Building a robust content recommendation platform for 60 million news readers

Or,
Content Recommendations to Make Readers Content

Matt Chapman, Manager, Data Engineering, mPulse Mobile
PIVOTING

THE SPECS SUFFERED MINOR CHANGES OVER TIME

DEPRECATED

HOPE

MONKEYUSER.COM
“Kaboom!”

That’s the word used by Joe Rohde, the Disney executive overseeing the transformation of the California Adventure ride formerly known as the Twilight Zone Tower of Terror into Guardians of the Galaxy: Mission Breakout!

“Kaboom!”

He says it repeatedly — once mixing in the word “kablam!” — occasionally with his arms outstretched to convey his excitement. His ornate, beady earrings, which hang below his jaw, rattle each time.
Lean Start-up Methodologies

- My PoV as a solo Engineer working with 2 Data Scientists
- May not work for large technology teams
- Reliable failover is essential
Some Requirements

- Offline Learning

```
Collect -> Train -> Deploy
```

- Inputs
- Model
- Prediction
Some Requirements

- Online Learning
Some Requirements

- A/B/n Testing
Some Requirements

- Horizontal Scalability
How a Page becomes a Personalized Experience
Web Page (NGUX)

HTTP(S)
HTTP(S) - Ads
HTTP(S) - Videos
HTTP(S) - Video Ads
Why Python?

- Data Scientist Friendly
- Rapid Prototyping
- Machine time is cheaper than Developer Time
Why not Kubernetes?

- It was not yet the clear winner for Container Orchestration.
- It was easier to deploy.
- You should probably use Kubernetes.
- Unless:
  - You have Mesos Expertise.
  - You have a clearly demonstrated advantage in running Spark on the same physical nodes as your services.
  - You want to run a bunch of isolated Kubernetes Clusters.
Incremental Monitoring

- Start with the High Level Metrics that Matter
  - E.g., Click-through rates
- Add low level metrics when troubleshooting
- Scale the tools too
  - E.g., Graphite, then Grafana
Web Page (NGUX) -> AWS ELB

HTTP(S) -> REC API
(1) Javascript
(2) HTTP(S)

REC API
(1) Sends JS
(2) Handles Requests for Recommendations

Monitoring

DCOS
Web Page (NGUX) ➔ AWS ELB ➔ REC API

(1) Javascript
(2) HTTP(S)

DCOS

REC API
(1) Sends JS
(2) Handles Requests for Recommendations
(3) Splits Traffic for A/B testing

Monitoring

tronce
Web Page (NGUX) → AWS ELB

REC API
1. Sends JS
2. Handles Requests for Recommendations
3. Splits Traffic for A/B testing
4. Routes to consistent Backend

DCOS

Recommendation Algorithms

Monitoring

tronic
Microservices != ReST or HTTP
**Rec API**
1. Sends JS
2. Handles Requests for Recommendations
3. Splits Traffic for A/B testing
4. Routes to consistent Backend

**Recommendation Algorithms**

**Monitoring**

**AWS ELB**
1. Javascript
2. HTTP(S)
3. HTTP

**Web Page (NGUX)**

**DCOS**

**tronic**
REC API
1. Sends JS
2. Handles Requests for Recommendations
3. Splits Traffic for A/B testing
4. Routes to consistent Backend
5. Gets Details from C.S.

DCOS
Recommendation Algorithms

Web Page (NGUX)
AWS ELB
Content Services
Monitoring

tronce
Web Page (NGUX) to AWS ELB:
1. Javascript
2. HTTP(S)
3. Rec. HTML

AWS ELB to REC API:
- HTTP(S)
- HTTP

REC API:
1. Sends JS
2. Handles Requests for Recommendations
3. Splits Traffic for A/B testing
4. Routes to consistent Backend
5. Gets Details from C.S.

DCOS to Recommendation Algorithms:
- (3) HTTP
- (4) ZMQ

DCOS to Monitoring:
- HTTP

DCOS to Content Services:
- HTTP
**REC API**
1. Sends JS
2. Handles Requests for Recommendations
3. Splits Traffic for A/B testing
4. Routes to consistent Backend
5. Gets Details from C.S.
6. Handle Ensihten updates

**AWS ELB**
- (1) Javascript
- (2) HTTP(S)
- Rec. HTML

**Web Page (NGUX)**
- HTTP(S)
- HTTP

**Jenkins (6)**
- HTTP

**DCOS**
- Monitoring
- Recommendation Algorithms
- SQL

**Backbone**
- Ensihten Last Hour

**Content Services**
- HTTP
**REC API**

1. Sends JS
2. Handles Requests for Recommendations
3. Splits Traffic for A/B testing
4. Routes to consistent Backend
5. Gets Details from C.S.
6. **New!** Handle Model Updates

**DCOS**

- Recommendation Algorithms
- Model Updates
- Monitoring

**Web Page (NGUX)**
- HTTP(S)
- Javascript
- HTTP(S)
- Rec. HTML

**Apps**
- HTTP(S)
- Rec. JSON

**AWS ELB**
- HTTP

**Content Services**
- **HTTP**
Centralized Training

- Horizontal Scalability
Centralized Training

- Predictor Service Instances
  - State
- Accumulator
- Trainer
  - State
- Updater Service Instance
  - State
- Tensorflow / Spark
  - State
Centralized Data Store options

- Specify Requirements: Key-value store with non-critical data
- Generate Options
  - Cassandra
  - Memcached
  - Redis
  - Scylla
- Develop sample data set & queries
- Determine Metrics
- Divide Options among team members
Centralized Data Store options (cont’d)

- ... Use the first one that works.
- Use the one where you have the most operational experience.
- Because Machine Time is cheaper than developer time.
Scylla

- Durable Writes: Off
- Consistency: One
Pickle all the Things
“Why Messaging Queues Suck” - Mulesoft

- They don’t, but
  - Don’t use them when they aren’t needed
  - Use them correctly (as buffer for consumer; w/ many producers)
- Myths about Pub-Sub with Queues
  - The Publisher doesn’t need to know anything about the consumers: **FALSE**
  - The Consumer doesn’t need to know how to connect to the Publisher: **It still needs to be able to connect to the Queue.**
- Allows late binding. **So do Web Hooks.**
- It scales better. **No, It just scales different.**
Pub-Sub using Topic and Message Queues

Direct to Endpoint

ØMQ
We did end up using Kafka

- As a buffer for the consumer
What is Cranial?

- The glue that puts it all together.
- An Ontology of Data Engineering for Machine Learning.
- 4 Python Packages hosted on Github.
  - Abstract Base Classes
  - AND Concrete Implementation for many common services and protocols.
- An Open-source Project initiated by the company formerly known as tronc.
- A **toolkit** to help Data Engineers help Data Scientists.
Integrated Technologies

- Any DBAPI2 Database (PostgreSQL, MySQL, SQLite, etc)
- Cassandra
- Amazon S3
- Google Cloud Storage
- HTTP APIs
- ZMQ
- Kafka
- Amazon Kinesis Firehose
- Celery
- TensorFlow
- SciKitLearn
- Spacy
- GenSim
- DCOS / Marathon
My Beliefs

- A Data Engineer’s role is to make Data Scientist’s lives easier.
- Data Scientists are smart enough to make Data Engineers lives easier with just a little guidance.
- Data Scientists should never need to think about where data lives, only what it is.
- Data Engineers should be doing more than writing ETL jobs.
- Data Storage and Transport Technologies should be abstracted so as to be easily replaced.
What Can Cranial Do for You?

- Accelerate the time-to-market of Data Models
- Promote standard interfaces & reusable code
- Abstract Solutions away from specific Data Storage and Transport Technologies.
Service: script(model, config) -> service

Model: (Record -> Record)
Service: script(model, config) -> service

Model: (Record -> Record)

Steps [1..n]
- fn(Record) -> Record
- Model
- fn(Record) -> Record
- Model

Record
Dict[str, List]
Service: script(model, config) -> service

Model: (Record -> Record)

Steps [1..n]
- fn(Record, State) -> Record
- Model
- fn(Record, State) -> Record
- Model

State [0|1]

Record
Dict[str,List]
Service: script(model, config) -> service

Model: (Record -> Record)

Steps [1..n]
- fn(Record, State)->Record

State [0|1]

Trainer [0|1]
- Local, Remote
- Accumulator [1]
- Scheduler [1]
  - Count, Time

Record
Dict[str,List]
Listener / Fetcher → Model → Putter

Extract → Transform → Load
Packages: github.com/tribune

- cranial-models
- cranial-messaging
  - Messenger & notifiers
- cranial-datastore
  - Fetchers & Connectors
  - KeyValueDB for DBAPI2 & Cassandra
- cranial-common
  - Low-level utilities & convenience functions
  - Logging
  - 12-factor App Configuration
mPulse Mobile, the leader in Conversational AI solutions for the healthcare industry, drives improved health outcomes and business efficiencies by engaging individuals with tailored and meaningful dialogue.

10 YEARS OF EXPERIENCE

70+ HEALTHCARE CUSTOMERS

>150M MESSAGES SENT ANNUALLY
Rate today’s session

Effective enterprise architecture

Who is this presentation for?
- Architects of all stripes, tech leaders, senior developers, and managers

Prerequisite knowledge
- Experience as a senior developer or architect working on software projects

What you’ll learn
- Learn a holistic approach to architecture that explains how to bring business architecture, information architecture, data architecture, and infrastructure architecture together to have the best chance for your system’s success
- Explore a practical set of architecture practices to create winning technical architectural guidance
- Understand how architecture works effectively with development teams, management, and product management teams through the value chain
- Get usable templates you can start incorporating into your teams immediately

O'Reilly Events App

Session page on oreillysacon.com/ny
Thanks!

Matt Chapman
mattchapman.ninja
matt@ninjitsuweb.com
@matt2000