QoS in Rucio

(and ATLAS)

Martin.Barisits@cern.ch, Mario.Lassnig@cern.ch
on behalf of the Rucio team

RUCIO
SCIENTIFIC DATA MANAGEMENT

ATLAS
EXPERIMENT

ATLAS QoS activities

- Given a storage space constraint, primary ATLAS target is to keep the CPUs full
  - Can we still do that if we start to use "high latency"/"low throughput" storage to keep costs down?
- Conceptual Rucio developments
- Data Carousel
  - R&D to study feasibility of getting inputs from tape in multiple phases (e.g., ramp-up, tail effects)
    - Phase 1: Tape throughput test (frontend limiting factor, file dispersion problematic)
    - Phase 2: Integration & Orchestration efforts between WFMS, DDM, FTS, and storage
  - Reprocessing campaigns exposing further needs of adjustments (e.g., sliding window)
- MAS — Multilayer Automated Storage
  - Automatically put unused data at site to tape to reduce need to buy expensive disks
  - cf. Eric's talk later this afternoon
- Commercial cloud storage — Hot/Cold prototype project with Google
  - Use cloud storage as a cheap intermediary buffer between our tapes and disks

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 **global single namespace** 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*
- Open community-driven development process

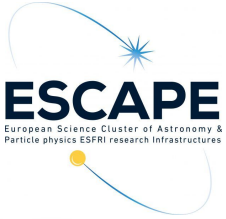
Rucio main functionalities

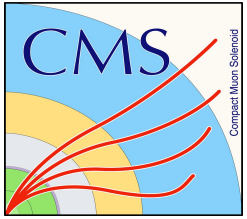
- Provides many features that can be enabled selectively
  - Horizontally scalable catalog for files, collections, and metadata
  - Transfers between facilities including disk, tapes, clouds, HPCs
  - Authentication and authorisation for users and groups
  - Web-UI, CLI, FUSE, and REST API
  - Extensive monitoring for all dataflows
  - Expressive policy engines with rules, subscriptions, and quotas
  - Automated corruption identification and recovery
  - Transparent support for caches and CDN dataflows
  - Data-analytics based flow control and SDNs
  - …

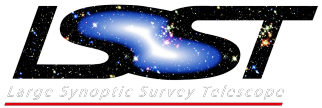
More advanced features

- Rucio is not a distributed filesystem, it connects existing storage infrastructure
  - No Rucio software needs to run at the data centres
  - Data centres are free to choose what suits them best, even within a single community

A growing community







3rd Rucio Community Workshop

- March 10-12, 2020
- Fermilab, USA
- https://indico.cern.ch/e/rucio2020/

Replica Management in Rucio

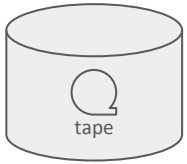
- Replica management in Rucio is based on replication rules
  - Put `1` copy of `file.001` on a Rucio Storage Element (RSE) in `country=uk&type=disk`
  - `country` and `type` are RSE specific attributes
  - Rule engine finds eligible RSEs and picks one based on a set of criteria
- `type=disk` already is an expression of QoS, but it is only used in limiting the set of eligible RSEs for the replication rule

Integration of Storage QoS in Rucio:

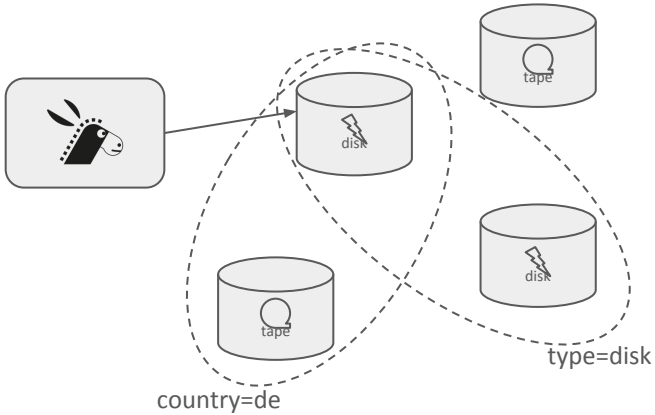
- QoS will become explicit part of the replication rule
- Beyond RSE selection, the QoS requirement (of the data) has to be communicated to storage if the storage system offers multiple QoS classes/zones
- Rule concept very well suited to express VO QoS policies (cf. whitepaper)

QoS with replication rules

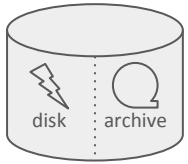
Today: 1 RSE = 1 QoS



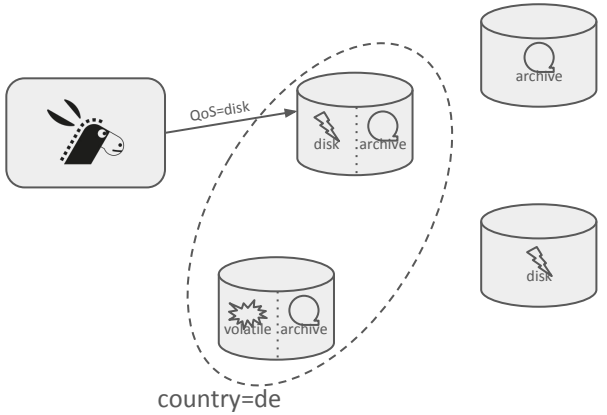
`1` copy of data on `country=de&type=disk`



Tomorrow: 1 RSE = 1 or many QoS



`1` copy of data on `country=de` QoS `disk`

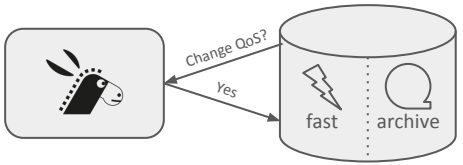


Classes and Properties

- QoS whitepaper foresees classes; definition of QoS classes with additional properties might add additional flexibility
- Classes
  - “`fast`”, “`custodial`”, “`cold`”, “`online`”, …
  - RSEs get tagged with classes (can be multiple, if multiple “zones” are supported)
  - Rule example: `1` copy of data on `country=de` QoS `disk`
- Properties
  - Fixed amount of properties: `latency`, `throughput`, `resilience`, `cost`, ?
  - RSE QoS classes get values assigned for each properties
  - Rule example: `1` copy of data on `country=de` QoS `latency<50&throughput>2000`
  - Property shortcut/class possible: e.g. `fast=latency<20&throughput>5000`
- Rucio will probably support properties for the QoS classes

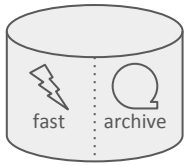
Bits & Pieces I

- How will QoS requirements be communicated to storage
  - XDC uses the CDMI standard
  - Storage protocol specific options foreseeable?
  - Encoded via hostname/namespace?
- Can storage independently move a file into different QoS classes?
  - Rucio needs to know the replica location/QoS of it's managed data
  - Storage could ask for permission to change QoS of data
  - Autonomous QoS changes (notification, no permission) might be difficult due to invalidation of replication rules (Perhaps possible in certain directions)



Bits & Pieces II

- Resilience
  - Rucio will **not** keep track of how many replicas a storage creates internally
  - This is reflected in the QoS class
- Mixing of classes
  - `archive` and `fast` possible?
  - If yes, do we need a concept of class conflicts?
  - `fast_and_archive`?



- QoS transitions will become a major workflow of the Rucio rule engine (next to transfer requests)

More information

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
Publications	https://rucio.cern.ch/publications.html
Twitter	https://twitter.com/RucioData