

Microservices

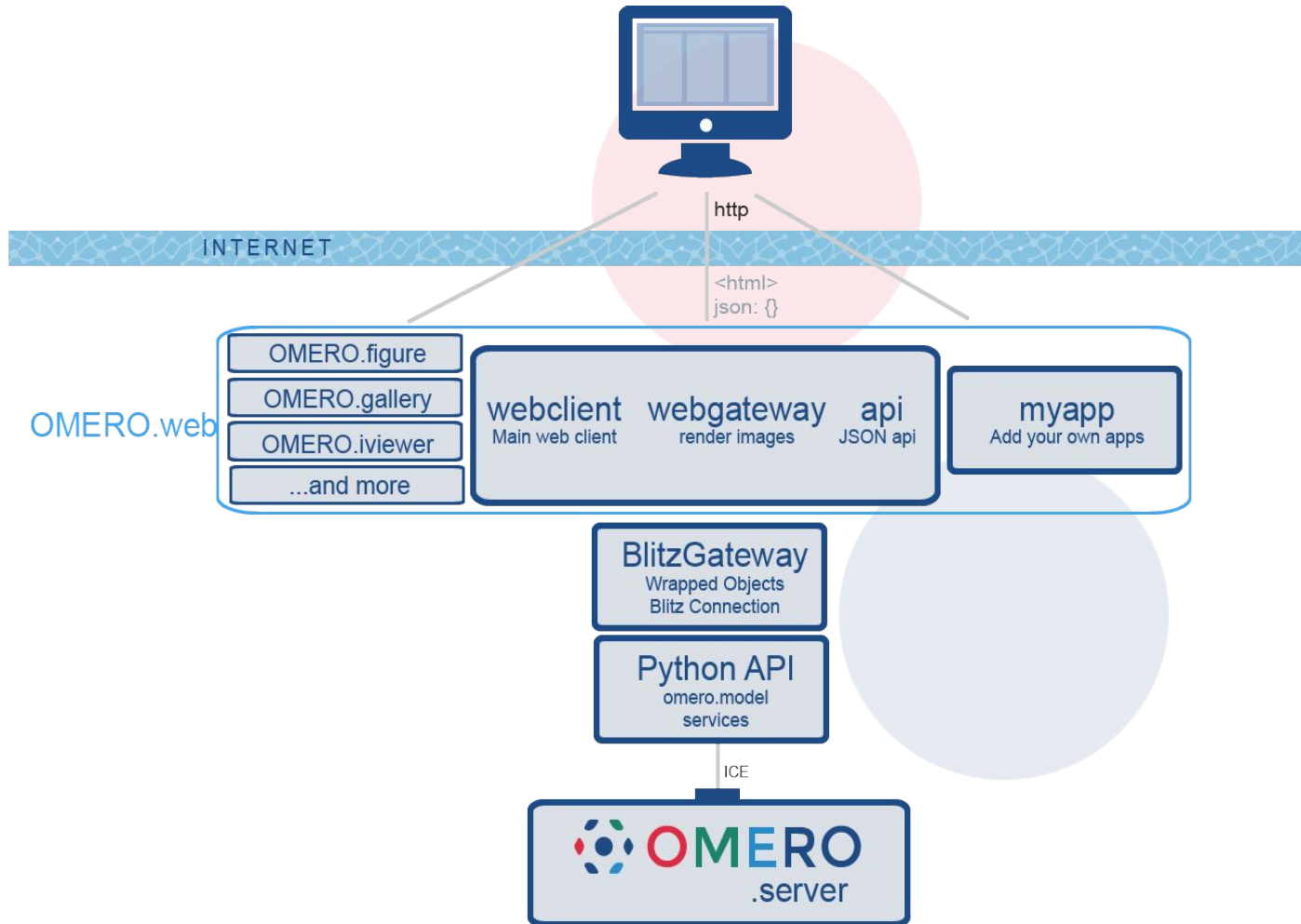
Riad Gozim

2018 OME Annual Users Meeting



Open Microscopy Environment
Centre for Gene Regulation & Expression
School of Life Sciences, University of Dundee
Dundee, Scotland, UK

OMERO - a recap



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

(<https://github.com/ome/omero-mapr#installing-from-pypi>)

What does it do?

- Search and browse attributes linked to images in the form of annotations (key-value)

OMERO-mapr

The screenshot displays the OMERO web interface. At the top, a navigation bar includes tabs for Studies, Genes, Phenotypes, Cell Lines, siRNAs, Antibodies, Compounds, and Organisms. The 'Genes' tab is highlighted with a red rectangle. Below this, a search bar labeled 'Type Gene Symbol...' is shown with a red arrow pointing to it. The left sidebar shows a tree view of data, with 'CDC1 (24) 4' selected. The main area displays three thumbnail images of cells. On the right, the 'General' tab of the image details panel is active, showing information for 'DTT p5 [Well 219, Field 1 (Spot 655)]'. A red arrow points to the 'Gene' section, which lists 'Gene Identifier' as YDR182W and 'Gene Symbol' as CDC1.

Public User

Search:

Public

Type Gene Symbol...

Match case

Add filter

Gene 1

- CDC1 (24) 4
 - idr0003-breker-plasticity/screenA (15)
 - DTT p5 3
 - ...p5 [Well 219, Field 1 (Spot 655)]
 - ...p5 [Well 219, Field 2 (Spot 656)]
 - ...p5 [Well 219, Field 3 (Spot 657)]
 - H2O2 p5 3
 - SD1 p5 3
 - SD2 p5 3
 - starvation p5 3
 - ...-ledesmafernandez-dad4/screenB (5) 3
 - Plate1-TS-Green-B 1
 - Plate2-TS-Green-A 2
 - Plate3-TS-Green-A 2
 - ...-ledesmafernandez-dad4/screenD (3) 1
 - ...-ledesmafernandez-dad4/screenE (1) 1

DTT p5 [Well 219, Field 1 (Spot 655)]

Image ID: 126100
Owner: Public data

Show all

Image Details

Import Date: 2015-09-22 18:24:14
Dimensions (XY): 672 x 512
Pixels Type: uint16
Pixels Size (XYZ) (μm): 0.22 x 0.22 x -
Z-sections/Timepoints: 1 x 1
Channels: GFP, Transmitted, Cherry
ROI Count: 0

Attributes 4

Gene

Added by: Public data

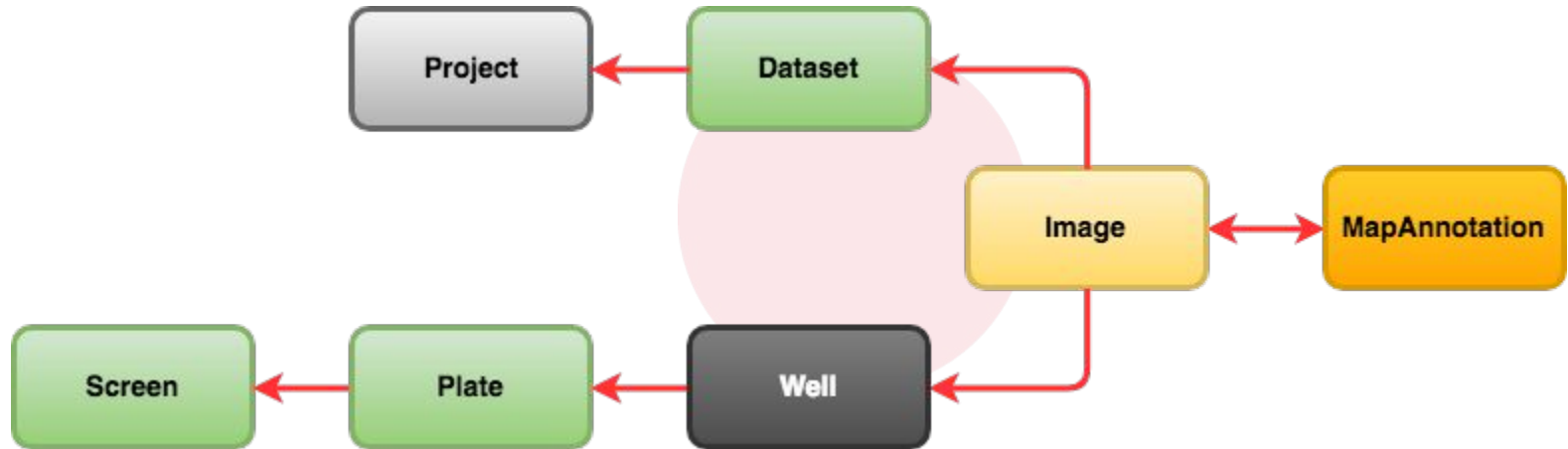
Gene Identifier YDR182W

Gene Symbol CDC1

OMERO.mapr querying “map-annotations”

```
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 i.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



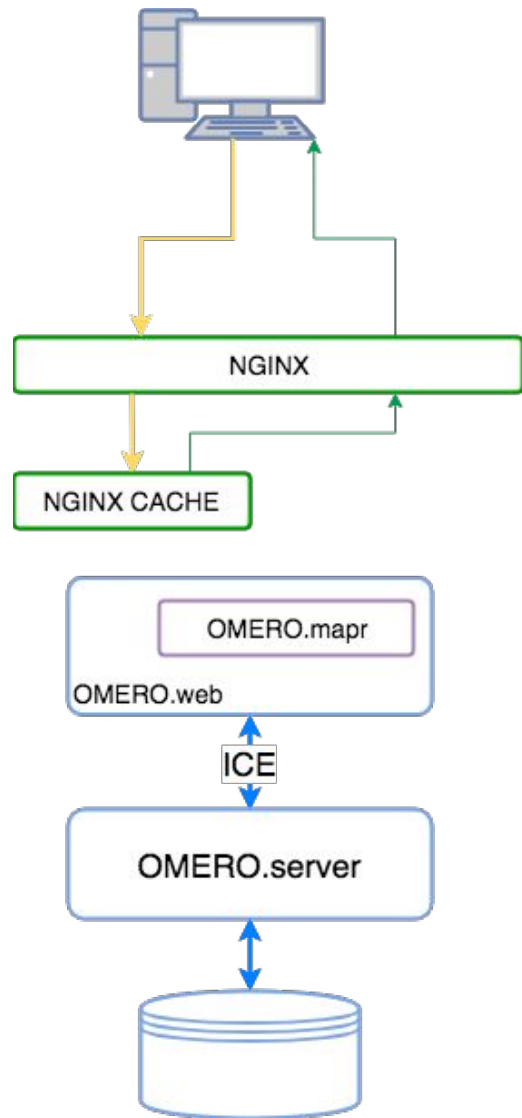
- IDR database - 160GB
- Query pulls together properties from numerous tables
- Perceivably slow for deployments outside of the IDR

How do we keep OMERO.mapr fast

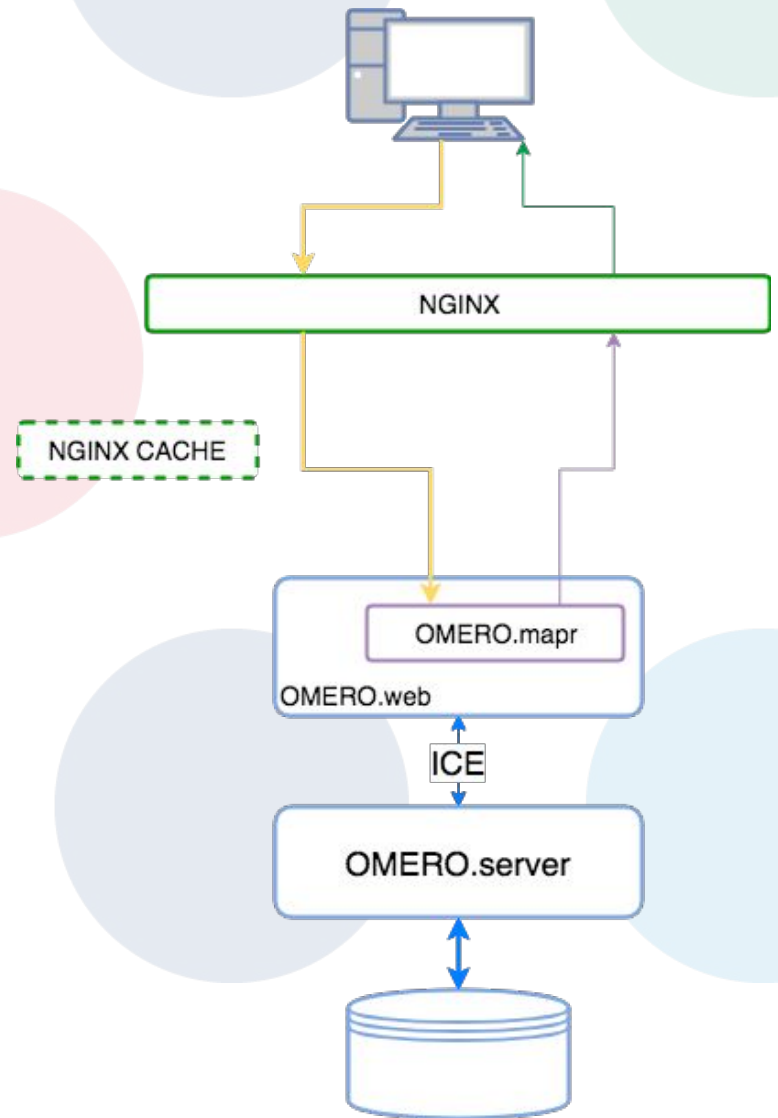
- Nginx - load balancer
- Nginx - cache



With cache hit



With cache miss



→ Data request

→ Data result

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

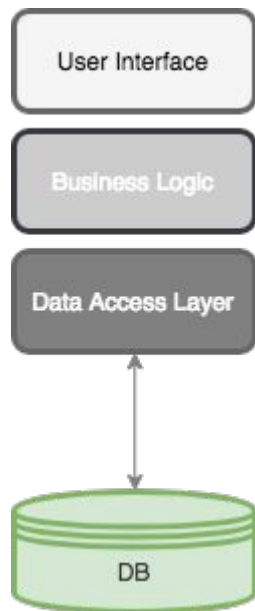
How do we optimise queries in OMERO.mapr?

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

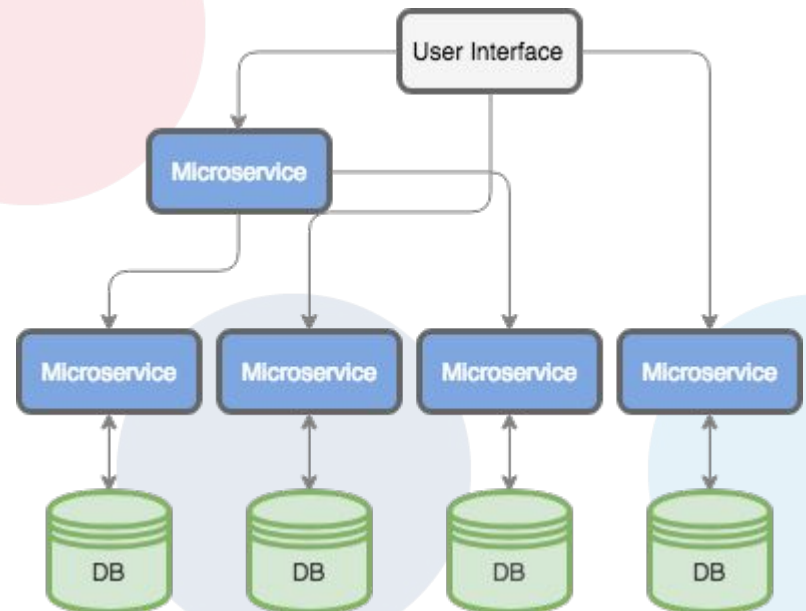
How do we deploy those updates without requiring a big update?

Microservices

Monolithic Architecture

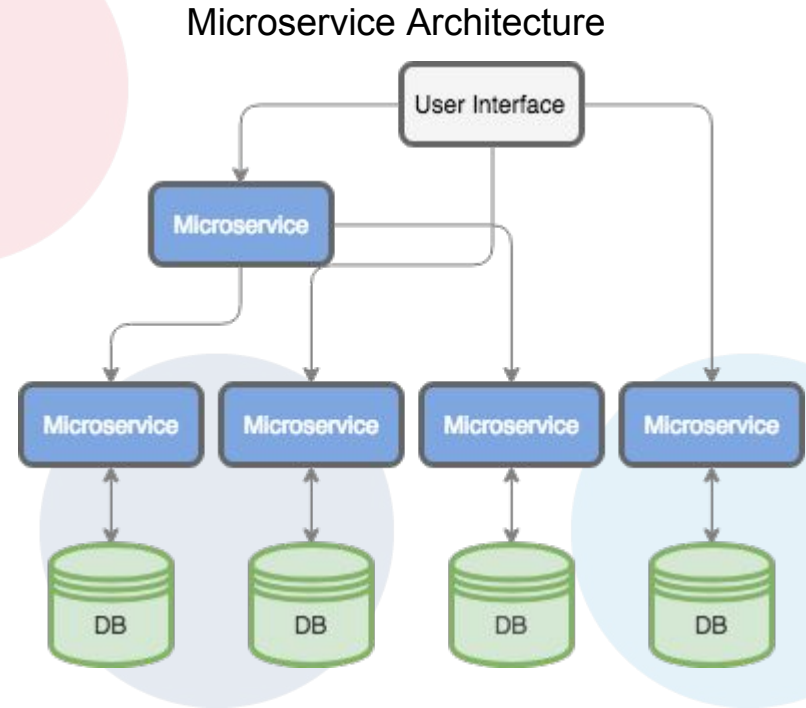


Microservice Architecture



Microservices

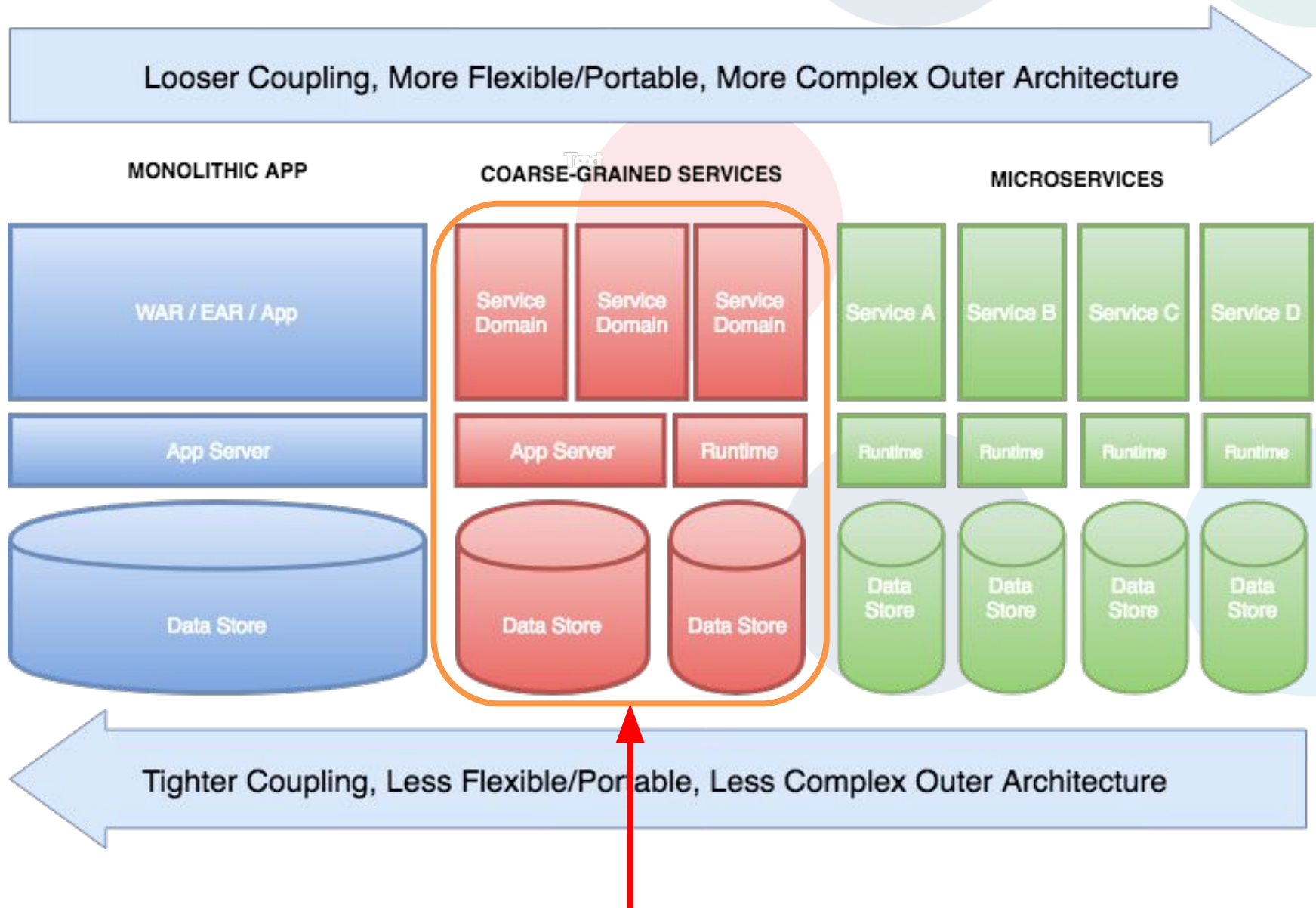
- New features can be added with well-defined boundaries
- Allows developers to work separately on independent parts
- Microservices can be deployed, maintained, updated, and scaled independently of each other in a continuous fashion



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

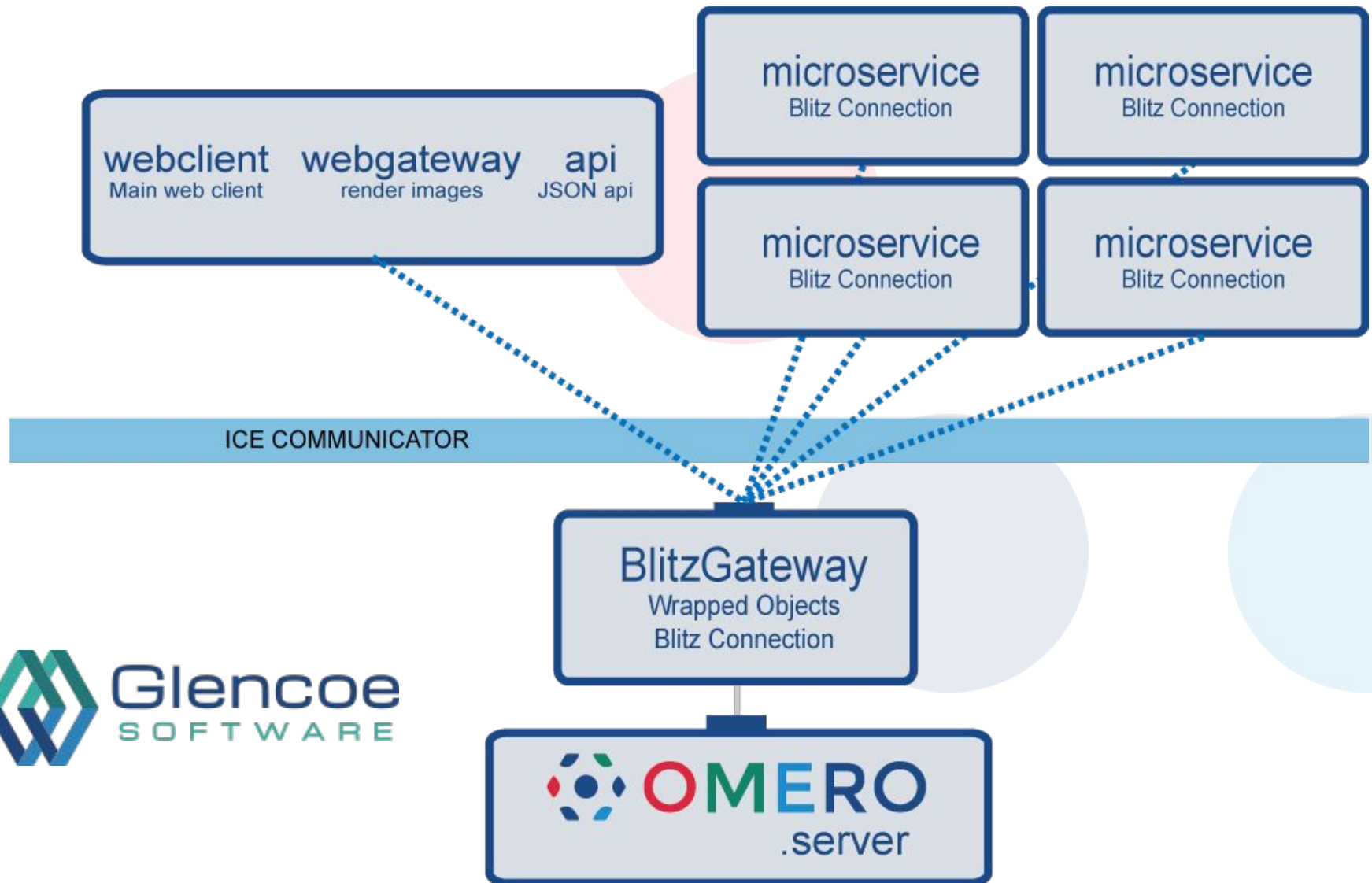
Granularity



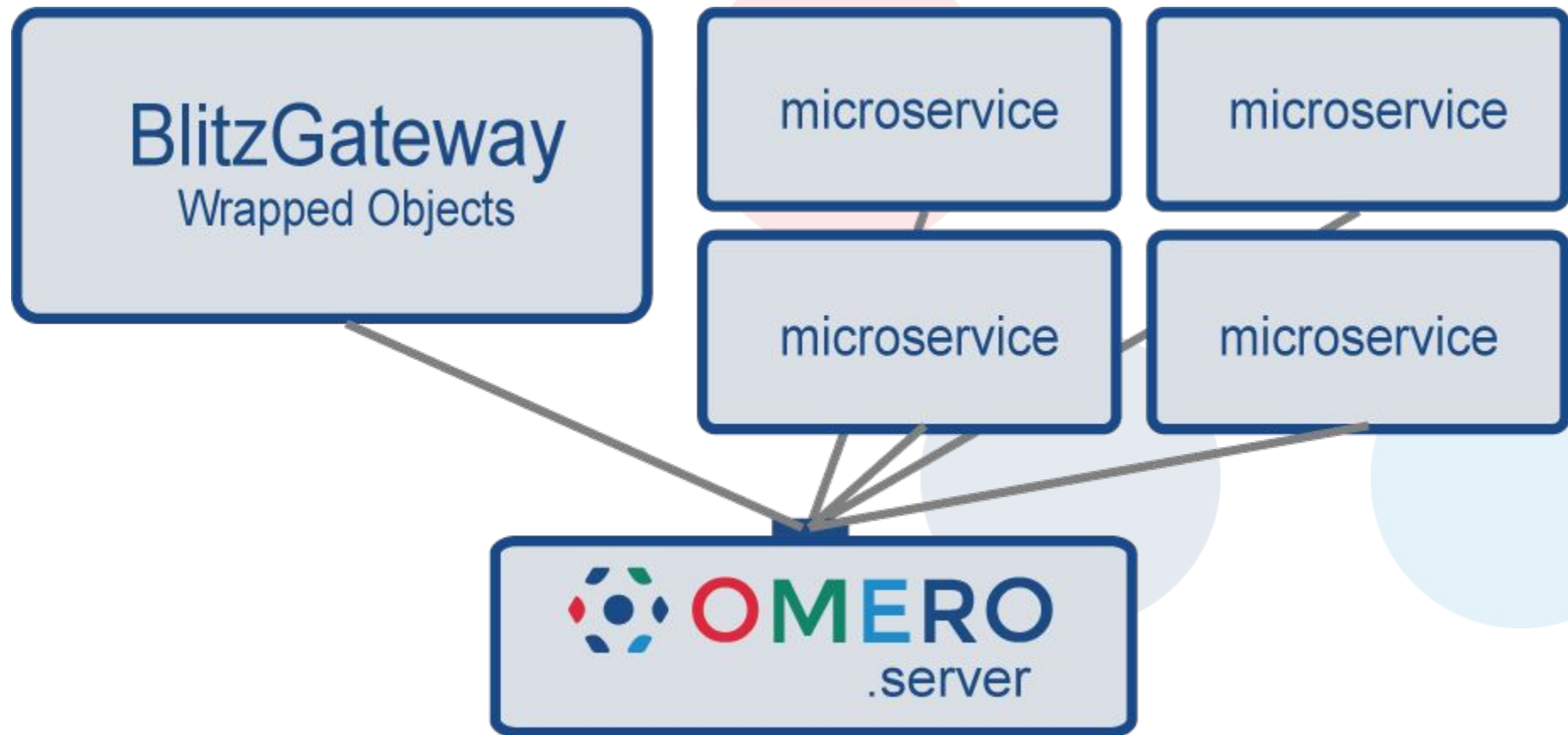
Our types of microservice

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

Microservices as a **client**



Microservices as a **server**



Microservices as server, for OMERO.mapr

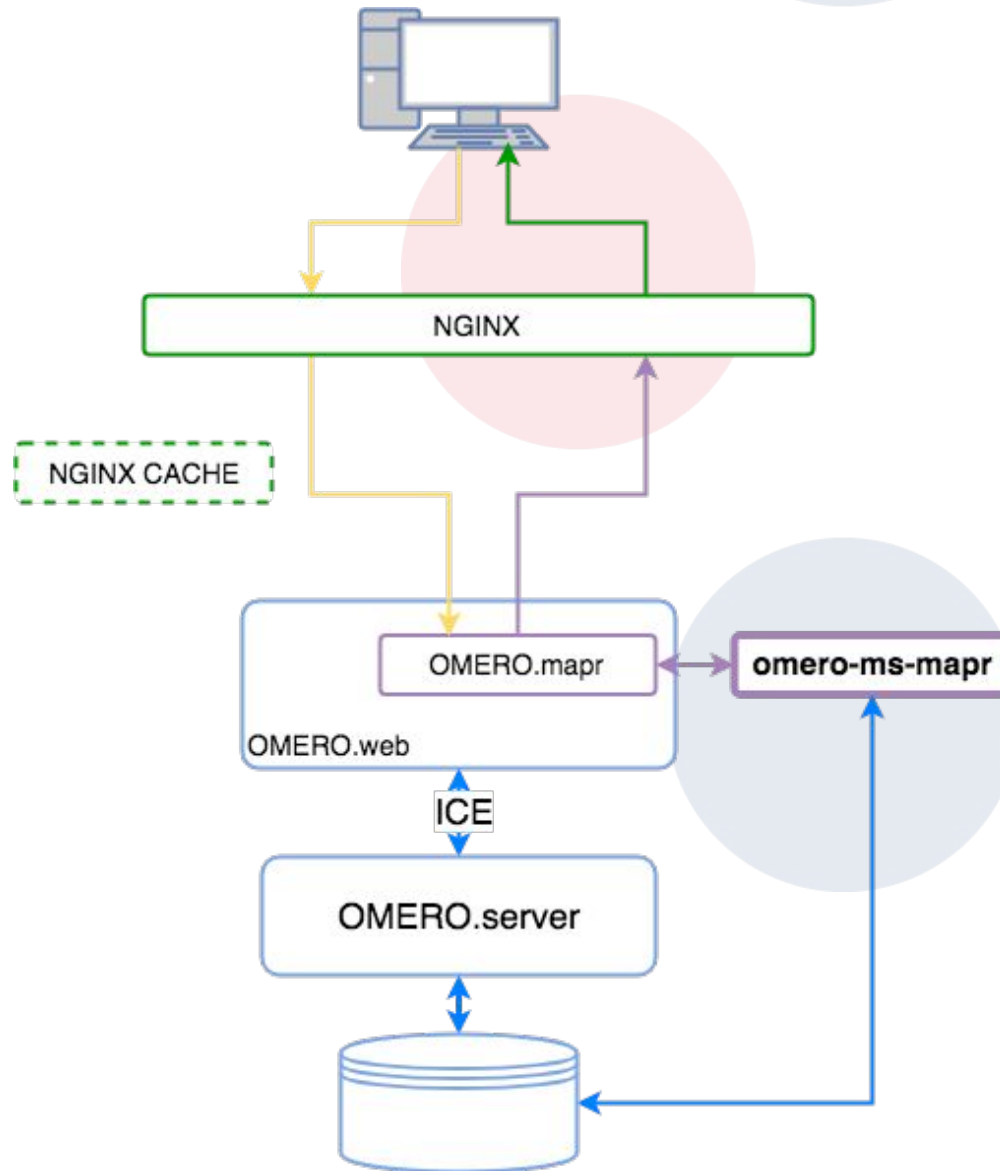
Change the way OMERO.mapr queries the database

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**

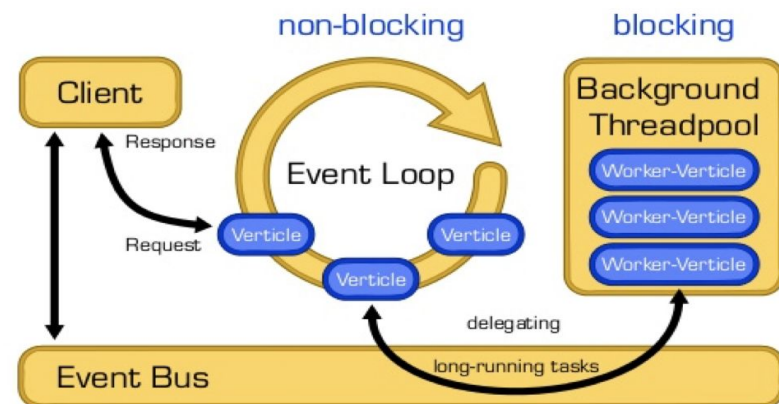
Microservice built with Java and Vert.x



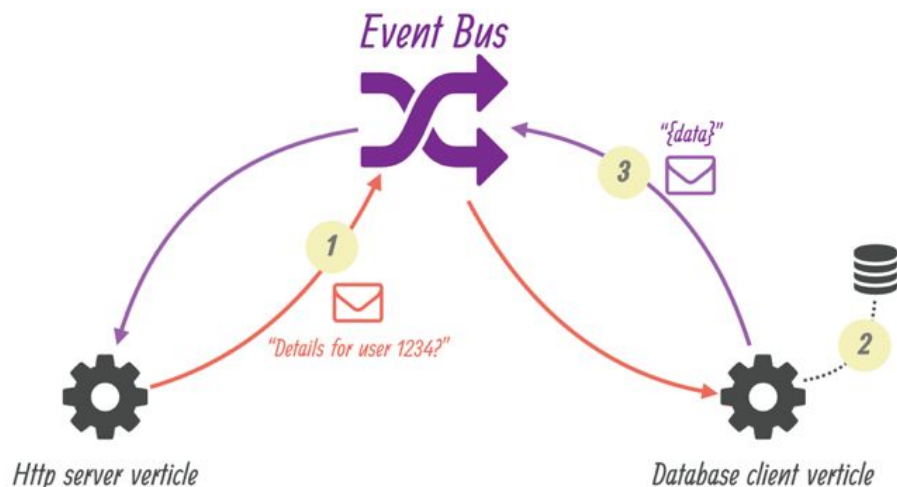
A bit about Vert.x

- Polyglot fundamentals
- Async IO HTTP implementation
- Single and multithreaded worker “verticles”
- EventBus
- Well suited to microservices
- Easy to build with

Architecture



codecentric AG





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

```
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 OMERO.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