IGWN-Virgo Computing Model

Franco Carbognani – EGO

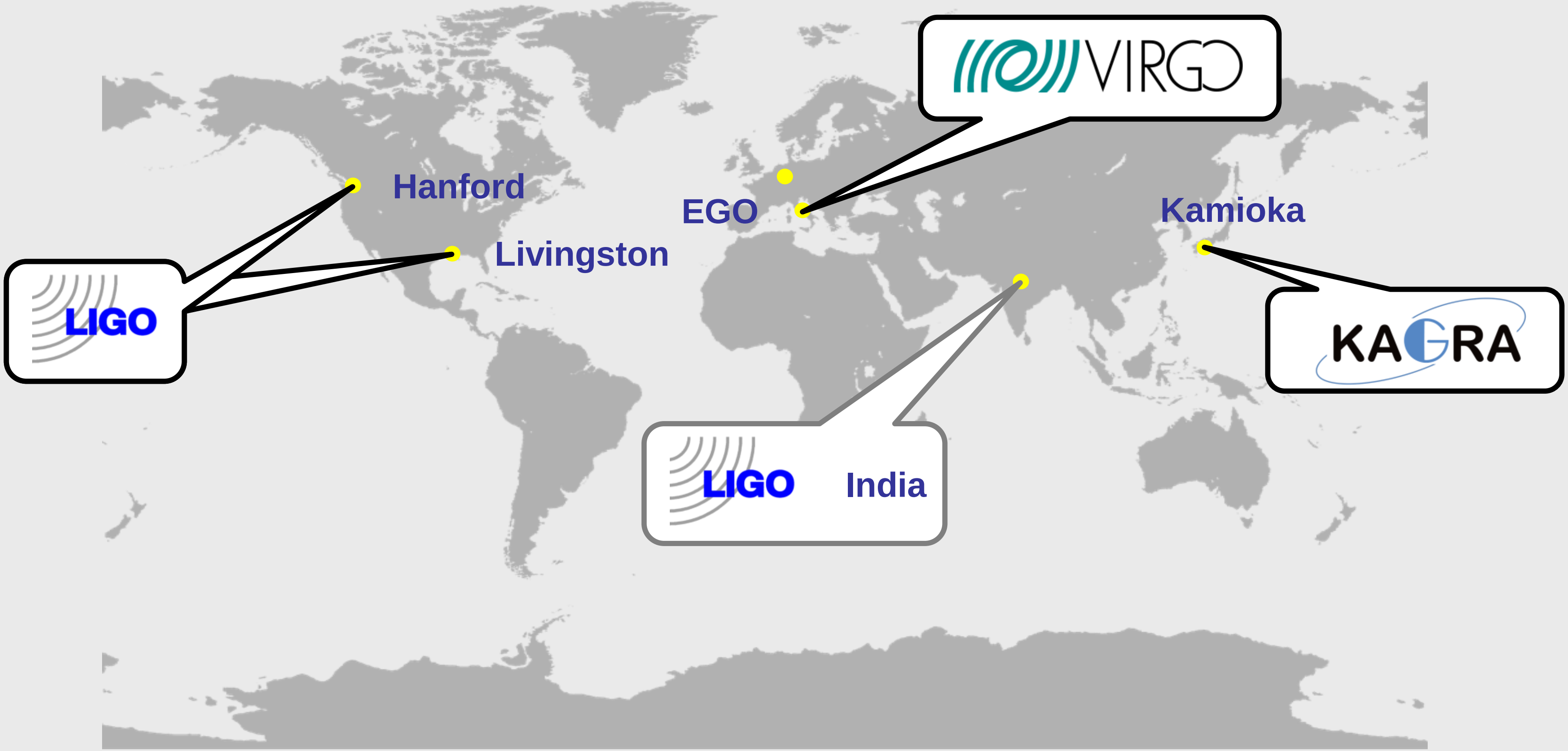
on behalf of the LIGO, Virgo and KAGRA collaborations

ET-EiB workshop| 10 May 2023

Overview

- A Worldwide Network: The International Gravitational Waves observatory Network (IGWN)
- Data and data flow schema
- Low-latency Analysis timescale
- What is needed
- The common solutions and the common tools
- Conclusions

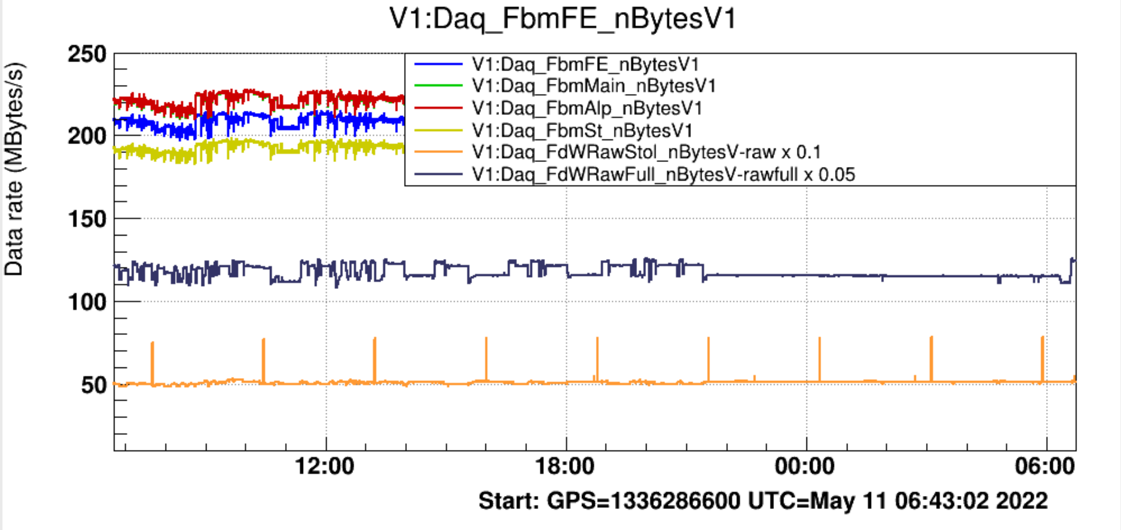
A Worldwide Network



- International Gravitational Waves observatory Network (IGWN)
  - A coordination effort aimed at jointly discussing the computing policy, management, and architecture issues of LIGO, Virgo, and KAGRA.
  - igwn.org will host common Virgo/LSC services, and KAGRA is joining
  - Migration planning for common services (GitLab, software tarballs archive, etc.) is unfolding
  - The effort comprises all computing domains (online, offline, LL), with different levels of engagement.
- Plans for migration from custom/legacy tools to common, mainstream tools are being implemented
  - Data management and transfer (CVMFS/ OSDF Cache, Rucio and Kafka)
  - Software management (GitLab, CMake/Meson, Conda, CVMFS)
  - The tools (and the general architecture) are chosen to be consistent with many large scale computing projects such as WLCG

Raw Data, ~1.5 PB/yr:

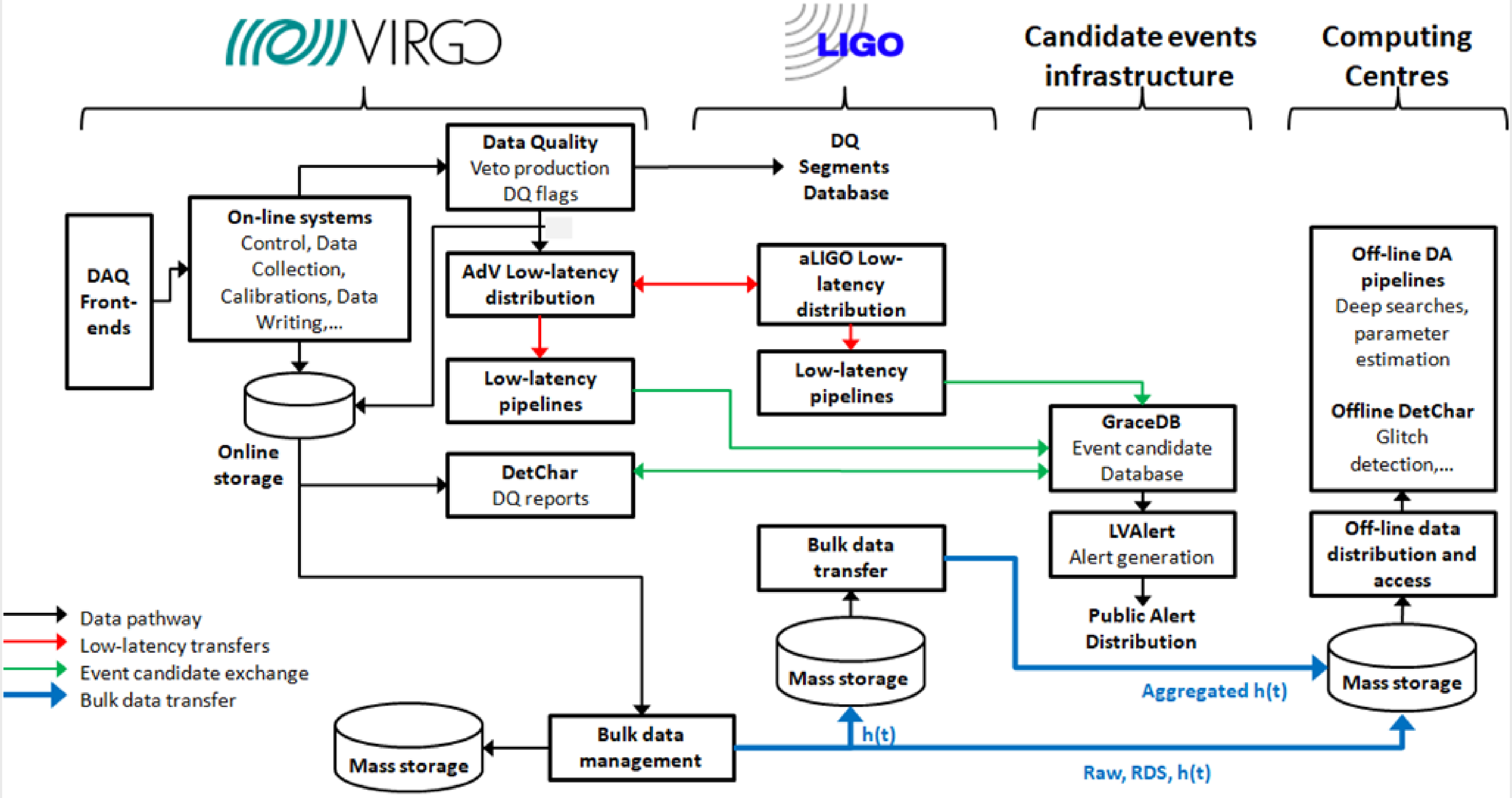
- **Full Bandwidth Raw**, not exported
- **Raw Data:** downsampled, include all auxiliary channels and calibrated/uncalibrated h(t) (~ 50MB/sec -> 4 TB/day -> 1.5 PB/yr), exported to Virgo Tier1 CC (CNAF, CC-IN2P3) for custodial storage during scientific runs
- **A few levels of reduced data sets** for various uses



Data for physics, ~ 5TB/yr/detector:

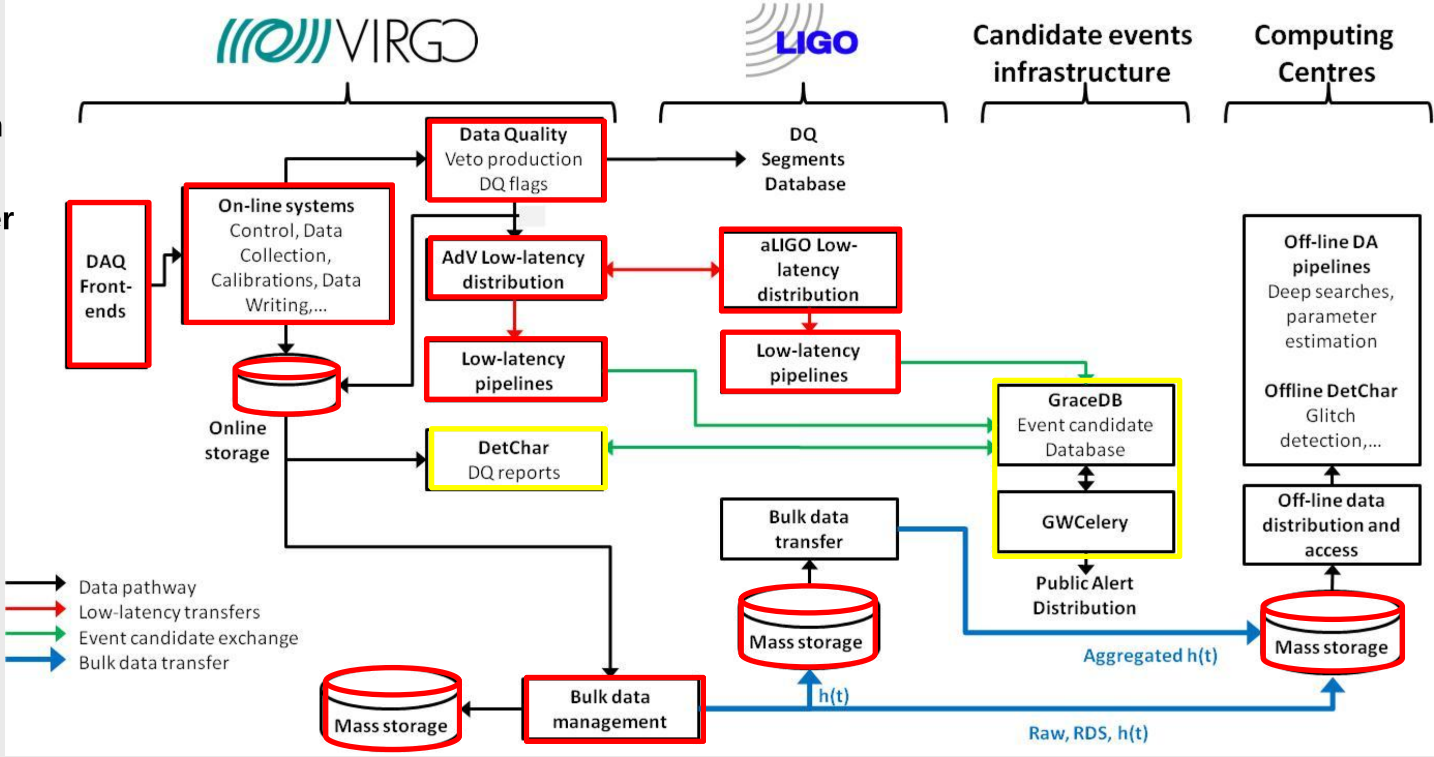
- **Virgo h(t): calibrated «strain» data**
  - sampled at 10 KHz, stored as ~1kSec frame files: aggregated h(t)
  - Includes state vector (data quality flags, vetoes,…)
- **LIGO (and KAGRA) h(t)**
  - Transferred online to EGO for low-latency searches and made available to IGWN CCs for offline analysis

The Data Flow Schema

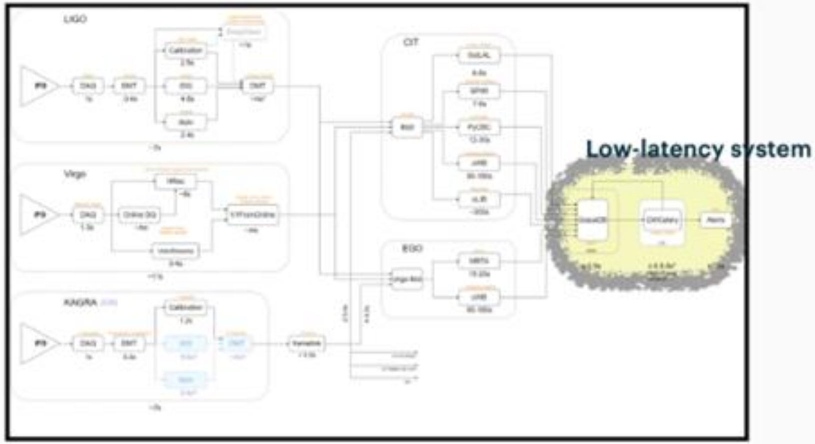


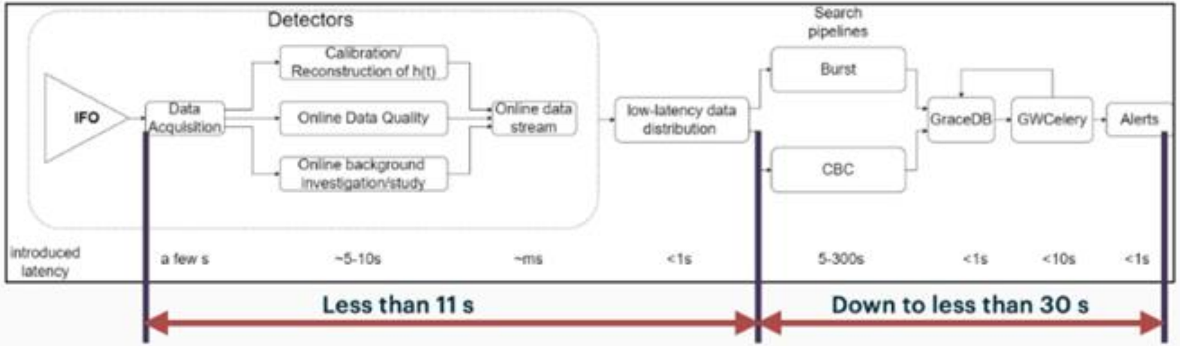
Data Flow: The O3 case

1. The signal arrives
2. Data composed into frames
3. Calibration of the data
4. Veto, DQ flags production
5. h(t) transfer
6. Low-latency matched-filter pipelines
7. Upload to GraceDB
8. Data written into on-line storage
9. Low-latency data quality
10. Low-latency sky localization
11. GCN Circular sent out
12. Data written into Cascina Mass Storage
13. Data transfer toward CCs



- We are running extensive tests (already started - up to engineering runs) from data acquisitions (synthetic) to alert generation, and we are monitoring latency.
- We have the signal ready to be analyzed online in less than 11 seconds from the arrival of the (GW) signal at the detectors.
- That makes pre-merger alerts possible (with negative latency) and the first preliminary alerts in less than a minute.
- The study will also allow us to test the effectiveness of the online pipeline to detect and assess the properties of the signal.





End-to-end latency study group (https://wiki.ligo.org/Operations/O4EndToEndLatency)

roberto.depietri@unipr.it

The Challenge

- The geographical separation of the detectors and the short timescale involved (e2e latency from signal detection to alerts generation in the order of 30 sec) imply the creation of a common distributed cyber infrastructure which must guarantee:
  - Adequate **storage and computing resources** for detector characterization, low-latency searches and alerts generation
  - **Low-latency data distribution** among the different observatories and computing clusters for low-latency searches
  - An ubiquitous and uniform **running environment** on dedicated resources and heterogeneous infrastructures
  - An homogeneous model for **offline data distribution and access**
  - A robust **support for development and operation**

The Common Solutions

- Storage and computing resources
  - Present: dedicated or highly-prioritized resources provided by observatory computing centers
  - Future: High Availability (HA) deployment on top of Kubernetes
- Low-latency data distribution
  - Kafka
- Running environment
  - IGWN environments: Conda + CVMFS
- Bulk data transfer
  - Rucio
- Offline data distribution and access
  - StashCache + CVMFS
- A robust support for development and operation
  - GitLab

Storage and Computing Resources

Low-latency storage and computing mainly provided by observatory computing centers

- Low-latency alert infrastructure runs on dedicated resources with high priority to burst out into pool
- Search pipelines run on dedicated or highly-prioritised resources in an HTCondor-managed resource pool
- Fast, direct access to small data files / shared memory
- GraceDB production instance currently deployed in High Availability (HA) on AWS
- Alternative HA deployment via Kubernetes is being tested on INFN-Cloud at CNAF

High Availability (HA) deployment

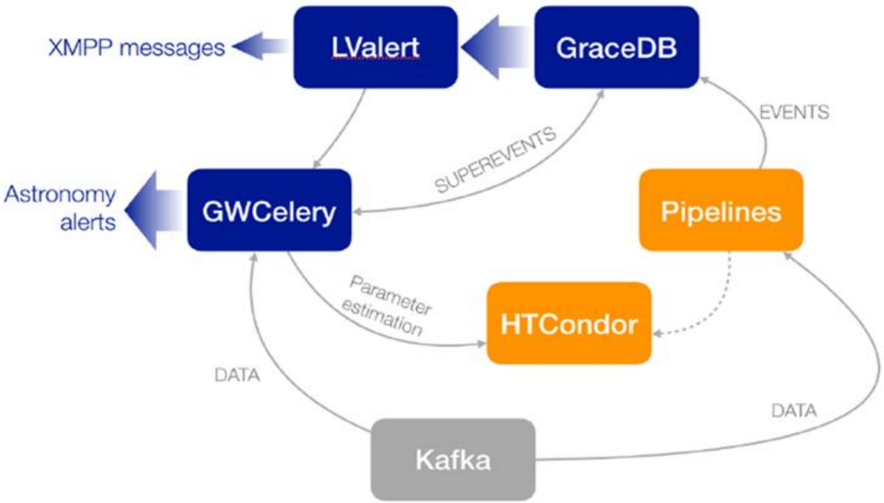
A research project is ongoing on a Kubernetes-based HA deployment for the alerts generation components (GraceDB, GWCelery, LVAlert) into the CNAF Computing Center Cloud.

- To exploit support from CCs staff and achieve cloud provider independence
- Provide support for both high-availability deployment using K8S and on-demand HTCondor cluster using cloud-native tools
- A preliminary step to provide infrastructure to run low-latency searches off-site
- Not (yet) targeting a production deployment for O4 (baseline remain O3 deployment), possibly the test/playgroud tiers
- Performance and stress tests in progress

VIRGO Kubernetes Cluster

Deployment plan within EGI-ACE

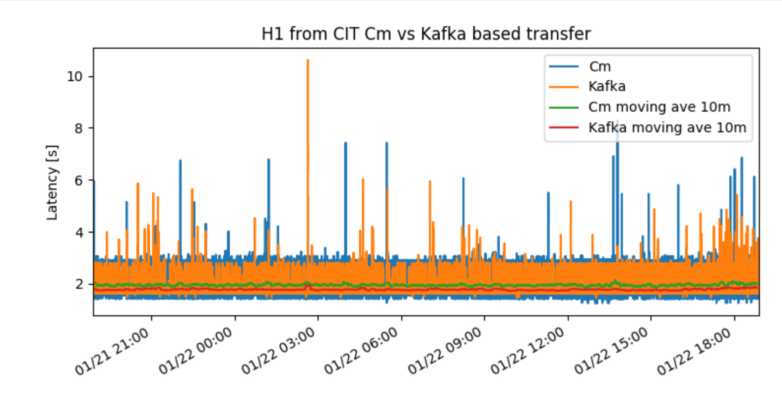
- SERVICES: require high-availability deployment
- COMPUTING: requires sizable amount of resources



- **High availability Kubernetes Cluster** managed by the INDIGO PaaS Orchestrator to deploy the relevant services
- **HTCondor cluster on-demand (DoDAS)** with access to streaming data to run the analysis pipelines with deterministic queue latency

Sara.Vallero@to.infn.it

- Kafka is a modern high throughput stream processing software
  - Built in redundancy
    - Can survive if stream stops or Kafka broker goes down
    - Replication so no data loss from service downs
  - Highly scalable and reconfigurable
    - Can easily add additional observatories data
  - Tests has been in progress in Virgo for months (also comparing to legacy solution: Cm)



Running Environment: IGWN Conda

Sustainable software development and distribution = **CVMFS** + **Conda**

Conda environments hosted in CVMFS file system available at all IGWN sites and CC

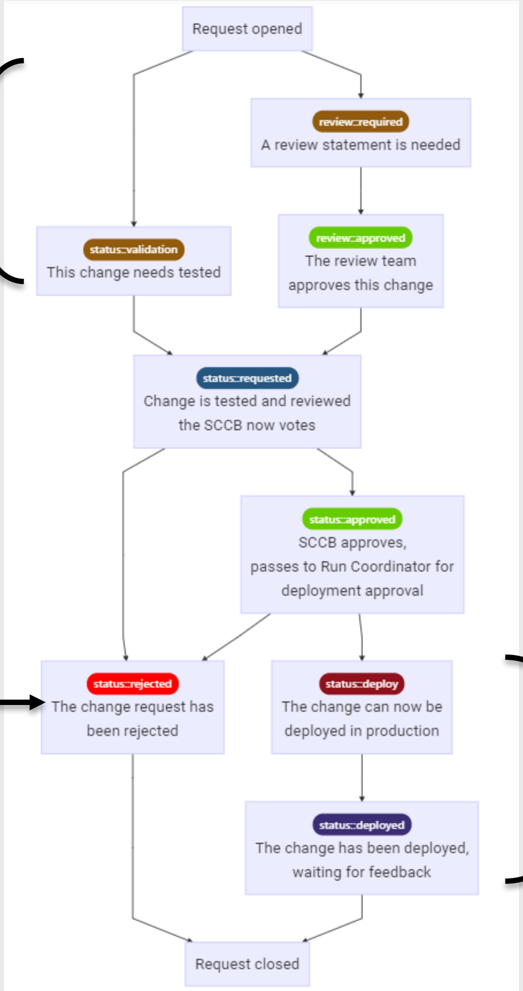
Running Environment: IGWN Conda

- IGWN Conda Distribution provides pre-built, automatically-distributed environments of approved software
- available via CVMFS on any machine (no authentication required)
- can be replicated on any workstation
- Provide effective OS independence
- Leverage on CMake/Meson for software build
- Provide a very effective solution for otherwise unmaintainable number of custom software builds
- On large part of IGWN sites (including Virgo Cascina) the IGWN environment can be activated manually or is by default at user login

Software deployment into IGWN Conda Distribution

Includes:

- **Packaging**
- **Testing**
- **Scientific review (with the package developer)**



Software deployment is controlled via change control board (SCCB)

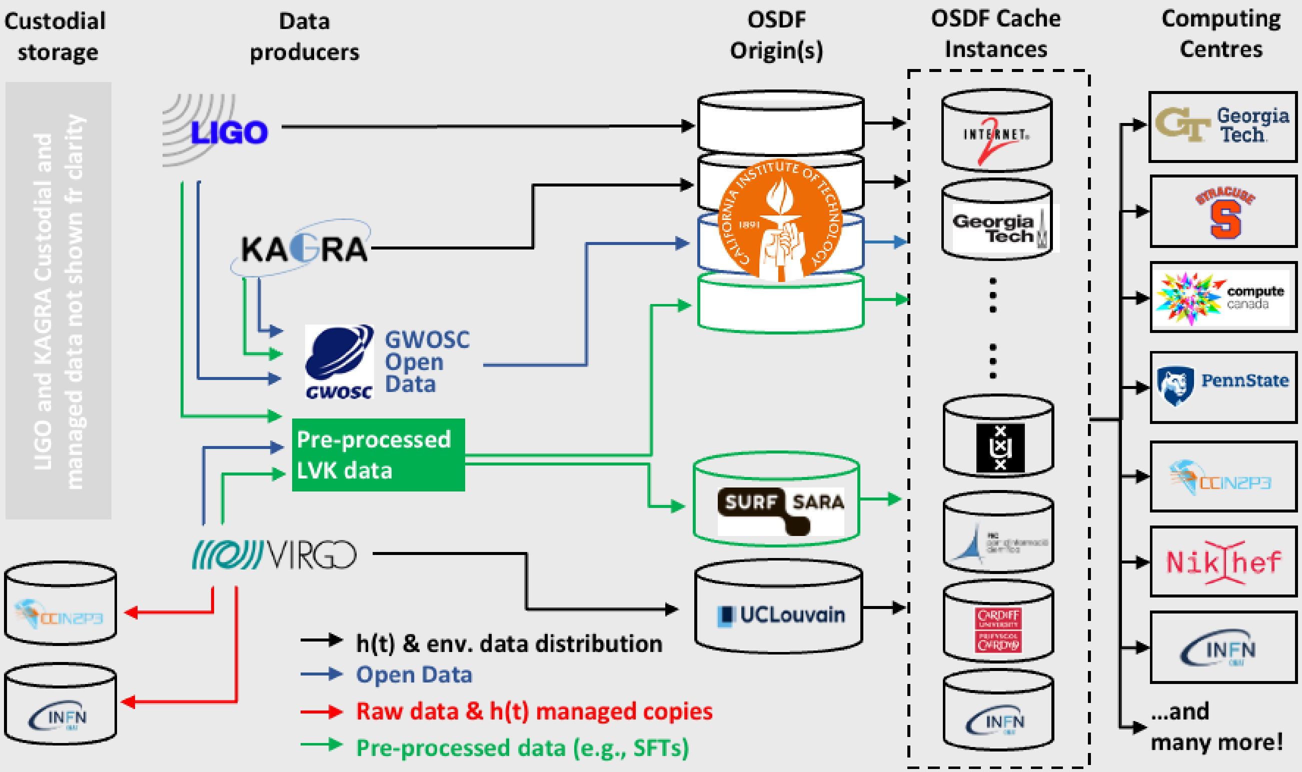
Software must be approved by SCCB and the relevant scientific review committee before being used for production analysis

Very rarely happens. Typically, only done if the person requesting the package no longer wants it in the IGWN Conda Dist.

This stage is done by a Run coordinator

Offline data distribution and access

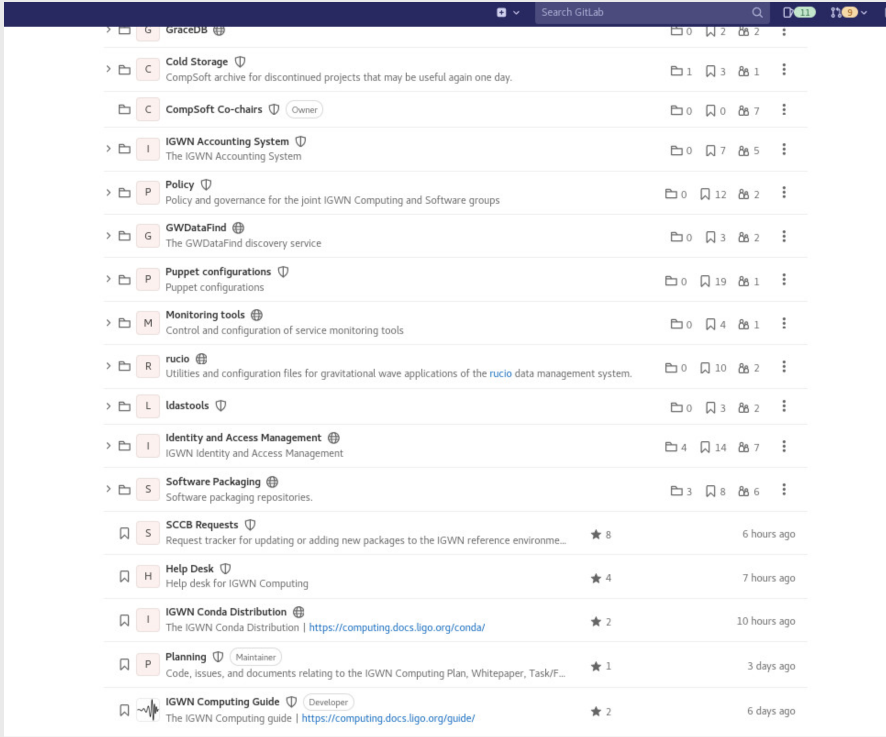
- data transfer via Rucio (still using a legacy solution for raw data)
- Main access point will be provided via OSDF Cache+CVMFS



Development and Operation

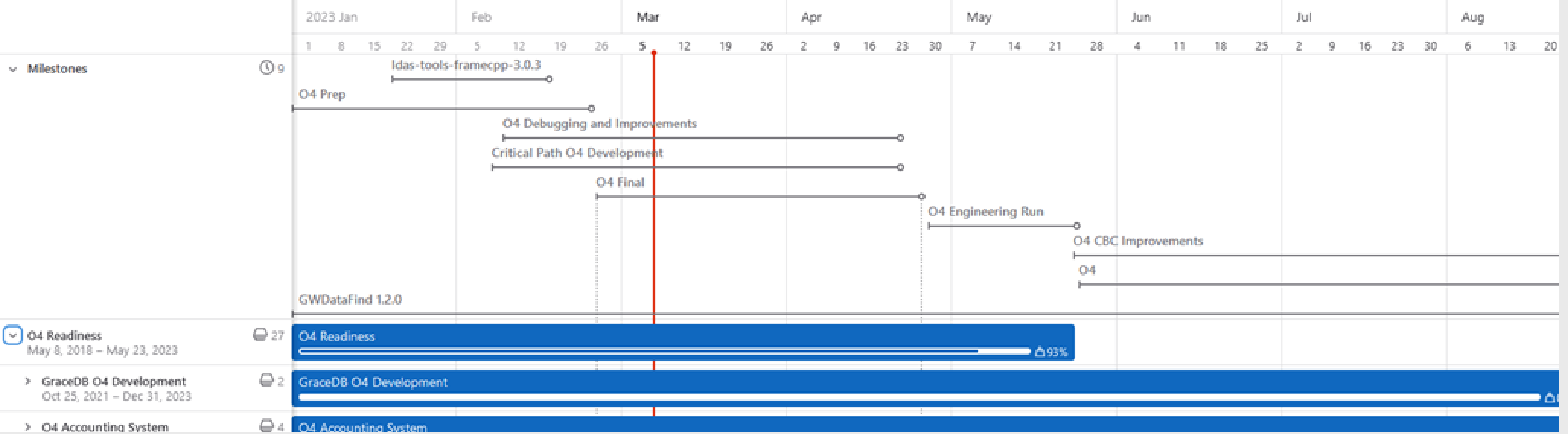
LIGO, Virgo and KAGRA Computing & Software teams increasingly working together using GitLab centered DevOps practices. IGWN Gitlab provides/supports:

- Code repository for IGWN Detectors Control, Data Analysis and Low-latency software
  - Enables easy collaboration on software
- Continuous integration (CI) capabilities to enable the automation of building, testing and deployment of code.
- Software Configuration Control Board (SCCB) activities
- IGWN Computing HelpDesk (via tickets)



Development and Operation

- Conda software distribution management
- Online IGWN Computing Guide (via GitLab Pages)
- IGWN Computing Planning (via the growing GitLab support for projects management)
  - Issues
  - Epics



Conclusions

- IGWN Computing Model, envisioning the transition from custom, incompatible LIGO, Virgo and KAGRA-specific infrastructures to a common multi-collaboration infrastructure, based on mainstream tools, well ongoing has payed already many dividends (e.g. Gitlab, Conda, IGWN grid)
- Most of the planned upgrades are targeted for the O4 Science Run and being exercised during ongoing Enrineering Run (ER15)
- Further steps on the implementation of the identified common solutions implementation will be made in the O5 time frame.