RUCIO STORAGE INTERFACES

ATLAS Data Management

ph-adp-ddm-lab@cern.ch

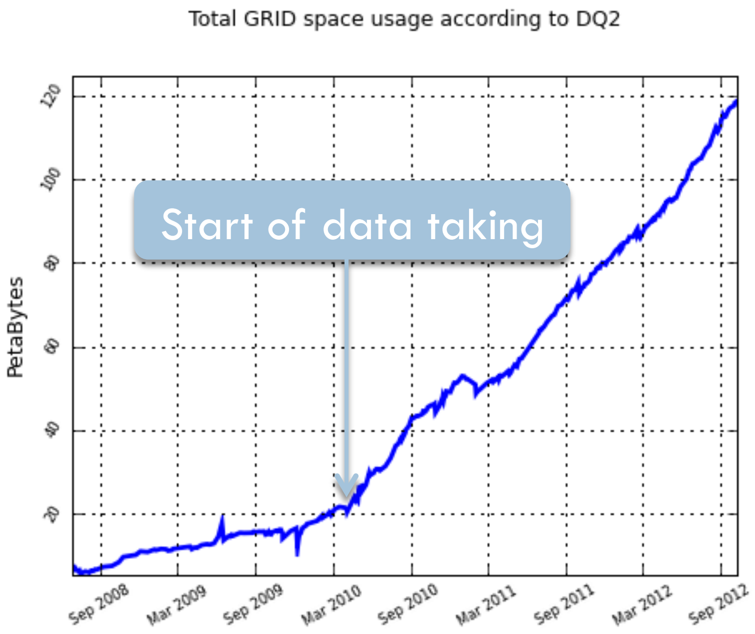
GDB Meeting, Annecy, 2012-10-09

ATLAS Data Management: DQ2

- Manages files and datasets
  - Bookmarking and reporting
  - Interaction with WLCG storage and transfer systems

- Current system: Don Quijote 2 (DQ2)
  - 120 Petabytes
  - 500k datasets with 350 million files
  - 800 active users
  - 130 sites with 700 endpoints

- DQ2 successful, but
  - Operational burden is high
    - Manpower required to keep system smooth
  - Component interaction is complex & complicated
    - Not easily scalable
  - Difficult to extend
    - Adding new features or technologies infeasible (originally designed for SRM on top of FTS)
    - Has been “engineered” into a dead end over the years
    - Only HEP community support



Start of data taking

ATLAS Data Management: Rucio

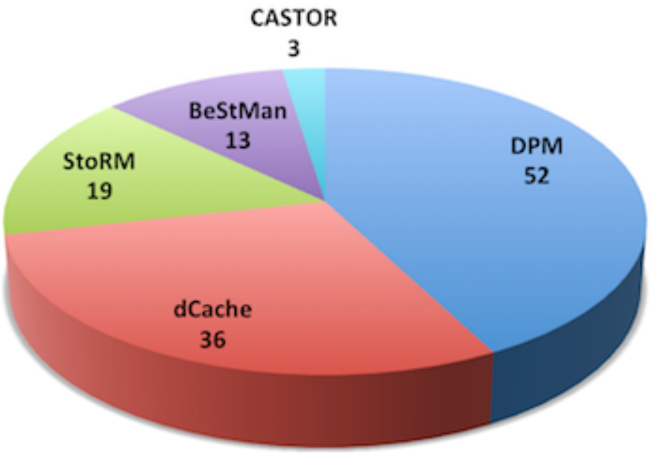
- Next-generation data management system
  - Ensure scalability and adaptability
  - Reduce operational overhead
  - Support new ATLAS use cases
  - Use free, open, and standard technologies
- Timeline
  - 2011
    - Technical meetings with other LHC experiments
    - User surveys
    - Collection of use cases
    - Rucio conceptual model
  - 2012
    - Parallel and incremental development (Early prototype in November)
  - 2013
    - Functional testing
    - Gradual migration from DQ2 to Rucio
    - Gradual migration of external applications (e.g., PanDA)
  - 2014
    - Rucio in production after LS1

Current SRM usage

Operation	Command	Component
Copy	lcg-cp	lcg_utils
SURL to TURL conversion	lcg-getturls	lcg_utils
Third party transfer	glite-transfer-submit/status	FTS2
Deletion	gfal_deletesurls	GFAL
Staging	gfal_prestage/prestagestatus	GFAL
Space collection	lcg_stmd	lcg_utils
Service discovery/Sanity check *	srmping, lcg-getturls	dcache-srmclient, lcg_utils
Consistency check *	srm-ls	lcg-utils

* manual operation

Alternative access protocols



- Free, open, and standard technologies

  (or: “The life after SRM, the way we see it”)

- `http/dav://, xroot://, s3://, gsiftp2://, file://`
  - Some federated/redirector protocols
- No information system about protocol+endpoint available
  - For example, EOS@CERN
    - `gsiftp://eosatlassftp.cern.ch`
    - `root://eosatlas.cern.ch`
    - `srm://srm-eosatlas.cern.ch`
  - Another example, NDGF
    - `https://fozzie.ndgf.org:2881`
    - `root://fozzie.ndgf.org:1095`
    - `srm://srm.ndgf.org:8443`
  - There are some sites that publish HTTP(S) in BDII (but the access failed :-)
- lcg-getturls returns specific, sometimes ephemeral, pools
  - Can’t automatically build an site access protocol catalogue from that
  - Some protocols are URI-redirection capable
- Right now, it’s regexps on SURLs and operator knowledge

ATLAS Storage Interfaces

- Two-dimensional approach
  - Need to support site view on files and the ATLAS view on sets of files
- Rucio Storage Element (RSE)
  - High-level abstraction
    - Single sites, and federation of sites
  - Deterministic mapping of files to replicas in a scoped namespace
    - Reduce external catalogue interaction
  - Interface with existing storage and transfer systems using standard protocols, or dedicated protocols if necessary
- Open and standard protocols
  - HTTP, WebDAV, metalink, NFS4.1, …
  - Allow storage systems to access the Rucio namespace via standard protocols (no need for Rucio clients)
    - Directory/File view is different than the ATLAS scoped dataset view
  - Prototype implementation with DMLite (rucio-plugin)
    - Will query Rucio in the background, so sites get an automatic ATLAS view if necessary

Third party transfer

- Integration with FTS3 foreseen by May 2013
  - Validation starting now (WLCG Ops Coordination: FTS3 task force)
- Full URI needs to be specified, if protocol != SRM
  - Problematic when you don't have a catalogue to look that up
- Clients requested fine-grained sharing
  - Currently First-Come-First-Serve
    - Needed: Request interleaving of different users within a share, and reordering based on size/time/some_metric
  - Will FTS3 deliver something like this?
    - If not, please tell us, then we have to do it in Rucio
- Quality of service guarantees?
  - Will FTS3 support source selection based on connectivity/uptime/some_metric of involved sites?
    - If not, please tell us, then we have to do it in Rucio

Other random things

- Space usage collection
  - Usually provided by SRM
  - Probe that gathers JSON files with storage info for a few gsiftp sites

    -
      ```
      cat /atlas/dq2/site-size
      {"sizes": {"total": 19996300279808, "available": 19655592639488},
      "time":"2012-03-06T15:10:01}
      ```
- Catalogue synchronisation
  - Can sites also publish their contents to a flatfile?
- Remote mass renaming
  - Not available in SRM, needed for DQ2-to-Rucio migration
  - Currently evaluating various options
    - dpns-rename/dav, gridftp-rename, xroot?, dCache?
- Data locality and federations
  - Publish information about federation content (file locations, caching, connectivity, …)
- Throttling
  - Sites need to be able to protect themselves
  - Storage systems should
    - abort overwhelming incoming requests quickly
    - reply with an estimate when to try again

Summary

- There are multiple alternative for all the features SRM provides (except the stage-in…)
  - … but I’m sure a solution can be found for this
  - Prototypical alternatives already in place
- Automatic site information updates are required
  - Absolutely essential (protocol, hostname, usage, …)
  - Many possibilities: catalogues, flat files, message queues, …
- Two dimensional approach
  - Manage data based on ATLAS requirements
  - Access/transfer data without special clients