

Policy Based Data Management

Reagan W. Moore

Arcot Rajasekar

Mike Wan

Wayne Schroeder

Mike Conway

Jason Cposky

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

michael_conway@unc.edu

<http://irods.diceresearch.org>



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Agenda

8:30- 9:00	Registration
9:00-10:00	Introduction
10:00-10:30	Installation Guidelines
10:30-11:00	Coffee break
11:00-11:30	User interfaces
11:30-12:00	New features in current release
12:00-12:30	Workflow integration & rule examples
12:30-14:00	Lunch
14:00-15:00	Use cases
15:00-15:30	SRM-iRODS interface, Weilong Ueng
15:30-16:00	Coffee break
16:00-17:30	Desired feature discussion

General Overview

iRODS Development Motivation



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Common Infrastructure

- Data grids to support collaborations
 - Shared collections
- Data processing pipelines to support data mining
 - Data-driven science based on data mining
 - Detect significant events
 - Generate standard products and statistics (vary input)
- Digital libraries to support publication within a discipline
 - Provide services for use of standard data products
- Preservation Environments to support reference collections
 - Digital holdings on which future research is based



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Digital Library

Texas Digital Library
French National Library

Data Processing Pipeline

Ocean Observatories Initiative
Large Synoptic Survey Telescope

Data Grid

Teragrid
Temporal Dynamics of Learning Center
Australian Research Collaboration Service

Preservation Environment

NARA Transcontinental Persistent Archive Prototype
Taiwan National Archive
Carolina Digital Repository



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Requirements

- Observe that many projects are generating massive data collections
 - Observational data (astronomy, climate change, oceanography)
 - Experimental data (high energy physics, biology)
 - Simulation output (high energy physics, seismology, earth systems, cosmology)
- Data are widely distributed
 - Sources, storage systems, analysis systems, users
- Desired scale is now tens to hundreds of petabytes, hundreds of millions to billions of files



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Cloud Storage



Institutional Repositories

Carolina Digital Repository

Texas Digital Library

Federal Repositories

National Climatic Data Center

National Optical Astronomy Observatory



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Questions

- What is the expected impact of data intensive computing:
 - Support remote processing of data
 - Move processing to the data
- What is the expected impact of research collaborations:
 - Support sharing of data
 - Enforce management policies
- What is the expected impact of data life cycle management:
 - Virtualize the data life cycle



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Policy-based Data Environments

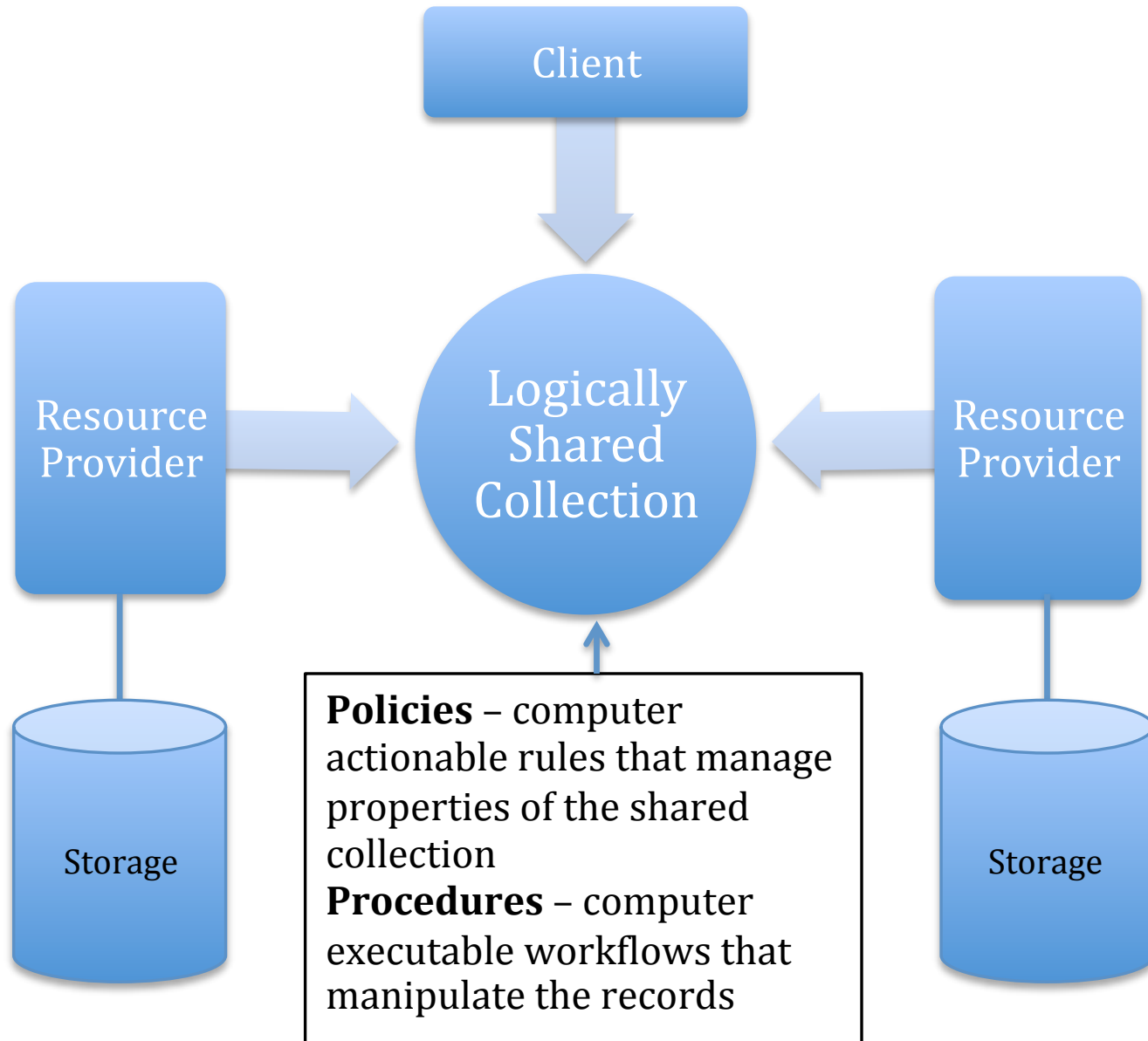
- *Purpose* - reason a collection is assembled
- *Properties* - attributes needed to ensure the **purpose**
- *Policies* - controls for enforcing desired **properties**,
- **mapped to computer actionable rules**
- *Procedures* - functions that implement the **policies**
- **mapped to computer actionable 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**



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Policy Based Interoperability



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**

Tracks Policies



**iRODS
Metadata
Catalog**

Track information

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

iRODS Server

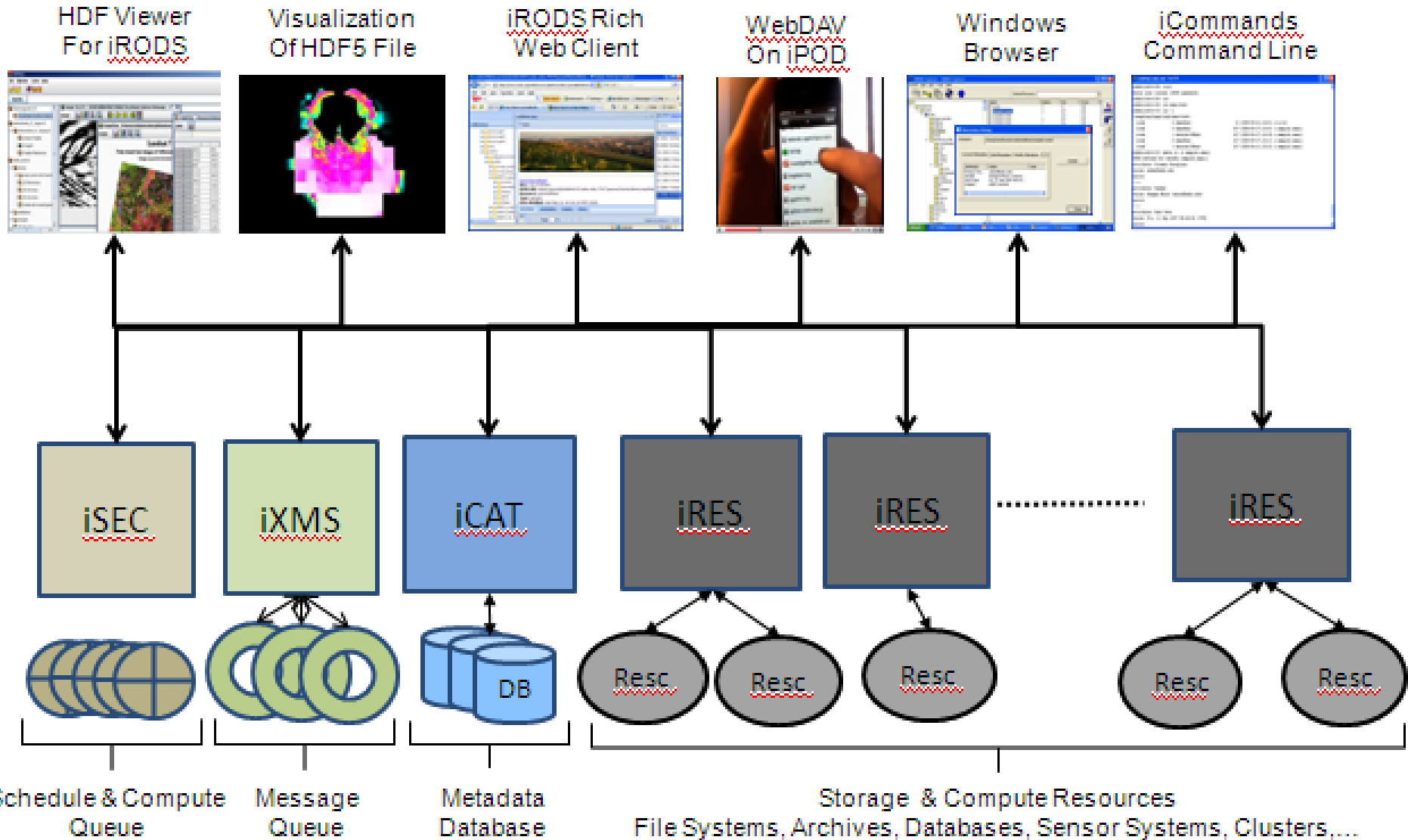
Peer to Peer Architecture



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



iRODS Distributed Data Management



Server Functions

- Peer-to-peer architecture
 - Interact with remote metadata catalog
 - Forward requests to server where data reside
- Translate from client action to storage protocol
- Authenticate and authorize operation
- Map from logical file name space to physical path
- Enforce policies in local policy engine
- Manage execution of processes at the storage location

Data Virtualization

Access Interface

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

Policy Enforcement Points

Map from policy to standard micro-services.

Standard Micro-services

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

Standard I/O Operations

Storage Protocol

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

Storage System

Data Grid



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Data Grid Clients (48)

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



iRODS Clients (Cont.)

Grid			
	GridFTP - Griffin	ARCS	
	Jsaga	IN2P3	Jargon
	Parrot	UND - Doug Thain	
	SRM	Academia Sinica	
	Saga	KEK	
I/O Libraries			
	PRODS - PHP	Renci - Lisa Stillwell	
	C API	DICE-Mike Wan	C
	C I/O library	DICE-Wayne Schroeder	C
	Fortran	Schroeder	C
	Eclipse file system	CDR - UNC-CH	Jargon
	Jargon	DICE-Mike Conway	Jargon
	Pyrods - Python	SHAMAN-Jerome Fusillier	Python
Portal			
	EnginFrame	NICE / RENCi	Jargon
	Petashare Portal	LSU	Jargon
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	Jargon
	Stork - interoperability	LSU	
	Workflow Virtualization	LSU	
	Taverna - actor	RENCi	

Rules

Distributed Rule Engine

Distributed Rule Base – core.irb

Computer Actionable 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



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Format of a Rule

- Action | Condition | MS₁, ..., MS_n | RMS₁, ..., RMS_n
- Action
 - Name of action to be performed
 - Name known to the server and invoked by server
- Condition – condition under which the rule apply
- Micro-services - If applicable micro services will be executed
- Recovery micro-service - If any micro service fails, recovery micro service(s) executed to maintain transactional consistency
- Example of MS/RMS
 - createFile(*F) removeFile(*F)
 - ingestMetadata(*F,*M) rollback

Policy Enforcement Points

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

Data Virtualization

Access Interface

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

Policy Enforcement Points

Map from policy to standard micro-services.

Standard Micro-services

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

Standard I/O Operations

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

Storage Protocol

Storage System

Data Grid



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Policy Enforcement Points (71)

ACTION

acCreateUser
acDeleteUser
acGetUserbyDN
acTrashPolicy
acAclPolicy
acSetCreateConditions
acDataDeletePolicy
acRenameLocalZone
acSetRescSchemeForCreate
acRescQuotaPolicy
acSetMultiReplPerResc
acSetNumThreads
acVacuum
acSetResourceList
acSetCopyNumber
acVerifyChecksum
acCreateUserZoneCollections
acDeleteUserZoneCollections
acPurgeFiles
acRegisterData
acGetIcatResults
acSetPublicUserPolicy
acCreateDefaultCollections
acDeleteDefaultCollections

PRE-ACTION POLICY

acPreProcForCreateUser
acPreProcForDeleteUser
acPreProcForModifyUser
acPreProcForModifyUserGroup
acChkHostAccessControl
acPreProcForCollCreate
acPreProcForRmColl
acPreProcForModifyAVUMetadata
acPreProcForModifyCollMeta
acPreProcForModifyDataObjMeta
acPreProcForModifyAccessControl
acPreprocForDataObjOpen
acPreProcForObjRename
acPreProcForCreateResource
acPreProcForDeleteResource
acPreProcForModifyResource
acPreProcForModifyResourceGroup
acPreProcForCreateToken
acPreProcForDeleteToken
acNoChkFilePathPerm
acPreProcForGenQuery
acSetReServerNumProc
acSetVaultPathPolicy

POST-ACTION POLICY

acPostProcForCreateUser
acPostProcForDeleteUser
acPostProcForModifyUser
acPostProcForModifyUserGroup
acPostProcForDelete
acPostProcForCollCreate
acPostProcForRmColl
acPostProcForModifyAVUMetadata
acPostProcForModifyCollMeta
acPostProcForModifyDataObjMeta
acPostProcForModifyAccessControl
acPostProcForOpen
acPostProcForObjRename
acPostProcForCreateResource
acPostProcForDeleteResource
acPostProcForModifyResource
acPostProcForModifyResourceGroup
acPostProcForCreateToken
acPostProcForDeleteToken
acPostProcForFilePathReg
acPostProcForGenQuery
acPostProcForPut
acPostProcForCopy
acPostProcForCreate



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Policy Invocation

iput ../src/irm.c

checks 10 policy hooks

srbrick14:10900:ApplyRule#116:: acChkHostAccessControl

srbrick14:10900:GotRule#117:: acChkHostAccessControl

srbrick14:10900:ApplyRule#118:: acSetPublicUserPolicy

srbrick14:10900:GotRule#119:: acSetPublicUserPolicy

srbrick14:10900:ApplyRule#120:: acAclPolicy

srbrick14:10900:GotRule#121:: acAclPolicy

srbrick14:10900:ApplyRule#122:: acSetRescSchemeForCreate

srbrick14:10900:GotRule#123:: acSetRescSchemeForCreate

srbrick14:10900:execMicroSrvc#124:: msiSetDefaultResc

(demoResc,null)

srbrick14:10900:ApplyRule#125:: acRescQuotaPolicy

srbrick14:10900:GotRule#126:: acRescQuotaPolicy

srbrick14:10900:execMicroSrvc#127:: msiSetRescQuotaPolicy(off)

Policy Invocations (Cont.)

srbrick14:10900:ApplyRule#128:: acSetVaultPathPolicy

srbrick14:10900:GotRule#129:: acSetVaultPathPolicy

srbrick14:10900:execMicroSrvc#130:: msiSetGraftPathScheme(no,1)

srbrick14:10900:ApplyRule#131:: acPreProcForModifyDataObjMeta

srbrick14:10900:GotRule#132:: acPreProcForModifyDataObjMeta

srbrick14:10900:ApplyRule#133:: acPostProcForModifyDataObjMeta

srbrick14:10900:GotRule#134:: acPostProcForModifyDataObjMeta

srbrick14:10900:ApplyRule#135:: acPostProcForCreate

srbrick14:10900:GotRule#136:: acPostProcForCreate

srbrick14:10900:ApplyRule#137:: acPostProcForPut

srbrick14:10900:GotRule#138:: acPostProcForPut

srbrick14:10900:GotRule#139:: acPostProcForPut

srbrick14:10900:GotRule#140:: acPostProcForPut

irule command

- irule execution file (test.ir) has three lines – Workflow : Input : Output

myTestRule|"condition"|"workflow"|"recovery-workflow"

Input-parameters separated by %

Output-parameters separated by %

Include ruleExecOut as an output parameter

Listing the Rule Base

showCore.ir rule (text file)

myTest | | msiAdmShowIRB(*A) | nop

rule

Null

input

*A%ruleExecOut

output

- We can execute the rule to show the rule base:
 - irule -vF showCore.ir

Sample Rule

Parameters

RuleName

```
OnIngestObject (*D ) {
```

```
ON ($userDept == sils)
```

Condition

```
{
```

Actions

```
msiComputeChkSum(*D) ::: null;
```

```
acReplicateFile(*D, tapeResource)
```

```
MicroService ::: acTrimFile(*D, tapeResource);
```

Policy: On ingestion of a new file by a 'sils' user, immediately

compute its checksum and store it in the iCAT. Also replicate the file in 'tapeResource'. If the replication fails remove all evidence of the replica

Recovery

```
}
```

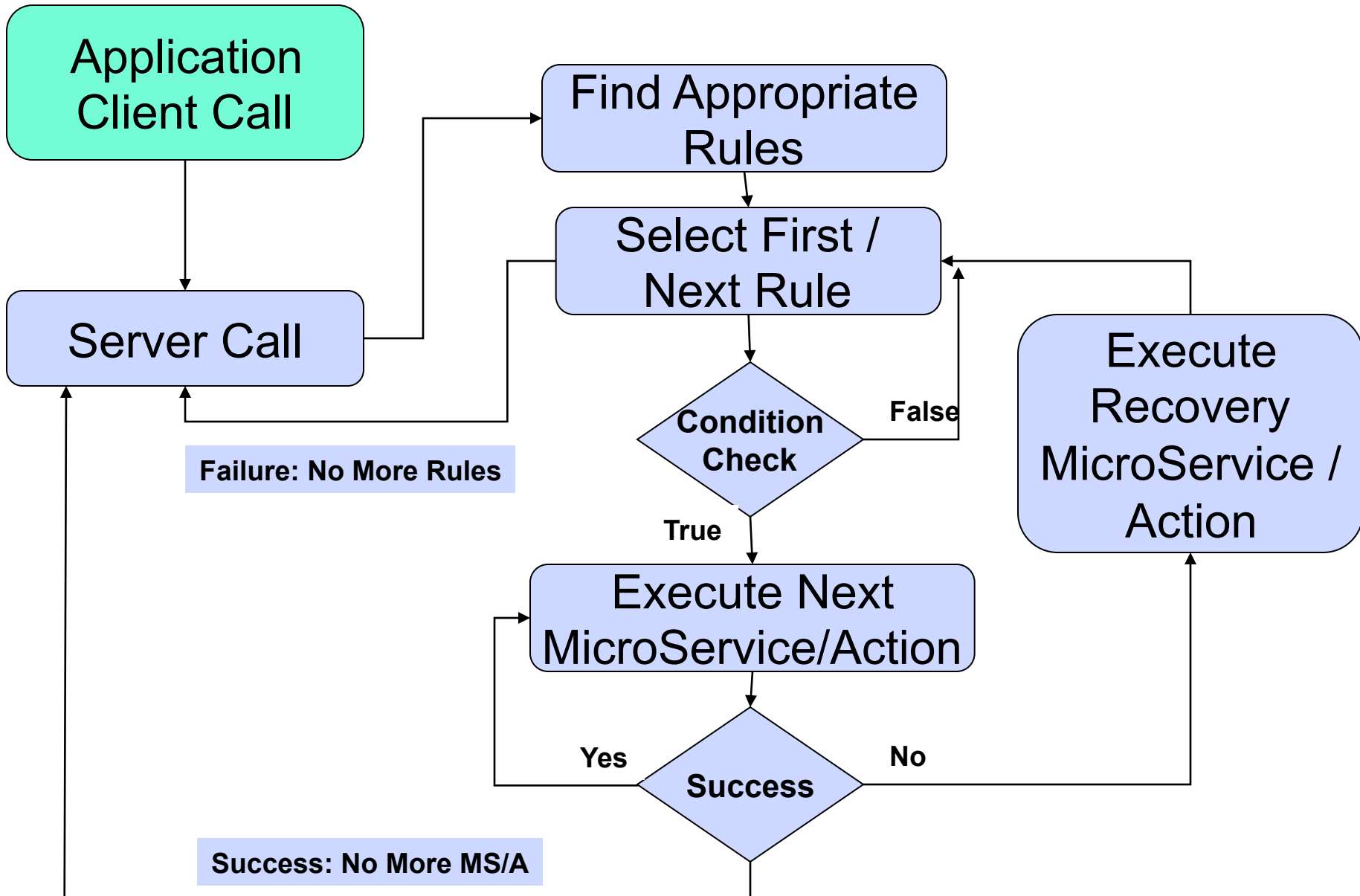
Sample Rule – Internal Form

```
OnIngestObject (*D ) {  
    ON ($userDept == sils)  
    {  
        msiComputeChkSum(*D) ;  
        acReplicateFile(*D, tapeResource)  
            ::: acTrimFile(*D, tapeResource);  
    }  
}
```

Conversion done by “rulegen” utility found in icommands/rulegen
See also the “Rules” page in iRODS Wiki

```
OnIngestObject(*D) | $userDept == sils | msiComputeChkSum  
(*D)##acReplicateFile(*D,tapeResource) |  
null##acTrimFile(*D,tapeResource)
```


Rule Flow



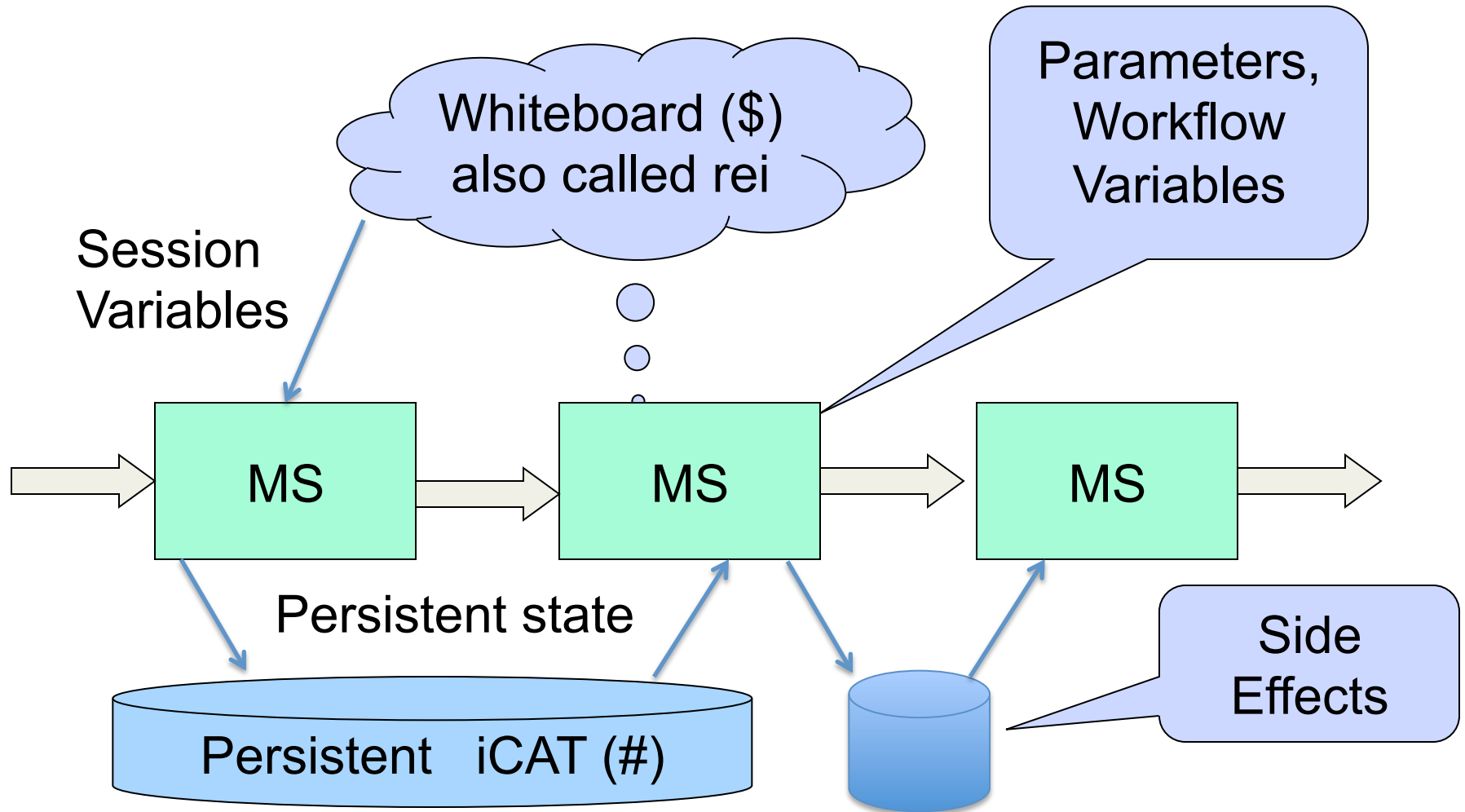
Micro-Services

Manage exchange of structured information between micro-services

Micro-service Communication

- Micro-services communicate through:
 - Arguments/Parameters
 - Inside a rule from one micro-service to another rule or micro-service
 - Session Memory – white board (\$-variables)
 - Stores common (context) information
 - User and resource information
 - More information about Data or collection of interest
 - Persistent Memory – persistent state information
 - Query an iCAT for information (coded inside the micro-service)
 - XMessages – out-of-band communications
 - Like a Post Office

Data Flow between Micro-services



Literals

- Literals are strings or numbers.
- They are constants (not variables which can be assigned value)
- In a rule, if an argument does not begin with a special character (`#`, `$` or `*`), it is treated as a character string input.
- Example:
`msiSortRescSortScheme("random", $rescName)`

the character string "random" will be passed in as input. Literal can only be used as input parameters and not output parameters.

Session Variables (\$)

- **Session State Information** is temporary information that is maintained only during the server-session. It does not persist beyond the session.
 - Session variables are persistent across two rule executions in the same session. So, can be used to pass information between rule executions.
- Session variables carry information about the connection between client and server, data objects, user information, resource information, etc.
 - They also carry information that can be sent back to the client. Example: **stdout, stderr**.

\$-variable: More details

- \$-variables are pre-defined by iRODS
 - (unlike *-variables which are user created)
- \$-variables are stored in memory as a complex (tree-like) C-structure (called the **rei structure**).
- The \$-variable names map to specific location in this structure.
- The mapping is given in **core.dvm** file in the directory **server/config/reConfigs**
- Example:
 - dataType||rei->doi->dataType
 - userNameClient||rei->uoic->userName
 - collName||rei->coi->collName
 - collParentName||rei->coi->collParentName
- More than one mapping is also allowed!!

Workflow Variables

- `writeLine(stdout,*D)`
- The workflow variable in this case is 'D'
- The 'stdout' parameter is a structure managed by iRODS
- The "ruleExecOut" parameter is a structure managed by iRODS

Workflow Variable Example

- For example, in the following workflow chain:
- `msiDataObjOpen(/x/y/z,*FD)`
`##msiDataObjRead(*FD,10000,*BUF)`
 - `msiDataObjOpen` opens a data object with the input path `/x/y/z` and the output file descriptor is placed in the variable parameter `*FD`. `*FD` is then used by `msiDataObjRead` as an input parameter for the read.
- Note that `*FD` and `*BUF` are workflow variables

Rule Condition

- Condition under which the Rule applies
- Examples
 - `$rescName == demoResc8`
 - `$objPath like /x/y/z/*`
- Many operators
 - `==, !=, >, <, >=, <=`
 - `%%, !!` (and, or)
 - `expr like reg-expr, expr not like reg-expr, expr ::= string`

Rules with Conditions

#These rules generate the genQueryOut_ structure for each action for the given condition

```
acGetIcatResults(*Action,*Cond,*GenQOut)|
    (*Action == replicate) %% (*Action == trim) %%
    (*Action == chksum) %% (*Action == copy) %% (*Action == remove)
| msiMakeQuery("DATA_NAME,COLL_NAME",*Cond,*Query)##
    msiExecStrCondQuery(*Query, *GenQOut)##cut
| nop##nop
```

```
acGetIcatResults(*Action,*Cond,*GenQOut)|
    (*Action == chksumRescLoc)
| msiMakeQuery("DATA_NAME, COLL_NAME, RESC_LOC",
    *Cond,*Query)##
    msiExecStrCondQuery(*Query, *GenQOut)##cut
| nop##nop
```

Micro-Services

Functional Units for Building
Procedures

Micro-Services

- Functions written in C
- Provided with the iRODS server code
- Provide:
 - Standard operations
 - Queries on metadata catalog
 - Interaction with web services
 - Invocation of external applications
 - Workflow constructs (loops, conditionals, exit)
 - Remote and delayed execution control

Micro-services - How many are needed?

print_hello_arg
msiVacuum
msiQuota
msiGoodFailure
msiSetResource
msiCheckPermission
msiCheckOwner
msiCreateUser
msiCreateCollByAdmin
msiSendMail
recover_print_hello
msiCommit
msiRollback
msiDeleteCollByAdmin
msiDeleteUser
msiAddUserToGroup
msiSetDefaultResc
msiSetRescSortScheme
msiSysReplDataObj
msiStageDataObj
msiSetDataObjPreferredResc
msiSetDataObjAvoidResc
msiSortDataObj
msiSysChksumDataObj
msiSetDataTypeFromExt
msiSetNoDirectRescInp
msiSetNumThreads
msiDeleteDisallowed
msiOprDisallowed

msiDataObjCreate
msiDataObjOpen
msiDataObjClose
msiDataObjLseek
msiDataObjRead
msiDataObjWrite
msiDataObjUnlink
msiDataObjRepl
msiDataObjCopy
msiExtractNaraMetadata
msiSetMultiReplPerResc
msiAdmChangeCoreIRB
msiAdmShowIRB
msiAdmShowDVM
msiAdmShowFNM
msiAdmAppendToTopOfCoreIRB
msiAdmClearAppRuleStruct
msiAdmAddAppRuleStruct
msiGetObjType
msiAssociateKeyValuePairsToObj
msiExtractTemplateMDFFromBuf
msiReadMDTemplateIntoTagStruct
msiDataObjPut
msiDataObjGet
msiDataObjChksum
msiDataObjPhymv
msiDataObjRename
msiDataObjTrim
msiCollCreate

msiRmColl
msiReplColl
msiCollRepl
msiPhyPathReg
msiObjStat
msiDataObjRsync
msiFreeBuffer
msiNoChkFilePathPerm
msiNoTrashCan
msiSetPublicUserOpr
whileExec
forExec
delayExec
remoteExec
forEachExec
msiSleep
writeString
writeLine
writeBytesBuf
writePosInt
writeKeyValPairs
msiGetDiffTime
msiGetSystemTime
msiHumanToSystemTime
msiStrToBytesBuf
msiApplyDCMetadataTemplate
msiListEnabledMS
msiSendStdoutAsEmail
msiPrintKeyValPair

msiGetValByKey
msiAddKeyVal
assign
ifExec
break
applyAllRules
msiExecStrCondQuery
msiExecStrCondQueryWithOptions
msiExecGenQuery
msiMakeQuery
msiMakeGenQuery
msiGetMoreRows
msiAddSelectFieldToGenQuery
msiAddConditionToGenQuery
msiPrintGenQueryOutToBuffer
msiExecCmd
msiSetGraftPathScheme
msiSetRandomScheme
msiCheckHostAccessControl
msiGetLcatTime
msiGetTaggedValueFromString
msiXmsgServerConnect
msiXmsgCreateStream
msiCreateXmsgInp
msiSendXmsg
msiRcvXmsg
msiXmsgServerDisconnect
msiString2KeyValPair
msiStrArray2String
msiRdaToStdout



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Micro-services (229)

msiRdaToDataObj
msiRdaNoResults
msiRdaCommit
msiAW1
msiRdaRollback
msiRenameLocalZone
msiRenameCollection
msiAclPolicy
msiRemoveKeyValuePairsFromObj
msiDataObjPutWithOptions
msiDataObjReplWithOptions
msiDataObjChecksumWithOptions
msiDataObjGetWithOptions
msiSetReServerNumProc
msiGetStdoutInExecCmdOut
msiGetStderrInExecCmdOut
msiAddKeyValToMspStr
msiPrintGenQueryInp
msiTarFileExtract
msiTarFileCreate
msiPhyBundleColl
msiWriteRodsLog
msiServerMonPerf
msiFlushMonStat
msiDigestMonStat
msiSplitPath
msiGetSessionVarValue
msiAutoReplicateService

msiDataObjAutoMove
msiGetContInxFromGenQueryOut
msiSetACL
msiSetRescQuotaPolicy
msiPropertiesNew
msiPropertiesClear
msiPropertiesClone
msiPropertiesAdd
msiPropertiesRemove
msiPropertiesGet
msiPropertiesSet
msiPropertiesExists
msiPropertiesToString
msiPropertiesFromString
msiRecursiveCollCopy
msiGetDataObjACL
msiGetCollectionACL
msiGetDataObjAVUs
msiGetDataObjPSmeta
msiGetCollectionPSmeta
msiGetDataObjAIP
msiLoadMetadataFromDataObj
msiExportRecursiveCollMeta
msiCopyAVUMetadata
msiGetUserInfo
msiGetUserACL
msiCreateUserAccountsFromDataObj
msiLoadUserModsFromDataObj

msiDeleteUsersFromDataObj
msiLoadACLFromDataObj
msiGetAuditTrailInfoByUserID
msiGetAuditTrailInfoByObjectID
msiGetAuditTrailInfoByActionID
msiGetAuditTrailInfoByKeywords
msiGetAuditTrailInfoByTimeStamp
msiSetDataType
msiGuessDataType
msiMergeDataCopies
msilsColl
msilsData
msiGetCollectionContentsReport
msiGetCollectionSize
msiStructFileBundle
msiCollectionSpider
msiFlagDataObjwithAVU
msiFlagInfectedObjs



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



State Information - How Many?

ZONE_ID
ZONE_NAME
ZONE_TYPE
ZONE_CONNECTION
ZONE_COMMENT
ZONE_CREATE_TIME
ZONE_MODIFY_TIME
USER_ID
USER_NAME
USER_TYPE
USER_ZONE
USER_DN
USER_INFO
USER_COMMENT
USER_CREATE_TIME
USER_MODIFY_TIME
RESC_ID
RESC_NAME
RESC_ZONE_NAME
RESC_TYPE_NAME
RESC_CLASS_NAME
RESC_LOC
RESC_VAULT_PATH
RESC_FREE_SPACE
RESC_FREE_SPACE_TIME

RESC_INFO
RESC_COMMENT
RESC_CREATE_TIME
RESC_MODIFY_TIME
RESC_STATUS
DATA_ID
DATA_COLL_ID
DATA_NAME
DATA_REPL_NUM
DATA_VERSION
DATA_TYPE_NAME
DATA_SIZE
DATA_RESC_GROUP_NAME
DATA_RESC_NAME
DATA_PATH
DATA_OWNER_NAME
DATA_OWNER_ZONE
DATA_REPL_STATUS
DATA_STATUS
DATA_CHECKSUM
DATA_EXPIRY
DATA_MAP_ID
DATA_COMMENTS
DATA_CREATE_TIME
DATA_MODIFY_TIME

DATA_ACCESS_TYPE
DATA_ACCESS_NAME
DATA_TOKEN_NAMESPACE
DATA_ACCESS_USER_ID
DATA_ACCESS_DATA_ID
COLL_ID
COLL_NAME
COLL_PARENT_NAME
COLL_OWNER_NAME
COLL_OWNER_ZONE
COLL_MAP_ID
COLL_INHERITANCE
COLL_COMMENTS
COLL_CREATE_TIME
COLL_MODIFY_TIME
COLL_ACCESS_TYPE
COLL_ACCESS_NAME
COLL_TOKEN_NAMESPACE
COLL_ACCESS_USER_ID
COLL_ACCESS_COLL_ID
META_DATA_ATTR_NAME
META_DATA_ATTR_VALUE
META_DATA_ATTR_UNITS
META_DATA_ATTR_ID
META_DATA_CREATE_TIME



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



State Information (112)

META_DATA_MODIFY_TIME	RULE_EXEC_REI_FILE_PATH	SL_HOST_NAME
META_COLL_ATTR_NAME	RULE_EXEC_USER_NAME	SL_RESC_NAME
META_COLL_ATTR_VALUE	RULE_EXEC_ADDRESS	SL_CPU_USED
META_COLL_ATTR_UNITS	RULE_EXEC_TIME	SL_MEM_USED
META_COLL_ATTR_ID	RULE_EXEC_FREQUENCY	SL_SWAP_USED
META_NAMESPACE_COLL	RULE_EXEC_PRIORITY	SL_RUNQ_LOAD
META_NAMESPACE_DATA	RULE_EXEC_ESTIMATED_EXE_TIME	SL_DISK_SPACE
META_NAMESPACE_RESC	RULE_EXEC_NOTIFICATION_ADDR	SL_NET_INPUT
META_NAMESPACE_USER	RULE_EXEC_LAST_EXE_TIME	SL_NET_OUTPUT
META_RESC_ATTR_NAME	RULE_EXEC_STATUS	SL_CREATE_TIME
META_RESC_ATTR_VALUE	TOKEN_NAMESPACE	SLD_RESC_NAME
META_RESC_ATTR_UNITS	TOKEN_ID	SLD_LOAD_FACTOR
META_RESC_ATTR_ID	TOKEN_NAME	SLD_CREATE_TIME
META_USER_ATTR_NAME	TOKEN_VALUE	
META_USER_ATTR_VALUE	TOKEN_VALUE2	
META_USER_ATTR_UNITS	TOKEN_VALUE3	
META_USER_ATTR_ID	TOKEN_COMMENT	
RESC_GROUP_RESC_ID	AUDIT_OBJ_ID	
RESC_GROUP_NAME	AUDIT_USER_ID	
USER_GROUP_ID	AUDIT_ACTION_ID	
USER_GROUP_NAME	AUDIT_COMMENT	
RULE_EXEC_ID	AUDIT_CREATE_TIME	
RULE_EXEC_NAME	AUDIT_MODIFY_TIME	



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



iRODS Installation

Client

Server

Metadata Catalog



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Accounts Set Up by Academia Sinica

- Your user name will be
stuXX where XX is a number from 01 to 50
- Your password will be
lsgCXX where XX is a number from 01 to 50

[http://www2.twgrid.org/APTeam/index.php/
20110319_iRODS_Workshop](http://www2.twgrid.org/APTeam/index.php/20110319_iRODS_Workshop)

Windows Installation

- From the URL <https://www.irods.org/index.php/windows> go to the section labeled Windows i-Commands and click on the file
Windows i-commands 2.4d
- This will download the file
win_icmds_2_4.zip
- Uncompress the file



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Detailed Windows Install

- Extract the exe files. This will be a long list of separate executable commands, one for each type of operation that you may need to perform. The list will include:
 - `iadmin` - used by the data grid administrator to set up resources and accounts
 - `icd` - change to a different directory in the data grid
 - `ils` - list files in a data grid directory
- To use these icommands, you will need to set up an environment variable file which has default settings for the data grid that the class will use.
- Note the directory name where you have put the executables



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Detailed Windows Install

- On the URL <https://www.irods.org/index.php/windows> there are instructions in the section labeled

"Setting up the iRODS User Environment file in Windows (for i-commands only)"

- To create the `.irodsEnv` file:
 - * Launch a "Command Prompt" by navigating to the menu "Start" -> "Accessories" -> "Command Prompt".
 - * Change directory to the user home directory.
 - > `cd %HOMEDRIVE%%HOMEPATH%`
 - * Type the following Windows command to create a folder, ".irods", and move into this directory.
 - > `md .irods`
 - > `cd .irods`
 - > `Notepad .irodsEnv`
- (This will launch a Notepad and create a text file named ".irodsEnv".)



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Detailed Windows Install

- Enter the following information into Notepad and click save.

```
irodsHost 'irods05.grid.sinica.edu.tw'
```

```
irodsPort 1247
```

```
irodsHome '/tempZone/home/stuXX' (where XX is 01-50)
```

```
irodsUserName 'stuXX' (where XX is 01-50)
```

```
irodsZone 'tempZone'
```

- These are the Environment variables for a rods_user account on the renci data grid.
- You will need to replace the three occurrences of “user_name” with your iRODS account name
- Your password is IsgCXX where XX is 01-50



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Detailed Windows Install

- To run i-commands in any directory in a Windows machine, the path to where i-commands reside should be set in the Windows PATH environment variable.
- To do this, launch the System dialogue via:
 - * Start -> settings -> control panel.
 - * Click the "System" icon.
 - * In the "Advanced" tab, click the "Environment variables" button.
- Add the path name for the i-commands directory to the "PATH" either in user category or the system category. The path name can be found from the window that shows the icommand executables. Add a semi-colon and this path name to the end of the PATH text.
- Then close the window and start a new command prompt window. You will be able to execute the icommands from any directory on your system.



Detailed Windows Install

- To connect to the data grid, type
`iinit`
You will be prompted for your password.
- To change your password, type
`ipasswd`
You will be prompted for your current password
You will then be asked for the new password



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



iRODS Installation

- <http://irods.diceresearch.org>
 - Downloads
 - BSD license
 - Registration / agreement
 - Tar file
 - Installation script (Linux, Solaris, Mac OSX)
 - Automated download of PostgreSQL, ODBC
 - Installation of PostgreSQL, ODBC, iRODS
 - Initiation of iRODS collection



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



iRODS Installation

- Unpack the release tar file
 - `gzip -d irods.tar`
 - `tar xf irods.tar`
- `cd` into the top directory and execute
 - `./irodssetup`
- It will prompt for a few parameters



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



irodssetup

- Set up iRODS
- -----
- iRODS is a flexible data archive management system that supports many different site configurations. This script will ask you a few questions, then automatically build and configure iRODS.
- There are four main components to iRODS:
 - 1. An iRODS server that manages stored data.
 - 2. An iCAT catalog that manages metadata about the data.
 - 3. A database used by the catalog.
 - 4. A set of 'i-commands' for command-line access to your data.
- You can build some, or all of these, in a few standard configurations. For new users, we recommend that you build everything.



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



iRODS Client Installation

- iRODS configuration setup
- -----
- This script prompts you for key iRODS configuration options.
- Default values (if any) are shown in square brackets [] at each prompt. Press return to use the default, or enter a new value.

- For flexibility, iRODS has a lot of configuration options. Often
- the standard settings are sufficient, but if you need more control
- enter yes and additional questions will be asked.

- Include additional prompts for advanced settings [no]?



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



iRODS Client Installation

- iRODS configuration (advanced)
- -----
- iRODS consists of clients (e.g. i-commands) with at least one iRODS server. One server must include the iRODS metadata catalog (iCAT).
- For the initial installation, you would normally build the server with the iCAT (an iCAT-Enabled Server, IES), along with the i-commands.
- After that, you might want to build another Server to support another storage resource on another computer (where you are running this now).
- You would then build the iRODS server non-ICAT, and configure it with the IES host name (the servers connect to the IES for ICAT operations).
- If you already have iRODS installed (an IES), you may skip building the iRODS server and iCAT, and just build the command-line tools.
- Build an iRODS server [yes]? no



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



iRODS Client Installation

- iRODS can make use of the Grid Security Infrastructure (GSI)
- authentication system in addition to the iRODS secure
- password system (challenge/response, no plain-text).
- In most cases, the iRODS password system is sufficient but
- if you are using GSI for other applications, you might want
- to include GSI in iRODS. Both the clients and servers need
- to be built with GSI and then users can select it by setting
- `irodsAuthScheme=GSI` in their `.irodsEnv` files (or still use
- the iRODS password system if they want).

- Include GSI [no]? no



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



iRODS Client Installation

- Confirmation
- -----
- Please confirm your choices.
- -----
- GSI not selected
- -----
- Build iRODS command-line tools
- -----
- Save configuration (irods.config) [yes]?
- Saved.
- Start iRODS build [yes]?



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



iRODS Client Installation

- Build and configure
- -----
- Preparing...
- Configuring iRODS...
-
- Step 1 of 4: Enabling modules...
- properties
-
- Step 2 of 4: Verifying configuration...
- No database configured.
-
- Step 3 of 4: Checking host system...
- Host OS is Mac OS X.
- Perl: /usr/bin/perl
- C compiler: /usr/bin/gcc (gcc)
- Flags: none
- Loader: /usr/bin/gcc
- Flags: none
- Archiver: /usr/bin/ar
- Ranlib: /usr/bin/ranlib
- 64-bit addressing not supported and automatically disabled.



iRODS Client Installation

- Step 4 of 4: Updating configuration files...
 - Updating config.mk...
 - Created /storage-site/iRODS/config/config.mk
 - Updating platform.mk...
 - Created /homs/sdc/iRODS/config/platform.mk
 - Updating irods.config...
 - Updating irodsctl...
- Compiling iRODS...
 - Step 1 of 2: Compiling library and i-commands...
 - Step 2 of 2: Compiling tests...
- Done!



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



iRODS Client Installation

- -----
- To use the iRODS command-line tools, update your PATH:
- For csh users:
 - `set path=(/storage-site/iRODS/clients/icommands/bin $path)`
- For sh or bash users:
 - `PATH=/storage-site/iRODS/clients/icommands/bin:$PATH`
- Please see the iRODS documentation for additional notes on how to manage the servers and adjust the configuration.
- Change the path name to your installation path



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



.irodsEnv file

```
irodsHost 'irods05.grid.sinica.edu.tw'  
irodsPort 1247  
irodsHome '/tempZone/home/stuXX'  
irodsUserName 'stuXX'  
irodsZone 'tempZone'
```

Your password is lsgCXX

Replace XX with a number from 01 to 50 in both lsgCXX and stuX



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Full Install

- iRODS configuration:
- -----
- Build an iRODS server? (yes/no) yes
- Include an iCAT catalog? (yes/no) yes
- For security reasons, the build process will create a new iRODS administrator account named 'rods' for managing the system.
- Enter a new password for the iRODS account?
(password) xxxxxx



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Input Parameters

- Database configuration:
- -----
- The iCAT uses a database to store metadata. You can build and configure a new Postgres database now or use an existing database.
- Build Postgres? (yes/no) yes
- You can select the directory for Postgres:
 - If you are creating a new iRODS installation, select a new directory. Postgres will be automatically downloaded, built, and installed there.
 - If you are upgrading an iRODS installation and wish to re-use an existing database, enter the path to that Postgres directory.
- Where should Postgres be installed? (directory path) /Astorage-site/Postgres
- For security reasons, the new database will create an administrator account for 'reaganmoore' and assign a password.
- Enter a password for the new database account? (password) xxxxxxxx



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Check Input Parameters

- The iRODS build and setup is ready to begin.
- iRODS server: build
 - account 'demo'
 - password 'demo'
 - path '/home/sdsc/iRODS'
- iCAT catalog: build
- Postgres: install a new database
 - enable iRODS scripts to start/stop database
 - account 'DBadmin'
 - password 'UKdemo'
 - path '/storage-site/irods/postgresql'
- I-commands: build
- Ready? (yes/no) yes



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Installation

- Track the completion status of each step:
- Preparing...
- Installing Postgres database...
 - Step 1 of 4: Preparing to install...
 - Step 2 of 4: Installing Postgres... About 11 minutes
 - Step 3 of 4: Installing UNIX ODBC... About 26 minutes
 - Step 4 of 4: Setting up Postgres...
 - Step 5 of 4: Setting up iRODS...
- Configuring iRODS... **About 1 minute**
 - Step 1 of 5: Enabling modules...
 - Step 2 of 5: Verifying configuration...
 - Step 3 of 5: Checking host system...
 - Step 4 of 5: Updating configuration files...
 - Step 5 of 5: Cleaning out previously compiled files...
- Compiling iRODS... **About 3 minutes**
 - Step 1 of 3: Compiling library and i-commands...
 - Step 2 of 3: Compiling iRODS server...
 - Step 3 of 3: Compiling tests...



iRODS Source Distribution

- INSTALL.txt
 - LICENSE.txt
 - MANAGE.txt
 - README.txt

 - Makefile
 - irodsctl
 - irodssetup
 - irodsupgrade
- COPYRIGHT
 - CVS
 - bin
 - clients
 - config
 - Doc
 - installLogs
 - jargon
 - lib
 - Modules
 - nt
 - runDoxygen.rb
 - scripts
 - server



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



User Configuration

- To use the iRODS 'i-commands', update your PATH:
- For csh users:
 - set path=(/storage-site/iRODS/clients/icommands/bin \$path)
- For sh or bash users:
 - ./add-clients.sh
 - PATH=/storage-site/iRODS/clients/icommands/bin:\$PATH
- To start and stop the servers, use 'irodsctl':
 - irodsctl start
 - irodsctl stop
 - irodsctl restart
- Add '--help' for a list of commands.



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



irodsctl options

- Usage is:
 - /storage-site/iRODS/scripts/perl/irodsctl.pl [options]
[commands]
- Help options:
 - --help Show this help information
- Verbosity options:
 - --quiet Suppress all messages
 - --verbose Output all messages (default)
- iRODS server Commands:
 - istart Start the iRODS servers
 - istop Stop the iRODS servers
 - irestart Restart the iRODS servers



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



irodsctl options

- Database commands:
 - dbstart Start the database servers
 - dbstop Stop the database servers
 - dbrestart Restart the database servers
 - dboptimize Optimize the iRODS tables in the database
 - dbvacuum Same as 'optimize'
- General Commands:
 - start Start the iRODS and database servers
 - stop Stop the iRODS and database servers
 - restart Restart the iRODS and database servers
 - status Show the status of iRODS and database servers
 - devtest Run a developer test suite
 - loadtest Run a concurrency (load/pound) test suite



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Environment Variables

- In home directory
 - `cd ~/.irods`
 - `vi .irodsEnv`



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



User Interfaces



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



User Interfaces

- Hands-on use of
 - iCommands – Unix shell commands
 - Web browser - Firefox or Internet Explorer
 - iDrop - Dropbox style interface
 - Windows - Windows browser
- Demonstration of FUSE file system interface

iRODS i-Commands

Unix Shell

i-Commands

- iRODS shell commands similar to Unix
 - Change the working directory icd
 - Set access permissions ichmod
 - Copy between directories icp
 - List files ils
 - Move a file between directories imv
 - Change your password ipasswd
 - Display active connections ips
 - Remove a file irm
 - Make a directory imkdir
 - Print current working directory ipwd

Unique iRODS i-Commands

- List all i-Commands `ihelp`
- Initialize access (authenticate) `iinit`
- Exit from data grid `iexit`
- Put a file into the data grid `iput`
- Get a file from the data grid `iget`
- Physically move a file `iphymv`
- Upload tar files `ibun`
- Replicate a file `irepl`
- Trim replicas `itrim`
- Remove files from trash `irmtrash`
- Register a file `ireg`
- Check whether local file is registered `iscan`
- List resources `ilsresc`

iRODS i-Commands

- Rules

- Execute a rule `irule`
- List status of delayed rules `iqstat`
- Delete a delayed rule `iqdel`

- Metadata

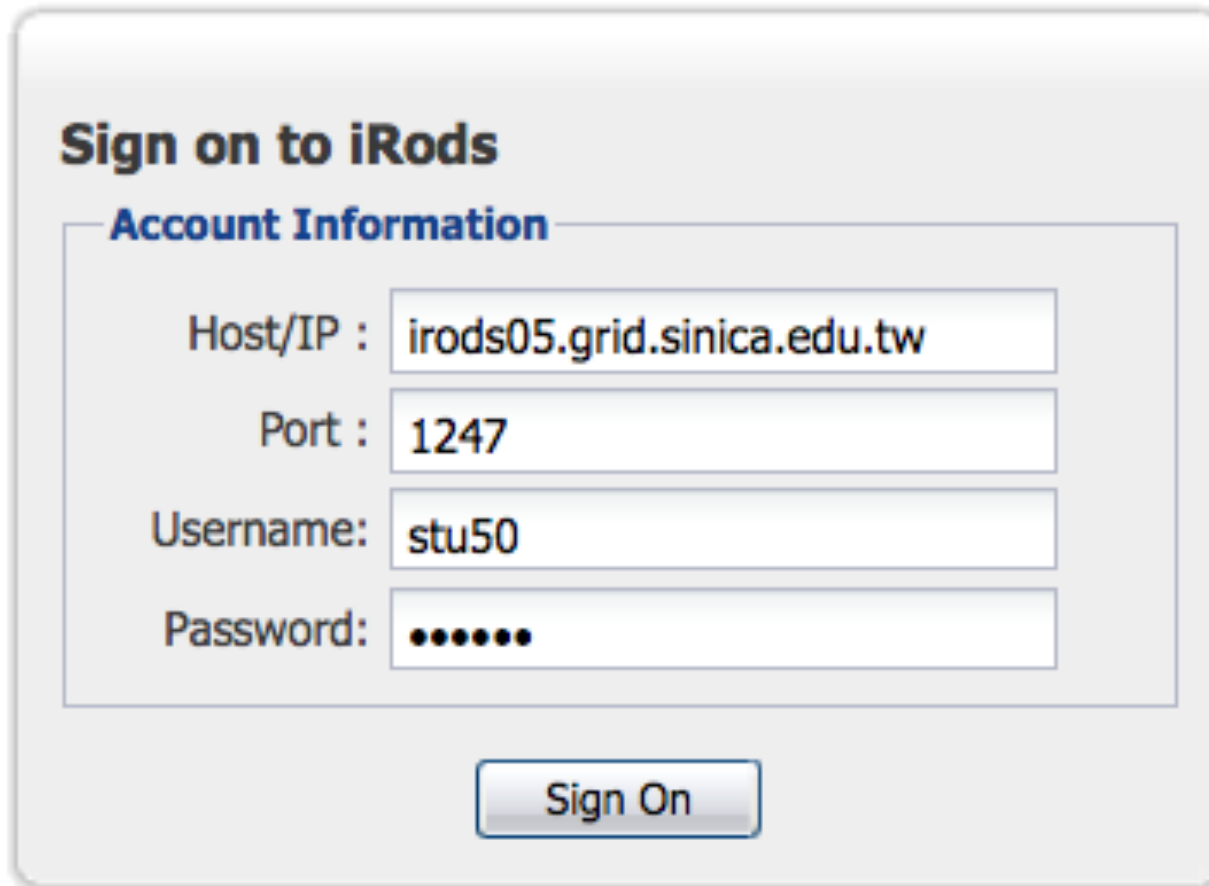
- Add metadata `imeta`
- Query the metadata catalog `iquest`
- Show system metadata `isysmeta`
- List user information `iuserinfo`
- List server information `imiscsvrinfo`

- Messaging

- Send/receive messages `ixmsg`

Rich Web Client – Web Browser

- <https://www.irods.org/web/index.php>



Sign on to iRods

Account Information

Host/IP :

Port :

Username:

Password:

Rich Web Client Browser

Address bar: rods://rods@iren.renci.org:1247/renCI/home/rods/rules

Browser tabs: irods.org | https://www.irods.org/web/browse.php#ruri=rods@ire

Navigation: Most Visited | Getting Started | Latest Headlines

Current path: rods://rods@iren.renci.org:1247/r...

User: rods@iren.renci.org:1247 | [Sign Out](#)

Collections

- richardsonb
- rlopez
- rods
 - ANT1scripts
 - Backup
 - Publication
 - State_Amelia_Earhart
 - Test3
 - arch
 - archive
 - datanet
 - demo
 - dvn2irods
 - irodsbook
 - loading
 - logs
 - looptest
 - monitoring
 - nsfdemo1
 - nvo
 - other
 - rules**
 - ruletest
 - tarfiles
 - test
 - tg
- rods#RENCI_VO
- rods#TDLC
- rods#ncdc
- rods#oici
- rods#sdscSpatial

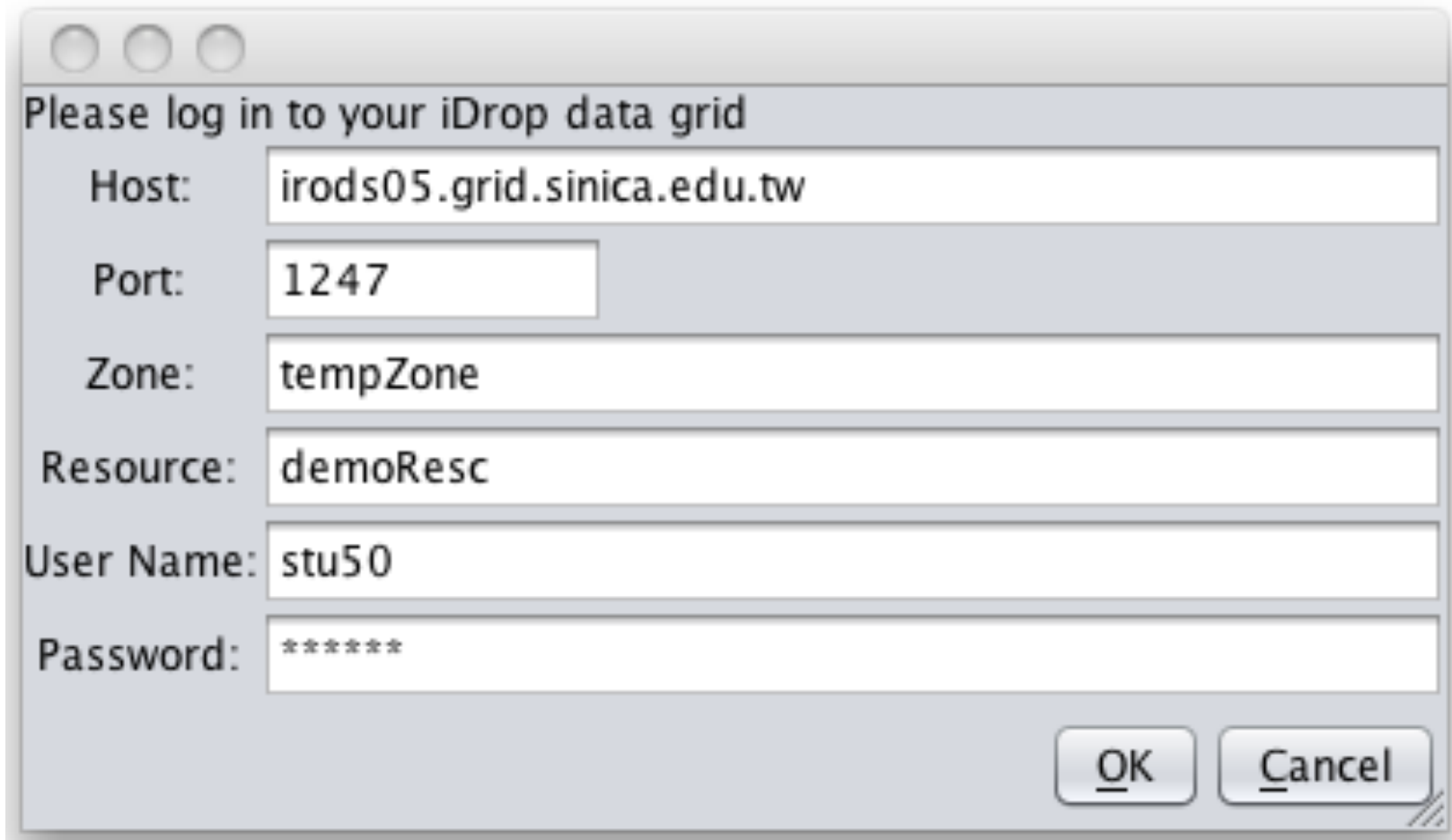
File List

Name	Resource	Size	Date Modified
d1.r	renCI-vault1	3.42 KB	June 2, 2010, 12:02 p
d1.r	renCI-vault2	3.42 KB	May 14, 2010, 4:57 p
showcore.ir	renCI-vault1	50 B	May 12, 2010, 9:40 a
.irodsEnv	renCI-vault1		January 14, 2011, 11:09 a
fits.tag	renCI-vault1	158 B	November 3, 2010, 2:29 p
rosat_pspc_rdf2_3_bk1.fits	renCI-vault1	523.13 KB	November 3, 2010, 2:20 p
SAA-award.jpg	renCI-vault1	48.67 KB	November 3, 2010, 9:12 a
ruletest.r	renCI-vault1	95 B	February 17, 2010, 11:54 a
listMS.ir	renCI-vault1	108 B	October 29, 2009, 8:16 a
sample.email	renCI-vault1	627 B	August 3, 2009, 1:00 p

Page 1 of 1 | Displaying objects 1 - 10 of 1

iDrop Interface

- <http://irendb.renci.org:8080/llclient/launchLLDrop.html>



Please log in to your iDrop data grid

Host:

Port:

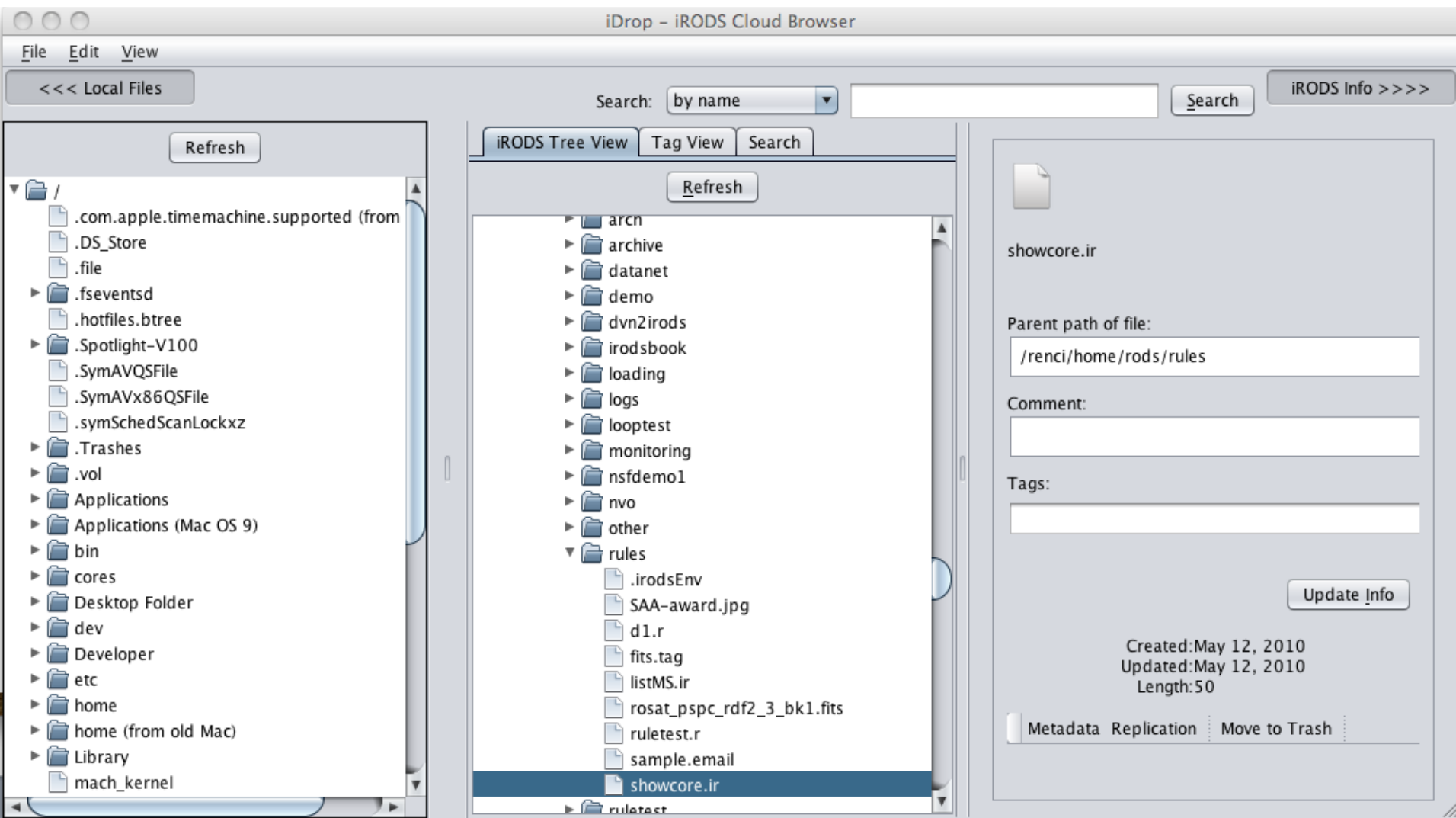
Zone:

Resource:

User Name:

Password:

iDrop Interface



Windows iExplorer

<https://www.irods.org/index.php/windows>

iRODS Login

Name

Host


Zone

Port

Previous Logins:

Name	Zone	Server Host
rods	renci	iren.renci.org
rwmoore	lifelibZone	diamond.ils.unc.edu
stu50	tempZone	irods05.grid.sinica...

To login, enter the password for the selected connection



www.irods.org

Windows iExplorer

The screenshot shows the iRODS Explorer application window. The title bar reads "iRODS Explorer - iRODS Explorer". The menu bar includes "iRODS", "Edit", "View", "Rule", and "Help". The toolbar contains various icons for file operations. The "Storage Resource" is set to "renci-vault1".

The left sidebar shows a tree view with the following structure:

- /
- renci
- home
- rods

The main pane displays a table of files and folders. The table has the following columns: Name, Replica, Size, Owner, Modified Time, Repl. Status, Resource, and Resource Group.

Name	Replica	Size	Owner	Modified Time	Repl. Status	Resource	Resource Group
ANT1scripts							
Backup							
Publication							
State_Amelia_Earhart							
Test3							
arch							
archive							
datanet							
demo							
dvn2irods							
irodsbook							
loading							
logs							
looptest							
monitoring							
nsfdemo1							
nvo							
other							
rules							
ruletest							
tarfiles							
test							
tg							
.DS_Store	0	12.00 KB	rods	2010-12-04.01:04:44	1	renci-vault1	
00008.pdf	0	61.38 KB	rods	2010-12-04.08:11:09	1	renci-vault1	
00008.pdf	1	61.38 KB	rods	2010-12-06.19:09:15	1	renci-vault2	
00008.pdf	2	61.38 KB	rods	2010-12-06.19:09:16	1	renci-vault4	
100MB_testfile.tar	0	100.04 MB	rods	2010-06-30.15:45:16	1	renci-vault1	
253MB_testfile.tif	0	253.00 MB	rods	2010-06-29.23:08:21	1	renci-vault1	
Proposal.pdf	0	784.05 KB	rods	2010-01-23.13:39:14	1	renci-vault1	
antlr-2.7.7.jar	0	449.34 KB	rods	2010-12-04.01:04:28	1	renci-vault1	
jms-1.1.jar	0	30.26 KB	rods	2010-09-17.13:13:51	1	renci-vault1	
nara-ICAT-backup.gz	0	906.88 MB	rods	2010-04-21.17:00:58	1	renci-vault1	

The status bar at the bottom right indicates "sub-collections: 23, files: 10".

Executing a Rule



The image shows a dialog box titled "Submit an iRODS Rule Dialog" with a close button in the top right corner. The dialog contains three text input fields and three buttons. The first field, labeled "Rule:", contains the text "myTest||msiAdmShowIRB(*A)|nop". To its right is an "Import..." button. The second field, labeled "Input Parameters:", contains the text "null". The third field, labeled "Output Parameters:", contains the text "*A%ruleExecOut". At the bottom right of the dialog are "Submit" and "Close" buttons.

Submit an iRODS Rule Dialog

Rule: myTest||msiAdmShowIRB(*A)|nop Import...

Input Parameters: null

Output Parameters: *A%ruleExecOut

Submit Close

FUSE File System Interface

- Based on FUSE environment
 - Mac, Solaris, Unix
- Instructions for installing iRODS-FUSE driver
 - https://www.irods.org/index.php/iRODS_FUSE
- Mount iRODS directory as local directory
 - `mkdir ~/fmount`
 - `irodsFs ~/fmount`
- Can then apply local unix shell commands on remote iRODS directory

Current Release Features

Features in iRODS 2.5

- Table driven resources
- SQL-based queries
- DDN WOS Support
- Non-blocking driver for Unix file system
- Fortran I/O library
- Xmessage system enhancements
- Network transport enhancements

Table Driven Resources

- **Database Resources** (table-driven resources, external databases), initial version. This initial version of the Database Resources feature provides some of the foundational functionality for accessing external databases (a.k.a. table driven resources), including multi-tiered access control, SQL based queries and updates, data-object SQL definitions, XML-like result streams, and server-side redirection of result streams to data-objects.

Database Resources

- A 'database resource' (**DBR**) is a database (a 'schema' in Oracle terminology, 'database' in PostgreSQL) instance (or similar tabular information) that can be queried and updated via SQL statements (or other, for non-SQL)
- A database object (**DBO**) is a interface to a set of tables, typically a query that returns results. The 'database objects' contain SQL statements for RDBMSes. Note that the Agent will Open and Close the database as needed, the caller will be just getting information (results of the query) and returning it or storing it.
- In addition to returning the results of a DBO execution (on a DBR) directly to the iRODS client, the system will also optionally store the results in a specific text format into an iRODS data object, a DBO Results file (**DBOR**).
- iRODS access controls are applied on the DBR and DBO.

idbo command

- A single command can be entered on the command line or, if blank, it will go into interactive mode and prompt for commands.
- Commands are:
 - open DBR (open a database resource)
 - close DBR (close a database resource)
 - exec DBR DBO [arguments] (execute a DBO on a DBR)
 - output [-f] DBOR (store 'exec' results in another data-object)
 - commit DBR (commit updates to a DBR (done via a DBO))
 - rollback DBR (rollback updates instead)
 - ls (list defined Database-Objects in the Zone)
 - help (or h) [command] (this help, or help on a command)
 - quit (or 'q', exit idbo)
- Where DBR and DBO are the names of a Database Resource and Database Object.

Access Controls

- iRODS administrators can create DBOs, since they can give anyone (including themselves) 'write' access to the DBR.
- iRODS users with 'write' access to the DBR will also be allowed to create DBOs.
- iRODS users with 'read' access to the DBR will be allowed to execute DBOs that were created by users with 'write' access to the same DBR. The 'read' users, for some DBO SQL, will provide parameters to include in the SQL, which will be executed as SQL bind variables (to restrict capabilities). This access mode will allow more privileged users to create controlled access for additional users.

SQL-based Queries

- The administrator can add SQL strings that can then be invoked by users of the data grid
 - `iadmin asq 'query-string' alias-name`
- The string can be executed with `iquest`
 - `iquest --sql alias-name`

Example query

- To generate string, turned on debugging
 - Set environment variable `irodsDebug` to `CATSQL`
 - Turned on `spLogSql` in `irodsctl.pl`
- Issued `iquest` query
 - `iquest "select sum(DATA_SIZE)"`
- Looked in `server/log/rodsLog.2011.3.11` for the query
 - `select distinct sum(R_DATA_MAIN.data_size) from R_DATA_MAIN`

Example Query

- Added the query string as an iRODS administrator
 - iadmin asq 'select distinct sum (R_DATA_MAIN.data_size) from R_DATA_MAIN' size
 - Note that the alias “size” is defined for the string
- Can now execute this query from iquest
 - iquest --sql size

Storage System Drivers

Compound Resource



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



WOS Driver

- Implemented as a compound resource with a unix disk cache in front of the “get – put – delete” WOS interface
- Create WOS resource type
 - iadmin at resc_type wos
- Create compound resource
 - iadmin mkresc wosResc wos compound “IP-address-iRODS-server” “IP-address-WOS-Resource”/”WOSPolicy-file”

HPSS Drivers

- Implemented as compound resources
- Native HPSS driver
 - Supports parallel I/O
- Universal Mass Storage System driver
 - Developed by IN2P3
- File system interface to HPSS
 - Integration through GPFS

HPSS Documentation

- iRODS Wiki
- https://www.irods.org/index.php/HPSS_Resource

1) Edit the config/config.mk file:

- Uncomment the line HPSS=1, e.g.,
 - HPSS=1
- If you are running HPSS v7 or higher, Uncomment the line HPSS7=1, e.g.,
 - HPSS7=1
- Define the HPSS_LIB_DIR (the hpss libraries directory) and HPSS_HDR_DIR (the hpss header directory). e.g.,
 - HPSS_LIB_DIR=/opt/hpss/lib HPSS_
 - HDR_DIR=/opt/hpss/include

The HPSS driver supports 3 HPSS authentication mode - UNIX password, UNIX keytab and Kerberos keytab. The default mode is UNIX keytab and no action is needed for this mode. If the the UNIX password authentication is to be used, uncomment the line

- HPSS_UNIX_PASSWD_AUTH=1
- If the the Kerberos authentication is to be used, uncomment the line
 - HPSS_KRB5_AUTH=1

2) cd to the iRODS home directory and type in "make" to re-make the server.

HPSS Configuration

- **1) cd to the server/config directory and use the template files hpssAuth.template and hpssCosConfig.template for the hpssAuth and hpssCosConfig files. e.g.,**
 - cp hpssAuth.template hpssAuth
 - cp hpssCosConfig.template hpssCosConfig
- Edit the hpssAuth and hpssCosConfig files according to the instructions given in these files.
 - The hpssAuth file configures the HPSS authentication for the driver
 - The hpssCosConfig configures the COS (class of services) for the driver.

HPSS Configuration

- **2) Create an HPSS Resource**
- The HPSS driver is implemented as a compound class resource because of the parallel transfer mode of HPSS. As explained in [resource](#), the compound resource implementation requires a cache class resource to be configured in the same resource group as the compound resource. Data stored in the compound resource cannot be accessed directly but only through the cache resource. The following gives an example of creating an HPSS resource using iadmin:
 - `iadmin mkresc hpssResc hpss compound nacho.sdsc.edu /home/irods/Vault`
- Note: If the resource type "hpss" does not already exist in the iCAT, you may need to run this command to generate one before running "iadmin mkresc":
 - `iadmin at resc_type hpss`

HPSS Configuration

- **3) Add the HPSS and cache resources to a resource group. e.g.,**
 - iadmain atrg myrescGroup hpssResc
 - iadmain atrg myrescGroup cacheResc
- Note: the cacheResc resource must be on a HPSS enabled server and does not have to be on the same host as the hpssResc. This way, multiple cache resources on different hosts can be used as the front-end for the HPSS resource.

Workflow Integration

Sreekanth Pothanis

LSU



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

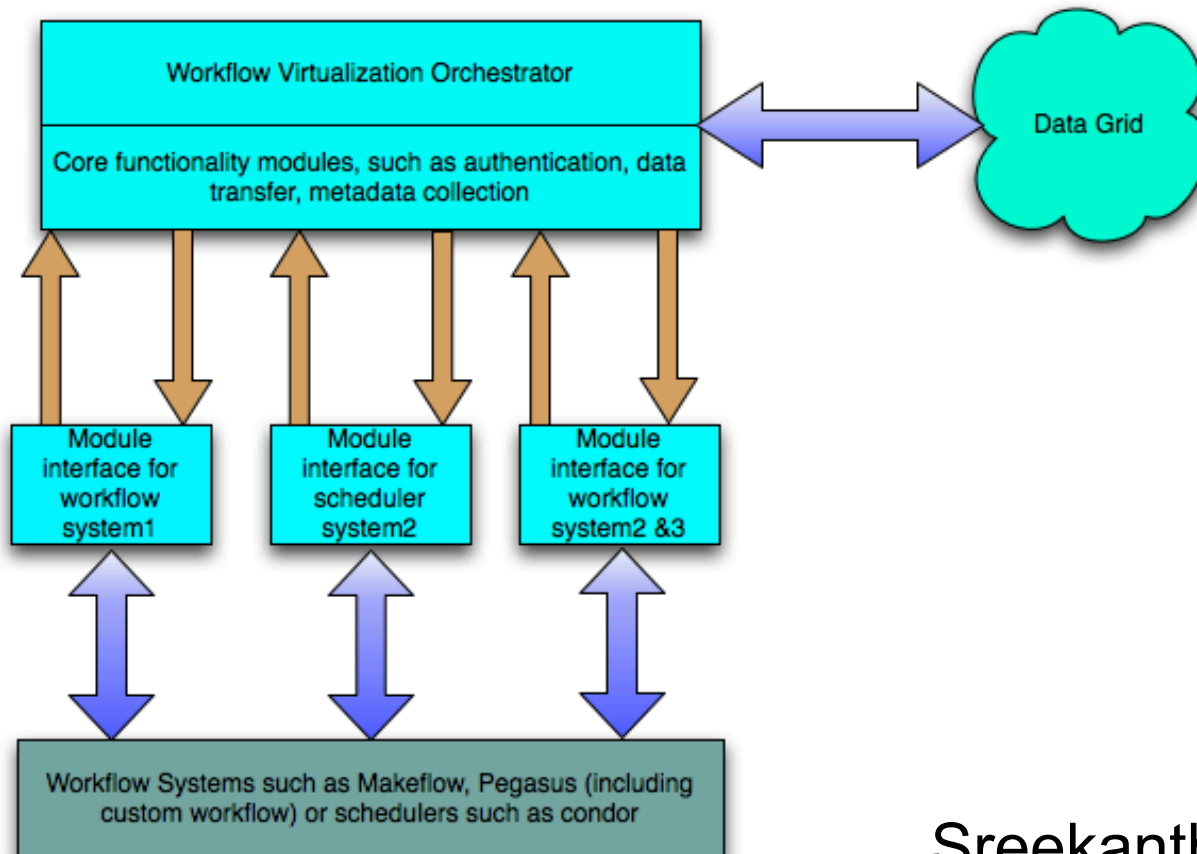


Workflow Virtualization

- Management of a processing pipeline
- Manage interactions with each workflow system for input and output of files.
- Provides higher control
- Enables execution of complex workflows spanning multiple different workflow systems
- External to the environment that actually runs the workflow
 - Increases generality

Workflow Virtualization Server (WVS)

- Stand alone and modular
- External to any workflow



WVS: Authentication and Context Handling

- Handled at two levels
 - Grid level to perform grid transactions
 - OS level to execute workflows
- Data grid context
 - Provides information about data grid
 - User privileges, quotas
- Workflow context
 - Generated during the execution
 - List of output files, destination, metadata

WVS: Staging, Execution and post Processing

- Sets up the working environment before initiating the interfacing module
- Decreases execution time by pipelining where possible
- Executed by invoking appropriate modules
 - Modularity allows high level of customization
 - Provides higher control
- Handles custom post processing scenarios

Integration with iRODS

- Implemented through micro-services and rules
 - Client interface
- Client design and configuration
 - Configuration file and rules

```
WORKFLOW=MAKEFLOW
CONFIG=/tempZone/home/wfuser/
test.makeflow
INPUT=/tempZone/home/wfuser/capitol.jpg
INPUT=/tempZone/home/wfuser/local.jpg
INPUT=/tempZone/home/wfuser/meta.jpg
DEST=/tempZone/home/wfuser/test_dest/
METADATA=NAME1=VAL1
METADATA=NAME2=VAL2
```

Integration with iRODS

- Server Configuration
 - Authentication
 - Data Transfer
 - Metadata
 - Module execution
- Interacts with iRODS server as an admin

```
[MAKEFLOW] path=/usr/local/cctools/redhat5/bin/  
makeflow  
args= -T condor  
[MAKEFLOW]  
[MAKEFLOW1]  
path=/usr/local/Makeflow/bin/makeflow  
args= -p 9876  
[MAKEFLOW1]  
#[KEPLER]  
#path=path to kepler  
#args=-t -P  
#[KEPLER]  
[PEGASUS] path=/usr/local/Pegasus/Pegasus-  
plan  
path_to_sites.xml = /usr/local/Pegasus/sites.xml  
path_to_rc.data /usr/local/Pegasus/rc.data  
path_to_tc.data = /usr/local/Pegasus/tc.data  
[PEGASUS]
```

Implication of Executing Procedures at the Storage Location

Complexity Analysis



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Distributed Workflows

- When should data be processed at the remote storage location?
 - Low complexity operations
- When should data be processed at a supercomputer?
 - High complexity operations
- When should data be processed at the display?
 - Interactive presentation manipulation



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



“Ohm’s” Law for Computer Science

- Relationship between
 - Computational complexity (operations per byte)
 - Execution rate
 - Data access bandwidth

$$\eta = R / B$$

Complexity = Execution Rate / Bandwidth
for a balanced application

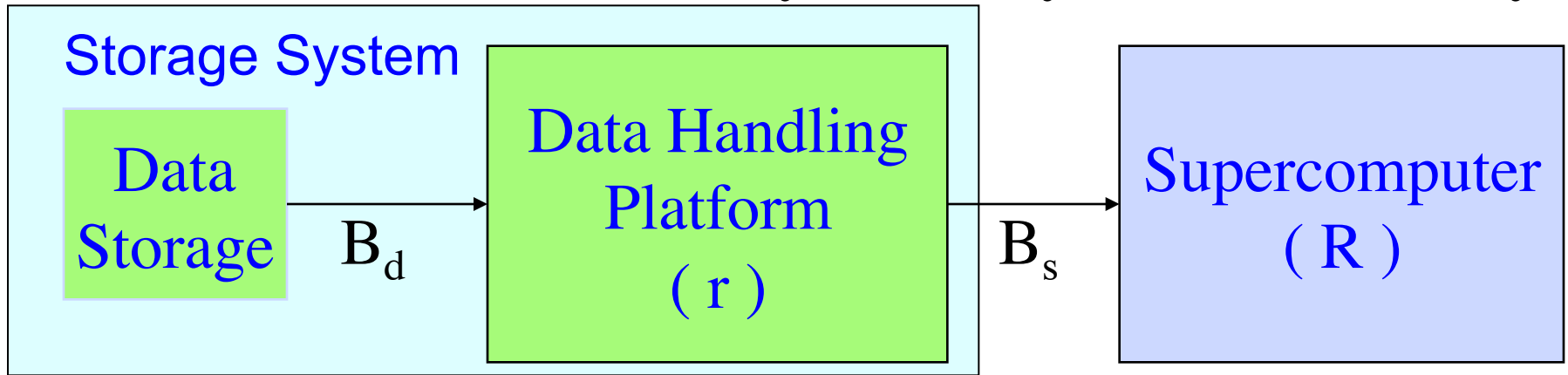


THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Data Distribution Thought Experiment

Reduce size of data from S bytes to s bytes and then analyze



Execution rates are

$$r < R$$

Bandwidths linking systems are

$$B_d > B_s$$

Operations per byte for analysis is

$$\eta_s$$

Operations per byte for data transfer is

$$\eta_t$$

Should the data reduction be done before transmission?



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

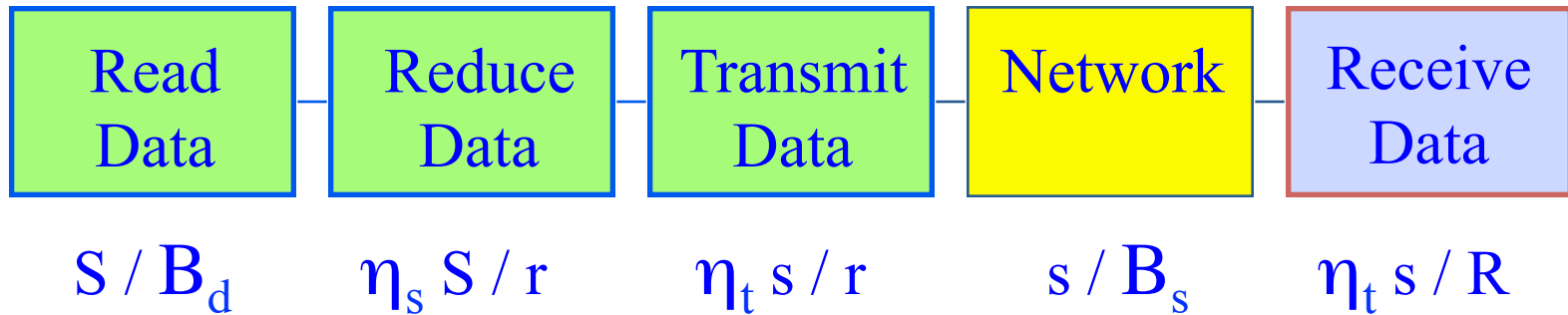


Distributing Services

Compare times for analyzing data with size reduction from S to s

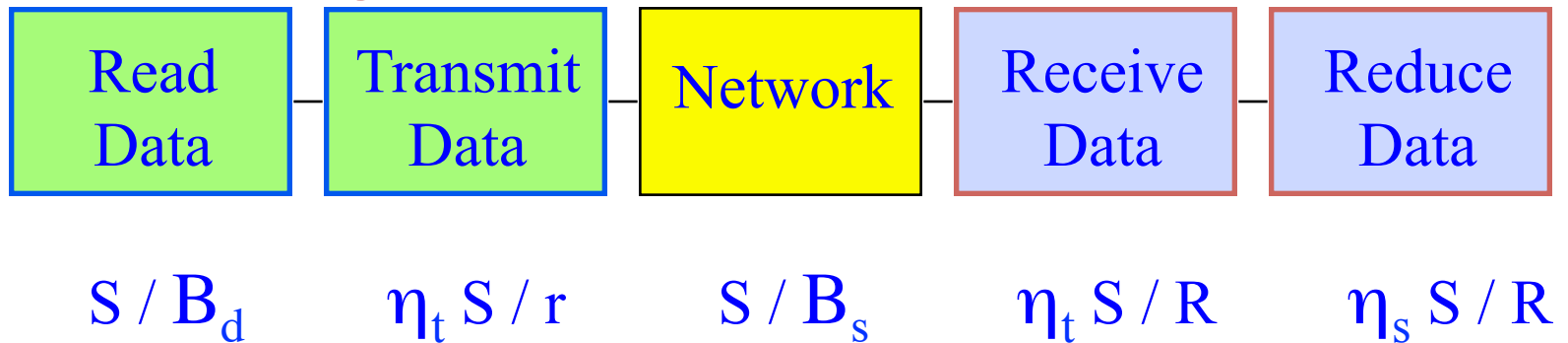
Data Handling Platform

Supercomputer



Data Handling Platform

Supercomputer



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Comparison of Time

Processing at archive

$$T(\text{Archive}) = S/ B_d + \eta_s S/r + \eta_t s/r + s/ B_s + \eta_t s/R$$

Processing at supercomputer

$$T(\text{Super}) = S/ B_d + \eta_t S/r + S/ B_s + \eta_t S/R + \eta_s S/R$$



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Selecting Analysis Location

Have algebraic equation with eight independent variables.
Faster to move the data if:

$$T(\text{Super}) < T(\text{Archive})$$

$$S/B_d + \eta_t S/r + S/B_s + \eta_t S/R + \eta_s S/R$$

$$< S/B_d + \eta_s S/r + \eta_t s/r + s/B_s + \eta_t s/R$$



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Scaling Parameters

Data size reduction ratio	s/S
Execution slow down ratio	r/R
Problem complexity ratio	η_t / η_s
Communication/Execution	$r/(\eta_t B_s)$

Note (r/ η_t) is the number of bytes/sec that can be processed.

When $r/(\eta_t B_s) = 1$, the data processing rate is the same as the data transmission rate.

Optimal designs have $r/(\eta_t B_s) = 1$



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Bandwidth Optimization

Moving all of the data is faster, $T(\text{Super}) < T(\text{Archive})$,
if the network is sufficiently fast?

$$B_s > (r / \eta_s) (1 - s/S) / [1 - r/R - (\eta_t / \eta_s) (1 + r/R) (1 - s/S)]$$

Note the denominator changes sign when

$$\eta_s < \eta_t (1 + r/R) / [(1 - r/R) (1 - s/S)]$$

Even with an infinitely fast network, it is better to do the
processing at the archive if the complexity is too small.



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Execution Rate Optimization

Moving all of the data is faster, $T(\text{Super}) < T(\text{Archive})$,
if the supercomputer is sufficiently fast?

$$R > r [1 + (\eta_t / \eta_s) (1 - s/S)] / [1 - (\eta_t / \eta_s) (1 - s/S) (1 + r/(\eta_t B_s))]$$

Note the denominator changes sign when

$$\eta_s < \eta_t (1 - s/S) [1 + r/(\eta_t B_s)]$$

Even with an infinitely fast supercomputer, it is better to process at the archive if the complexity is too small.



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Data Reduction Optimization

Processing at the archive is faster, $T(\text{Super}) > T(\text{Archive})$,
if the data reduction is large enough?

$$s < S \left\{ 1 - (\eta_s / \eta_t)(1 - r/R) / [1 + r/R + r/(\eta_t B_s)] \right\}$$

Note criteria changes sign when

$$\eta_s > \eta_t [1 + r/R + r/(\eta_t B_s)] / (1 - r/R)$$

When the complexity is sufficiently large, it is faster to process on the supercomputer even when data can be reduced to one bit.



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Complexity Analysis

Moving all of the data is faster, $T(\text{Super}) < T(\text{Archive})$
if the complexity is sufficiently high!

$$\eta_s > \eta_t (1-s/S) [1 + r/R + r/(\eta_t B_s)] / (1-r/R)$$

Note, as the execution ratio approaches 1,
the required complexity becomes infinite

Also, as the amount of data reduction goes to zero,
the required complexity goes to zero.

**For sufficiently low complexity, it is faster to do the
computation at the storage location**



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



iRODS Micro-services

- Expect to apply micro-services at the remote storage location for tasks such as:
 - Data subsetting
 - Metadata extraction
 - Integrity checks
 - Data retention and disposition
 - Replication

Use Cases

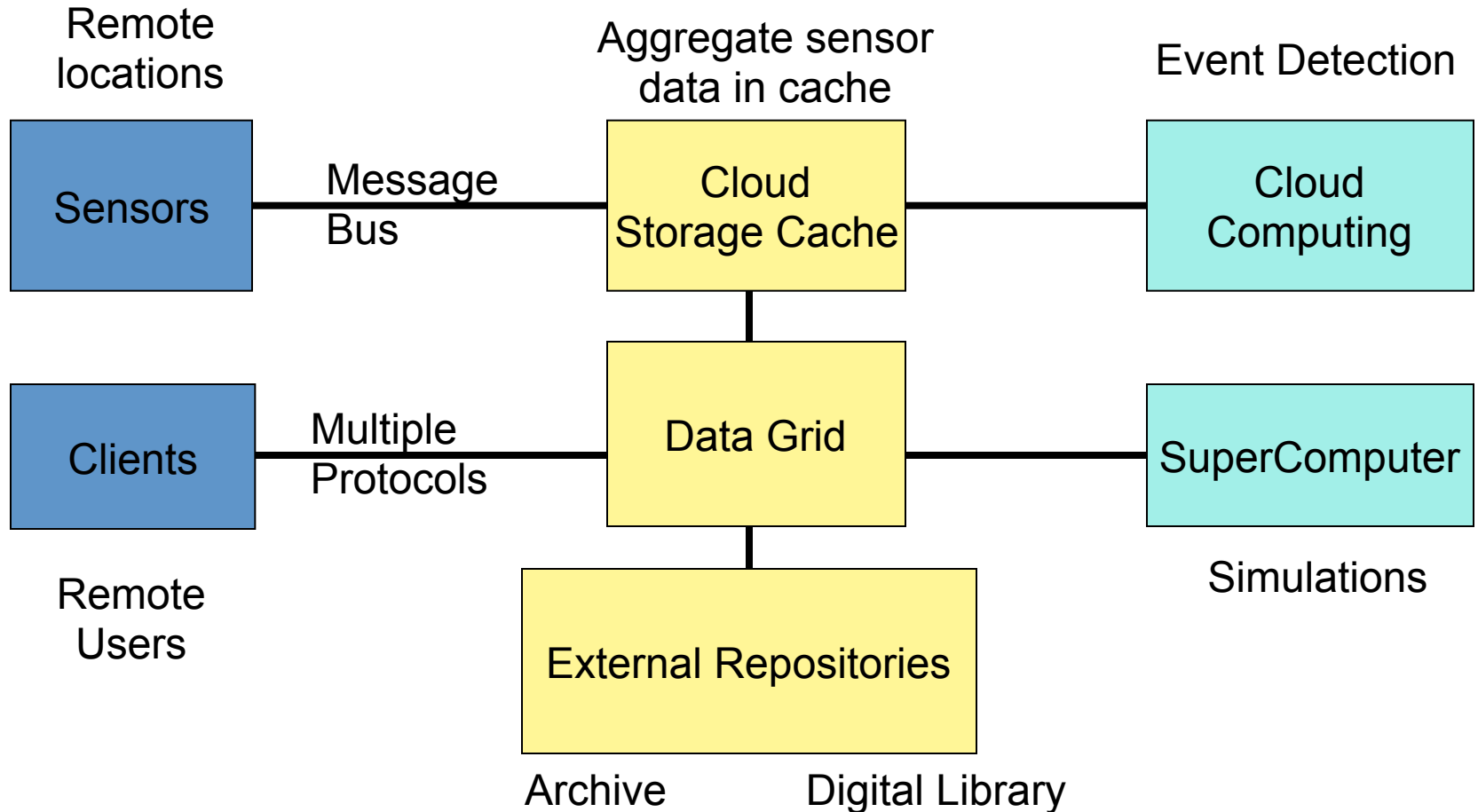
Presentations from the
iRODS User Group
Meeting



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Ocean Observatories Initiative



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



NASA Paper

Federated Observational and Simulation Data in the NASA Center for Climate Simulation Data Management System Project

*John L. Schnase, Glenn Tamkin,
David Fladung, Scott Sinno, and Roger Gill*

[https://www.irods.org/index.php/
iRODS_User_Group_Meeting_2011](https://www.irods.org/index.php/iRODS_User_Group_Meeting_2011)

NASA Center for Climate Simulation

- Observational data
 - MODIS data set, 1 PB, 54 million files, 300 million attributes
 - Created script to control registration of MODIS data into iRODS and an associated database to monitor the submission process
 - Provided access to data products
- Simulation data
 - Added auditing extensions to track status of products
 - New lookup and historical tables
 - Added policy enforcement points

NASA Center for Climate Simulation

- Integrated data grids with Earth System Grid
- Used FUSE file system interface
 - Effective for read operations on remote data grid
- Integrated with NASA Cloud Services
 - Amazon EC2 cloud model

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](https://www.irods.org/index.php/iRODS_User_Group_Meeting_2011)

iRODS in an Neutrino Experiment

- Tokai to Kamioka data grid in Japan
 - Provide access to global collaborators
 - Must aggregate files for storage in HPSS in 1-GB containers
 - File sizes ranged from kiloBytes to MegaBytes
- Created policies to:
 - Automate bundling of files
 - Replicate containers into HPSS
 - Purge cache and backup resources

Rule to Bundle Files

```
acKEKBundle(*collPath, *bundlePath, *cacheRes, *compRes, *archive,
*threshold)||
    msiCheckCollSize(*collPath, *cacheRes, *threshold,
        *aboveThreshold, *status)###
ifExec(*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),
nop###nop###nop###nop###nop, nop, nop) |nop###nop
```

iRODS Rule to Replicate Files

```
acKEKReplicate(*collPath, *cacheRes, *archive, *threshold)||
msiCheckCollSize(*collPath, *cacheRes, *threshold, *aboveThreshold, *status)##
ifExec(*aboveThreshold == 1, nop, nop,
    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)##
        ifExec(*dataRes == *cacheRes,
            msiWriteRodsLog("Replicating file *Coll/*Data", *status)##
            msiDataObjRepl(*Coll/*Data, verifyChksum++++backupRescName=
                *archive, *status)##
            msiWriteRodsLog("Completed replicating file *Coll/*Data",
                *status),
        nop##nop##nop, nop, nop), nop##nop##nop), nop##nop##nop)|nop##nop
```


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)##
  ifExec(*DataResc == *cacheRes,
    msiWriteRodsLog("About to trim file *Coll/*Data", *status)##
    msiDataObjTrim(*Coll/*Data, *cacheRes, *DataRepl, 1,
      IRODS_ADMIN_KW=irodsAdmin, *status)##
    msiWriteRodsLog("Completed trimming replicas of *Coll/*Data",
      *status),
  nop##nop##nop, nop, nop), nop##nop##nop##nop##nop) |nop##nop
```

iRODS at RENCi

*Leesa Brieger, Jason Cposky,
Vijay Dantuluri, Kevin Gamiel, Ray Idaszak,
Oleg Kapeljushnik, Nassib Nassar, Jason
Reilly, Michael Stealey, Lisa Stillwell*

[https://www.irods.org/index.php/
iRODS_User_Group_Meeting_2011](https://www.irods.org/index.php/iRODS_User_Group_Meeting_2011)



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



irods@renci

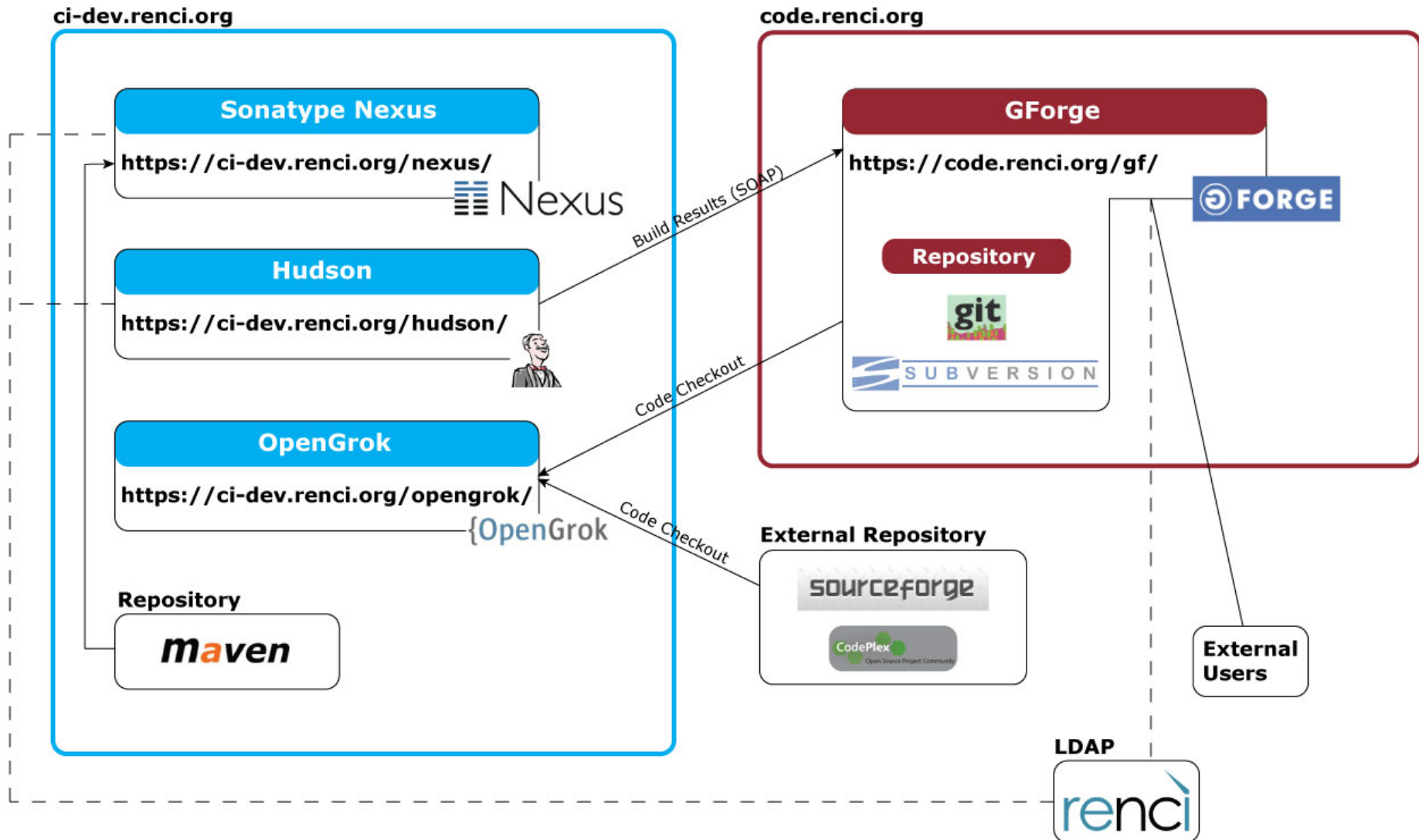
- A new initiative at the Renaissance Computing Institute (RENCI), a research unit of UNC
- An investment by UNC
- A step-up of the collaboration with DICE, already administratively tied to RENCI:
 - DICE-UCSD: Institute of Neural Computing (INC)
 - DICE-UNC: RENCI and the School of Information and Library Science (SILS)
- Stepping toward long-term sustainability

Collaborative Development Environment

- Git – distributed revision control system
- GForge – project and software development management system:
 - hosting & version control
 - bug-tracking
 - messaging
- Hudson – continuous integration environment: incremental quality control
- Nexus – Maven repository that tracks dependencies and bundles for check-out (Java)

Infrastructure Overview

Supports community-based software development



Security

Control mechanisms



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Control Levels

- Policy controls which sites are allowed to connect to the data grid
- Users are authenticated
 - GSI, Kerberos, Challenge-response
- Access control lists on every file
 - User ACLs, User Group ACLs
- All actions are authorized
 - Condition on the execution of every procedure
- Separate rule base for every storage system
 - Data access must comply with local policy

iput ../src/irm.c

checks 10 policy hooks

srbrick14:10900:ApplyRule#116:: acChkHostAccessControl

srbrick14:10900:GotRule#117:: acChkHostAccessControl

srbrick14:10900:ApplyRule#118:: acSetPublicUserPolicy

srbrick14:10900:GotRule#119:: acSetPublicUserPolicy

srbrick14:10900:ApplyRule#120:: acAclPolicy

srbrick14:10900:GotRule#121:: acAclPolicy

srbrick14:10900:ApplyRule#122:: acSetRescSchemeForCreate

srbrick14:10900:GotRule#123:: acSetRescSchemeForCreate

srbrick14:10900:execMicroSrvc#124:: msiSetDefaultResc(demoResc,null)

srbrick14:10900:ApplyRule#125:: acRescQuotaPolicy

srbrick14:10900:GotRule#126:: acRescQuotaPolicy

srbrick14:10900:execMicroSrvc#127:: msiSetRescQuotaPolicy(off)

srbrick14:10900:ApplyRule#128:: acSetVaultPathPolicy

srbrick14:10900:GotRule#129:: acSetVaultPathPolicy

srbrick14:10900:execMicroSrvc#130:: msiSetGraftPathScheme(no,1)

srbrick14:10900:ApplyRule#131:: acPreProcForModifyDataObjMeta

srbrick14:10900:GotRule#132:: acPreProcForModifyDataObjMeta

srbrick14:10900:ApplyRule#133:: acPostProcForModifyDataObjMeta

srbrick14:10900:GotRule#134:: acPostProcForModifyDataObjMeta

srbrick14:10900:ApplyRule#135:: acPostProcForCreate

srbrick14:10900:GotRule#136:: acPostProcForCreate

srbrick14:10900:ApplyRule#137:: acPostProcForPut

srbrick14:10900:GotRule#138:: acPostProcForPut

srbrick14:10900:GotRule#139:: acPostProcForPut

srbrick14:10900:GotRule#140:: acPostProcForPut



Analyses

- NASA
 - security assessment in progress
- University of Wisconsin
 - vulnerability analysis in progress
- UNC
 - vulnerability analysis
- WPAFB
 - Added Kerberos authentication

Support Multiple Transport Protocols

- TCP/IP
 - For small messages < 32 MB, encapsulate data in request to send.
 - For large messages > 32 MB, use multiple I/O streams. Can configure number of streams.
- RBUDP
 - Requires adjusting amount of data sent to be compatible with system and network buffers.
- Redirect streams from source to the client.

RBUDP

- Reliable Blast UDP
 - Implemented by University of Chicago
 - Works well over reliable networks, outperforms parallel TCP/IP streams
- RBUDP incident
 - Network connection failed
 - RBUDP attempted to reconnect in a tight loop
 - Corrected problem by limiting the number of retries to 1000.
 - Used in production by NOAO to move data from Chile to the US
 - Used in production by Cinegrid to move data from Japan to the US, achieved 350 MB/sec with 2 streams

Security Fundamental Assumption

- The purpose for the collection drives the required properties.
- The collection is assembled to meet the driving purpose.
- The collection data are managed under an account defined for the data grid.
- The data grid middleware manages interactions with the collection on behalf of the users
 - Implication: Authenticate every access, authorize every operation

Federation



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Independent Data Grids

- Each zone continues to be a separate iRODS instance, administered separately, but the users in the multiple zones, if given permission, will be able to access data and metaData in the other zones.
- No user passwords are exchanged, as each system will, in a secure manner, check with the user's local zone for authentication when the user connects.

Federation Administration

- The following is some introductory descriptions of how to administer irods federation. Note that if you want to change the name of your zone, you may now do so via an iadmin command (see iadmin -h).
- In this example, we have a local zone, A, and wish to federate with a remote zone B.
- In A: `iadmin mkzone B remote Host:Port`
- where Host:Port is the full host address and Port is the port used by that zone's irods system, for example: `zuri.unc.edu:1247`
- In B: `iadmin mkzone A remote Host:Port`
- At this point, the users in zone B will be able to view some collections in our zone A by doing a 'icd' into the /A tree (in most cases, see immediately below). The user might want to change their home directory.
- If the host defined as the remote zone host is a non-ICAT-Enabled-Server, you do need to define the remote users for even the above authentication to succeed, due to server-to-server interaction. This is needed for most operations anyway.

RENCI Federations

- RENCi federates with 10 data grids
 - CUAHSI data grid
 - LoC VidArch data grid
 - NARA TPAP data grid
 - NCDC data grid
 - NSF Ocean Observatory data grid
 - NSF Temporal Dynamics of Learning Center
 - NSF Teragrid
 - RENCi Visualization data grid
 - Texas Advanced Computing Center -
 - TUCASI data grid (Duke, NCSU)

iRODS Feature Discussion



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Google Hits (top 155) on iRODS

- **6 Blogs:** Day of the iRODS Workshop - GridCast Mar 8, 2010 ... Know what iRODS is? If not, then click here and hopefully lmgty will give you some pointers, or you could use your favourite search engine. ... gridtalk-project.blogspot.com/2010/03/day-of-irods-workshop.html
- **6 DICE Group:** IRODS:Data Grids, Digital Libraries, Persistent Archives, and Real ... The third annual User Group Meeting for iRODS, the Integrated Rule-Oriented Data System, has been announced by the DICE Center at UNC at Chapel Hill www.irods.org/
- **7 Evaluation:** Artefactual IRODS – Artefactual Feb 16, 2009 ... IRODS is a framework for managing large scale data networks. At its core are a set of user developed rules, written in C that handle the way ... www.artefactual.com/wiki/index.php?title=IRODS
- **15 Integration:** ARCS iRODS – PODD iRODS is an open-source distributed filesystem/ data grid project developed in the USA. It provides a middleware layer between the user and the underlying ... projects.arcs.org.au/trac/podd/wiki/iRODS
- **29 News:** Bio-IT World Distributed Bio's Chris Smith on the Rise of iRODS - Bio-IT World Distributed Bio, two projects with iRODS (Integrated Rule-Oriented Data System), including one with the Broad Institute...
www.bio-itworld.com/news/12/23/10/Distribution-Bio-Chris-Smith-IRODS.html

Google Hits on iRODS

- **28 Papers:** ARCS Davis: A generic interface for iRODS and SRB, Shunde Zhang digital.library.adelaide.edu.au
- **4 Book:** Amazon.com Amazon.com: iRODS Primer: integrated Rule-Oriented Data System ... Policy-based data management enables the creation of community-specific collections. Every collection is created for a purpose. The purpose defines the set ... www.amazon.com/iRODS-Primer-integrated-Rule-Oriented-Information/dp/1608453332
- **13 Presentations:** GGF GGF Data Grid Interoperability Demonstration www.ogf.org/OGF23/materials/1289/astro-ogf23-moore.pdf
- **28 Provision:** DICE iRODS | Download iRODS software for free at SourceForge.net Sep 12, 2010 ... Get iRODS at SourceForge.net. Fast, secure and free downloads from the largest Open Source applications and software directory. sourceforge.net/projects/irods/
- **2 Theses:** UCSD Thesis Extensions and an explanation module for the iRODS Rule Oriented Verifier. Jul 12, 2010 ... Data grids provide data sharing environments for the management of globally distributed data. Software systems such as iRODS simulate an ... www.docstoc.com/docs/46795075/Extensions-and-an-explanation-module-for-the-iRODS-Rule-Oriented-Verifier
- **13 Users:** OSG iRODS in the Duke, NERSC, RENCi Collaboration | An Open Science Grid Worklog... Sep 8, 2010 ... iRODS is a mature distributed data management system. The iRODS administrative database is called the iCAT. iRODS servers connected to each ... osglog.wordpress.com/2010/09/08/gsi-authentication-in-irods/

iRODS Wiki

- <http://irods.diceresearch.org>
- “iRODS Primer: integrated Rule-Oriented Data System”
 - Morgan&Claypool Publishers
 - Synthesis Lectures on Information Concepts, Retrieval, and Services



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Feature Requests

Component	Feature Requested	Importance
Message Server	AMQP compliance for message format	100
Clients	LDAP support in iRODS for identity management - PAM/NSS Implies identification of users and groups	12
Clients	Support restart of very large file transfer in iput, iget from last successful buffer	12
Clients	Add Kerberos/AD support in Jargon	11
Security / Authentication	Support ticket-based access to iRODS for limited time and # accesses	11
Network Transfer	Extensions to Jargon for checksum verification on file transfer	10
Rule Engine	Early access to new version of rule engine	10
Clients	Communication between Object Storage and iRODS for policy information exchange	9
Clients	Generic control policies from iRODS to storage system	9
Clients	Mapping from iRODS audit trails to Premis Events	9

Feature Requests

Network Transfer	Support transfer of multiple files using multiple I/O streams.	9
Clients	Resource monitoring system attributes for physical media	8
Clients	Add file soft links	8
Clients	Encrypt all iRODS communication	8
Clients	ACLs on micro-services	7
Clients	Export iRODS through CIFS	7
Table -driven resources	JSON format instead of common-separated-value	7
Clients	Metadata support associated with Fuse	6
Clients	ACLs on metadata	6
Clients	Need information published from iRODS back to Fedora for events – through audit trails	6
Interfaces	Add regular expressions to i-commands (wild cards).	6

Feature Requests

Clients	iExplore streaming interface to files, start display before entire file arrives	5
Clients	Set persistent executable permission on Fuse mounted files	5
File Manipulation	Decompress at client.	5
File Manipulation	Manage locks for collaborative editing (either on storage system, in metadata, or portal).	5
File Manipulation	Support compressed files end-to-end	5
Security / Authentication	Support token-based identification such as SecureID (part of PAM)	5
Table -driven resources	Export KML view of the iCAT catalog	5
Clients	Request for JDK 1.7 compliance - future	4
File Manipulation	Checksum for compressed files	4
Interfaces	Develop Perl API.	4

Feature Requests

Micro-services	Create script for automating module creation. Provide default template for creating new Micro-service.	4
Table -driven resources	KML data format for database results on spatial databases	4
Table -driven resources	table-driven resource access to SQLServer	4
Clients	PyRods support in Gforge (Univ Liverpool)	3
Clients	Synchronize very large files using partial data transfer restarts	3
Clients	Use SQL Server as an iCat catalog	3
Micro-services	Support Perl-based Micro-services by including Perl interpreter in the Micro-service.	3
Table -driven resources	Use SQL Server as an iCat catalog	3
File Manipulation	System level lock manager for core.irb, shared memory, collection (shared cache for rule engine)	2
Security / Authentication	Support a session shell in iRODS, issh	2

Feature Requests

Table -driven resources	URI link to an external database through RESTful interface	2
Administration	Add ability for a project PI to create user accounts for group members. Groupadmin	1
Drivers and Access	Support mounting of a Webdav directory into iRODS. (done through DAVIS)	1
iRODS Metadata Catalog (iCAT)	Create RDA interface to Sybase.	0
iRODS Metadata Catalog (iCAT)	Port iCAT to Sybase.	0
iRODS Metadata Catalog (iCAT)	Support logical registration into iRODS. Ability to associate metadata with a name without requiring a file.	0
Message Server	Integration of messages with workflow service	0
Table -driven resources	Client providing JSON format for iCat query	0
Table -driven resources	Execute control on remote database procedures	0
Information	For SRB to iRODS metadata migration, handle migration of SRB zones. How can multiple SRB zones be re-federated within iRODS easily?	

Feature Requests

Information	Port SRB APIs on top of iRODS, will avoid having to rewrite many application scripts.	
Information	Post link to VBrowser, links to Taverna and EGEE grid.	
iRODS Metadata Catalog (iCAT)	Create signing registration to be able to track origin of files. Given signature, find original copy.	
Security / Authentication	Support client-level encryption and decryption of files. Store encryption keys. Consider a version with encryption done only during transfer. Data is stored unencrypted or encrypted.	
Table -driven resources	Cache for metadata	
Micro-services	Support Python-based Micro-services by including Python interpreter in the Micro-service.	U. Liverpool
Server	A more general mechanism to access external databases. Admin will define location and specify SQL, client will be able to provide arguments. Independent of ICAT.	Table- driven resource
Installation	When compile, verify that only changed files are recompiled.	ok
iRODS Metadata Catalog (iCAT)	Save RDA request results for use in a session, want to pass result list to another Micro-service.	obsolete
Clients	Want support for load leveling	LSU

Feature Requests

Clients	For file movement, automate file caching, staging to final location or replication	KEK
Rule base Management	Provide mechanism to add rule base extensions to a remote rule base.	In Progress
Rule base Management	Provide mechanism to synchronize rule bases across servers within a data grid.	In Progress
Rule base Management	Provide versioning support for rules.	In Progress
Windows	Create a Windows only environment, using Postgres and GSI (both now run on Windows)	In Progress
Windows	Create a Windows only environment, using SqlServer	In Progress
Drivers and Access	Mount a flash drive.	imount
Clients	iFile command - file identification	GaTech
Installation	Create a VM build for use of iRODS in tutorials.	NCSU
Network Transfer	Fix UDP on Solaris.	done?

Summary

iRODS - Policy-based Data Management

- Turn policies into computer actionable rules
 - Constrain application of policies by user group, storage resource, file type, file size, processing flag, system property, time dependence
- Compose rules by chaining standard operations
 - Standard operations (micro-services) executed at the remote storage location
- Manage state information as attributes on namespaces:
 - Files / collections / users / resources / rules
- Validate assessment criteria
 - Queries on state information, parsing of audit trails
- Automate administrative functions
 - Minimize labor costs



Open Source Software

- **Community driven software development**
 - Focus on features required by user communities
 - Focus on bug-free software
 - Focus on highly reliable software
 - Focus on highly extensible software
 - Approximately 3-4 software releases per year
- **Distributed under a BSD license**
 - International collaborations on software development
 - IN2P3 (France), SHAMAN (UK), ARCS (Australia), Academia Sinica (Taiwan)



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Supported Storage Systems

- File systems - Windows, Linux, Mac
- Tape archives - HPSS, Sam-QFS
- Repositories - Flickr, Web sites
- Cloud storage- Amazon S3, EC2
- Relational database - PostgreSQL, Oracle, MySQL
- Table driven resources



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Integrated Rule Oriented Data System

- **iRODS - middleware**
 - Organize distributed data into a sharable collection
 - Manage namespaces for users, files, collections, resources
 - Manage context for each file
 - Enforce management policies at each storage site
 - Highly extensible software, capable of evolving to address new requirements (new policies, new procedures, new state information)
 - Infrastructure independence
 - Open source software
- **Manage petabytes of data and hundreds of millions of files**
 - Shared collections
 - Digital libraries
 - Preservation environments
 - Data analysis pipelines



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

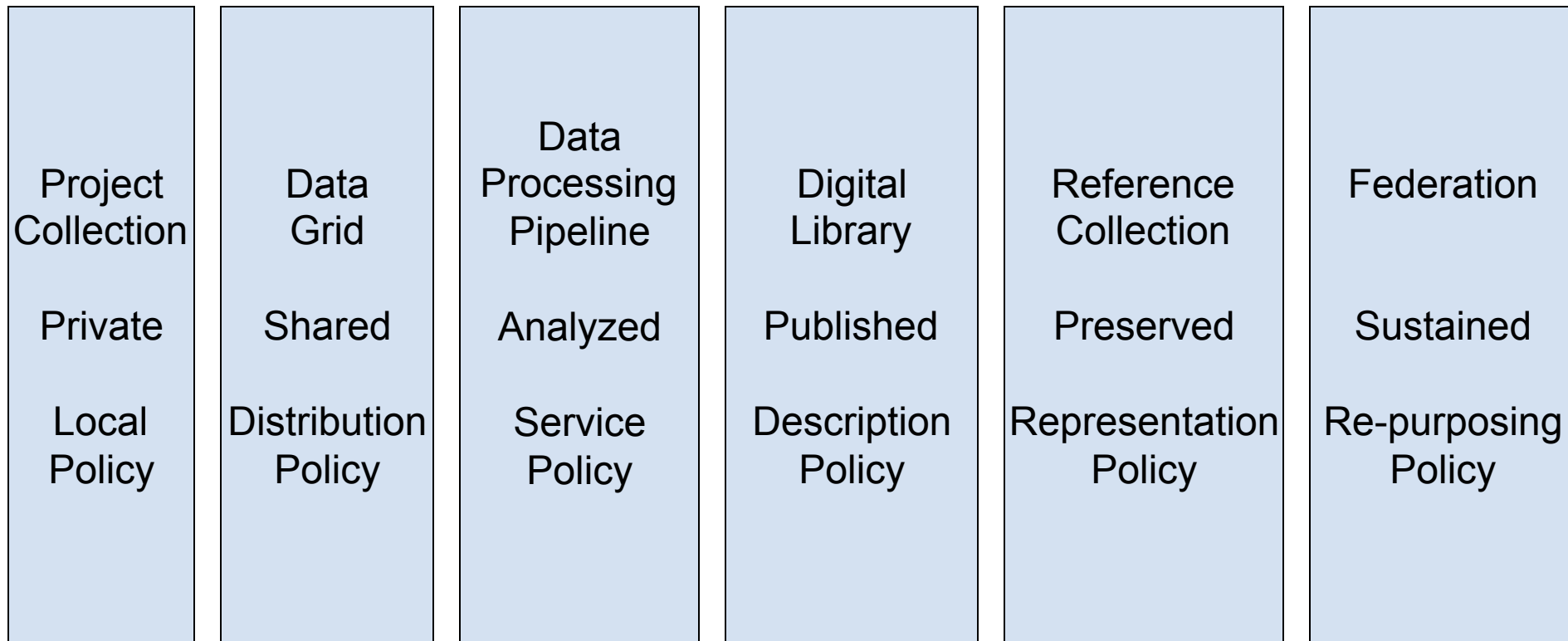


Purpose for Shared Collection

- Enable data processing pipeline that spans multiple institutions
- Enable collaborative research on a shared collection
- Enable formation of a digital library that incorporates material from multiple sources
- Enable creation of a preservation environment that incorporates a deep archive

Data Life Cycle

The driving purpose changes at each stage of the data life cycle



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



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Data Virtualization

Preferred Access Client (C library, Unix, Web Browser)

Data Collection

Storage Repository

- Storage location
- User name
- File name
- File context
- Access controls

Data Grid

- Logical resource name space
- Logical user name space
- Logical file name space
- Logical context (metadata)
- Policies

Data is organized as a shared collection



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL



Topics on μ Services: What do we learn here?

Design of μ Services for achieving a goal

- Extraction & Ingestion of template-identified metadata

Implementation of μ Services

- How each module is coded to be μ -compliant

Testing of μ Services

- From the command line – no less.
- A demo of all the services as a workflow



Design of a μ Service



Design: Prologue

Problem Statement: Extract metadata from an Email and associate with the Email file.

Generic Solution Steps:

How to

Extract Metadata Attr-Value pairs from one file

Based on a Template defined in a second file, and

Associate the metadata to a third file?

Problem Break up:

Extract
Metadata



Ingest
Metadata



Commit
Metadata

Sounds like 3 μ Services!

Design: Lets look at the input files

Sample Input File (in our case also Metadata File):

Date: Thu, 01 Feb 2007, 22:33:35 +000

From: adil hasan <a.hasan@rl.ac.uk>

To: Michael Wan <mwan@sdsc.edu>

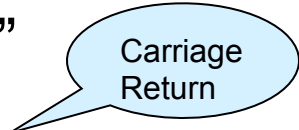
Template Files contain tags that are used to identify keyword/value pairs in a document

Sample Tags:

<PRETAG>Date: </PRETAG>SentDate<POSTTAG> </POSTTAG>

<PRETAG>From: </PRETAG>Sender<POSTTAG> </POSTTAG>

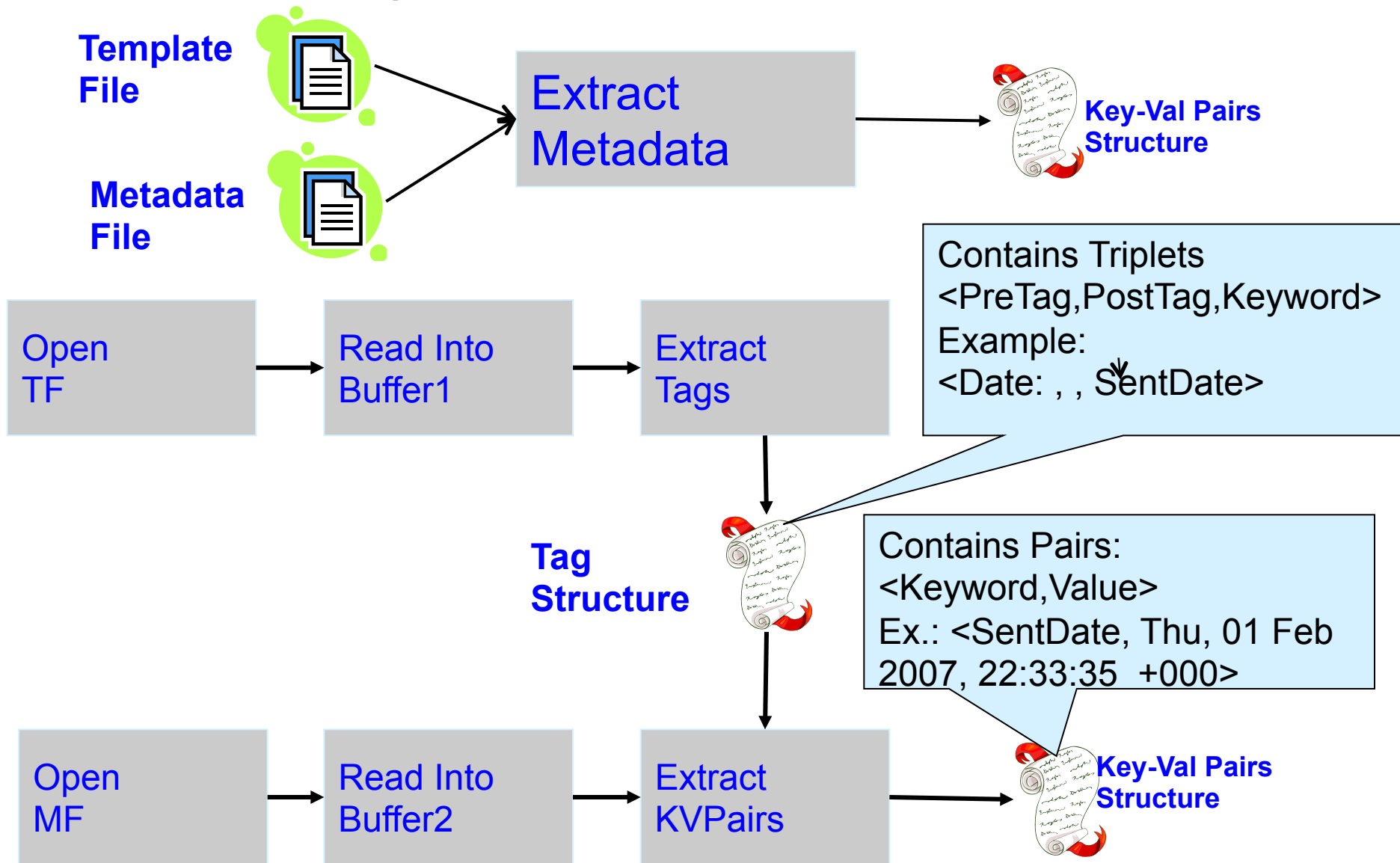
Meaning: Whatever is found between “Date :” and “ ” provides the “value” for the keyword: “SentDate”



Carriage Return

Metadata Files provide the actual metadata that need to be ingested.

Design: Extract Metadata



Design: Ingest Metadata

Key-Val Pairs
Structure



Target
FileName



Ingest
Metadata



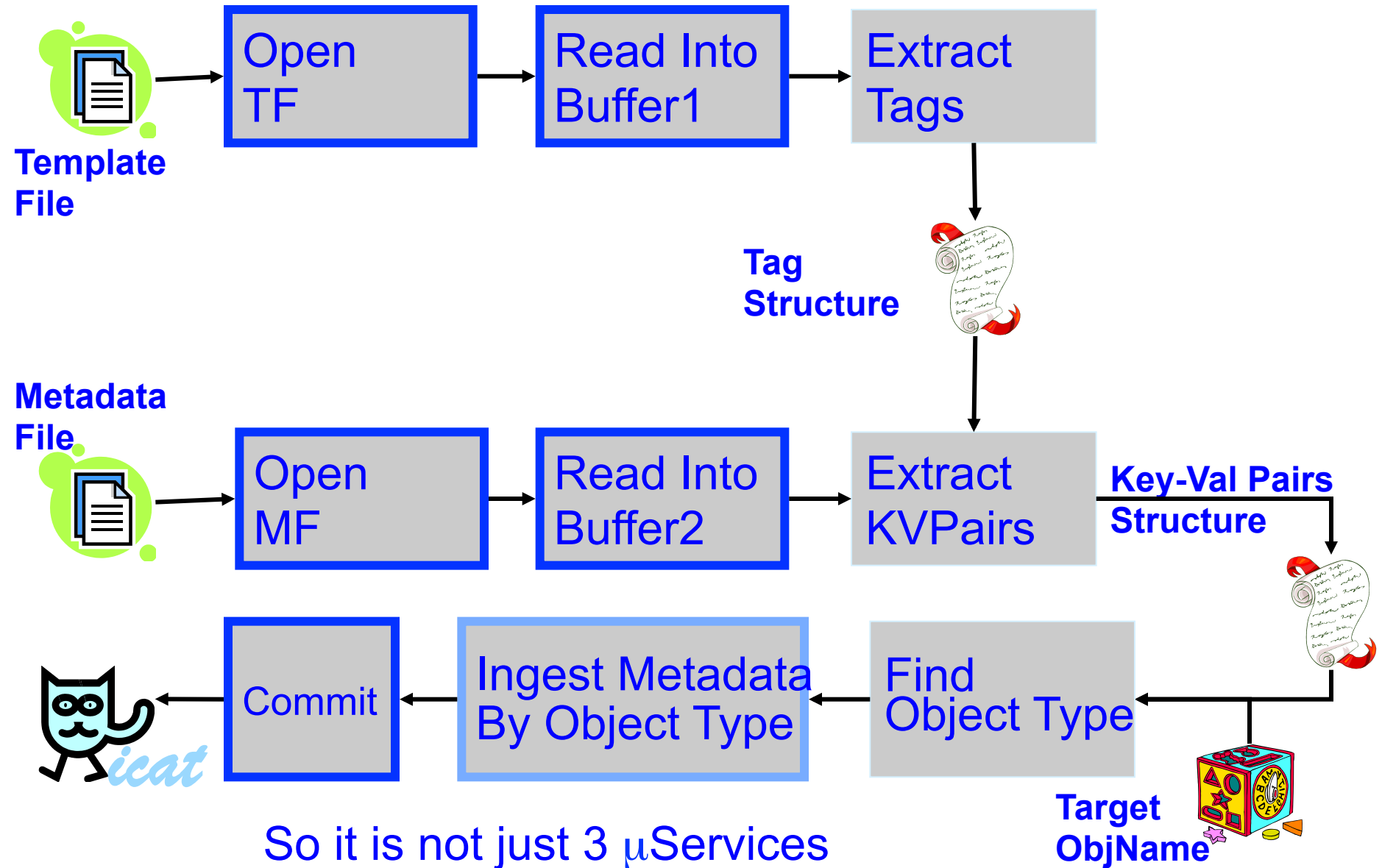
Instead of writing a μ Service for file-metadata ingestion only, design a μ Service that can be applied to any iRODS object (data, collection, resource, user or **token or metadata**)

Find
Object Type

Ingest Metadata
By Object Type

Question: How to convert a C-function into a μ Service?

Design: Epilogue



Implementation of μ Services

Implementation: Prologue

Four Easy Steps:

Define the signature of the μ Service

Register the μ Service as an invokable method by the rule engine

Create the μ Service

- This may need other function calls (new and old)

Describe the μ Service

We will look at two Examples:

- “FindObjectType” μ Service: [msiGetObjType](#)
- “Extract Tag” μ Service: [msiReadMDTemplateIntoTagStruct](#)

Implementation: Signature Definitions

All μ Services have only two types of parameters

Params 1...(n-1) are of the type `msParam_t`

Param n is of the type `ruleExecInfo_t`

`msParam_t` is defined as:

```
typedef struct MsParam {  
    char *label;  
    char *type;  
    void *inOutStruct;  
    bytesBuf_t *inpOutBuf; } msParam_t;
```

`ruleExecInfo_t` is the “white board” used for passing session-oriented parameters that can be used by the Rule Engine and the micro-services.

```
int msiGetObjType (msiParam_t *objParam, msiParam_t *typeParam,  
                  ruleExecInfo_t *rei);  
int msiReadMDTemplateIntoTagStruct (msiParam_t *bufParam,  
                                    msiParam_t *tagParam, ruleExecInfo_t *rei);
```

Implementation: Registration: “FindObjectType”

The Rule Engine only executes μ Services that are enumerated in the list structure:

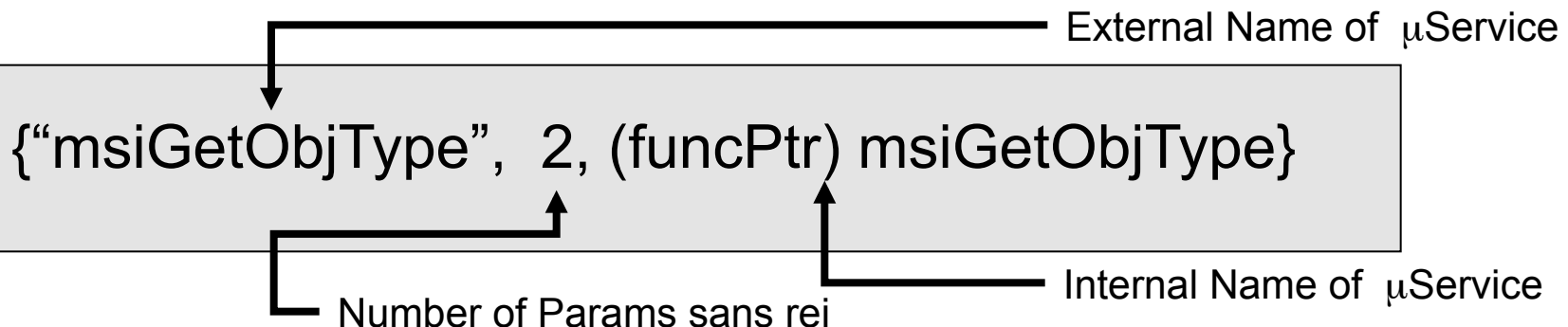
```
microsdef_t  MicrosTable[ ] = { } ;
```

(can be found in the file reAction.h or reAction.table)

For the μ Service

```
int msiGetObjType (msiParam_t *objParam, msiParam_t  
*typeParam, ruleExecInfo_t *rei);
```

We add the following to the ActionTable



Implementation: Registration “ExtractTag”

For the μ Service

```
int msiReadMDTemplateIntoTagStruct (msiParam_t  
    *bufParam, msiParam_t *tagParam, ruleExecInfo_t *rei);
```

We add the following to the ActionTable

```
{“msiReadMDTemplateIntoTagStruct”,  
    2, (funcPtr) msiReadMDTemplateIntoTagStruct}
```

Implementation: Creation “FindObjectType”

Here we program the code for the μ Service

```
int msiGetObjType (msiParam_t *objParam,  
                  msiParam_t *typeParam, ruleExecInfo_t *rei)  
{  
    char*  objName;  
    char   objType[MAX_NAME_LEN];  
    int    i;  
  
    RE_TEST_MACRO(“Looping back on msiGetObjType”);  
  
    if (strcmp(objParam->type, STR_MS_T) != 0)  
        return(USER_PARAM_TYPE_ERROR);  
  
    objName = (char *) objParam->inpOutStruct;  
  
    i = getObjType (rei->rsComm, objName, objType);  
    if (i < 0) return(i);  
  
    fillStrInMsParam(typeParam, objType);  
    return(0);  
}
```

Local
Variables

Needed for Loop Back
Testing of Workflow
and Rules

Internal Function
that is used for
finding types of
Objects. This
routine makes
calls to iCAT to
find the type of
the Object

Type
Checking

Returning value
being malloc'd into
Param Structure and
type-cast properly

Implementation: Describe “FindObjectType”

We want to provide enough material for users to call the μ Service and for a program to identify it automatically in the future.

```
/**
 * \fn  GetObjType
 * \author  Arcot Rajasekar
 * \date  2007-02-01
 * \brief  this function finds from the iCat the type of a given object
 * \param[in]  objParam  is a msParam of type STR_MS_T
 * \param[out] typeParam is a msParam of type STR_MS_T
 * \return integer
 * \retval 0 on success
 * \retval USER_PARAM_TYP_ERROR when input param does not match
 *         the type
 * \retval from getObjType
 * \sa  getObjType
 * \post
 * \pre
 * \bug  no known bugs
 **/
```

Implementation: Creation “Extract Tag”

Here we program the code for the μ Service

```
int msiReadMDTemplateIntoTagStruct (msiParam_t *bufParam,
                                   msiParam_t *tagParam, ruleExecInfo_t *rei)
{
    bytesBuf_t *tmpObjBuf;
    tagStruct_t *tagValues;
    /*other internal variables are defined here */
    RE_TEST_MACRO(“Looping back on msiReadMDTemplateIntoTagStruct”);
    if (strcmp(bufParam->type, BUF_LEN_MS_T) != 0 || bufParam->inpOutBuf == NULL)
        return(USER_PARAM_TYPE_ERROR);
    tmpObjBuf = (bytesBuf_t *) bufParam->inpOutBuf;
    tagValues = (tagStruct_t *) mallocAndZero(sizeof(tagStruct_t));
    /* the main code segment that reads the buffer and identifies the
       the <preTag, KeyWord, postTag> triples goes in here. The triplets
       are store in tagValues. */
    if (tagValues->len == 0) { free(tagValues ); return(NO_VALUES_FOUND); }
    tagParam->inOutStruct = (void *) tagValues;
    tagParam->type = (char *) strdup(TagStruct_MS_T);
    return(0);
}
```

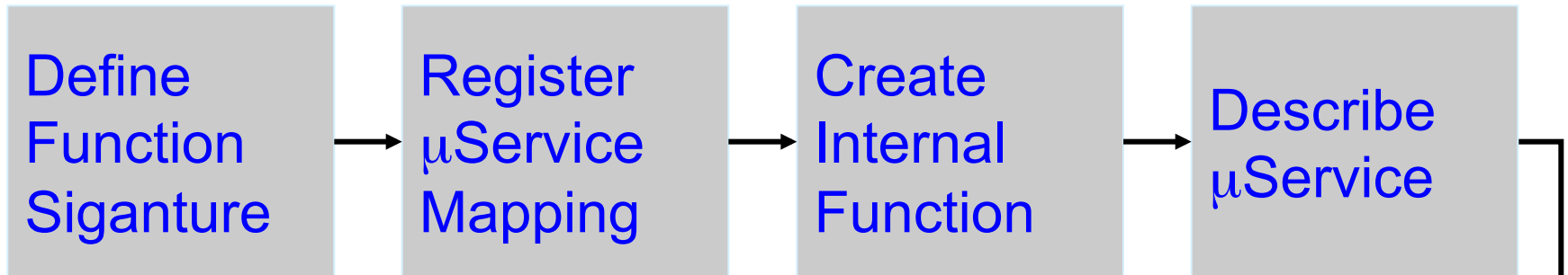
Implementation: Describe “Extract Tag”

We want to provide enough material for users to call the μ Service and for a program to identify it automatically in the future.

```
/**
 * \fn msiReadMDTemplateIntoTagStruct
 * \author Arcot Rajasekar
 * \date 2007-02-01
 * \brief this function parses a buffer containing a template-style file
 * and stores the tags in a tag structure.
 * \note the template buffer should contain triplets be of the form
 * <PRETAG>re1</PRETAG>kw<POSTTAG>re2</POSTTAG>
 * re1 identifies the pre-string and re2 identifies the post-string, and any value
 * between re1 and re2 in a metadata buffer can be associated with keyword kw.
 * \param[in] bufParam is a msParam of type BUF_MS_T
 * \param[out] tagParam is a msParam of type TagStruct_MS_T
 * \return integer
 * \retval 0 on success
 * \retval USER_PARAM_TYP_ERROR when input param don't match the type
 * \retval INVALID_REGEXP if the tags are not correct
 * \retval NO_VALUES_FOUND if there are no tags identified
 * \retval from addTagStruct
 * \sa addTagStruct
 * \post
 * \pre
 * \bug no known bugs
 **/
```

Implementation: Epilogue

RECAP:



Any Function can be easily converted into a μ Service ----- μ Compliant. Except that

Important!!
Implement
recovery μ Service

Testing of μ Services

Testing: Prologue

Client-side: the *irule* command

- Create a workflow of μ Services

 - Test with the “loop” functionality

 - Test with “verbose” functionality

 - Test without these side-effects

Server-side:

- Create a rule out of the workflow, or

- Add the μ Service to an existing rule

Client-side:

- Test the rule using the *irule* command

Semantics Testing is under research

Testing: Micros

msiDataObjOpen

opens a iRODS File

openObj

msiDataObjRead

reads an open iRODS File

readObj

msiReadMDTemplateIntoTagStruct

reads Tag Info into Struct

getTagsForKV

msiExtractTemplateMDFromBuf

gets MD using Tag Struct

getKVPairsUsingTags

msiGetObjType

finds type of object

findObjType

msiAssociateKeyValuePairsToObj

ingests extracted metadata

ingestBulkMD

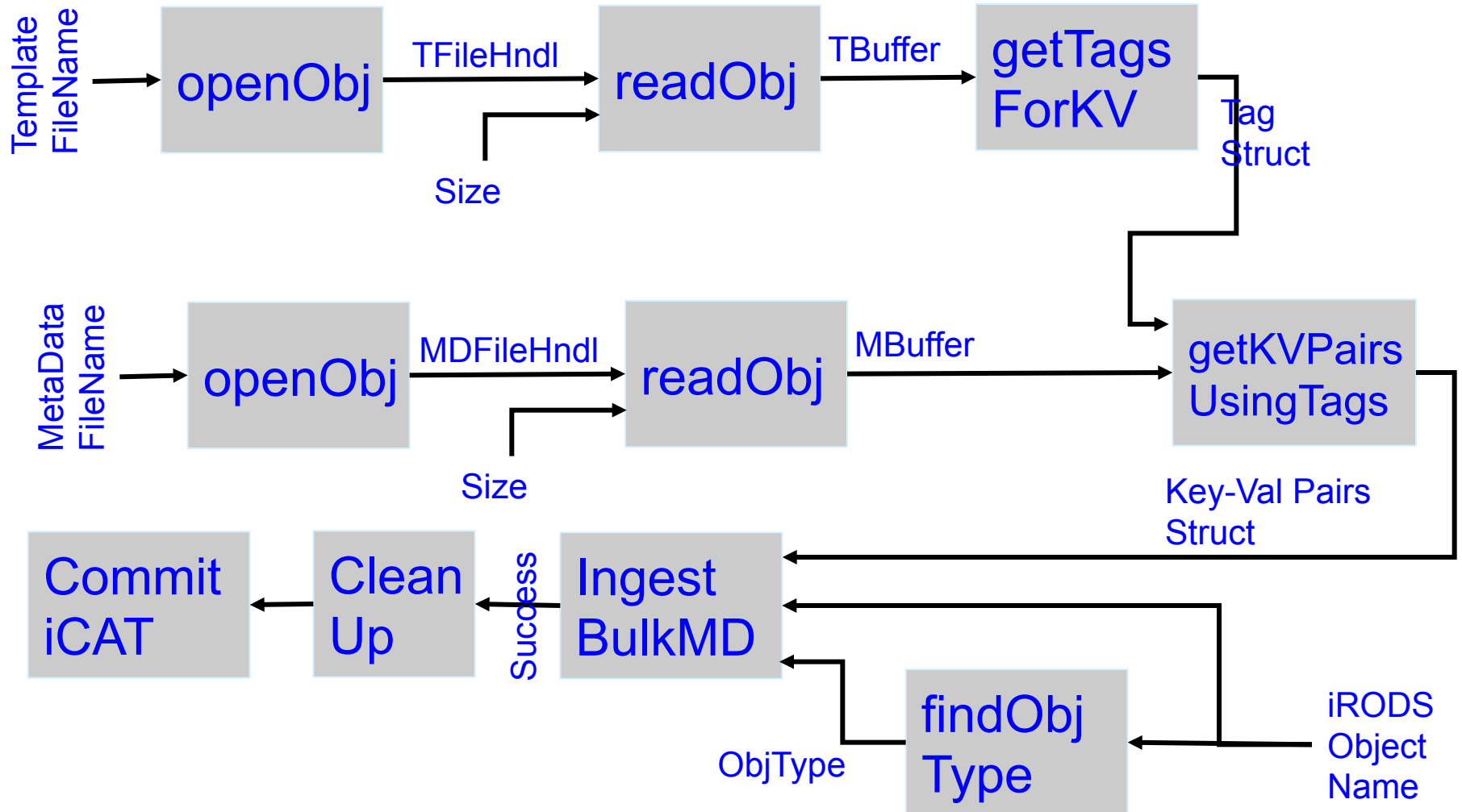
msiCommit

commit transaction in iCAT

commitIcat

External Aliases Help Application Developers and Users

Testing: Workflow Diagram



Blue values are inputs

Testing: CommandLine WorkFlow

Pretty Printed Listing of File “ruleInp5”

```
mDExtract || openObj( *A, *T_FD)##getSizeData(*A,*S)##  
             readObj( *T_FD, *S, *R1_BUF)##  
             getTagsForKV( *R1_BUF, *TSP)##  
             openObj( *B, *M_FD)##  
             readObj( *M_FD, 10000, *R2_BUF)##  
             getKVPairsUsingTags( *R2_BUF, *TSP, *KVP)##  
             findObjType( *C, *OTYP)##  
             ingestBulkMD( *KVP, *C, *OTYP)##  
             closeObj(*T_FD,*J1)##closeObj(*M_FD,*J2)##  
             commitIcat
```

WorkFlow
Rule

*A=/tempZone/home/rods/Templates/mdTemplate1.txt%

*B=/tempZone/home/rods/test1.email%

*C=/tempZone/home/rods/test2.email

*R1_BUF%*TSP%*R2_BUF%*KVP%*A%*B%*C%*OTYP

Inputs

PrintOuts

How to run it: `irule -v -F ruleInp5`

Testing: Making a Rule

The rule is very similar to the workflow we had seen in the previous slide.

No Conditions are here

```
mDExtract(*A,*B,*C) || openObj( *A, *T_FD)##  
    readObj( *T_FD, 10000, *R1_BUF)##  
    getTagsForKV( *R1_BUF, *TSP)##  
    openObj( *B, *M_FD)##  
    readObj( *M_FD, 10000, *R2_BUF)##  
    getKVPairsUsingTags( *R2_BUF, *TSP, *KVP)##  
    findObjType( *C, *OTYP)##  
    ingestBulkMD( *KVP, *C, *OTYP)##  
    closeObj(*T_FD,*J1)##closeObj(*M_FD,*J2)##  
    commitIcat
```

Recovery
Section

```
|closeObj(*T_FD)##nop##  
    recover_getTagsForKV( *R1_BUF, *TSP)##  
    closeObj(*M_FD)## nop##  
    recover_getKVPairsUsingTags( *R2_BUF, *TSP, *KVP)##  
    nop##  
    recover_ingestBulkMD( *KVP, *C, *OTYP)##  
    nop##nop##rollbackIcat
```

Delaying a μ Service

One can delay the execution of any μ Service either in the irule execution or in a rule at the server side.

Example:

The μ Service `msiSysReplDataObj(*R)` replicates an existing iRODS file.

In order to delay this, one can use:

```
delayExec(<PLUSET>015m</PLUSET>, msiSysReplDataObj( tgReplResc ),nop)
```

In a rule this might be used as follows:

```
acPostProcessForPut | $objPath like /tmpZone/home/tg/* |  
    delayExec((<PLUSET>015m</PLUSET>, msiSysReplDataObj( tgReplResc ), nop)  
    | nop  
acPostProcessForPut | $objPath like /tmpZone/home/nvo/* |  
    msiSysReplDataObj( nvoReplResc ) | nop  
acPostProcessForPut | | nop | nop
```

Recap: How to build μ Services

Create the μ Service

- Create the micro-service function as needed.

```
int myPetProc(char *in1, int in2,  
              char *out1, int *out2)  
{  
    ... my favorite code ...  
}
```

Create the μ Service Interface (msi)

- Create the micro-service interface.

```
int  msiMyPetProc(msParam_t *mPin1, msParam_t *mPin2,
                 msParam_t *mPout1, msParam_t *mPout2,
                 ruleExecInfo_t *rei)
{
    char *in1, *out1;
    int  i, in2, out2;
    RE_TEST_MACRO (" Calling myPetProc")
    /* the above line is needed for loop back testing using irule -i option */
    if (in1 = parseMspForStr (mPin1) == NULL) return(USER_PARAM_TYPE_ERR);
    if (in2 = parseMspForPosInt(mPin2) < 0) return (in2);
    i = myPetProc(in1, in2, out1, &out2);
    fillIntInMsParam (mPout2, out2);
    fillStrInMsParam(mPout1, out1);
    return(i);
}
```

Register the μ Service Interface (msi)

- Add the signature

```
int  msiMyPetProc(msParam_t *mPin1, msParam_t *mPin2,  
                 msParam_t *mPout1, msParam_t *mPout2,  
                 ruleExecInfo_t *rei)
```

In the file: reAction.header

- Make the micro-service interface visible to the rule engine by adding:

```
{"msiMyPetProc",4, (funcPtr) msiMyPetProc},
```

In the File: reAction.table

NOTE: Adding a μ Service to a module is quite different

Demonstration & Conclusion

Design of μ Services for achieving a goal

- Extraction & Ingestion of template-identified metadata

Implementation of μ Services

- How each module is coded to be μ -compliant

Testing of μ Services

- From the command line – no less.
- A demo of all the services as a workflow



*Hope you
Enjoyed it !
Any Questions !!*

iRODS is a "coordinated NSF/OCI-Nat'l Archives research activity" under the auspices of the President's NITRD Program and is identified as among the priorities underlying the President's 2009 Budget Supplement in the area of Human and Computer Interaction Information Management technology research.

Reagan W. Moore

rwmooore@renci.org

<http://irods.diceresearch.org>

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



THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

