Federating Databases and File Systems through Policy Based Data Management

Reagan W. Moore

Arcot Rajasekar

Mike Wan

Wayne Schroeder

Mike Conway

Jason Coposky

{moore,sekar,mwan, schroeder}@diceresearch.org

michael_conway@unc.edu

http://irods.diceresearch.org

Topics

- Policy-based data management
  - Automate administrative functions
  - Enforce management policies
  - Validate assessment criteria
- Collection-based data organization
  - Descriptive metadata
- Storage-based data processing
  - Execution of workflows at the data location

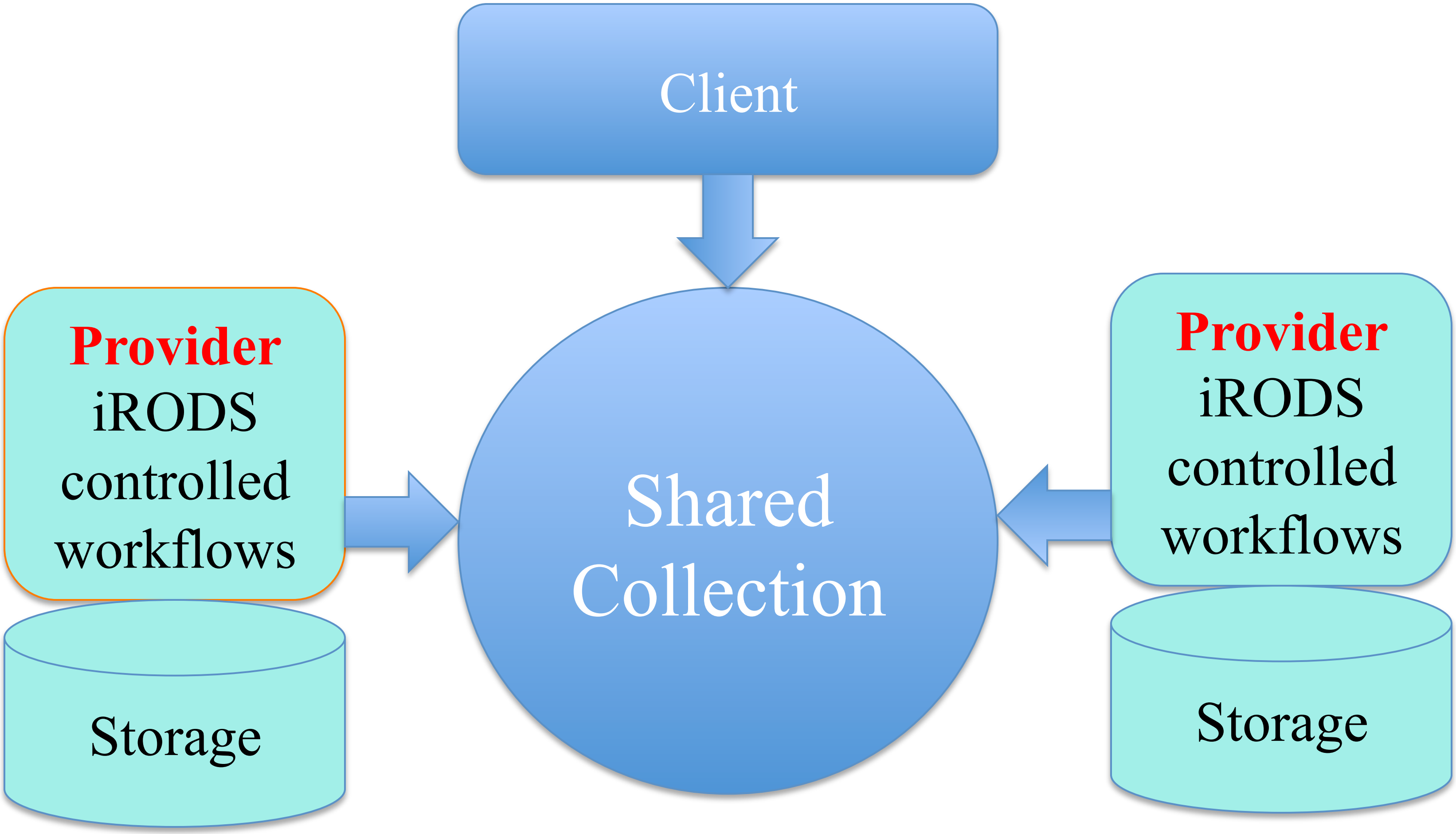
Massive Data Challenges

- Minimization of labor required for data management
- Discovery of an individual file among billions of distributed files
- Management of metadata attributes about each file
- Analysis of massive collections
- Processing at distributed storage systems

Policy-based Data Environments

- *Purpose* - reason a collection is assembled
- *Properties* - attributes needed to ensure the **purpose**
- *Policies* - controls for enforcing desired **properties,**
- **mapped to computer actionable rules**
- *Procedures* - functions that implement the **policies**
- **mapped to computer actionable workflows**
- *State information* - results of applying the **procedures**
- **mapped to system metadata**
- *Assessment criteria* - validation that **state information** conforms to the desired **purpose**
- **mapped to periodically executed policies**

Policy-based Data Sharing

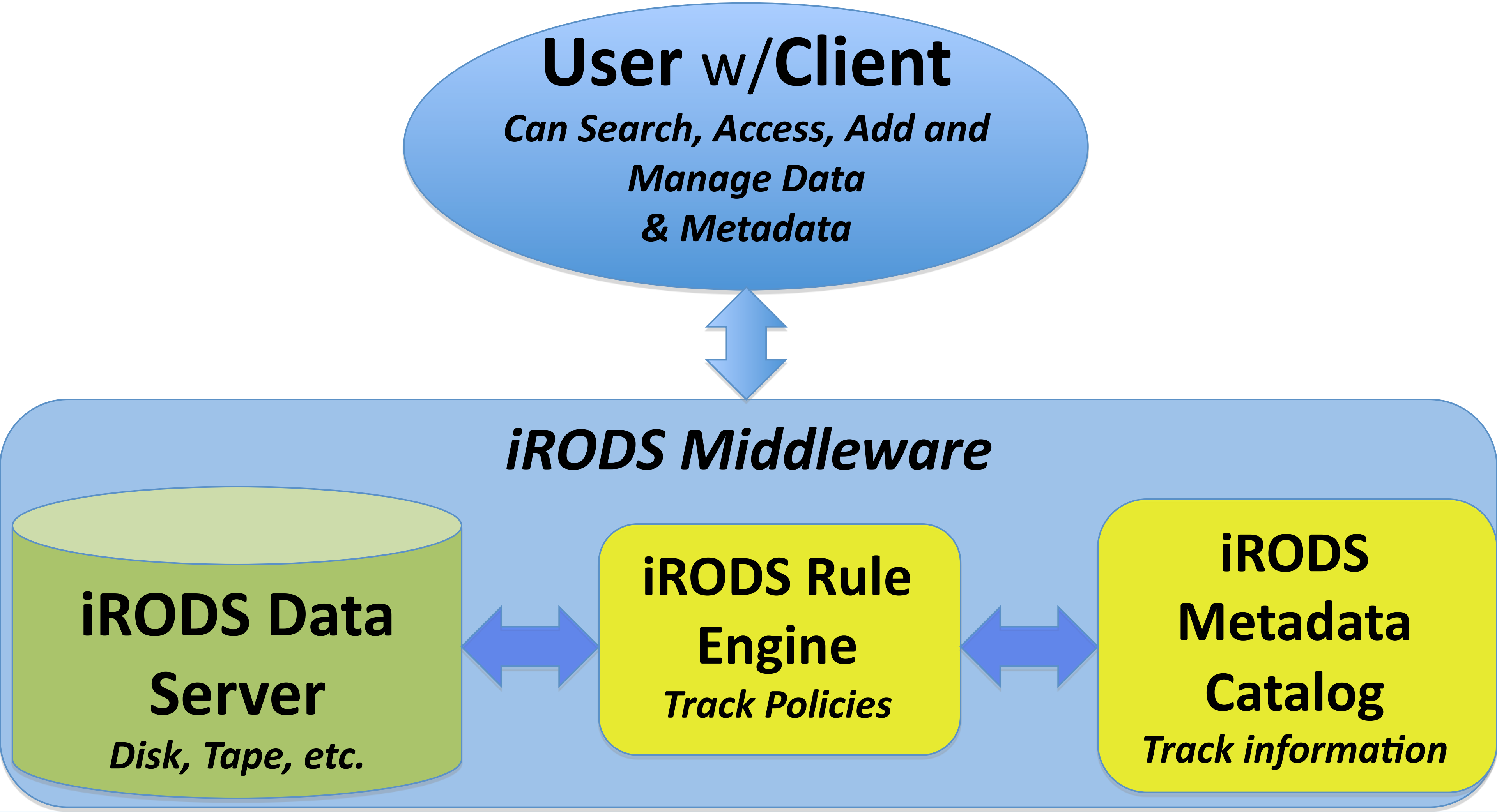


Consensus on Policies and Procedures
controlling the shared data

Applications

- Data grids – PB-size distributed collections
  - Astronomy – NOAO, CyberSKA
  - High Energy Physics – BaBar, KEK
  - Earth Systems – NASA (MODIS data set)
  - Australian Research Collaboration Service
  - Genomics – UNC-CH/RENCI
- Institutional repositories
  - Carolina Digital Repository
- Libraries
  - Texas Digital Libraries
  - Seismology - Southern California Earthquake Center
- Archives
  - Ocean Observatories Initiative
- Data processing pipelines
  - Large Synoptic Survey Telescope

Overview of iRODS Architecture



User w/Client

Can Search, Access, Add and Manage Data & Metadata

iRODS Middleware

iRODS Data Server

Disk, Tape, etc.

iRODS Rule Engine

Track Policies

iRODS Metadata Catalog

Track information

Access distributed data with Web-based Browser or iRODS GUI or Command Line clients.

iRODS Extensible Infrastructure

- Specific to data management application
  - Clients
  - Policies
  - Procedures
- Remaining infrastructure is generic
  - Authentication / Authorization
  - Network transport
  - Distributed storage access
  - Metadata management
  - Rule engine (automated rule invocation)
  - Remote procedure execution
  - Message passing (debugging, progress control)

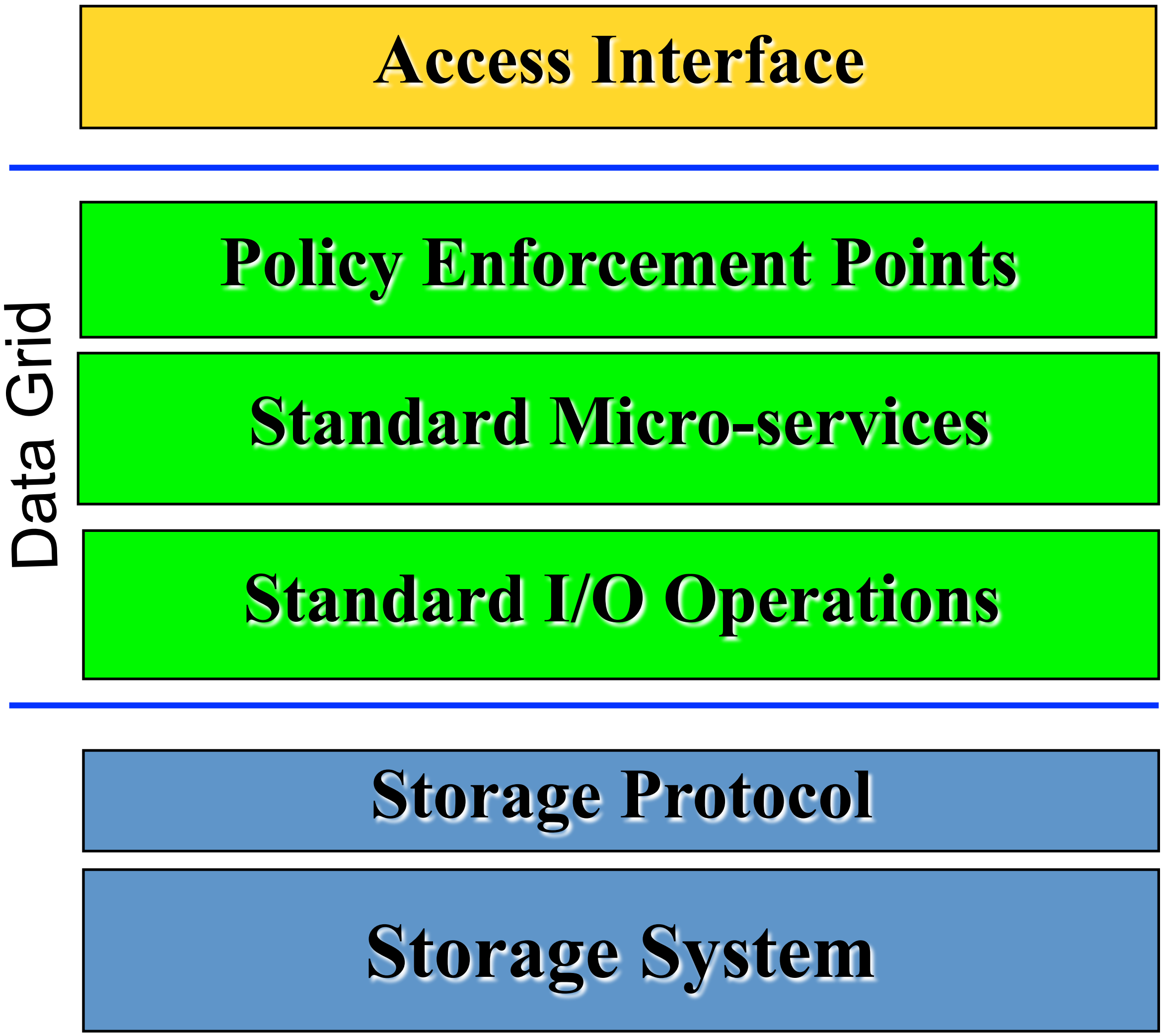
Data Life Cycle

Each data life cycle stage re-purposes the original collection

Project Collection	Data Grid	Data Processing Pipeline	Digital Library	Reference Collection	Federation
Private	Shared	Analyzed	Published	Preserved	Sustained
Local Policy	Distribution Policy	Service Policy	Description Policy	Representation Policy	Re-purposing Policy

Stages correspond to addition of new policies for a broader community
Virtualize the stages of the data life cycle through policy evolution

Highly Extensible Architecture



Access Interface

Data Grid

Policy Enforcement Points

Standard Micro-services

Standard I/O Operations

Storage Protocol

Storage System

Map from the actions requested by the client to multiple policy enforcement points.

Map from policy to standard micro-services.

Map from micro-services to standard Posix I/O operations.

Map standard I/O operations to the protocol supported by the storage system

D·I·C·E i·R·O·D·S THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL UCSD NSF THE NATIONAL ARCHIVES ARCHIVES.GOV SRB

Data Grid Clients (48)

API	Client	Developer
Browser		
	DCAPE	UNC
	iExplore	RENCI-Oleg
	JUX	IN2P3
	Peta Web browser	PetaShare
	iDrop web browser	Mike Conway
	Davis web interface	ARCS
	Rich web client	Lisa Stillwell - RENCI
Digital Library		
	Akubra/iRODS	DICE
	Dspace	MIT
	Fedora on Fuse	IN2P3
	Fedora/iRODS module	DICE
	Islandora	DICE
	Curators Workbench	CDR-UNC-CH
File System		
	Davis - Webdav	ARCS
	Dropbox / **iDrop**	DICE-Mike Conway
	FUSE	IN2P3, DICE,
	FUSE optimization	PetaShare
	OpenDAP	ARCS
	PetaFS (Fuse)	Petashare - LSU
	Petashell (Parrot)	PetaShare

D·I·C·E

i·R·O·D·S

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

UCSD

NSF

THE
NATIONAL
ARCHIVES
ARCHIVES.GOV

SRB

iRODS Clients (Cont.)

Grid	GridFTP - Griffin	ARCS
	Jsaga	IN2P3
	Parrot	UND - Doug Thain
	SRM	Academia Sinica
	Saga	KEK
I/O Libraries	PRODS - PHP	Renci - Lisa Stillwell
	C API	DICE-Mike Wan
	C I/O library	DICE-Wayne Schroeder
	Fortran	Schroeder
	Eclipse file system	CDR - UNC-CH
	Jargon	DICE-Mike Conway
	Pyrods - Python	SHAMAN-Jerome Fusillier
Portal	EnginFrame	NICE / RENCI
	Petashare Portal	LSU
Tools	Archive tools-NOAO	NOAO
	Big Board visualization	RENCI
	iFile	GA Tech
	i-commands	DICE
	Pcommands	PetaShare
	Resource Monitoring	IN2P3
	Sync-package	Academica Sinica
	URSpace	Teldap - Academica Sinica
Web Service	VOSpace	IVOA
	Shibboleth	King's College
Workflows	Kepler - actor	DICE
	Stork - interoperability	LSU
	Workflow Virtualization	LSU
	Taverna - actor	RENCI

Policy Enforcement Points

- Currently have 71 locations within iRODS framework where policies are checked.
  - Each action may involve multiple policy enforcements points
- Policy enforcement points
  - Pre-action policy (selection of storage location)
  - Policy execution (file deletion control)
  - Post-action policy (derived data products)

Policy Enforcement Points (71)

ACTION	PRE-ACTION POLICY	POST-ACTION POLICY
acCreateUser	acPreProcForCreateUser	acPostProcForCreateUser
acDeleteUser	acPreProcForDeleteUser	acPostProcForDeleteUser
acGetUserbyDN	acPreProcForModifyUser	acPostProcForModifyUser
acTrashPolicy	acPreProcForModifyUserGroup	acPostProcForModifyUserGroup
acAclPolicy	acChkHostAccessControl	acPostProcForDelete
acSetCreateConditions	acPreProcForCollCreate	acPostProcForCollCreate
acDataDeletePolicy	acPreProcForRmColl	acPostProcForRmColl
acRenameLocalZone	acPreProcForModifyAVUMetadata	acPostProcForModifyAVUMetadata
acSetRescSchemeForCreate	acPreProcForModifyCollMeta	acPostProcForModifyCollMeta
acRescQuotaPolicy	acPreProcForModifyDataObjMeta	acPostProcForModifyDataObjMeta
acSetMultiReplPerResc	acPreProcForModifyAccessControl	acPostProcForModifyAccessControl
acSetNumThreads	acPreprocForDataObjOpen	acPostProcForOpen
acVacuum	acPreProcForObjRename	acPostProcForObjRename
acSetResourceList	acPreProcForCreateResource	acPostProcForCreateResource
acSetCopyNumber	acPreProcForDeleteResource	acPostProcForDeleteResource
acVerifyChecksum	acPreProcForModifyResource	acPostProcForModifyResource
acCreateUserZoneCollections	acPreProcForModifyResourceGroup	acPostProcForModifyResourceGroup
acDeleteUserZoneCollections	acPreProcForCreateToken	acPostProcForCreateToken
acPurgeFiles	acPreProcForDeleteToken	acPostProcForDeleteToken
acRegisterData	acNoChkFilePathPerm	acPostProcForFilePathReg
acGetIcatResults	acPreProcForGenQuery	acPostProcForGenQuery
acSetPublicUserPolicy	acSetReServerNumProc	acPostProcForPut
acCreateDefaultCollections	acSetVaultPathPolicy	acPostProcForCopy
acDeleteDefaultCollections		acPostProcForCreate

iput ../src/irm.c

checks 10 policy hooks

```
srbbrick14:10900:ApplyRule#116:: acChkHostAccessControl
srbbrick14:10900:GotRule#117:: acChkHostAccessControl
srbbrick14:10900:ApplyRule#118:: acSetPublicUserPolicy
srbbrick14:10900:GotRule#119:: acSetPublicUserPolicy
srbbrick14:10900:ApplyRule#120:: acAclPolicy
srbbrick14:10900:GotRule#121:: acAclPolicy
srbbrick14:10900:ApplyRule#122:: acSetRescSchemeForCreate
srbbrick14:10900:GotRule#123:: acSetRescSchemeForCreate
srbbrick14:10900:execMicroSrvc#124:: msiSetDefaultResc(demoResc,null)
srbbrick14:10900:ApplyRule#125:: acRescQuotaPolicy
srbbrick14:10900:GotRule#126:: acRescQuotaPolicy
srbbrick14:10900:execMicroSrvc#127:: msiSetRescQuotaPolicy(off)
srbbrick14:10900:ApplyRule#128:: acSetVaultPathPolicy
srbbrick14:10900:GotRule#129:: acSetVaultPathPolicy
srbbrick14:10900:execMicroSrvc#130:: msiSetGraftPathScheme(no,1)
srbbrick14:10900:ApplyRule#131:: acPreProcForModifyDataObjMeta
srbbrick14:10900:GotRule#132:: acPreProcForModifyDataObjMeta
srbbrick14:10900:ApplyRule#133:: acPostProcForModifyDataObjMeta
srbbrick14:10900:GotRule#134:: acPostProcForModifyDataObjMeta
srbbrick14:10900:ApplyRule#135:: acPostProcForCreate
srbbrick14:10900:GotRule#136:: acPostProcForCreate
srbbrick14:10900:ApplyRule#137:: acPostProcForPut
srbbrick14:10900:GotRule#138:: acPostProcForPut
srbbrick14:10900:GotRule#139:: acPostProcForPut
srbbrick14:10900:GotRule#140:: acPostProcForPut
```

Policies

- Retention, disposition, distribution, arrangement
- Authenticity, provenance, description
- Integrity, replication, synchronization
- Deletion, trash cans, versioning
- Archiving, staging, caching
- Authentication, authorization, redaction
- Access, approval, IRB, audit trails, report generation
- Assessment criteria, validation
- Derived data product generation, format parsing
- Federation of independent data grids

Collection-based Management

- Data grids associate metadata with each file
  - Provenance information
  - Description information
  - System state information
  - Assessment results
- Discovery is based on queries on metadata
  - Result is returned as a list
  - Support processing on items in list

Integration of Databases with File Systems

- Central approach – iRODS iCAT catalog
  - Manage attributes on files, collections, storage, users, rules
  - Query catalog / retrieve metadata / loop over result set / apply operations on each file
- Use schema indirection to add metadata to any file
- Use extensible schema to add tables to schema

KEK Paper

IRODS in an Neutrino Experiment

Adil Hasan

for

Francesca Di Lodovico (QMUL), Yoshimi Iida (KEK), Takashi Sasaki (KEK)

https://www.irods.org/index.php/iRODS_User_Group_Meeting_2011

iRODS Rule to Bundle Files

```
acKEKBundle(*collPath, *bundlePath, *cacheRes, *compRes, *archive,
*threshold) {
 msiCheckCollSize(*collPath, *cacheRes, *threshold, *aboveThreshold,
     *status);
 IF(*aboveThreshold == 1)
     {
          msiWriteRodsLog("Creating bundle", *status);
          msiPhyBundleColl(*collPath, *compRes,*status);
          msiWriteRodsLog("Finished bundling, starting to replicate", *status);
          msiCollRepl(*bundlePath, verifyChksum++++backupRescName
                =*archive, *status);
          msiWriteRodsLog("Finished replicating bundle", *status)’
     }
}
```

iRODS Rule to Replicate Files

```
acKEKReplicate(*collPath, *cacheRes, *archive, *threshold) {
msiCheckCollSize(*collPath, *cacheRes, *threshold, *aboveThreshold, *status);
IF(*aboveThreshold == 1) {
 msiWriteRodsLog("Starting to backup files", *status);
 acGetIcatResults(list, COLL_NAME LIKE '*collPath', *List);
 forEachExec(*List) {
  msiGetValByKey(*List, DATA_NAME, *Data);
  msiGetValByKey(*List, COLL_NAME, *Coll);
  msiGetValByKey(*List, DATA_RESC_NAME, *dataRes);
  IF(*dataRes == *cacheRes) {
    msiWriteRodsLog("Replicating file *Coll/*Data", *status);
    msiDataObjRepl(*Coll/*Data, verifyChksum++++backupRescName=
               *archive,  *status);
    msiWriteRodsLog("Completed replicating file *Coll/*Data”,*status);
    }
   }
 }
}
```

iRODS Rule to Trim Replicas

```
acKEKTrimData(*collPath, *cacheRes){
  acGetIcatResults(list, COLL_NAME LIKE '*collPath', *List);
  forEachExec(*List) {
      msiGetValByKey(*List, DATA_NAME, *Data);
      msiGetValByKey(*List, COLL_NAME, *Coll);
      msiGetValByKey(*List, DATA_RESC_NAME, *DataResc);
      msiGetValByKey(*List, DATA_REPL_NUM, *DataRepl);
      IF(*DataResc == *cacheRes) {
            msiWriteRodsLog("About to trim file *Coll/*Data", *status);
            msiDataObjTrim(*Coll/*Data, *cacheRes, *DataRepl, 1,
                  “irodsAdmin”, *status);
            msiWriteRodsLog("Completed trimming replicas of *Coll/*Data",
                  *status);
      }
  }
}
```

Database Federation

- External distributed catalog approach
  - Define database resource, accessed through database driver - DBR
  - Define database object, a file containing the query that can be issued against the remote database - DBO
  - Create database object record, a file containing the result of the query - DBOR
  - Loop over results in DBOR, processing files
  - Example is query across multiple iRODS catalogs

iRODS - Open Source Software

Reagan W. Moore

rwmoore@renci.org

http://irods.diceresearch.org

NSF OCI-0848296 “NARA Transcontinental Persistent Archives Prototype”
NSF SDCI-0721400 “Data Grids for Community Driven Applications”