Introduction to iRODS

Jean-Yves Nief

Talk overview

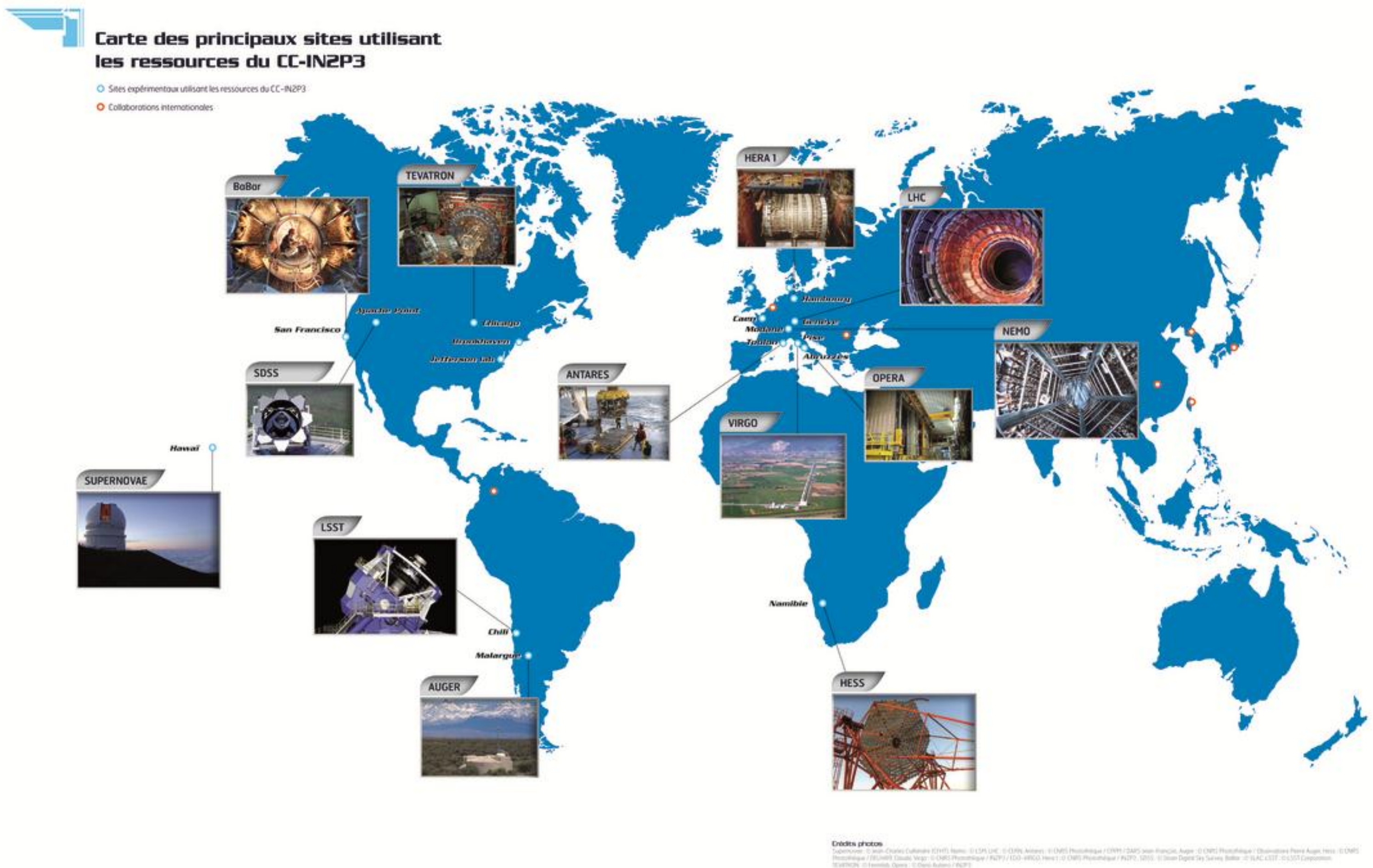
- Data management context.
- Some data management goals:
  - Storage virtualization.
  - Virtualization of the data management policy.
- Examples of data management rules.
- Why choose iRODS ?
- What is iRODS ?
- iRODS usage examples.
- Propects: scalability.
- Summary.

CC-IN2P3 activities

- Data centers like CC-IN2P3 (Lyon, France) works for international scientific collaborations.
- Examples:
  - High Energy Physics: CERN (Fr/Switzeland), SLAC (USA), Fermilab (USA), BNL (USA) etc…
  - Astroparticle physics / astrophysics: Auger (Argentina), HESS (Namibia), AMS (Int. Space Station).

➔ ***Distributed environment: experimental sites, data centers, collaborators spread around the world.***

CC-IN2P3 activities



Data management context

- Multidisciplinary environment:
  - – High Energy and nuclear physics.
  - – Astroparticle, astrophysics.
  - – Biology, Biomedical apps, Arts and Humanities.

➔ ***Various constraints, various needs for data management.***

Data management context

- Data stored on various sites.
- Heterogeneous storage:
  - Data format: flat files, databases, data streams…
  - Storage media, server hardware: disks , tapes.
  - Data access protocols, information systems.
- Heterogeneous OS on both clients and servers side.

➔ ***Needs to federate all this in a homogeneous way.***

Data management context

- *Not in the scope here:*
  - *Intensive parallel I/O for data analysis.*
- In the scope here:
  - Data preservation (replication, consistency …).
  - Data access distributed over different sites.
  - Data life cycle (file format transformation, data workflows, interactions with various info systems).
- Need for virtualization of the storage:
  - Logical view and organization of the data.
  - ➔ Data migration to new hardware/software transparent to the end clients tools: no view of the physical location of the data and underneath technologies.
  - ➔ Logical view of the data unique to all the users independently of their location.
  - Virtual organization (VO) of the users:
    - Unique id for each user.
    - Organization by groups, role (simple user, sysadmin etc…).
    - Access rights to the data within the VO.

Some data management goals

- Storage virtualization not enough.
- For client applications relying on these middlewares:
  - No safeguard.
  - No guarantee of a strict application of the data preservation policy.
- Real need for a data distribution project to define a coherent and homogeneous policy for:
  - data management.
  - storage resource management.

➔ Crucial for massive archival projects (digital libraries …).

➔ No grid/cloud/… tool had these features until 2006.

Virtualization of the data management policy

- Typical pitfalls:
  - No respect of given pre-established rules.
  - Several data management applications may co-exist at the same moment.
  - Several versions of the same application can be used within a project at the same time.

  ➔ *potential inconsistency.*
- Remove various constraints for various sites from the client applications.
- Solution:
  - Data management policy virtualization.
  - Policy expressed in terms of rules.

Examples of data management rules

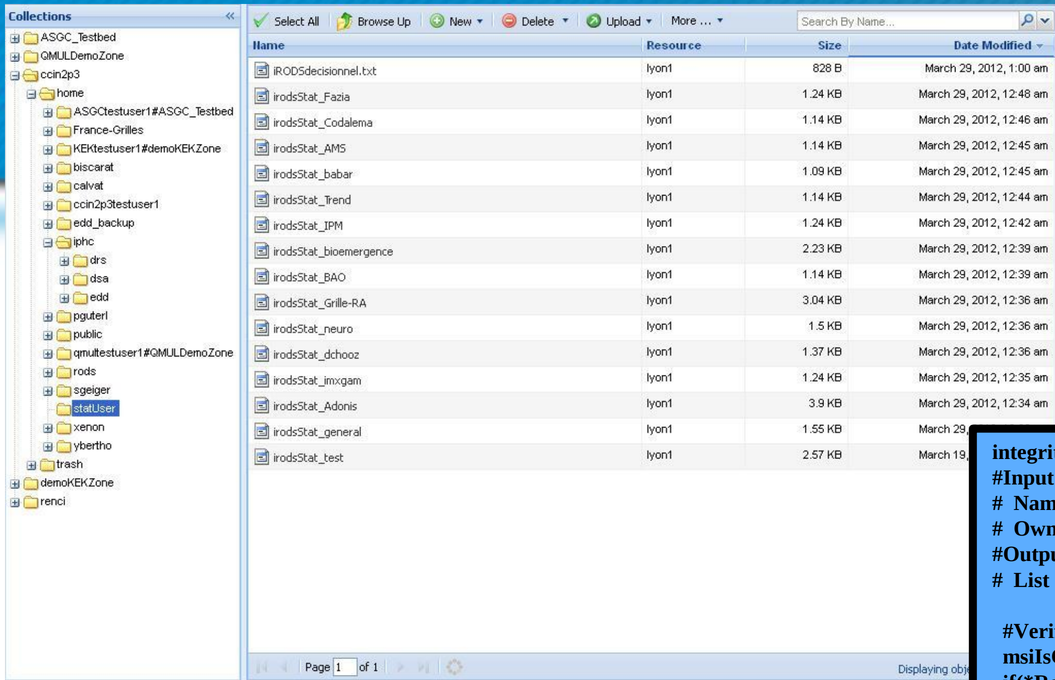
- Customized access rights to the system:
  - Disallow file removal from a particular directory even by the owner.
- Security and integrity check of the data:
  - Automatic checksum launched in the background.
  - On the fly anonymization of the files even if it has not been made by the client.
- Metadata registration:
  - Automated metadata registration associated to objects (inside or outside the iRODS database).
- Small files aggregation before migration to MSS.
- Customized transfer parameters:
  - Number of streams, stream size, TCP window as a function of the client or server IP.
- … up to your needs …

Why did we choose iRODS ?

- Provide a solution to the above requirements.
- SRB (iRODS predecessor) has been used so far:
  - Data virtualization.
  - But no policy rule based mechanisms.
- In 2007, no « grid » tools except iRODS could provide data management policies based on rule.
- Scalable.
- Can be customized to fit a wide variety of use cases.

What is iRODS ?

- **iR**ule **O**riented **D**ata **S**ystems (DICE team: UNC, San Diego):
  - started in 2006.
  - open source.
  - CC-IN2P3: collaborator
- In a « zone » (administrative domain):
  - One or several several servers connected to a Centralized Metacatalog (RDBMS) with files metadata, user informations, data locations etc… ➔ Logical view of the data in a given ***zone***.
  - Data servers spread geographically within a zone.
- Possibility to have different ***zones*** (separate administrative domains) interconnected.
- Data management policies expressed with rules in a « C-like » language:
  - Can be triggered automatically for various actions (put, get, list, rename….).
  - Can be run manually.
  - Can be run in batch mode.
  - Rules versioning.
- Client interactions with iRODS:
  - APIs (C, Java, PHP, Python), shell commands, GUIs, web interfaces.



Logical name space (example of a web interface)

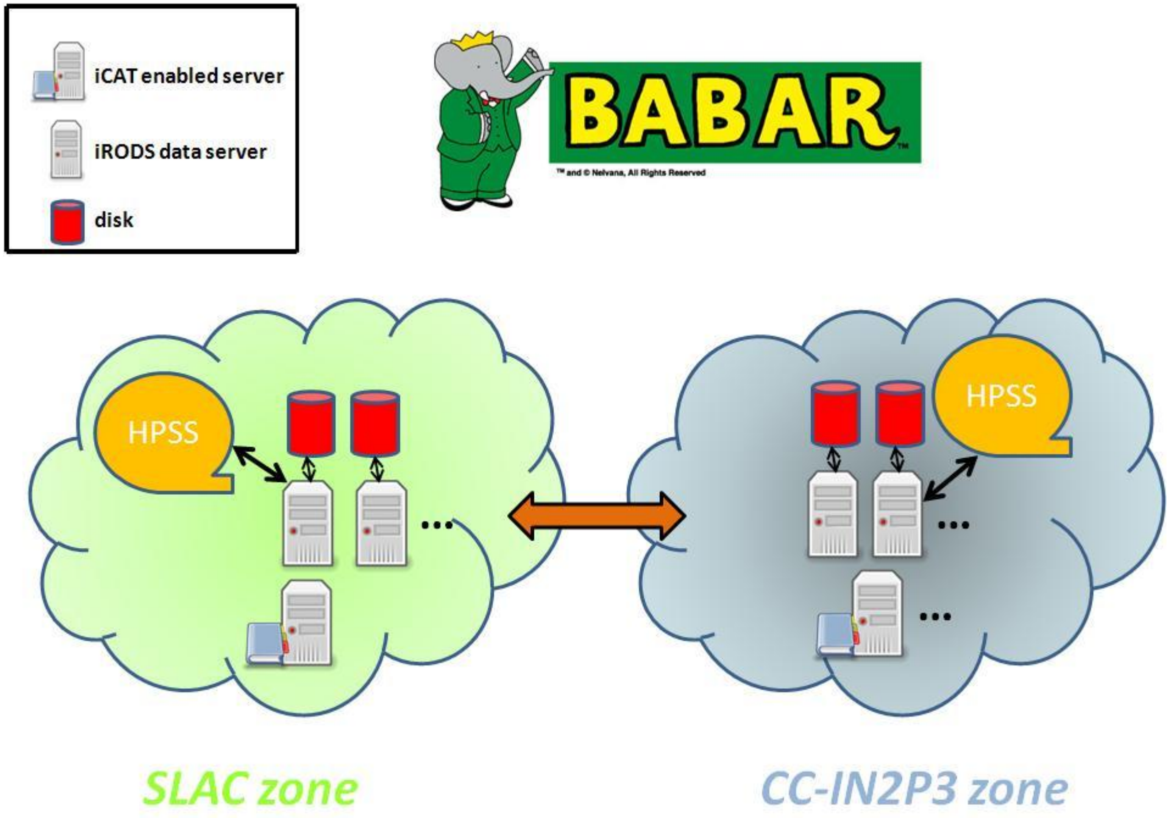
Rule example (e.g.: *Are all the files in a given collection all owned by a given user ?)*

```
integrityFileOwner {
#Input parameter is:
#  Name of collection that will be checked
#  Owner name that will be verified
#Output is:
#  List of all files in the collection that have a different owner

  #Verify that input path is a collection
  msiIsColl(*Coll,*Result, *Status);
  if(*Result == 0) {
   writeLine("stdout","Input path *Coll is not a collection");
   fail;
  }
  *ContInxOld = 1;
  *Count = 0;

  #Loop over files in the collection

msiMakeGenQuery("DATA_ID,DATA_NAME,DATA_OWNER_NAME","COLL_NA
ME = '*Coll'",*GenQInp);
  msiExecGenQuery(*GenQInp, *GenQOut);
  msiGetContInxFromGenQueryOut(*GenQOut,*ContInxNew);
  while(*ContInxOld > 0) {
   foreach(*GenQOut) {
    msiGetValByKey(*GenQOut,"DATA_OWNER_NAME",*Attrname);
    if(*Attrname != *Attr) {
     msiGetValByKey(*GenQOut,"DATA_NAME",*File);
     writeLine("stdout","File *File has owner *Attrname");
     *Count = *Count + 1;
    }
   }
   *ContInxOld = *ContInxNew;
   if(*ContInxOld > 0) {msiGetMoreRows(*GenQInp,*GenQOut,*ContInxNew);}
  }
  writeLine("stdout","Number of files in *Coll with owner other than *Attr is *Count");
}
INPUT *Coll = "/tempZone/home/rods/sub1", *Attr = "rods"
OUTPUT ruleExecOut
```

iRODS usage @ CC-IN2P3

- Used for data distribution, archiving, integration in analysis or data life cycle management workflows.
- Interfaced with:
  - Mass Storage Systems: HPSS.
  - External databases or information systems: RDBMS, Fedora Commons.
  - Web servers.
- **High energy and nuclear physics**:
  - BaBar: data management of the entire data set between SLAC and CC-IN2P3: total foreseen 2PBs.
  - dChooz: neutrino experiment (France, USA, Japan etc…): 550 TBs.
- **Astroparticle and astrophysics**:
  - AMS: cosmic ray experiment on the International Space Station (600 TBs).
  - TREND, BAOradio: radioastronomy (170 TBs).
- **Biology and biomedical applications** (50 TBs).
- **Arts and Humanities**: Adonis (70 TBs).

iRODS usage example



- archival in Lyon of the entire BaBar data set (total of 2 PBs).
- automatic transfer from tape to tape: 3-4 TBs/day (no limitation).
- automatic recovery of faulty transfers.
- ability for a SLAC admin to recover files directly from the CC-IN2P3 zone if data lost at SLAC.

Some rules examples (HEP)

- Mass Storage System integration:
  - Using compound resources: iRODS disk cache + tapes.
  - Data on disk cache replication into MSS asynchronously (1h later) using a delayExec rule.
  - Recovery mechanism: retries until success, delay between each retries is doubled at each round.
- ACL management:
  - Rules needed for fine granularity access rights management.
  - Eg:
    - 3 groups of users (**admins**, **experts**, **users**).
    - ACLs on /<zone-name>/*/rawdata => **admins** : *r/w*, **experts** + **users** : *r*
    - ACLs on all others subcollections => **admins** + **experts** : *r/w*, **users** : *r*

Prospects: scalability

- 5.5 PBs managed by iRODS so far.
- 26 millions files registered.

➔ No scalability issues foreseen.

- Pitfalls:
  - Metadata scalability ? (billions of entries in the catalog ?).
  - Control of the number of simultaneous connections to be enforced (like for Apache servers): needed in a wide opened environment.

Summary

- iRODS:
  - Lots of features for data management in a distributed environment.
  - Provides the flexibility and the freedom to interface or work directly with a non limited list of systems.
  - Not the same goal as a parallel file system.

Summary

- Some iRODS user examples:
  - USA: DataNet, NASA, NOAO, iPlant.
  - Canada: Virtual Observatory.
  - France: National Library, Strasbourg Observatory, Grenoble Campus, CINES etc…
  - Europe: EUDAT, Prace.
  - Private sector: DDN etc…
- Entreprise version ramping up (E-iRODS):
  - Wider source of fundings.

Learn more

- https://www.irods.org/index.php/Main_Page