of deployed system
    - System status is assessed using key figures
- The purpose of continuous delivery is to
  - put development results into production as fast as possible
  - improve quality by holding the team to high operational standards

- 1. Test automation**
- 2. Continuous integration**
- 3. Continuous deployment [1]**

[1] Short for “development operations”

# Continuous Delivery 2 / 2

- Test automation =
  - Tests and testing
- Continuous integration =
  - Test-driven development +
  - Automated building +
  - Test automation
- Continuous deployment =
  - Continuous integration +
  - Deploy to production +
  - Monitoring and rollback
- DevOps
  - Continuous deployment +
  - Operations and culture



# Review / Summary of Session

- Continuous delivery
  - Test automation
  - Continuous integration
  - Continuous deployment
  - Development operations

# Thank you! Questions?

[dirk.riehle@fau.de](mailto:dirk.riehle@fau.de) – <http://osr.cs.fau.de>

[dirk@riehle.org](mailto:dirk@riehle.org) – <http://dirkriehle.com> – [@dirkriehle](#)

# Credits and License

- Original version
  - © 2020 Dirk Riehle, some rights reserved
  - Licensed under [Creative Commons Attribution 4.0 International License](#)
- Contributions
  - None yet