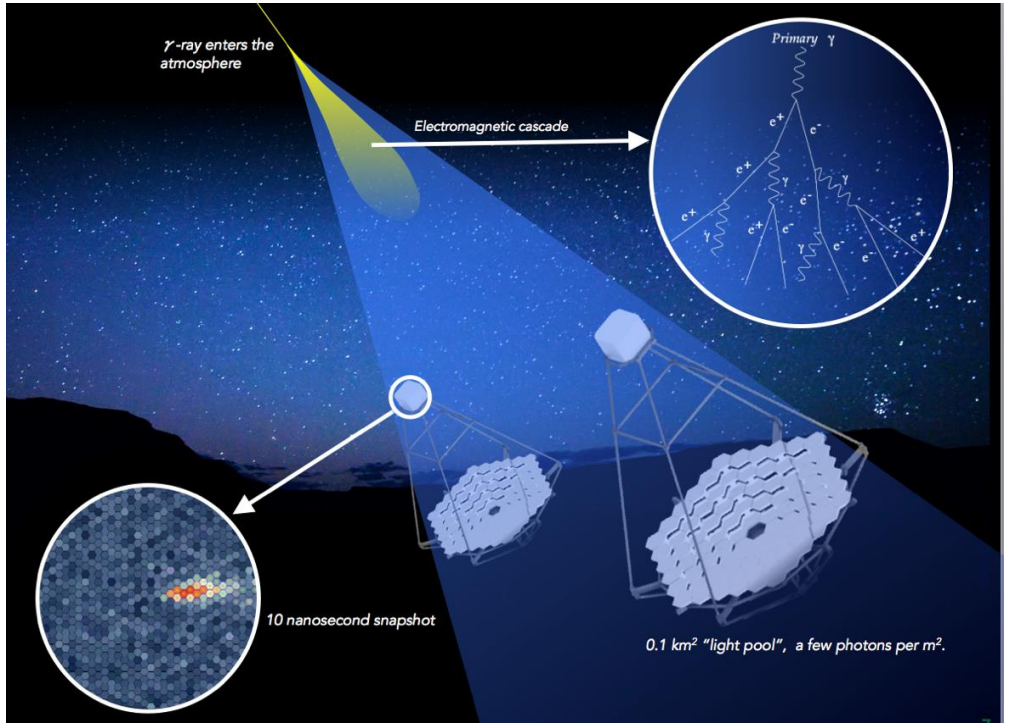
Data Processing and Preservation System (DPPS) Assembly Integration Verification

Volodymyr Savchenko, CTAO DPPS AIV

Volodymyr.savchenko@epfl.ch volodymyr.savchenko@cta-consortium.org

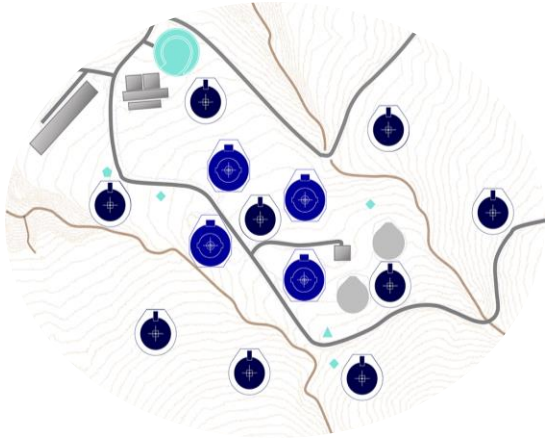
ADASS XXXV, Gorlitz, 11 November 2025

(Imaging Atmospheric) Cherenkov Telescope Array:

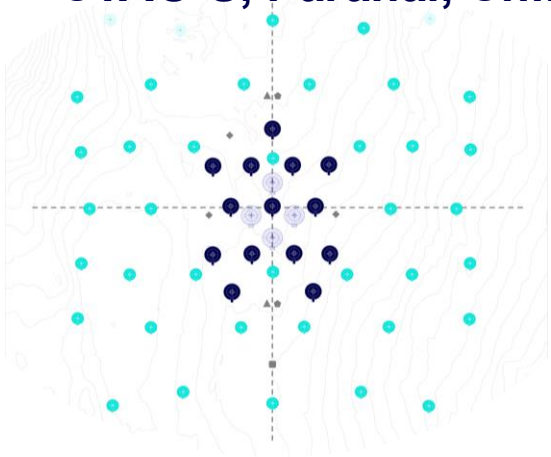


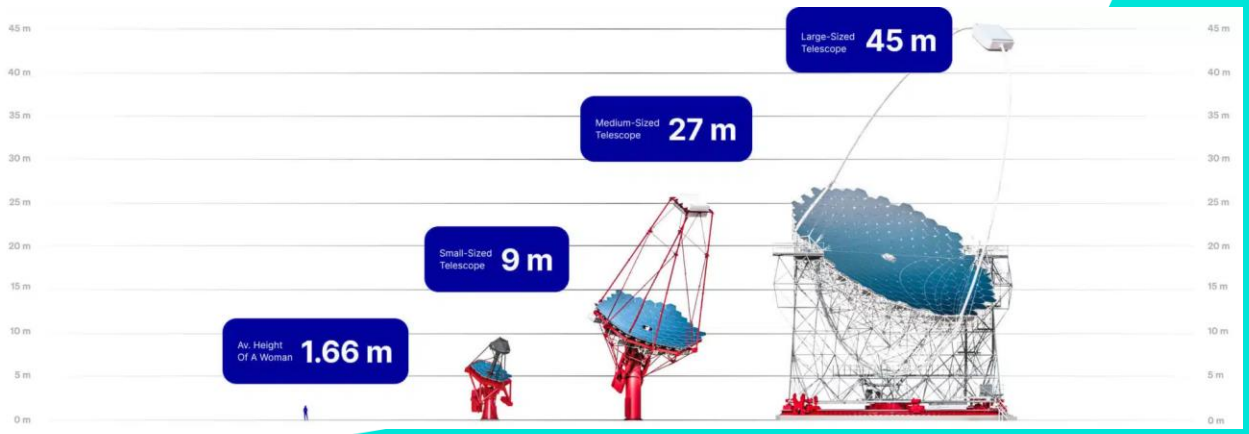
- **Large collection areas** (10^5 – 10^6 m^2)
- Major **sensitivity** improvement.
- Wide energy range **20 GeV – 300 TeV**
- **Good** energy (15 to 7%) and angular **resolution** (0.15 – 0.02 deg)
- **Fast reaction** to science alerts (30s to any point), **fast real-time analysis** and internal science alerts
- First IACT as Astronomical **Observatory**

CTAO-N, La Palma, Spain

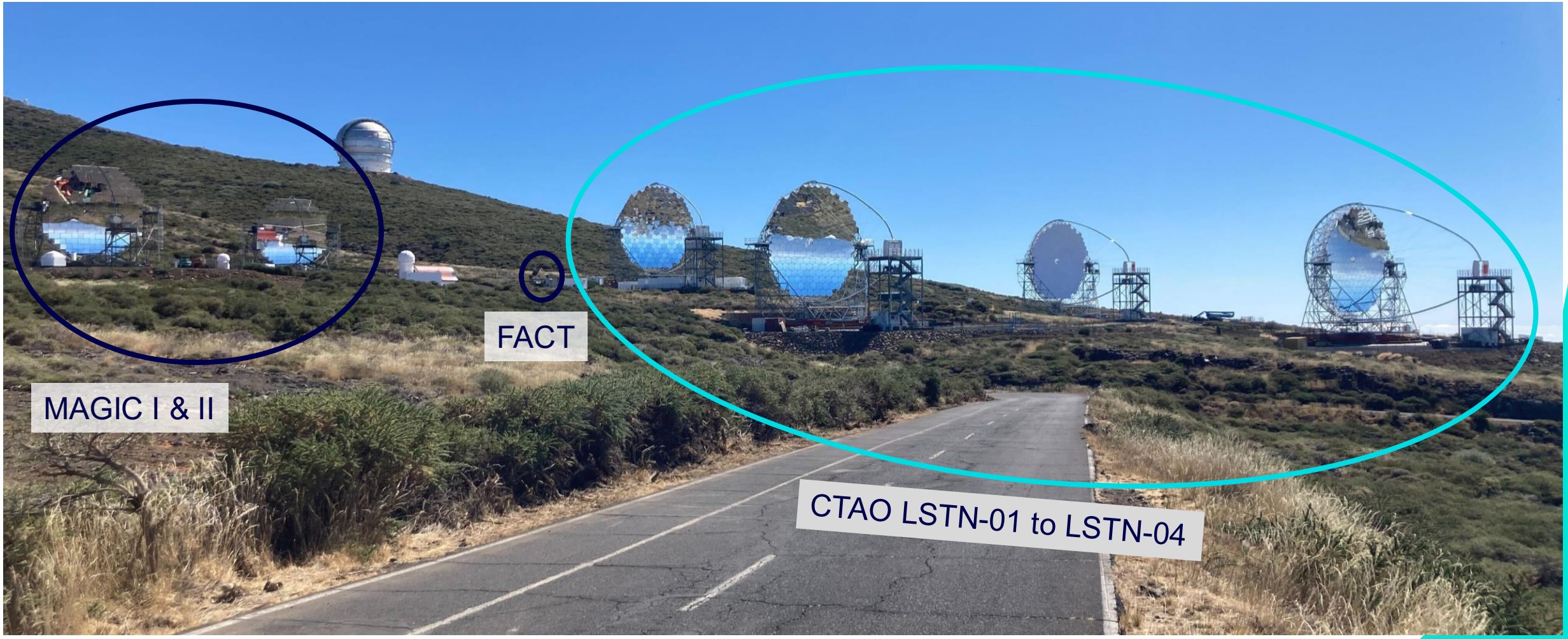


CTAO-S, Paranal, Chile



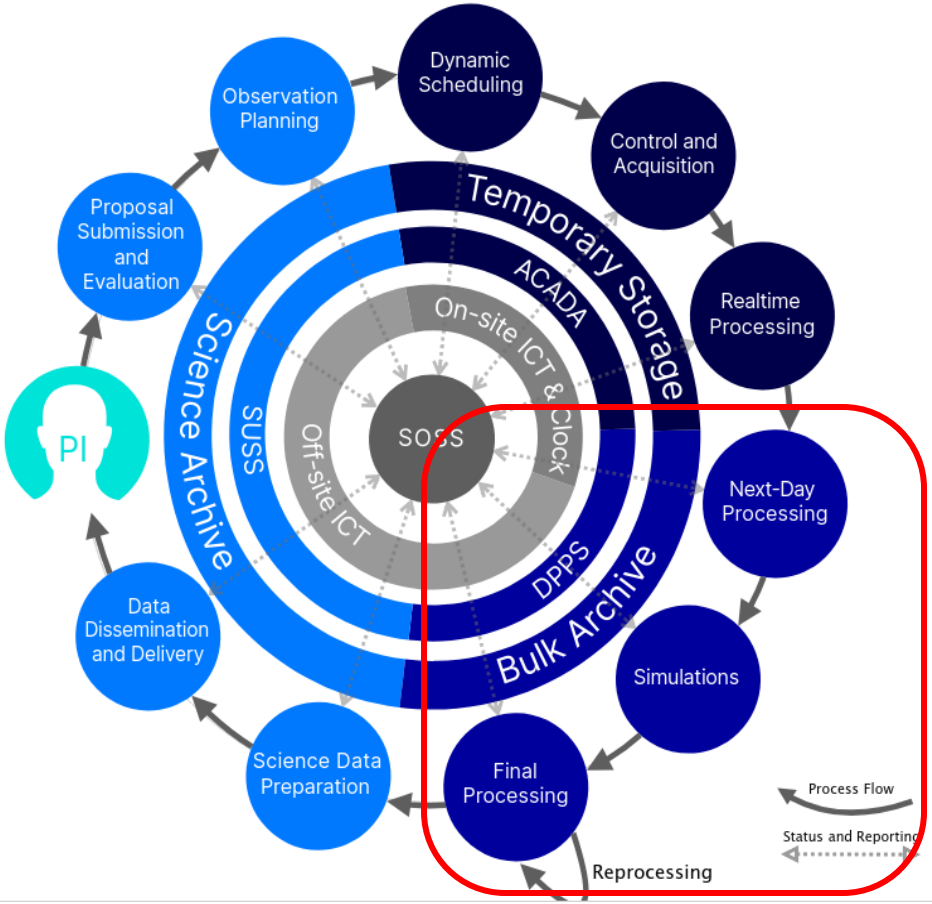


CTAO *now*

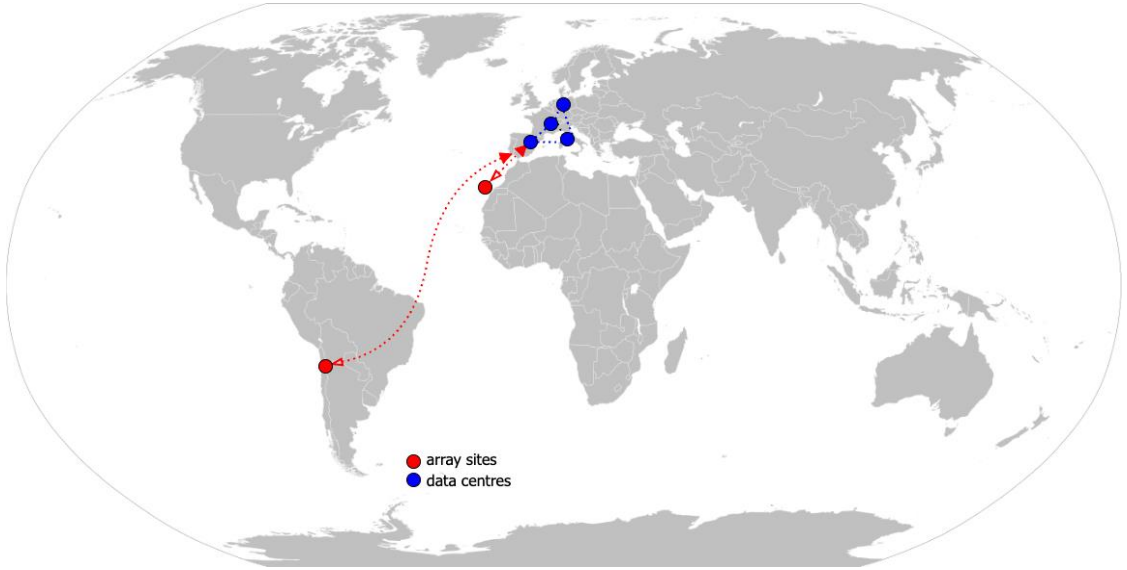


CTAO North Site, 2025-10-27, Koji Noda

DPPS: Data Processing and Preservation System



6 Data Centers (4 Off-Site and 2 On-Site)

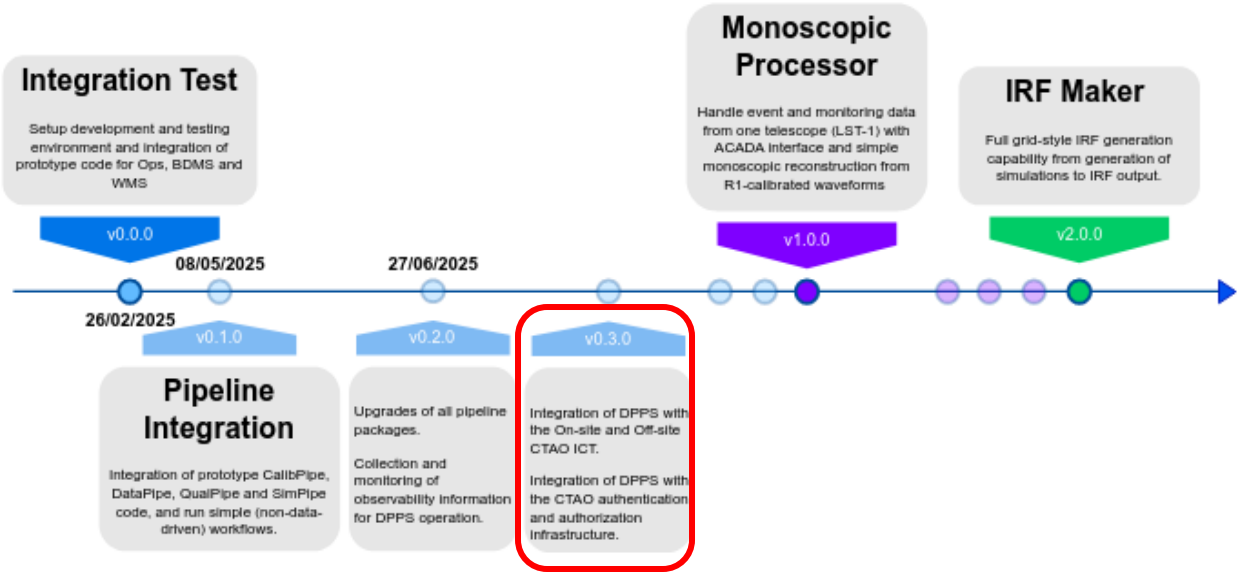


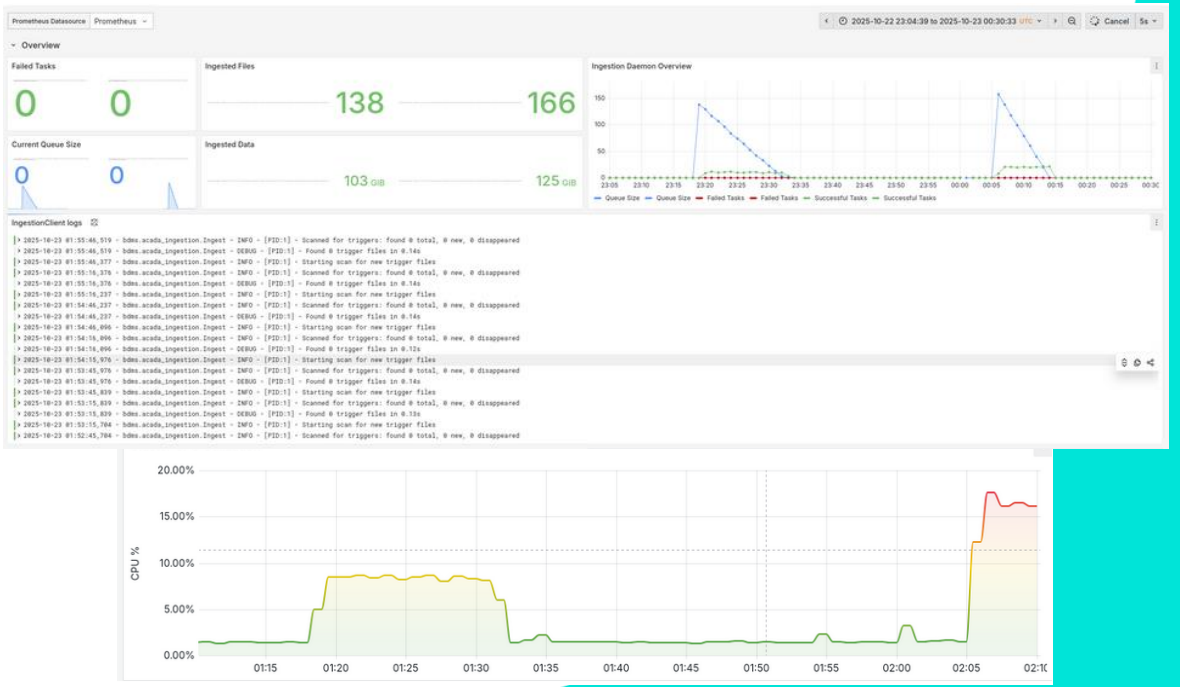
- **Distributed Large Storage** (sub-exabyte) & **Computing**
- **Robust** Storage (30 years, 2+ copies at different locations, tape to reduce **costs**)
- **Processing in-place** (minimize network, maximize CPUs)
- Multi-stage **data-driven workflows** complex science pipelines
- **Multiple pipeline passes** (fast next-day results, high-quality results within 1 month, years after for new software)

DPPS Releases and deployments: *now*

- DPPS minor release every **6 weeks**, patch as need, first release Feb 2025, 4 releases since then.

- **Pre-production test** on 2 **data centers** (1 Off-Site, 1 On-Site) First tests, just ~30 min of observations, transfer of ~200 files with a total volume of ~200 GB



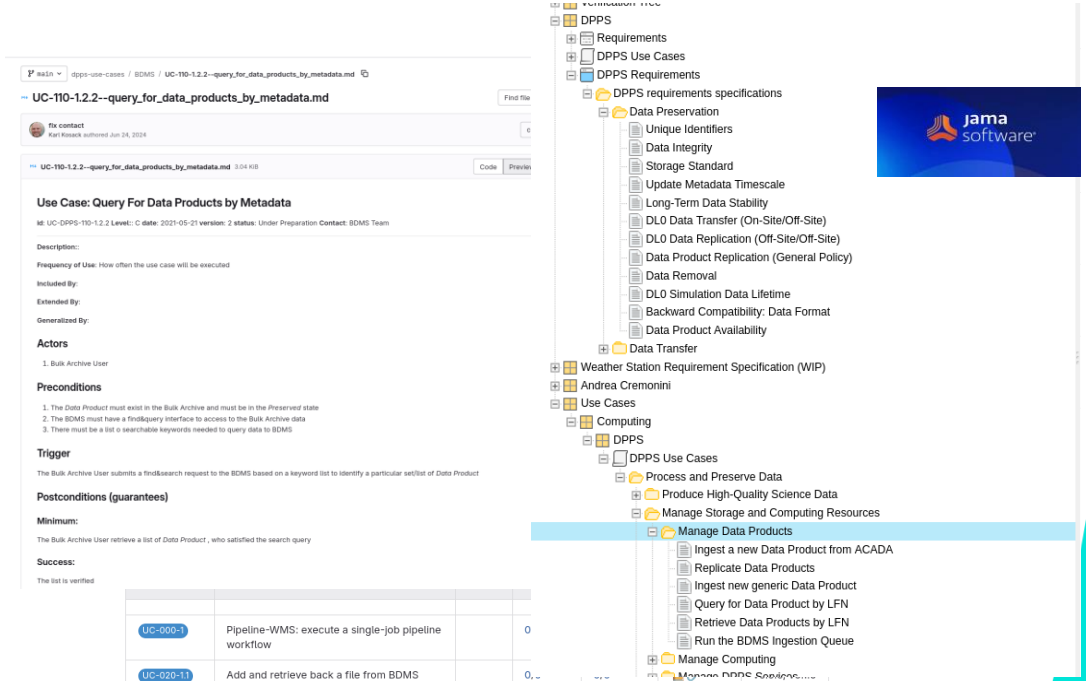


AIV incorporates stakeholder Requirements, Use Cases

Stakeholders require well-structured project documentation and **requirements**, so

- **Test Cases** are linked to **Use Cases** and **Requirements**
- Almost all **Test Cases** are expressed as automated integration tests producing **JUnit** reports annotated with pytest plugin, or any other way e.g. playwright
- **Manual** Demonstration **Tests Cases** remain possible, but *discouraged* (since they are harder to repeat)
- **Non-functional requirements**: quality and documentation are verified with **automated static analysis** (SonarQube, container image scanning)
- Verification by **Analysis** for dependencies

To understand how DPPS **Test Cases** are executed reliably, we need to address the **deployment strategy.**



UC-000-1	Pipeline-WMS: execute a single-job pipeline workflow		0			
UC-020-1.1	Add and retrieve back a file from BDMS using WMS		0/			
UC-100-2	Execute single-job workflow		0/0	0/0		v0.1.0
UC-110-1.1.1	Ingest a new Data Product from ACADA		0/0	0/0		v0.1.0
UC-110-1.1.3	Ingest new generic Data Product		0/0	0/0		v0.0.0
UC-110-1.1.4	Run the BDMS Ingestion Queue		1/0	1/0		v0.2.0, v0.3.0
UC-110-1.2.1	Query for Data Product by LFN		0/0	0/0		v0.0.0
UC-110-1.3.1	Retrieve Data Products by LFN		2/0	2/0		v0.0.0, v0.2.0
UC-110-1.6	Replicate Data Products		0/0	0/0		v0.1.0
UC-120-1.10	Request/retrieval of atmospheric reanalysis data for reference atmospheric profiles					
UC-120-1.2	Compute 12-MACOBAC					
UC-120-1.3	Selection of reference seasonal atmospheric model					

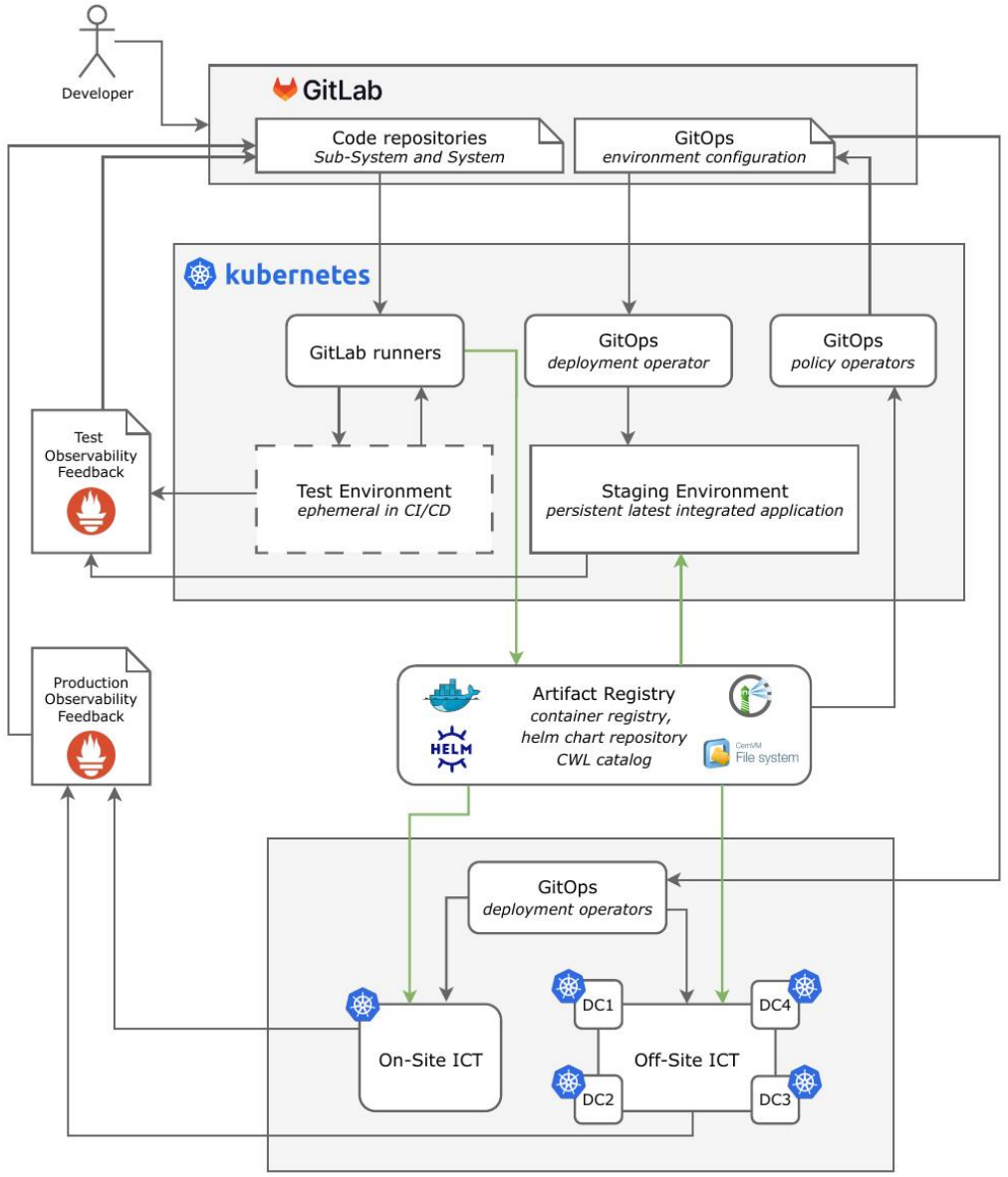
```
@pytest.mark.verifies_usecase("UC-110-1.1.1")
def test_add_onsite_replica_with_minio_fits_file(
    file_location: str,
    metadata_dict: dict,
    test_scope: str,
    tmp_path: Path,
    storage_mount_path,
    test_vo: str,
    caplog,
):
    """Test the add_onsite_replica method of IngestionClient

    ingestion_client = IngestionClient(
        data_path=storage_mount_path,
        rse=ONSITE_RSE,
        vo=test_vo,
        scope=test_scope,
    )
```

AIV and Deployment strategy

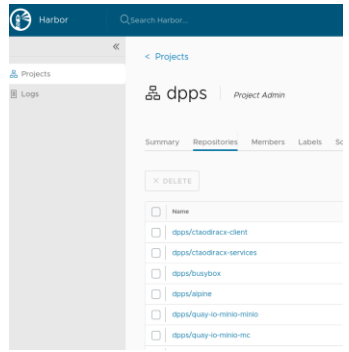
- **DevOps-driven reproducible** deployment built to mimic closely **production** using containers, kubernetes, helm charts.
  - **Development** setup (can be laptop or not) - volatile, pre-commit, but can be recreated as needed.
  - **Gitlab CI** pipelines – pure **reference test environment**
  - **Persistent GitOps**-defined:
    - **Staging** in the Test Cluster
    - **Pre-Prod** and **Prod** on Data Centers

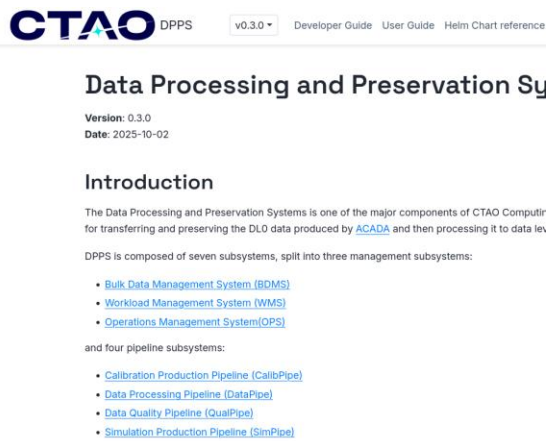
- **Observability** is compatible across all environments: learning to debug production issues from the start.
- Creating DPPS (>50 live containers) from scratch in a pristine environment, from build artifacts stored in **harbor**
  - Bootstrap takes **~5-15 min** depending on the host
  - We tested a **state-preserving** setup, reusing DPPS cluster, which takes **< 1min** to load from state snapshot

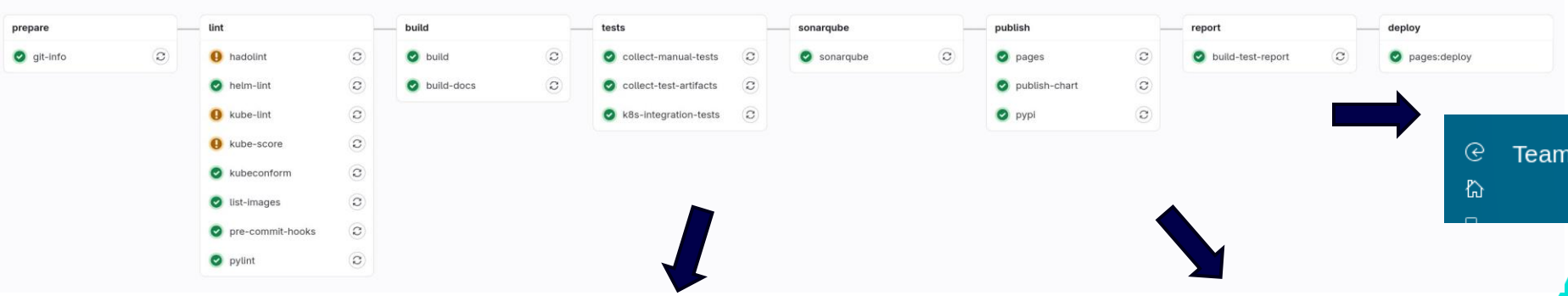


DPPS AIV CI pipelines, Toolkit

- Made to approximate **production** environment, allowing to repeat selected **verification** tests on clean live system.
- **Gitlab CI** pipelines are **reused** across all relevant **gitlab** projects.
- Built on python package, also for **local dev environment.**
- Collects **observability** in **production-**ready way
- Summarizes **dependencies** for sustainability analysis







3.1.5.1 QA metrics

Metric	Value	Threshold	Status
Reliability Rating	1	1	OK
Security Rating	1	1	OK
Maintainability Rating	1	1	OK
Coverage	100.0	80	OK
Duplicated Lines (%)	0.0	3	OK
Security Hotspots Reviewed	100.0	100	OK

3.1.5.2 Issues

SonarQube detected no issues in this project.

Use case summary: 1 partially completed, 41 completed of 42 total.

UC ID	Name	Rev.[1]	TCs	✓	Status
000-1	Pipeline-WMS: execute a single-job pipeline workflow		2	2	full
020-1.1	Add and retrieve back a file from BDMS using WMS		5	5	full
100-2	Execute single-job workflow		2	2	full
110-1.1.1	Ingest a new Data Product from ACADA		3	3	full
110-1.1.3	Ingest new generic Data Product		2	2	full
110-1.1.4	Run the BDMS Ingestion Queue	Rev.	2	2	full
110-1.2.1	Query for Data Product by LFN		1	1	full
110-1.3.1	Retrieve Data Products by LFN		1	1	full
110-1.6	Replicate Data Products		2	2	full
120-1.10	Request/retrieval of atmospheric reanalysis data for reference atmospheric profiles		7	7	full

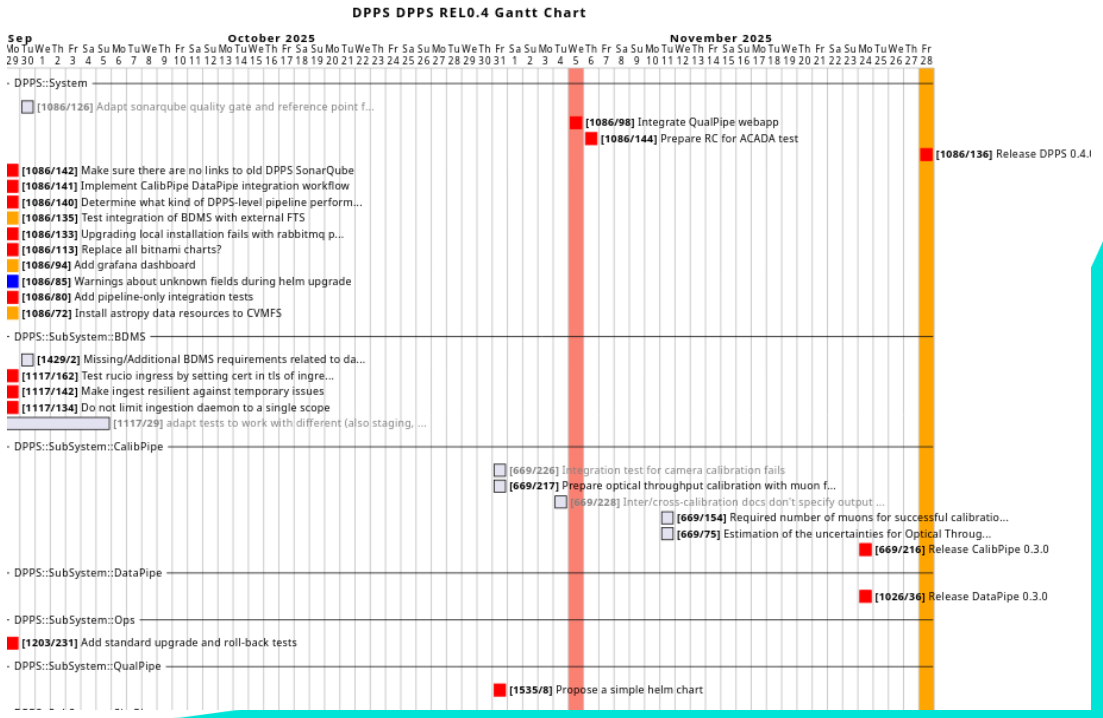
DPPS Component	Gitlab Project	Helm name	pyproject name	Version
BDMS	bdms	bdms	ctao-bdms-clients	v0.4.1
WMS	WMS	wms	ctao-wms-clients	v0.4.0
SimPipe	simpipe	simpipe	ctao-simpipe	v0.2.0
DataPipe	datapipe		ctao-datapipe	v0.2.1
CalibPipe	calibpipe		ctao-calibpipe	v0.2.0
QualPipe	qualpipe		ctao-qualpipe	v0.2.1
CVMFS	CVMFS Helm Chart	cvmfs	cvmfs-chart	v0.4.0
bdms-rucio-policy	bdms-rucio-policy		ctao-bdms-rucio-policy	v0.2.0
CTADIRAC	CTADIRAC		CTADIRAC	3.0.3
CVMFS	CVMFS Helm Chart	cvmfs	cvmfs-chart	v0.5.2
mdps	MDPs		molecularprofiles	v2.1.2
QualPipe-WebApp	qualpipe-webapp		ctao-qualpipe-webapp	v0.1.1

https://gitlab.cta-observatory.org/cta-computing/dpps/aiv/dpps-aiv-toolkit

AIV role in release cycle

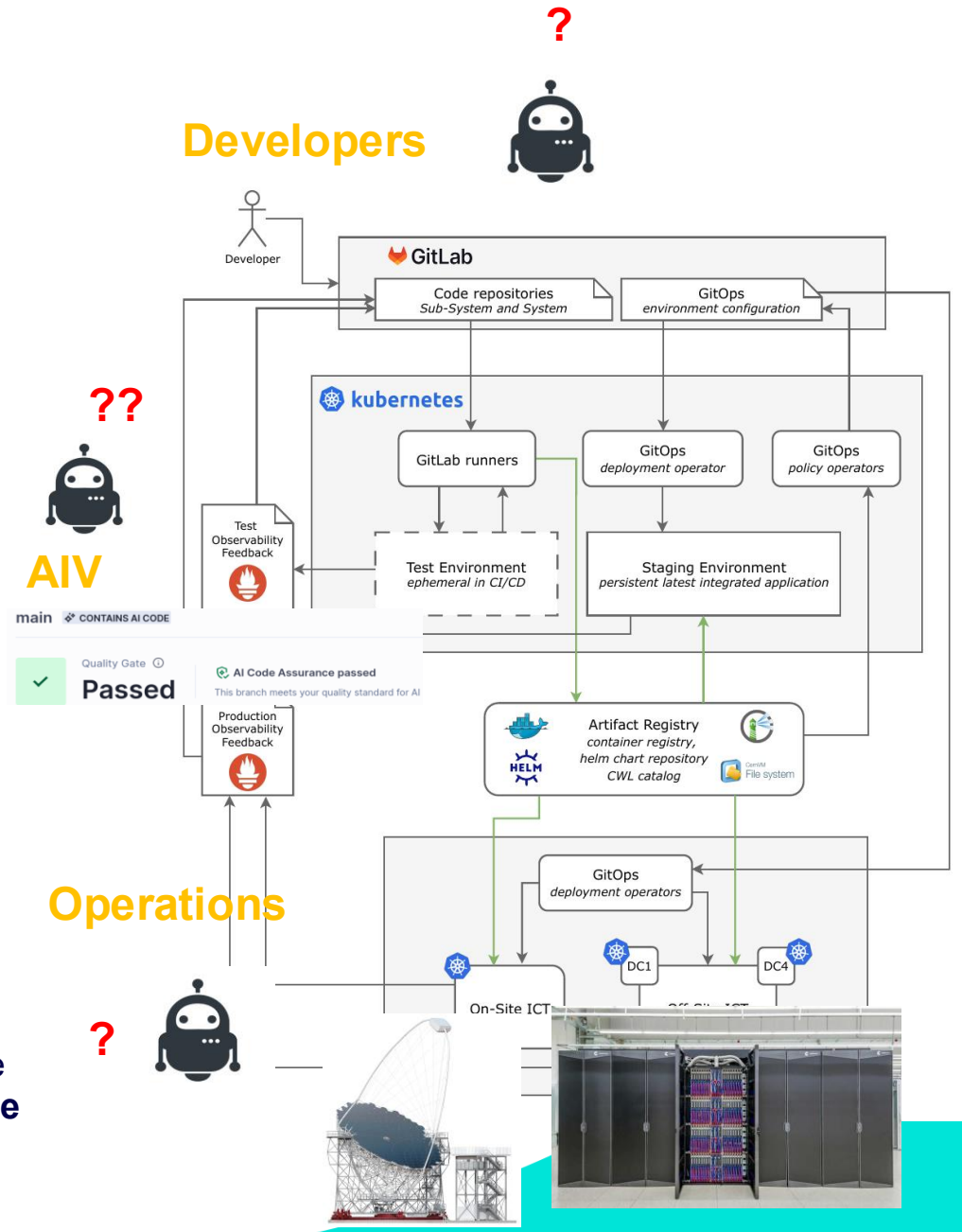
- AIV executes development, testing, and release cycle
  - **Trunk-based** development – all changes merged to main, no long-living side-branches.
  - **Test-driven** development: **start with requirements**, then **gitlab issue/tasks**, then (x-failing) **test**, only then **code**
  - **Short sprints** allowing frequent and **less costly integration exercises**
  - Fully **automated integration pipeline** can be run at **any time**, and is done regularly by **AIV Bot** (but now it's not run often to save resources)
- Support in **gitlab use** practices: customized **gantt**, **issues table**, manage labels

Name	Contact	Pipeline	Artifacts	UC	Impl	Test	Rel	All
DPPS	Volodymyr S.	pipeline running	Docs PDF	🔴 2,2 (0,0)	🟢 0,0 (0,0)	🟢 0,0 (0,3)	🔴 1,1 (0,0)	🟠 19,23 (2,5)
BDMS	Syed A.	pipeline passed	Docs PDF	🟢 0,0 (0,0)	🔴 1,2 (0,0)	🟢 0,0 (0,0)	🟢 0,0 (0,0)	🟠 16,25 (3,6)
WMS	Natthan P.	pipeline passed	Docs PDF	🟢 0,0 (0,0)	🔴 2,2 (0,0)	🟢 0,0 (0,0)	🔴 2,2 (0,0)	🔴 9,9 (0,0)
SimPipe	Gernot M.	pipeline passed	Docs PDF	🟢 0,0 (0,0)	🟢 0,0 (0,0)	🟢 0,0 (0,0)	🔴 1,1 (0,0)	🔴 2,3 (0,0)
DataPipe	Karl K.	pipeline passed	Docs PDF	🟢 0,0 (0,0)	🟢 0,0 (0,0)	🟢 0,0 (0,0)	🔴 1,1 (0,0)	🔴 4,4 (0,0)
CalibPipe	Mykhailo D.	pipeline passed	Docs PDF	🟠 0,0 (2,2)	🔴 4,4 (0,0)	🟢 0,0 (0,0)	🔴 1,1 (0,0)	🟠 8,9 (2,3)
QualPipe	Mykhailo D.	pipeline passed	Docs PDF	🟢 0,0 (0,0)	🟢 0,0 (0,1)	🟢 0,0 (0,0)	🟢 0,0 (0,0)	🟢 0,0 (0,1)
Ops	Volodymyr S.		Docs PDF	🟢 0,0 (0,0)	🟢 0,0 (0,0)	🟢 0,0 (0,0)	🟢 0,0 (0,0)	🔴 3,4 (0,0)
DPPS all	Volodymyr S.		Docs PDF	🟠 6,6 (4,4)	🟠 14,16 (12,17)	🟢 0,1 (0,3)	🔴 6,12 (0,0)	🟠 169,206 (39,54)



AI in the CTAO DPPS AIV

- CTAO has **no** AI strategy, AI agency not recognized, people represent all tools
- In CTAO DPPS Computing:
  - Big topic in CERN which built base technologies we use, MCP is added in rucio
  - Key DPPS projects use **github** with its AI features, but own **gitlab** misses them
  - Autocomplete, agents, chatbots, even vibe-coding is daily work of **many CTAO DPPS developers**, especially docs and manuals.
- In **operations** has potential threat detection, security scanning, built-in DPPS observability stack
- AIV Verification
  - Basic goal of development can be seen as **translation of requirement to code** but doing that fully AI appears too wishful currently.
  - Creating **tests** based on **codebase and requirements** works better
  - **AIV** operates **Bot** managing gitlab, jama, making code contributions, with experimental interface to Claude and Apertos
- AIV Quality Assurance
  - Core role of the AIV is to ensure transparency, compliance, safety, minimize **technical** and **cognitive** debt, so it acts as **control on AI code. SonarQube** actively helps in this
  - QA starts to face **easy to generate** but **hard to review** AI slop



Summary so far, plans

- **CTAO is under construction** to be done in **2030**, is deployed incrementally so is getting **real already now**!
- DPPS Released 4 minor versions (0.0, 0.1, 0.2, 0.3) in 6 months, covering first broad set of key use cases
- CTAO DPPS AIV embraces, continuous automated verification and delivery, **DevOps**, **human processes** are **displaced** but **emphasized**.
- At the same time, producing **high-quality project documentation** (release reports, verification claims) demonstrating compliance to **stakeholder requirements.**
- As requirement-to-code conversion includes a lot of human work, AIV emphasizes more adversarial aspect to **AI**, preventing technical and cognitive debt, more GenAI means more AIV than development. Safety-critical systems of CTAO need a lot of oversight over AI contribution
- CTAO grows by months not years, stay tuned for more updates!