

# Agile + DevOps **WEST**

A TECHWELL EVENT

## **AD50**

Agile Testing & Test Automation

4:30 PM

## **AD50 - Getting to Continuous Testing**

Presented by:

**Max Saperstone**

Coveros

Brought to you by:




888-268-8770 · 904-278-0524 - [info@techwell.com](mailto:info@techwell.com) - <https://agiledevopswest.techwell.com/>


# Max Saperstone


Max Saperstone has been working as a software and test engineer for over a decade, with a focus on test automation within the CI/CD process. He specializes in open source tools, including the Selenium tool suite, JMeter, AutoIT, Cucumber, and Chef. Max has led several test automation efforts, including developing an automated suite focusing on web-based software to operate over several applications for Kronos Federal. He also headed a project with Delta Dental, developing an automated testing structure to run Cucumber tests over multiple test interfaces and environments, while also developing a system to keep test data "ageless." He currently heads up the development of Selenified, an open source testing framework to allow for testing of multiple interfaces, custom reporting, and minimal test upkeep.

# Getting to Continuous Testing

## A Greenfield Test Automation Story

 max.saperstone@coveros.com

 maxsaperstone

 @Automate\_Tests



# Max Saperstone

---

- Director of Testing and Automation at Coveros
- Over a decade of test automation experience
- Often Test Architect on consulting projects
- Helped transform multiple organizations to test effectively within sprints
- Mainly focuses on open source tools
- Lover of the outdoors and cheap beer



# Coveros

---

- Coveros helps organizations accelerate the delivery of secure, reliable software
- Our consulting services:
  - Agile software development
  - Application security
  - Software quality assurance
  - Software process improvement
- Our key markets:
  - Financial services
  - Healthcare
  - Defense
  - Critical Infrastructure

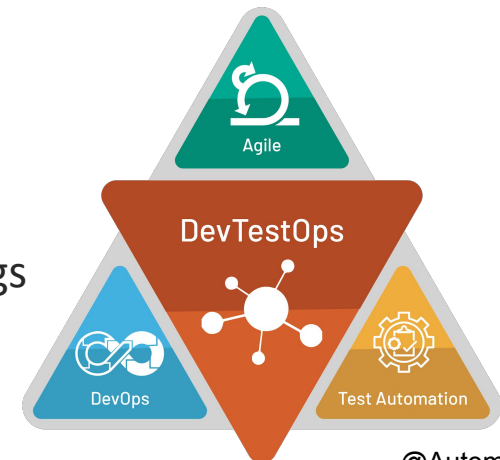




# Continuous Testing

---

- Continuous means testing is happening before, during, and after each software change is made
- Enabling testing activities:
  - Collaborate on requirements (ex: BDD)
  - Validating change constantly (ex: regression testing)
  - Dev/Test Pairing (ex: exploratory testing, reviewing test cases)
  - Automated testing in CI (ex: unit testing, API testing, code analysis)
  - Continuous improvement of test approach, test suites, test scripts
  - Review of customer feedback and product ratings

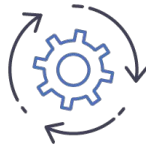


# Seven Continuous Testing Enablers

- Whole team quality
- Production-like environments (cloud if possible!)
- Build orchestration / Continuous integration
- Automated testing below the UI
- Service virtualization / emulation / simulation
- Infrastructure as code and/or container orchestration
- Test data management



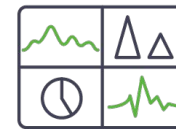
of New Code



of Every Build



of Every Deployment



in Production



**Shift Left**

**Expand Right**



# Background

---

- Healthcare company aiming for continuous release process
- No test automation
- No test automation team
- No continuous integration pipeline
- No test data management
- No test tooling or support



# Initial Investigation

---

- Previous tries at test automation did not stick
- 'Agile' organization with some waterfall practices
- Complex application, with multiple levels of the application not being tested
- Considerable time spent performing manual regression testing
- Identified some low hanging fruit for initial automation



# Roadmap

- Develop automation framework
- Build CI pipeline
- Build test automation pipeline
- Integrate test automation
- Determine quality thresholds
- Implement quality gates
- Setup feedback loops

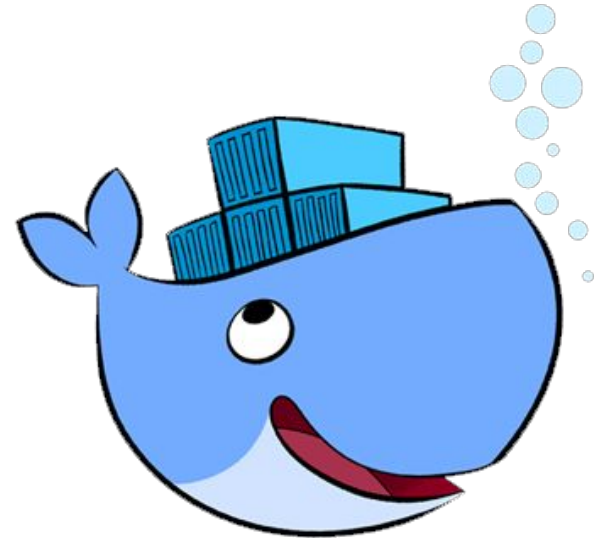
## ENTERPRISE AGILE TEST ADOPTION



# Develop Automation Framework

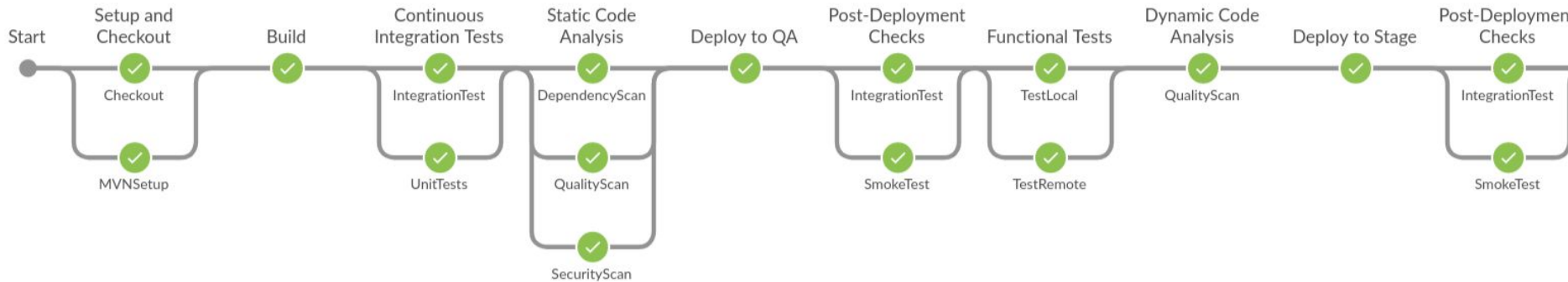
---

- Utilized BDD framework due to
  - QA technical capabilities
  - Application breakdown
- Went with separate framework repository
- Created separate repositories for each application 'flavor'
- Implemented Git and JIRA integrations
- Created Docker test containers



# Build CI Pipeline

- Created Jenkinsfiles for each source code repository
  - Compiles, unit tests, statically analyzes, and packages the application
  - Builds docker container
- Successful builds have containers pushed to artifact repository
- For dependent services, latest artifacts can just be pulled
  - Simple dependency system
- Deploying application is then a matter of pulling all latest artifacts



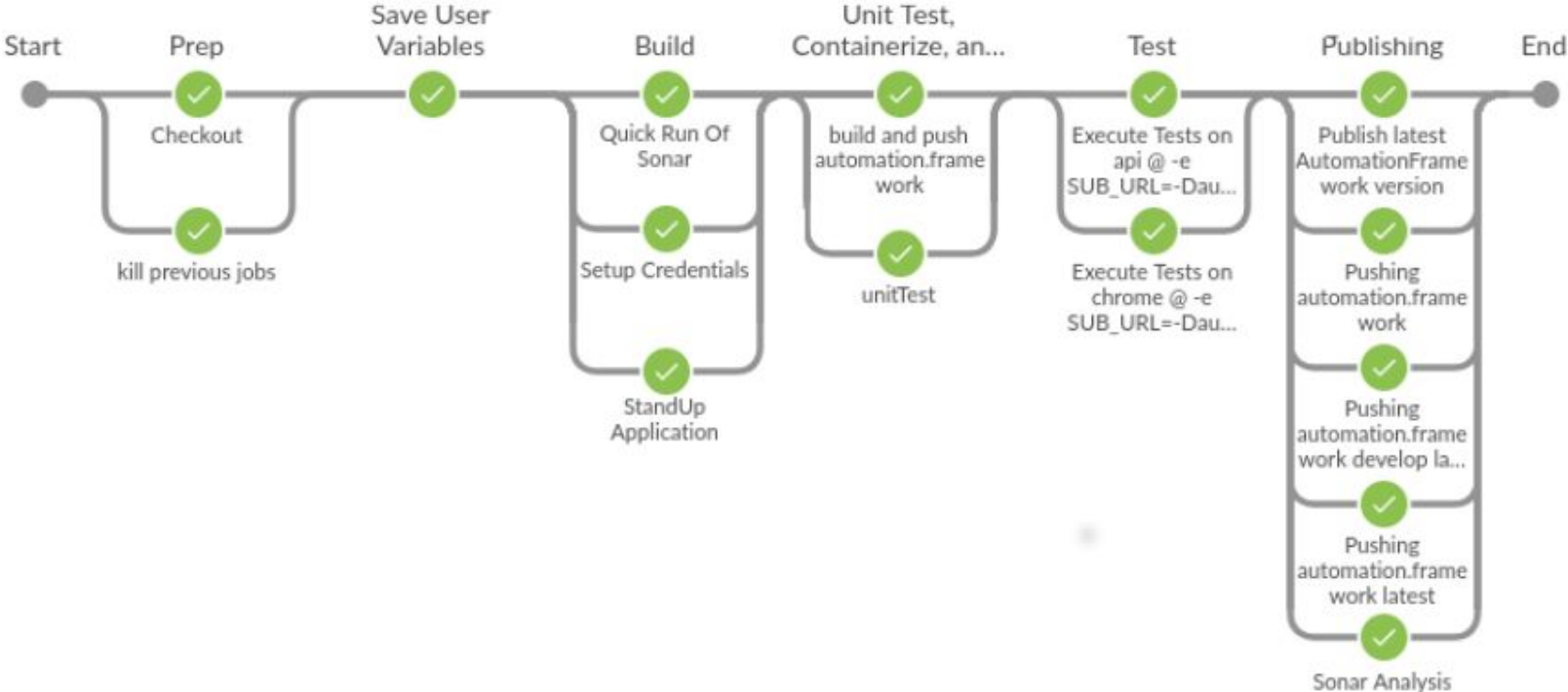
# Build Test Automation Pipeline

---

- Ensures tests provided to Devs are stable
- Added Jenkinsfiles for each test repository
  - Compiles, unit tests, and statically analyzes the test cases and test code
  - Builds docker container
- Stands up new instance of application
- Runs smoke tests against application
- Successful runs have containers pushed to artifact repository



# Test Automation Pipeline



# Integrate Test Automation

---

- Application architecture necessitated interdependent services
  - As a result, often an API change breaks the front end, and vice versa
- Added step into CI pipelines to run functional tests
  - Similar to Test automation pipeline, latest version of application is stood up
    - Service under test has docker container deployed
    - Other services have latest pulled from artifact repository
  - Functional tests are run against latest deployed coded from all services
- Passing tests indicate no major integration problems





# Quality Checks and Feedback Loops

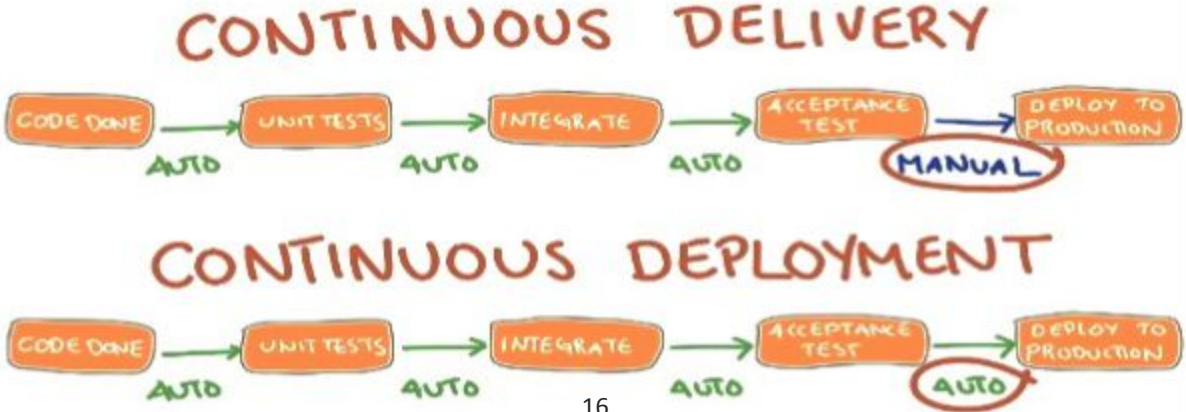
---

- Desired as much information back to developers as possible
- Created software quality thresholds in Sonarqube
- Used pull/merge requests (PR/MRs) as CI trigger
- For PR to merge:
  - No failed unit tests
  - Certain code coverage required
  - No blocker or critical code issues
  - No failed functional tests
  - Green/passing pipeline



# Beyond Continuous Integration

- Limited Continuous Deployment Capabilities
  - Currently working on expanding
  - Web smoke testing performed
- Nightly testing occurring to fill in gaps
- CI acts as gate for QA team receiving software
- After CI, software testing follows traditional deployment testing processes



# Continuous Improvement

---

- Further integrate manual and automated teams
- Expand out CI testing capabilities
- Incorporate testing into all application services
- Extend parallelization of testing
- Expand continuous delivery to continuous deployment
- Aim for daily releases



# Some Lessons Learned Along The Way

- Go with small iterative changes
  - No boiling the ocean
- Fully understand testing and deployment pain-points
- Manual and automated QA needs alignment to be successful
- Siloed testing organizations will not be successful performing continuous testing
  - Whole team buy in is required to truly be successful
- Identifying and implementing small wins will get the whole team on board





# Questions?

Join me on the  
**TechWell Hub**

[hub.techwell.com](https://hub.techwell.com)

#testing  
@max

