Microservices

Riad Gozim

2018 OME Annual Users Meeting
OMERO - a recap

- OMERO.figure
- OMERO.gallery
- OMERO.iviewer
- ...and more

webclient  webgateway  api
Main web client  render images  JSON api

myapp
Add your own apps

BlitzGateway
Wrapped Objects
Blitz Connection

Python API
omero.model
services

OMERO.server
A quick refresh (IDR)

Publically available

- Runs on OMERO
- Analysis, sharing and reuse of scientific image data

Highly flexible

- Suitable for many projects
- Human cells, images from the Tara Oceans, fungi

Big data

- 46TB of data
- 50M images
- 160GB database
OMERO.mapr

What is Mapr?

- Plugin to OMERO.web
- Built in python, pip installable

What does it do?

- Search and browse attributes linked to images in the form of annotations (key-value)
OMERO-mapr
OMERO.mapr querying “map-annotations”

```sql
select mv.value as value,
    count(distinct i.id) as imgCount,
    count(distinct sl.parent.id) as childCount1,
    count(distinct pdl.parent.id) as childCount2
from ImageAnnotationLink ial
    join ial.child a join a.mapValue mv
    join ial.parent i left outer
    join i.wellSamples ws left outer
    join ws.well w left outer
    join w.plate pl left outer
    join pl.screenLinks sl left outer
    join sl.datasetLinks dil left outer
    join dil.parent ds left outer
    join ds.projectLinks pdl
where mv.name in (:filter) and a.ns in (:ns) and lower(mv.value) like :query and

    
    (dil is null
        and ds is null and pdl is null
        and ws is not null
            and w is not null and pl is not null
                and sl is not null)
    OR
    (ws is null
        and w is null and pl is null and sl is null
        and dil is not null
            and ds is not null and pdl is not null)

    group by mv.value
order by count(distinct i.id) DESC
```
“map-annotations” query flow

- IDR database - 160GB
- Query pulls together properties from numerous tables
- Perceivably slow for deployments outside of the IDR
How do we keep OMERO.mapr fast

- Nginx - load balancer
- Nginx - cache
Cache can only go so far

Cache needs to be manually populated when data is added to IDR

Cache has be deleted and reprimed - takes time

Not a good solution for our users who can’t keep up with updates
The challenge

How do we optimise queries in OMERO.mapr?

How do we release any changes without rewriting lots of code?

How do we deploy those updates without requiring a big update?
Microservices

Monolithic Architecture

- User Interface
- Business Logic
- Data Access Layer
- DB

Microservice Architecture

- User Interface
- Microservice
- Microservice
- Microservice
- Microservice
- DB
- DB
- DB
- DB
Microservices

- New features can be added with well-defined boundaries
- Allows developers to work separately on independent parts
- Microservices can be deployed, maintained, updated, and scaled independently of each other in a continuous fashion
Microservices

- New features can be added with well-defined boundaries for each piece of functionality
- Allows developers to work separately on independent parts of OMERO
- Microservices can be deployed, maintained, updated, and scaled independently of each other in a continuous fashion
- Compute can be spread across more hardware
Granularity

Looser Coupling, More Flexible/Portable, More Complex Outer Architecture

MONOLITHIC APP

WAR / EAR / App

App Server

Data Store

COARSE-GRAINED SERVICES

Service Domain

Service Domain

Service Domain

App Server

Runtime

Data Store

Data Store

MICROSERVICES

Service A

Service B

Service C

Service D

Runtime

Runtime

Runtime

Runtime

Data Store

Data Store

Data Store

Data Store

Tighter Coupling, Less Flexible/Portable, Less Complex Outer Architecture
Our types of microservice

1. Microservice as a **client**

2. Microservice as a **server**
Microservices as a **client**

- **webclient**: Main web client
- **webgateway**: Render images
- **api**: JSON api

- **microservice**
  - Blitz Connection

- **BlitzGateway**
  - Wrapped Objects
  - Blitz Connection

- **OMERO.server**
Microservices as a server

BlitzGateway
Wrapped Objects

microservice

microservice

microservice

OMERO.server
Microservices as server, for OMERO.mapr

Change the way OMERO.mapr queries the database

Add optimisations at the database layer

Make changes *silently*

Does not use ICE
Architecture overview with microservice
Mapr microservice “omero-ms-mapr”

Split up monolithic code base

Added OMERO.server optimisation at the database layer

PSQL Materialised Views

Microservice built with Java and Vert.x
A bit about Vert.x

- Polyglot fundamentals
- Async IO HTTP implementation
- Single and multithreaded worker “verticles”
- EventBus
- Well suited to microservices
- Easy to build with
As a user of IDR, what changes will I see?

Nothing - Just better performance when you click to view something that hasn’t been cached
Nginx configuration

```nginx
server {

   ...# Redirects OMERO.mapr URL's, depending on what's configured as the active API # in Nginx

location /ms/v0/ {
   proxy_pass http://idr1-slot2.openmicroscopy.org:51000/ms/v0/;
   proxy_redirect http://idr1-slot2.openmicroscopy.org:51000/ms/v0/ $scheme://$host/ms/v0/;
   proxy_set_header SCRIPT_NAME /ms/v0;
   proxy_set_header Host $host;
}

location /ms/v1/mapr/api/ {
   proxy_pass http://idr1-slot2.openmicroscopy.org:8080/v1/mapr/api/;
   proxy_redirect http://idr1-slot2.openmicroscopy.org:8080/v1/ $scheme://$host/ms/v1/;
   proxy_set_header SCRIPT_NAME /ms/v1;
   proxy_set_header Host $host;
}
}```
To sum up

- Give us the ability to address performance bottlenecks
- Allow us to do this with minimal disruption
- Make it easier to develop more features in future
- Allow us to deploy to **more** hardware
Additional

- gitlab - build OMERO.server:
  - https://gitlab.com/openmicroscopy/incubator/omero-dsl
  - https://gitlab.com/openmicroscopy/incubator/omero-all

- gitlab - mapr
  - https://gitlab.com/openmicroscopy/incubator/omero-ms-map

- Glencoe microservices:
  - https://github.com/glencoesoftware/omero-ms-core
  - https://github.com/glencoesoftware/omero-ms-thumbnail
  - https://github.com/glencoesoftware/omero-ms-image-region
  - https://github.com/glencoesoftware/omero-ms-pixel-buffer
Deployment options
Future directions
Some results of speed
Lessons

1. Introduction to gitlab/incubator OME\textsc{RO}.server
2. Opening omero-all with an IDE
3. A bit about gradle
4. Building omero-all with gradle or IDE
5. Publishing to maven local
6. Creating an application to use omero-all