

Zocalo: a high-throughput data processing framework

NOBUGS 2022

Richard Gildea

Outline

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

What is Zocalo?

What is Zocalo?

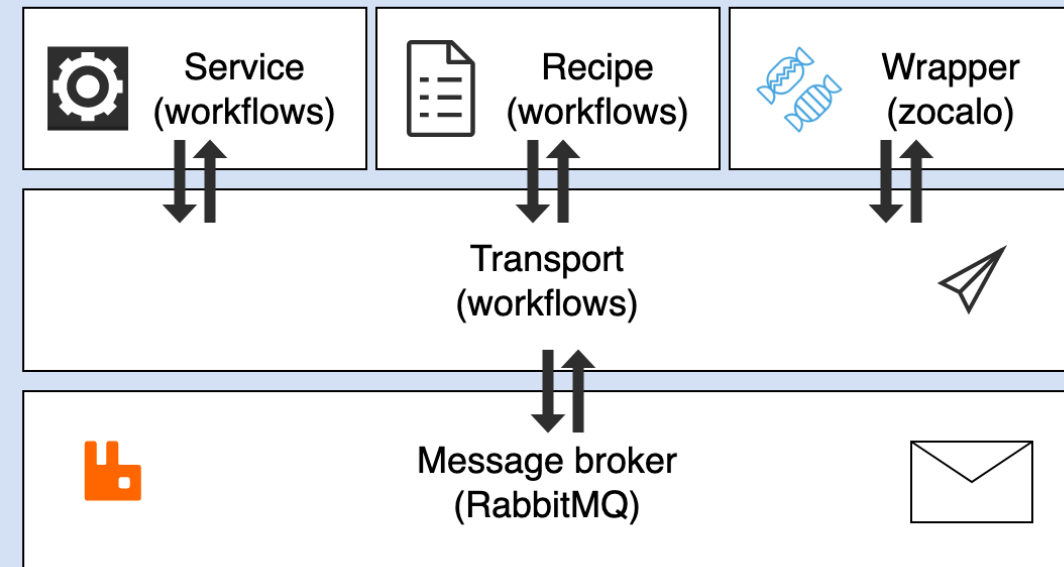
Data analysis infrastructure

- Recipes
- Services
- Wrappers

Communication via

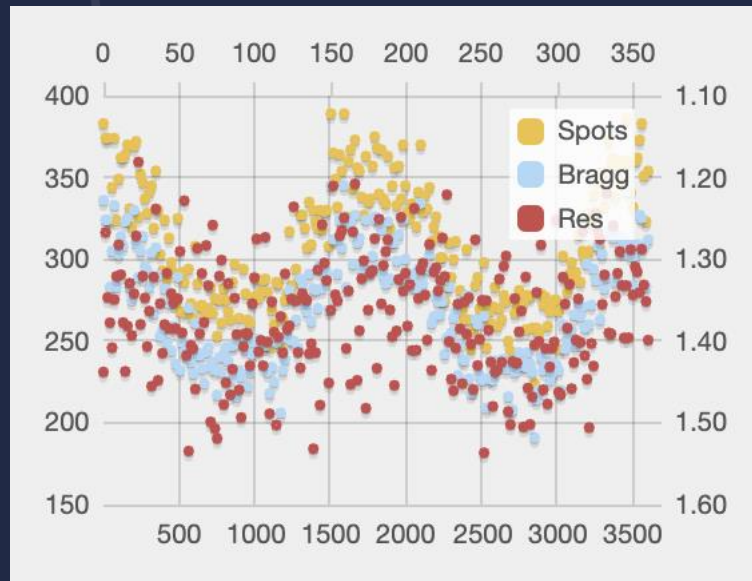
- Transport layer
- Message broker

Zocalo



What can it do?

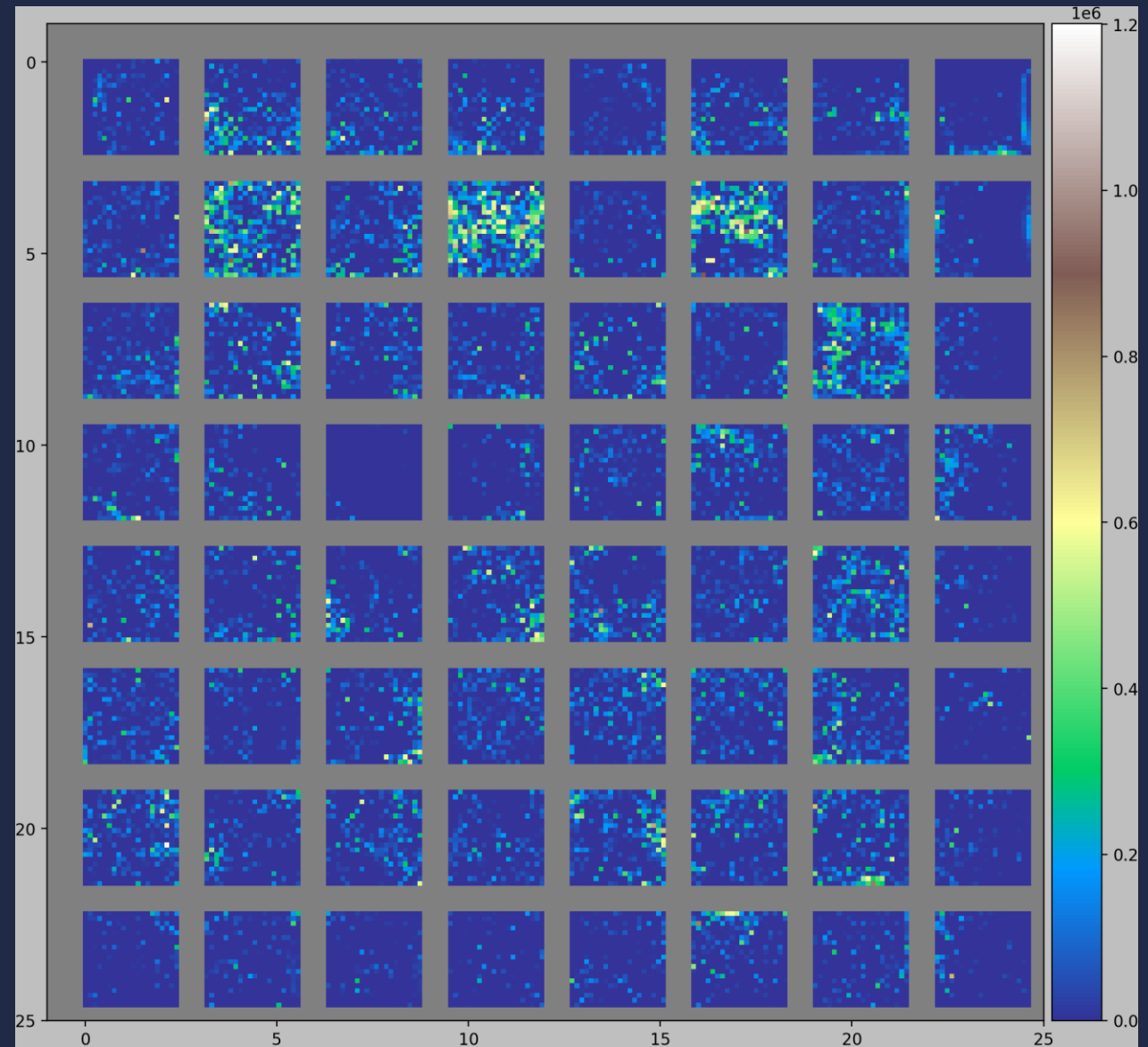
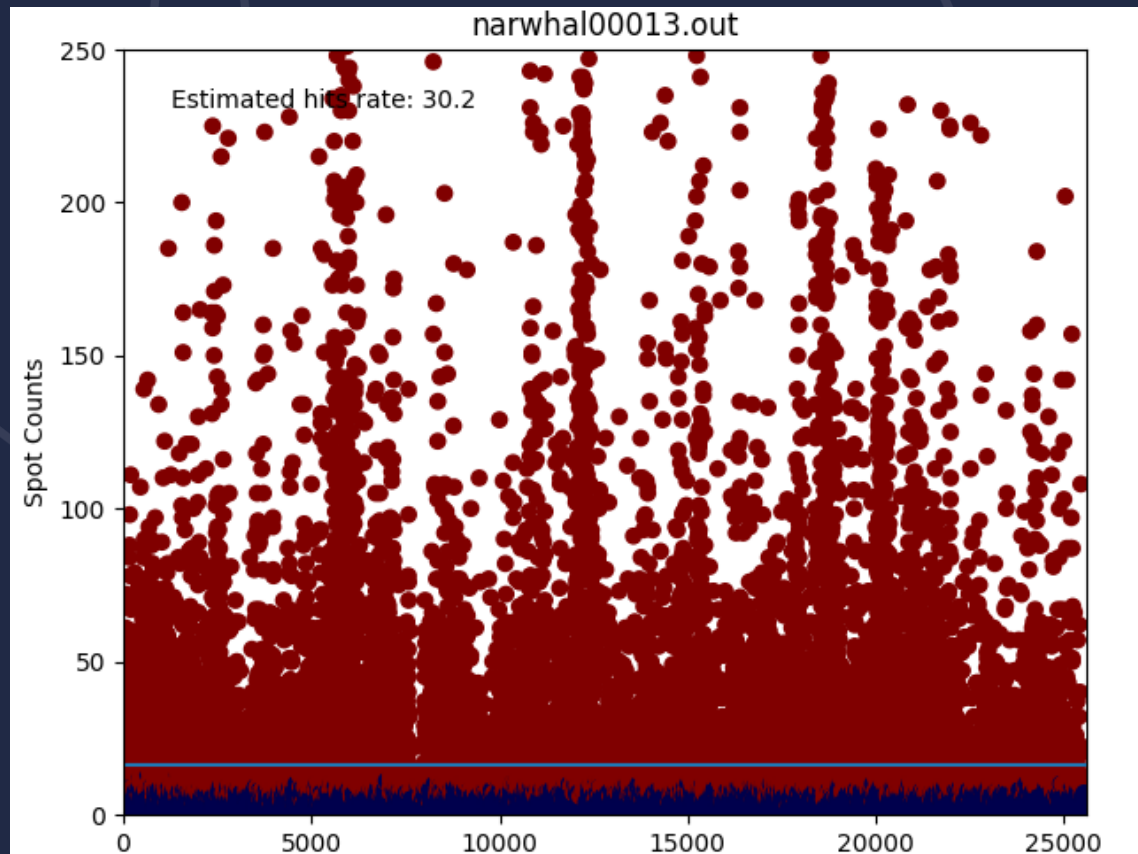
Online analysis



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93
124	123	122	121	120	119	118	117	116	115	114	113	112	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96	95	94
125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155
186	185	184	183	182	181	180	179	178	177	176	175	174	173	172	171	170	169	168	167	166	165	164	163	162	161	160	159	158	157	156
187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217
248	247	246	245	244	243	242	241	240	239	238	237	236	235	234	233	232	231	230	229	228	227	226	225	224	223	222	221	220	219	218
249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279
310	309	308	307	306	305	304	303	302	301	300	299	298	297	296	295	294	293	292	291	290	289	288	287	286	285	284	283	282	281	280
311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341
372	371	370	369	368	367	366	365	364	363	362	361	360	359	358	357	356	355	354	353	352	351	350	349	348	347	346	345	344	343	342
373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403
434	433	432	431	430	429	428	427	426	425	424	423	422	421	420	419	418	417	416	415	414	413	412	411	410	409	408	407	406	405	404
435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465
496	495	494	493	492	491	490	489	488	487	486	485	484	483	482	481	480	479	478	477	476	475	474	473	472	471	470	469	468	467	466
497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527
558	557	556	555	554	553	552	551	550	549	548	547	546	545	544	543	542	541	540	539	538	537	536	535	534	533	532	531	530	529	528
559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	576	577	578	579	580	581	582	583	584	585	586	587	588	589
620	619	618	617	616	615	614	613	612	611	610	609	608	607	606	605	604	603	602	601	600	599	598	597	596	595	594	593	592	591	590
621	622	623	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639	640	641	642	643	644	645	646	647	648	649	650	651
682	681	680	679	678	677	676	675	674	673	672	671	670	669	668	667	666	665	664	663	662	661	660	659	658	657	656	655	654	653	652
683	684	685	686	687	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703	704	705	706	707	708	709	710	711	712	713
744	743	742	741	740	739	738	737	736	735	734	733	732	731	730	729	728	727	726	725	724	723	722	721	720	719	718	717	716	715	714
745	746	747	748	749	750	751	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767	768	769	770	771	772	773	774	775
806	805	804	803	802	801	800	799	798	797	796	795	794	793	792	791	790	789	788	787	786	785	784	783	782	781	780	779	778	777	776
807	808	809	810	811	812	813	814	815	816	817	818	819	820	821	822	823	824	825	826	827	828	829	830	831	832	833	834	835	836	837

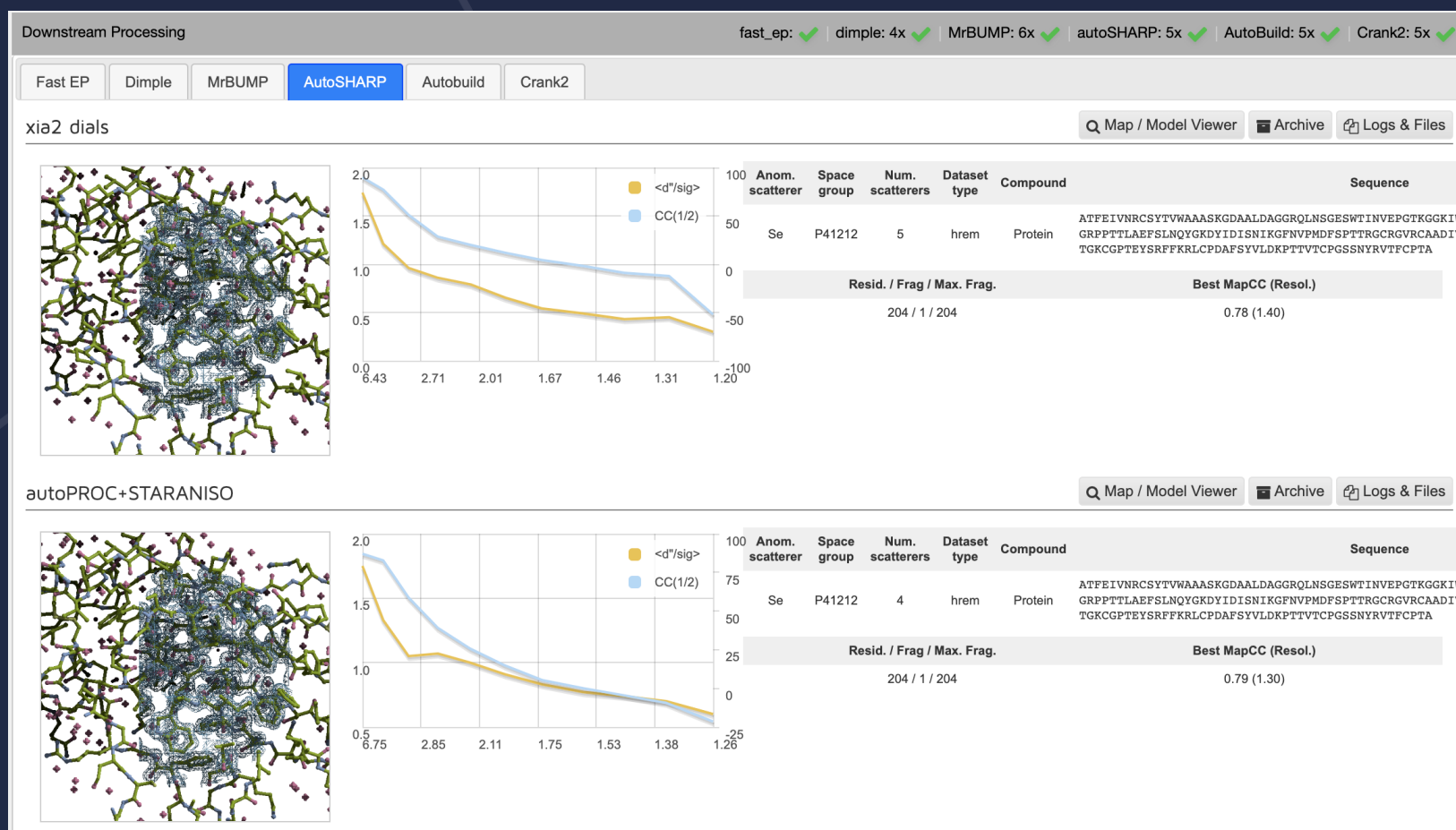
What can it do?

Online analysis



What can it do?

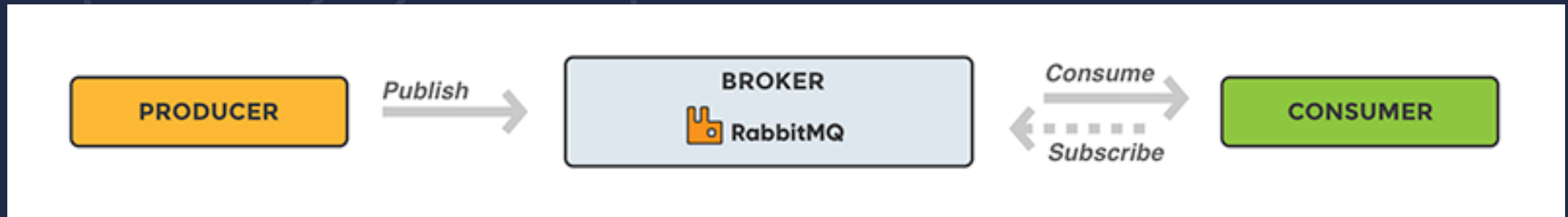
Offline analysis



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/AMQP) 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

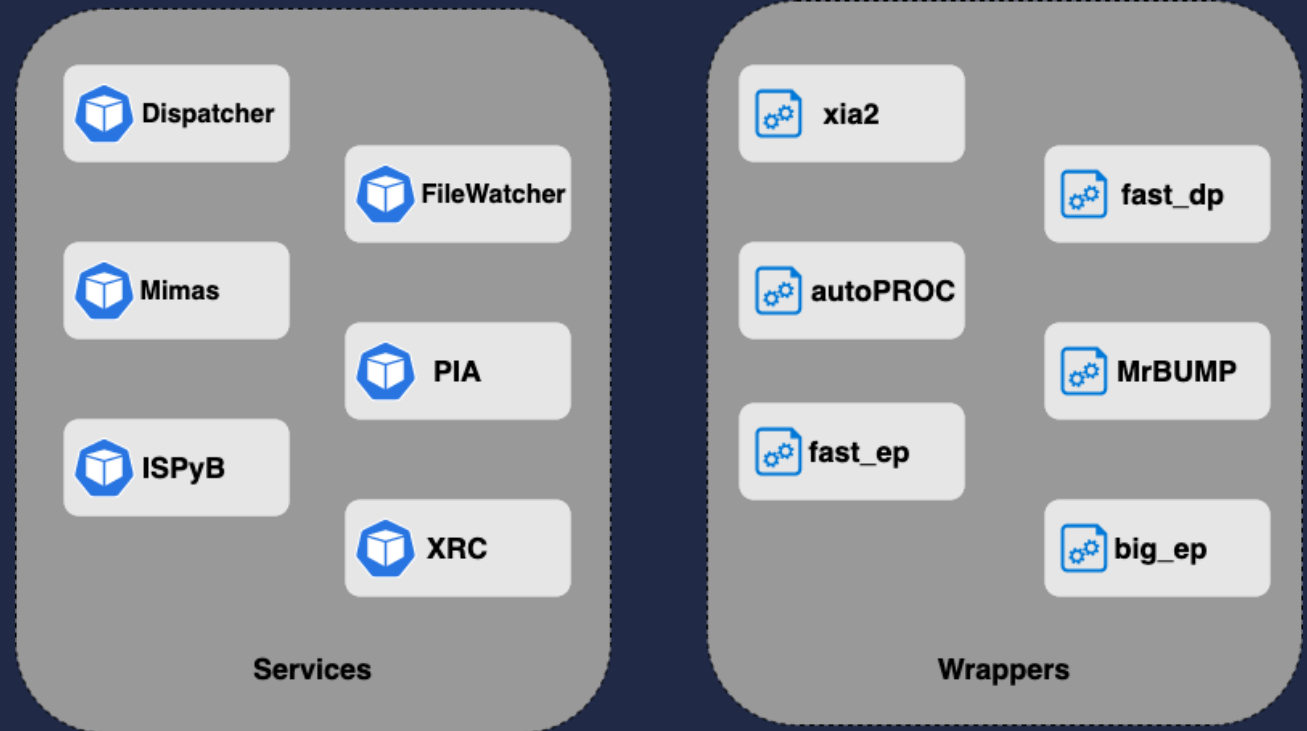
- A **service** consumes messages from a queue, performing some action based on the incoming message
- Optionally sends output to another queue
- Suitable for discrete short-lived tasks, e.g. spotfinding on an individual image or inserting results into a database
- Long-running background processes that 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

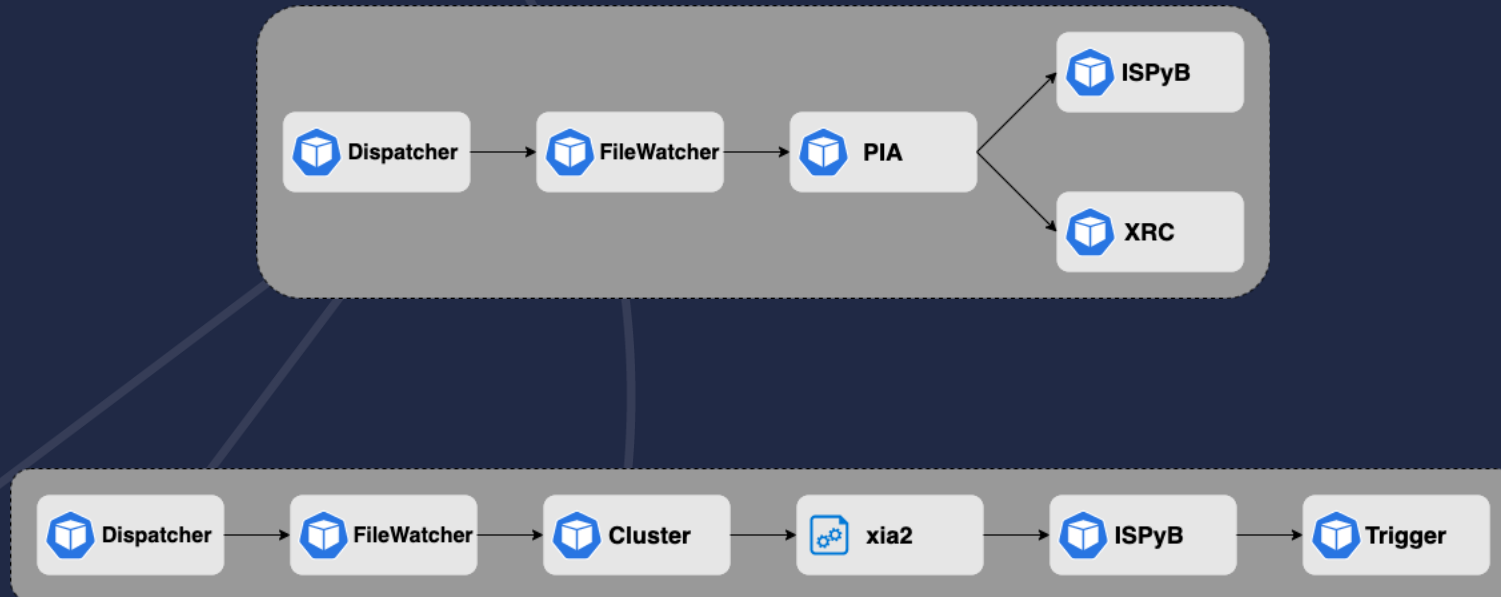
- Used for longer-running tasks, e.g. data processing programs like xia2 or fast_ep
- Only run when needed (typically on in-house cluster or STFC Cloud)
- Wrap something that isn't necessarily aware of zocalo
- A typical wrapper:
 - Takes an input message
 - Converts to suitable command line input
 -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

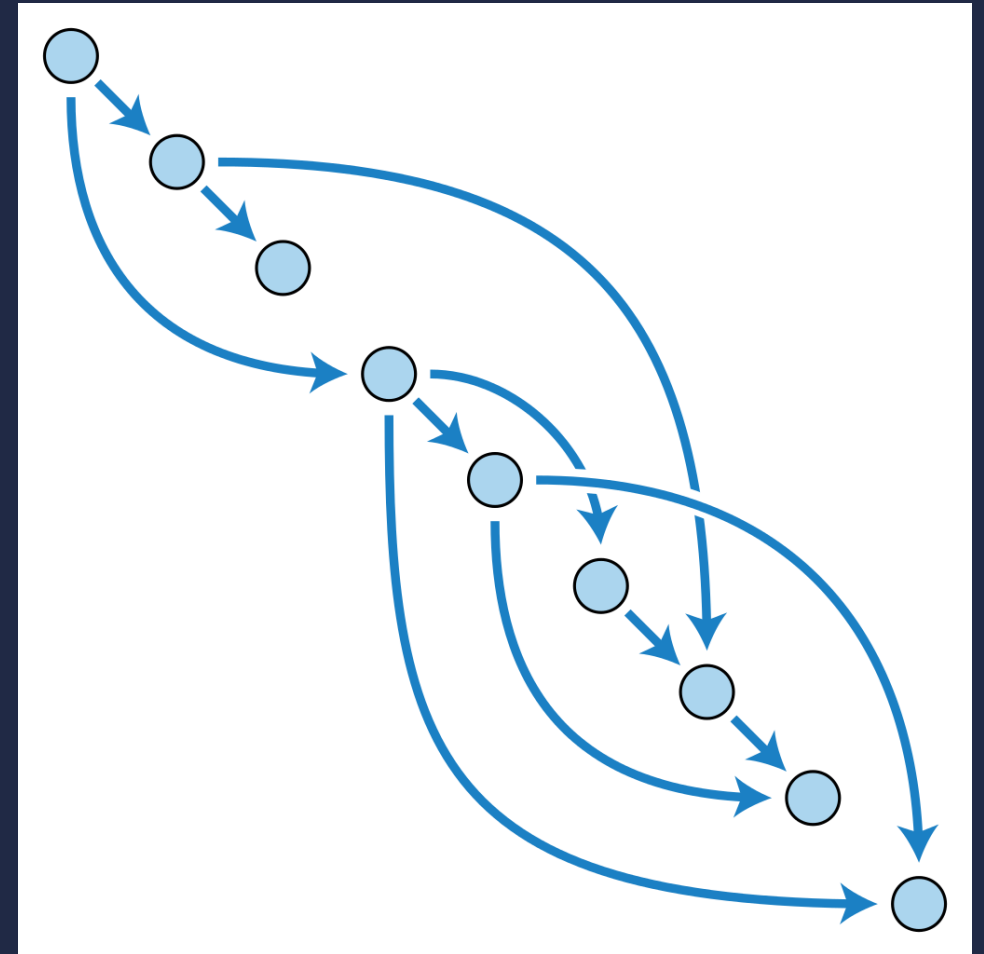


Services and wrappers - now what?



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**

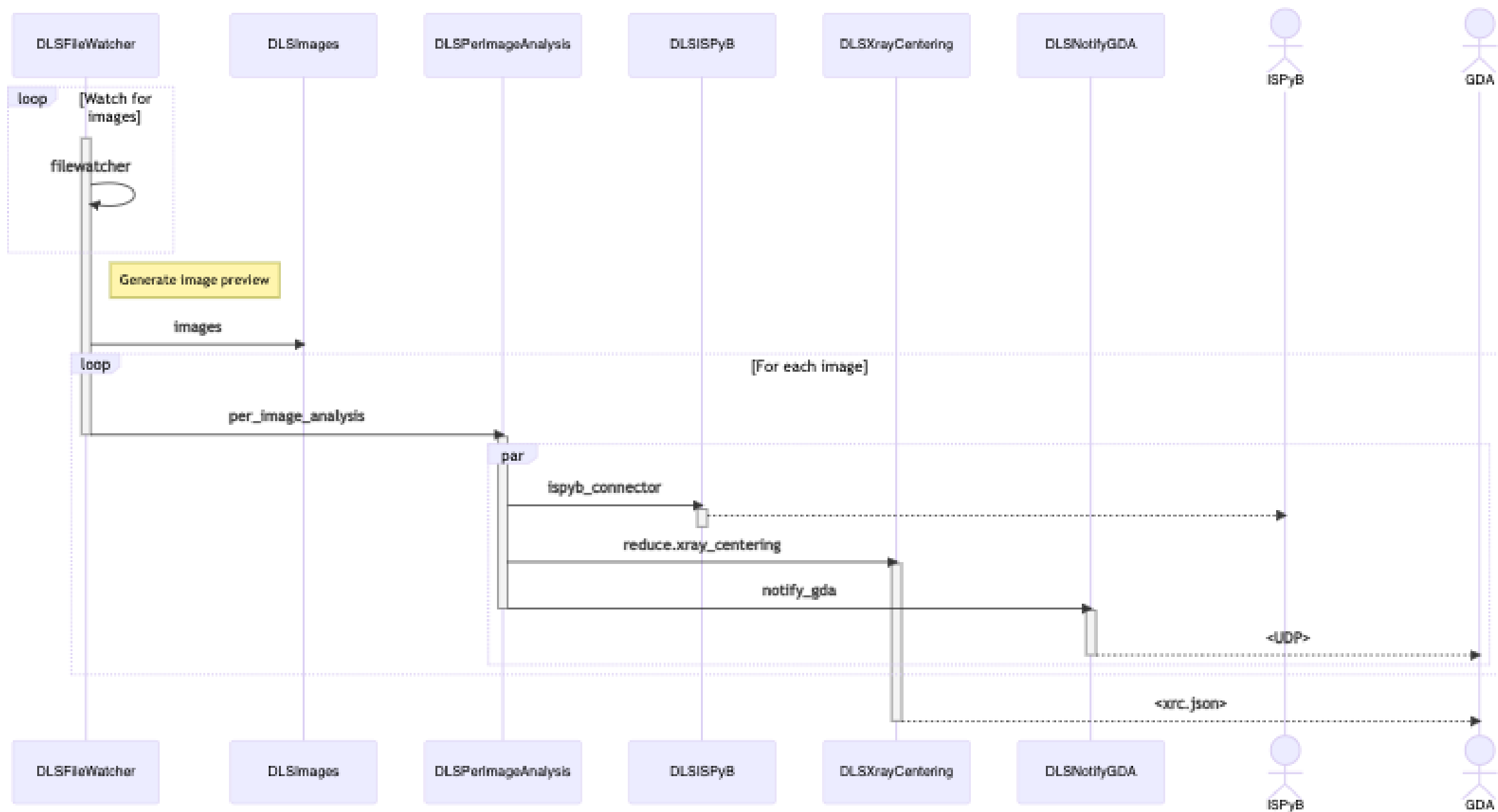


Workflows: recipes

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



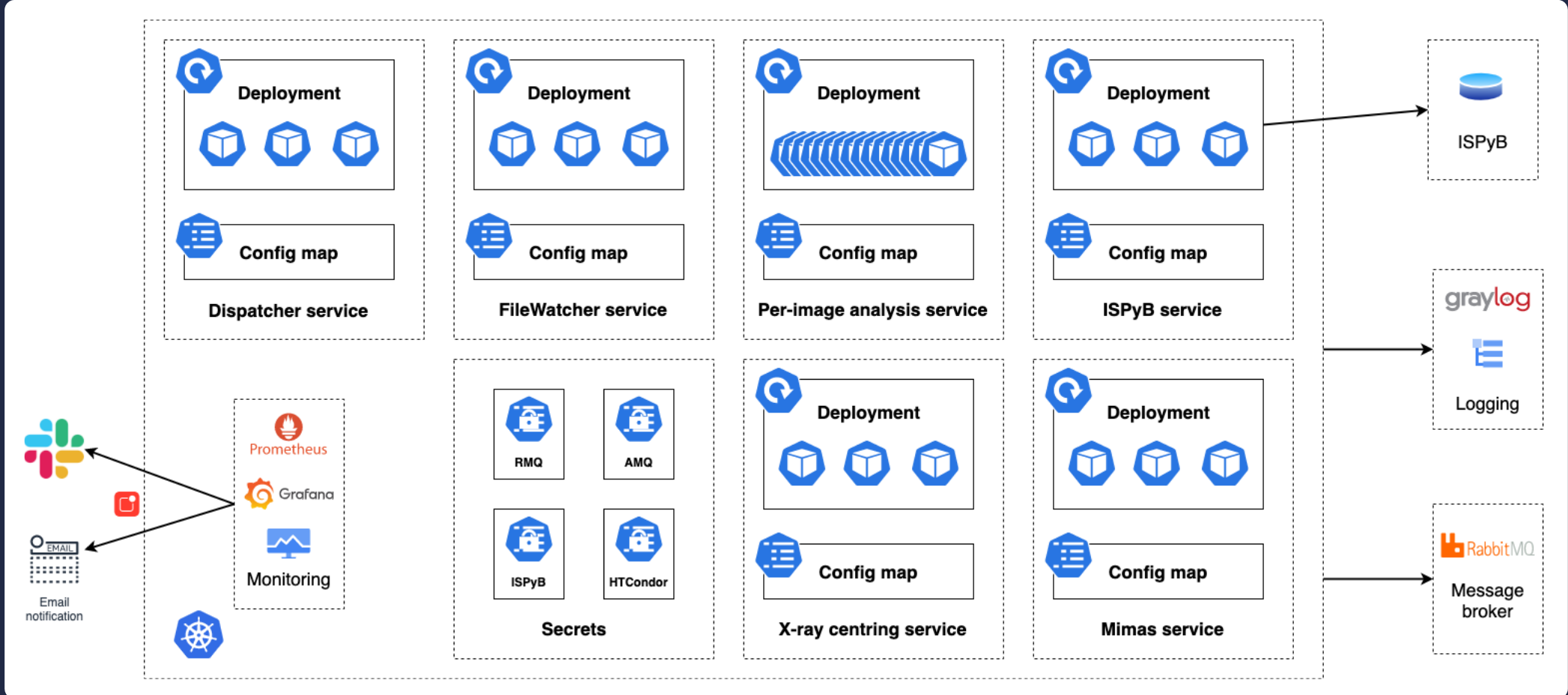
Where do the services run?



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

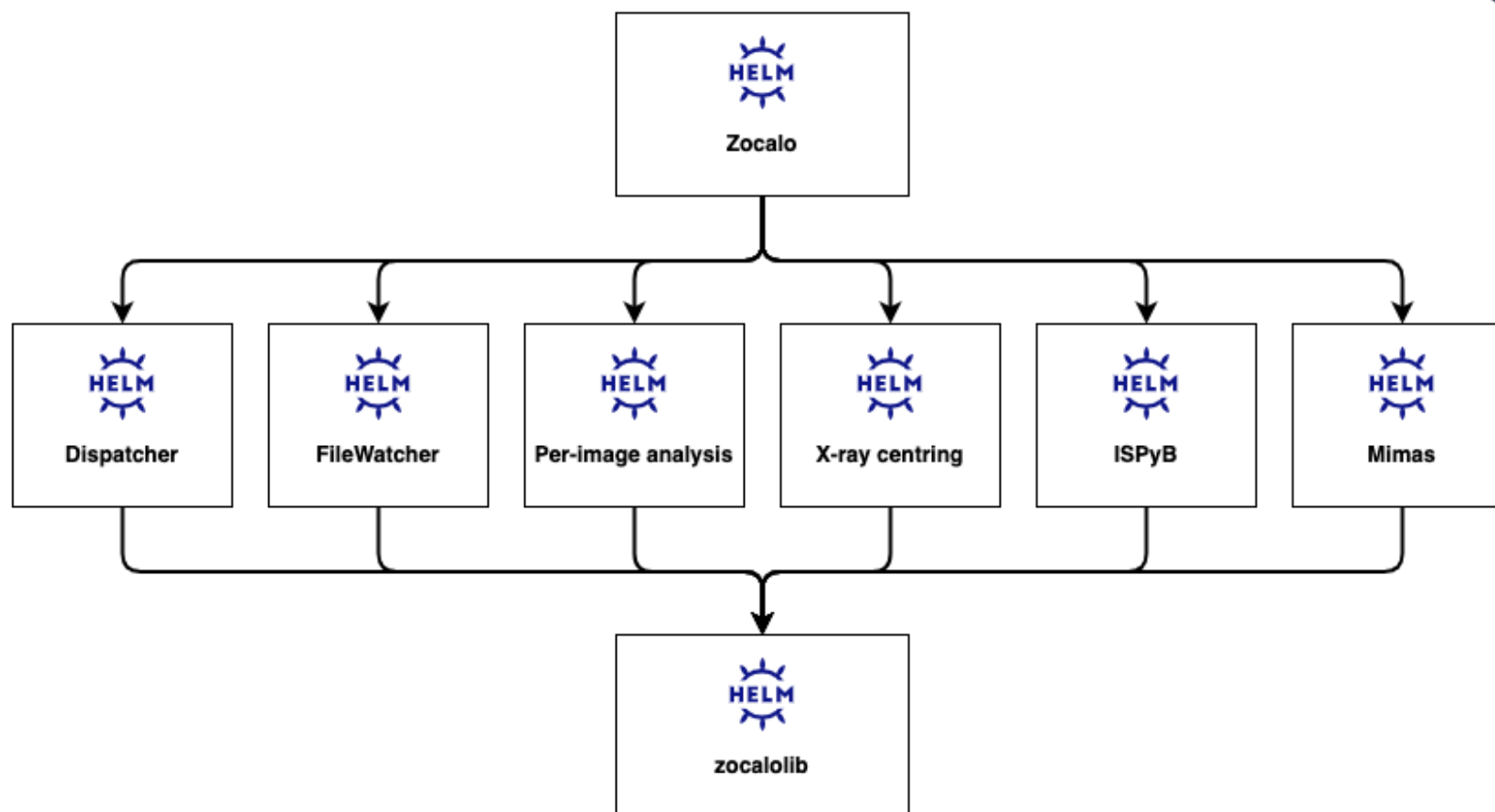


Zocalo helm chart

Umbrella chart

Service chart

Library chart





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

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`

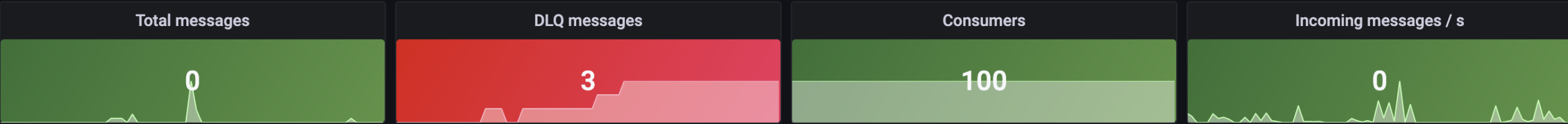
Description: 1 DLQ messages for dlq.per_image_analysis

Summary: Messages in dead-letter queue

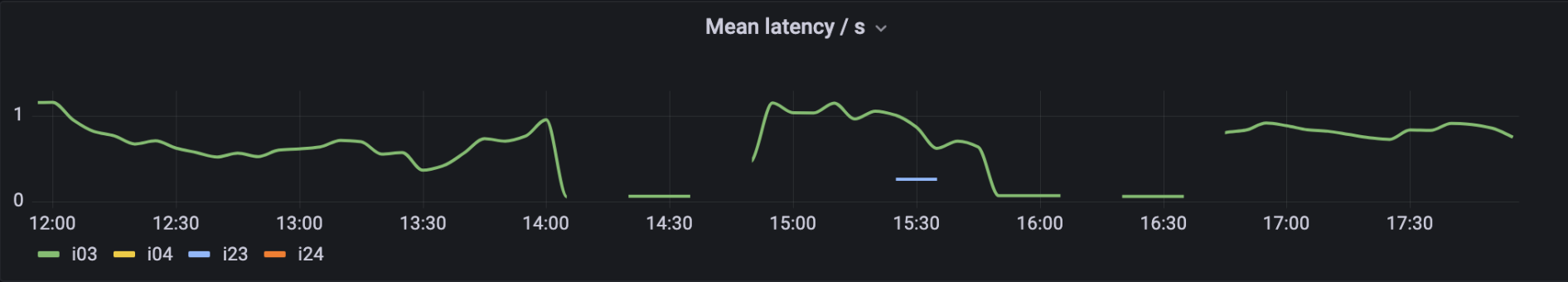
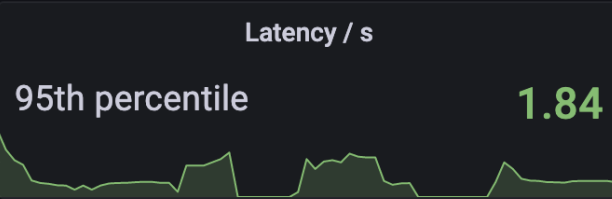
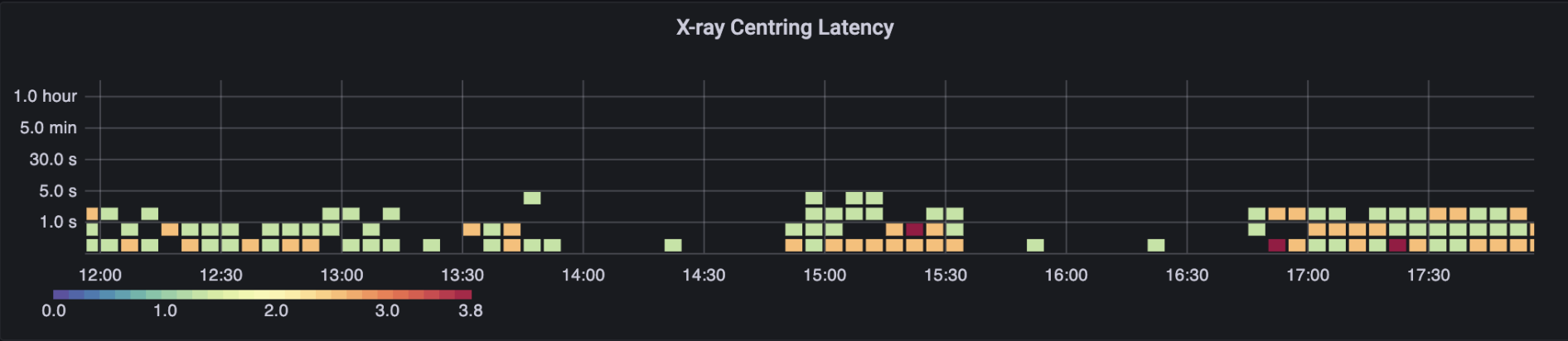
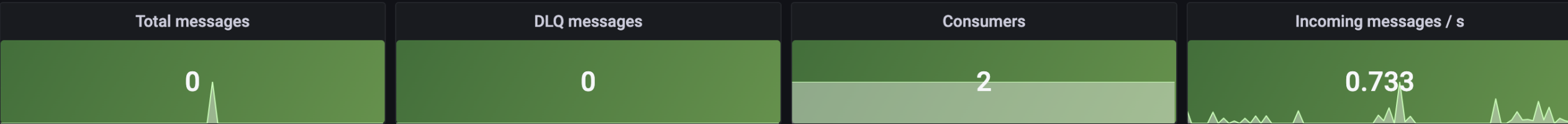
Labels:

[Show more](#)

Per-image analysis



X-ray Centring



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
```