EUDAT PRACE EGI
workshop

iRODS services and use cases

Giuseppe Fiameni, CINECA

Amsterdam, 27th November

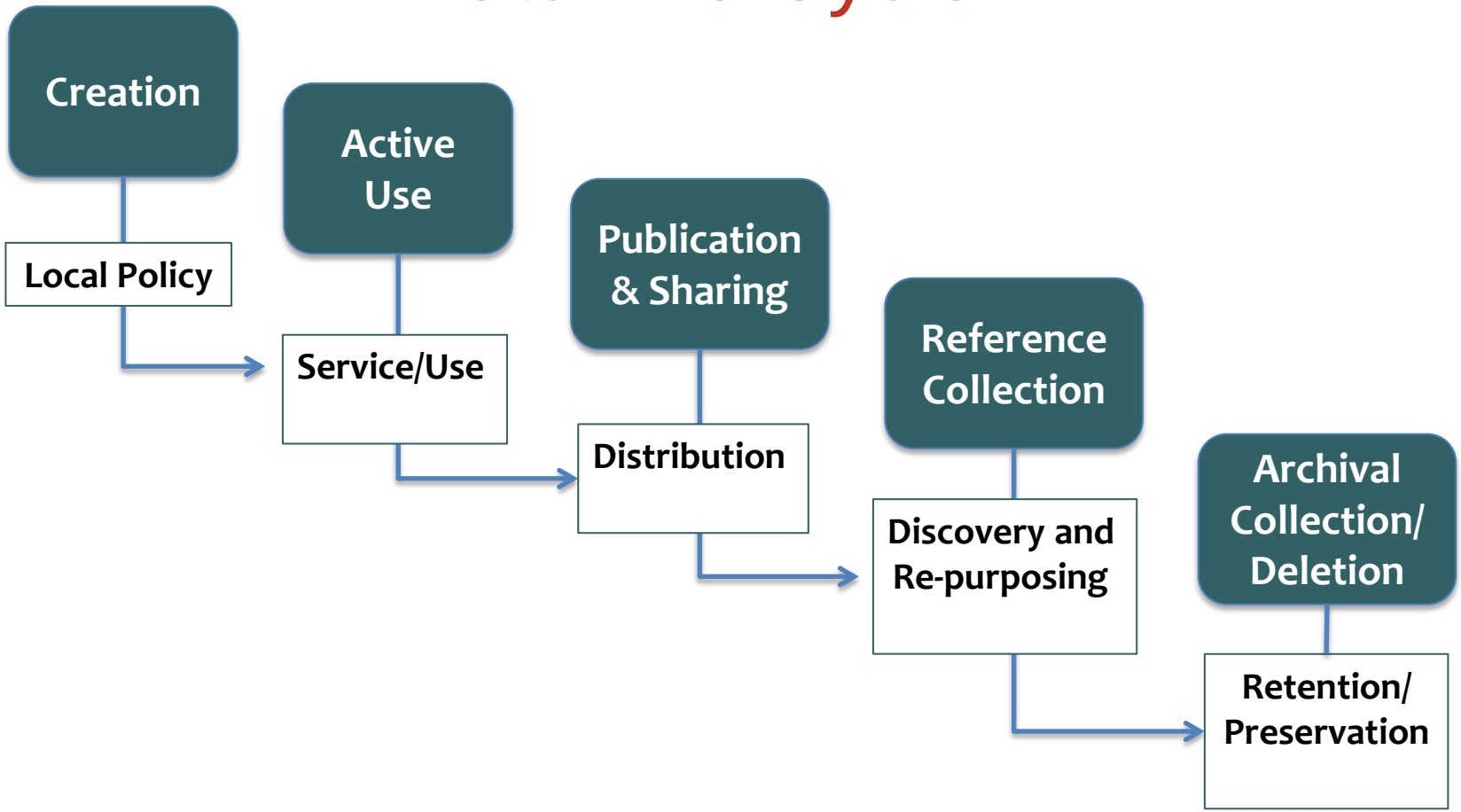
Part of the slides are courtesy of Leesa Brieger, RENCI, UNC

Issues in Data Management

Organization and Usage

- Distributed data/virtual collections
- Distributed access to data (groups); data sharing (remote access and permissions/protection)
- Publishing (general distribution)
- Back-ups and replicas
- Metadata collection and tagging
- Evolution in usage model (life cycle)

Data Life Cycle



Usage evolution across the stages of the data life cycle

More Issues in Data Management

Requirements on Data Infrastructure

- Data integrity, authenticity/verification, provenance
- Discoverability (metadata management and query support)
- Audit tracking/accounting
- Reanalysis, reproducibility
- Interfaces to the data
- Data services (derived products, new formats,…)

iRODS for Data Management

Need an overall data management plan in order to define data strategies

A data management plan = set of data policies

iRODS can be used to implement those data policies

What is iRODS?

- integrated Rule Oriented Data System
- Permit efficient data management in a distributed environment
- Based on decade-long SRB development experience for managing distributed data
- Community-driven
- Modular, extensible, customizable
- Open source (BSD license)
- Supported at UNC by DICE (Data Intensive Cyber Environments) and RENCI
  - DICE: iRODS
  - RENCI: Enterprise iRODS (E-iRODS)

Basic functionalities

- keep ordered a huge number of files
- assign simple meta-data to your files
- duplicate files across different storage resources
- move unused file from FS to tape and vice versa automatically
- search among distributed collections without bothering about their physical location
- share your data sets with other scientists
- implement your own functions

Relevant characteristics

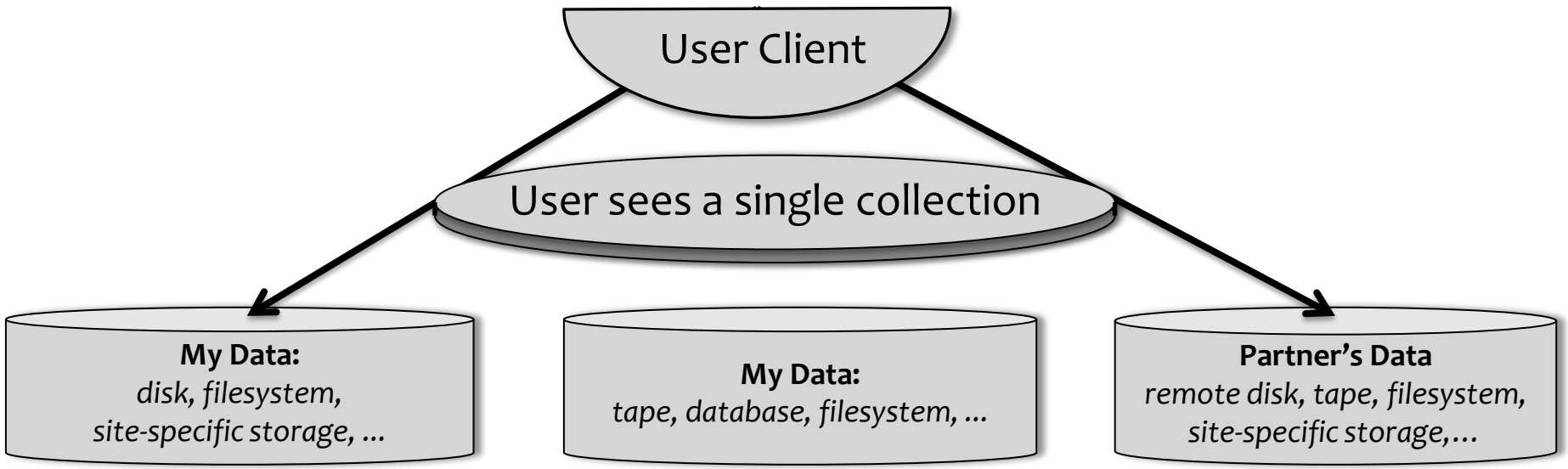
iRODS is implemented following these lines:

- a **data grid architecture** based on a client/server model that controls interactions with distributed storage
- a **metadata catalog (iCat)** implemented through a relational database system for maintaining the attributes of data and state information generated by remote operations
- a **rule system** for procedural implementation of data management policy (policy-driven data management)

Data Grid Architecture

Data Grid Architecture

iRODS View of Distributed Data

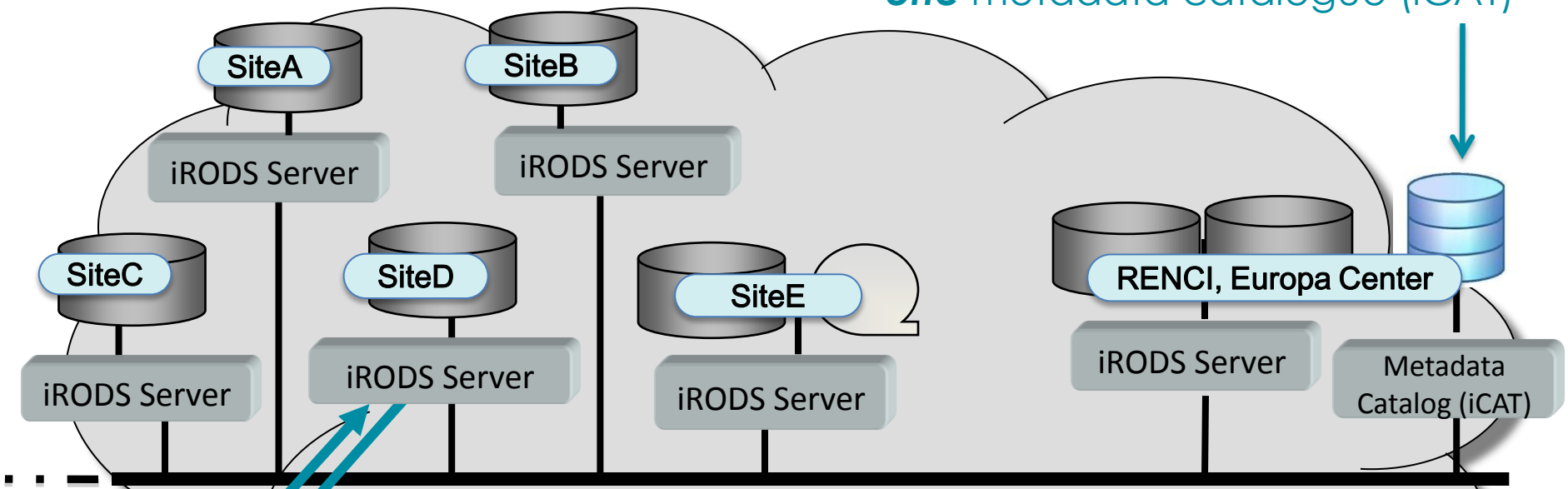


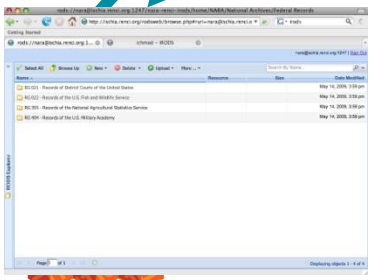
- iRODS works over heterogeneous data resources
- Users can share & manage distributed data as a single collection

iRODS as a Data Grid

- Sharing data across:
  - geographic and institutional boundaries
  - heterogeneous resources (hardware/software)
- Virtual (logical) collections of distributed data
- Global name spaces
  - data: files and collections
  - users: single sign on
  - Storage: virtual resources
- Supporting a multi-site repository model
- Metadata catalogue (iCAT) manages mappings between logical and physical name spaces

A complete data grid (**zone**) has ***one*** metadata catalogue (iCAT)

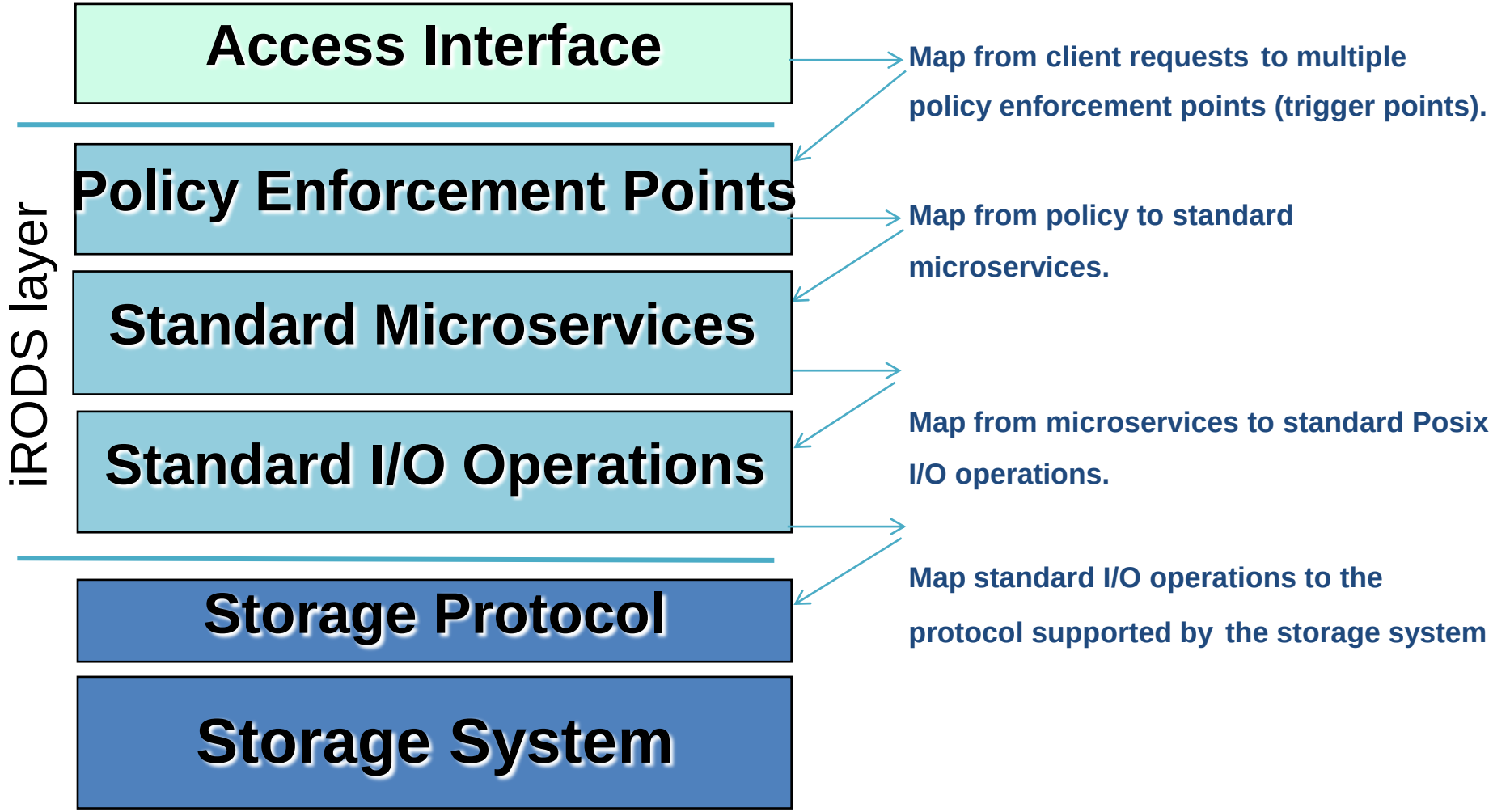




- **Client asks for data – request goes to an iRODS server**
- **Server contacts the iCAT-enabled server**
- **Information (location, access rights, etc) is retrieved from the iCAT**
- **Server containing data is signaled to send data to authorized client**

Data Virtualisation & policy management

Data virtualisation



Policy Enforcement Points

What is Policy-Driven Data Management?

- Policy determines when management procedures are run
- Define a data policy
- Identify the management steps to carry out the policy
- Define computer procedures for implementation
- Trigger the procedures when policy requires
  - Events in the data grid such as
    - putting, getting, and replicating collections or files
    - creating users, resources, groups

    can trigger procedures.
  - State of the data grid can trigger procedures
    - when a given time period has elapsed
    - whenever a file of prescribed format is ingested
    - when data reaches a prescribed age

iRODS Policy Implementation

Microservices and Rules

- Micro-service – the functional unit of work (C programs)
- Rules – workflows of micro-services (and rules)
- Provide server-side (data-side) services
- Event-triggered rule execution
- New libraries of micro-services (modules) can be developed without touching the core code
- Allow customization and extension of the data grid for community-specific policy
- **Rules** define your policy, **micro-services** implement them

Micro-services

- C code
- the unit of work within iRODS
- called by rules
- composed into workflows by rules

Running Rules

- triggered by events/policy points
- contained in the (distributed) rule base:
  - iRODS_dir/server/config/reConfigs/core.re
  - first rule with satisfied condition is executed; others are skipped
- can be run with *irule*: manual execution
- delayed execution
  - *iqstat*
  - *iqdel*

Format of a Rule

```
Rule_name{
    microservice1(…,*A,…,*B);
    microservice2(*A,…);
}
INPUT *A=”first_input", *B=”second_input"
OUTPUT ruleExecOut
```

*A and *B are workflow variables

ruleExecOut accesses the internal ruleExecInfo (rei) structure managed by iRODS.

OR

```
Rule_name(*arg)  {
    on(exp)  {
      microservice1(…,*arg,*C);
      microservice2(*C,…);
    }
}
INPUT null
OUTPUT ruleExecOut
```

- A rule can take arguments.
- A rule can be executed conditionally.
- Use “null” if there are no input parameters.

An example – list all microservices

- listMS.r (lists all available microservices)

```
ListAvailableMS {
    msiListEnabledMS(*KVPairs);
    writeKeyValPairs("stdout", *KVPairs, ": ");
}
INPUT null
OUTPUT ruleExecOut
```

- Run it with irule:

```
irule –F listMS.r
```

irule – to run a rule manually

- Example rules to tweak and run in the software distribution

  iRODS/clients/icommands/test/rules3.0

- *irule -F listMS.r*

- *irule -F rulemsiAdmShowCoreRE.r* #can only be run by admin users

Running rules (cont.)

```
# Create a new PID after a copy
acPostProcForCopy{
        on(($objPath like "/CINECA/home/rods/archive/*")
        || ($objPath like "/CINECA/EPOSReplica/archive/*"))
        {
#               addPID(*newPID);
                addPIDWithChecksum();
        }
}
```

Rules and Parameters

- Literals
  - constants: strings or numbers
  - a variable name not beginning with a special character (#, $ or *) is taken as string input
  - can only be used as input parameters (not output)
- Workflow variables
- Session state variables
- Persistent state variables

Delayed Execution

- Example

```
myTestRule{
        delay("<PLUSET>1m</PLUSET>"){
                writeLine("stdout",”Writing message with a delay.");
                msiSendStdoutAsEmail(*Mailto, "Sending email");
                }
}
INPUT *Mailto="leesa@renci.org"
OUTPUT ruleExecOut
```

- Queue management:
  - iqstat
  - iqdel
  - iqmod

Periodic Execution

Example

```
myTestRule {
# Input parameters are:
#     Source collection path
#     Target collection path
#     Optional target resource
#     Optional synchronization mode:  IRODS_TO_IRODS
# Output parameter is:
#     Status of the operation
# Output from running the example is:
#  Synchronized collection 1 with collection 2
#
   delay("<PLUSET>5m</PLUSET><EF>1h</EF>"){
   msiCollRsync(*srcColl,*destColl,*Resource,"IRODS_TO_IRODS",*Status);
   writeLine("stdout","Synchronized collection *srcColl with collection *destColl");
   }
}
INPUT *srcColl="/compZone/home/leesa/tutorials",
     *destColl="/compZone/home/leesa/tutorials2", *Resource="demoResc"
OUTPUT ruleExecOut
```

Workflow Variables (*variables)

- For example, in the following workflow chain:

```
myRule{
        msiDataObjOpen(*file,*FD);
        msiDataObjRead(*FD,10000,*BUF);
        writeLine("stdout",*BUF);
…
}

INPUT *file="/compZone/home/leesa/hello"
OUTPUT ruleExecOut
```

(‘stdout’ is a structure managed by iRODS.)

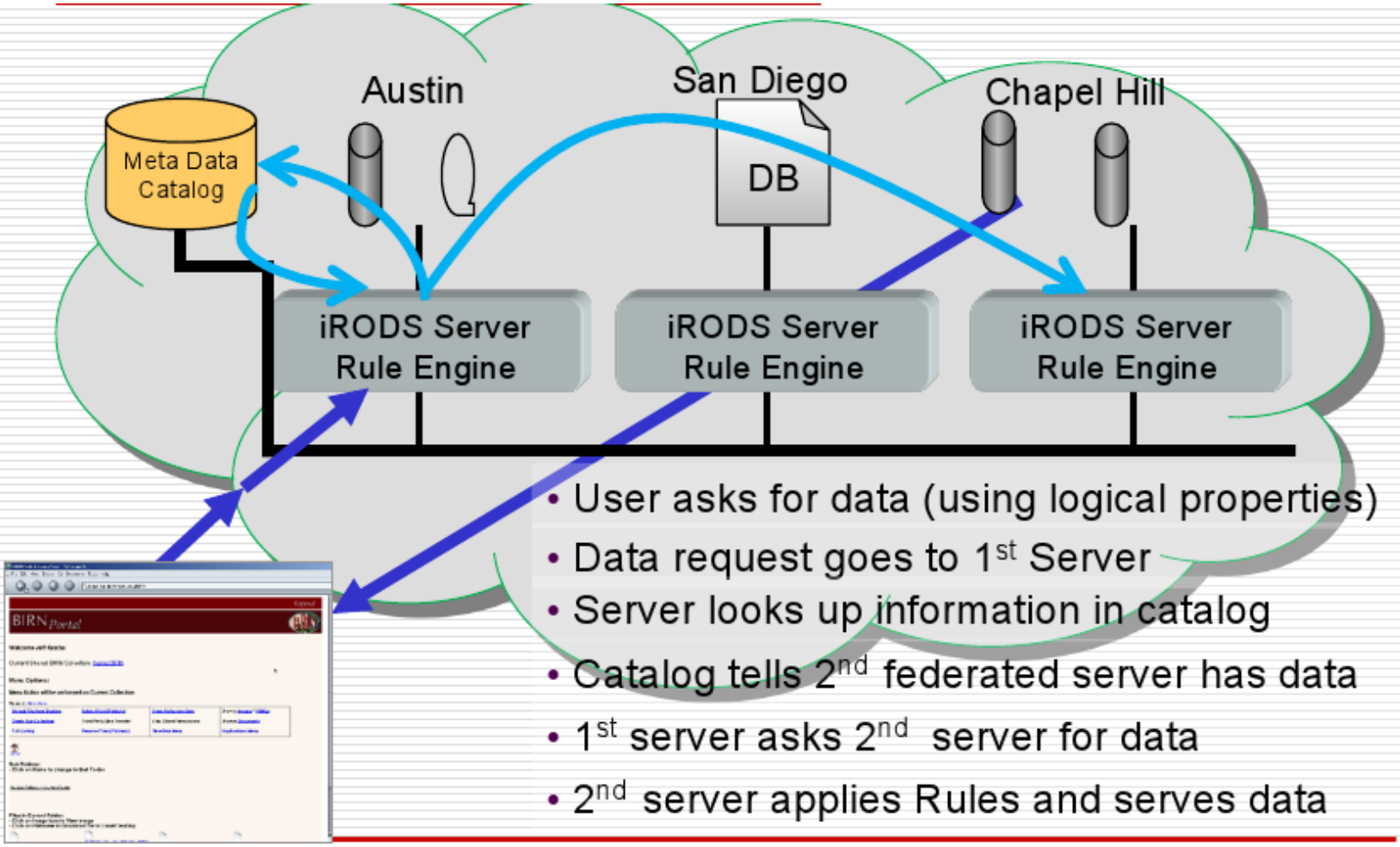
- *file is an input parameter
- *FD is output from msiDataObjOpen and input to msiDataObjRead.
- *file, *FD, and *BUF are workflow variables

Federation

iRODS federation

- iRODS can work across heterogeneous and distributed resources being part of different administration domains
- Integrated iRODS systems sharing a mutual trust form a **Federation**
- “Each” iRODS system (iCAT enabled) is called **Zone**
- Each Zone continues to be a separate iRODS instance, but the users in the multiple zones could be enabled to access data and metaData in the other zones
- Inside a federation, users of a certain zone are recognized as **remote user** for the others
- For instance, if granted, a remote user can **replicate** his data on remote zone

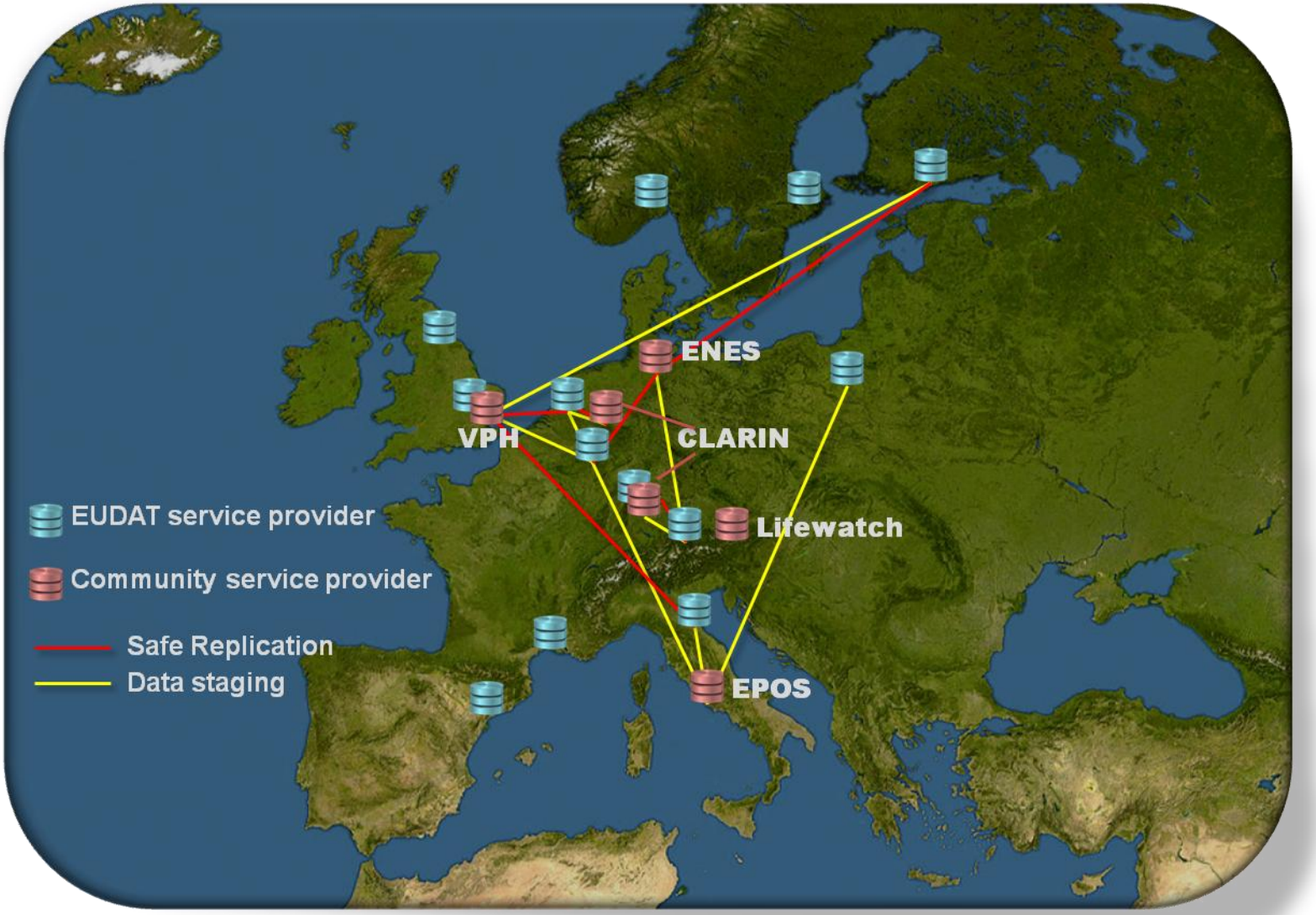
How iRODS federation works



- User asks for data (using logical properties)
- Data request goes to 1st Server
- Server looks up information in catalog
- Catalog tells 2nd federated server has data
- 1st server asks 2nd server for data
- 2nd server applies Rules and serves data

Courtesy of iRODS Documentation

iRODS federation within EUDAT



iRODS possible resources

- **File System**
  - either local or remote supporting POSIX
- **DataBase** (DBR)
  - an external database (or similar tabular information) that can be queried and updated via SQL statements (or other, for non-SQL)
- **Cloud**
  - Amazon S3
- **Archiving** system
  - IBM TSM
  - HPSS (?)

Some iRODS Clients

- **iDrop web – iDrop, iDrop-lite**
  - http://iren-web.renci.org:8080/idrop-web/login/login
- **PHP web browser**
  - http://iren-web.renci.org/rodsweb
- **icommands – unix client**
  - https://www.irods.org/index.php/icommands
- **FUSE (Filesystem in Userspace) client**
  - https://www.irods.org/index.php/iRODS_FUSE
- **Griffin – GridFTP protocol implementation**
  - https://projects.arcs.org.au/trac/griffin/wiki/releasenotes/0.6.0
- Many others supplied by user communities

Unix client: icommands

See
https://www.irods.org/index.php/icommands

Unix-like

ils
ipwd
icd
ichmod
irm
imkdir

ipasswd
irsync
ichksum
imv
icp
ienv

(Not an exhaustive list.)

FTP-like

iinit
iexit
iput
iget

icommands (cont.)

Metadata

- imeta
- iquest
- idbo

Functional

- ireg
- ibun
- irepl

Informational

- ienv
- ilsresc
- iuserinfo
- ihelp

Rule-oriented

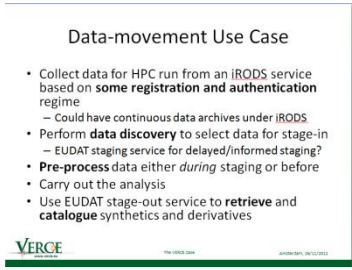
- irule
- iqstat
- iqdel
- iqmod
- idbug

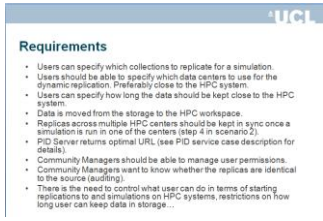
What's iRODS, what's not?

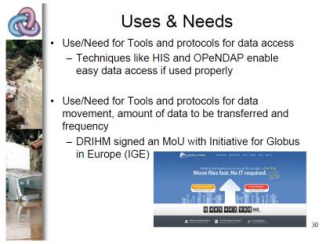
- It is **NOT** a File System, but it is able to interact with it (POSIX compliant)
- It is **NOT** a data transfer technology, but it supports different transfer protocols, including GridFTP
- It is **NOT** a virtualisation technology, but it realizes a abstraction layer between physical storage and local data path
- It is **NOT** an archiving technology, but it helps the preservation of data for long-term archiving
- It is **NOT** a back-up technology, but it can replicate data easily
- It is **NOT** a object-storage, but it can interact with existing commercial solutions (DDN-Web Object Scaler)

iRODS vs Communities

- **Integration/access of distributed data sets**
  - iRODS federation, multi-node installation
- **Stage of large data-sets onto different computing centers for processing**
  - iRODS + Griffin technology for GridFTP based transfer
- **Advanced mechanism to select collections to replicate/stage/move**
  - Search over meta-data, browsing of collections across distributed resources
- **Capability to integrate existing community systems**
  - API (Python, Java, *C++, PHP*)
- **Registration of data sets through PIDs (Persistent Identifiers)**
  - Through a specific micro-service (see. EUDAT)
- **Implementation of data management work-flow**
  - iRODS Rule engine supports transaction status process but delegation of credentials could be an issue



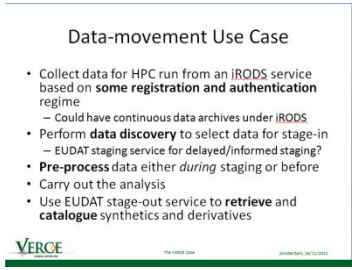


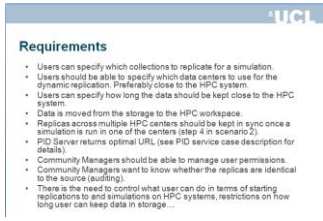


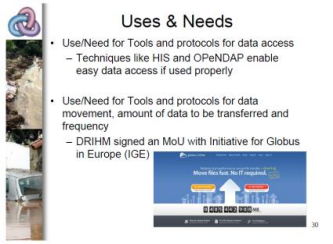
4) DATA MANAGEMENT TOOLS (data location, registration, access, movement, storage and persistency) FOR ATMOSPHERIC POLLUTION
CHIMERE application

iRODS vs Communities (cont.)

- **Manage of permission among stored data sets**
  - iRODS supports groups, ACL but not roles
- **Replica of data for long-term persistency**
  - Registration with PID
  - Replica
- **Integration with OpenDAP protocol**
  - THRREDS: https://groups.google.com/forum/?fromgroups=#!topic/irod-chat/1bRdJvJm3S4
- **Management NetCDF collections**
  - NETCDF API calls were wrapped into new iRODS API calls and micro-services so that NETCDF operations can be performed on the iRODS servers for NETCDF data stored in iRODS.
  - https://www.irods.org/index.php/NETCDF
- **Support for HDF5**
  - HDF5-iRODS module is a client-server system that provides interactive and efficient access to remote HDF5 files at iRODS server
  - http://www.hdfgroup.org/projects/irods
- **Description of data sets**
  - key-value metaData pairs already available
  - Being extendible through micro-service







4) DATA MANAGEMENT TOOLS (data location, registration, access, movement, storage and persistency) FOR ATMOSPHERIC POLLUTION
CHIMERE application

iRODS Info

- Main page: http://www.irods.org
- Chat list: irods-chat@irods.org
- iRODS Documentation:
  https://www.irods.org/index.php/Documentation
- On-line tutorial:
  https://www.irods.org/index.php/Tutorial