Rucio beyond ATLAS
Experiences from Belle II, CMS, DUNE, EISCAT3D, LIGO/VIRGO, SKA, Xenon

Mario Lassnig **& Martin Barisits** *(ATLAS),* **Paul J Laycock & Cedric Serfon** *(Belle-II),*
Eric W Vaandering & Katy Ellis *(CMS),* **Robert Illingworth** *(DUNE),* **Vincent Garonne & John White** *(EISCAT3D)*
James A Clark & Gabriele Fronze *(LIGO/VIRGO),* **Rohini Joshi & Ian Johnson** *(SKA),*
Boris Bauermeister *(Xenon), and many more!*

RUCIO
SCIENTIFIC DATA MANAGEMENT

Why a common data management solution?

- Shared use of the global research infrastructures will become the norm, especially with sciences at the scale of HL-LHC, DUNE, and SKA
  - Competing requests on a **limited set of storage and network**, data centres will be **multi-experiment**
  - **Compute** is usually well-covered, e.g., via common scheduling, interfaces, and specifications
  - **Data** was always missing **a common open-source solution** to tackle our **shared challenges**

- Ensure more efficient use of available data resources across multiple experiments
  - **Allocate storage and network based on science needs**, not based on administrative domains
  - **Orchestrate dataflow policies across experiments**
  - Dynamically support compute workflows with **adaptive data allocations**
  - **Unify monitoring**, reporting and analytics to data centres and administration
  - Potential for **shared operations across experiments**

Rucio in a nutshell

- Rucio provides a mature and modular scientific data management federation
  - **Seamless integration** of **scientific and commercial** storage and their network systems
  - Data is stored in files and can contain **any potential payload**
  - Facilities can be **distributed at multiple locations** belonging to **different administrative domains**
  - Designed with **more than a decade of operational experience** in very large-scale data management
- Rucio manages location-aware data in a heterogeneous distributed environment
  - Creation, location, transfer, deletion, and annotation
  - Orchestration of dataflows with both low-level and high-level policies
- Principally developed by and for ATLAS, now with many more communities
- Rucio is open-source software licenced under *Apache v2.0*
- Makes use of established open-source toolchains

Rucio main functionalities

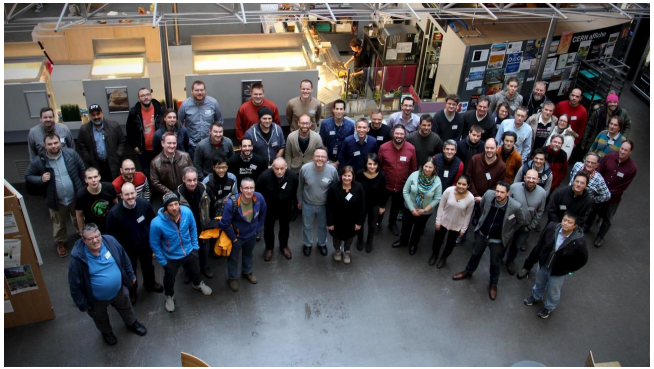
- Provides many features that can be enabled selectively

More advanced features

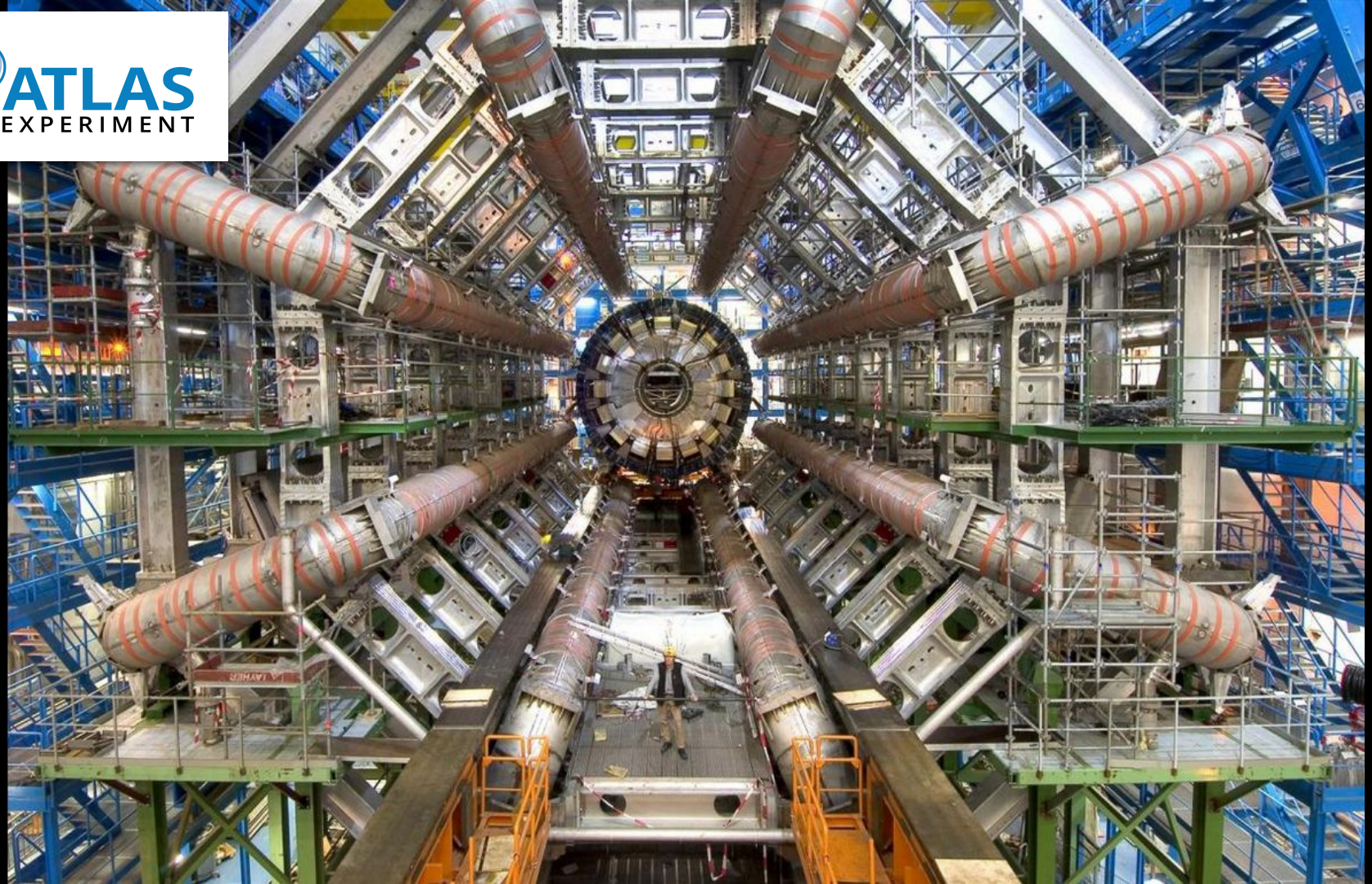
  - File and dataset catalog
  - Transfers between facilities including disk, tapes, clouds, HPCs
  - Web-UI, CLI, and API to discover/download/upload/transfer/annotate data
  - Extensive monitoring for all dataflows
  - Support for caches and CDN workflows
  - Expressive policy engines with rules and subscriptions
  - Automated corruption identification and recovery
  - Data popularity based replication
  - …
- Rucio can be integrated with Workload and Workflow Management System
  - Already supporting PanDA, the ATLAS WFMS
  - Communities evaluate & develop integrations, e.g., CRAB/WMAgent, DIRAC, Pegasus, or Condor

Regular events

- Rucio Community Workshops [2018] [2019]
- Rucio Coding Camps [2018] [2019]
- Development Meetings [Weekly]

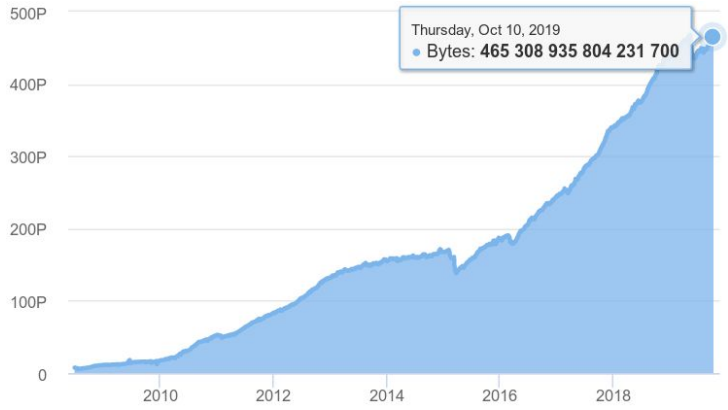


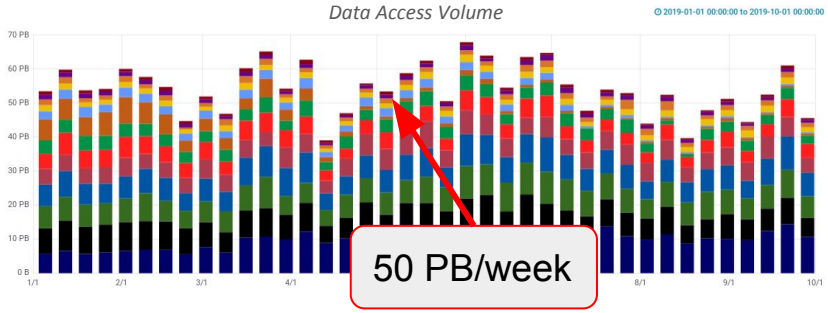
A growing community

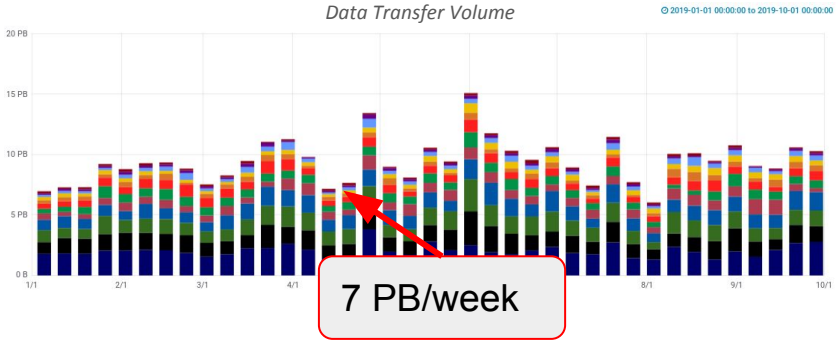

ATLAS
EXPERIMENT

Data management for ATLAS

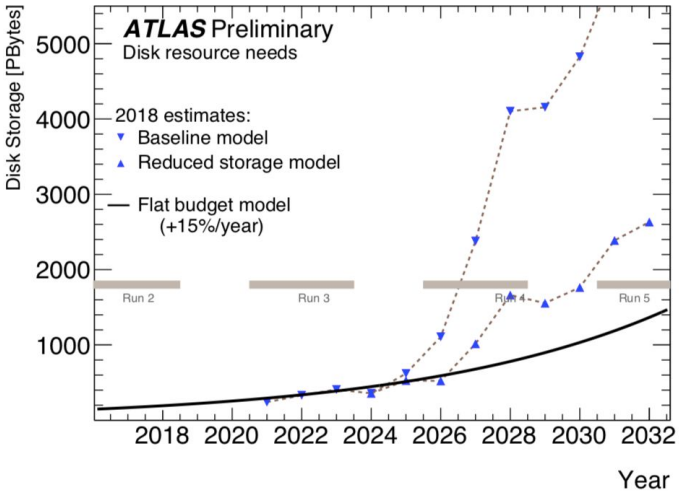
- A few numbers to set the scale
  - 1B+ files, 450+ PB of data, 400+ Hz interaction
  - 120 data centres, 5 HPCs, 2 clouds, 1000 users
  - 500 Petabytes/year transferred & deleted
  - 2.5 Exabytes/year uploaded & downloaded
- Increase 1+ order of magnitude for HL-LHC





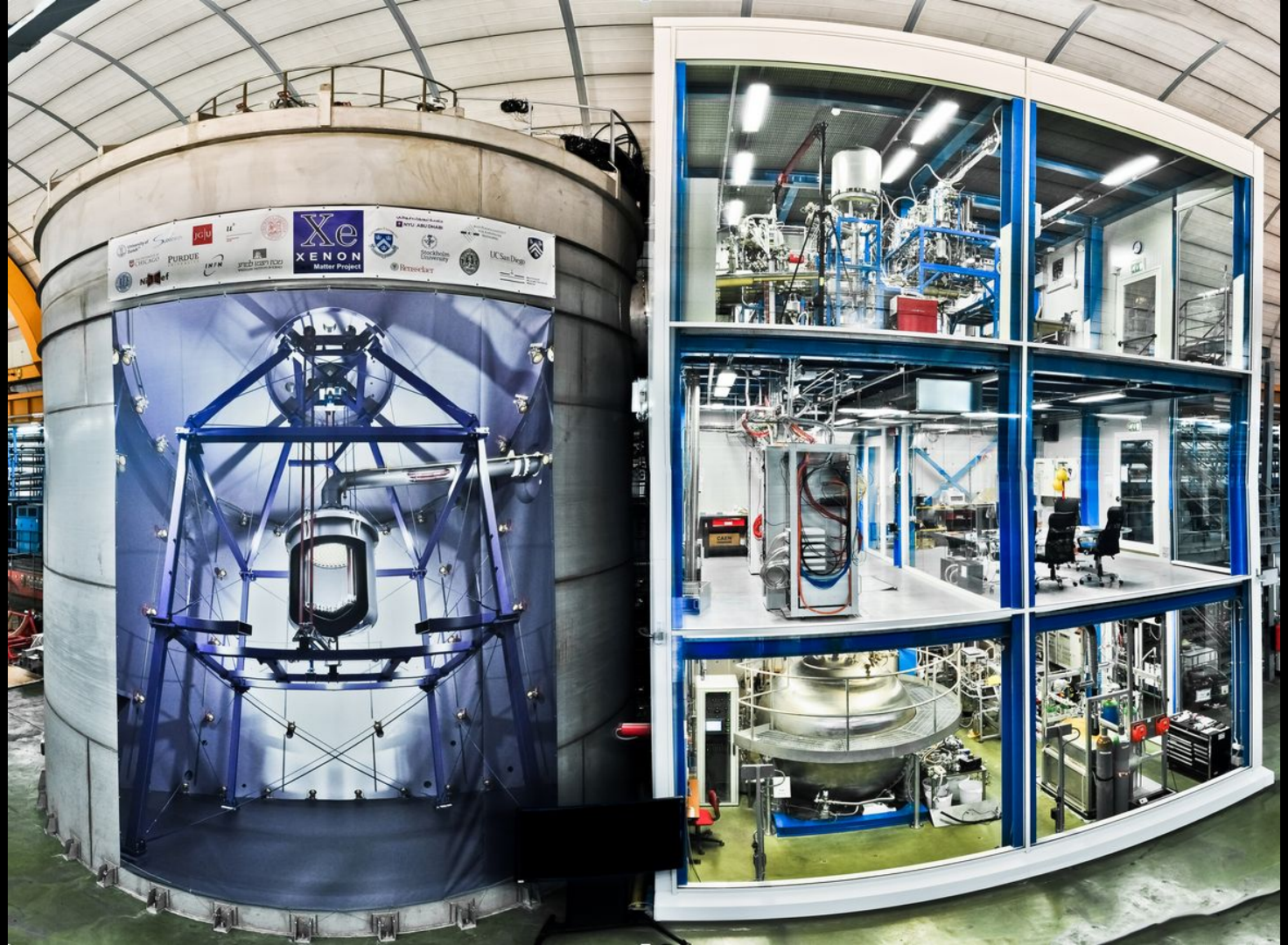


Data management for ATLAS at HL-LHC



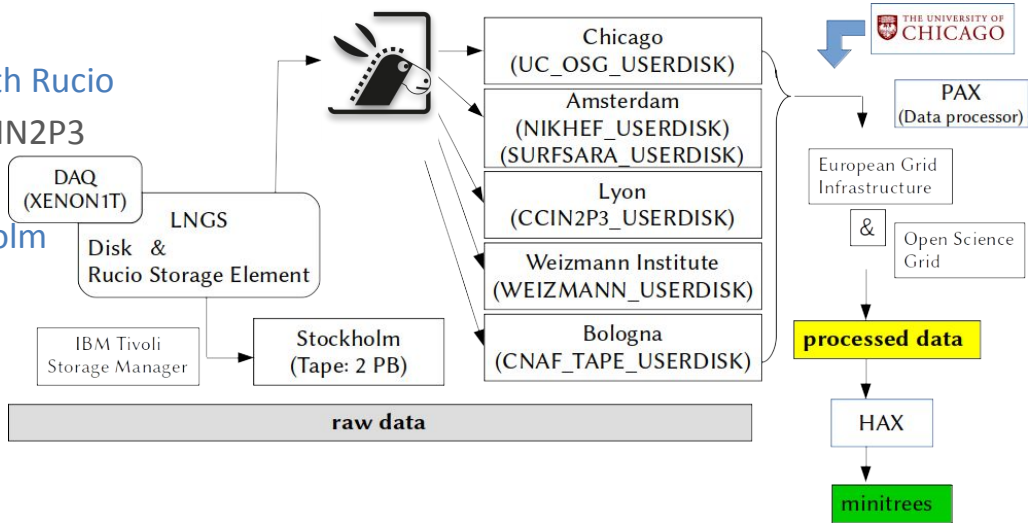
- Rucio is a central component to tackle HL-LHC data
  - Smart orchestration of the dataflow
  - Easy integration of new systems, ideas, and components
- Several combined effort R&D activities launched
  - Distributed storage and caching *Data Lakes*
  - Fine-grained data delivery services *iDDS & ServiceX*
  - Commercial cloud integration *Google & Co*
- R&D Highlight for HL-LHC: *Data Carousel*
  - Tight integration of workflow and dataflow for more **efficient use of high-latency storage** (i.e., tape)
  - New algorithms on **multi-site I/O scheduling** for both writing and reading
  - **Smart placement** of data on based on estimated access patterns

Community experiences


Xe
XENON
Matter Project
PURDUE
UC San Diego

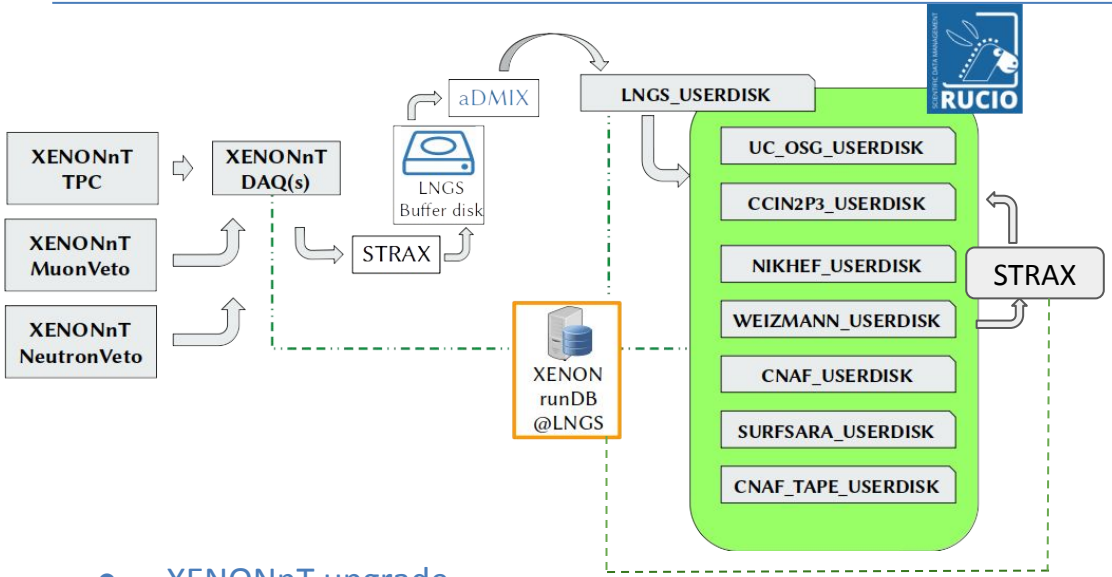
The XENON Dark Matter Experiment

- Gran Sasso National Laboratory LNGS
- Nuclear recoils in a liquid xenon target with TPC
- Data products
  - *raw*, *processed*, and *minitrees*
- Raw data are distributed and archived with Rucio
  - RCC Chicago, NIKHEF/SURFsara, CCIN2P3 Lyon, Weizmann, CNAF Bologna
- A Rucio independent tape copy in Stockholm
- Taken ~800 TB of raw data in XENON1T
- XENONnT upgrade will take 1PB/year
- Processing on three systems
  - European Grid Infrastructure (EGI)
  - Open Science Grid (OSG)
  - SDSC's Comet and HPC Campus



"The XENON1T Data Distribution and Processing Scheme" [EPJ Web of Conferences 214, 03015 (2019)]
"Dark Matter Search Results form a One Tonne x Year Exposure of XENON1T" [10.1103/PhysRevLett.121.111302]

The XENON Dark Matter Experiment

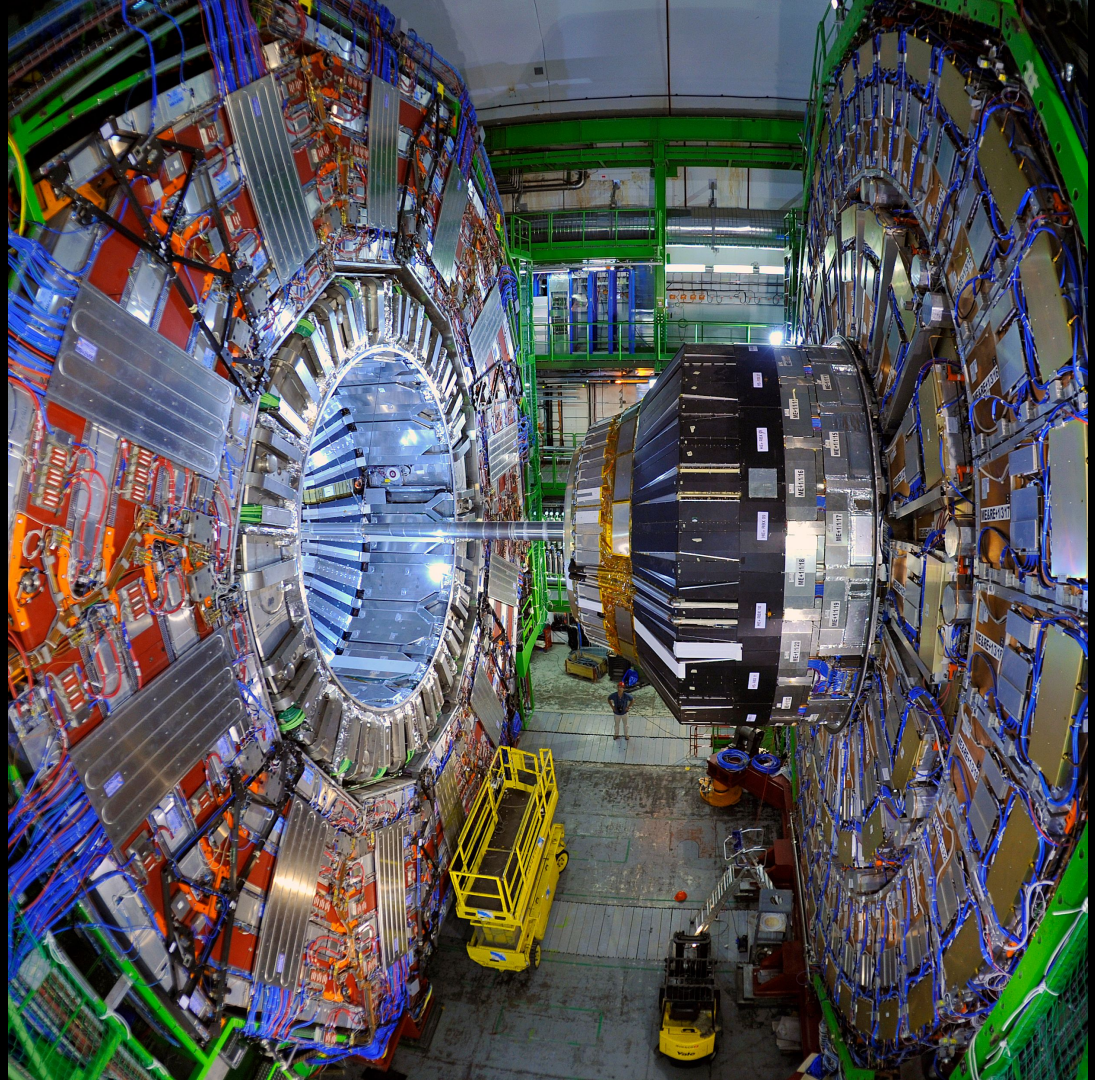


- XENONnT upgrade
  - Stream processor (STRAX) for multiple data products
  - aDMIX administration tool with Rucio integration
  - All data are distributed with Rucio
  - Tape storage integrated in Rucio this time
  - Hard Python 3 dependency

- Reprocessing campaigns
  - Job submitter (OUTSOURCE) for reprocessing campaigns on EGI & OSG with STRAX
  - Reprocessed plugins are distributed (aDMIX, Rucio) to analysts and registered to XENON run database

- Analysts
  - RCC Chicago is the data analysis center
  - User access high level data types at a near location via STRAX and aDMIX
  - Notebooks, Anaconda, Python, job submission like in XENON1T
  - Analysts can define/produce their own plugins for analysis purposes outside the run database

"The XENONnT Computing Scheme" [2nd Rucio Workshop]



CMS Data Management Challenge

- Data on tape O(100 PB) and disk O(50 PB)
  - 8 sites with tape, O(100) with managed disk
  - Production file size O(1 GB), user file size O(100 MB)
  - Per day transfers ~2 PB, 1 M files (user & production)
- Current data management is done by two layers of in-house products
  - **Each site must host an agent** to manage its own data including tape
  - Requires **non-trivial effort** at each of our sites
  - Transfer portion is aging and **may not scale** to HL-LHC
  - **Second layer** makes requests to **dynamically distribute and clean up** data based on experiment plans and popularity
- No user data *management*
  - User data transfers with thin layer over FTS

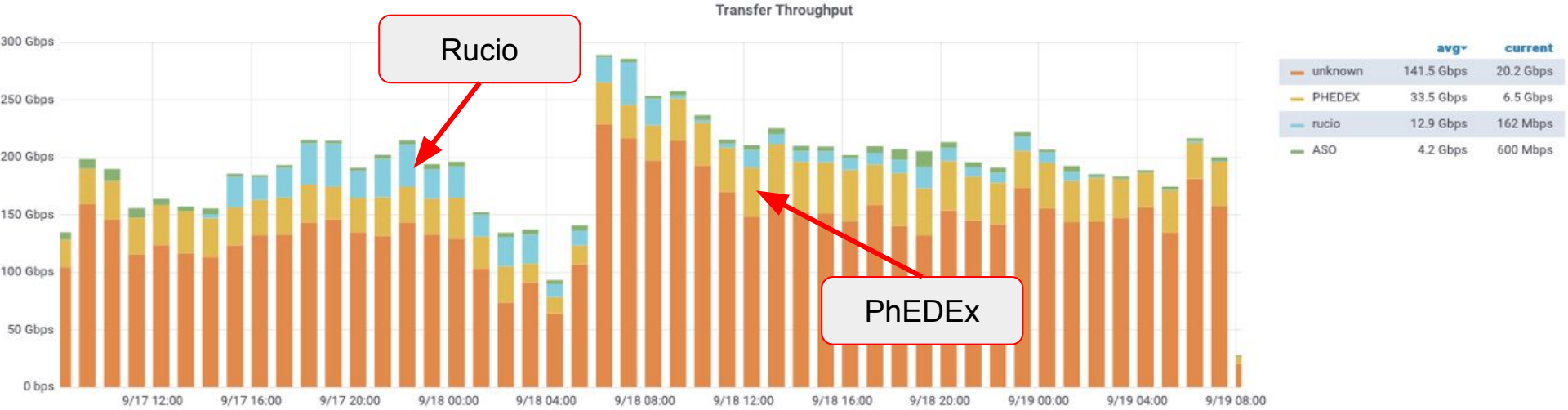
CMS Selection and Transition Processes

- Performed an evaluation and down-select from early 2018 through summer 2018
- **Ready for LHC Run 3**: Transition period from 2018 - 2020

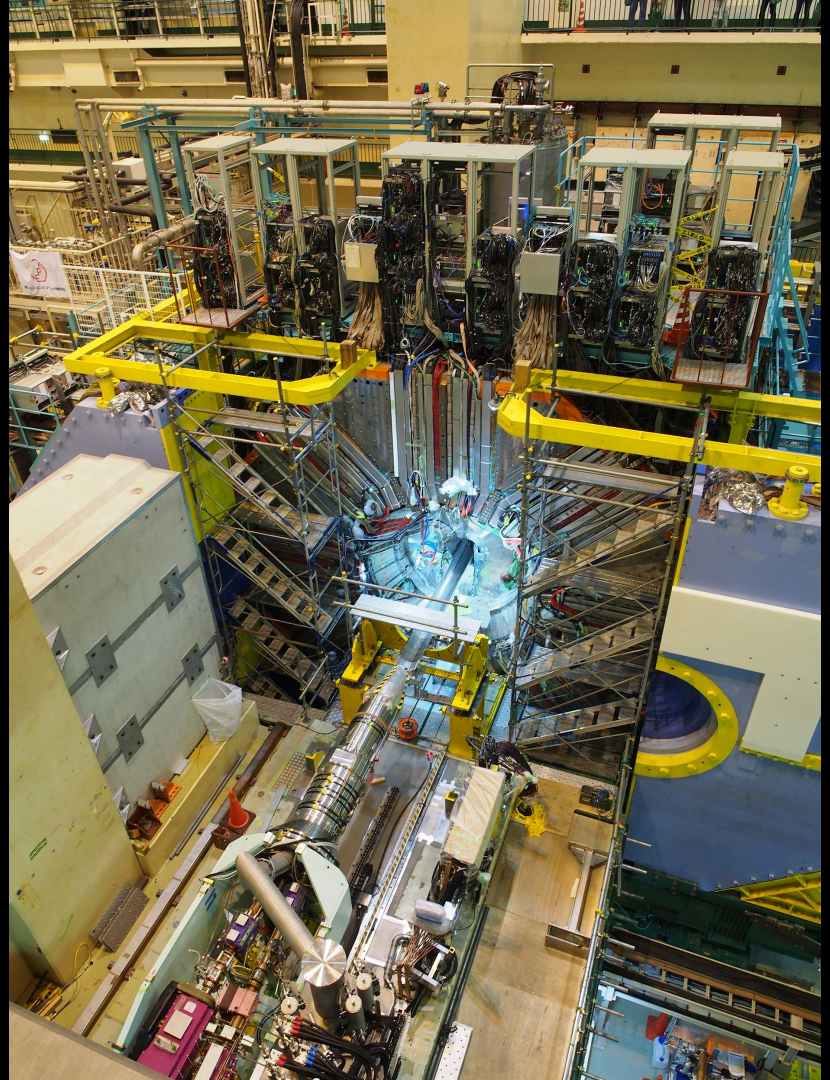
- **Excited to participate in a community project with a plan for the future!**
- Production infrastructure based on Docker, Kubernetes, Helm, OpenStack
  - Customizing official Rucio helm charts enables minimal config changes for CMS
  - Zero to operating cluster including dependencies is ~30 minutes
  - Upgrades are nearly instantaneous
- Allows CMS to have production and testbed on a shared set of resources
- Developer's environment is identical to various flavors of central clusters

CMS Million File Tests

- Distribution of 1 million files between all CMS T1 and T2
  - Critical factor for data management scalability is number of files, not volume of data to be moved
  - Entire test took 1.5 days, purely driven by dataset injection rate
- Ran in parallel to regular experiment activity



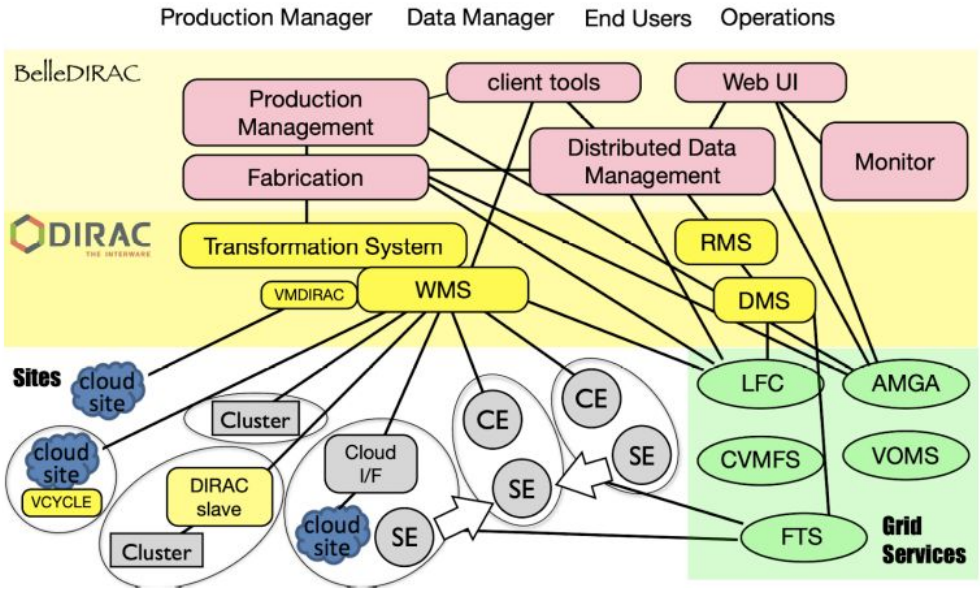
Belle II



Belle II Computing Challenge

Study the properties of B mesons

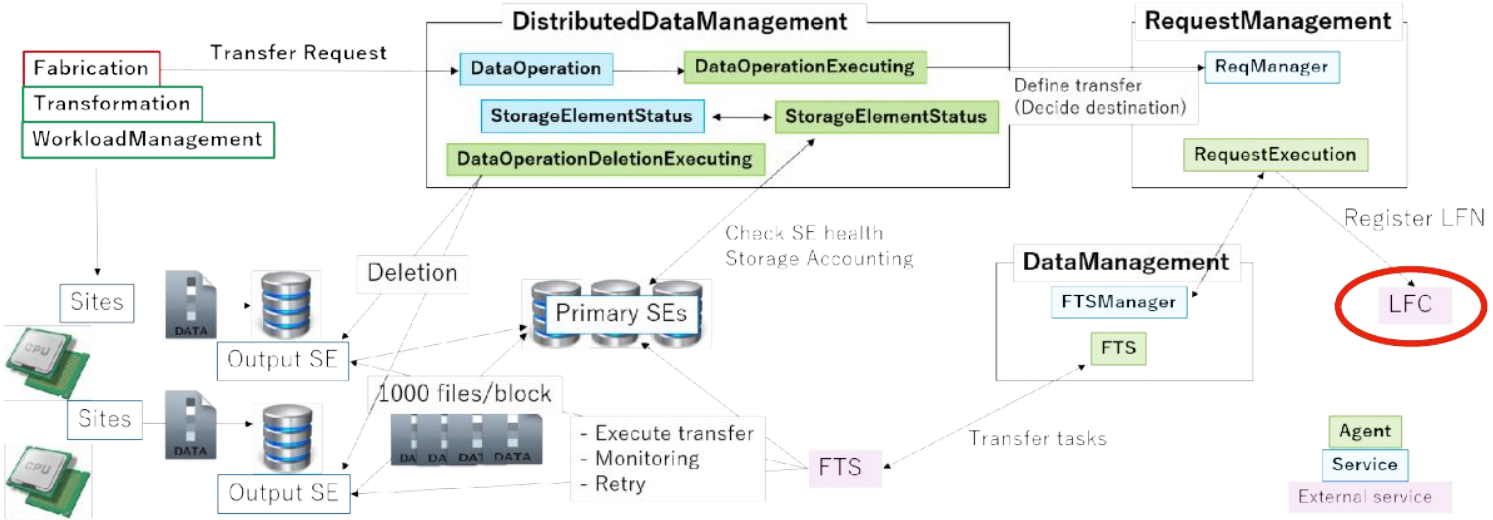
981 members
118 institutes
26 countries



70 storage areas
130 PB of raw data
with 2 replicas

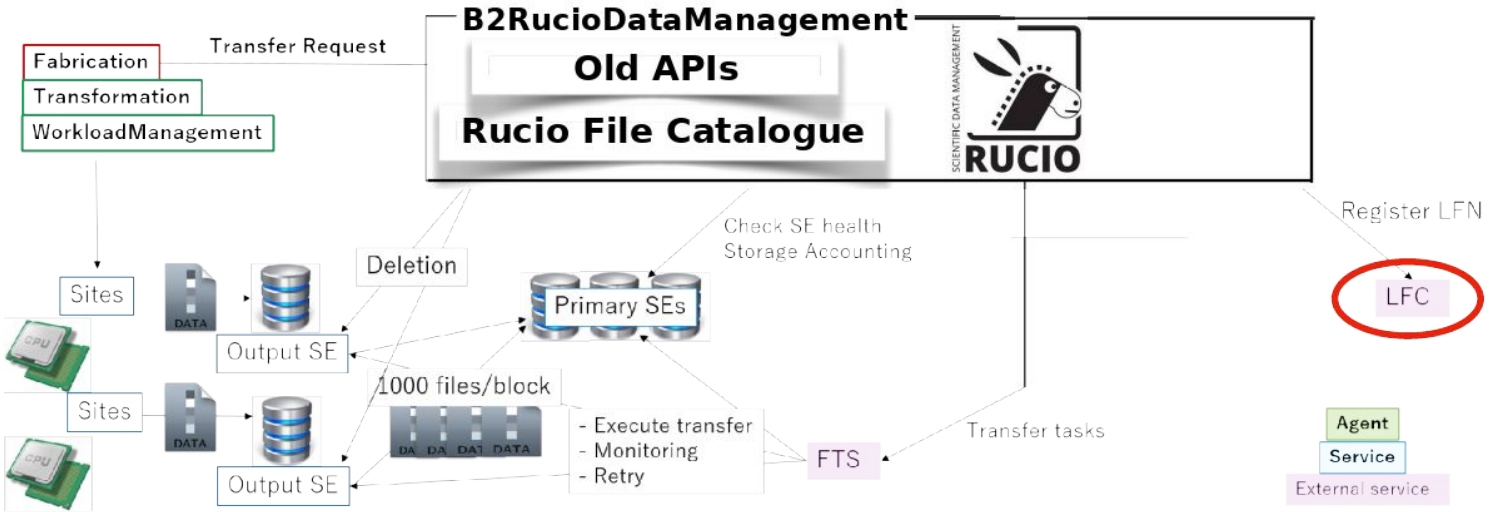
Physics data taking Phase 3 started this year

Belle II Distributed Data Management



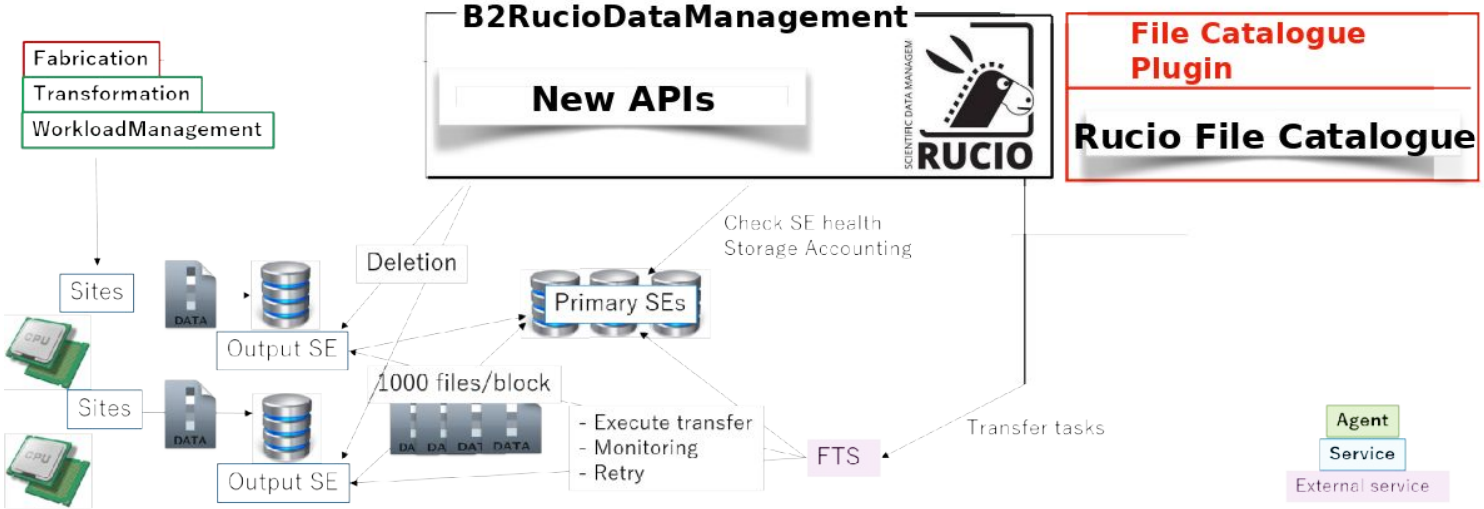
- Currently using a **bespoke design**, performance ok, supports up to 150k transfers/day
  - Some scalability issues addressed, some scalability issues inherent to design
  - **Lack of automation**: data distribution/deletion by experts with too fine granularity
- **Evaluate Rucio** as an alternative, all studies so far look promising
- Performance on PostgreSQL @ BNL shows capability beyond Belle II requirement

Belle II Distributed Data Management Plans

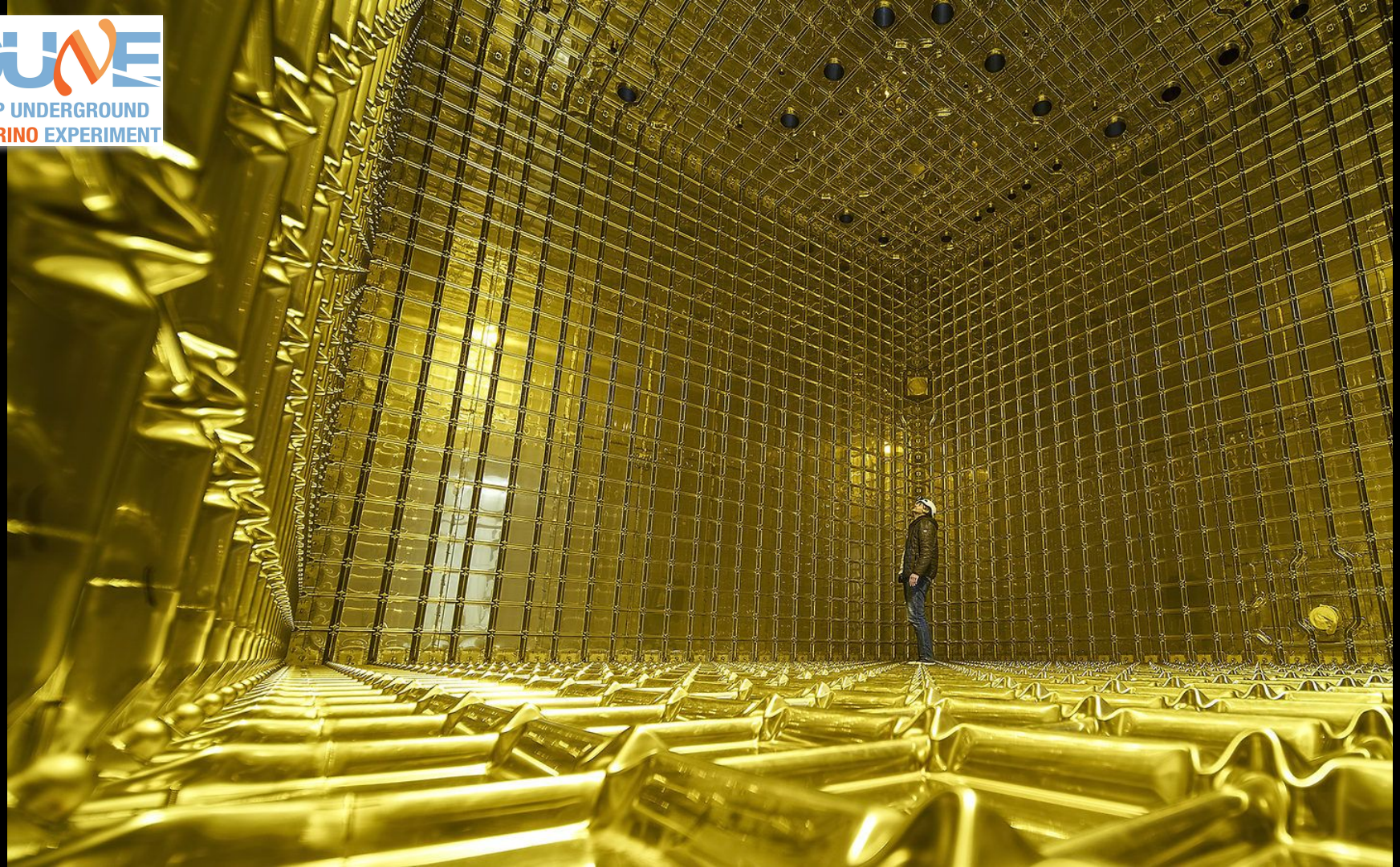


- **First stage migration:** Replace implementation with Rucio under-the-hood
  - **Pro: Mostly transparent** to the rest of Belle II, capable of backing out if really needed
  - **Con:** Still relying on LFC as file catalogue, **not taking full advantage of Rucio**
- **Aim:** Gain **experience in production** environment of using Dirac with Rucio

Belle II Distributed Data Management Plans

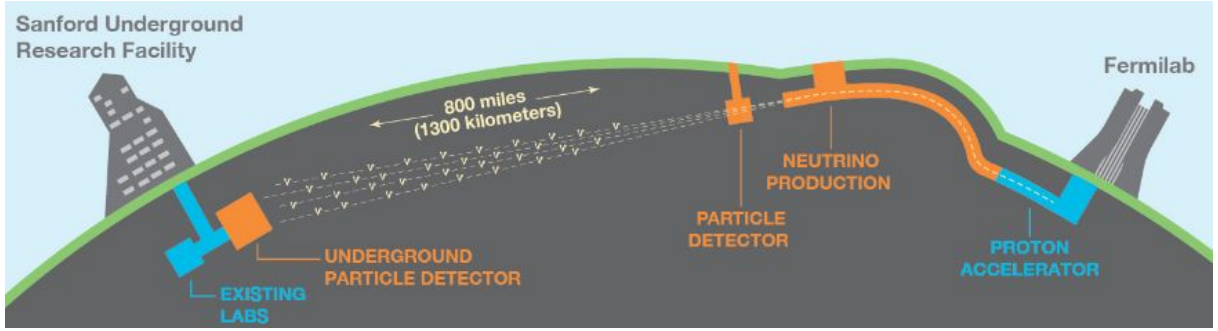


- **Second stage migration**: Rucio is master file catalogue using a plugin to remove dependency on LFC
  - Every component has to interact with the **master file catalogue**
  - File catalogue plugin must **hide Rucio requirements** from Dirac and Belle II users
- Working **in collaboration with UK (Imperial)** on the file catalogue plugin


DUNE
DEEP UNDERGROUND
NEUTRINO EXPERIMENT

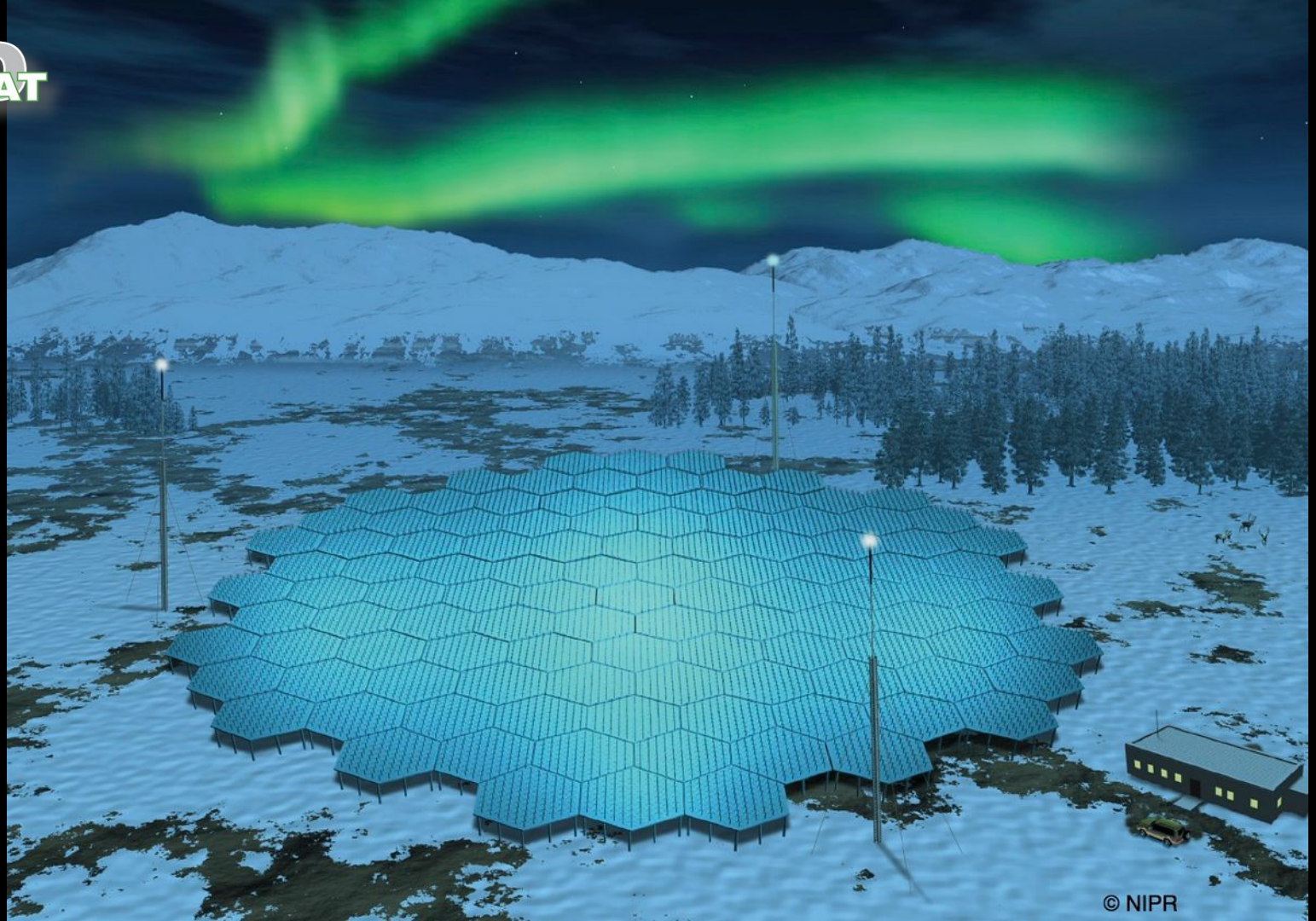
DUNE Data Management Challenges

- Multiple **geographically separated detectors** asynchronously collecting data
  - Eventually expect **10s PB/yr**
  - **Sensitive to supernovae**: potentially produce 100s TB over a 100 second period
- Large collaboration intends to **store and process data at many sites** worldwide
  - ProtoDUNE (*previous slide*) recorded + test-beam (Sep 2018) reconstructed 6PB data
  - Expecting next test beam run for both single and dual phase prototypes in 2021-22 timeframe



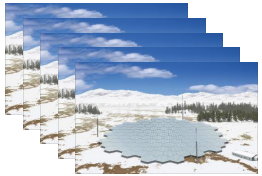
DUNE Rucio

- DUNE has a **Rucio instance at Fermilab** with PostgreSQL backend
  - ~1 million files catalogued so far — ProtoDUNE raw and reconstructed
- Rucio is being used to **distribute ProtoDUNE data** from CERN and FNAL to other sites
  - **Replication rules make this easy**; make a rule for a dataset and site or group of sites and just wait...
- Integration plan
  - **Progressively replace** the legacy data management system, transition **to a purely Rucio based** solution
- Challenges
  - DUNE intends to make **heavy use of HPC resources**
    the data management needs to integrate with many very heterogeneous supercomputing sites
  - DUNE data could **benefit from fine grained object store style** access
    not clear how to combine this with the traditional file based approach


3D
EISCAT
© NIPR

EISCAT_3D: Atmosphere & Ionosphere 3D Radar

- Data intensive instrument generates a high volume of data



3 phased-array antenna fields
distributed in the Nordics
One Transmit/Receive and two Receivers

High volume of data:
radio waves at 233 MHz,
signal bandwidth ~5 MHz

10s of TFLOPS compute cluster for real time signal processing and batch data analysis tasks

Data replication

- Researchers need to analyse data and share their results
- Can the data replication be automated? Can it by synchronised with third-party systems, such as data management tools and catalogs?

Data management services for EISCAT_3D

- Service portfolio for full data management with the same components as the LHC experiments

Distributed storage	NDGF NDGF-PREPROD SNIC	srm.ndgf.org preprod-srm.ndgf.org gsiftp.swestore.se	pools: nsc.liu.se pools: uio.no pools: snic	1 PB 1 TB 1 PB
Transfer service	FTS	https://fts.grid.uiocloud.no:8449		
Data orchestration	Rucio Clients	https://beauregard.ndgf.org:443 https://hub.docker.com/r/vingar/rucio-clients-eiscat3d		
Monitoring	Kibana	https://chaperon.ndgf.org/kibana/		

Automatic replication exercise

Data upload

/eiscat3d/upload/

dCache@NDGF preprod

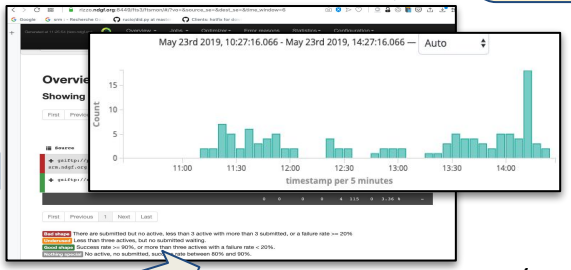
Data replication

/eiscat3d/

dCache@NDGF

Data replication

Storage events
New files



/eiscat3d/

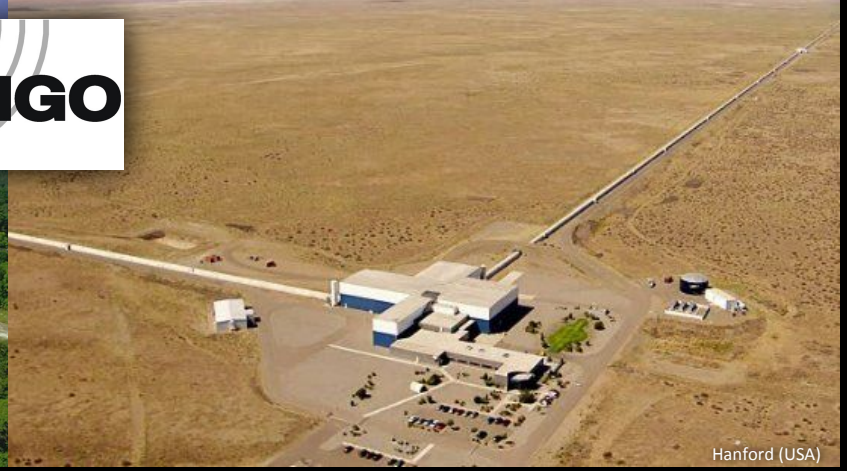
dCache@SNIC

Panoptes Service
https://pypi.org/project/dcachecliient/
Detect new files and register them into Rucio for replication and data sharing

Rucio, FTS

Data orchestration
automatic transfers, distribution and deletion

Livingston (USA)



Hanford (USA)

Cascina (IT)

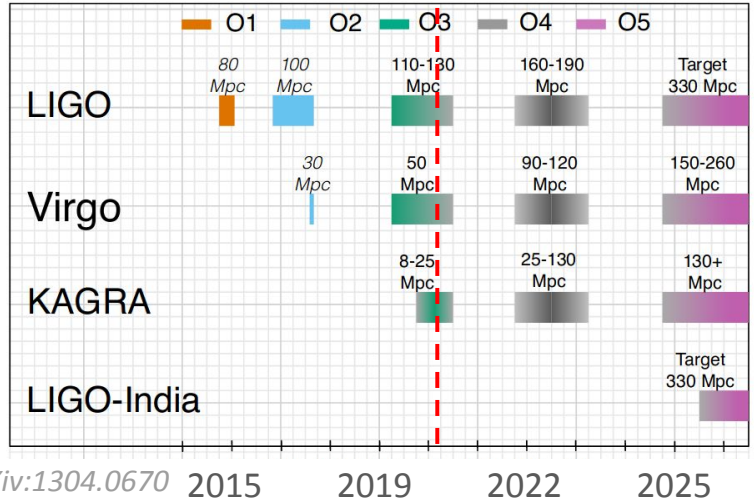
International Gravitational Wave Network (IGWN)

LIGO/Virgo ~20TB of astrophysical strain data

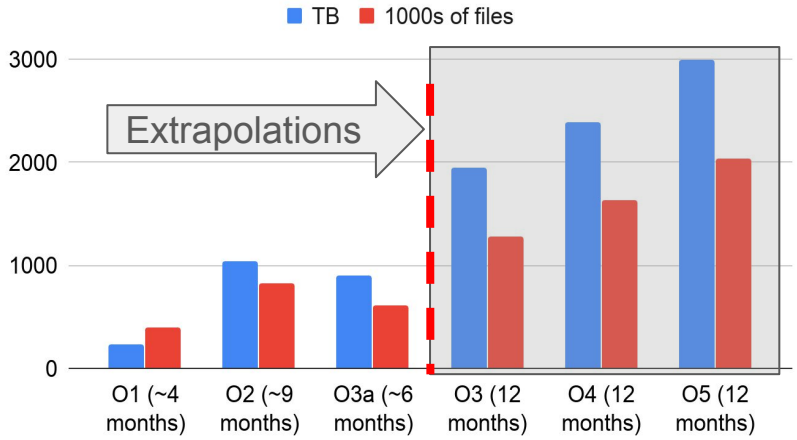
~1PB of raw data (environmental, instrumental monitors) per instrument per observing year

Near real-time "online" analyses: data streamed with Kafka to dedicated computing resources

Offline "deep" searches, parameter estimation: dedicated + opportunistic resources & archival data

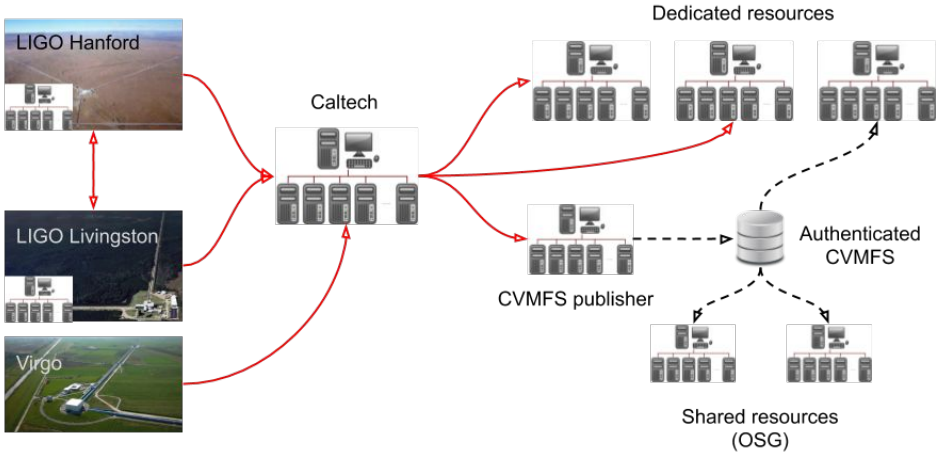


arXiv:1304.0670



IGWN Archival Data Distribution

- LIGO Data Replicator (LDR)
  - Legacy data distribution system
  - Using MySQL & Globus
- Rucio enhances data management
  - Choice of protocols
  - Accessible catalog
  - Comprehensive monitoring
- Detector data
  - Domain-specific daemons register new frames in Rucio catalog
  - Rucio rules/policies trivially implement dataflow to archives and resources
- Many opportunities beyond this!



- Data from each instrument is archived at detector sites
- All data archived centrally at Caltech
- Reduced datasets replicated to selected dedicated resources and published to CVMFS for broader access

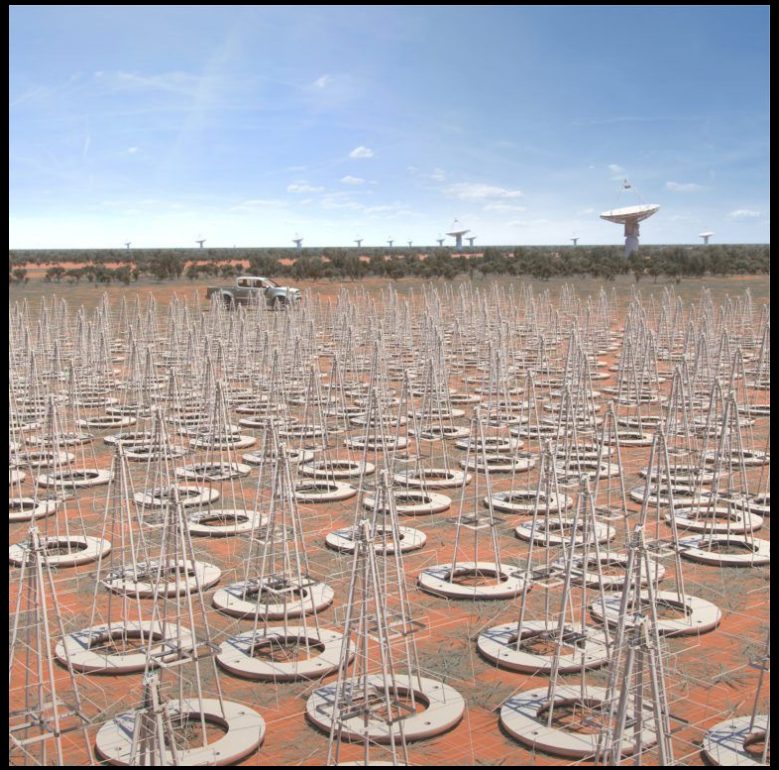
IGWN Rucio Deployment & Opportunities

- Collaboration with OSG & IceCube personnel → Rucio services deployed on Nautilus hypercluster
  - Web server, daemons & PostgreSQL running in Kubernetes
  - PostgreSQL database persistence through CephFS @ Nautilus
  - Using CERN FTS, interest in hosting our own as needed
  - Kubernetes state monitoring @ https://ligo-rucio-grafana.nautilus.optiputer.net/

- Rucio now being used in production for limited frame data replication to volunteering sites, expect transition away from LDR over coming months

- As well as updating to a modern, high-availability version of existing functionality, excited to explore
  - Integration of existing data discovery services & remote data access, e.g., HTCondor file transfers
  - Enhanced database redundancy
  - Management of new data products, e.g., analysis pipeline data products
  - Mountable Rucio POSIX namespace under development as CVMFS data distribution alternative

SKA
SQUARE KILOMETRE ARRAY

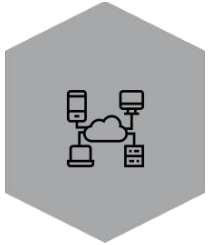


SKA Regional Centres

- SRCs will provide a **platform for transparent data** access, data distribution, post-processing, archive storage, and software development
- **Up to 1 PB/day** to be ingested from each telescope, and made available for access and post-processing
- Need a way to **manage data in a federated way** across many physical sites transparent to the user

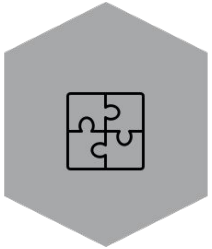
ARCHIVE

DATA DISCOVERY



DISTRIBUTED DATA PROCESSING

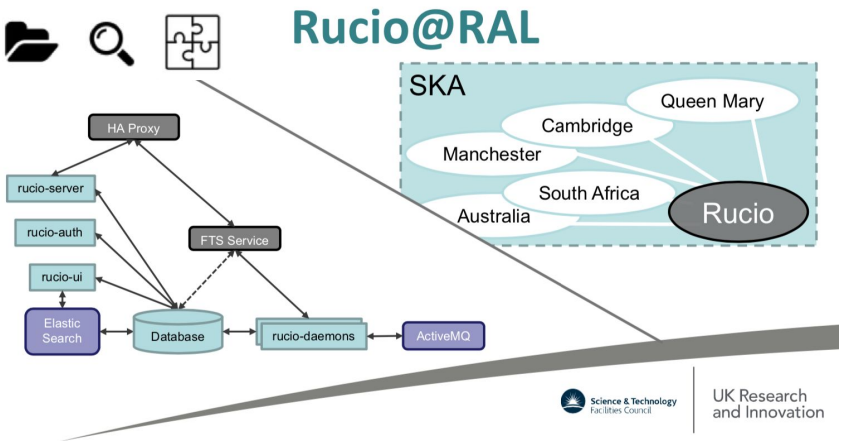
USER SUPPORT



INTEROPERABILITY

Evaluating Rucio for SRC data management

- Data uploaded, replicated, deleted from storage elements using parameterised **replication rules**
- Demonstrated **data transfers** from ZA to Manchester
- **Functional tests** demonstrating a network mesh test
- SKA **Pathfinder data** used for tests
- **ELK monitoring stack** is up, 8M events from 1+ years



Count Matrix - transfer events

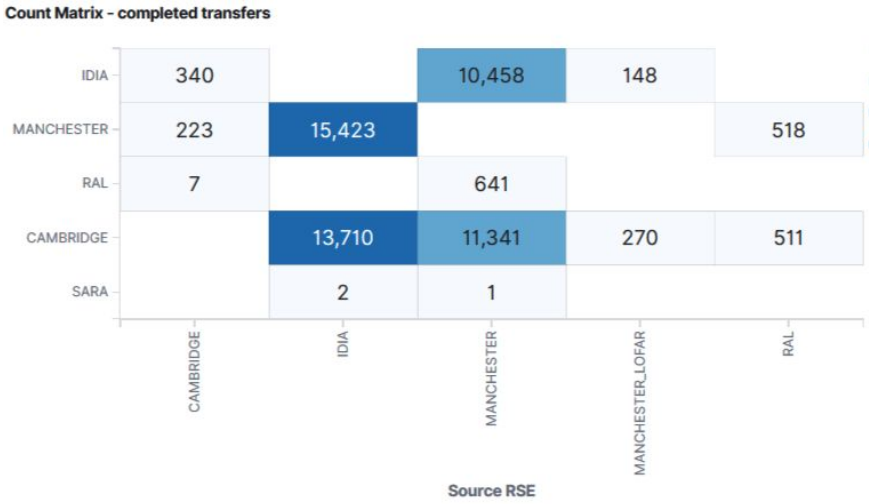
Source RSE	CAMBRIDGE	IDIA	MANCHESTER	MANCHESTER_LOFAR	RAL
IDIA	7,261		2,005,828	183	
MANCHESTER	3,764	45,880			23,307
QMUL	4,279		182,284		175,873
RAL	13,176		28,693	15,737	
CAMBRIDGE		92,012	123,768	1,019	23,640
SARA		2			
AARNET_MEL			12		

Thanks to Eli Chadwick, Ian Johnson

Thanks to Andrew Lister

Experience using Rucio / Looking ahead

- **X.509 certificates are painful**, looking forward to **token-based authentication** and authorisation
- In-depth look at the Kibana **dashboards and monitoring**, and what they can provide
- **WMS integration**
  - DIRAC with Rucio for a full end to end use case
  - Event-driven data management/processing
- **More endpoints** including Australian storage
- Participation in **ESCAPE H2020**
  - European Science Cluster for Astronomy and Particle Physics ESFRI research infrastructures
  - Rucio is the primary candidate for the data management and orchestration service



Thanks to Eli Chadwick, Ian Johnson

What did we learn?

The recurring topics and themes

- Appreciation
  - **Easy to integrate** into existing infrastructure and software
  - **Automation** of dataflows
  - Detailed **monitoring**
  - Easy to **contribute** code/extensions

- Feedback for improvements
  - *"Installation is only easy when you've done it before."* → Containerisation & K8s
  - *"Configuration relies on too many ambiguous things."* → Redesign of configuration
  - *"Documentation is too dispersed and out-of-date."* → Documentation generation; Establish community knowledge base

Addressed by

Community-driven development

- We have successfully moved to **community-driven development**
  - Requirements, features, issues, release are **publicly discussed** (e.g., weekly meetings, GitHub, Slack)
  - The core team is usually only **providing guidance** for architecture/design/tests
  - Usually 1-2 persons from a **community then take responsibility** to **develop** the software extension and also its **continued maintenance**

- Communities are helping each other **across experiments**
  - Effective across time zones due to US involvement
  - Automation and containerisation of development **lowers barrier of entry** for newcomers
  - Core team then only takes care about the management and packaging of the releases

- **Dedicated talks** about selected ongoing developments
  - Third-party-copy, Data carousel, Quality of Service, Token-based authn/z, SDN and Networks, …

Summary

- Several experiments and communities went from evaluation to production
  - AMS and Xenon as early adopters
  - Adoption by CMS was a decisive moment
  - Strong US and UK participation for support, development, and deployment
  - Successful integrations with existing software and computing infrastructures
- Emerging strong cooperation between HEP and multiple other fields
  - Notably neutrino and astronomy, with growing interest from more diverse sciences
- Community-driven innovations to enlarge functionality and address common needs
- Rucio is developing into a common standard for scientific data management
  - A successful collaborative open source project

Thank you!

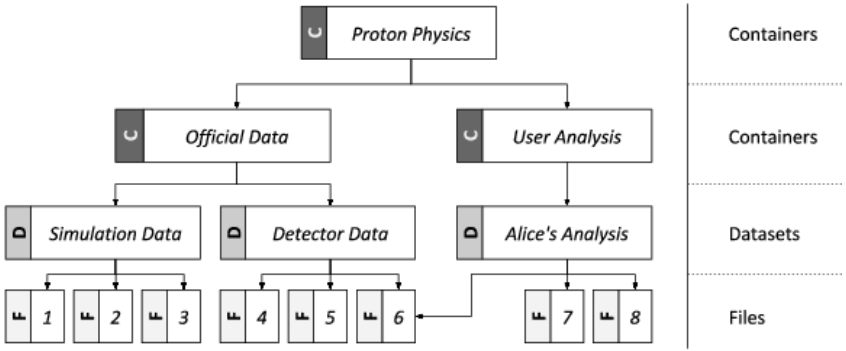
Website	http://rucio.cern.ch
Documentation	https://rucio.readthedocs.io
Repository	https://github.com/rucio/
Images	https://hub.docker.com/r/rucio/
Online support	https://rucio.slack.com/messages/#support/
Developer contact	rucio-dev@cern.ch
Journal article	https://doi.org/10.1007/s41781-019-0026-3
Twitter	https://twitter.com/RucioData

Backup

Rucio concepts - Namespace

- All data stored in Rucio is identified by a **D**ata **ID**entifier (DID)

- There are different types of DIDs
  - Files
  - Datasets Collection of files
  - Container Collection of dataset and/or container
- Each DID is uniquely identified and composed of a scope and name, e.g.:




```
detector_raw.run34:observation_123.root
```

scope name

Rucio concepts - Declarative data management

- Express what you want, not how you want it
  - *e.g., "Three copies of this dataset, distributed evenly across multiple continents, with at least one copy on TAPE"*
- Replication rules
  - Rules can be dynamically added and removed by all users, some pending authorisation
  - Evaluation engine resolves all rules and tries to satisfy them by requesting transfers and deletions
  - Lock data against deletion in particular places for a given lifetime
  - Primary replicas have indefinite lifetime rules
  - Cached replicas are dynamically created replicas based on traced usage and popularity
  - Workflow system can drive rules automatically, e.g., job to data flows or vice-versa
- Subscriptions
  - Automatically generate rules for newly registered data matching a set of filters or metadata
  - e.g., *project=data17_13TeV* and *data_type=AOD* uniformly across *T1s*

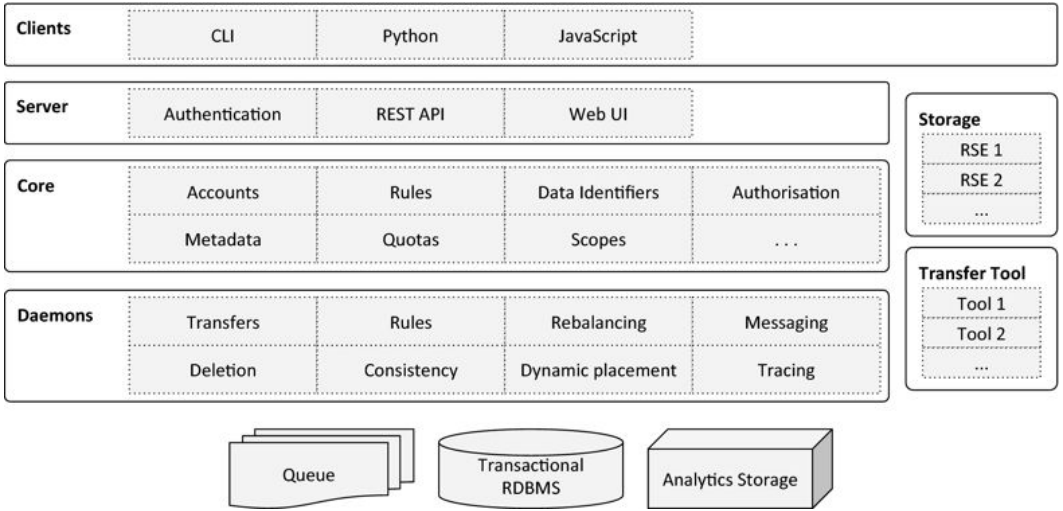
Rucio concepts - RSEs

- Rucio Storage Elements (RSEs) are logical entities of space
  - No software needed to run at the facility except the storage system, e.g., EOS/dCache/S3, …
  - RSE names are arbitrary, e.g., "CERN-PROD_DATADISK", "AWS_REGION_USEAST", …
  - Common approach is one RSE per storage class at the site
- RSEs collect all necessary metadata for a storage system
  - Protocols, hostnames, ports, prefixes, paths, implementations, …
  - Data access priorities can be set, e.g., to prefer a different protocol for LAN-only access
- RSEs can be assigned metadata as well
  - Key/Value pairs, e.g., *country=UK*, *type=TAPE*, *is_cached=False, …*
  - You can use RSE expressions to describe a list of RSEs, e.g. `country=FR&type=DISK` for the replication rules

Rucio concepts - Metadata

- Rucio supports different kinds of metadata
  - File internal metadata, e.g., *size, checksum, creation time, status*
  - Fixed physics metadata, e.g., *number of events, lumiblock, cross section, …*
  - Internal metadata necessary for the organisation of data, e.g., *replication factor, job-id,*
  - Generic metadata that can be set by the users
- Generic metadata can be restricted
  - Enforcement possible by types and schemas
  - Naming convention enforcement and automatic metadata extraction
- Provides additional namespace to organise the data
  - Searchable via name and metadata
  - Aggregation based on metadata searches
  - Can also be used for long-term reporting, e.g., evolution of particular metadata selection over time

Architecture



- **Servers**
  - HTTP REST/JSON APIs
  - Token-based security (x509, ssh, kerberos, …)
  - Horizontally scalable
- **Daemons**
  - Orchestrates the collaborative work e.g., transfers, deletion, recovery, policy
  - Horizontally scalable
- **Messaging**
  - STOMP / ActiveMQ-compatible
- **Persistence**
  - Object relational mapping
  - Oracle, PostgreSQL, MySQL/MariaDB, SQLite
- **Middleware**
  - Connects to well-established products, e.g., FTS3, DynaFed, dCache, EOS, S3, …
- **Python**
  - Support for Python2 and Python3

Operations model

- Objective was to minimise the amount of human intervention necessary
- Large-scale and repetitive operational tasks can be automated
  - Bulk migrating/deleting/rebalancing data across facilities at multiple institutions
  - Popularity driven replication and deletion based on data access patterns
  - Management of disk spaces and data lifetime
  - Identification of lost data and automatic consistency recovery
- Administrators at the sites are not operating any Rucio service
  - Sites only operate their storage exposed via protocols (POSIX, ROOT, HTTP, WebDAV, S3, gsiftp, …)
  - Users have transparent access to all data in a federated way
- Easy to deploy
  - PIP packages, Docker containers, Kubernetes

Monitoring & analytics

- RucioUI
  - Provides several views for different types of users
  - Normal users: Data discovery and details, transfer requests and monitoring
  - Site admins: Quota management and transfer approvals
  - Central administration: Account / Identity / Site management
- Monitoring
  - Internal system health monitoring with Graphite / Grafana
  - Transfer / Deletion / … monitoring built on HDFS, ElasticSearch, and Spark
  - Messaging with STOMP
- Analytics and accounting
  - e.g., Show which the data is used, where and how space is used, …
  - Data reports for long-term views
  - Built on Hadoop and Spark

Development

- Release cycle and support period
  - Bi-weekly patch releases (Bugfixes, minor enhancements)
  - ~3 feature (named) releases per year (Features, major changes)
  - Once a year a feature version is designated as a Long-Term Support (LTS) release
- Development organized as open-source community project
  - Weekly development meetings; Release roadmap for each feature release
  - Contributors describe their planned developments, receive comments from community
  - Extensive integration and unit testing across all supported databases