Modelling Microservices

Petter Måhlén
February 2016
About me

- Building software professionally since 1995
- Infrastructure at Spotify since 2013
- Service Discovery (Nameless)
- Service Framework (http://spotify.github.io/apollo/)
- System-Z
- Currently doing Data infrastructure
About this talk

1. Why model microservices?
2. Our solution: System-Z
3. Design
4. Impact so far

=> Ideas about running microservices at scale
Why model microservices?

Y axis: Splitting

Z axis: Sharding

X axis: Cloning

http://artofscalability.com/
Why model microservices?

- X axis: Cloning
  - ~10k servers

- Y axis: Splitting
  - ~1100 things

- Z axis: Sharding

http://artofscalability.com/
Why model microservices?

- **Y axis: Splitting**
  - \(~1100\) things

- **Z axis: Sharding**

- **X axis: Cloning**
  - \(~10k\) servers

Plus \(~100\) teams writing code

If you need to know exactly who is making decisions, you are in the wrong place.

https://labs.spotify.com/2014/03/27/spotify-engineering-culture-part-1/
Problems to Solve

Discovering and understanding:

- What things are “out there”
- Deployments and configurations
- The system as a whole, how do things fit together?
- How to get more information: ownership
- What’s broken and how to fix it.
Our solution
What came before

Emil
What came before

Emil

ServiceDB
What came before

Emil

ServiceDB

System-Z
Enter System-Z

- A system for system metadata
- Z? Wat?
Some terminology

- Component - some thing
- A microservice, data store, data pipeline, or library
- System - some related things

These terms, like most things Z, are intentionally vague
Main features

● General component information
● Dependency Tracking
● Managing deployments
● Ownership and alerting
Apollofied aggregation search view service

[Link to documentation](https://ghe.spotify.net/kiwi/searchview/blob/master/README.md)

<table>
<thead>
<tr>
<th>INFO</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>owner</td>
<td>kiwi</td>
</tr>
<tr>
<td>system</td>
<td>search</td>
</tr>
<tr>
<td>visibility</td>
<td>private</td>
</tr>
<tr>
<td>service_discovery</td>
<td>searchview</td>
</tr>
<tr>
<td>slack_channel</td>
<td>kiwi</td>
</tr>
<tr>
<td>tags</td>
<td>kiwi, apollo, search</td>
</tr>
<tr>
<td>pagerdutykey</td>
<td>searchview</td>
</tr>
<tr>
<td>role</td>
<td>searchview</td>
</tr>
<tr>
<td>artifact_id</td>
<td>searchview-alone</td>
</tr>
<tr>
<td>maintainers</td>
<td>backend3</td>
</tr>
</tbody>
</table>

[Edit] [Refresh] What fields can I edit?
# searchview

**Apollofied aggregation search view service**

---

### INFO

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>owner</td>
<td>kiwi</td>
</tr>
<tr>
<td>system</td>
<td>search</td>
</tr>
<tr>
<td>visibility</td>
<td>private</td>
</tr>
<tr>
<td>service_discovery</td>
<td>searchview</td>
</tr>
<tr>
<td>slack_channel</td>
<td>kiwi</td>
</tr>
<tr>
<td>tags</td>
<td>kiwi</td>
</tr>
<tr>
<td></td>
<td>apollo</td>
</tr>
<tr>
<td></td>
<td>search</td>
</tr>
</tbody>
</table>

---

### DEPENDENCIES

<table>
<thead>
<tr>
<th>Dependency</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>charts-api-backend</td>
<td>declared + runtime</td>
</tr>
<tr>
<td>chartview</td>
<td>declared + runtime</td>
</tr>
<tr>
<td>collection (collection2)</td>
<td>declared + runtime</td>
</tr>
<tr>
<td>facebook</td>
<td>declared + runtime</td>
</tr>
<tr>
<td>identity (identity2)</td>
<td>declared + runtime</td>
</tr>
<tr>
<td>mergedprofile (socialgraph3)</td>
<td>declared + runtime</td>
</tr>
<tr>
<td>metadata (metadata-proxy)</td>
<td>declared + runtime</td>
</tr>
<tr>
<td>playlist -&gt; playlist-proxy</td>
<td>declared + runtime</td>
</tr>
<tr>
<td>playlist-mosaic</td>
<td>declared + runtime</td>
</tr>
<tr>
<td>popcount (popcount2)</td>
<td>declared + runtime</td>
</tr>
<tr>
<td>search</td>
<td>declared + runtime</td>
</tr>
</tbody>
</table>
Apollofied aggregation search view service

https://github.com/kiwi/searchview/blob/master/README.md

INFO

- **owner**: kiwi
- **system**: search
- **visibility**: private
- **service_discovery**: searchview
- **slack_channel**: kiwi
- **tags**: kiwi, apollo, search
- **paddle_key**: searchview
- **role**: searchview
- **artifact_id**: searchview-alone
- **maintainers**: backend3

DEPENDENCIES

- **charts-api-backend**: declared + runtime
- **chartview**: declared + runtime
- **collection (collection2)**: declared + runtime
- **facebook**: declared + runtime
- **identity (identity2)**: declared + runtime
- **mergedprofile (socialgraph3)**: declared + runtime
- **metadata (metadata-proxy)**: declared + runtime
- **playlist -> playlist-proxy playlist4**
- **playlist-mosaic**: declared + runtime
- **popcount (popcount2)**: declared + runtime
- **search**: declared + runtime
gaboeventservice

OVERVIEW  ACTIVITY  DEPLOYMENT  PIPELINE HISTORY  ROUTING  CONFIGURATION  API

Filters  Only show deployed versions  Only consider 10 last builds

MACHINES

Host names redacted

VERSION
0.1.32-SNAPSHOT-c5f6f70-20160112-0833
created 2015-01-12 09:33 by Jenkins

0.1.30-SNAPSHOT-872b690-20151221-1357
created 2015-12-21 13:57 by Jenkins

0.1.29-SNAPSHOT-0905391-20151216-1510
created 2015-12-16 15:10 by Jenkins

0.1.28-SNAPSHOT-76b8ee7-20151216-1410
created 2015-12-16 14:10 by Jenkins

0.1.27-SNAPSHOT-168c3e0-20151230-2055
created 2015-12-30 20:55 by Jenkins

0.1.26-SNAPSHOT-96ce070-20151224-1832
created 2015-12-24 18:32 by Jenkins

0.1.25-SNAPSHOT-8be5ae3-20151224-1715
created 2015-12-24 17:15 by Jenkins

http://github.com/spotify/helios
searchview

Manage routing between pods in our infrastructure. If you need any support with this please talk to tools@ or #tools.

This view is powered by podlinks

Configuration for service discovery name searchview

<table>
<thead>
<tr>
<th>From</th>
<th>Link</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>→</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

18 running instances

Change history

<table>
<thead>
<tr>
<th>Time</th>
<th>From</th>
<th>Update</th>
<th>Protocol</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 months ago</td>
<td>D</td>
<td>E → A</td>
<td>hm</td>
<td>alexey</td>
</tr>
<tr>
<td>2 months ago</td>
<td>D</td>
<td>F → E</td>
<td>hm</td>
<td>alexey</td>
</tr>
<tr>
<td>2 months ago</td>
<td>D</td>
<td></td>
<td>hm</td>
<td>alexey</td>
</tr>
<tr>
<td>1 month ago</td>
<td>D</td>
<td></td>
<td>hm</td>
<td>alexey</td>
</tr>
</tbody>
</table>
Apollofied aggregation search view service

Owner: kiwi
System: search
Visibility: private
Service_discovery: searchview
Slack_channel: kiwi

Tags: kiwi, apollo, search
Pagerdutykey: searchview
Role: searchview
Artifact_id: searchview-alone
Maintainers: backend3

What fields can I edit?
Apollofied aggregation search view service

**INFO**

<table>
<thead>
<tr>
<th>owner</th>
<th>kiwi</th>
</tr>
</thead>
<tbody>
<tr>
<td>system</td>
<td>search</td>
</tr>
<tr>
<td>visibility</td>
<td>private</td>
</tr>
<tr>
<td>service_discovery</td>
<td>searchview</td>
</tr>
<tr>
<td>slack_channel</td>
<td>kiwi</td>
</tr>
<tr>
<td>tags</td>
<td>kiwi, apollo, search</td>
</tr>
<tr>
<td>pagerdutykey</td>
<td>redacted</td>
</tr>
<tr>
<td>role</td>
<td>searchview</td>
</tr>
<tr>
<td>artifact_id</td>
<td>searchview-alone</td>
</tr>
<tr>
<td>maintainers</td>
<td>backend3</td>
</tr>
</tbody>
</table>

**DEPLOYMENT [HERMES]**

searchview 0.2.74

76 instances

**DEPENDENCIES**

- charts-api-backend (declared + runtime)
- chartview (declared + runtime)
- collection (collection2) (declared + runtime)
- facebook (declared + runtime)
- identity (identity2) (declared + runtime)
- mergedprofile (mergedprofile3) (declared + runtime)
- metadata (metadata-proxy) (declared + runtime)
- playlist-direct (playlist-proxy) (declared + runtime)
- playlist-mosaic (playlist-mosaic2) (declared + runtime)
- popcount (popcount2) (declared + runtime)
- search (declared + runtime)
And...

- Create new components
- Compare live configs
- Check activity (logins, system updates, etc)
- Check Global stats (Apollo versions, etc)
- API documentation
- Provision hardware
- Set up service monitoring dashboards
- Configure data processing pipeline monitoring
- ...
Design
Core data model

Many many-to-many relations
Features add specific data
Discovery names as indirection
YAML files

Loose/multiple schemas

Files live with code
Dirty Data

Organisational change => ownership confusion
Infrastructure evolution => runtime confusion
Owners don’t benefit from metadata quality
Ten Things You Didn't Know About **System-Z** That Will Amaze You!

1. Migrating your service from the old "servicedb" takes less than two minutes and unlocks many of System-Z’s useful features!

2. **System-Z’s New Service** feature helps you create a skeleton Apollo backend service in seconds! You can get your build pipeline going on any existing Jenkins instance.

3. You can see an Apollo-Hermes service's dependencies (both declared and detected in runtime) and drill down to explore service relationships.

4. You can quickly see exactly where any service using Hermes is currently deployed!

5. For any service that has registered its PagerDuty key, you can see all current incidents and escalation policies!

6. System-Z’s homepage shows you all the system components owned by you and your teams.

7. You can see and manage routing between pods in our infrastructure!

8. You can see the published API for a service and all the endpoints on which it will respond. All services are clearly marked "public" or "private" to indicate if you can use them.

9. You can compare current configurations of running instances of Hermes-based services, making it easy to spot production anomalies!

10. You can give the Tools team feedback and suggest new features that will help you in your work!
Impact

About 200 WAU/400 MAU of ~800 in TPD

Teams integrating features, making them easier to find

Teams talking about features => more consistent

System-Z mentioned as ‘great’ in 2016 ‘What sucks’

Swiss Army Knife (good?)
Conclusions

Microservices => many small things, big picture is hard
Metadata about microservices helps understand the system
Our metadata is dirty; this is probably unavoidable
Combining many tools => better collaboration and consistency