TECHWELL™
STAR EAST
A TECHWELL EVENT

T13
DevOps/Continuous Delivery
Thursday, May 3rd, 2018
1:30 PM

Release Automation: Yes, Testers Should Care

Presented by:

Tracy Ragan

OpenMake Software

Brought to you by:

TECHWELL™

350 Corporate Way, Suite 400, Orange Park, FL 32073
888-268-8770 - 904-278-0524 - info@techwell.com - http://www.stareast.techwell.com/

Tracy Ragan

OpenMake Software

Tracy Ragan has extensive experience in Agile DevOps implementations for large organizations. She has devoted her career to helping organizations define a lifecycle process that is not only lean but also repeatable—from automated builds through continuous deployment. For the past twenty years Tracy has worked with Fortune 1000 organizations to improve the delivery of software updates through testing and production. An author and frequent speaker at industry conferences, Tracy has served on the board of the Eclipse Foundation, an open source community of tools and projects. She is currently CEO of OpenMake Software and on the board of DeployHub, an open source continuous deployment community.

STAR EAST
A TECHWELL EVENT

Release Automation, Why Testers should care.

Presented by: Tracy Ragan, CEO, OpenMake Master Agile's Last Mile™

T13 – Thursday May 3rd, 1:30

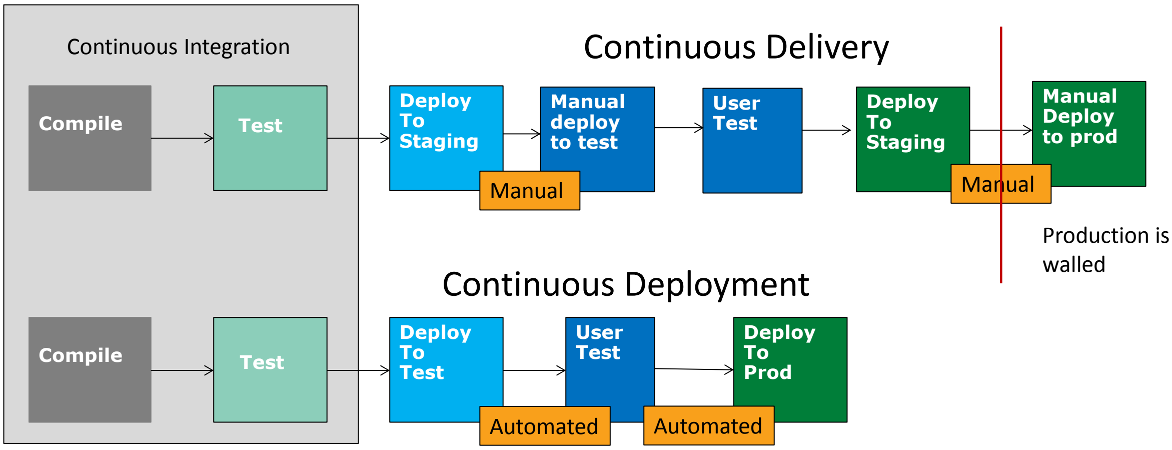
About the Presenter

CEO, OpenMake Software

Tracy was a consultant to Wall Street firms such as JP Morgan and Bankers Trust on build and release management for 7 years prior to OpenMake. She was a founding member of the Eclipse organization and served on the board for 5 years. She is a recognized leader and has been published in multiple industry publications and presented to wide audiences at industry conferences.

About OpenMake Software:
We are software engineers who build scalable Agile DevOps solutions that solve continuous delivery problems. Nothing drives us more than helping customers dramatically accelerate software release cycles from continuous build through continuous deployment

What is Application Release Automation



Continuous Integration

Compile → Test

Compile → Test

Continuous Delivery

Deploy To Staging → Manual deploy to test → User Test → Deploy To Staging → Manual Deploy to prod

Manual

Manual

Production is walled

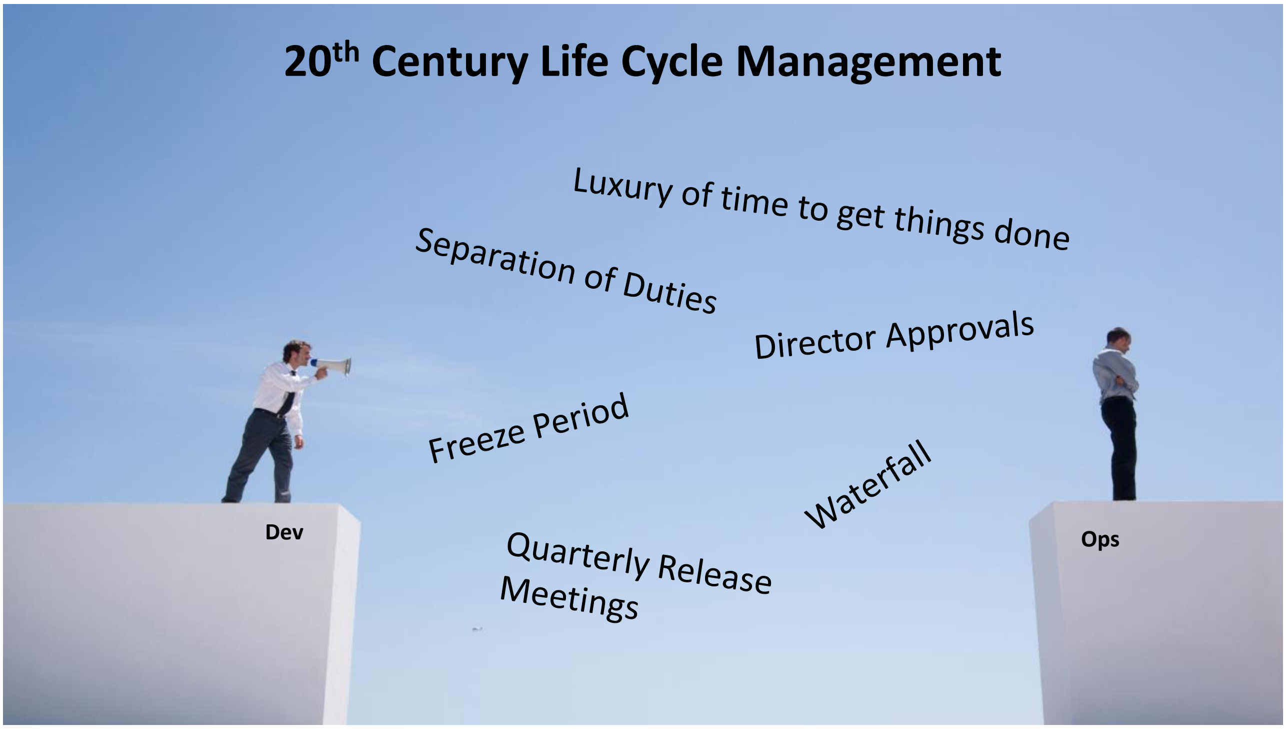
Continuous Deployment

Deploy To Test → User Test → Deploy To Prod

Automated

Automated

20th Century Life Cycle Management



Luxury of time to get things done

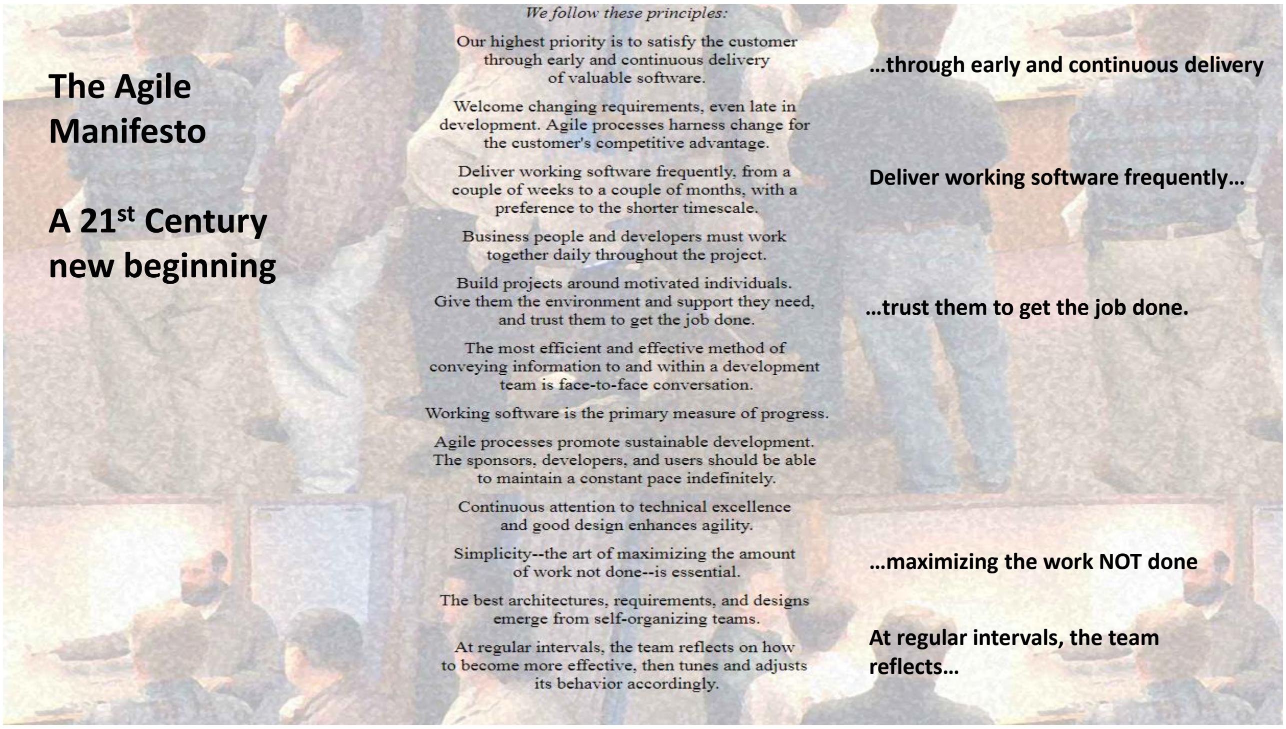
Separation of Duties

Director Approvals

Freeze Period

Waterfall

Quarterly Release Meetings



The Agile Manifesto

A 21st Century new beginning

We follow these principles:

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress.

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Continuous attention to technical excellence and good design enhances agility.

Simplicity--the art of maximizing the amount of work not done--is essential.

The best architectures, requirements, and designs emerge from self-organizing teams.

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

…through early and continuous delivery

Deliver working software frequently…

…trust them to get the job done.

…maximizing the work NOT done

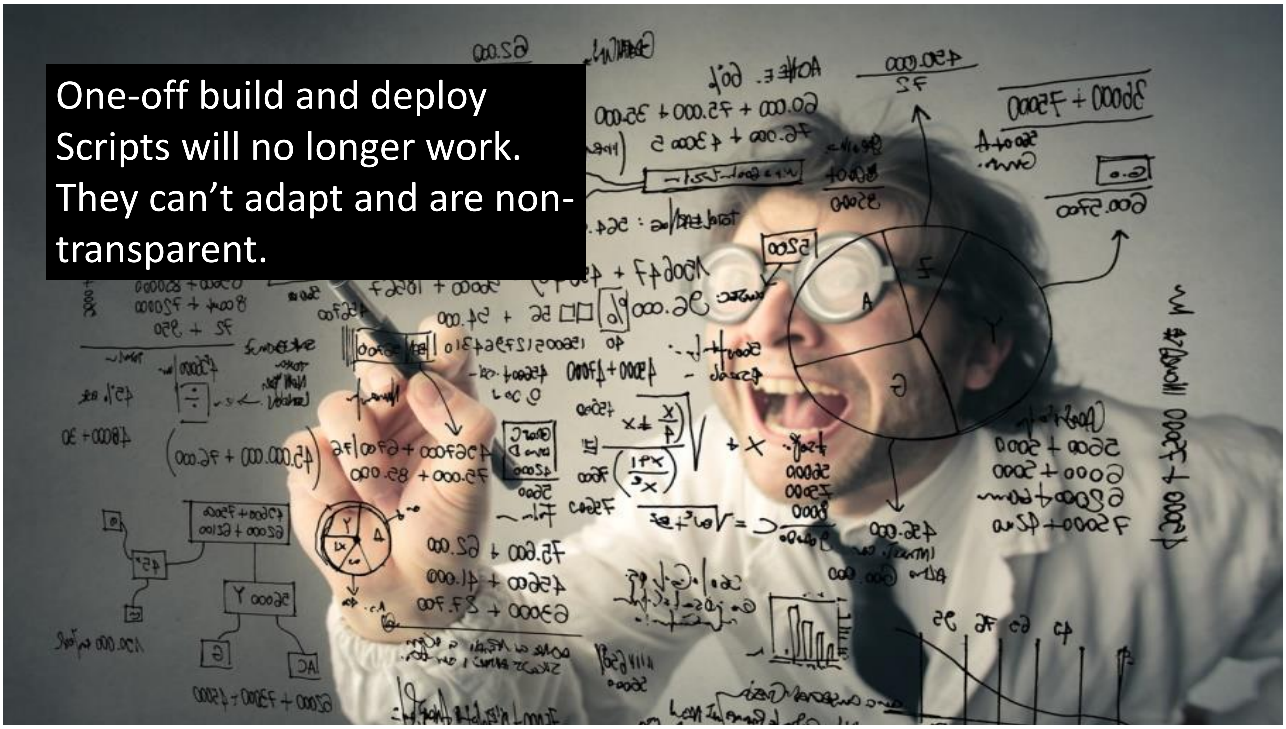
At regular intervals, the team reflects…

Through early and continuous delivery, working software is frequently delivered to customers.

This can be accomplished by:

- **maximizing the work NOT done through automation,**
- **providing continuous feedback at regular intervals for continuous improvement,**
- **and trusting developers to get the job done.**

In essence, DevOps Breaks the Barrier

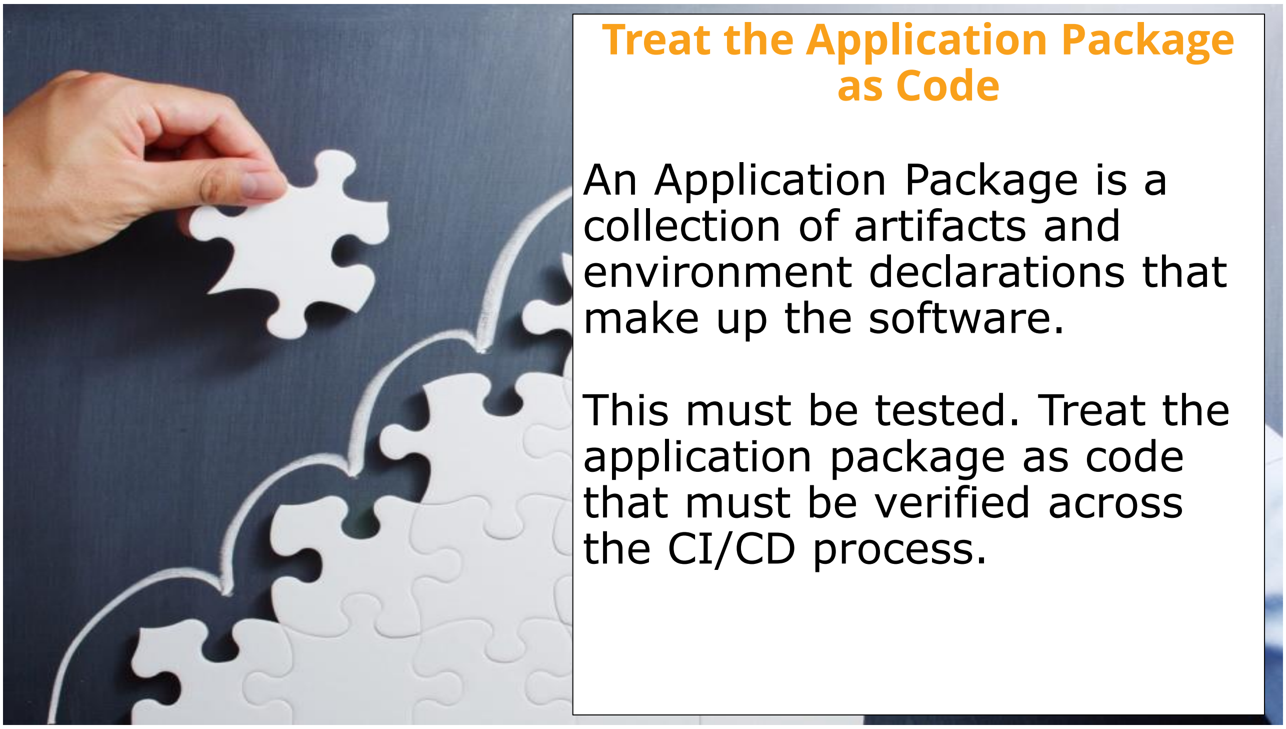


One-off build and deploy Scripts will no longer work. They can’t adapt and are non-transparent.

Testing in the Agile DevOps Landscape

Start testing beyond the tip – test your package installations across Environments

Ultimately, the QA team must be able to validate not just that the source code works, but works in a all environments – this means “it worked on my machine” just will not meet DevOps fast paced demands.

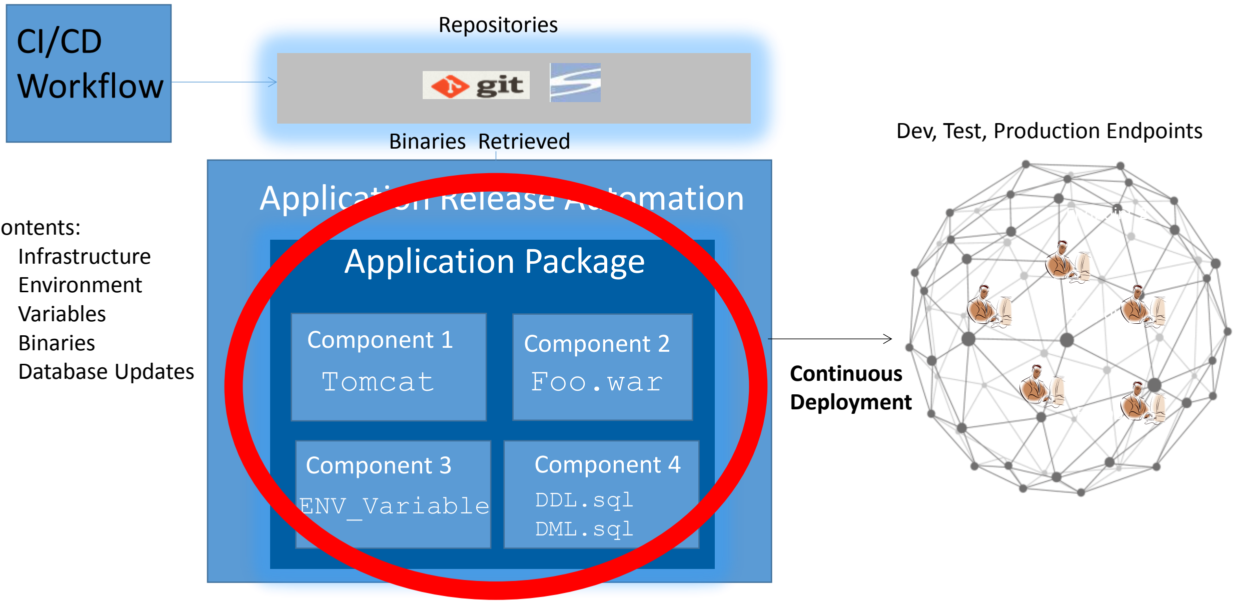


Treat the Application Package as Code

An Application Package is a collection of artifacts and environment declarations that make up the software.

This must be tested. Treat the application package as code that must be verified across the CI/CD process.

What's in a Package?



Contents:

- Infrastructure
- Environment Variables
- Binaries
- Database Updates

In theory, Agile teams self serve – developers must create the package, and testers must validate the install across all Environments.

ARA Test Insights

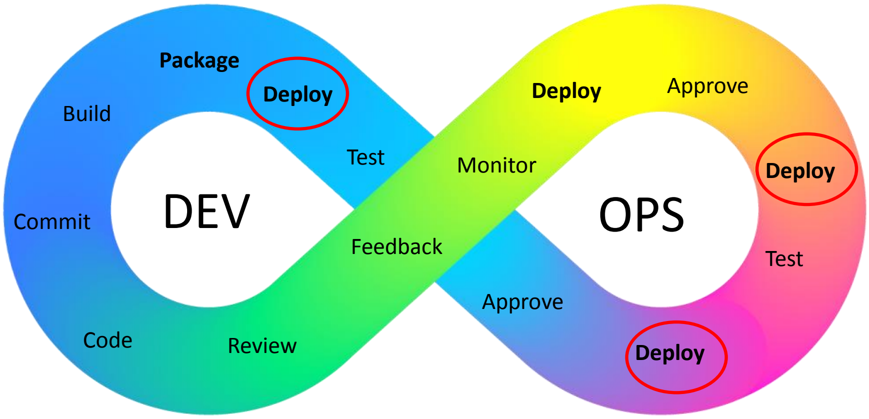
- Environment Configurations
  - What version of the infrastructure is available in each subsequent environment. ARA helps you validate versions of Java runtime, Tomcat, Oracle and even the OS level dependencies.
- Environment Variables
  - ARA allows you to view and confirm the correct variable substitutions before the apps deployed to each environment.

ARA Test Insights

- Track Artifact Versions
  - ARA will show you dependencies between binaries and define what versions are required – with change request and burn down rates. This shows how close an application is to its scheduled release date.
- Database
  - Includes the database as part of the process bringing the DBAs into the loop.

We are moving to a continuous Loop - not a pipeline

Package Once – Deploy Many



DEV

OPS

Build

Package

Deploy

Test

Commit

Code

Review

Feedback

Monitor

Deploy

Approve

Deploy

Test

Approve

Deploy

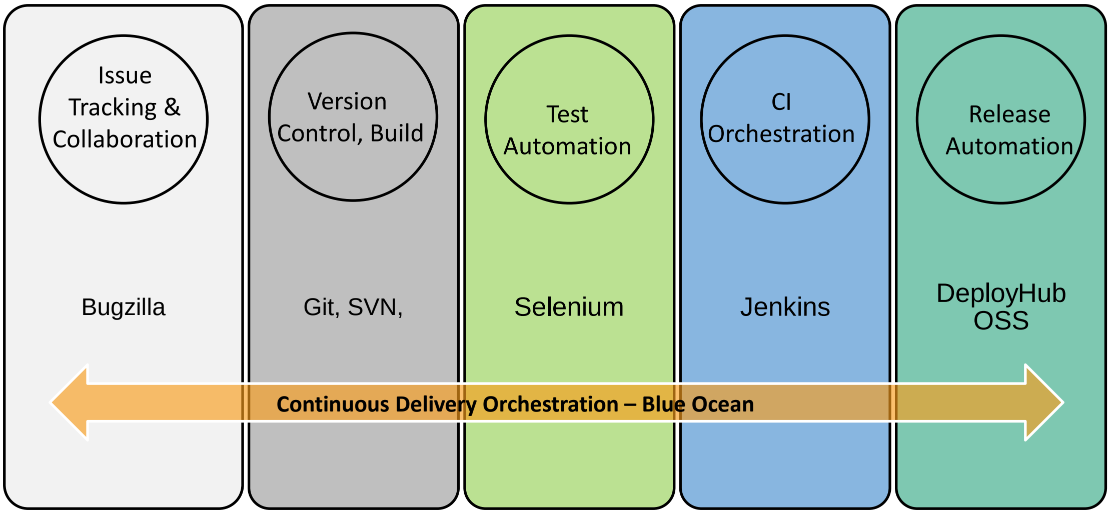
Test the Package for its ability to easily adapt to the changing environments, even when they are mixed.
A ‘rolling’ deployment can allow you to ‘smoke test’ at production. Containers make this even easier.

By taking on more responsibility around the validation of the software release, testing teams can better predict a software deployments success or failure, even before it is released.

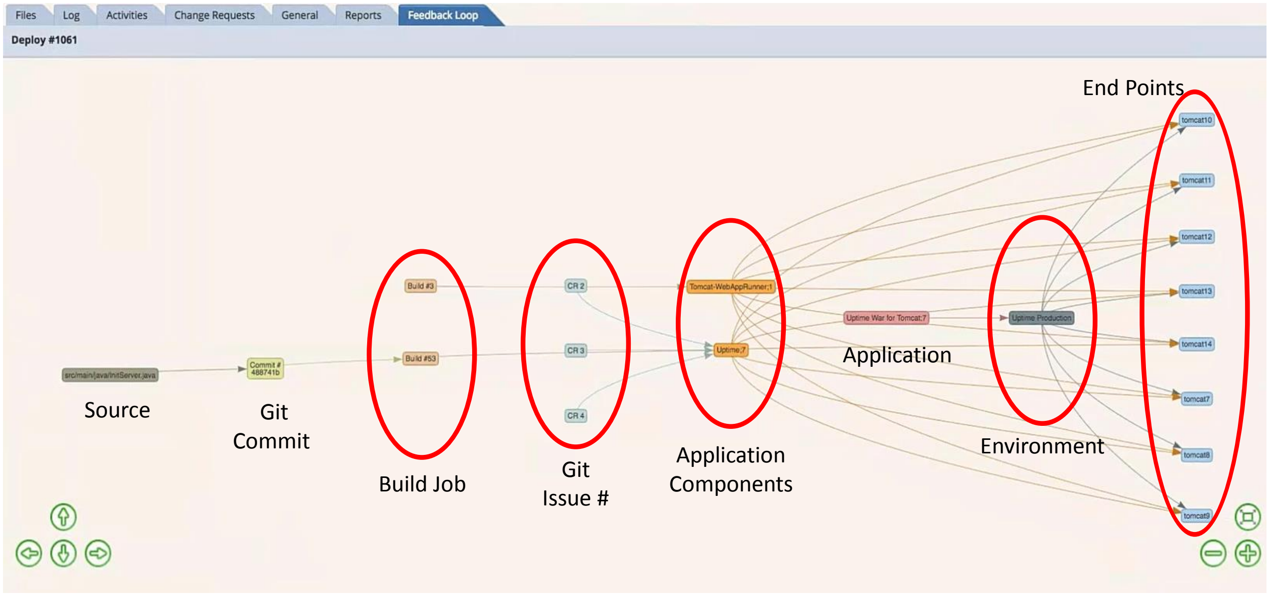
Test experts will become part of the site reliability equation.

No Budget Authority?

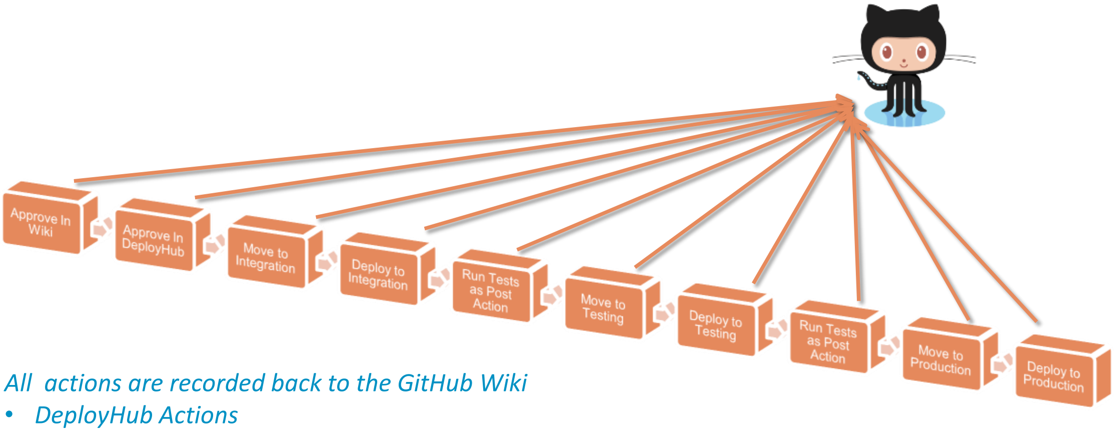
Open Source Tooling to get you started…



DeployHub/GitHub Continuous Loop



GitHub and WebHooks



All actions are recorded back to the GitHub Wiki

- *DeployHub Actions*
- *Build Logs*
- *Test Results*
- *Change Requests*
- *Bugzilla – Git Issue Tracking*

Questions? Answers?