Zocalo: a high-throughput data processing framework

NOBUGS 2022

Richard Gildea

diamond

Outline

- What is Zocalo?
- What can it do?
- Message brokers/RabbitMQ
- Zocalo components
- Where do things run?
- Monitoring

What is Zocalo?

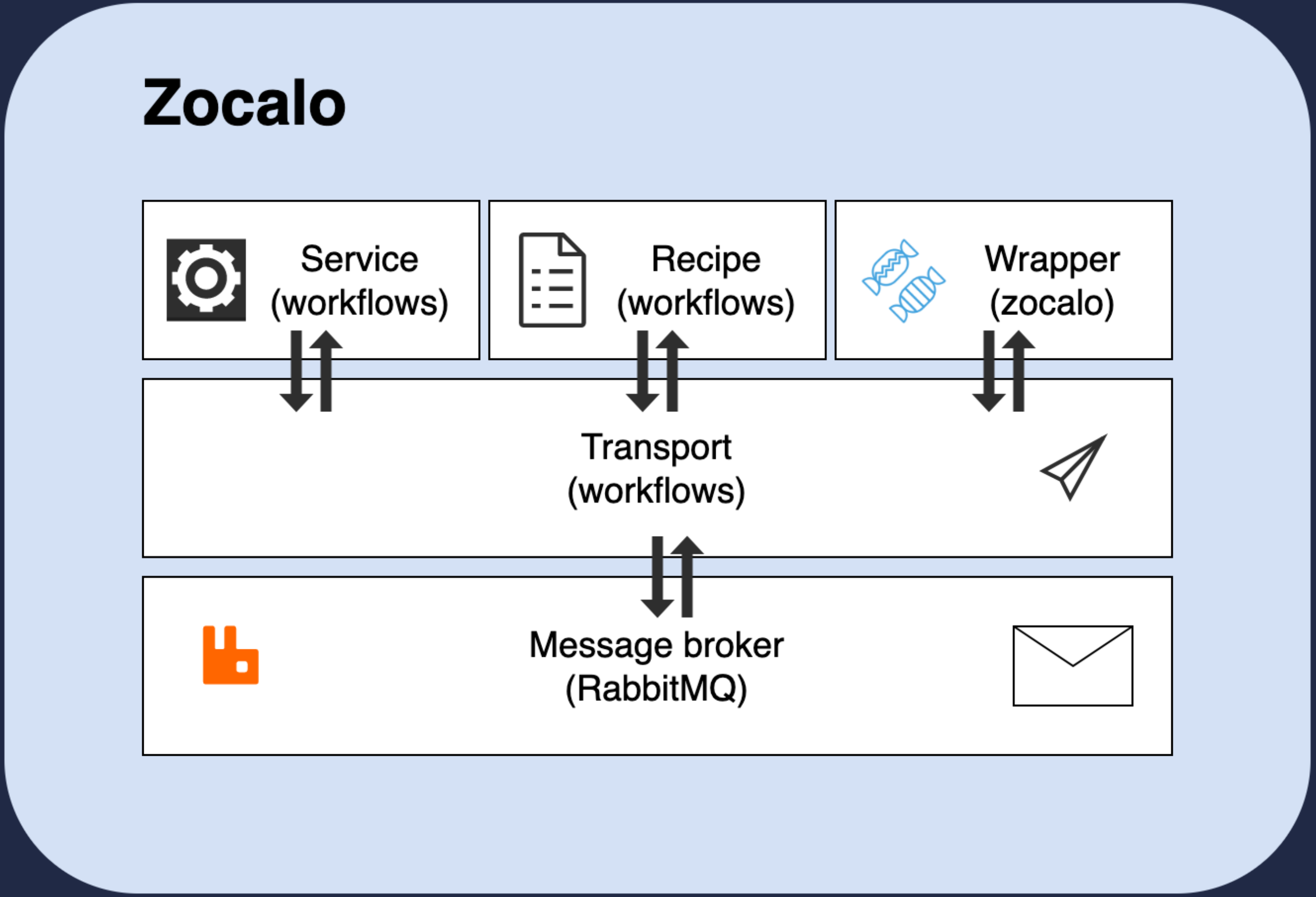
What is Zocalo?

Data analysis infrastructure

- Recipes
- Services
- Wrappers

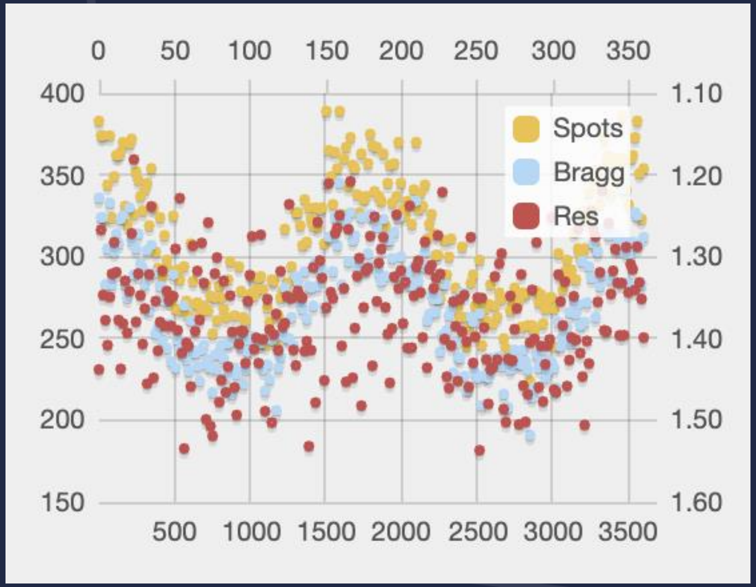
Communication via

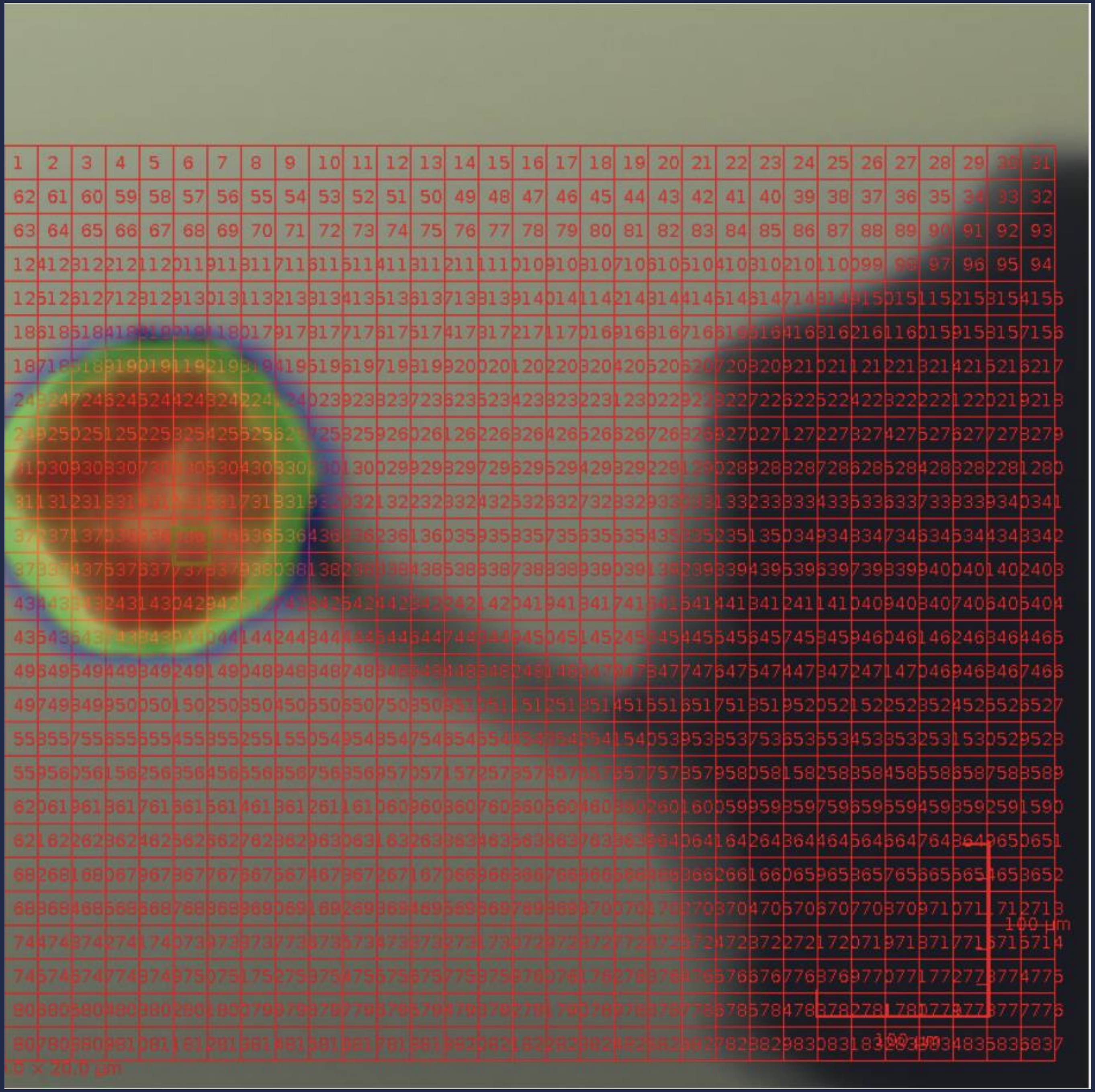
- Transport layer
- Message broker



What can it do?

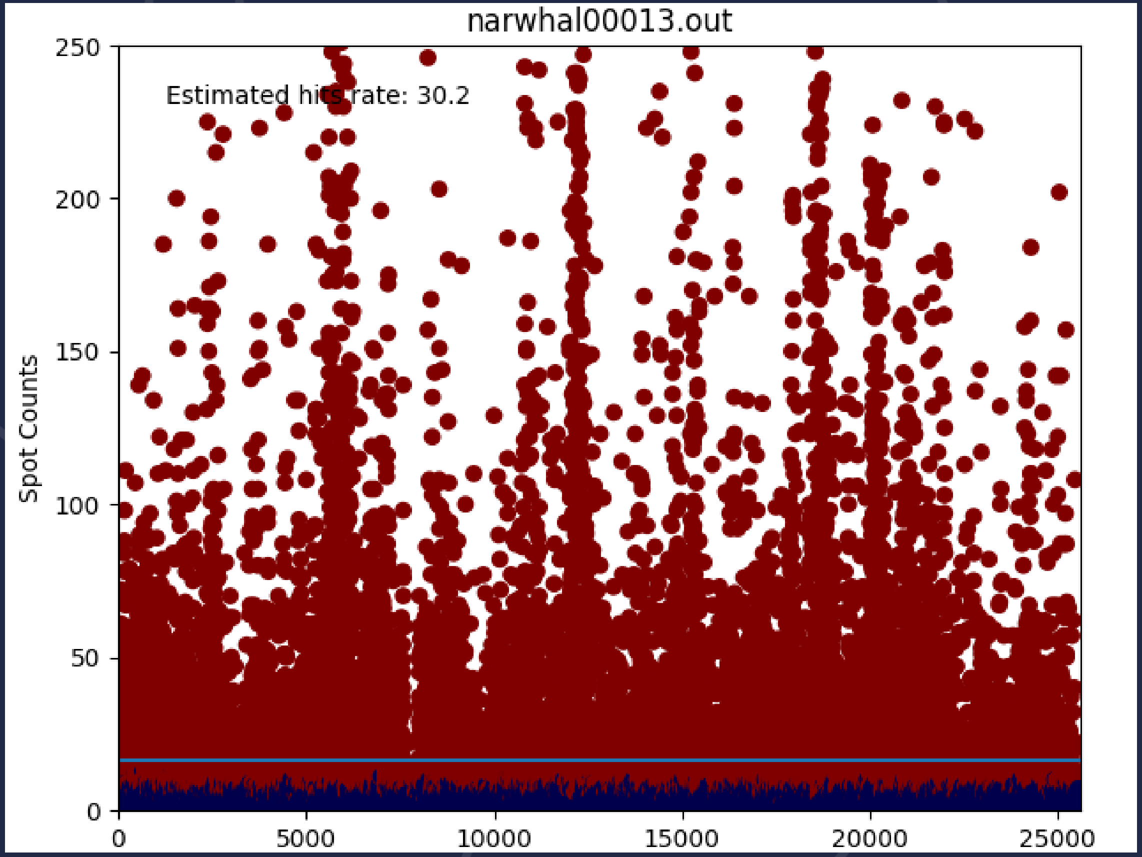
Online analysis

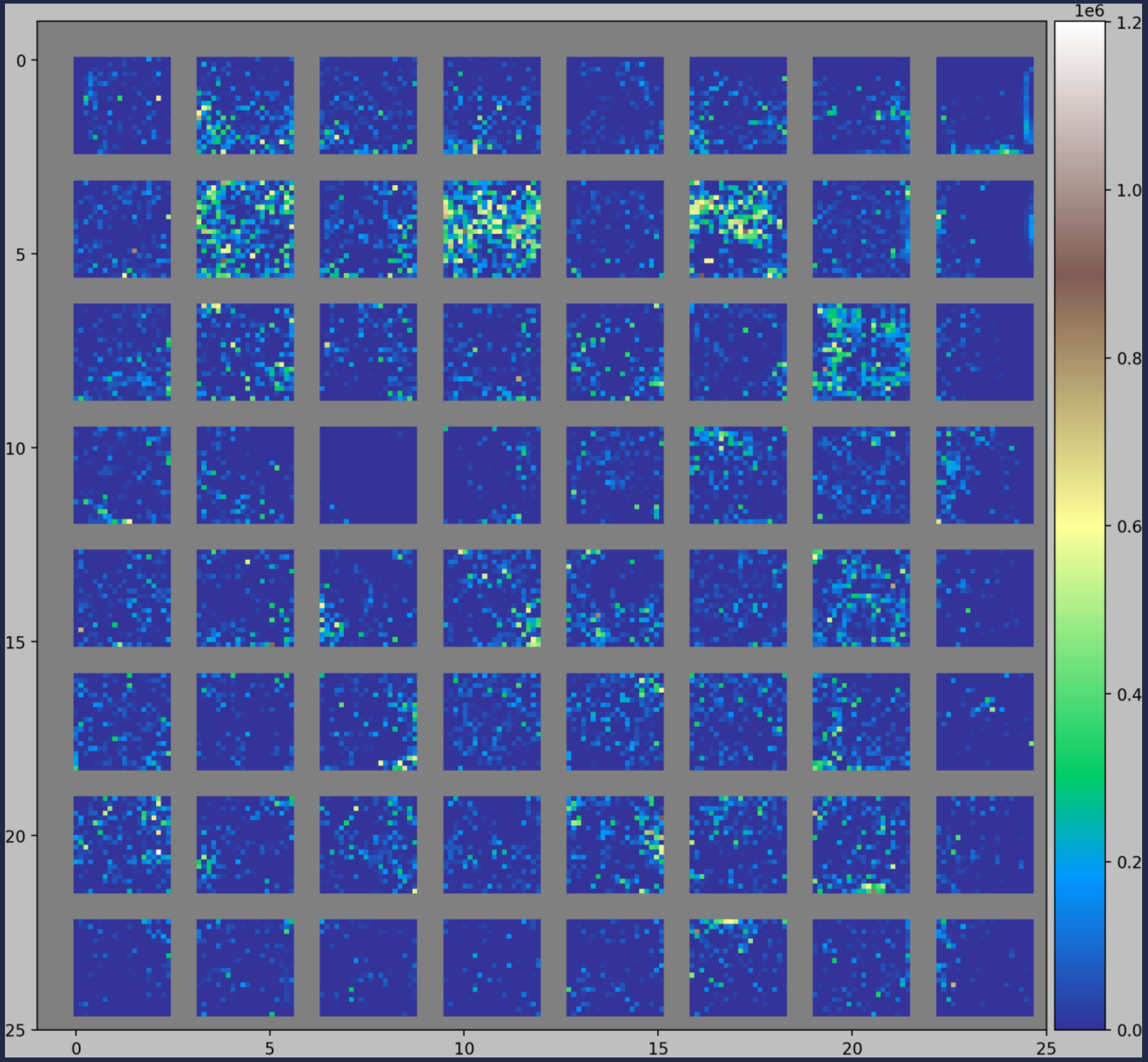




What can it do?

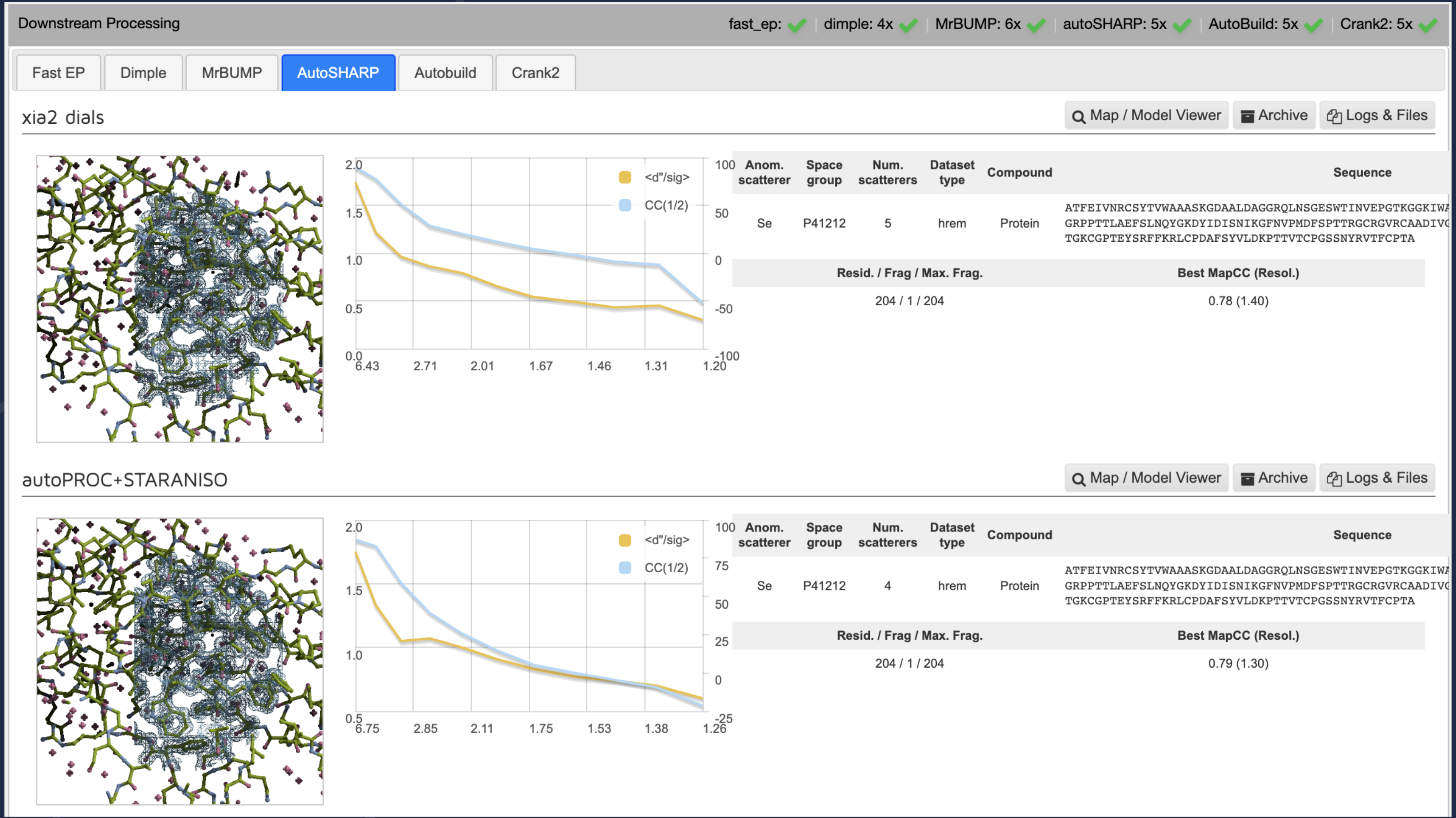
Online analysis





What can it do?

Offline analysis



Downstream Processing — fast_ep: ✓ | dimple: 4x ✓ | MrBUMP: 6x ✓ | autoSHARP: 5x ✓ | AutoBuild: 5x ✓ | Crank2: 5x ✓

xia2 dials

Anom. scatterer	Space group	Num. scatterers	Dataset type	Compound	Sequence
Se	P41212	5	hrem	Protein	ATFEIVNRCSYTVWAAASKGDAALDAGGRQLNSGESWTINVEPGTKGGKIWA GRPPTTLAEFSLNQYGKDYIDISNIKGFNVPMDFSPTTRGCRGVRCAADIVG TGKCGPTEYSRFFKRLCPDAFSYVLDKPTTVTCPGSSNYRVTFCPTA

Resid. / Frag / Max. Frag.	Best MapCC (Resol.)
204 / 1 / 204	0.78 (1.40)

autoPROC+STARANISO

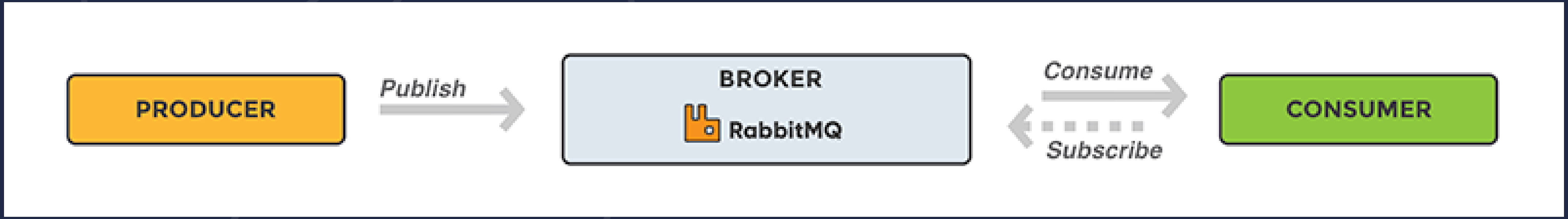
Anom. scatterer	Space group	Num. scatterers	Dataset type	Compound	Sequence
Se	P41212	4	hrem	Protein	ATFEIVNRCSYTVWAAASKGDAALDAGGRQLNSGESWTINVEPGTKGGKIWA GRPPTTLAEFSLNQYGKDYIDISNIKGFNVPMDFSPTTRGCRGVRCAADIVG TGKCGPTEYSRFFKRLCPDAFSYVLDKPTTVTCPGSSNYRVTFCPTA

Resid. / Frag / Max. Frag.	Best MapCC (Resol.)
204 / 1 / 204	0.79 (1.30)

Message brokers

Message broker

- RabbitMQ (or ActiveMQ)
- Manages a set of queues to which applications can connect in order to send or receive messages



Message broker – why?

- Asynchronous (delayed) message delivery
- Distribute a message to multiple consumers
- Balance loads between multiple worker processes
- Reduced coupling between senders and receivers
- Improved fault tolerance

RabbitMQ

- Wide user base
- Open development model
- Excellent resources for users and administrators
- Flexible messaging topology
- Redundant cluster of three RabbitMQ servers on real machines

Zocalo components

Workflows: transport layer

- Abstraction on top of message broker
- Implements `PikaTransport` (RabbitMQ/AMPQ) and `StompTransport` (ActiveMQ/STOMP)
- Services and wrappers send and receive messages via the `workflows` transport layer rather than interacting directly with the message broker

Workflows: services

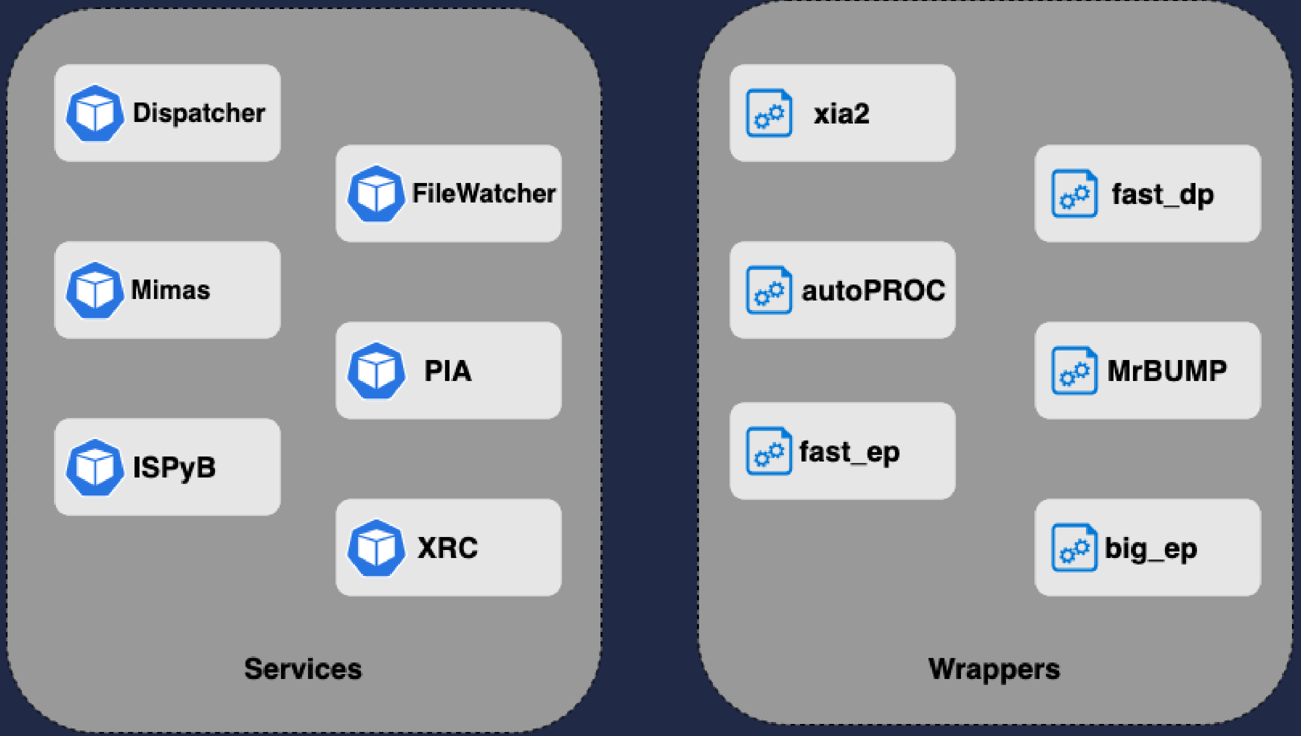
- A **service** consumes messages from a queue, performing some action based on the incoming message
- Optionally sends output to another queue
- Suitable for discrete short-lived tasks, e.g. spotfinding on an individual image or inserting results into a database
- Long-running background processes that wait for work
- Zocalo itself agnostic to where or how the services are run
- At DLS majority of services now running on Kubernetes

Zocalo: wrappers

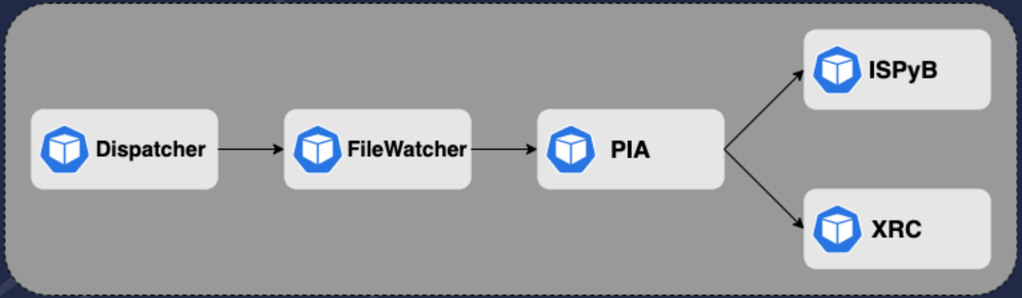
- Used for longer-running tasks, e.g. data processing programs like xia2 or fast_ep
- Only run when needed (typically on in-house cluster or STFC Cloud)
- Wrap something that isn't necessarily aware of zocalo
- A typical wrapper:
  - Takes an input message
  - Converts to suitable command line input
  - Runs the software
  - Interprets the results into an onward message to send back to Zocalo

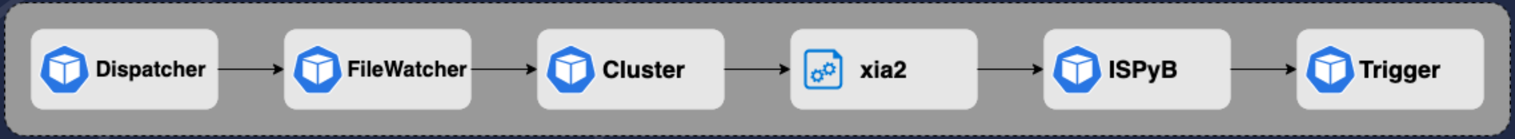
Services and wrappers - now what?

- How to link them together?
- Queue a given service consumes from is well-defined
- Where should a service send output to?



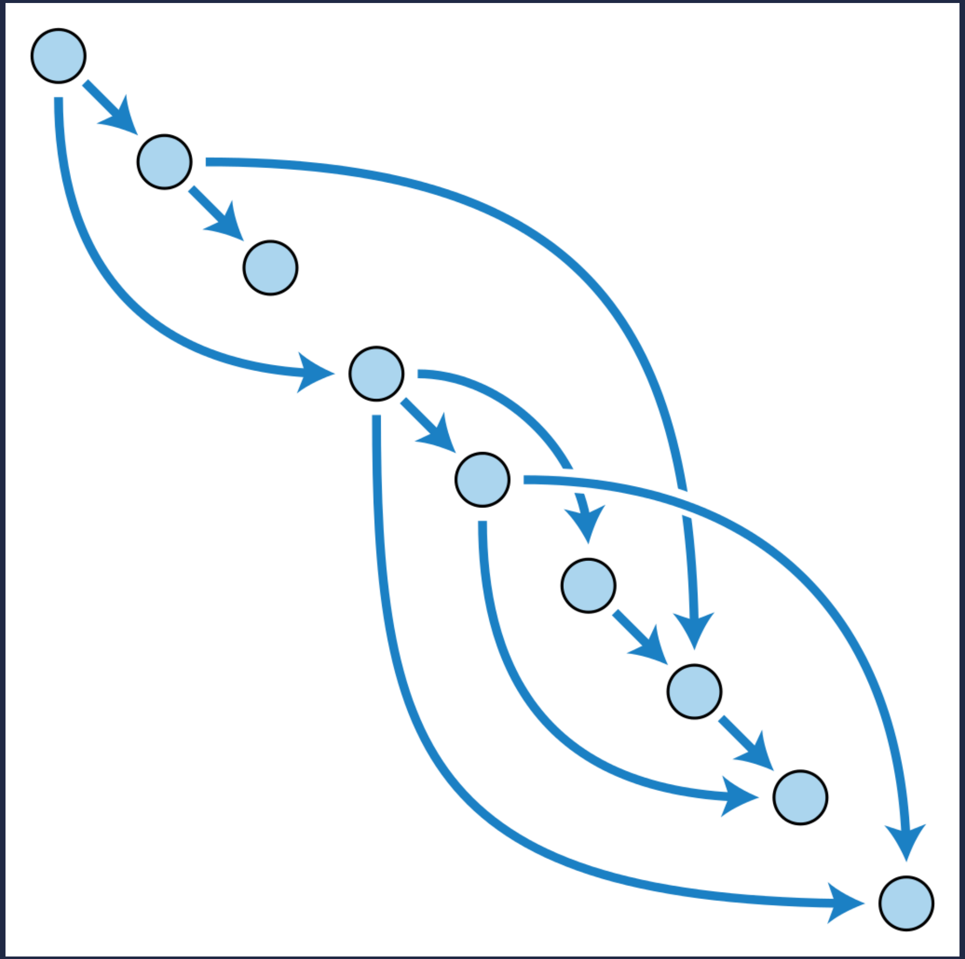
Services and wrappers - now what?





Workflows: recipes

- A recipe encodes the connections between services and wrappers
- Services are connected in a **directed acyclic graph**
- Nodes correspond to services
- Directed edges represent connections between services
- Nodes have one or more **input**
- Nodes can have zero, one or many **outputs**



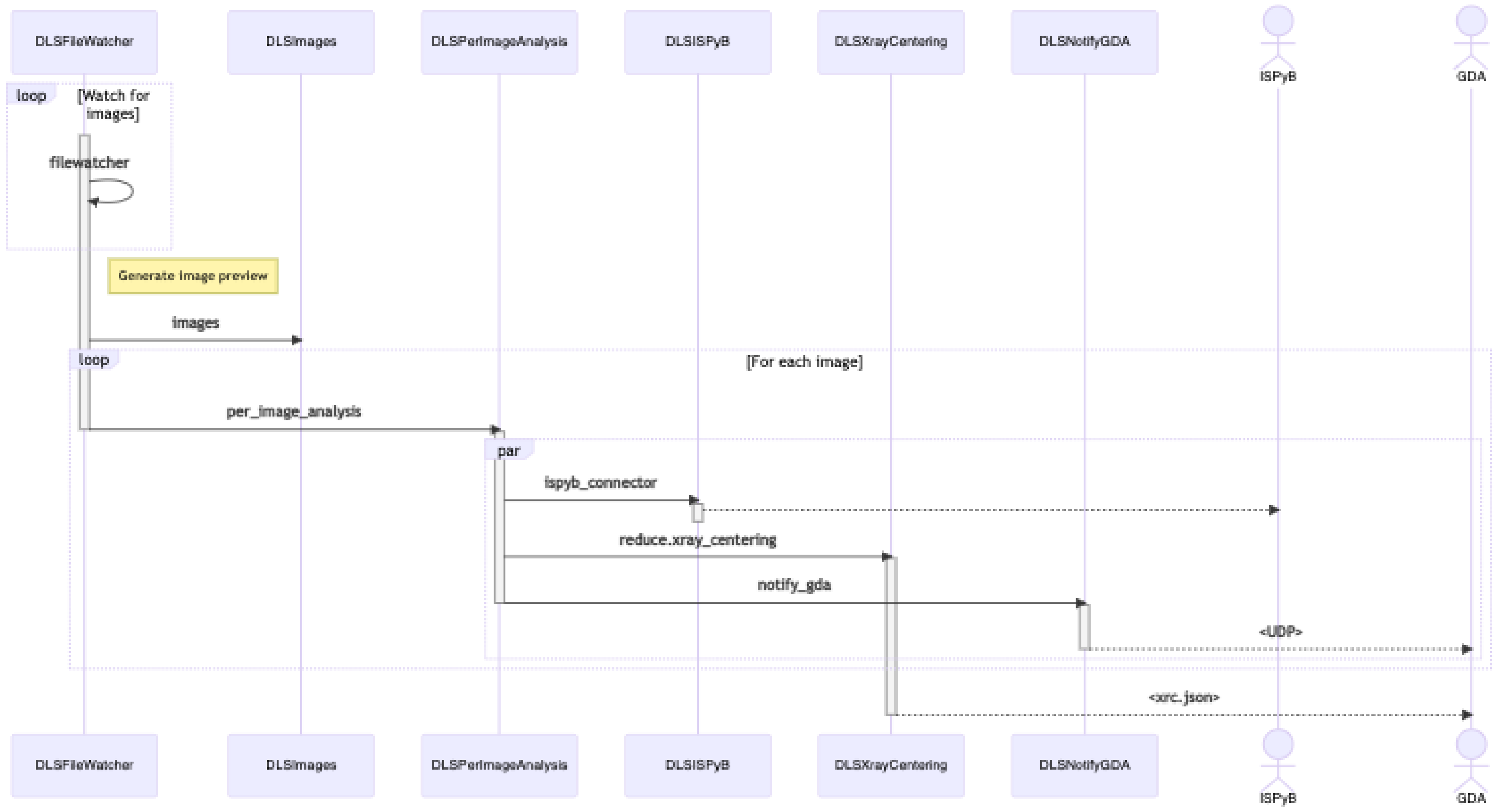
diamond

Workflows: recipes

A recipe can be represented as a Python dictionary:

```
{
  1: { (..), 'output': 2, (..) },
  2: { (..), 'output': 3, (..) },
  3: { (..) },
   (..)
  'start': [ (1, 'some data'), (2, { 'this can also be': 'a data structure' }) ]
}
```

X-ray centring

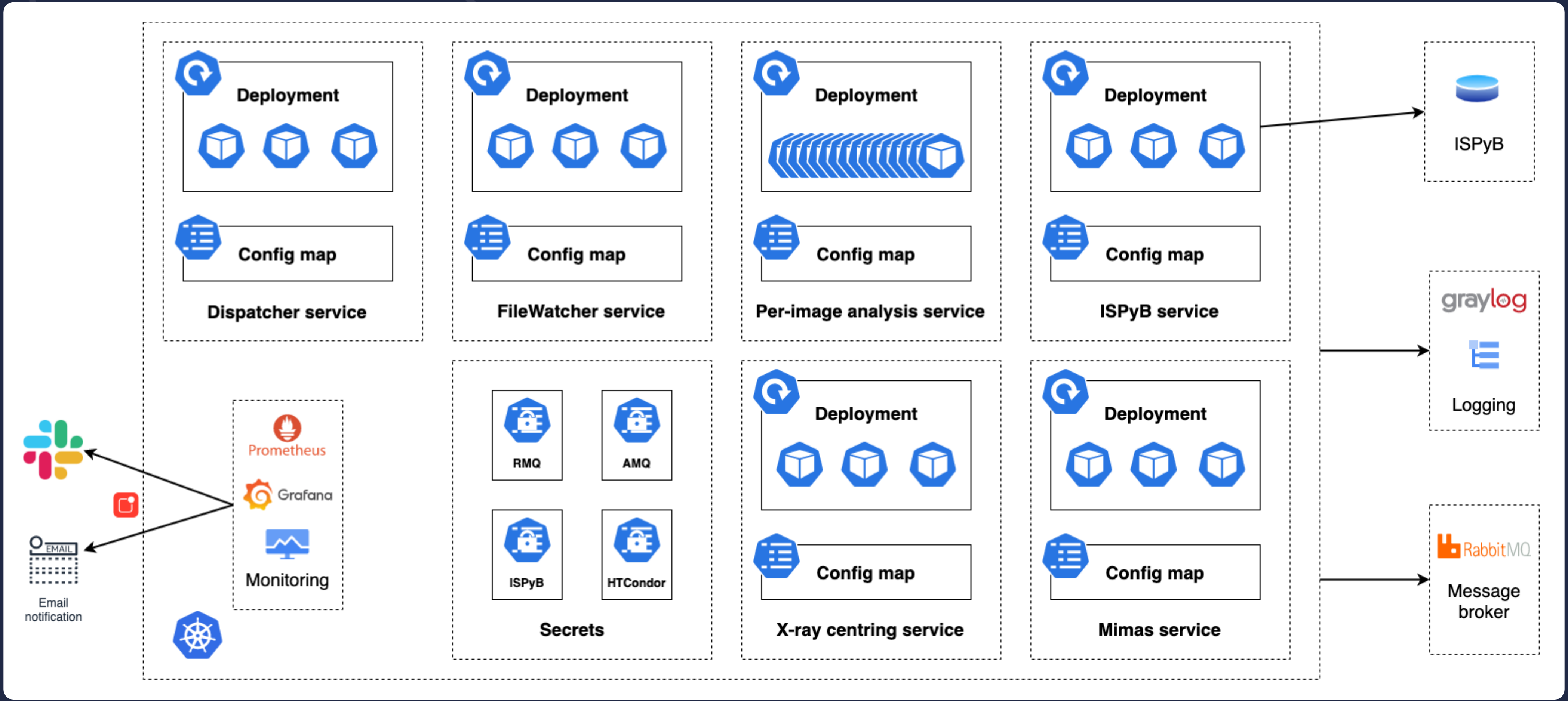

DLSFileWatcher
DLSImages
DLSPerImageAnalysis
DLSISPyB
DLSXrayCentering
DLSNotifyGDA
ISPyB
GDA
loop
[Watch for images]
filewatcher
Generate image preview
images
loop
[For each image]
per_image_analysis
par
ispyb_connector
reduce.xray_centering
notify_gda
<UDP>
<xrc.json>
DLSFileWatcher
DLSImages
DLSPerImageAnalysis
DLSISPyB
DLSXrayCentering
DLSNotifyGDA
ISPyB
GDA

Where do the services run?

Kubernetes

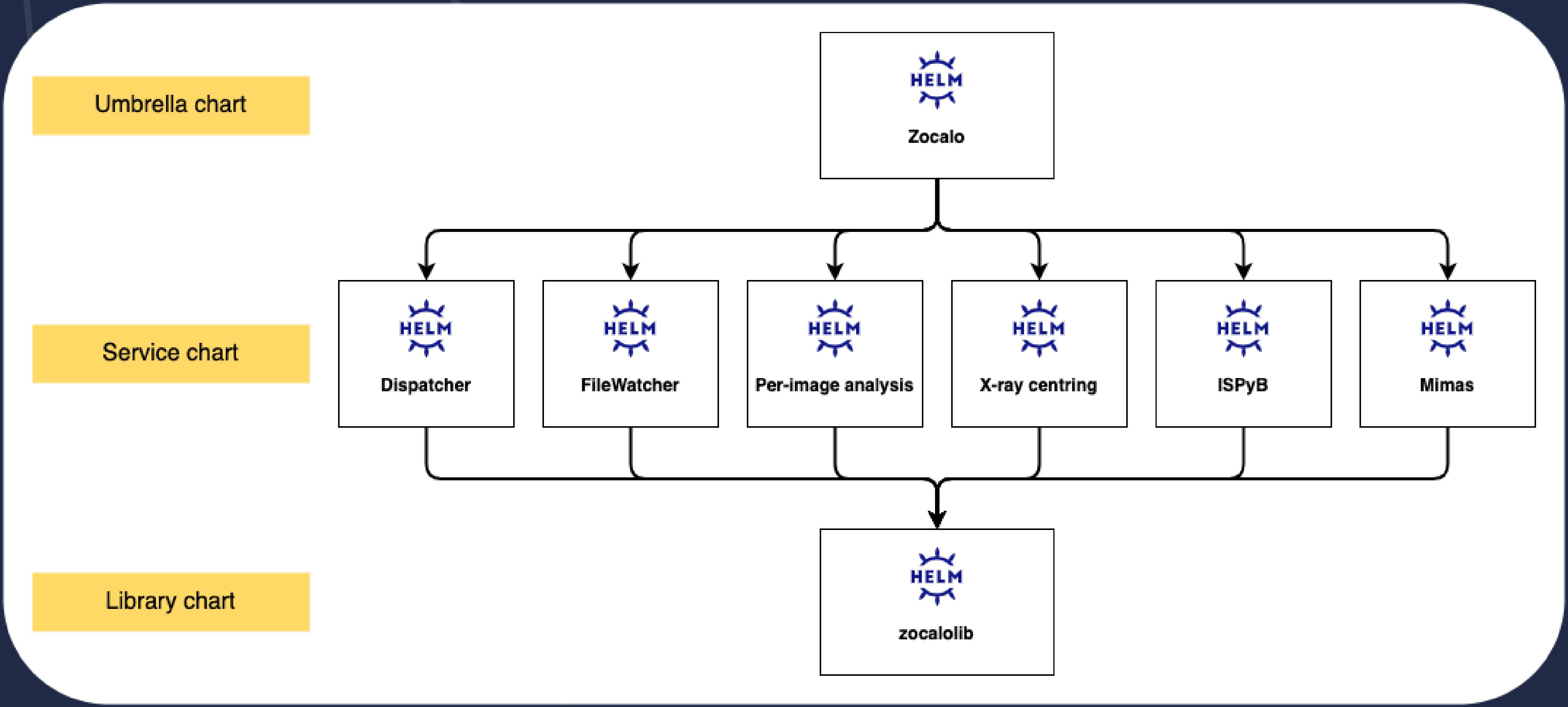
- Open-source system for automating **deployment**, **scaling**, and management of **containerized** applications
- Control and automate application deployments and updates
- Declarative deployment pattern
- Automated rollouts and rollbacks
- Self-healing – automatic restarts of failing containers
- Auto-scaling of applications
- Service discovery

Zocalo on Kubernetes



diamond

Zocalo helm chart



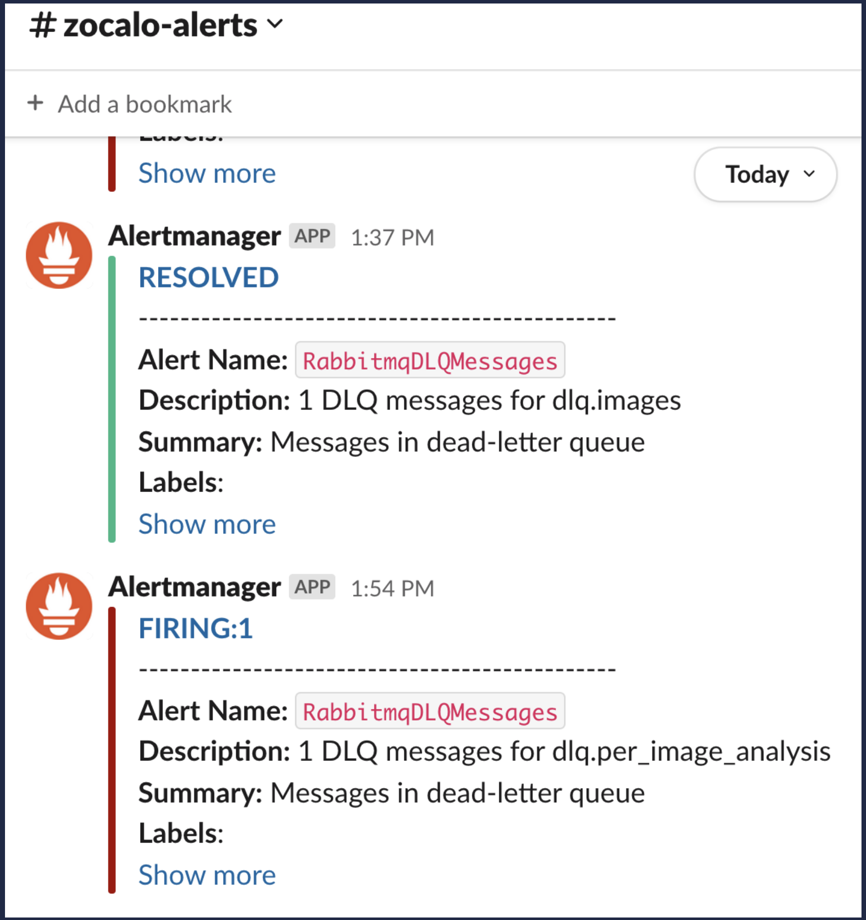
diamond

Monitoring

Is everything OK?

Monitoring

- Take advantage of popular open-source monitoring tooling
  - Prometheus/Alertmanager/Grafana
- Highly configurable alerts via email/slack
- Intelligent grouping of alerts
- Running Zocalo services on Kubernetes provides Prometheus service discovery "for free"
- Grafana dashboard displaying current and historic metrics



zocalo-alerts

+ Add a bookmark

Show more

Today

Alertmanager APP 1:37 PM

RESOLVED

```
---------------------------------------------
```

Alert Name: `RabbitmqDLQMessages`
Description: 1 DLQ messages for dlq.images
Summary: Messages in dead-letter queue
Labels:
Show more

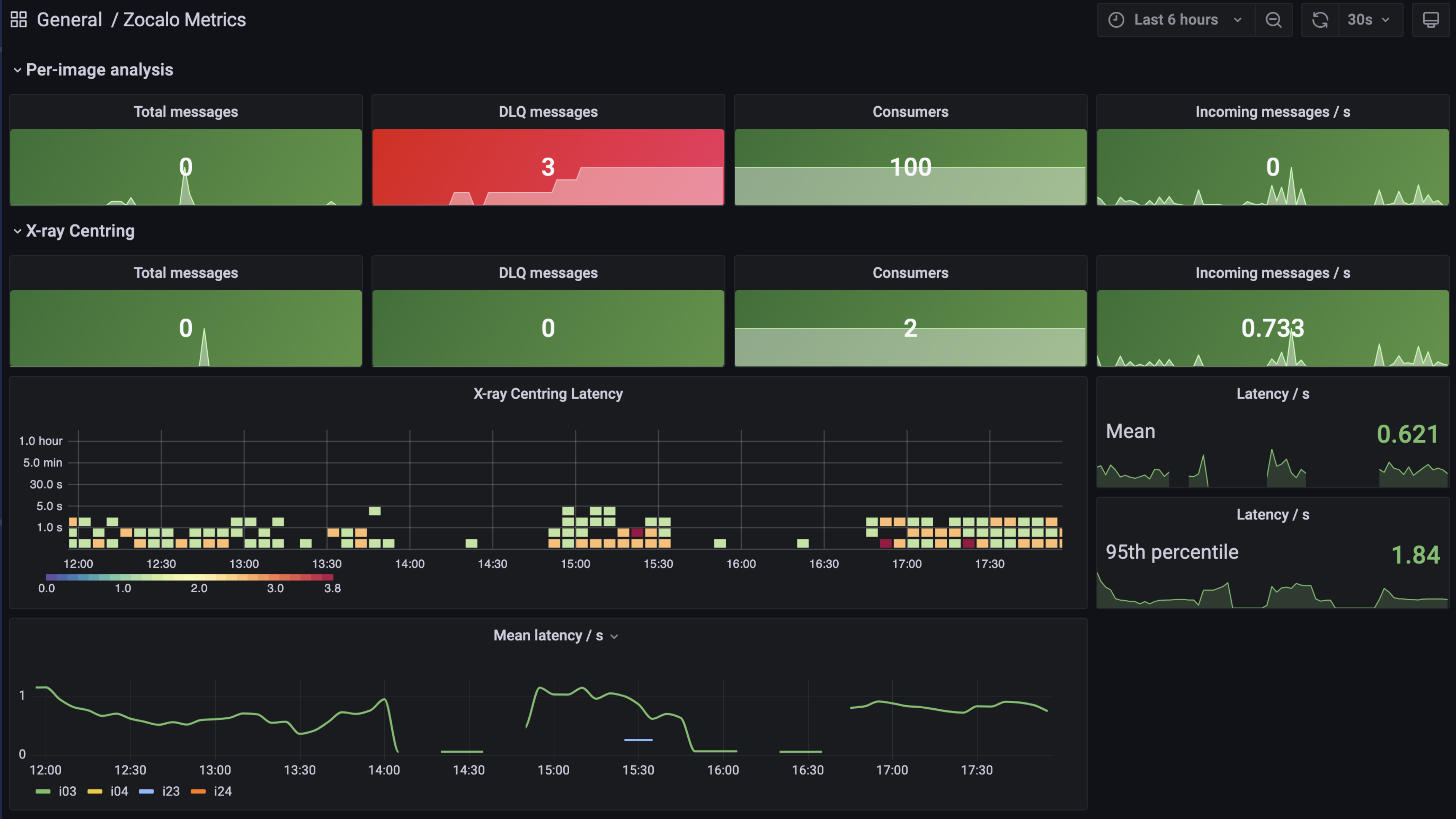
Alertmanager APP 1:54 PM

FIRING:1

```
---------------------------------------------
```

Alert Name: `RabbitmqDLQMessages`
Description: 1 DLQ messages for dlq.per_image_analysis
Summary: Messages in dead-letter queue
Labels:
Show more

diamond



Per-image analysis

Total messages	DLQ messages	Consumers	Incoming messages / s
0	3	100	0

X-ray Centring

Total messages	DLQ messages	Consumers	Incoming messages / s
0	0	2	0.733

X-ray Centring Latency

Latency / s

Mean 0.621

Latency / s

95th percentile 1.84

Mean latency / s

Acknowledgments

Nick Devenish
Irakli Sikharulidze
Graeme Winter
Ben Williams
Markus Gerstel
Dan Hatton
Anna Horstmann
Jacob Filik
Abi Emery
Chris Reynolds
Thomas Hartland
Richard Parke

Availability

https://github.com/DiamondLightSource/python-zocalo

https://github.com/DiamondLightSource/python-workflows

https://zocalo.readthedocs.io/

```
$ conda install -c conda-forge zocalo

$ pip install zocalo
```