Zero to Federated at the Speed of Jenkins

A Case Study of Success in DevOps

Richard Mills June 18, 2015

Jenkins User Conference

cdSummit Continuous Delivery Seminars

Zero to Continuous in 90 Days

A Case Study of Success in DevOps

Richard Mills June 18, 2015

Jenkins
User Conference

cdSummit
Continuous Delivery Seminars

Who is this guy?

- *Me*: Mad Software Developer turned Mad Software Engineer turned DevOps Solution Lead. Particular focus on tools and automation. CI, CD, DevOps … what's next?
  - PS: Thanks for inventing the term "DevOps" to describe what I like to do.

- *Pays my bills*: Coveros helps organizations accelerate the delivery of secure, reliable software using agile methods.
  - Agile transformations, development, and testing
  - DevOps implementations
  - Training course in Agile, DevOps, Application Security

Agility. Security. Delivered.

- *Keeps me intrigued*: SecureCI
  - Open-source DevOps product
  - Integrated CI stack with security flavor

Why is he here?

…to tell you a little about some successes we had introducing Continuous Delivery during a client engagement with a major media and advertising company.

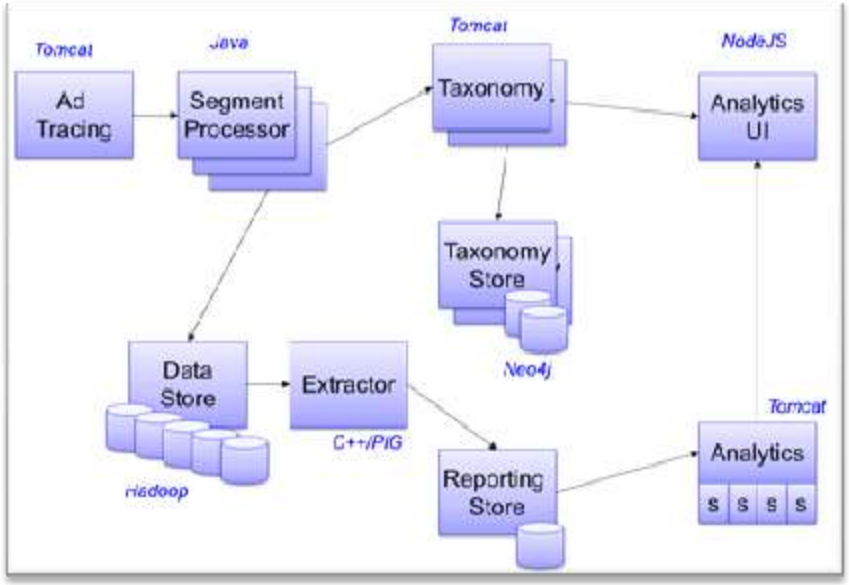
Where did we start?

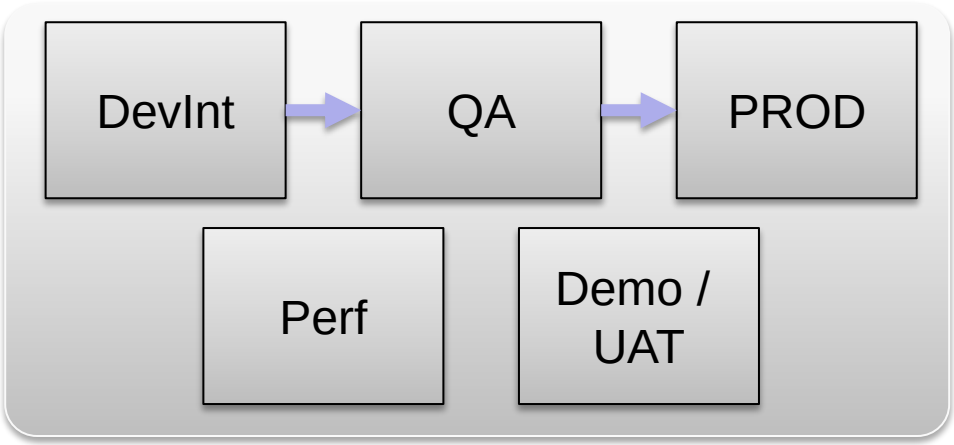
- Jenkins for continuous build
- Manual downstream activities
  - Testing and test environments managed by QA teams
  - Isolated siloes of development, QA, tech-ops groups
  - "… takes us 6 weeks to roll out completed software releases"
  - "… major pain configuring fleet of servers"
  - Long, hard-to-integrate release branches
  - "We want CI/CD"

- Two years of failed attempts to introduce Continuous Delivery (CD) into their development processes
- Green-field project to re-engineer the internal data management platform for processing advertising data
  - Approx 50 people distributed between US and Europe

What were we working with?

- Large, federated system with about 18 independent system components
  - Front-end UI, Tomcat REST web services, Java-based distributed data pumps, Hadoop back end
  - Installed into 3-5 environments ranging from ~15-40 servers each





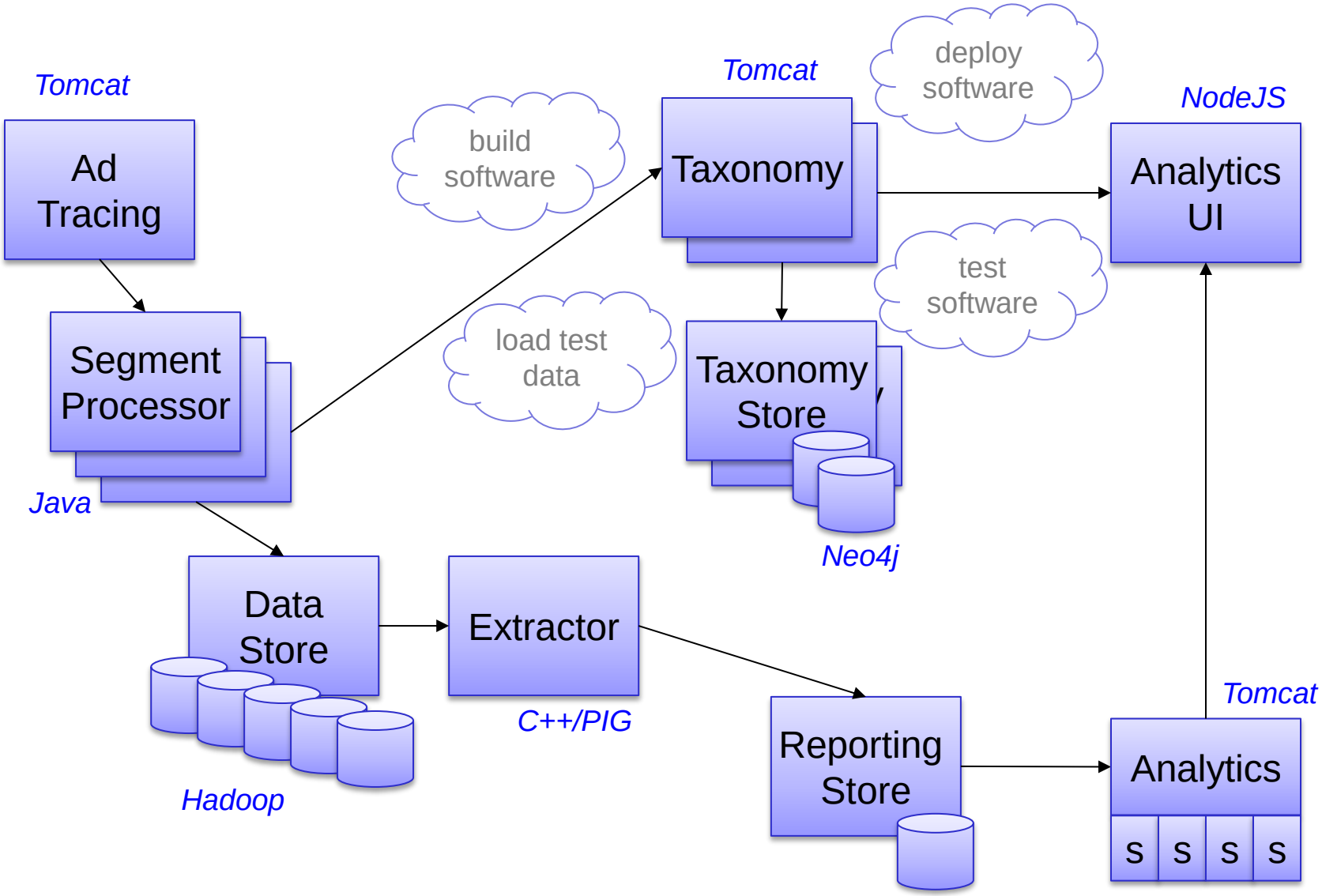
What did we accomplish?

- Continuous build, test, delivery of most of the systems through multiple environments
  - Automated Unit, Deploy, Smoke, Integration, Functional, Performance testing
  - Automated configuration (provisioning) of new hosts into the ecosystem
  - Push-button deployment to Production

- Benefits
  - Software delivered to DEV in minutes, TEST within an hour, PROD upon push-button approval
  - Developers able to develop and test across latest components
  - Developers rapidly identify problems, deliver functionality to PROD
  - 1-2 weeks from requirements to delivery to production

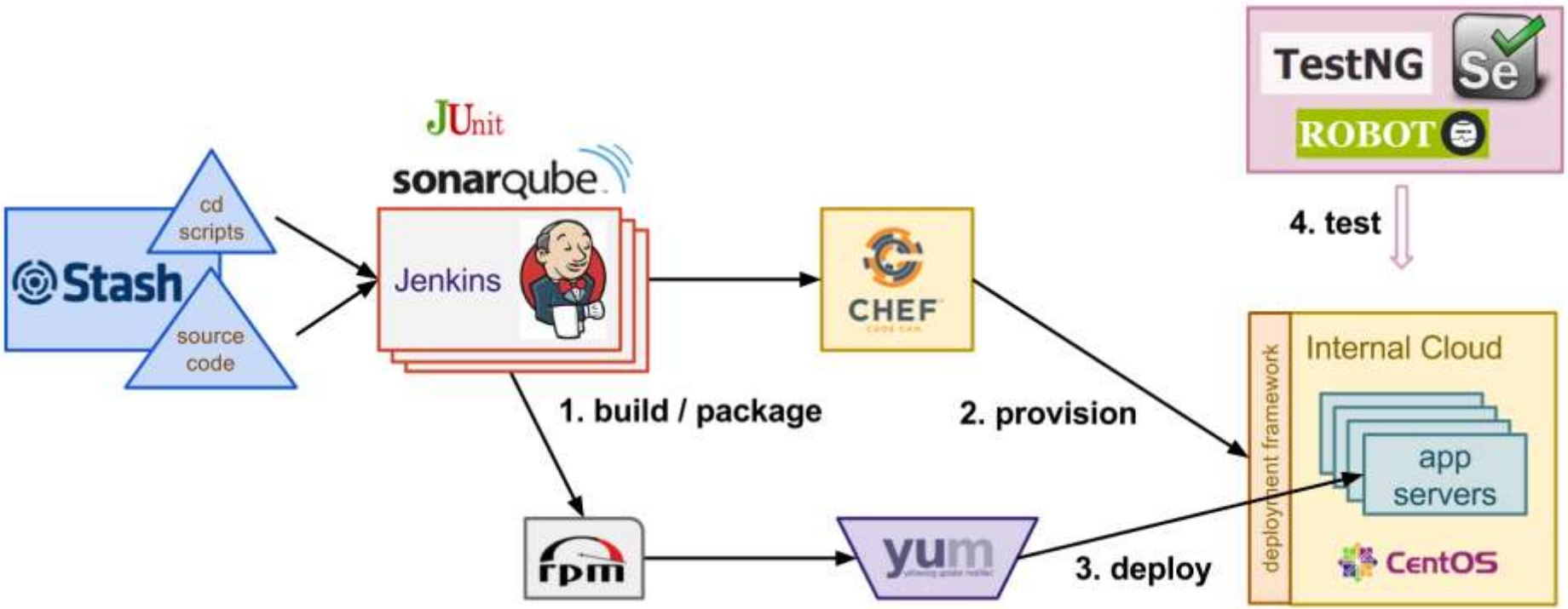
So… what were the challenges and how did we do that?

Federations Make for Complicated Delivery



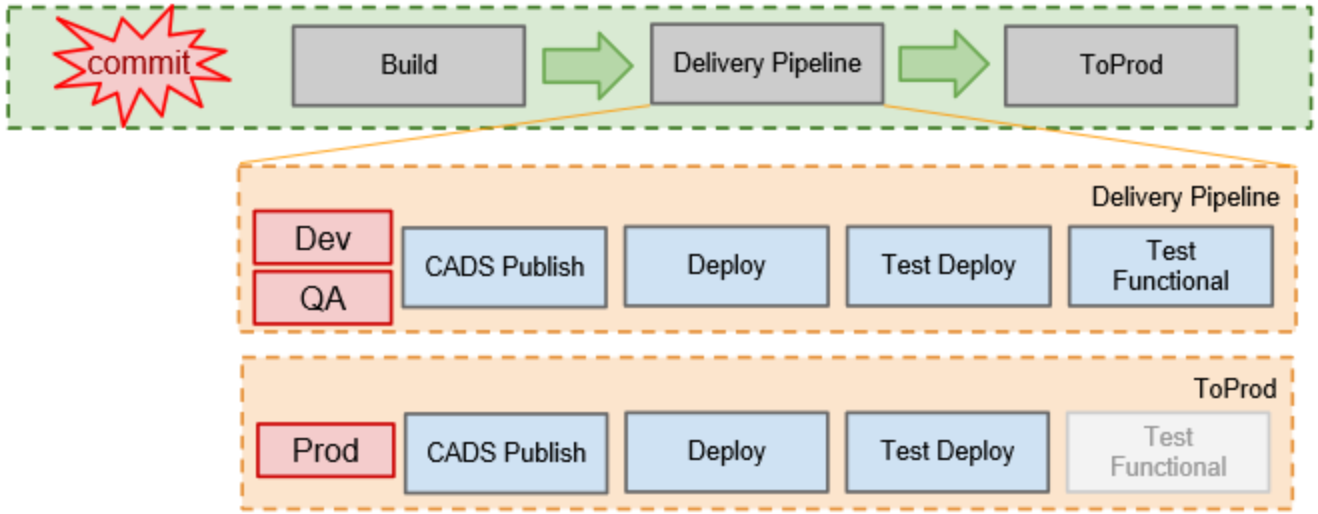
Our Solution

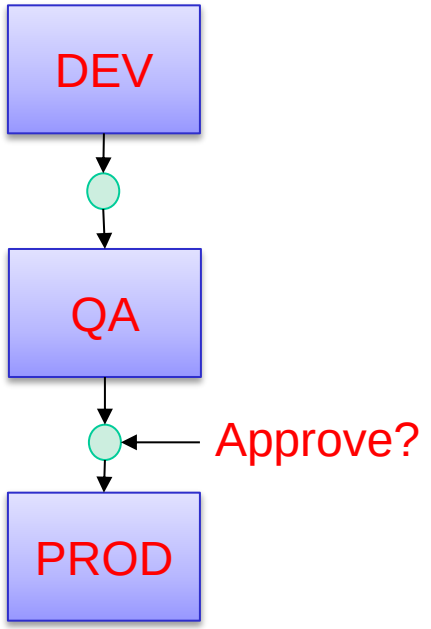
- Pipeline using Git, Jenkins, Sonar, Chef to deliver (mostly) Java-based solutions onto internally managed CentOS Linux cloud environment



Delivery Pipeline Design

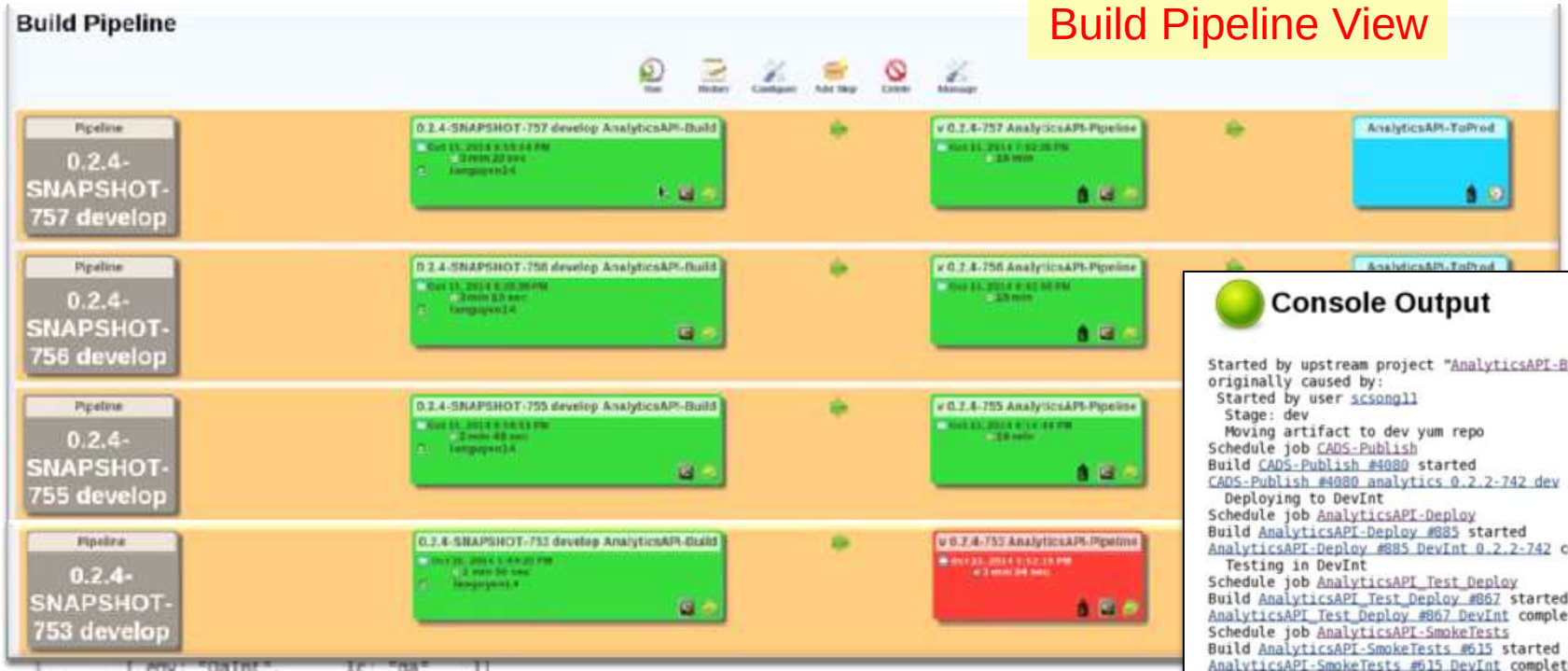
- A pattern emerged as we built independent pipelines for our federated system components





Pipeline Implementation in Jenkins

Build Pipeline View




```
envs.each() {
  print "  Stage: " + it.get('lc') + "\n"
  print "  Moving artifact to " + it.get('lc') + " yum repo \n"
  if( it.get('env') == "DevInt" ) {
    build( "CADS-Publish", ARTIFACT_NAME: "analytics", ARTIFACT_VERSION: params["ARTIFACT_VERSION"], ARTIFA
"dev" )
  } else {
    build( "CADS-Promote", ARTIFACT_NAME: "analytics", ARTIFACT_VERSION: params["ARTIFACT_VERSION"], ARTIFA
envs[envs.indexOf(it)-1].get('lc'), TO_LIFECYCLE: it.get('lc') )
  }
  print "  Deploying to " + it.get('env') + "\n"
  build( "AnalyticsAPI-Deploy", DMP_ENVIRONMENT: it.get('env'), ARTIFACT_VERSION: params["ARTIFACT_VERSION"
  print "  Testing in " + it.get('env') + "\n"
  build( "AnalyticsAPI_Test_Deploy", DMP_ENVIRONMENT: it.get('env') )
  build( "AnalyticsAPI-SmokeTests", DMP_ENVIRONMENT: it.get('env') )
  build( "AnalyticsAPI-FunctionalTests", DMP_ENVIRONMENT: it.get('env') )
}
```

Build Flow Plugin

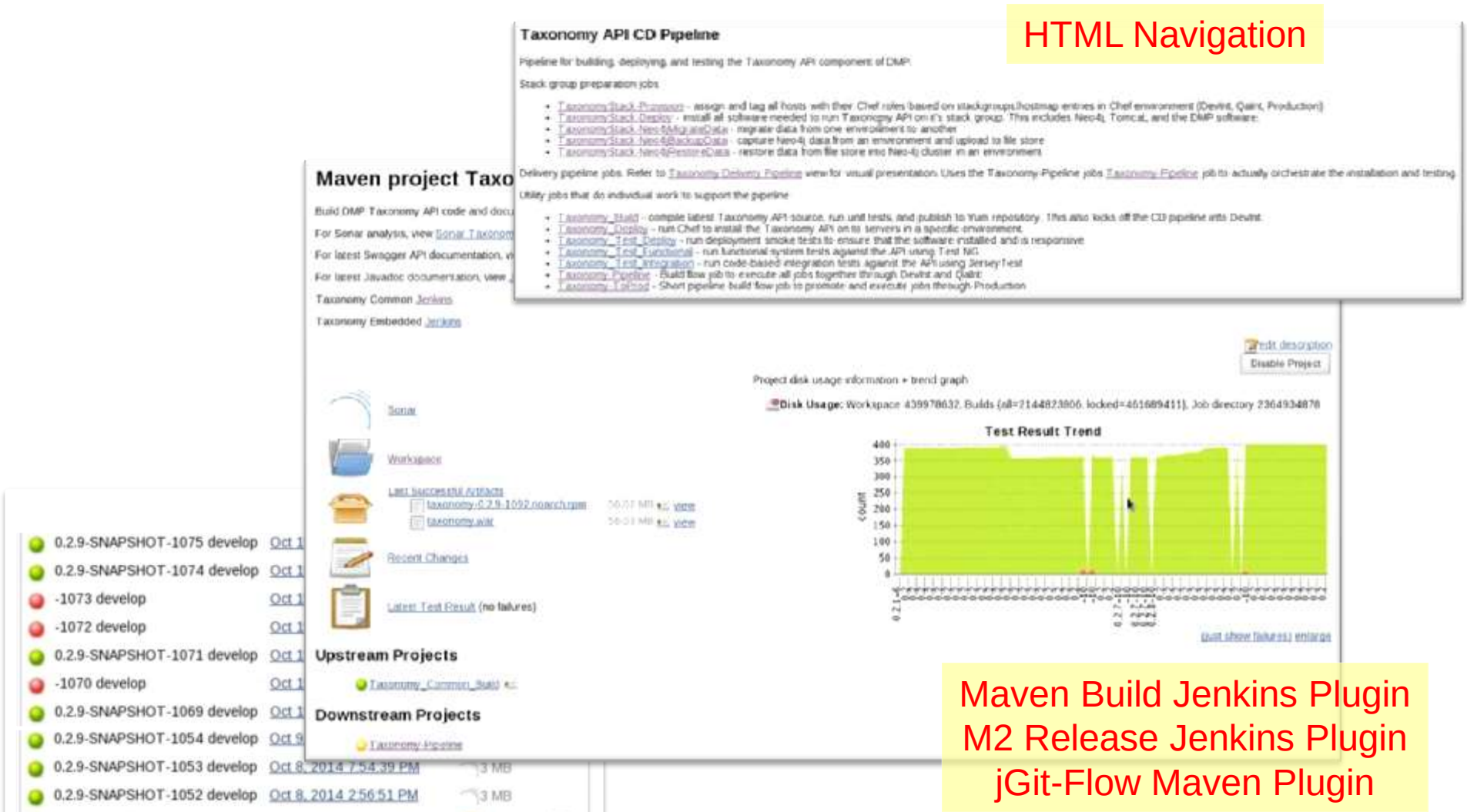
Console Output

```
Started by upstream project "AnalyticsAPI-Build" build number 742
originally caused by:
 Started by user scsong11
  Stage: dev
  Moving artifact to dev yum repo
Schedule job CADS-Publish
Build CADS-Publish #4080 started
CADS-Publish #4080 analytics 0.2.2-742 dev completed
  Deploying to DevInt
Schedule job AnalyticsAPI-Deploy
Build AnalyticsAPI-Deploy #885 started
AnalyticsAPI-Deploy #885 DevInt 0.2.2-742 completed
  Testing in DevInt
Schedule job AnalyticsAPI_Test_Deploy
Build AnalyticsAPI_Test_Deploy #867 started
AnalyticsAPI_Test_Deploy #867 DevInt completed
Schedule job AnalyticsAPI-SmokeTests
Build AnalyticsAPI-SmokeTests #615 started
AnalyticsAPI-SmokeTests #615 DevInt completed
Schedule job AnalyticsAPI-FunctionalTests
Build AnalyticsAPI-FunctionalTests #91 started
AnalyticsAPI-FunctionalTests #91 DevInt completed
  Stage: qa
  Moving artifact to qa yum repo
Schedule job CADS-Promote
Build CADS-Promote #1548 started
CADS-Promote #1548 analytics  0.2.2-742 completed
  Deploying to QaInt
Schedule job AnalyticsAPI-Deploy
Build AnalyticsAPI-Deploy #886 started
AnalyticsAPI-Deploy #886 QaInt 0.2.2-742 completed
  Testing in QaInt
Schedule job AnalyticsAPI_Test_Deploy
Build AnalyticsAPI_Test_Deploy #868 started
AnalyticsAPI_Test_Deploy #868 QaInt completed
Schedule job AnalyticsAPI-SmokeTests
Build AnalyticsAPI-SmokeTests #616 started
AnalyticsAPI-SmokeTests #616 QaInt completed
Schedule job AnalyticsAPI-FunctionalTests
Build AnalyticsAPI-FunctionalTests #92 started
AnalyticsAPI-FunctionalTests #92 QaInt completed
Finished: SUCCESS
```

A Note on Jenkins Orchestration with Chef

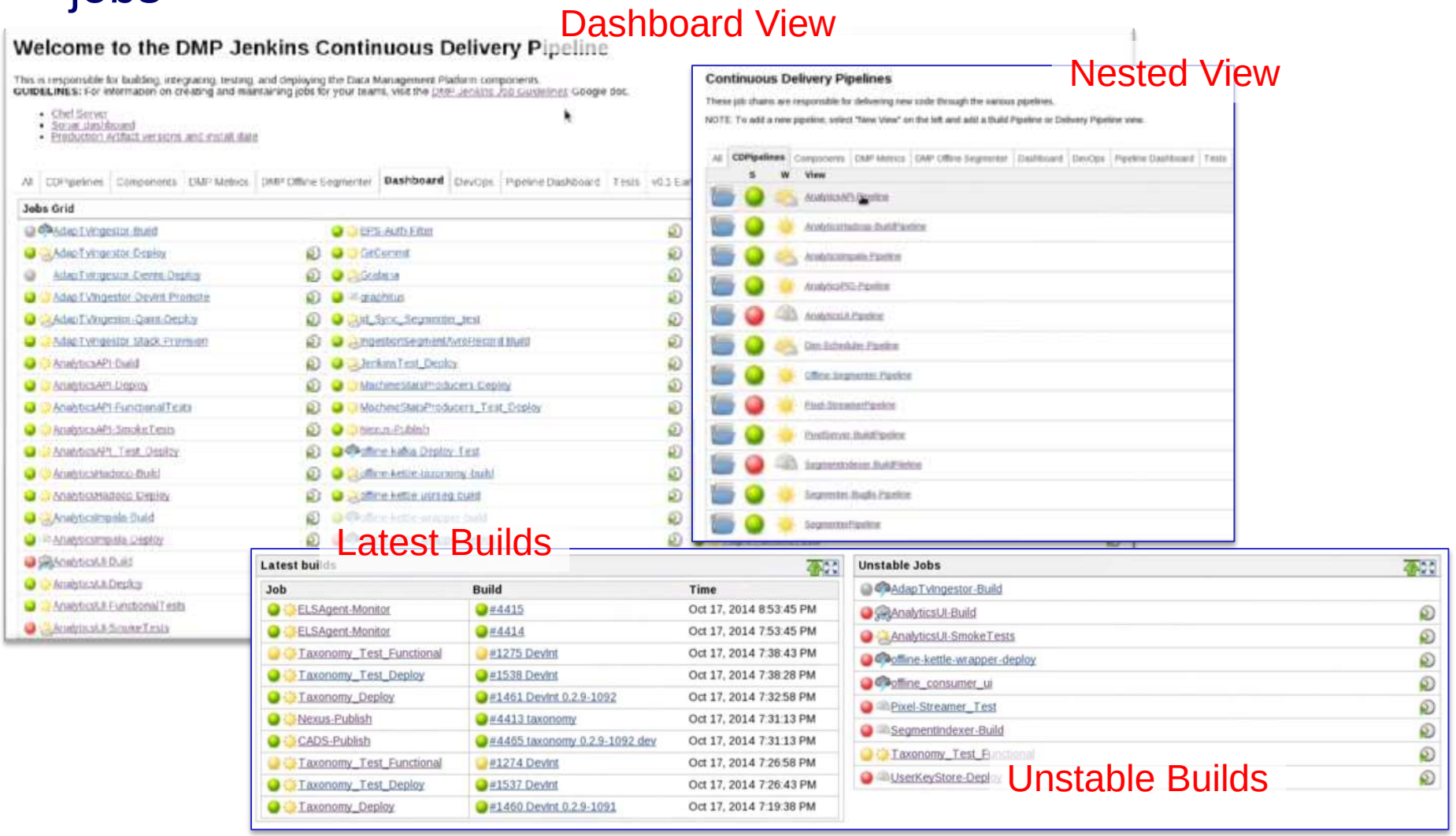
- Often, the hardest part of CD is orchestrating the configuration of multiple sets of servers
- We used Chef “environments” to map a host name to a Chef tag and Chef role
  - Jenkins use Chef **`knife`** script to read the environment data
  - Jenkins used Chef **`knife`** to assign tags and roles during one-time “provisioning” phase
  - Jenkins used Chef **`knife ssh`** commands to execute **`chef-client`** based on a tag search to execute assigned roles/recipes
  - Jenkins used **`knife`** to write versions and configuration info into Chef data bags

Component Build Implementation



Central Jenkins Dashboard

- Digesting a multi-node Jenkins CD system with LOTS of jobs



Proudest Moment. Ever.

Scene: At the end of a well-played demo of the brand new UI by the product team after a week long "hack-a-thon" with 20 Dev/QA/Ops guys in a room…

CTO: "That was an awesome demo … and it's all being done with CICD, right? *<awkward laughing>*"

Me: "Yes. Of course."

CTO: "Really?!"

Me: "Yes. We built the automation with the dev team while they were writing the code."

CTO: <shocked look> "Damn. That really IS awesome."

Organizational Factors of Success

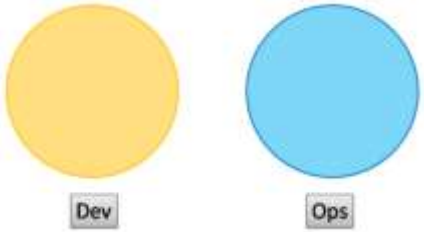
- Integrated, dedicated personnel
  - Ops … machine configuration, admin skills
  - Dev … code ownership, unit tests, configuration
  - QA … test code, rapid feedback

- DevOps “boot-strap” skillset (fill the gap)
  - Integrated DevOps mentoring
- Remove barriers: get people what they need when they need it (machines, tools, network access, whatever…)
- Automated testing is critical
  - Build confidence in your application
  - Continuous and incremental improvement: ratchet up quality goals that can’t be hit on day one

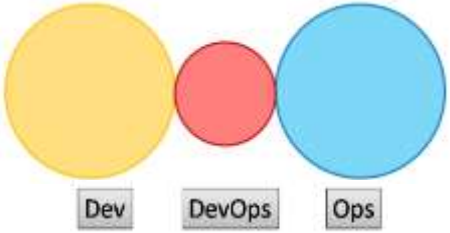
Organizational Structure – DevOps Anti-Patterns

Problematic

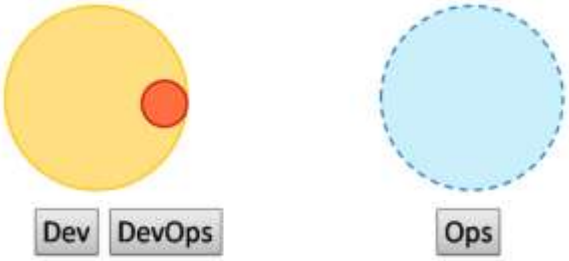
- Separate silos



- Dev, Ops, DevOps silos



- No Ops



Desired

http://blog.matthewskelton.net/2013/10/22/what-team-structure-is-right-for-devops-to-flourish/

Getting Started with Continuous Delivery

- Automate the most frequent (or painful) thing(s) first
  - Compile
  - Unit test
  - Deploy/configure your app
  - Test your deployment
  - Deploy/configure middleware
  - Configure new servers
  - Launch/harvest dynamic servers

- Don't forget automated testing!
- Use your build server as the center of everything (initially)
- Eventually: figure out how to "engineer" your pipeline

Thank you for your time!

Questions?

Contact Information:

Richard Mills
rich.mills@coveros.com
703.585.8961
@armillz

coveros
Agility. Security. Delivered.

www.coveros.com