Continuous Integration with Jenkins

Gabriel Mateescu
Software Engineer at DIPF
mateescu@acm.org

Agenda

- DevOps principles
- Continuous Integration (CI)
- CI with Jenkins
  - Demo Maven Project built with Jenkins
- Summary

Education

- Virginia Tech, USA (1993-1998)
  - Master’s in Computer Science (1997)
  - PhD in Computer Science (1998)

Work Experience

- System engineering
  - National Research Council Canada (2000-2007)
  - Leibniz Rechenzentrum (2007-2008)
  - Virginia Tech Adv. Research Comp. (2012-2013)
  - EURAC Research (2013 – 2014)
  - DIPF (2014-present)
- Software development
  - Virginia Bioinformatics Institute (2008-2009)
  - NCSA (2010-2011)
  - EPFL (2011-2012)

Projects (1)

- Continuous Delivery
  - Build systems:
    - GNU Make, CMake, Ant, Maven
  - Test automation
    - Junit, Maven, Jenkins
  - Source Code Management
    - Subversion, Git
  - Continuous Integration
    - Jenkins, Bamboo
  - Continuous deployment
    - Puppet
  - Monitoring
    - Icinga/Nagios, Ganglia

Projects (2)

- Big Data analytics
  - Data ingestion into Hadoop and export
    - Sqoop
  - Batch analyics
    - MapReduce, Hive
  - Relational databases with sharded data
    - MySQL cluster on AWS
  - NoSQL databases
    - HBase for semantic web

Projects (3)

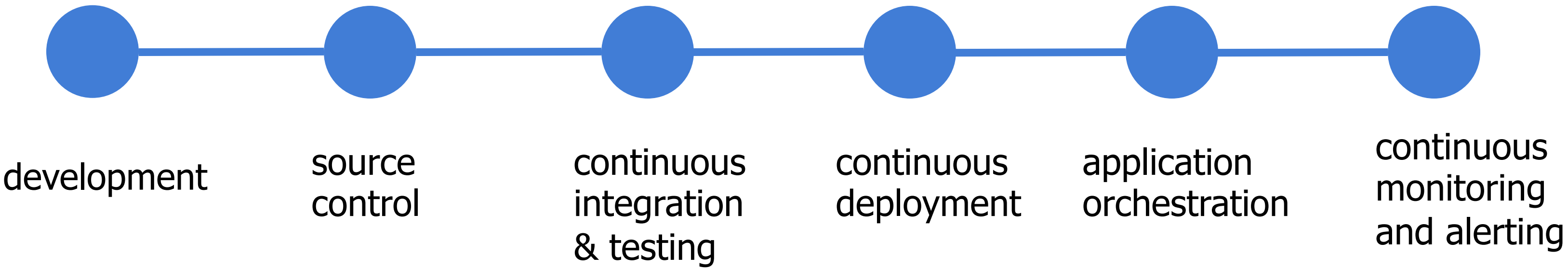
- Web Application development & scaling
  - Epidemiology Portal (VBI)
  - Proteomics data management (NSC)
  - Computer Based Testing web application (DIPF)
  - Load-balanced server clusters (DIPF)
    - Load testing with JMeter
    - Load balancing with Apache proxy_balancer

What is DevOps?

- A methodology for delivering better quality software faster and more reliably
  - emphasizes collaboration and communication between development and operations teams
- DevOps relies itself on software that manages developing, building, testing, deploying and monitoring applications
  - DevOps engineers have cross-functional role
  - Variants: Dev-Test-Ops, NoOps (Netflix)

The software delivery pipeline

- DevOps achieves software lifecycle management using the delivery pipeline



development

source control

continuous integration & testing

continuous deployment

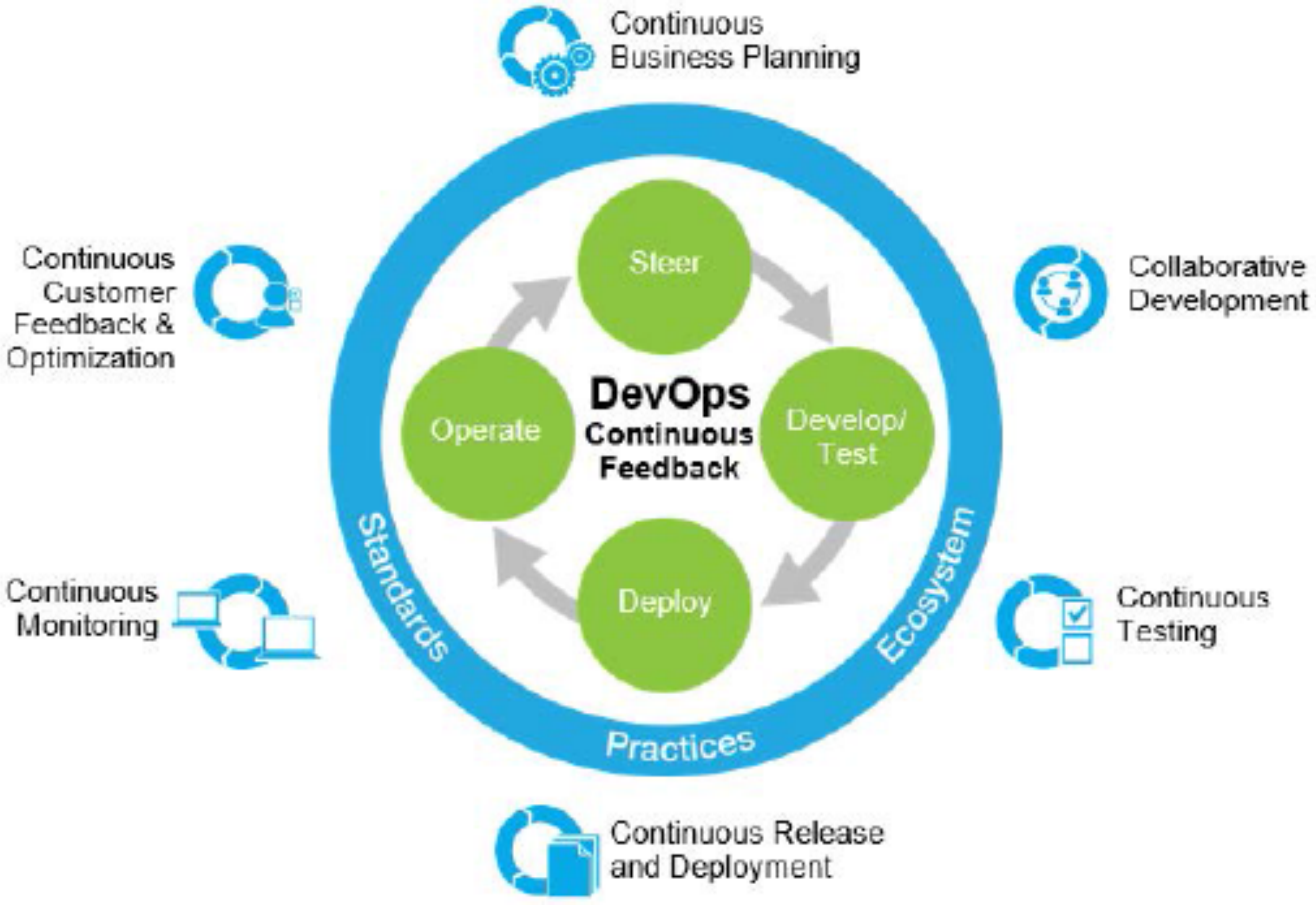
application orchestration

continuous monitoring and alerting

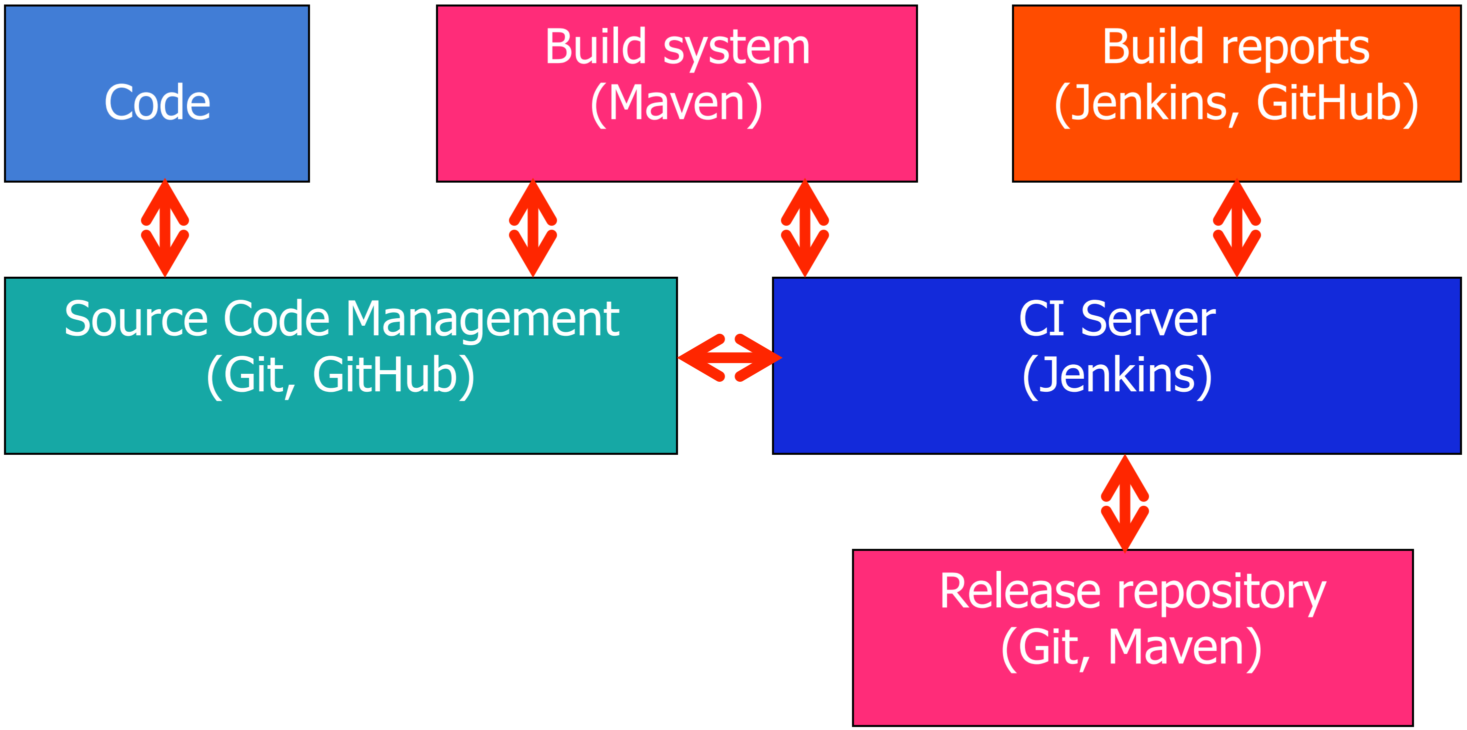
DevOps Principles

- Develop & test against production-like systems
  - test the app in an environment close to production
  - test the delivery process early in the lifecycle
- Deploy with repeatable and reliable processes
  - automate build, test, and deployment to the test, staging and production environments
- Monitor operational quality
  - achieve visibility of the entire environment
- Continuous feedback
  - continuous improvement of apps, environment, planning

DevOps Reference Architecture


Continuous Business Planning
Continuous Customer Feedback & Optimization
Collaborative Development
Continuous Monitoring
Continuous Testing
Continuous Release and Deployment
Steer
Develop/ Test
Deploy
Operate
DevOps
Continuous Feedback
Standards
Practices
Ecosystem

Continuous Integration System



Code

Build system
(Maven)

Build reports
(Jenkins, GitHub)

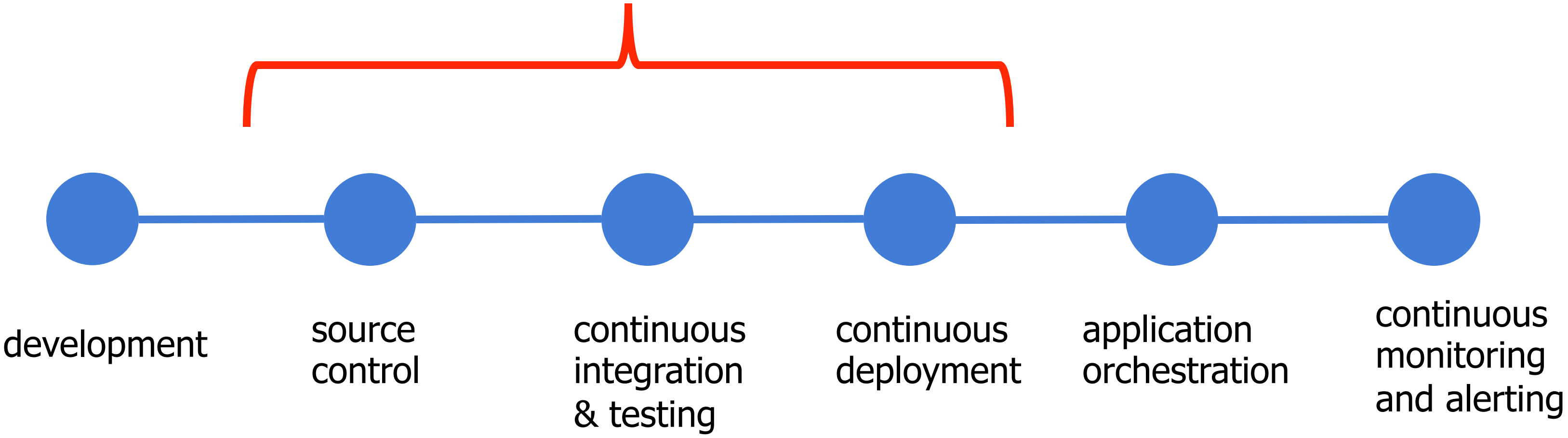
Source Code Management
(Git, GitHub)

CI Server
(Jenkins)

Release repository
(Git, Maven)

Jenkins & the delivery pipeline

- Jenkins



development

source control

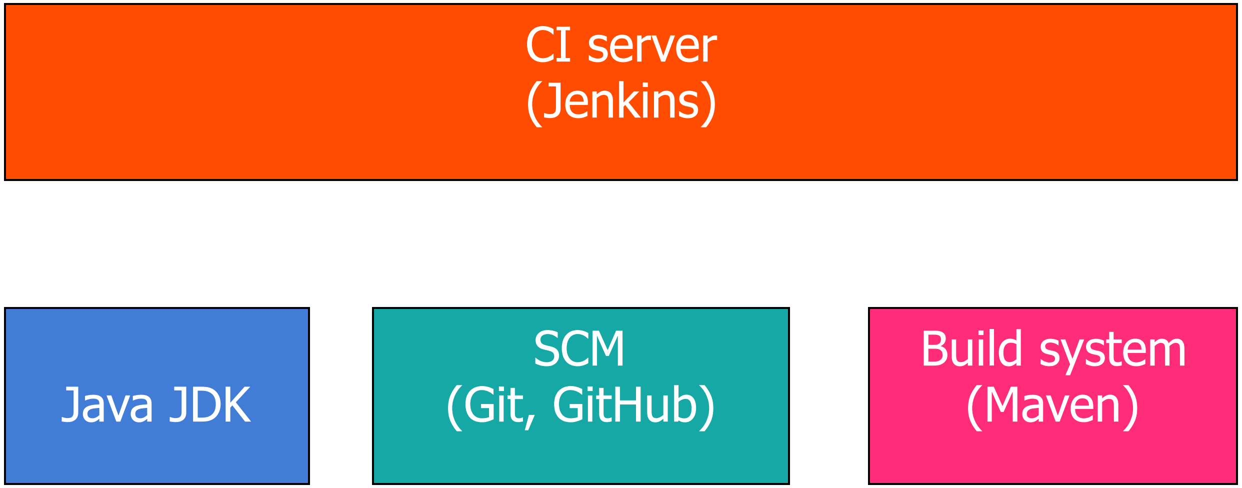
continuous integration & testing

continuous deployment

application orchestration

continuous monitoring and alerting

Jenkins Ecosystem



CI server
(Jenkins)

Java JDK

SCM
(Git, GitHub)

Build system
(Maven)

Conclusions

- DevOps enable agile software delivery
  - quickly change software to meet customer needs
  - lower failure rate of new software
- Jenkins provides a solid framework for CI delivery
  - continuous integration & testing
  - interface to build systems (Maven) and SCM (Git)
  - reporting & updating build status in GitHub