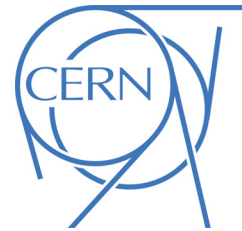**ATLAS NOTE**

9 October 2012
Version 2

Rucio: Conceptual Model

Vincent Garonne[1], Mario Lassnig[1], Angelos Molfetas[1], Martin Barisits[1], Thomas Beermann[1], Graeme A Stewart[1], Armin Nairz[1], Luc Goossens[1], Ralph Vigne[1], Cedric Serfon[1]

[1]*PH-ADP-CO, CERN*

Abstract

This document describes the conceptual model of the new version of the ATLAS Distributed Data Management (DDM) system: Rucio. Core concepts that Rucio uses to manage accounts, files and storage systems are introduced.

The DDM system is designed to allow the ATLAS collaboration to manage the large volumes of data, both taken by the detector as well as generated or derived, in the ATLAS distributed computing system. Any user of the system is mapped to an account in Rucio, which can represent an individual ATLAS user, physics group or central activity. Rucio supports permissions, accounting and quota on accounts. Rucio allows users to upload and register files in the system. Files may be grouped into datasets and datasets into containers. Rucio allows the setting of selected metadata properties on files, datasets and containers. The term data identifier set (DIS) is used to refer to any set consisting of files, datasets or containers. Replication rules can be set on DISs that instruct the system how to organise file replicas at sites. Rucio will trigger appropriate data replication to satisfy the current set of rules. Files are stored at physical locations that are managed as Rucio Storage Elements (RSEs). Rucio can group storage elements by setting RSE attributes that can be specified as part of a replication rule. As well as moving required files to them, Rucio will delete unnecessary files from storage elements.

1   Introduction

The new version of the ATLAS Distributed Data Management (DDM) system is called Rucio. In this document the key concepts of Rucio are introduced.

2   Rucio account

A Rucio account can represent individual users (e.g., lgoossen, graemes, vgaronne), a group of users (e.g., bphys, higgs, susy) or an organised production activity for the whole ATLAS collaboration (e.g., prod, tzero). A Rucio account is identified by a string and is the unit of assigning privileges in Rucio.

Actions in Rucio are always conducted by a Rucio account. Each account has a namespace identifier, called scope, that is included in every name assigned to a data identifier created by that account (see §3.1). By default, Rucio accounts can only create identifiers in their own scope and not in any other.

A Rucio user is identified by their credentials, such as X509 certificate or Kerberos token. Credentials can map to one or more accounts (N:M mapping). Rucio checks if the credentials used are authorized to use the supplied Rucio account. Figure 1 gives an example of the mapping between credentials and Rucio accounts:
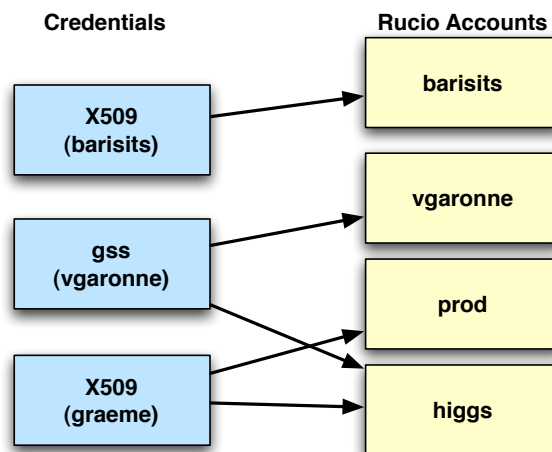

Figure 1: Credential to Rucio account mapping

3   Files, Datasets and Containers

ATLAS has a large amount of data, which is physically stored in files. For the data management system these files are the smallest operational unit of data (so sub-file operations are not possible). Physicists need to be able to identify and operate on any arbitrary set of files.

Files can be grouped into datasets (a named set of files) and datasets can be grouped into containers (a named set of datasets or, recursively, containers). All three types of names refer to data so the term data identifier set (DIS) is used to mean any set of file, dataset or container identifiers. A data identifier is just the name of a single file, dataset or container. The figure below gives an example of an aggregation hierarchy:

An example of a data identifier set (DIS) is: {prod:Dataset1, prod:File3, prod:File5}.
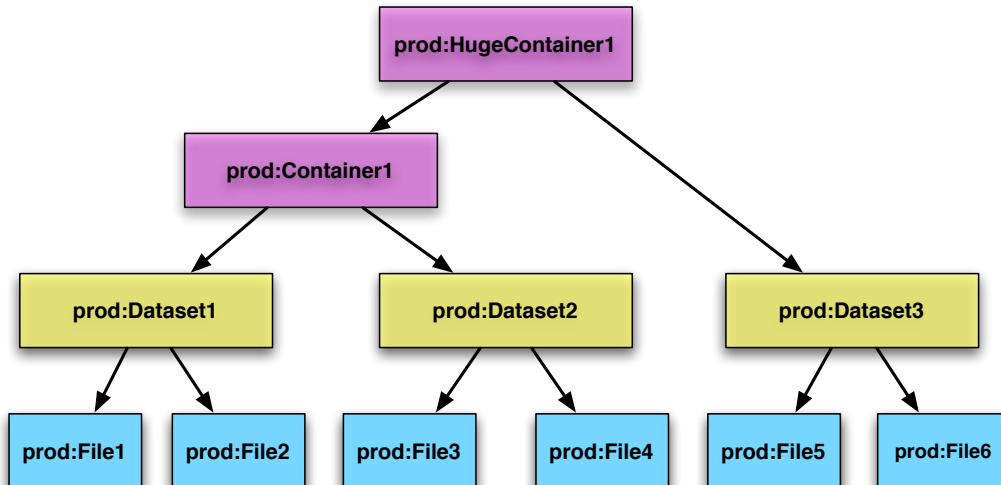
Figure 2: Rucio aggregation hierarchy

3.1 Data identifiers and scope

Files, datasets and containers follow an identical naming scheme which is composed of two strings: the scope and a name. The combination of both is called a data identifier (DI). For instance a file identifier (LFN) is composed of a scope and a file name. The scope string partitions the namespace in several sub spaces. The primary use case for this is to have separate scopes for production and individual users.

By default accounts will have read access to all scopes and write access only to their own scope. Privileged accounts will have write access to multiple scopes, e.g., production might use scopes such as mc11, data12_8TeV, tmp.prod.[1]

Files, datasets and containers are uniquely identified over all time. This implies that an identifier, once used, can never be reused to refer to anything else at all, not even if the data it referred to has been deleted from the system.

3.2 File, dataset and container status

3.2.1 File status

The following status attributes are supported for files:

- `availability`: LOST, DELETED, AVAILABLE

  A file is LOST if there are no known replicas of the file in Rucio, while at the same time at least one account requested a replica; a file is DELETED if no account requested a replica; otherwise the file is AVAILABLE. This is a derived attribute.

- `suppressed`: TRUE, FALSE

  This is a user settable flag. It indicates that the owner of the scope no longer needs the name to be present in the scope. Files that are suppressed (by default) do not show up in search and list operations on the scope. The setting of this flag is subject to conditions, e.g., one can not suppress

---

[1]Scopes can be marked as read-only, which will prevent the registration of new data in them.

a file while at the same time requesting it to be replicated somewhere. This flag will be ignored when explicitly listing the contents of datasets/containers.

3.2.2 Dataset/Container status

The dataset/container status is reflected by a set of attributes:

- `open`: TRUE, FALSE

  If a dataset/container is open, content can be added to it. Datasets/containers are created open and once closed, they cannot be opened again.[2]

- `monotonic`: TRUE, FALSE

  If the monotonic attribute is set, content cannot be removed from an open dataset/container. Datasets/containers are, by default, created non-monotonic. Once set to monotonic, this cannot be reversed.

- `complete`: TRUE, FALSE

  A dataset/container where all files have replicas available is complete. Any dataset/container which contains files without replicas is incomplete. This is a derived attribute.

- `suppressed`: TRUE, FALSE

  This attribute has the same meaning as for files.

There is no concept of versioning. Adding content to a closed dataset/container is not possible and instead a new dataset/container, with a new identifier, must be created.

4 Metadata attributes

Metadata associated with a data identifier is represented using attributes, which are key-value pairs. The set of available attributes is restricted. Some metadata attributes are user settable, e.g., physics attributes (number of events, run number, run period, POOL GUID) or production attributes (task ID, job ID). Metadata which is not user settable includes system attributes, such as size, checksum, creation time. For datasets and containers, it is possible that the value of a metadata attribute is a function of the metadata of its constituents, e.g., the total size is the sum of the sizes of the constituents. In this case it is also obviously not possible to assign a value to it.

When appropriate and requested, Rucio will check metadata values for validity, rejecting the attempt to set invalid values. This can be used to ensure that, e.g., POOL GUID is unique and that the Job ID is a positive integer.

The upload of data with incorrect attributes or values will be rejected. Rucio supports searching for files, datasets and containers based on metadata attributes.

5 Rucio Storage Element

A Rucio Storage Element (RSE) is a repository for physical files. It is the smallest unit of storage space addressable within Rucio. It has an unique identifier and a set of properties such as:

---

[2]Datasets from which files have been lost can be repaired when replacement files are available, even if open=FALSE. The replacements need not be binary identical to the lost files.

- supported protocols, e.g., file, https, srm

- quality of service; storage type, e.g., disk, tape

- physical space properties, e.g., used, available, non-pledged

- a weight value, used for data distribution (see §7.1)

- a threshold for deletion (see §7.4)

A set of RSEs can be identified directly by enumeration of their names, or indirectly by a boolean expression over their attributes. Attributes are key-value pairs, e.g., CLOUD=UK, Tier=1, T2D=True, MoUShare=15. A key whose value is True is equivalent to a tag. RSE attributes are used to manage data with replication rules (see §7.1).

Physical files stored on RSEs are identified by their Physical File Name (PFN). The PFN is a fully qualified path identifying a replica of a file. PFNs may take the form of file names, URIs, or any other identifier meaningful to a Rucio Storage Element. The mapping between the LFN and the PFN is a deterministic function of the LFN, RSE and protocol.

Normally the upload to an RSE and the registration of a replica is a single operation. For trusted users, like the Tier-0 and PanDA production systems, it is possible to register a replica uploaded independently.

6 Permission model

Rucio assigns permissions to accounts. Permissions are boolean flags designating whether an account may perform a certain action. For example, the check of the permission to add an account for the rucio administrative account will return true.

7 Replica management

7.1 Replica Rules

Replica management is based on replication rules defined on data identifier sets. Replication rules defined on datasets or containers will affect all contained datasets, containers and files (either directly or recursively), including content added to these datasets and containers in the future. A replication rule is owned by an account and defines the minimum number of replicas to be available on a list of RSEs.

Rules may have a limited lifetime and can be added, removed or modified at any time. Rules may specify a grouping policy that controls which files are grouped together at the same RSEs (see §7.1).

The list of RSEs can be specified with RSE attributes (see §5). RSE attributes are resolved at rule creation time to enumerate target RSEs. Post-facto changes to RSE attributes will not affect current replication rules.

Sample replication rules are given below (some parameters are optional, a value in [] denotes usage of the default value):

Account	Copies	Data Identifier Set	RSEs	Weight	Lifetime	Grouping[a]
prod	2	prod:container1	Tier=1	[MoU]	[No]	[Dataset]
prod	1	atlas:container3	Site=CERN	[MoU]	[No]	[Dataset]
prod	1	atlas:container3, atlas:dataset6	Tier=2 AND Cloud=DE	[MoU]	[No]	All
mario	1	mario:dataset1, mario:dataset2	(Cloud=US AND Tier=2) OR Site=CERN	[MoU]	2013-01-01	[Dataset]
luc	2	graeme:file1, graeme:container2	Cloud=UK AND T2D[b]	No	[No]	None

[a]See §7.2

[b]This is interpreted as T2D=True.

As transferring a file to a site can fail, a replication rule has a state to reflect its status:

`satisfied` – rule is fulfilled by current replicas

`replicating` – not all replicas are yet made for this rule

`stuck` – replication attempts for at least one replica have failed and have reached an attempt limit

`suspended` – a manually set status, indicating that no further attempts should be made to satisfy the rule

Rucio processes each replication rule and creates replica locks that satisfy the rule and prevent deletion of the replicas. A replica lock is defined for a replica on a specific RSE. It is owned by the account which issued the replication rule. Rucio will record the relationship between the replication rules and replica locks so that the correct locks can be removed when a replication rule is deleted.

A replication rule can generate a data transfer prior to the replica lock being satisfied. Rucio will only create the minimum set of necessary transfer primitives to satisfy all rules. If the weighting option of the replication rule is used, the choice of RSEs takes their weight into account.

For example, `vgaronne:Dataset1 = {File1, File2}` with the replication rule `'vgaronne: 2 replicas @ Tier=2'` is resolved into two replica locks for each file of the dataset:

`vgaronne: File1 @ LMU`

`vgaronne: File1 @ MANCHESTER`

`vgaronne: File2 @ LMU`

`vgaronne: File2 @ MANCHESTER`

If Rucio encounters failures when transferring files, it will apply a configurable policy to determine how to proceed. This policy can specify a retry strategy, whether alternative sites can be used and when the rule will be considered stuck.

7.2 File replica grouping

If the file content of a dataset is very widely distributed across many sites this will maximise the CPU resources that can be used to process this data. If the files are concentrated at one site this will be optimal for processing which requires the use of many files at once (e.g., merging and archiving to tape).

Rucio uses the dataset/container hierarchy to control how files are grouped together at sites. Rucio offers three levels of grouping, which will affect how the file replicas in a hierarchy are grouped:

all – All files will be replicated to the same RSE.

dataset – All files in the same dataset will be replicated to the same RSE; different datasets will be spread maximally over all allowed RSEs. This is the default.

none – Files will be completely spread over all allowed RSEs without any grouping considerations at all.

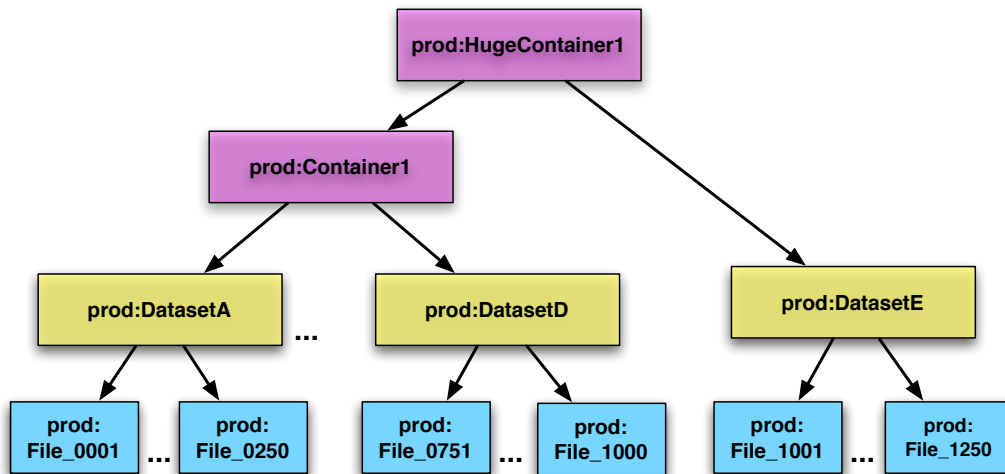To illustrate this consider the following dataset hierarchy:

Figure 3: Rucio dataset hierarchy with grouping

There are 1250 files organised in 5 datasets:

    prod:DatasetA = {prod:File_0001, ..., File_0250}

    prod:DatasetB = {prod:File_0251, ..., File_0500}

    prod:DatasetC = {prod:File_0501, ..., File_0750}

    prod:DatasetD = {prod:File_0751, ..., File_1000}

    prod:DatasetE = {prod:File_1001, ..., File_1250}

Some datasets are aggregated into a container:

    prod:Container1 = {prod:DatasetA, prod:DatasetB, prod:DatasetC, prod:DatasetD}

The container prod:Container1 and dataset prod::DatasetE are aggregated into a bigger container:

    prod:HugeContainer1 = {prod:Container1, prod:DatasetE}

If a replication rule is specified on `prod:HugeContainer1`, specifying that `all` grouping should be used, then Rucio will replicate all files belonging to `prod:HugeContainer1` (`prod:File_0001 ...File_1250`) to the same RSE.

If a replication rule with `dataset` grouping is specified on `prod:HugeContainer1`, then Rucio will spread the datasets as much as possible over the allowed RSEs, but keep all the files in each dataset at the same RSE.[3]

If a replication rule with `none` grouping is specified on `prod:HugeContainer1`, then Rucio will replicate all files belonging to the datasets to individually random selected RSEs.

Intermediate grouping options, e.g., by specifying a replication rule on `prod:HugeContainer1`, but only grouping all of the content of `prod:Container1` together are not supported. To achieve this an individual replication rule has to be specified on `prod:Container1` with `all` grouping.

Clients should ensure that their dataset/container hierarchies, in particular the definition of the datasets, are created in such a way as to provide suitable file groupings for further data processing.

7.3 Subscription

Subscriptions generate replication rules based on matching particular metadata at registration time. Subscriptions are owned by an account and can only generate rules for that account. Subscriptions do not have a lifetime and need to be removed when no longer required. The lifetime of the generated replication rules can be expressed in a relative way. An example of a subscription is given below:

Attribute	Value
Owner	tzero
Match	project=data12_8TeV AND dataType=RAW AND stream=physics_* AND DIType=dataset
Rule	1@Site=CERNTAPE, 1@T1TAPE, 1@T1DISK until NOW+90 days; grouping=dataset

7.4 Data deletion

In case a replica on a particular RSE has no associated replica locks anymore it can be deleted. Locks disappear when users decrease the number of desired replicas in their replica rules, or the whole rule altogether.

8 Accounting and quota

Accounting is the measure of how much resource, e.g., storage, an account has used as a consequence of its actions. Quota is a policy limit which the system applies to an account for some resource.

For storage accounting, Rucio accounts will only be charged for the files on which they have set replication rules. The accounting is based on the number of replicas an account requested, not on the number of physical replicas in the system. Accounting and quota calculations use the replica locks generated from replication rules. Quotas are constraints on the replica locks, based on limits set per account.

For example, the GROUPDISK RSE tag could be translated into the following RSE list: CERN_GROUPDISK, BNL_GROUPDISK, GLASGOW_GROUPDISK.

---

[3]Assuming that multiple RSEs are allowed by this rule - specifying a single RSE obviously negates grouping

Account	RSE list	Limit (TB)
user.vgaronne	CERN_USERDISK	< 1
group.phys-higgs	GROUPDISK AND NOT GLASGOW_GROUPDISK	< 30
group.phys-higgs	CERN_GROUPDISK	< 20
group.phys-higgs	BNL_GROUPDISK	< 20
group.phys-higgs	GLASGOW_GROUPDISK	< 100

The account user.vgaronne has a quota of 1TB at CERN_USERDISK. The account group.phys-higgs has an overall quota of 30TB on all GROUPDISK RSEs, excluding GLASGOW. CERN and BNL have an additional protection that group.phys-higgs cannot use more than 20TB on each site, despite the global quota. GLASGOW provides explicitly a quota of 100TB and is excluded from the GROUPDISK quota.

9 Notifications

External applications can require to synchronize on events relative to data availability and can subscribe to particular events, e.g., dataset state changes. Then Rucio will publish a messages to an external application when it detects these events.

Acknowledgments

Appendix A. Key concepts: Comparison matrix DQ2 vs. Rucio

Features	DQ2	Rucio
File identifier	GUID/LFN	Scope + name (DI)
Dataset identifier	DUID/DSN	Scope + name (DI)
Container identifier	CUID/CNT	Scope + name (DI)
Versioning	Yes	No
Namespace	Global/Flat	Scoped
File aggregation	Dataset	Dataset
Dataset aggregation	Container	Container
Container aggregation	No	Container
Unique PFN	No	Yes
Overlapping dataset	Yes	Yes
Storage Element Groups	No	Yes
Storage Element Attributes	No	Yes
Quota support	Group	Account (Group/User)
Replica grouping	Strict	Best effort
Replica Lifetime	Yes	No
Metadata namespace and support	System defined, Data placement	System defined, Physics, Production, Analysis, Data placement
Data discovery unit	Wildcard string matching	Metadata, dataset/container hierarchy
Data operation unit	Dataset	Dataset, File, Container (DIS)
Multiple replica ownership	No	Yes
Dynamic placement	No	Yes
Replication rule support	No	Yes
Hidden data	Yes	Yes
Reuse of dataset name	No	No (but possibility to resuscitate dataset)
Notifications	Yes	Yes
Fine-grained accounting	Partially	Yes

Appendix B. Acronyms and Abbreviations

LFN Logical File Name.

DI Data Identifier.

DIS Data Identifier Set.

PFN Physical File Name.

RSE Rucio Storage Element.

URI Uniform Resource Identifier.