The data management system Rucio Evolution for LHC Run-3 and beyond ATLAS

Martin Barisits, Thomas Beermann, Joaquin Bogado, Vincent Garonne, Tomas Javurek, Mario Lassnig, Cedric Serfon, Tobias Wegner
on behalf of the ATLAS collaboration

martin.barisits@cern.ch
Rucio

● 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

● 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
  ○ Declarative/policy-based
  ○ Popularity driven replication and 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
Rucio

- **ATLAS**
  - Approaching 400PB
  - 10M containers, 20M datasets, 1B files
  - 5K accounts, 10K identities, 1K endpoints
  - 1-2PB transfers / day, 3PB deletions / day
  - 130 sites, 600 storage endpoints

- **ASGC: AMS + others**
  - Several million files
  - 10 sites

- **Xenon1T**
  - 5.6 PB
  - 100k files
  - 6 sites
Open source project

- Established as an open-source project
  - 29 contributors, ~10 active developers
  - 131k lines of code, 120 commits per month
  - Supports various database backends
- 1st Rucio Community Workshop
  - 90 participants
  - 16 communities
- Full-stack testing via Travis
- Focus on established open source tools

Contributions are always welcome!
Generic metadata support

- Primarily needed by non-ATLAS experiments using Rucio
- Rucio supports different types of metadata
  - System (size, checksums, status, creation time, …)
  - Physics (number of events, lumi block, …)
  - Production (task id, job id, …)
- Generic metadata support via arbitrary JSON encoded cells
  - Supported by PostgreSQL, MySQL, and Oracle
    - SQLAlchemy ready for PostgreSQL and MySQL
  - Currently hosting Google Summer of Code (GSoC) student to implement a prototype
- Objective
  - Fully flexible metadata
  - Multi-billion cell search performance
  - Combined lookup queries
Workload aware service components

● Full workload aware system components
  ○ Based on the backlog and priority, service components increase their instances and split the workload accordingly
  ○ Provide a constant level of service performance
  ○ React to priority tasks/campaigns with the needed amount of resources

● Auto-scaling needed for all service components
  ○ Already the case for some (rules, messages, subscriptions) but needed for others (deletion, transfers)

● Load analyzing orchestrator
  ○ Decides to start/kill service instances based on workload and priority
  ○ Possibly a Kubernetes Controller
Deployment and packaging

● Python versions
  ○ Clients: 2.6, 2.7, 3.5  
    2.6 will be dropped end of 2018
  ○ Server: 2.7  
    3.5 coming soon
  ○ 3.6 compatibility planned

● Packaging
  ○ PIP: Will keep providing general pip package as well as one for clients and webui
  ○ Containers: Will keep providing different docker containers on dockerhub

● Kubernetes
  ○ Objective: Turnkey deployment of Rucio
  ○ Currently testing a Rucio kubernetes cluster deployment for server-only (no daemons)
  ○ Plan is to offer a full-stack service deployment, including all daemons and load-balancing provided by Kubernetes
Future ATLAS data flow

- Grid Sites
- HPC Centres
- Cloud Storage
- Federated Sites
- Caches
- Custodial Data
- Data Provisioning
- Physics Metadata

Data flow managed by Rucio
Quality of service

- Representation of QoS attributes in Rucio
  - RSE Attributes
  - Latency, Throughput, Bring-Online delay, …

- Cost of Service
  - Also RSE Attributes
  - Storage, ingress, egress (Can be combined with rule lifetime and size)

- System needs to be flexible to generically support storage classes
  - Together with storage and network providers

- **Objective**: Rucio replication rule can effectively select a storage destination based on QoS and CoS claims
  - Enables Rucio to not only optimize on #transfers and storage volume but also on cost
Authentication/Authorisation

● Clear wish for more authentication/authorisation methods
● Bearer tokens
  ○ Delegate authorisation decision back to experiment
  ○ Refresh/expire tokens on demand
● OAuth style workflows
  ○ OpenID
● SciTokens, Macaroons, Cloud Signatures
  ○ Support coming in various components
● For authentication, federated identity support with EduGAIN would be useful
Event-level DM & Increased usage of tape

● Ongoing work towards an "Event Streaming Service", cf. Nicolo Magini’s talk
  ○ Intelligently stream the input data to workers — marshal only the necessary data
  ○ Fully exploit backfill on HPC and idle CPU on regular Compute (Grid & Cloud)

● Representation of a physics event and association with a file
  ○ Research ongoing but different possibilities:
    ■ Extending the definition of a data identifier (file, dataset, container) to event
      ● All rucio commands can be executed with an event identifier
        ○ rucio download <event>
      ● Metadata (which?) on events
    ■ Fetching and syncing event information from external services

● Tapes: Due to disk limitations, more tape usage in “carousel” mode foreseen
  ○ Optimize tape workflows
  ○ Workflows need to be adapted for closer interaction with WM and production system
More information

Website  [http://rucio.cern.ch](http://rucio.cern.ch)

Documentation  [https://rucio.readthedocs.io](https://rucio.readthedocs.io)

Repository  [https://github.com/rucio/](https://github.com/rucio/)

Continuous Integration  [https://travis-ci.org/rucio/](https://travis-ci.org/rucio/)

Images  [https://hub.docker.com/r/rucio/](https://hub.docker.com/r/rucio/)

Online support  [https://rucio.slack.com/messages/#support/](https://rucio.slack.com/messages/#support/)

Developer contact  [rucio-dev@cern.ch](mailto:rucio-dev@cern.ch)