LIGO

Digital Infrastructure for Next Generation Gravitational Wave Detectors

XGCD

July 28, 2025

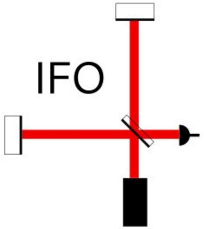
Jameson Rollins
LIGO Caltech

From LIGO to Next Gen

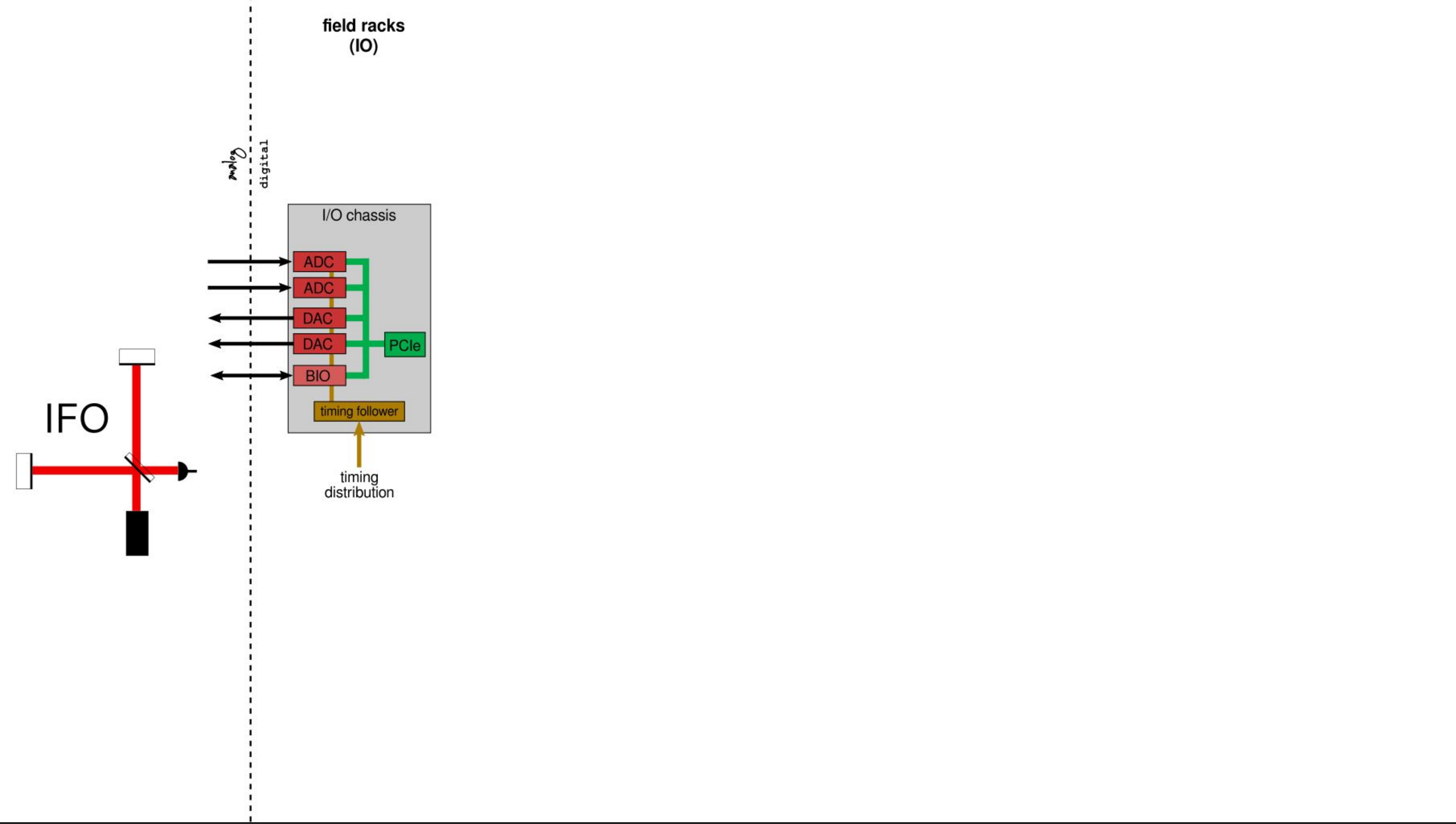
The LIGO CDS group sees a clear path to evolve our existing digital control, data acquisition, and data distribution systems to meet the needs of next generation facilities, and the research labs that support them.

And in fact the LIGO CDS systems continue to be under active development to improve, add new features, support new workflows and modes of operation, and to keep up with industry evolution and best practices.

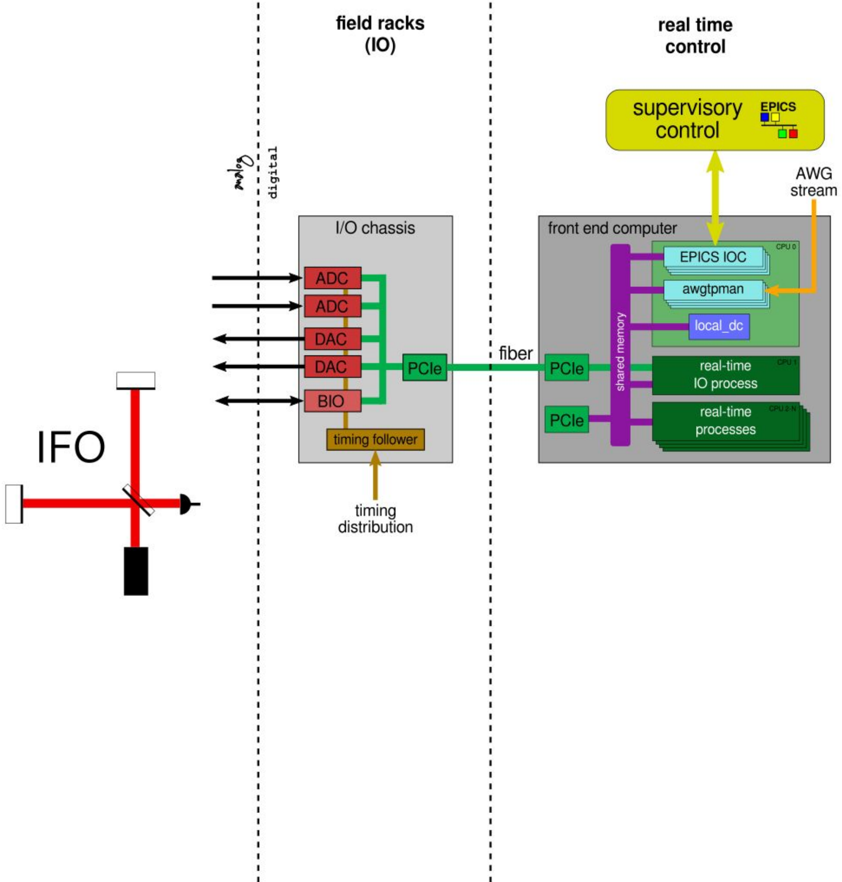
LIGO RT control and data architecture



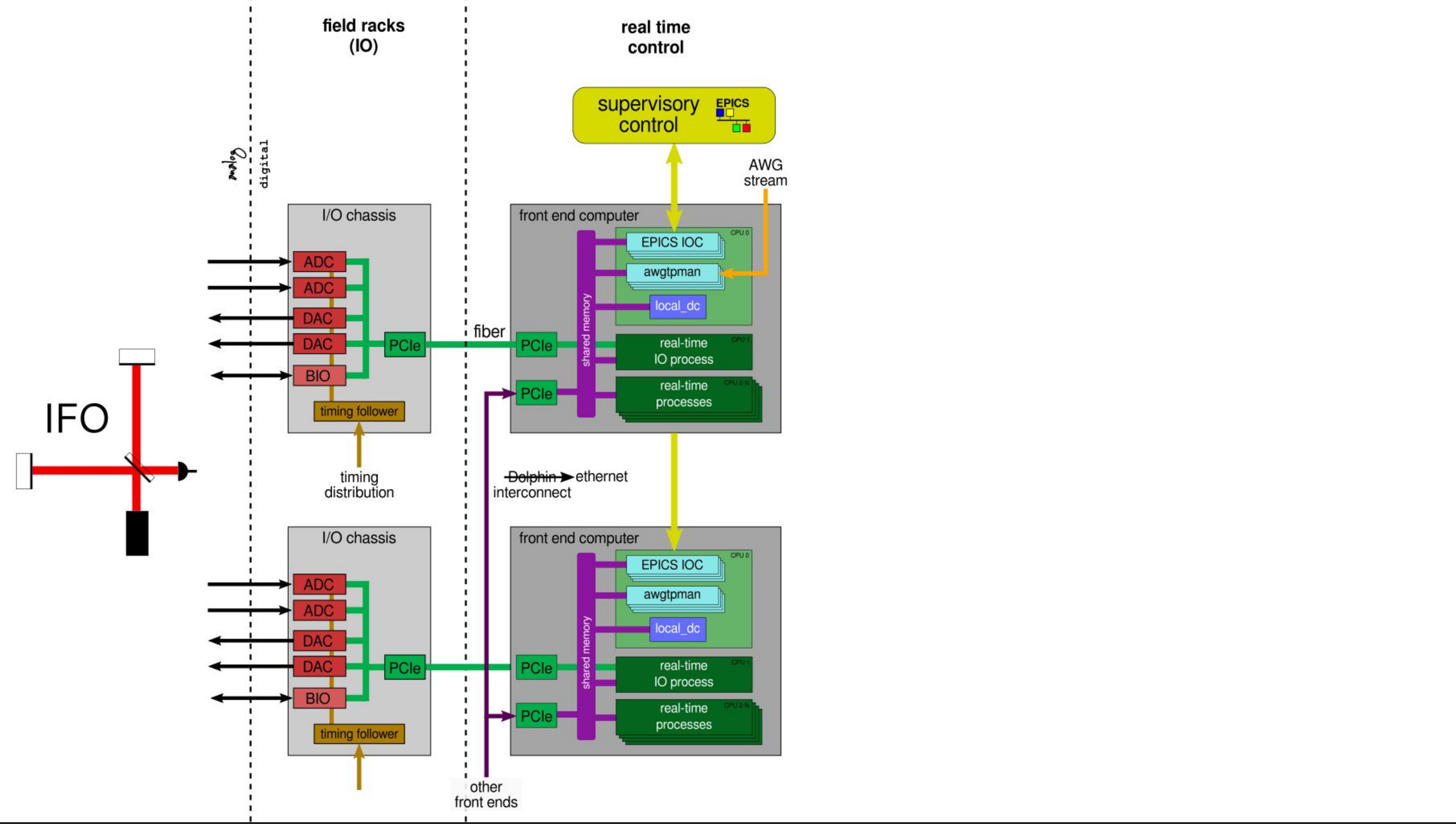
LIGO RT control and data architecture



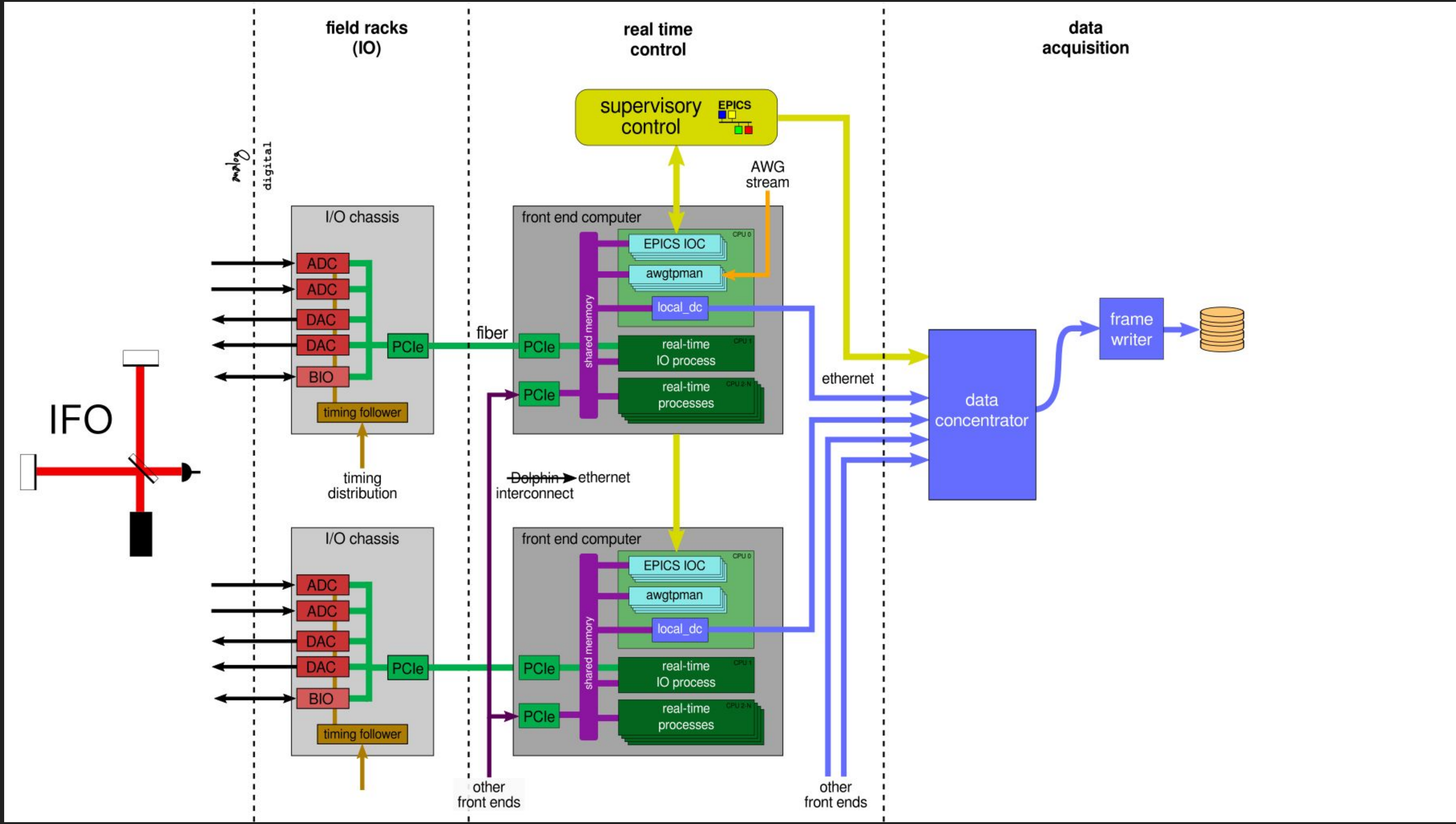
LIGO RT control and data architecture



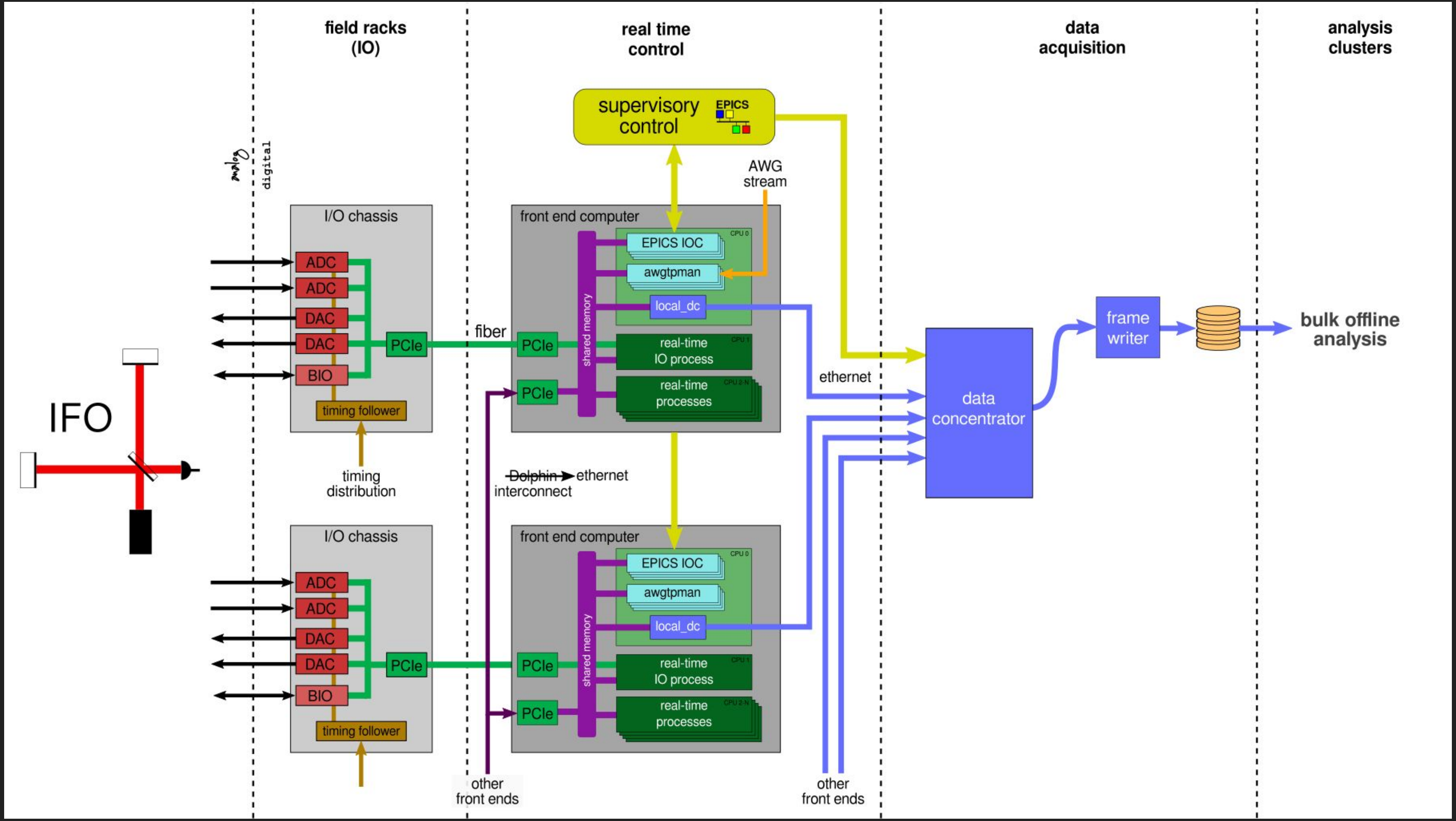
LIGO RT control and data architecture



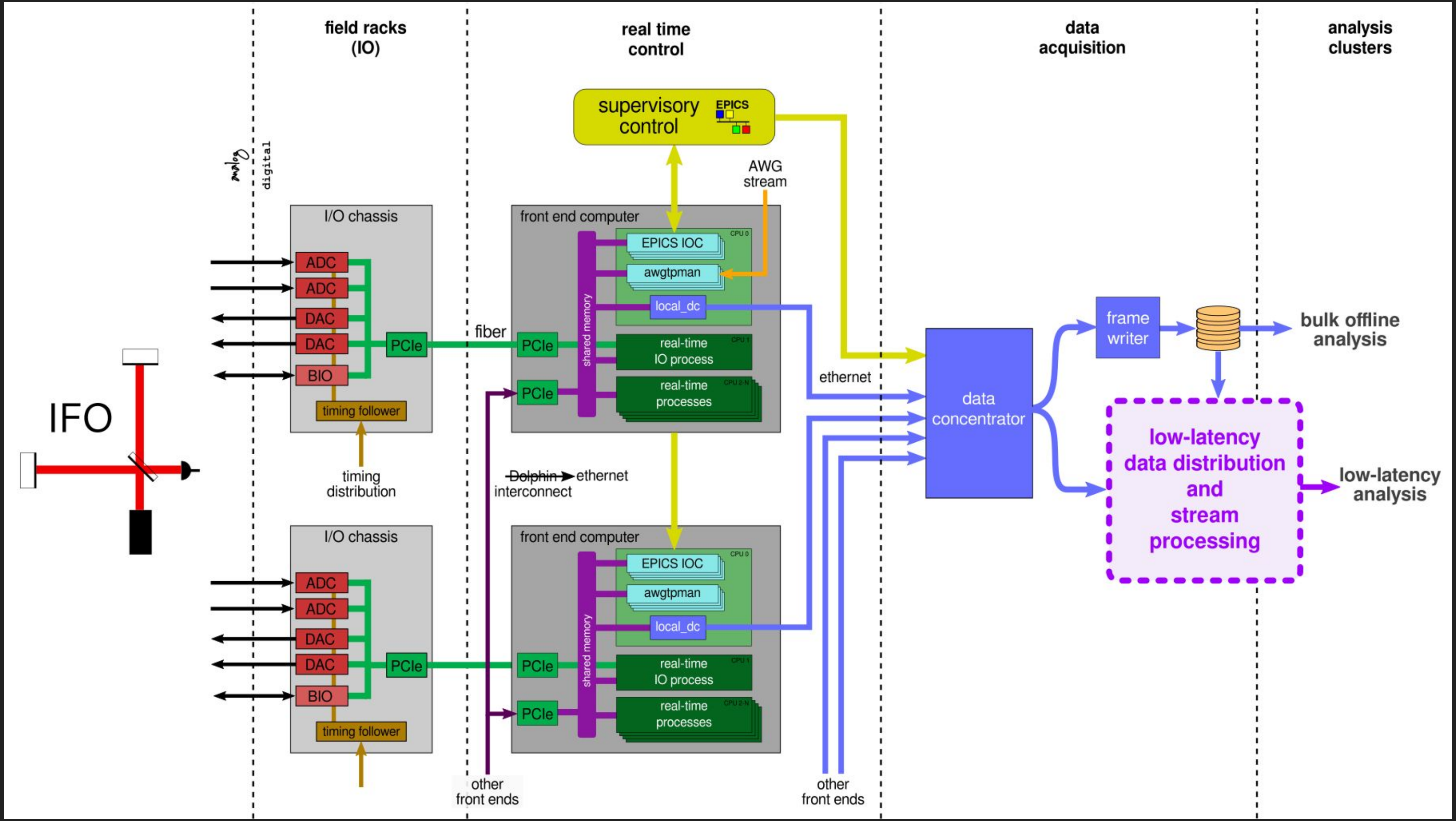
LIGO RT control and data architecture



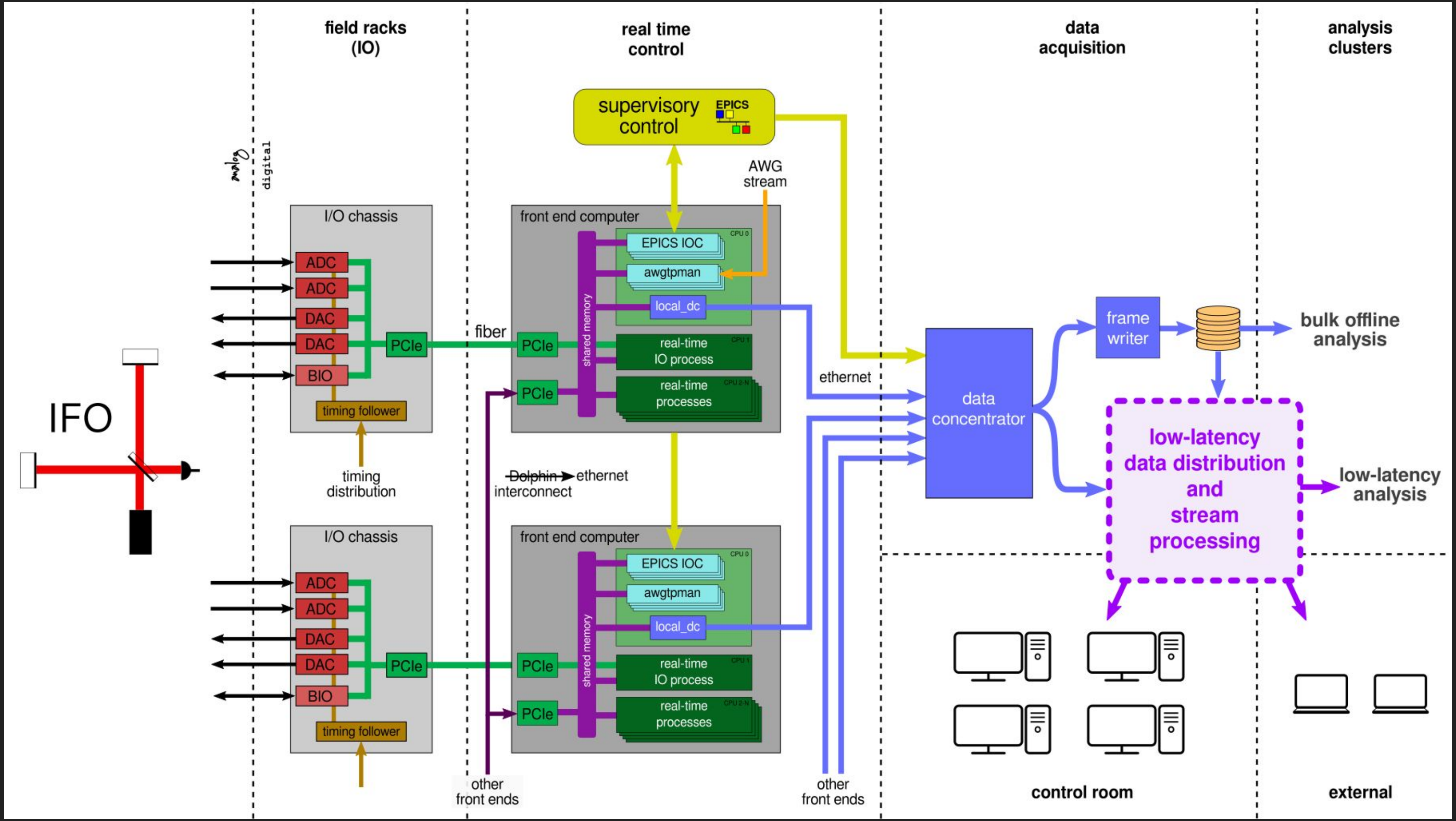
LIGO RT control and data architecture



LIGO RT control and data architecture



LIGO RT control and data architecture



Continuing RT development

Software:

- rewrite of RCG, library interface to models, better graph GUI
- first-class support for neural networks in the front end
- kernel → user space for real-time applications
- support for higher sample rates (currently 512kHz)
- many analysis tool and “backend” improvements
- better support for smaller deployments

Hardware:

- Dolphin →commodity ethernet for real-time interconnect
- in-house ADC/DAC cards
- better support for external devices

LIGO

3G challenges

Longer arms →lower bandwidth in arm control loops

- How much is this an issue? Affect on ISC design?

Faster loops?

- Where is this actually required? What are the requirements?
- The existing paradigm can support faster sample rates to a certain extent.
- LIGO is R&D'ing FPGA-based controller for common mode that will interface with the RTS.

From offline to online

So why the (relatively) new focus on low latency data processing?

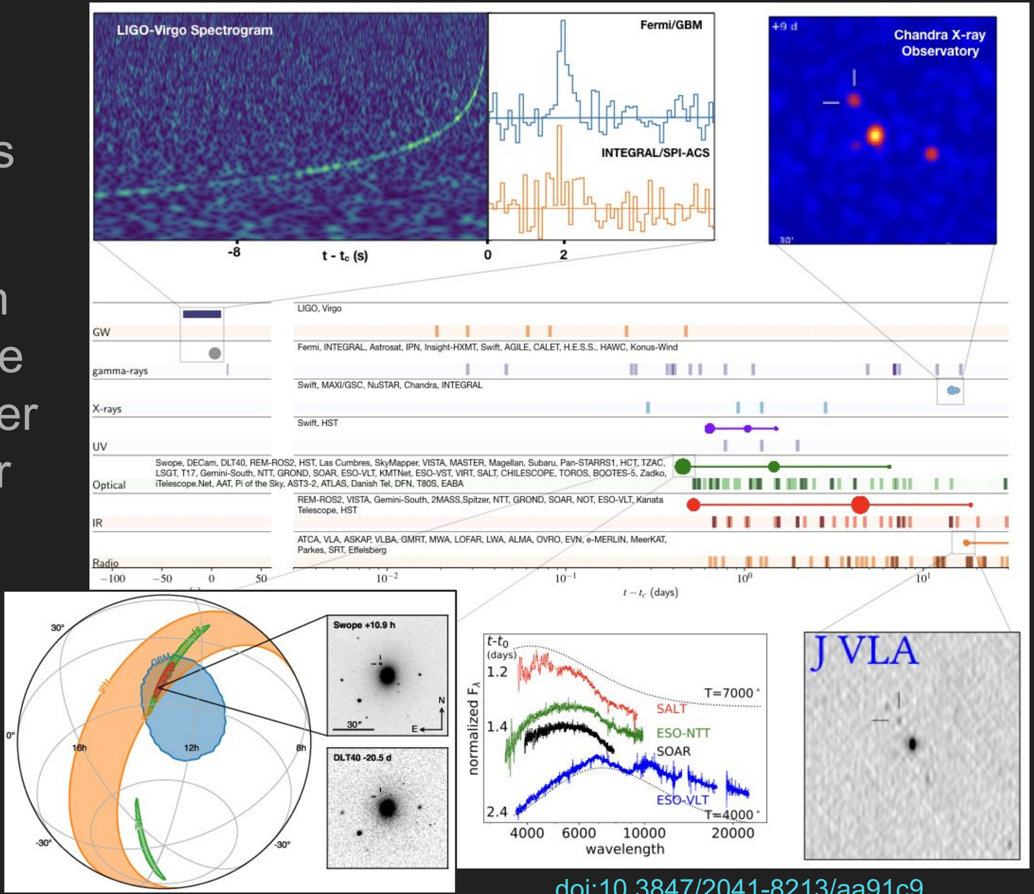
LIGO

A multi-messenger gold mine

The first “low latency” searches were deployed in 2009 during Initial LIGO’s sixth science run (S6).

But the effort really came to fruition in Advanced LIGO (2015), leading to the groundbreaking 2017 multi-messenger observation of the binary neutron star merger **GW170817** →

Informing astronomers of where to look for exceptional transient events soon became one of our most important science priorities.



doi:10.3847/2041-8213/aa91c9

Many challenges for low-latency alerts

But getting out high quality alerts to the astronomical community is *difficult*.

In particular, it requires good coordination between all observatories:

- *Localization improves with more detectors.*

Latency is also a major factor:

- *Direct correlation between the speed with which we can issue alerts and the quality of the science.*

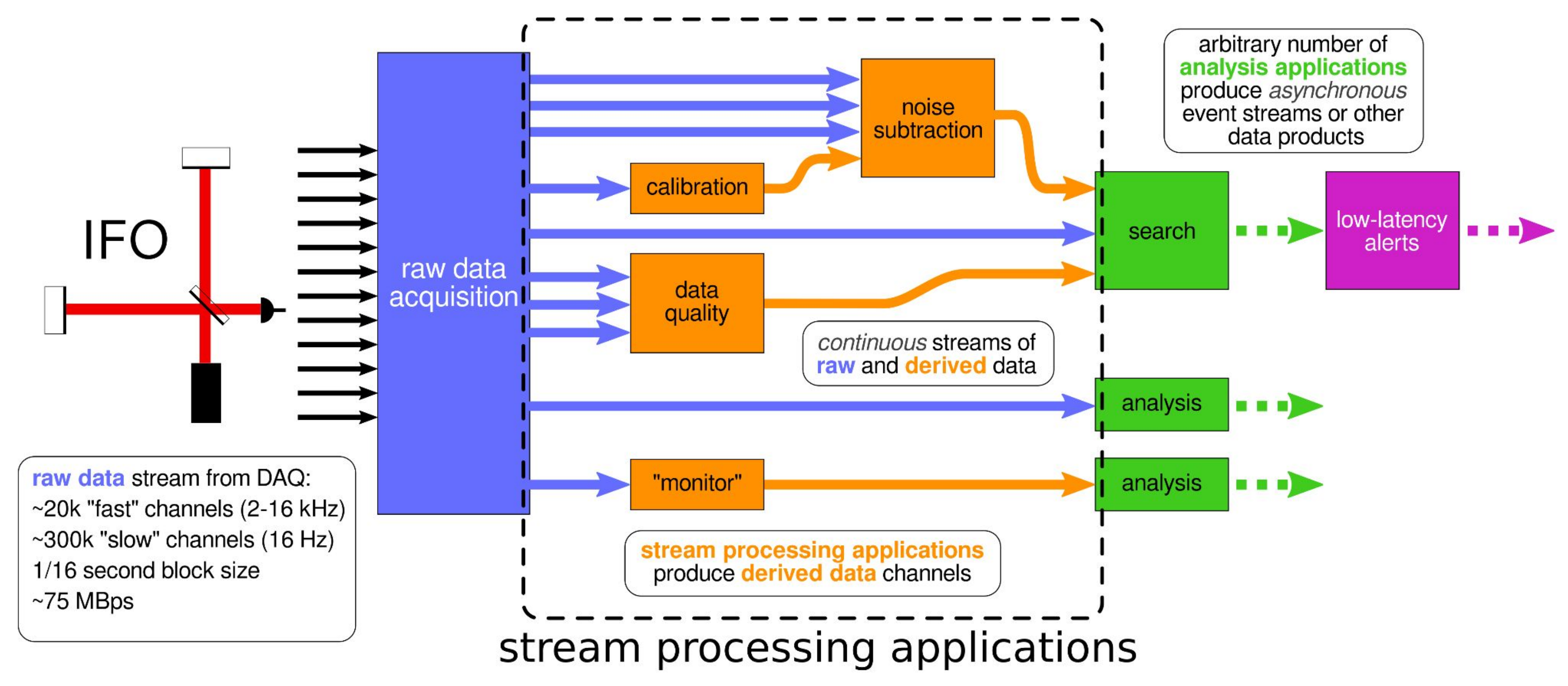
Each observatory must also do a lot of work to calibrate, clean, annotate, etc. their data before distributing it to the searches.

LIGO Next generation data delivery for LIGO

LIGO is overhauling of its stream processing and data distribution infrastructure, with the following goals:

- Focus on stream processing as primary for science goals.
- Lower latency data delivery.
- Modernize everything: use industry standard tools and protocols
- Unify data access methods: online/offline access with the same interface.
- Publish “derived” data streams that are immediately discoverable and accessible.
- Auto discovery of data, current and past, across all domains (federation).
- Lower barrier to entry: enable broader development of stream processing applications.

Low latency pipeline conceptual overview



Arrakis architecture and development

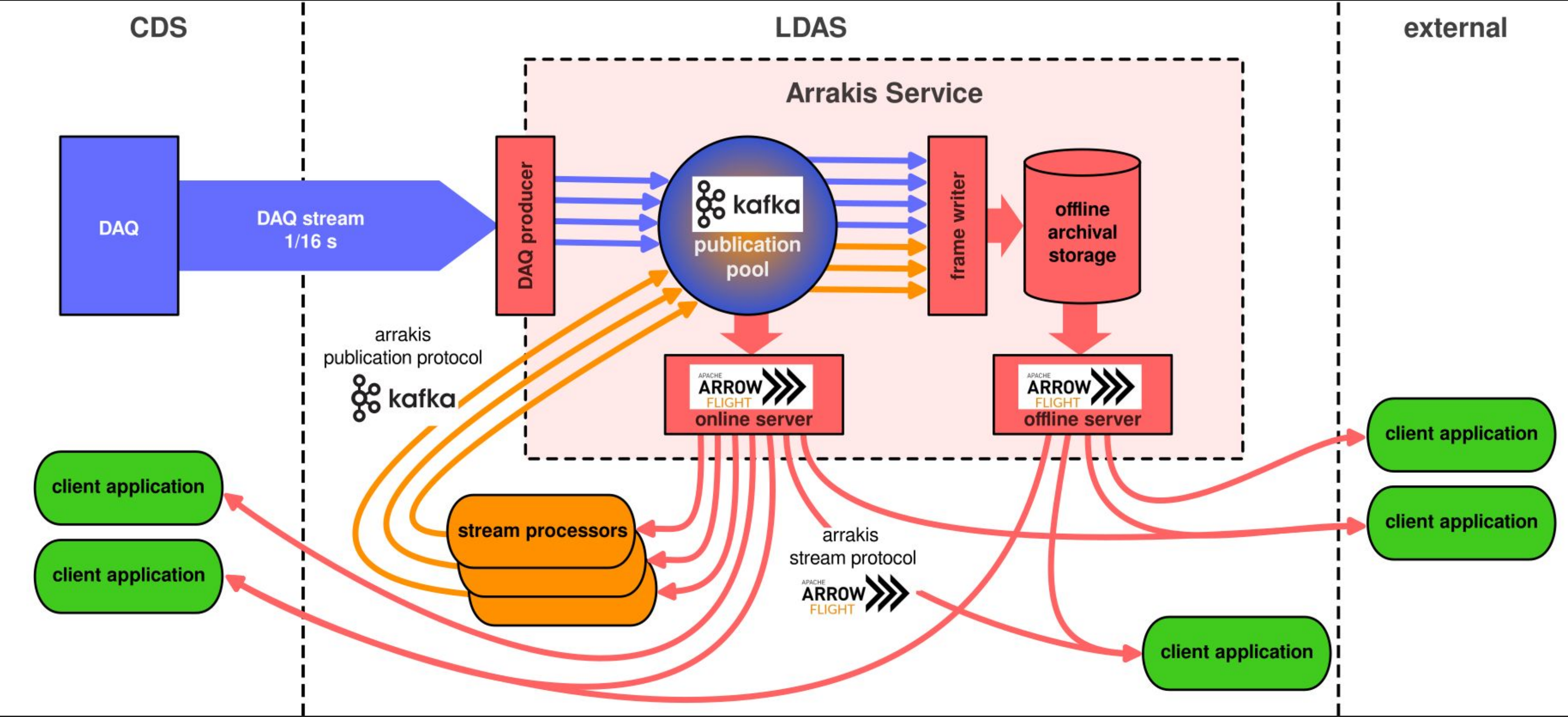
New **Arrakis** architecture overview:

- Arrow Flight (HTTP, gRPC) network protocol for both online and offline data access/streaming.
- Kafka-based derived channel publication.
- Phase 2: Kafka-based distribution of primary strain and state info channels directly to low-latency search pipelines.

Development status:

- Prototype services have been deployed at the sites, consuming all raw channels from the detectors.
- Reference client libraries for Python and Rust.
- Currently prototyping derived channel publication.

Arrakis overview



International Gravitational Wave Network

The next generation of facilities will need to work together and share data to enable the best science, as they do today.

Common infrastructure will be needed to support low-latency search, localization, and alerts.

Arrakis has been designed from the ground up to support a federated data distribution model.

LIGO Data Mesh: unifying disparate domains

Data Mesh is a data management and distribution concept that maps very well to our needs. The basic tenants are:

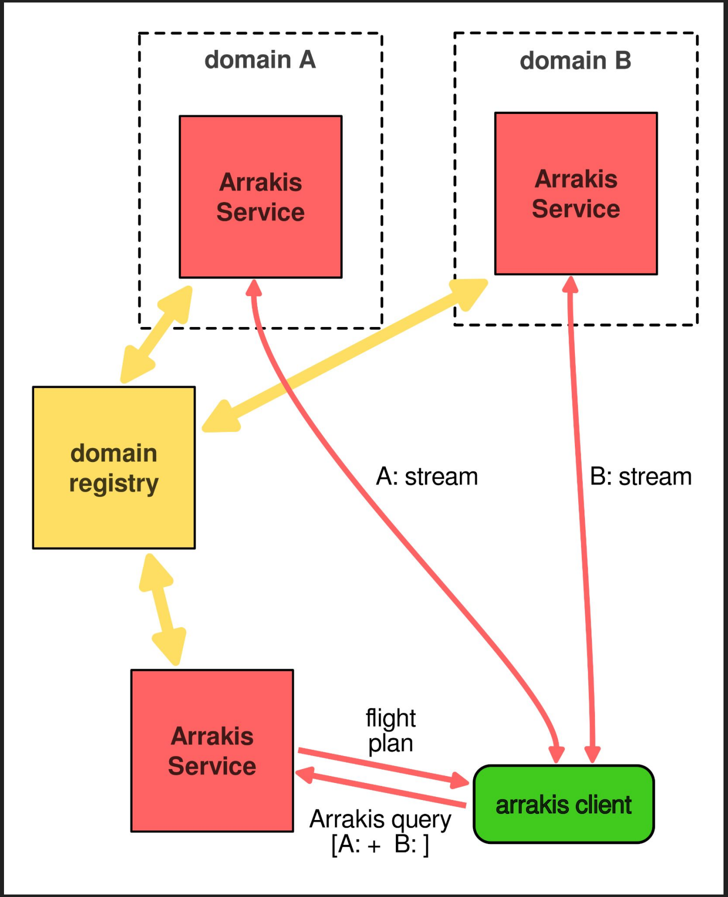
- *Distributed domain-driven architecture*
  - No centralized data store, different domains manage their own data products.
- *Data as a product*
  - Data is *discoverable, addressable, trustworthy, self-describing*, etc.
- *Self-serve data infrastructure*
  - Individual domains serve their own data.
- *Federated data governance*
  - Global standards for protocols and APIs, centralized registry of data products, etc.

Arrakis as a Data Mesh

How can we apply these principles to our network?

- Each “domain” (observatory, search pipeline, working group, etc.) defines and manages their own data products.
- Common shared protocols, APIs, and interfaces for access.
- Standardized data types.
- Distributed registry of available domains.
- Data discovery across domains.

Arrow Flight protocol enables much of this →



Arrakis as a Data Mesh for the global network

How can the Arrakis project enable this data sharing for the next generation of the world-wide network?

- DNS for distributed registry of domains
- Common standards:
    - on-the-wire discovery and access protocols
    - interface APIs
    - standard data types
- Reference client libraries and server implementations.
    - Reference clients already allows for streaming from multiple sources.
    - Reference server already provides plugable interface for multiple data backends.