Rucio

An Overview

Martin Barisits
on behalf of the Rucio team

RUCIO
SCIENTIFIC DATA MANAGEMENT

ATLAS
EXPERIMENT

Rucio in a nutshell

- Initially developed by the High-Energy Physics experiment ATLAS
- Rucio provides a complete and generic scientific data management service
  - Data can be scientific observations, measurements, objects, events, images saved in files
  - Facilities can be distributed at multiple locations belonging to different administrative domains
  - Designed with more than 10 years of operational experience in large-scale data management!
- Rucio manages multi-location data in a heterogeneous distributed environment
  - Creation, location, transfer, and deletion of replicas of data
  - Orchestration according to both low-level and high-level driven data management policies (usage policies, access control, and data lifetime)
  - Interfaces with workflow management systems
  - Supports a rich set of advanced features, use cases, and requirements
- Rucio is open source and available under Apache 2.0 license

Community


eneas
Advanced European Network of E-infrastructures
for Astronomy with the SKA


LCLS


Belle II


Science & Technology
Facilities Council

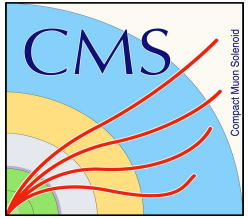

COMPASS


3D
EISCAT


ATLAS
EXPERIMENT


Alpha Magnetic Spectrometer
AMS-02
Europe • Asia
North America

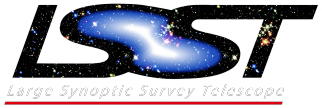

Xe
XENON
Dark Matter Project


CMS
Compact Muon Solenoid


egi


ARC


cta
cherenkov telescope array


LSST
Large Synoptic Survey Telescope


IceCube
SOUTH POLE NEUTRINO OBSERVATORY


FTS


eXtreme DataCloud


Fermilab


BROOKHAVEN
NATIONAL LABORATORY


DUNE
DEEP UNDERGROUND
NEUTRINO EXPERIMENT


LIGO

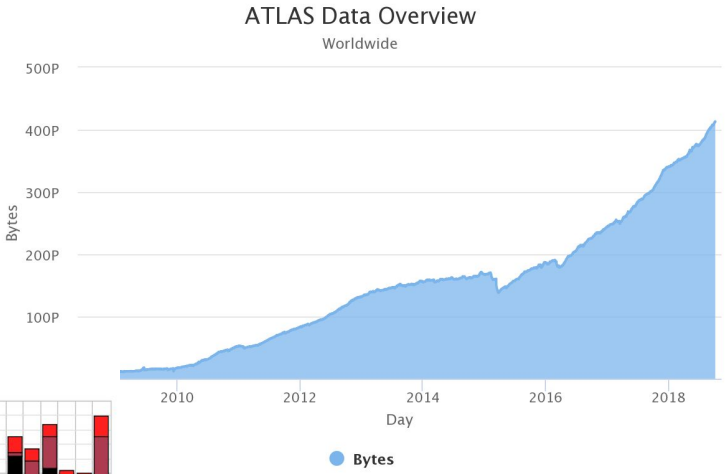

Open Science Grid

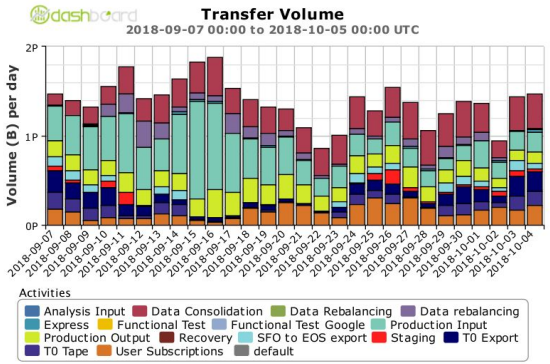
Community

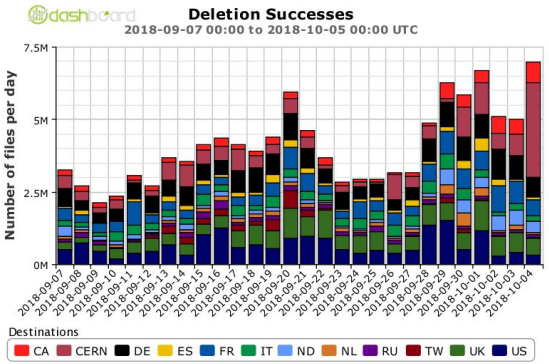
- 1st Rucio community workshop was held on March 1st-2nd 2018 to present Rucio to scientific communities
- Development on Github, Testing on Travis, Communication on Slack
- Weekly Development meeting
- 1st Rucio Coding Camp in November 2018
- 2nd Rucio community workshop

Data management at ATLAS

- ATLAS instance in a few numbers
  - More than 1B files, ~0.4 EB
  - Up to 4M files/2.5 PB transferred per day
  - More than 1000 active users
- Expect to gain one order of magnitude for Run4







Rucio main functionalities

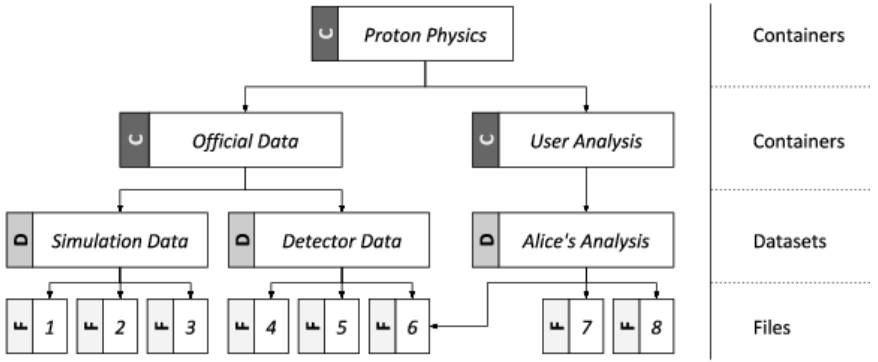
- Provides many features (Can be enabled selectively)
  - File and dataset catalog (logical definition and replicas)
  - Transfers between sites and staging capabilities
  - User Interface and Command Line Interface to discover/download/upload/transfer data
  - Extensive monitoring
  - Powerful policy engines (rules and subscriptions)
  - Bad file identification and recovery
  - Dataset popularity based replication
  - …

More advanced features

- Rucio can be integrated with Workload and Workflow Management System
  - Already supporting PanDA (ATLAS WFMS)
  - Possibilities of integration with other like Dirac

Rucio concepts - Namespace with DIDs

- All data stored in Rucio is identified by a Data IDentifier (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
  - Scope
  - Name
  - Example: `user.martin:test.file.001`



Rucio concepts - Metadata

- Rucio supports different kinds of metadata
  - System-defined, e.g., size, checksum, creation time, status
  - Physics, e.g., number of events, lumiblock
  - Production, e.g., which task or job produced the file
  - Data management internal: necessary for the organisation of data, e.g., replication factor
- Metadata are custom attributes on data identifiers
  - Enforcement possible by type, e.g., enum
  - 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)

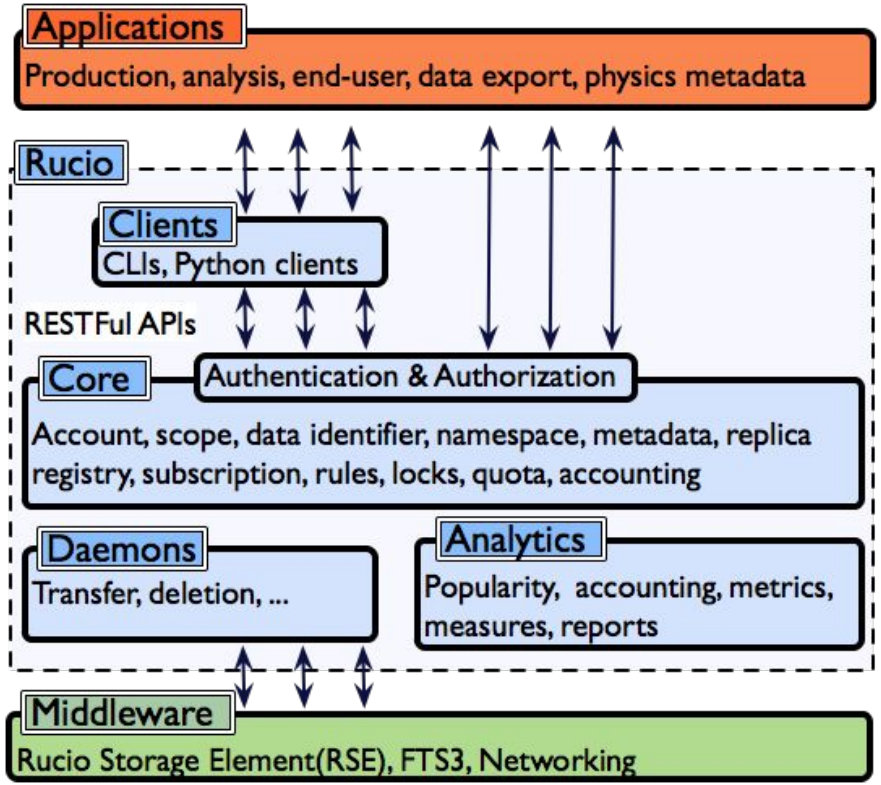
Rucio concepts - RSEs

- Rucio Storage Elements (RSEs) are logical entities of space
  - No software needed to run at the site
  - RSE names are arbitrary (e.g., "CERN-PROD_DATADISK", "AWS_REGION_USEAST", …)
  - Usually one RSE per site and storage data class
- 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 protocol for LAN access)
- RSEs can be assigned meta data
  - Key/Value pairs (e.g., *country=UK*, *type=TAPE*, *support=brian@unl.edu*)
  - You can use RSE expressions to describe a list of RSEs (e.g. `country=UK&type=TAPE`)

Rucio concepts - Declarative data management

- Express what you want, not how you want it
  - *e.g., "Three copies of this dataset, distributed evenly across two continents, with 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
  - Secondary replicas are dynamically created replicas based on traced usage and popularity
- Subscriptions
  - Automatically generate rules for newly registered data matching a set of filters or metadata
  - e.g., *project=data17_13TeV* and *data_type=AOD* evenly across *T1s*

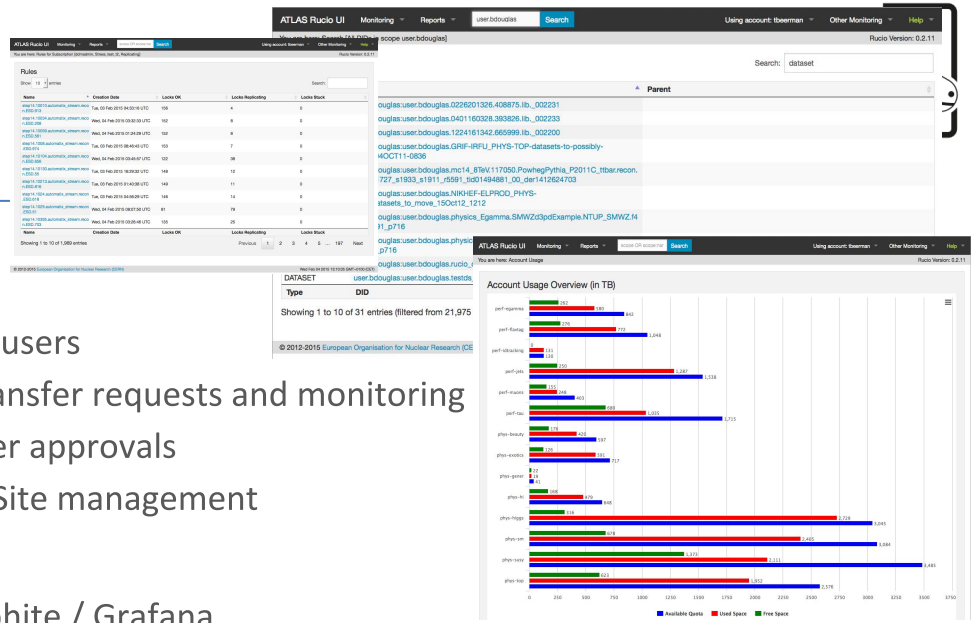
Architecture



Fully built on open standards and frameworks!

- **Servers**
  - HTTP REST/JSON APIs
  - Token-based authentication (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**
  - Clients: 2.6, 2.7, 3
  - Server: 2.7, 3

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

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 based on data access patterns
  - Popularity driven deletion
  - Management of disk spaces and data lifetime
  - Identification of lost data and automatic consistency recovery
- Administrators at the sites are not operating any local Rucio service
  - Sites only operate their storage
  - Users have transparent access to all data in a federated way
- Easy to deploy
  - PIP packages, Docker containers, Kubernetes

Future developments

- Generic & arbitrary metadata support
- Workload aware system components
  - Auto-scaling depending on load
- Multi-experiment data management features on shared infrastructures
- Quality of Service - Following the evolution of storage
  - Declarative Data Management based on QoS
- Expand support for commercial cloud providers
  - Transparent Google Cloud integration showed good results
- Capability-based authentication and authorisation
  - Bearer tokens, Sci-Tokens, Macaroons, OpenID, EduGain
- Event level data management
  - Include events and event metadata into Rucio - `rucio download <event>`

Rucio Community, March 2018

More information

Website	http://rucio.cern.ch
Documentation	https://rucio.readthedocs.io
Repository	https://github.com/rucio/
Continuous Integration	https://travis-ci.org/rucio/
Images	https://hub.docker.com/r/rucio/
Online support	https://rucio.slack.com/messages/#support/
Developer contact	rucio-dev@cern.ch