Architecture/UI

Roman Chyla & the ADS Team

ADS Users Group Meeting - 11/2/207

Recall when we were young….

(9 months ago in a far far away galaxy)

2 major problems:

- **How to scale up to 3000 reqs/s**
  - **(and don’t break the NASA budget)**
- **Accomplish data-parity with Classic**
  - **(which turned into: re-engineer the bl...y thing)**

Seems like we are in for another Happy-End...

Oh no! Not that again!

Scaling up (1.)

A story of how the ‘yesterday’s good’ became ‘today’s bad’

- 20 Amazon virtual machines
- 40-60 reqs/s
- Reaction time: minutes

- Kubernetes (on Amazon)
- Thousands reqs/s
- Reaction time: seconds

Scaling up (2.)

1. Write a very detailed script
   - Specifications 3000 reqs/s
   - Test technologies
   - Discover Kubernetes
2. Hire a dedicated devops engineer
3. Rest

Scaling up (3.) - unexpected gift

- When 3 pairs of eyes are not enough
  - You need glasses
- Removing shortcomings from internal api gateway
  - External libraries
- Discover the real bottleneck
  - And hopefully fix it
    - ...and discover another one, yay!

Data parity - a.k.a. backoffice pipeline

- Why a new one?
  - The old was ugly
  - It was slow
  - Did I say it was ugly?
- Re-engineered
  - Still based on brokering/messaging (Celery)
  - Standardized messages (Google Protocol Buffers)
  - Standardized libraries (ADSPipelineUtils)
  - Centralized (one master pipeline to rule ‘em all)
  - Modular (we now have 5 pipelines and new ones are coming)

Current pipeline status

- In testing
  - Very complex undertaking
  - “Almost production-ready” for the past 3 weeks (fix, re-run, repeat)
- Fast and getting faster
  - 48 hours (the old) vs 12 hours (the new) ← and that’s full re-ingest
  - Will get better still…
- More robust
  - Better logging, control
- Will exist when ADS Classic is no more

… and why is speed so important?

Bumblebee UI/UX Focus

- Error Handling
  - Improve way site reacts to slowdowns and server-related issues
  - Improve how/when users are made aware of errors
  - Improve messages, identifiers, and other visual cues
- ORCiD
  - Updated to 2.0 API
  - Faster, more error-resistant
- Under-the-Hood
  - Transition to newer framework -> Faster, more responsive experience
  - Cleanup and slimming down -> Better maintainability

Kicked the can further down the road...

(still paying the technical debt)

- SOLR
- Microservices architecture

- And where’s R&D?