OSG PUBLIC STORAGE

Open Science Grid

Tanya Levshina Fermilab

Motivations for OSG Public Storage

- Enable VO whose computation requires “large” data to use OSG sites more easily
  - LHC VOs have solved this problem (FTS, Phedex, LFC)
  - Smaller VOs are still struggling with large data in a distributed environment
- Ease the task of VO data management:
  - Providing quota management
  - Moving data and software to the sites
  - Retrieving the output data from the sites
  - Providing metadata catalog

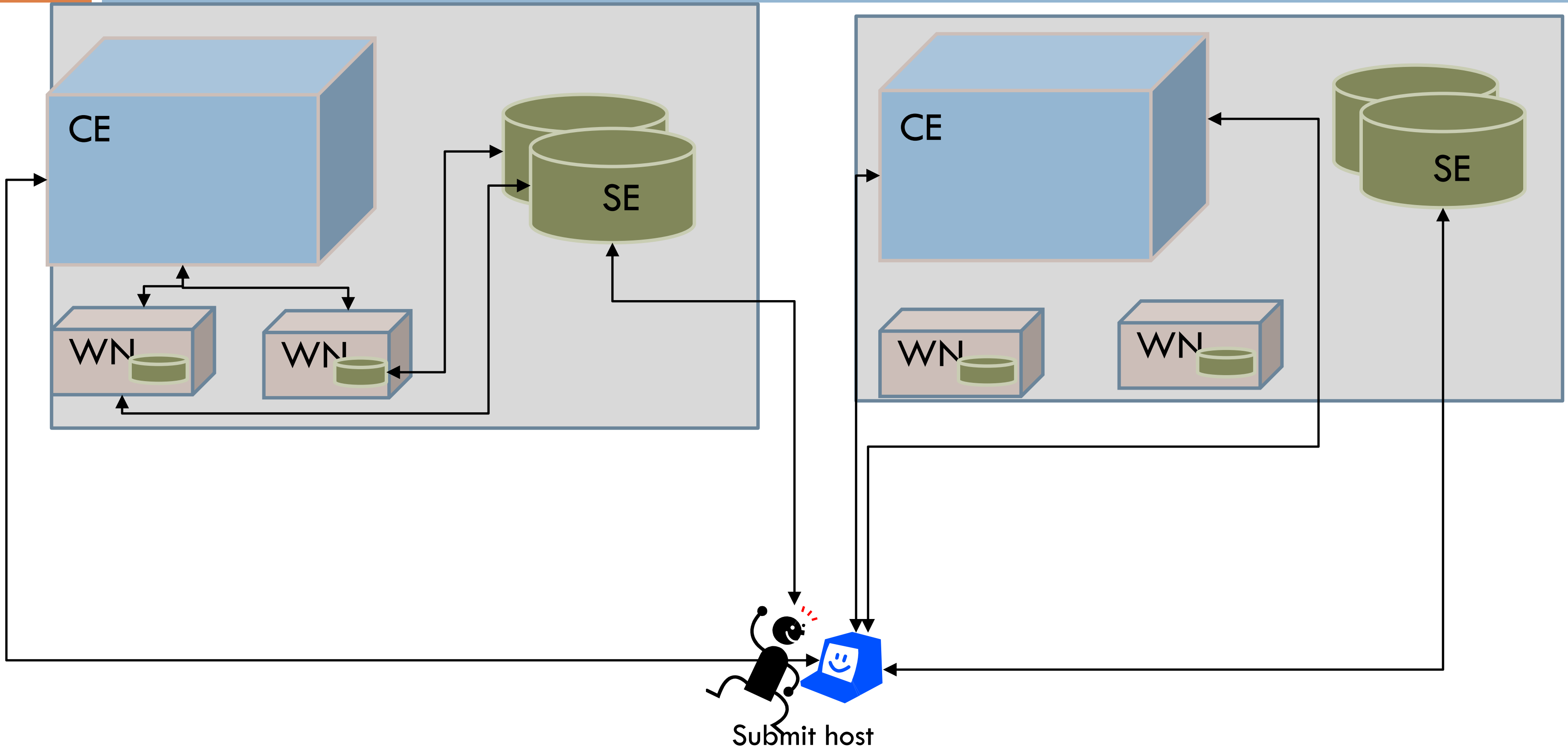
Challenges and Requirements

- Challenges:
  - Most of the OSG sites do not support dynamic storage allocation and do not have tools for automatic management of allocated storage
  - the VOs that rely on opportunistic storage have difficulties finding an appropriate storage, verifying its availability and monitoring its utilization
  - the involvement of a Production Manager, Site Admins and VO support personnel is required to allocate or rescind storage space.
- Requirements:
  - Allow the OSG Production manager to manage public storage allocation across all the participating sites.
  - Impose minimal burden on the participating sites.
  - Allow a VO Manager to manage data within VO quota
  - Simplify storage selection for data storage.

OSG Storage Types

- Per-Site persistent shared storage:
  - “Classic” Storage Element(SE)
    - Most of the OSG sites have a least one classic SE per gatekeeper
    - Software could be installed into a shares area ($OSG_APP) on a head node via GridFTP server
    - Data is pre-staged into a shared area ($OSG_DATA) on a head node via GridFTP server
    - Read (sometimes write for $OSG_DATA) access from the worker nodes (NFS)
    - Size limitation per non-owner VO (< 400GB)
  - SRM Storage Element has the following components:
    - Storage Resource Manager(SRM) endpoint
    - Distributed File System
    - GridFTP server(s) for transfer
    - Available space per not-owner VO is negotiable (in TBs)
    - Can be accessed from a worker node via SRM or fuse
- Local Storage:
  - Worker nodes have local disk available for each job($OSG_WN_TMP).
  - Nominally at least 10GB, but in practice can be less or more.
  - There is no (standard) way to prestage to these areas, and that data generally disappears when job ends.

Grid Job Access to Site Storage



iRODS

- The Integrated Rule-Oriented Data System (iRODS) is developed by the Data Intensive Cyber Environments research group and collaborators.
- iRODS implements a policy-based data management framework.
  - handles various objects (resources, collections and files)
  - each object has a set of properties (metadata) associated with it
  - properties are enforced by polices (set of Rules)
  - rules trigger a chain of actions (micro-services). A chain of actions may include recovery from failures and notification.
  - Provides means to set quota limit and enforce quota management
- iRODS performs transfers by
  - using implementation specific protocol to access POSIX compliant resources
  - using an external driver to Mass Storage. The driver should implement "put" and "get" methods to transfer entire files. File transfer is performed in two steps (disk cache is needed)
- The Metadata Catalog (iCAT) stores complete state information about the system in a database. iCAT contains information about resources, resource usage, quotas and users. It also serves as metadata catalog for users data collections.
- Widely used by scientific community (Biology, Environment , Physical Sciences, Geosciences, etc)

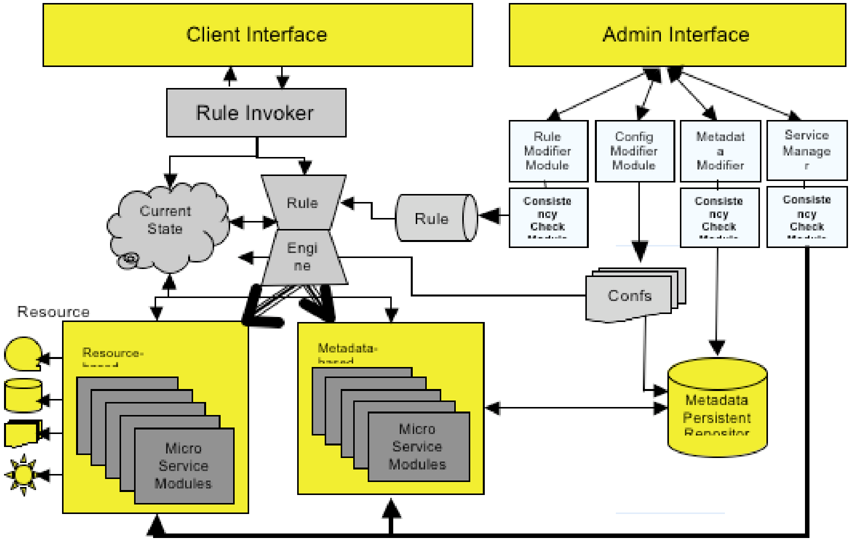
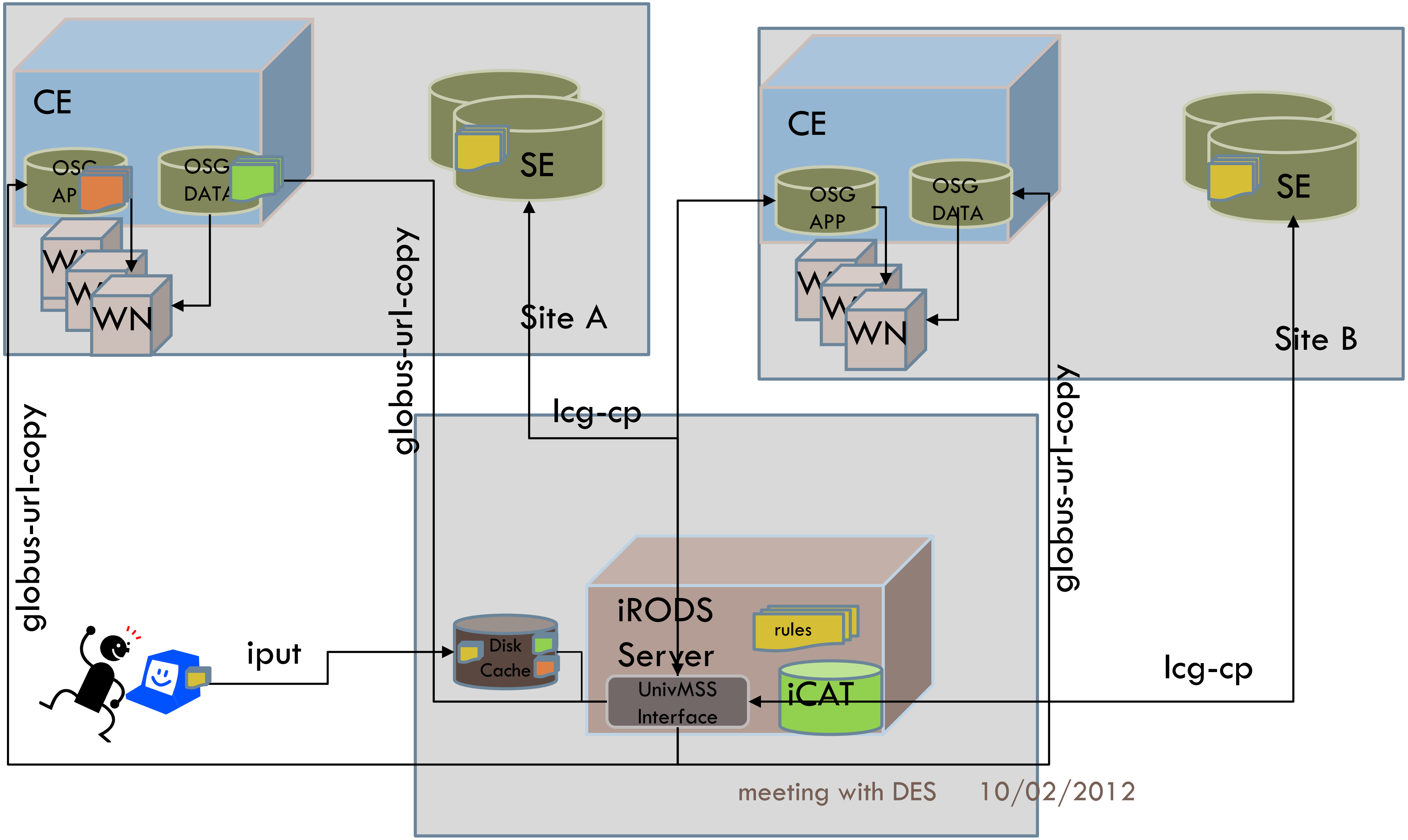


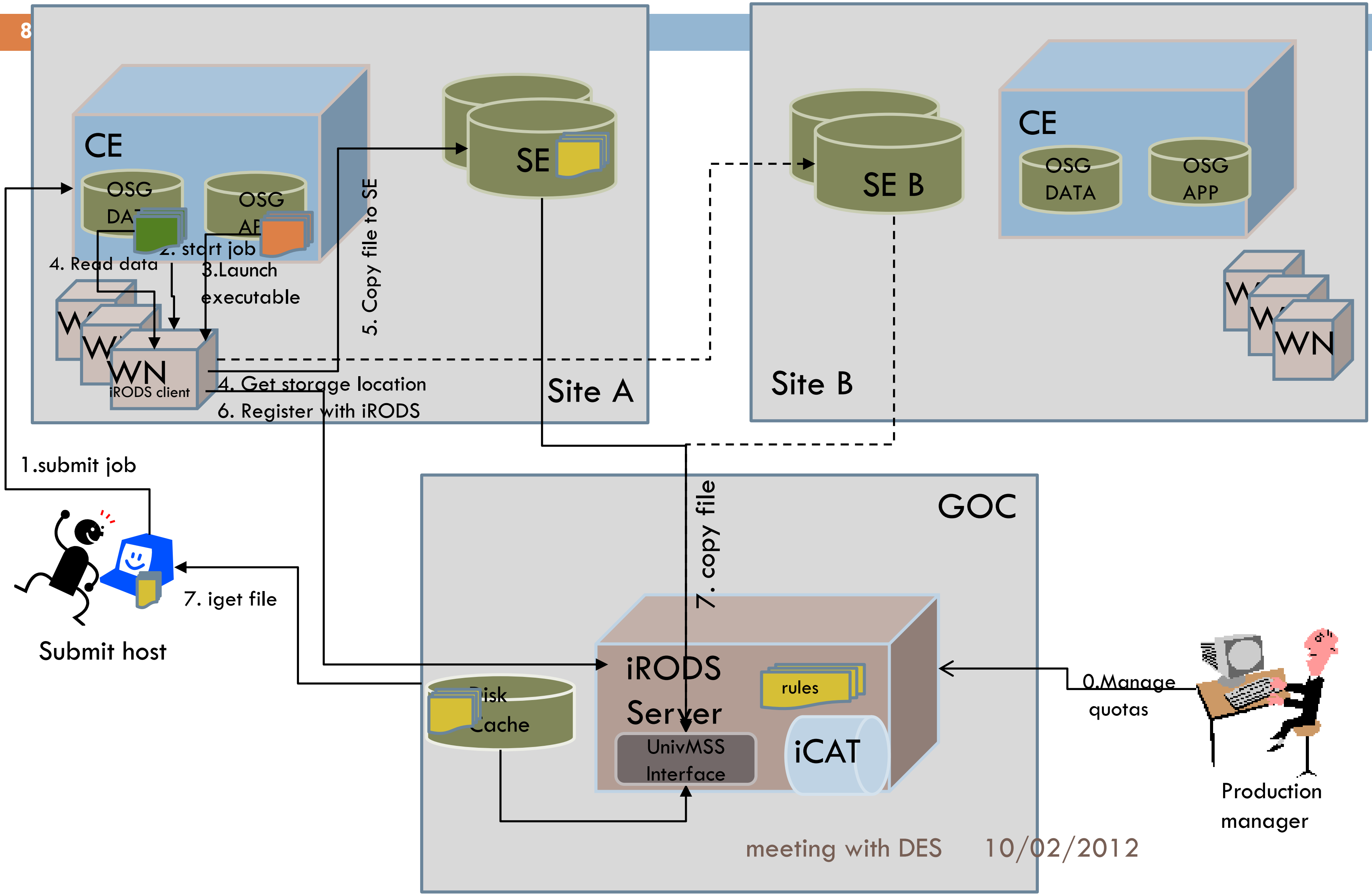
Figure 5. iRODS Architecture Components

https://www.irods.org/index.php/File:Irods-intro5.png

OSG/iRODS integration (pre-staging software and data)



OSG/iRODS integration (running grid job)



User Level Data Management (I)

- pre-stage file to a specific SE:

  ```
  iput –R Nebraska my_file
  ```

- pre-stage file to some SE:

  ```
  iput –R osgSrmGroup my_file
  ```

- download file from SE:

  ```
  iget my_file
  ```

- delete file from SE:

  ```
  irm –f my_file
  ```

- replicate file from one SE to all other available SEs:

  ```
  irepl-osg –R osgSrmGroup my_file
  ```

- list file detailed information :

  ```
  ils –l my_file
  ```

User Level Data Management (II)

- Login on submission node
- Add to condor job description file:

+UsesiRODS=True

- Add to your script:

```
$IRODS_PLUGIN_DIR/icp idrodse://irodsuser@irods.fnal.gov:1247?/osg/home/username/<input_file> <input_file>
$IRODS_PLUGIN_DIR/icp <output_file> idrodse://irodsuser@irods.fnal.gov:1247?/osg/home/username/<output_file>
```

- Submit job
- The job starts on a worker node on a site where a glidein pilot is running
- iRODS software is installed by pilot plugin script if UseiRODS is set to true
- Your job will
  - Check via iRODS the location of the input file (if you want to get file from SE)
  - Download file using srm client from the SE or cp command
  - Check via iRODS where to upload output file (finds the ‘best resource’ : closest first then space available)
  - Upload file to SE using srm client command or cp command
  - Register file with iRODS

iRODS integration pros and cons

- Advantages:
  - A user can pre-stage data to OSG_DATA, OSG_APP and SE SRMs via iRODS without dealing with sites, gathering scattered information about site resources, worrying about surl and end path, etc
  - Global namespace that have information about files location, size, etc
  - Quota management
- Disadvantages:
  - File pre-staging is happening in two hops. Performance test has shown that irods client – irods server transfer time is negligible comparing time consumed by srm copy command. icp-osg command can be used to copy file directly to storage and register file in iRODS.
  - One can not utilize iRODS features fully because of the architecture we are using:
    - We need to write and maintain custom scripts
    - Can not achieve same performance

Current Status

- Deployed on a VM at Fermilab
- Have demonstrated the feasibility of managing public storage at the OSG sites with iRODS.
  - A Production Manager can manage resource allocations at remote sites between various VOs.
  - No actions are required from the sites after initial allocation of resources.
  - A user can upload and download files from a user laptop or a worker node using iRODS commands and in-house developed scripts.
- 3 representatives from a user community have expressed their interest to try out the current installation.
  - EIC – pre-staging data to OSG_DATA on all sites
  - Pheno – pre-staging software to OSG_APP and upload files to SEs from worker nodes.
  - SAGA – pre-staging data to OSG_DATA on all sites

References and Contacts

- iRODS Home Page
  https://www.irods.org/index.php/IRODS:Data_Grids,_Digital_Libraries,_Persistent_Archives,_and_Real-time_Data_Systems
- OSG iRODS Docs and Tutorial:

  https://twiki.grid.iu.edu/bin/view/VirtualOrganizations/IRODSOSG
- iRODS-Chat google group:

  https://groups.google.com/forum/?fromgroups#!forum/iROD-Chat
- OSG Storage docs:

  https://www.opensciencegrid.org/bin/view/Documentation/StorageOverview

  https://www.opensciencegrid.org/bin/view/Documentation/StorageEndUser
- Contacts:
  - Reagan Moore rwmoore@renci.org – DICE director
  - developers:
    Wayne Schroeder <schroeder@diceresearch.org>
    Arcot Rajasekar rajaseka@email.unc.edu
  - Team at RENCI:
    "Leesa M. Brieger" <leesa@renci.org> - Sr Research Software Developer at RENCI
    Charles Schmitt <cschmitt@renci.org> - lead of iRODS and data science efforts
    - team
      Jason Coposky <jasonc@renci.org>
      Terrell Russell <tgr@renci.org>