Next Gen Repository: Breaking up to get back together

digitalcollections.library.northwestern.edu
2016-17 Landscape

- Designing a Digital Collections Repository (Next Gen Repo)
- Building our IR
- Campus had just signed a contract with AWS and were looking for volunteers (Guinea pigs)
- NUL devs were really interested in exploring the possibilities of “the cloud”
Everything starts as skunks works

- Scope a project small enough to get done quick
- Make sure it’s big enough to prove it out
- Give the team permission to fail
Opportunity in redesign

- Multiple repositories are a single “solution”
- Frontend and backend don’t have the same needs
- Cloud based architecture allows you to rethink everything
- Architecture should be iterated on as fast as software
Before you break it apart put it together

- Think of the “solution” as one big stack with a bunch of frontends
- Single Database
- Single Fedora (for now)
- Single SOLR (later elasticsearch)
Hard problems that someone else solved

- Move services that are hard/require a lot of resources and are commodified:
  - Transcoding (elastic transcoder)
  - Storage (s3)
  - Queuing (Redis)
  - Databases
  - Streaming
  - Monitoring
Hard problems we built custom solutions for

- IIIF as a serverless application (lambda)
  - Blazingly fast
  - Uses off the shelf parts
  - No servers to maintain
- S3 backed Fedora
User-facing should stand alone

- Break out a user-facing application that focuses on the needs of patrons
- Separate it as much as possible from the backend
- Changes to the backend can happen without affecting the frontend
- Heck, host it in a s3 bucket without a server
What we had to unlearn

- Many (dang near all) digital repository solutions are monolithic in nature.
- Monoliths scale monolithically
- Solutions assume some kind of local storage
People and Jobs

- A cloud native environment allows you to do multiple sets of retooling, (continuous improvement)
- Moving to the cloud requires a culture shift
- Sudden acquisition of knowledge (firehose)
- People get scared
- Investment in training is key
How Did This Turn Out?

- Budget planning and projection
  - Opex rather than Capex
  - Actionable insights into costs
- Effect on throughput
  - Limited by budget
  - Can be rethought in terms of per job
- How’s the team doing
Thanks!

Questions?