The Road Taken

Zsolt Szilárd Sztupák
Gamesys Ltd.

@sztupy
http://sztupy.hu
Audience

Developer / Architect

Operations / Architect

Manager
WORKED FINE IN DEV

OPS PROBLEM NOW
Approaches

Top-to-bottom

Bottom-up
Gamesys

- Online gaming
  - Slots, Casino, Bingo

- >1000 employees
  - 40% technical

- Java based
Road to DevOps
Road to DevOps
Road to Devops

• Monthly releases
• Downtime during release
• Costly meetings between teams
• Communication issues
• Late integration issues
Road to Devops

• We need a way to not have this
• We need to split up the monolith
• Let’s create a platform that allows people to create microservices easily
• We need to deploy these microservices
• Operations is overloaded with the manual deployments
• Let’s help them automate their processes
• While we’re there, let’s automate what we can on the dev’s side
Case Study

Ruby upgrade 1.8 -> 1.9

Python 2 -> Python 3
Java 1.7 -> Java 1.8
C++03 -> C++11

Windows XP -> Windows 7
Office 2003 -> Office 2007
Android 2.3 -> Android 4.0
Case Study

- Please upgrade ruby
  - Can’t do we’re still on RedHat 6

- Can you install it from other rpm
  - No available ones for RedHat 6

- Can you compile from source?
  - Not allowed to install gcc on live machines

- Can we compile on staging and move it to live?
  - They have slightly different libs and it segfaults
Case Study

1-2 weeks
• Please upgrade ruby
  • Can’t do we’re still on RedHat 6

1-2 weeks
• Can you install it from other rpm
  • No available ones for RedHat 6

1-2 weeks
• Can you compile from source?
  • Not allowed to install gcc on live machines

1-2 weeks
• Can we compile on staging and move it to live?
  • They have slightly different libs and it segfaults
Case Study

6 months
Case Study

6 months

5 months and 3 weeks
Case Study

Ruby upgrade 1.9 -> 2.1
FROM ruby:1.9.3  

\[ \Sigma 1 \text{ week} \]  

FROM ruby:2.1.5
Case Study 2

Java upgrade
Case Study 2

Security
  Oh, new version

Release Management
  Plan for release

Operations
  Assign engineers

  Engineer
    ssh <machine>

  Engineer
    then

  Engineer
    Do upgrade
Case Study 2

Security
- Oh, new version

Release Management
- Plan for release

Operations
- Assign engineers
  - Engineer
    - ???
  - Engineer
    - ???
  - Engineer
    - ???
Case Study 2

1. https://google.co.uk
2. How to access docker machines
3. $ ssh <box>
4. $ docker exec <container>
5. $ install-java-update.sh
Case Study 2

Developers

Security
   Oh, new version

Release Management
   Plan for release

Operations
   Assign engineers

   Engineer
   Engineer
   Engineer

   ???
   ???
   ???
Case Study 2

A new version of Java is available!

Java 8 Update 131 build 11 is now available—you have Java 8 Update 121 build 13. Would you like to download it now?

Release Notes:

Computer restart not required, however your browser or application will need to be relaunched.

Finish your work and then click Install Update to continue.

More information

Skip This Version  Remind Me Later  Install Update
Road to DevOps

TECHNOLOGY RADAR VOL. 16
Insights into the technology and trends shaping the future

Select an area to explore

Techniques   Tools
Platforms   Languages & Frameworks

Search   About the Radar
Platform team

- Built up from members of different teams
- Set up to drive the move to microservices
- Have more leeway in assessing new technologies
- Create a common platform
  - Dropwizard
- Drive a standard way of specifying our APIs
  - HATEOAS -> HAL
  - Use Apigee as the proxy
- DevOps wasn’t initially part of the setup
Virtual API Team

- Idea from Brian Mulloy
- Standardization of the API requires domain knowledge
- We need members from each domain team
- Meet every week
- Decide on the standards of the APIs
- Assess and make sure every new API adheres to these standards
Ansible, Docker, GOCD

- Teams started to create microservices
- There was no easy way of deploying them
- Let’s introduce containers
- Also introduce an easy way of deploying them
  - Make sure the artifact is actually the deployment instructions
  - Make sure it’s as similar on local / staging and live as possible
- Also introduce an easy way of building them
  - Using GoCD as the CD framework
  - Make sure everything is in git, and it builds automatically
Legacy-in-a-box

- Another item from the Technology Radar
- Microservice adoption was still slowed by the monolith
  - Hard to set up and test
  - Also hard to deploy
- Try to change the monolith to look like a microservice
  - Or at least make sure it can be deployed the same way as one
Vertical Team Structure

- Backend and frontend teams were separated before the change
- We realized that it’s not optimal
- Teams were effectively going at different speeds
- Frontend and backend teams were merged into verticals
  - They were related to various domain concerns
  - With the exception of the Platform Team
Platform team

Pros:
• Quick to assess and introduce new technologies
• The standardised framework allowed quick introduction of new microservices
• Making the artifact the deployment instructions meant teams could automate their special deployment requirements as well

Cons:
• Wasn’t short lived
  • became a support team for the framework
• The pros weren’t communicated well enough
  • Developers thought they had to join the Platform Team to work with DevOps technologies
Virtual API Team

Pros:
• Made sure that knowledge around HATEOAS / HAL is dispersed into the teams
• Made lots of – documented – decisions around the standards

Cons:
• Initially when there were a lot of new microservices it was a huge bottleneck to check and approve all of the APIs
Ansible, Docker, GoCD

Pros:
• These are the backbone of the DevOps technologies

Cons:
• Upgrade strategy was missing
  • The more they were used the harder it was to upgrade them
• Backward compatibility with ansible is hard
Legacy-in-a-box

Pro:
• Removed the convoluted build and deployment process
• Made it easy to make changes to the legacy monolith

Cons:
• Made it easy to make changes to the legacy monolith
Vertical Team Structure

Pros:
• Domain knowledge is closer to each other
• Features can be done front-to-back

Cons:
• Platform team wasn’t really added to the verticals
• Communication between teams became non-existent
The future

- Re-define the Platform Team
  - Spread the knowledge
  - Involve other teams in DevOps concerns
  - Re-brand it as a true vertical for Security

- More automation
  - Kubernetes
  - Continuous Deployment where possible
    - Bit hard in a regulated environment
The future

  - ITV’s DevOps transformation
  - Key points:
    - Every area should own its own stack
    - Every area should upgrade its own stack at its own pace
The future

- DevOps Guild
  - Each area sends us a few members who are interested in DevOps concerns
  - We provide a common forum to raise questions and demonstrate new technologies
    - Builds on pipelines are slow
    - We don’t really know the health of our microservices
  - Provide trainings for the members
    - Docker
    - Ansible
    - Kubernetes

- The first few projects were
  - Separate GoCD for each area
  - Kubernetes deployments for each area
The future

• To set this up we needed a new view on security
  • We had a lot of department-wide credentials used
  • If we wish to give more access to the areas we needed to make them area-based

• Area Root
  • Root provision scripts for your area’s VMs
    • Contains setup scripts for the GoCD server
    • Contains setup scripts for the GoCD agents
    • Contains setup scripts for a dev kubernetes cluster
  • All of these scripts set up the machines using credentials which are area based
The future

• So far feedback looks good
• Already opened up some of our areas to do R&D on their own
• Main concern: migration
Questions?

*We are hiring!
(ex-Gamesys employees hiding in the audience this applies to you as well)
Thank you