CONTINUOUS DELIVERY AND LARGE MICROSERVICE ARCHITECTURES

Reflections on IonCannon

Kevin Scaldeferri
OSCON 2015
BUILT MICROSERVICE ARCHITECTURES

WHEN THEY WERE STILL SOA
MICROSERVICES
to show you ads
GILT

MICROSERVICES

to sell you clothes
MICROSERVICES
to store and analyze your data
7 For All Mankind

The denim mainstay’s hottest seasonal styles, from classic skinnies to bright cut-offs and more

Shop this Sale

Luxury Vault: Vintage Hermès, Chanel & More

Rosegold Shoes

OFF-DUTY DAD

Gifts for the guy who likes to keep it casual and cool. Shop 7 For All Mankind, Trafalgar, Aqua by Toscano, and more.
A CLASSIC GROWTH STORY
2007 - 2009
MONOLITHIC RAILS APP
MICROSERVICES
2009 - 2011

CHAOS

• Multiple Languages

• Multiple Build Systems

• Multiple Service Frameworks

• Multiple Deploy Processes
STANDARIZATION

• One language (Scala) - early 2012
• One set of core libraries - mid 2012
• One build system - late 2012
• One deploy system
DEPLOY

ALL THE THINGS
GETTING FROM HERE TO THERE
THE LANDSCAPE

• Laptops
• Team Shared Dev Machines
• Integration / Staging
• Production
RELEASE
DEPLOY TO STAGING
TEST
THE PRODUCTION SCANNER
my-repo % sbt functional:test

another-repo % bin/integration_test
RELEASE
DEPLOY TO STAGING
TEST
DEPLOY TO PROD
POLITICS AND CULTURE
WHAT ABOUT QA?
WHAT ABOUT RELEASE MANAGEMENT?
WHAT IF THE TEST IS 
/bin/true?
THE FRONT-END / BACK-END DIVIDE
THE HEDONIC TREADMILL
“Everything is amazing and nobody’s happy”

–Louis C.K.
IMPLEMENTATION
V0:
JENKINS AND DUCT TAPE
JENKINS

- Fast to get off the ground
- Eliminated C&P capfiles
- Terrible UI
- One release at a time
V1: J.U.C. STYLE
TASKS AND EXECUTORS

- Straightforward code
- Any node inserts to the DB
- ZK Leader Election
- Poll DB for new releases
- Easy to get some parallelization
V2: AKKA FSM
AKKA FSM

- Seemed like an elegant match
- No more polling, just create an actor for each release
- Replaced leader election with distributed locks
AKKA CHALLENGES

• Loss of type safety
• Much harder to debug
• Awkward to restart actors
• FSM got increasingly convoluted
FEATURES ADDED ON THE WAY

• Forced deploys

• Canary deploys

• Schema migrations

• Artifact Archiving
REFLECTIONS
THE GOOD
## Summary of Latest IonCannon Activities

<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
<th>State</th>
<th>Targets</th>
<th>Released By</th>
<th>Released At</th>
<th>Deployed At</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc-mobile-sale</td>
<td>1.9.9</td>
<td>released</td>
<td>mobile1</td>
<td></td>
<td>Wed Sep 18 2013 12:32:22 PDT</td>
<td></td>
</tr>
<tr>
<td>svc-user</td>
<td>1.2.5</td>
<td>in_test</td>
<td>svc7</td>
<td></td>
<td>Wed Sep 18 2013 12:31:13 PDT</td>
<td></td>
</tr>
<tr>
<td>svc-mobile-search</td>
<td>1.6.18</td>
<td>promoted</td>
<td>mobile1,mobile2</td>
<td></td>
<td>Wed Sep 18 2013 12:28:44 PDT</td>
<td></td>
</tr>
<tr>
<td>svc-image</td>
<td>2.1.8</td>
<td>abandoned</td>
<td></td>
<td></td>
<td>Wed Sep 18 2013 12:00:17 PDT</td>
<td></td>
</tr>
<tr>
<td>svc-digital-inventory</td>
<td>0.3.8</td>
<td>in_production</td>
<td></td>
<td></td>
<td>Wed Sep 18 2013 11:56:33 PDT</td>
<td>Wed Sep 18 2013 12:02:49 PDT</td>
</tr>
<tr>
<td>svc-payment-manager</td>
<td>0.1.0</td>
<td>in_production</td>
<td></td>
<td></td>
<td>Wed Sep 18 2013 11:37:00 PDT</td>
<td>Wed Sep 18 2013 11:43:59 PDT</td>
</tr>
<tr>
<td>order_service</td>
<td>4.4.0</td>
<td>in_production</td>
<td></td>
<td></td>
<td>Wed Sep 18 2013 11:12:09 PDT</td>
<td>Wed Sep 18 2013 11:22:12 PDT</td>
</tr>
<tr>
<td>svc-invoice-detail</td>
<td>0.0.39</td>
<td>in_production</td>
<td></td>
<td></td>
<td>Wed Sep 18 2013 10:50:07 PDT</td>
<td>Wed Sep 18 2013 10:58:05 PDT</td>
</tr>
<tr>
<td>svc-digital-inventory</td>
<td>0.3.7</td>
<td>in_production</td>
<td></td>
<td></td>
<td>Wed Sep 18 2013 08:02:43 PDT</td>
<td>Wed Sep 18 2013 09:08:39 PDT</td>
</tr>
</tbody>
</table>
THE BAD

• Legacy systems left out (Rails monolith, etc)

• Bleeding-edge systems left out

• web-ioncannon vs svc-ioncannon

• No streaming logs
THE UGLY

Staging environment:

Trying to run 300 services (& databases) on one box
THE UGLY

Modeling the full complexity of the problem as an FSM
THE UGLY

Self-deployment
THE UGLY

• What can be parallelized?
  • RPM publication and promotion - Absolutely
  • Production deploys - Mostly
  • Staging deploys and testing - hmm…
THE DEPENDENCY TREE
THINGS I’D DO DIFFERENTLY
SPEND MORE TIME TALKING TO TEAMS ABOUT THEIR UNIQUE REQUIREMENTS
MORE EMPHASIS ON VISIBILITY AND “FAIRNESS”
MAKE STAGING MORE LIKE PRODUCTION
DITCH YUM
BETTER UI
K.I.S.S
THANK YOU

kevin@scaldeferri.com
@kscaldef