Scalable Orchestration and Dynamic Modeling of Distributed Workloads in SPD Online Filter

Nikita Greben

MLIT JINR

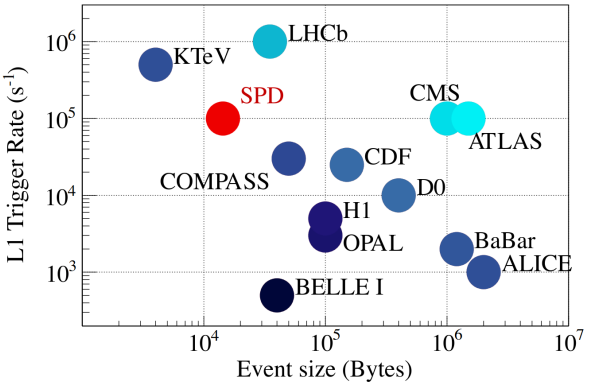
Dubna, 24 November 2026





SPD at the NICA collider

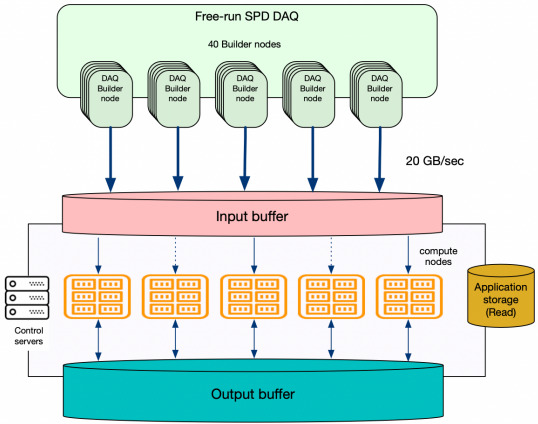
- Polarized protons and deuterons: the spin structure of the nucleon
- Bunches cross every 76.3 ns (13 MHz); ∼3 MHz of collisions
- 20 GB/s, or 200 PB a year; worth keeping: ∼10 PB
- Selection needs tracks and vertices: **no simple trigger**



The SPD Online Filter

The first stage after the DAQ: 200 PB → ∼10 PB a year

- No trigger. The DAQ writes **time slices**, not events
- Physics software rebuilds events, keeps ∼1 in 20
- Hardware: compute nodes between two disk buffers
- Middleware, **our part**: what runs, where and when



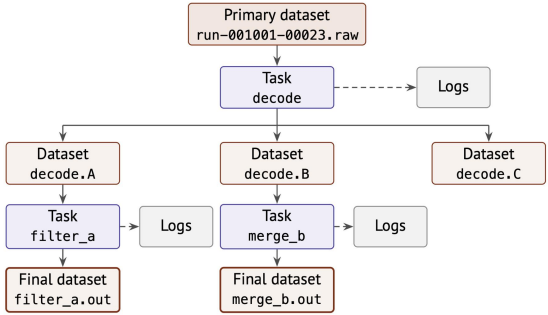
The main goal of SPD Online Filter Middleware

Workflow 1 / Template A

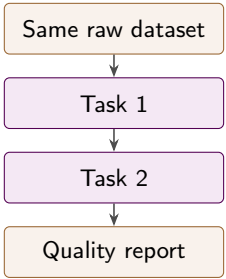
Template (YAML fragment)

```
steps:
  decode:
    outputs: [A, B, C]
  filter_a:
    dependencies: decode.A
    outputs: [out]
    is_final: true
  merge_b:
    dependencies: decode.B
    outputs: [out]
    is_final: true
```

Tree



Workflow 2 / Template B

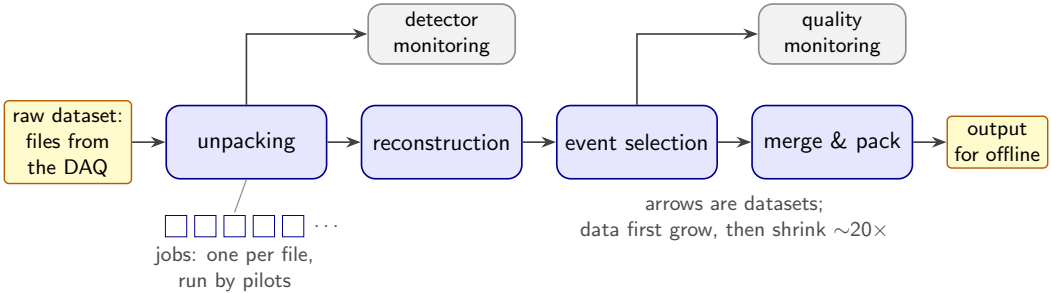


The middleware must execute an arbitrary number of concurrent workflows efficiently and take into account the buffer occupancy: what task to execute now and what not.

This is a major scheduling problem.

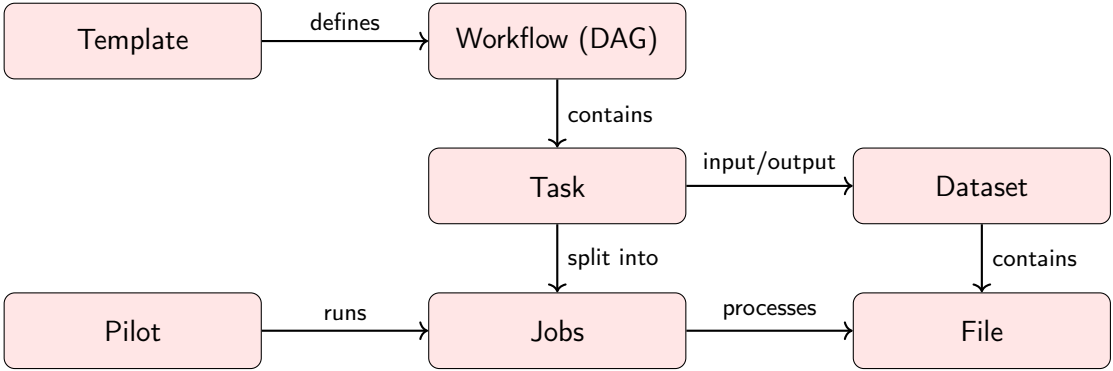
What the middleware runs: workflows

A raw dataset becomes a tree of steps, each run as many small jobs

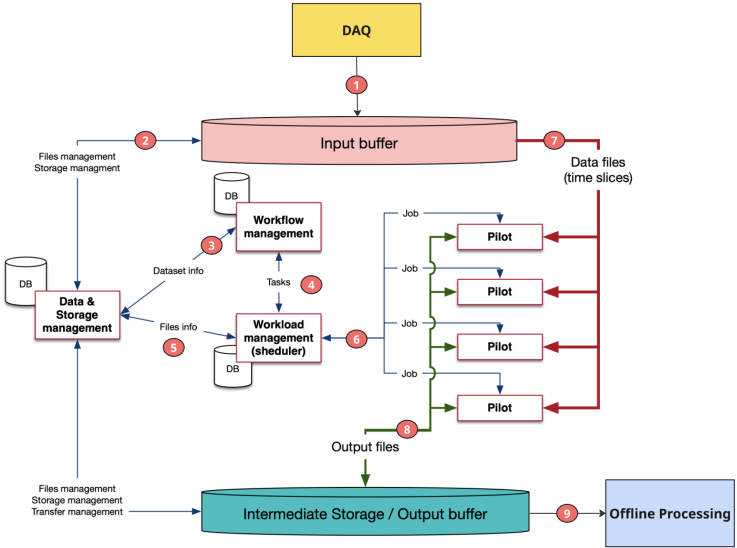


- **Pilot**: a program on a worker VM that pulls the next job

Core concepts

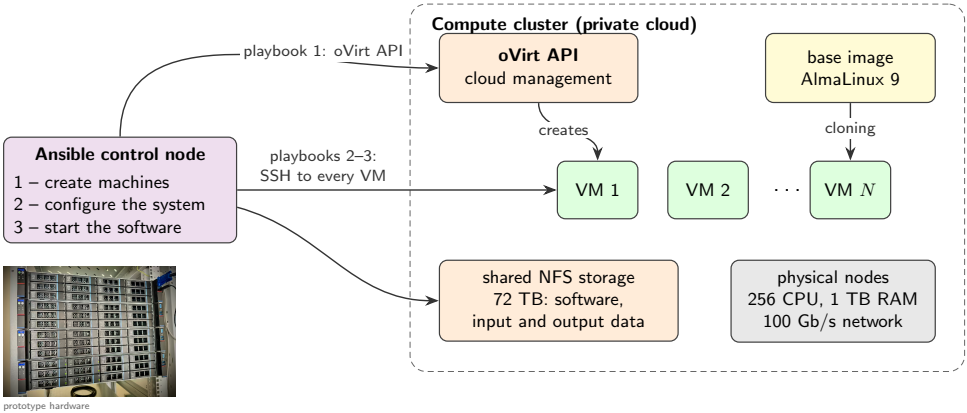


High-level architecture



The prototype cluster

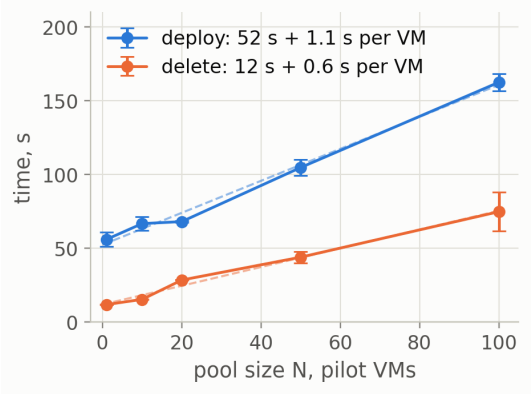
An oVirt-based private cloud



`pilot-ctl` (ours): one command builds or removes a pool of pilots.

It works

100 pilots in under 3 minutes



- It takes minutes from the empty cluster to the first job running.
- Mock jobs for now, SPD has no data yet

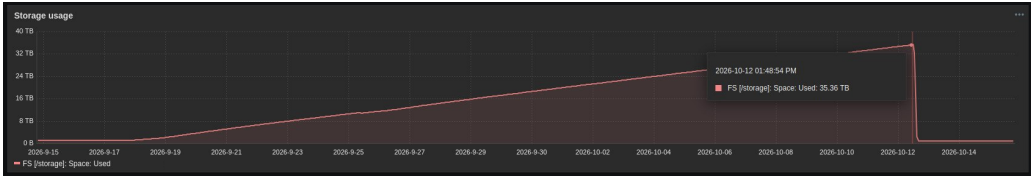
A test run on 52 pilots

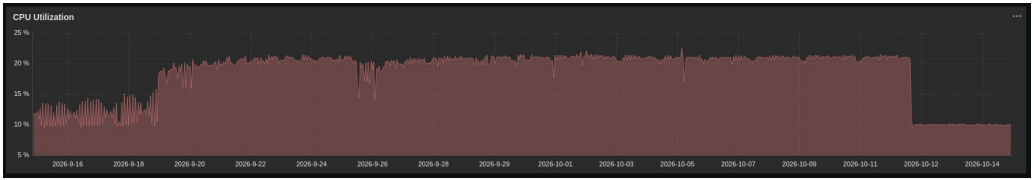
All pilots busy around the clock



- Monitoring: Kafka → OpenSearch → Grafana
- 37 089 job runs, 30 889 completed on the real middleware and cluster

Three weeks of continuous test load

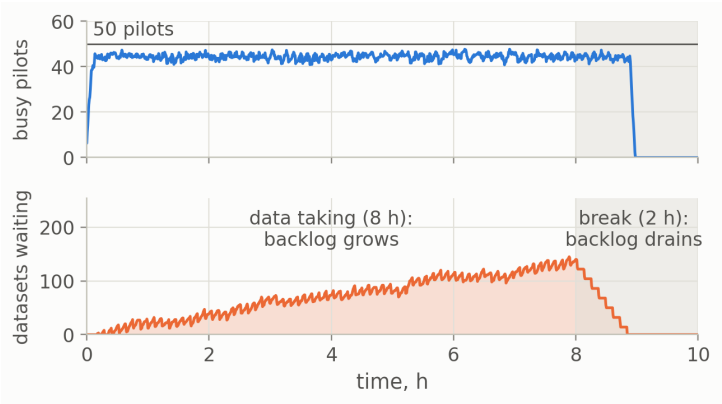




- Disk used: 1.5 → 35.4 TB from 18 September to 12 October, about 1.4 TB a day
- NFS storage CPU about 20 % while the test load runs, 10 % without

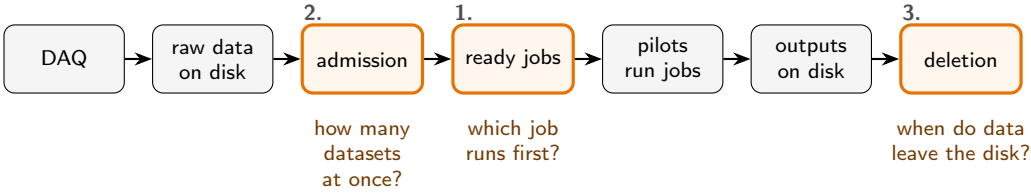
When the DAQ outruns the cluster

Data wait on disk. Scheduler decides who waits and what sits there



- Assumed cycle: 8 h of data taking, then a 2 h break
- Here data come 10 % faster than 50 pilots can process

Three questions



- We tested each one in a model of the new design

How we tested

What arrives

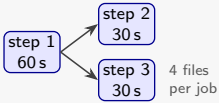


a **dataset**: 10 files × 512 MB

random (Poisson) arrival times, **8 h** of data taking, then a **2 h** break

1.0–1.1 × what the pilots can process: 155–170 datasets an hour

What runs: the new design



mock jobs of the acceptance test's template: fixed run time and output size

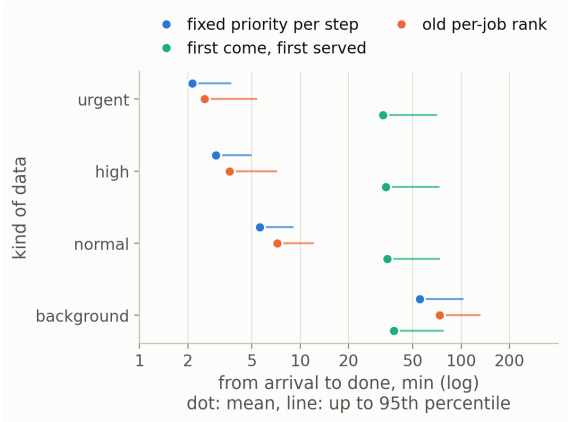
50 pilots; copying in and out: 1 s per 512 MB

What we measure

- time from arrival to done
- idle pilot time
- data on disk: still needed, or released and waiting for deletion

- Every policy gets the same arrivals and run times
- 20 simulated hours per run, 10–30 runs per point, ≈ 2 600 runs

1. Which job runs first?

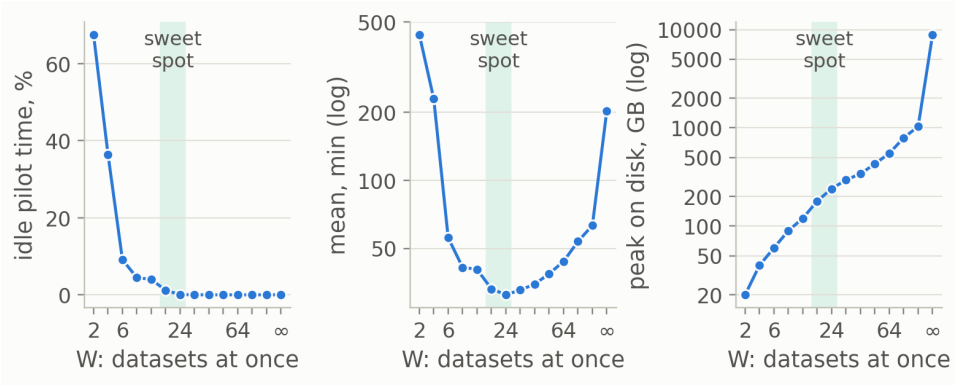


- Urgent to normal: minutes
- First come, first served: everyone takes ∼35 min
- Per-job priority based on age and failures: never better

Model · four classes, datasets of 30 files · data 10 % faster than processing · 30 runs per point

2. How many datasets at once?

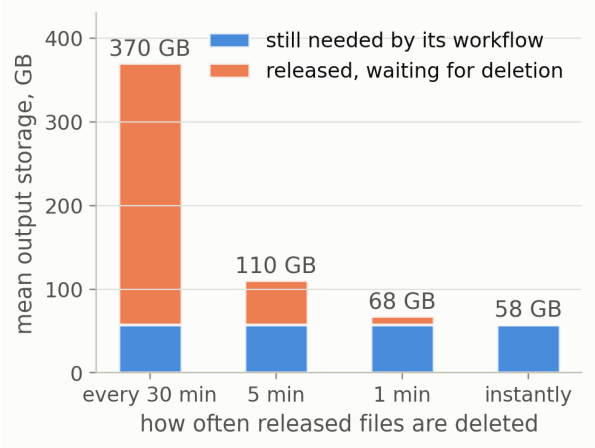
Sweet spot: 18-24 datasets in processing for 50 pilots



- No cap, new first steps overtake old last steps, old workflows stall

Model · test template · data 10 % faster than processing · deletion every minute · 10 runs per point

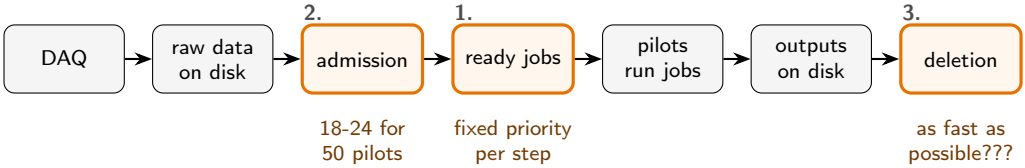
3. When do the data leave the disk?



- Deleted every 30 min, means 84 % is already released
- Every minute: 370 → 68 GB
- But if we delete faster, we might increase the NFS load.

Model · test template · data as fast as processing · 18 datasets at once · 20 runs per point

Conclusions



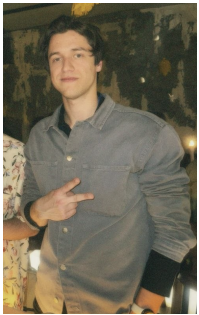
- The number of simultaneous workflows made an impact on the new design of the WfMS.
- The priority handling goes to the level of Tasks, not Jobs.
- There's a real trade-off between how fast we delete the data (increased NFS load) and how much released data we can keep in the system (increased disk usage).

Summary

- Achieved deployment of the entire SPD Online Filter Middleware on the cluster.
- Infrastructure as code and observability turned the middleware into a system we can rebuild, measure and model.
- The new design: static task weights for order, admission window W for the buffer.
- Next is to close the loop on the buffer level, and check the model on the stand, perform complex loading tests to see how the middleware behaves under different situations, and integrate the middleware with the applied software.

Thank you!

Информация об участнике: публикации



1. Greben N., Romanychev L., Oleynik D. et al. SPD On-Line Filter: Workload Management System and Pilot Agent. *Phys. Part. Nuclei* 55, 612–614 (2024). doi:10.1134/S1063779624030419
2. Abazov V. et al. Technical Design Report of the Spin Physics Detector at NICA. *Natural Sci. Rev.* 1, 1 (2024). doi:10.54546/NaturalSciRev.100101
3. Greben N., Oleynik D., Romanychev L. Workload Management System for SPD Online Filter. *Phys. Part. Nuclei Lett.* 22, 1044–1047 (2025). doi:10.1134/S1063779626700395
4. Greben N., Oleynik D., Sychugov A. SPD Online Filter High-Throughput Processing Middleware. *Phys. Part. Nuclei* 57, 627–629 (2026). doi:10.1134/S1063779626700334
5. Greben N., Oleynik D., Romanychev L., Korshunova P. A Unified Middleware for Data, Workflow, and Workload Management in SPD Online Filter. *Phys. Part. Nuclei Lett.* 23, 674–677 (2026). doi:10.1134/S1547477126700718
6. Гребень Н.В., Олейник Д.А., Кирьянов А.К., Сычугов А.А. Автоматизация развертывания вычислительного кластера в облачной инфраструктуре. *Радиотехника*, т. 90, № 9, с. 142–149 (2026). doi:10.18127/j00338486-202609-15
7. Plotnikov A.V., Oleynik D.A., Romanychev L.R., Greben N.V. Workflow Management System for SPD Online Filter. *Phys. Part. Nuclei* 57, 645–647 (2026). doi:10.1134/S1063779626700395
8. Romanychev L.R., Oleynik D.A., Greben N.V. The Design and Implementation of the Pilot Application for the SPD Online Filter. *Phys. Part. Nuclei Lett.* 23, 682–685 (2026). doi:10.1134/S1547477126700731

Информация об участнике: конференции

1. **GRID 2023**, Дубна. «Система управления нагрузкой для специализированной вычислительной системы SPD On-line Filter». Устный доклад.
2. **59th PAC for Particle Physics**, Дубна, 2024. «Workload Management System for SPD Online Filter». Постер.
3. **13th Conference of Young Scientists and Specialists (Alushta-2024)**, Alushta, 2024, «Система управления нагрузкой специализированной вычислительной системы SPD Online Filter». Устный доклад.
4. **AYSS-2024**, Дубна. «Workload Management System Development for SPD Online Filter». Устный доклад.
5. **14th Conference of Young Scientists and Specialists (Alushta-2025)**, Alushta, 2025. «An Overview of SPD Online Filter High-Throughput Processing Middleware». Устный доклад.
6. **GRID 2025**, Дубна. «SPD Online Filter High-Throughput Processing Middleware». Устный доклад.
7. **29th Young Scientists Conference**, Дубна, 2025. «A Unified Middleware for Data, Workflow, and Workload Management in SPD Online Filter». Устный доклад.
8. **15th Conference of Young Scientists and Specialists (Alushta-2026)**, Alushta, 2026. «Prototyping the SPD Online Filter as a Private Cloud Infrastructure: Towards a Dynamical System Formulation». Устный доклад.
9. **30th Young Scientists Conference**, Ташкент, 2026. «Scalable Orchestration and Dynamic Modeling of Distributed Workloads in SPD Online Filter». Устный доклад.

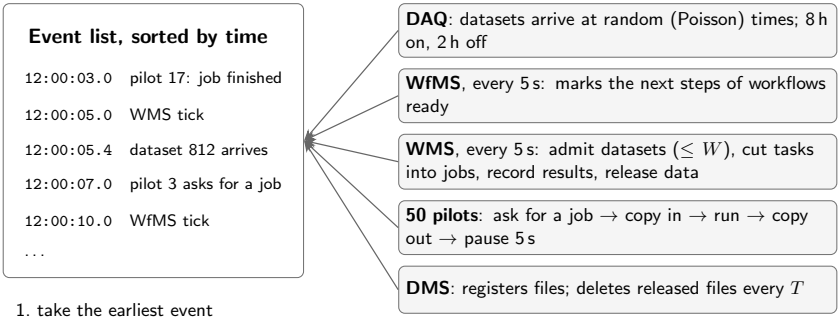
Backup

The model setup

	Test template	**Four classes**
Dataset	10 files $\times$ 512 MB	30 files of 3 / 1 / 0.3 / 0.05 GB: background, normal, high, urgent (40 / 35 / 20 / 5 %)
Workflow	test template of the acceptance test: step 1, 60 s (output = input) $\rightarrow$ step 2, 30 s (half the size), and step 3, 30 s, 4 files per job (half the size)	one step: 120 ± 30 / 60 ± 15 / 30 ± 8 / 15 ± 5 s; output 5 % of input
Failures	none	20 / 8 / 4 / 2 % per launch
Priorities	step weights 8 / 5 / 4	1 / 3 / 7 / 15
Arrivals	Poisson during 8 h of data taking, 2 h break; load 0.8, 1.0, 1.1 (125, 155, 170 datasets/h)	same; load 1.0, 1.1, 1.2
Datasets at once	$W = 2 \ldots \infty$	$W = 6$
Deletion	every 30 min; also 5, 1, 0 min	every 30 min
Runs	20 h, 20 runs per point	20 h, 30 runs per point

50 CPU pilots; scheduler ticks every 5 s, 50 jobs per tick; copying in and out 1 s per 512 MB; 5 s pilot pause between jobs; 3 launches per job (first try + 2 retries), a retry stays at the head of its task; deletion is instant when it runs. Every policy gets the same random arrivals, run times and failures.

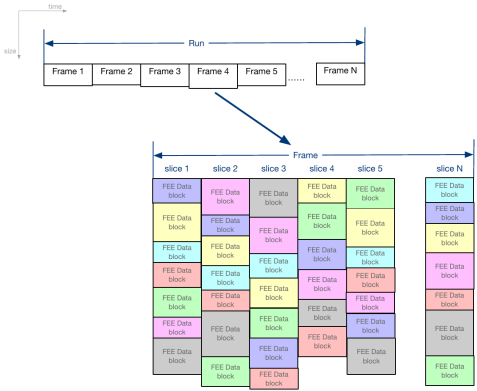
How the event-driven simulation runs



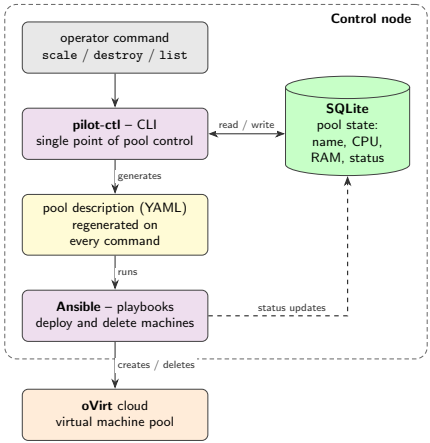
- One run: 20 simulated hours, about 640 000 events
- Every policy gets the same random arrivals, run times and failures

What the DAQ writes

- Time slices of 1 μs – 8.3 ms, packed into frames of up to 549 s and 16 GB
- A slice holds detector data blocks from several collisions
- The Online Filter rebuilds events from them

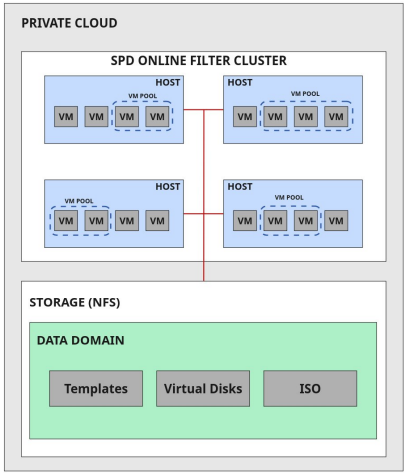


Managing the pilot pool with `pilot-ctl`



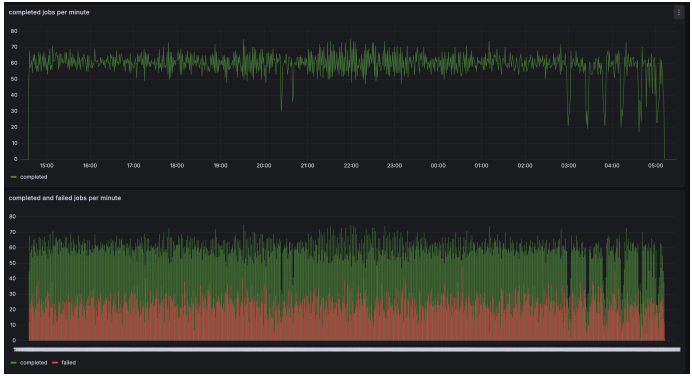
- Ansible alone keeps no state: it does not know which VMs exist
- `pilot-ctl` owns the pool state (SQLite) and regenerates the Ansible inputs on every command
- Measured: deploy 52 s per VM, delete 12 s per VM

The cluster layout



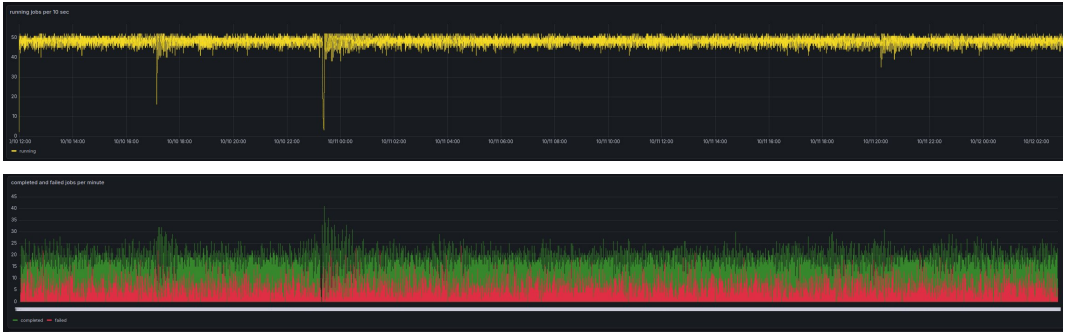
- **Four servers** with the same specs. One also hosts the shared storage.
- **Per server.** Two AMD EPYC 7513 CPUs, 64 physical cores (128 threads), 256 GB RAM and 2 TB NVMe.
- **Network.** 100 Gb/s in total.
- **Shared storage.** Six 18 TB SAS drives in RAID 6 provide 72 TB of usable space.

An earlier 15-hour test run



- 52 012 jobs completed, 20 588 attempts failed; each job gets 3 attempts

The 50-pilot run, wider window



Simple flow estimate

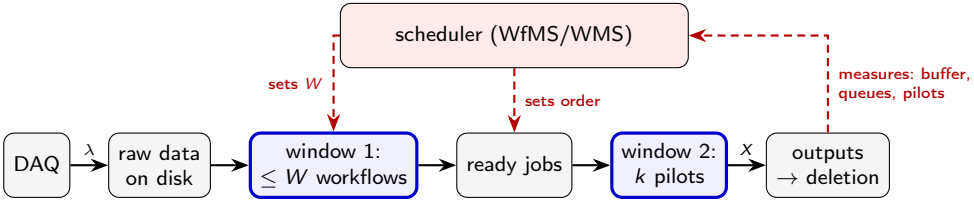
$$\frac{dQ}{dt} = \lambda(t) - \mu, \quad Q \geq 0, \qquad \lambda(t) = \begin{cases} \lambda, & \text{data taking, } 8\,\text{h} \\ 0, & \text{break, } 2\,\text{h} \end{cases}$$

$$Q_{n+1} = \max\bigl(0,\ Q_n + (\lambda(t_n) - \mu)\,\Delta t\bigr), \qquad Q_0 = 0, \quad \Delta t = 0.01\,\text{h}$$

$$8\,\text{h} \cdot (\lambda - \mu) \ \leq\ 2\,\text{h} \cdot \mu \quad \Longleftrightarrow \quad \mu \geq 0.8\,\lambda$$

Q	data waiting on disk	minutes of data taking
λ	data rate of the detector	1 (the other rates relative to it)
μ	what the pilots process	0.7, 0.9, 1.0 (the three curves)
$\rho = \lambda/\mu$	load of the pilots while taking data	1.1 in our model, so $\mu \approx 0.91\,\lambda$

An open system with a control loop



SPD Online Filter Middleware

SPD Online Filter is a **primary** data processing facility designed for the high-throughput, multi-step processing of data from the SPD detector.

Hardware component

Compute cluster with two storage systems and a set of working nodes: currently multi-CPU, GPU accelerated nodes are planned for the next phase.

Middleware component

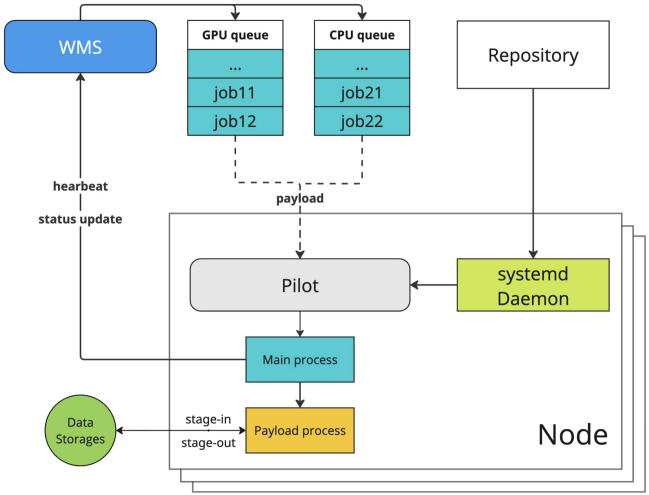
Software complex for reliable management of multistep data processing and efficient loading of the computing facility.

Applied software

Performs informational processing of data: unscrambling, reconstruction, etc.

Pilot architecture

On each compute node a lightweight agent which runs the actual applied software.



Example: pilot-ctl scale 100



pilot-ctl result: 100 compute nodes created and launched

```
pilot-agent-85  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-86  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-87  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-88  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-89  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-90  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-91  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-92  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-93  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-94  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-95  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-96  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-97  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-98  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
pilot-agent-99  : ok=16  changed=11  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0

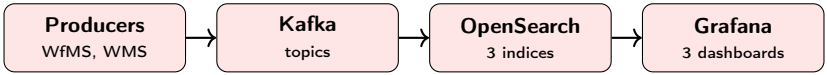
Promoted 100 pilot(s) to Running.
```

3 minutes starting from VM creation to running state on all 100 compute nodes.

oVirt monitor: ≈ 100 active compute nodes

		Name	Com	Host	IP Addresses	FQDN	Cluster	Data Cen	Memory	CPU
		AlmaLinux10.1-te		sof-srv1.jinr.ru	10.17.245.40 fe80::546f:e0ff:fe38:0	ansible	SOF	JINR	12%	0%
		daq-generator		sof-srv4.jinr.ru	10.17.245.45 fe80::546f:e0ff:fe38:4e	daq	SOF	JINR	14%	0%
		dms-dev		sof-srv1.jinr.ru	10.17.245.49 fe80::546f:e0ff:fe38:5a	dms-dev	SOF	JINR	12%	0%
		HostedEngine		sof-srv4.jinr.ru	10.17.245.31 fe80::216:3eff:fe52:c95b	sof-virt.jinr.ru	SOF	JINR	43%	2%
		pilot-agent-257		sof-srv1.jinr.ru	10.17.245.248 fe80::546f:e0ff:fe38:1c	pilot-agent-257	SOF	JINR	20%	26%
		pilot-agent-258		sof-srv1.jinr.ru	10.17.245.177 fe80::546f:e0ff:fe38:1d	pilot-agent-258	SOF	JINR	20%	19%
		pilot-agent-259		sof-srv1.jinr.ru	10.17.245.127 fe80::546f:e0ff:fe38:1e	pilot-agent-259	SOF	JINR	21%	22%
		pilot-agent-280		sof-srv3.jinr.ru	10.17.245.210 fe80::546f:e0ff:fe38:1f	pilot-agent-280	SOF	JINR	20%	29%
		pilot-agent-281		sof-srv3.jinr.ru	10.17.245.236 fe80::546f:e0ff:fe38:20	pilot-agent-281	SOF	JINR	20%	23%
		pilot-agent-282		sof-srv2.jinr.ru	10.17.245.168 fe80::546f:e0ff:fe38:21	pilot-agent-282	SOF	JINR	20%	19%
		pilot-agent-283		sof-srv3.jinr.ru	10.17.245.148 fe80::546f:e0ff:fe38:22	pilot-agent-283	SOF	JINR	20%	24%
		pilot-agent-284		sof-srv2.jinr.ru	10.17.245.78 fe80::546f:e0ff:fe38:23	pilot-agent-284	SOF	JINR	20%	18%
		pilot-agent-285		sof-srv3.jinr.ru	10.17.245.216 fe80::546f:e0ff:fe38:24	pilot-agent-285	SOF	JINR	20%	23%
		pilot-agent-286		sof-srv3.jinr.ru	10.17.245.235 fe80::546f:e0ff:fe38:25	pilot-agent-286	SOF	JINR	20%	27%
		pilot-agent-287		sof-srv3.jinr.ru	10.17.245.138 fe80::546f:e0ff:fe38:26	pilot-agent-287	SOF	JINR	20%	23%
		pilot-agent-288		sof-srv3.jinr.ru	10.17.245.81 fe80::546f:e0ff:fe38:27	pilot-agent-288	SOF	JINR	20%	19%
		pilot-agent-289		sof-srv3.jinr.ru	10.17.245.202 fe80::546f:e0ff:fe38:28	pilot-agent-289	SOF	JINR	20%	20%
		pilot-agent-290		sof-srv4.jinr.ru	10.17.245.163 fe80::546f:e0ff:fe38:29	pilot-agent-290	SOF	JINR	20%	16%
		pilot-agent-291		sof-srv4.jinr.ru	10.17.245.136 fe80::546f:e0ff:fe38:2a	pilot-agent-291	SOF	JINR	20%	25%

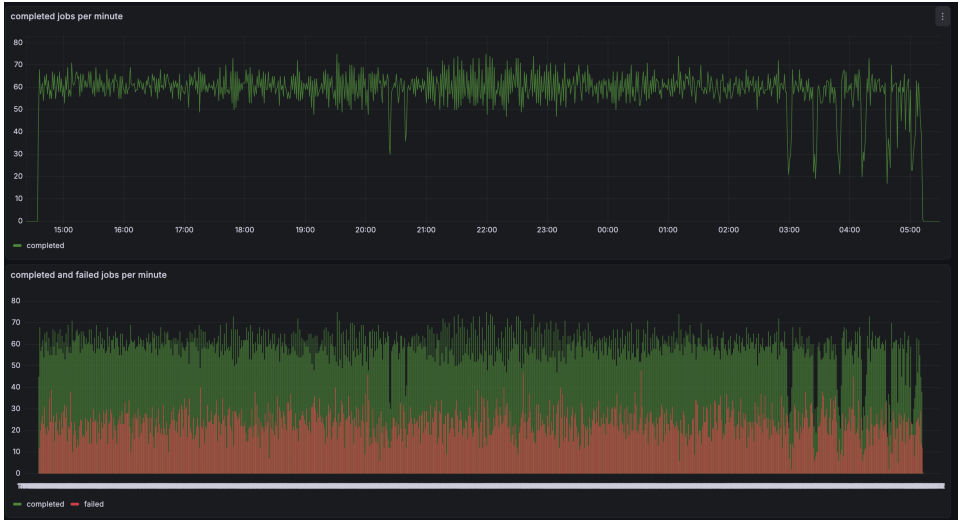
Observability: Kafka → OpenSearch → Grafana



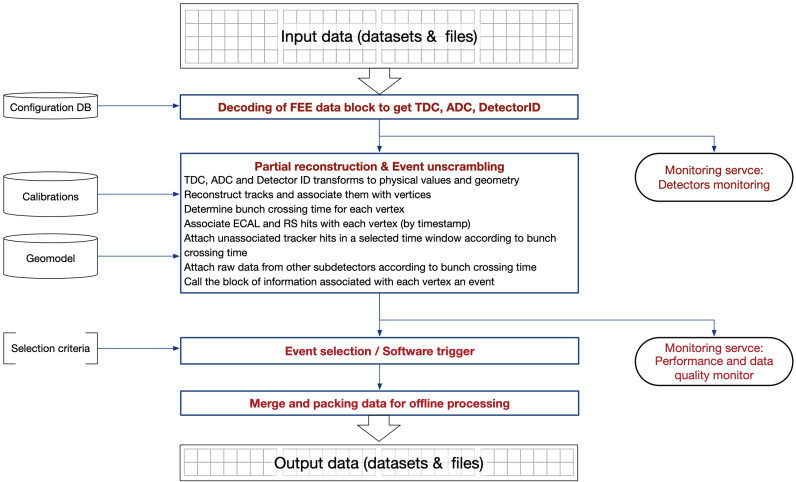
What gets collected:

Stream	Time field	Captured fields
Job lifecycle events	`datetime`	`job_id`, `task_id`, `state`, `error_code`, `exit_code`, `pilot_id` on every state transition
Pilot heartbeats	`heartbeat_ts`	`pilot_id`, `cpu_load`, memory, disk IO, network – at fixed interval
Task & dataset	`datetime`	`task_id`, `name`, `datasets`, ...

Jobs dashboards example: jobs per minute

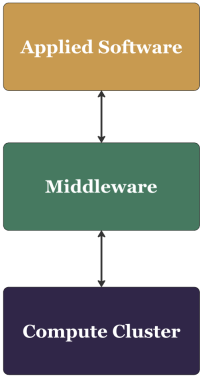


Data processing workflow in SPD Online Filter

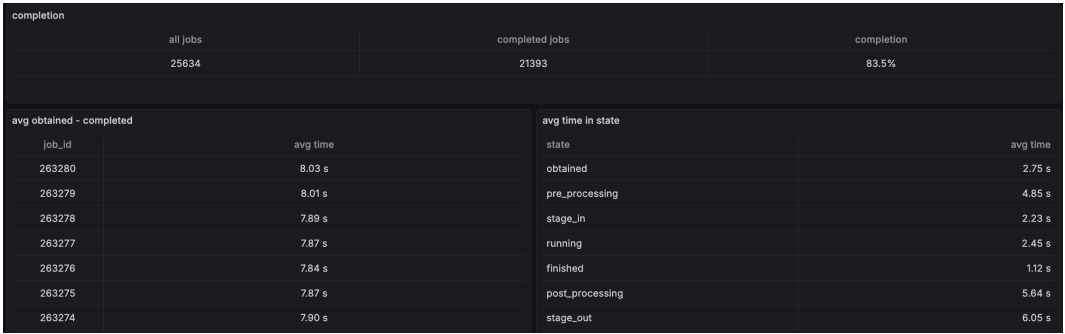


Core systems

- **Data Management System**
  - Data lifecycle support (data catalog, consistency check, cleanup, storage);
- **Workflow Management System**
  - Define and execute data processing chains by generating the required number of computational tasks;
- **Workload management System**
  - Create the required number of processing jobs to perform the task;
  - Control job execution through pilots working on compute nodes;
  - Handles efficient use of resources.



Jobs dashboards example



completion

all jobs	completed jobs	completion
25634	21393	83.5%

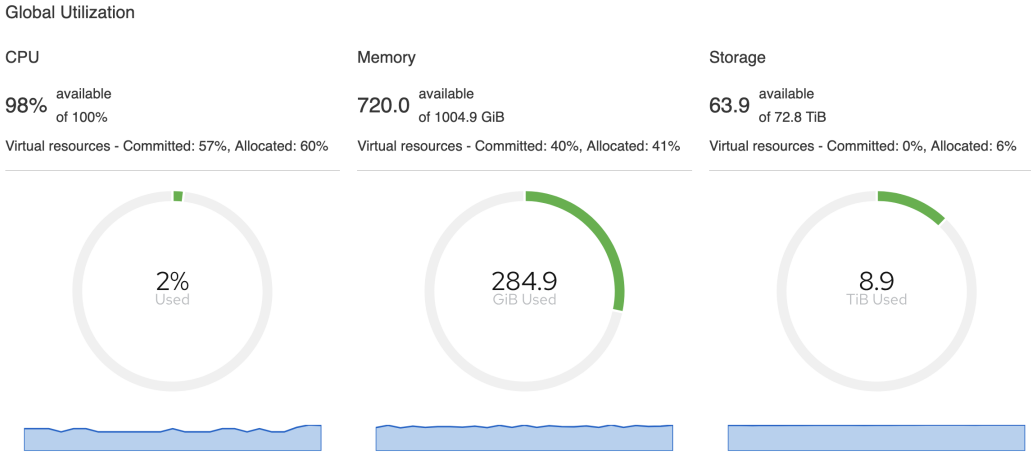
avg obtained - completed

job_id	avg time
263280	8.03 s
263279	8.01 s
263278	7.89 s
263277	7.87 s
263276	7.84 s
263275	7.87 s
263274	7.90 s

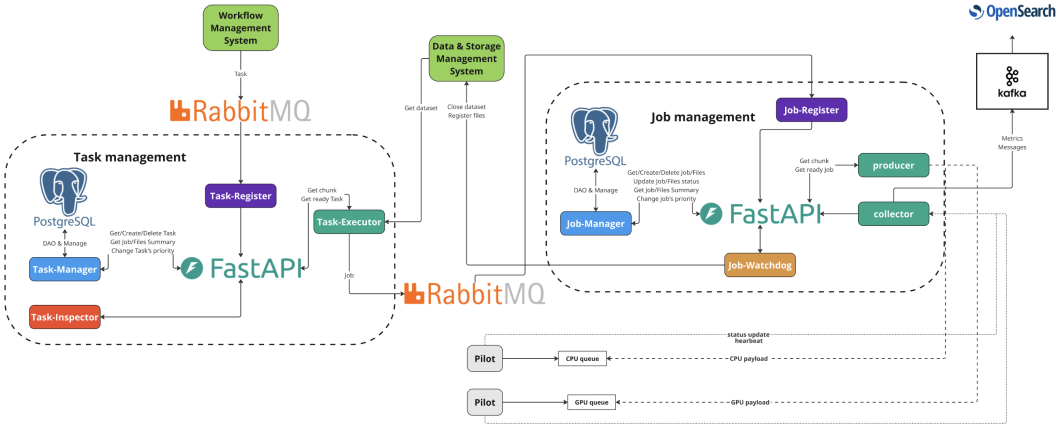
avg time in state

state	avg time
obtained	2.75 s
pre_processing	4.85 s
stage_in	2.23 s
running	2.45 s
finished	1.12 s
post_processing	5.64 s
stage_out	6.05 s

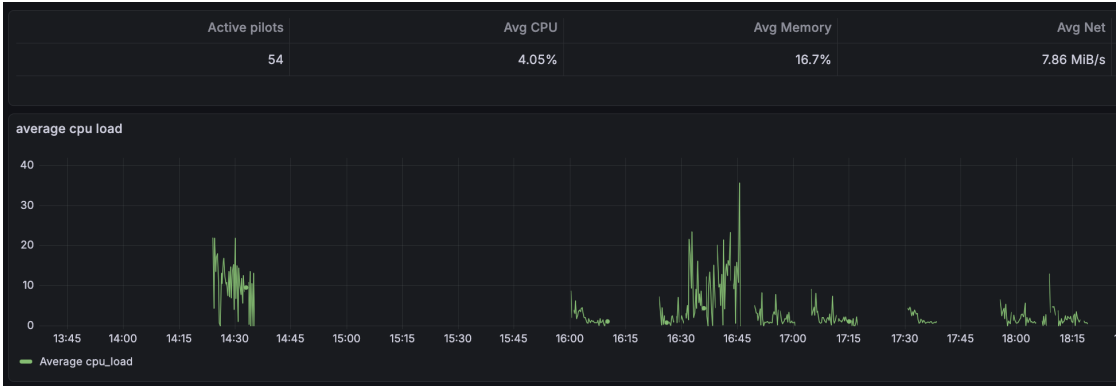
oVirt usage


Global Utilization
CPU
98% available of 100%
Virtual resources - Committed: 57%, Allocated: 60%
2% Used
Memory
720.0 available of 1004.9 GiB
Virtual resources - Committed: 40%, Allocated: 41%
284.9 GiB Used
Storage
63.9 available of 72.8 TiB
Virtual resources - Committed: 0%, Allocated: 6%
8.9 TiB Used

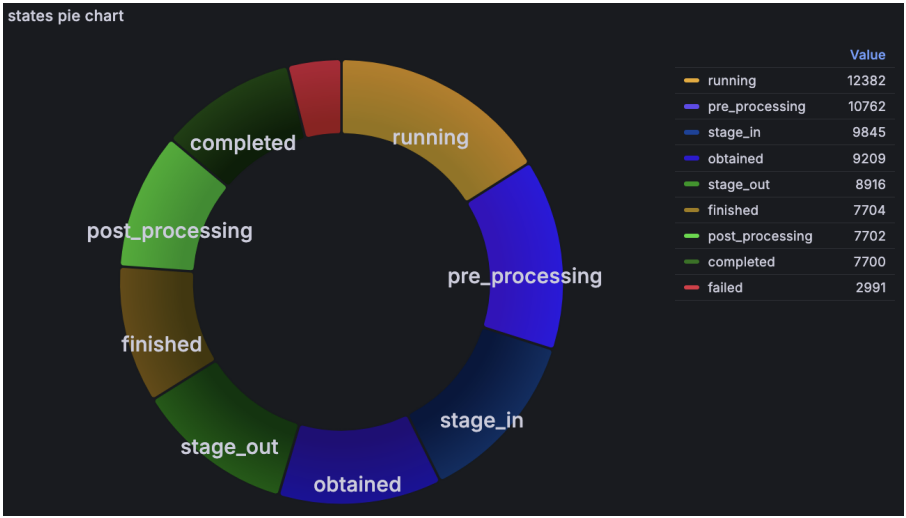
WMS Architecture



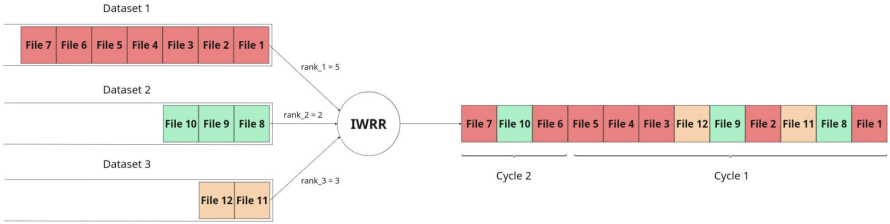
Pilots dashboards example



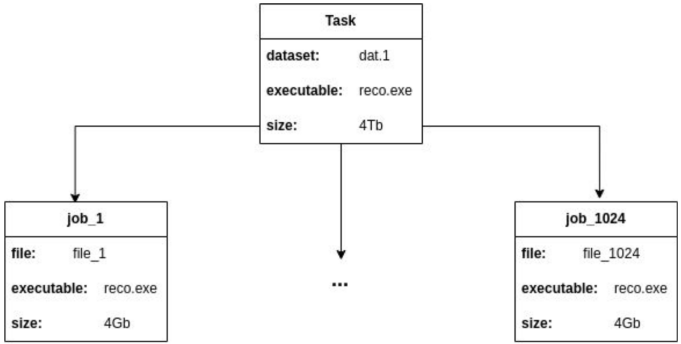
Jobs dashboards example: states pie chart



IWRR


Dataset 1
File 7
File 6
File 5
File 4
File 3
File 2
File 1
Dataset 2
File 10
File 9
File 8
Dataset 3
File 12
File 11
rank_1 = 5
rank_2 = 2
rank_3 = 3
IWRR
File 7
File 10
File 6
File 5
File 4
File 3
File 12
File 9
File 2
File 11
File 8
File 1
Cycle 2
Cycle 1

Task-jobs relationships



Data processing workflow

