## Data Management[1]

### Grid data management

- Different sources of data
  - Sensors
  - Analytic equipment
  - Measurement tools and devices
- Need to discover patterns in data to create information
- Need mechanisms to deal with large quantities of data

### Managing large amounts of data

- Want to move data around
  - Store it long term in appropriate places (tape silos)
  - Move input to where the job is running
  - Move output from the run location to where it is needed (workstation, long term storage)
- We need to deal with terabytes or petabytes of data and large number of files
- Standard tool to move data in grids is GridFTP
- Various other pieces of software to deal with storing data within a site
  - Designed as LRM for data, such as `dCache`
  - Keeping track of where data is placed, such as Globus Replica Location Service
- High performance tools needed to solve several data problems
  - Huge raw volume of data
    - Storing it
    - Moving it
    - Measured in terabytes, petabytes, ...
  - Huge number of filenames
    - Expectation of $10^{12}$ filenames
    - Such a collection difficult to handle efficiently
  - How to find the data
- Data questions on the grid
  - Where are the files I want?
  - How to move data/files to where I want?

### GridFTP

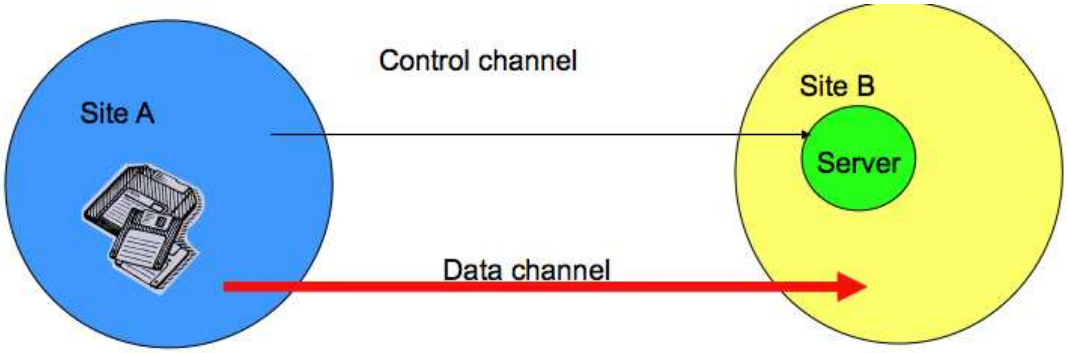
- High performance, secure, and reliable data transfer protocol based on standard FTP

[1] Most of the material in this set of notes is from the Educational division of Open Science Grid.

- Extensions include
  - Strong authentication, encryption via Globus GSI
    - GSI and Kerberos support
  - Multiple data channels for parallel transfers
    - Multiple TCP streams between two network endpoints
    - Striped data transfer
      - One or more TCP streams between $m$ network endpoints on the sending side and $n$ network points on the receiving side
      - Includes cases where $m \neq n$
  - Third-party control of data transfers
  - Tunable network and I/O parameters
    - Partial file transfers
    - Manual/automatic control of TCP buffer/window sizes
  - Authenticated reusable channels
    - Support for reliable and restartable data transfer
  - Server side processing, command pipelining
- Provides reliability and fault tolerance for file transfers using the RFT protocol
- Basic definitions

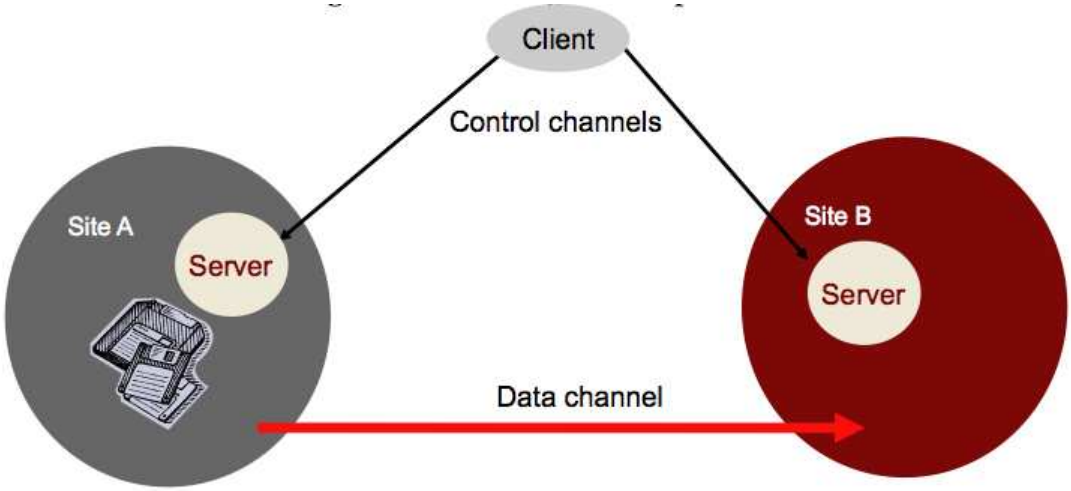
  **Control channel**
  - TCP link for the flow of commands and responses
  - Low bandwidth; encrypted and integrity protected by default

  **Data channel**
  - Communication link[s] for the flow of actual data of interest
  - High bandwidth; authenticated by default; encryption and integrity protection optional
- File transfer with GridFTP
  - Control channel can go either way
    - Depends on which end is client and which end is server
  - Data channel in the same direction

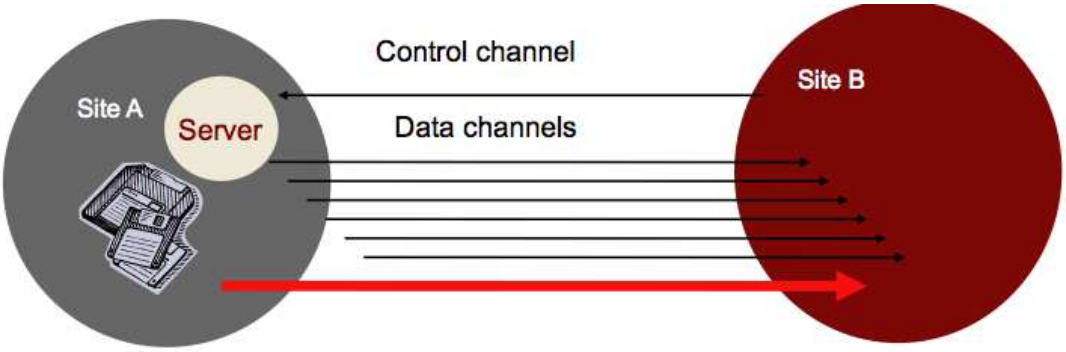



- Third party transfer
  - Data lives on a server while computation is to be performed on a different node
    - Option to transfer data from data server to client, and then, from client to compute node

  - File transfer without data flowing through the client
  - Controller can be separate from source or destination
    * Client opens two control channels – one to each of the servers
    * Use the control channels to get the two servers to establish a data channels between them and move the data across
  - Useful for moving data from storage to compute node
  - Useful when there is a very good connection between the two servers but not such a good connection from servers to client




- Parallel streams
  - Use several data channels
  - Establish several data channels and send the file in pieces across all the connections




  - Underlying network protocol, TCP used for most data transfers on the Internet, tries to share bandwidth equally between each stream
    * More streams get us more shares (cheating?)
  - If one or more channels do not run at full speed, this slowdown is ameliorated
    * A single lost packet can cause massive slow down
- Making GridFTP go really fast
  - Use fast disks/filesystems
    * Filesystem should read/write $>$ 30MB/s
  - Configure TCP for performance

    - Use the TCP tuning guide at: `http://www-didc.lbl.gov/TCP-tuning/`
    - Something of a black art
  - Patch your Linux kernel with web100 patch
    - Important work-around for Linux TCP feature
    - Give more control and information about network connections on your servers
    - `http://www.web100.org`
  - Understand your network path (between two servers)

- GridFTP usage
  - `globus-url-copy`
  - Convention on URL formats
    - `file:///home/sanjiv/dataex/largefile`
      - A file named `largefile` on the local filesystem, in directory `/home/sanjiv/dataex/`
    - `gsiftp://osg-edu.cs.wisc.edu/scratch/sanjiv/`
      - A directory accessible via `gsiftp` on the host `osg-edu.cs.wisc.edu` in directory `/scratch/sanjiv/`
  - Examples

```
$ globus-url-copy
  file:///home/sanjiv/dataex/myfile
  gsiftp://osg-edu.cs.wisc.edu/nsf/osgedu/sanjiv/ex1
$ globus-url-copy
  gsiftp://osg-edu.cs.wisc.edu/nsf/osgedu/sanjiv/ex2
  gsiftp://tp-osg.ci.uchicago.edu/sanjiv/ex3
```

- Working with GridFTP
  - Create a directory with your user name on the machine `osg-edu.cs.wisc.edu` using the GRAM command `globus-job-run`
  - Create some files of different sizes, to use for exercise

```
$ dd if=/dev/zero of=smlfile-$USER bs=1M count=10
$ dd if=/dev/zero of=medfile-$USER bs=1M count=50
$ dd if=/dev/zero of=lrgfile-$USER bs=1M count=200
```

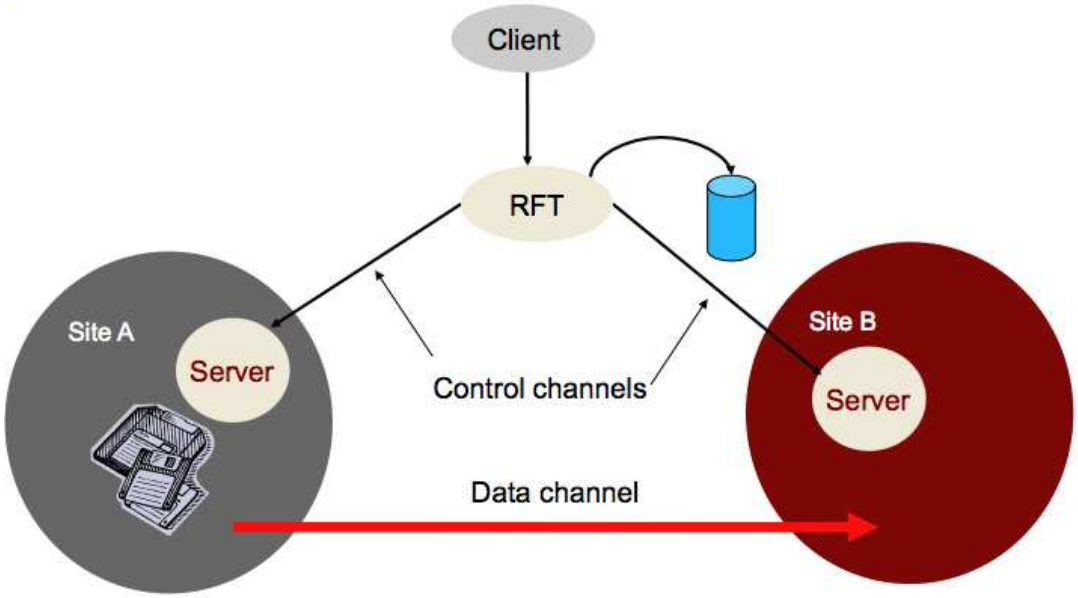
  - Use `globus-url-copy` to copy small file to remote machine

```
$ globus-url-copy file:///home/$USER/data-mgmt/smlfile-$USER \
    gsiftp://osg-edu.cs.wisc.edu/nfs/osgedu/$USER/ex1
$ echo $?
```

    - If the output status is 0, the command succeeded
  - Measuring transfer speed
    - Use the flag `-vb` when copying a large file
  - Parallel streams
    -
    - Use the flag `-p n` to open `n` parallel streams
  - Third party transfers
    - Done through two `gsiftp` URLs
    - `globus-url-copy` will control the transfer but data will *not* pass through local machine

**RFT – Reliable file transfer**

- Protocol to provide reliability and fault tolerance for file transfers
- Part of the Globus Toolkit
- Acts as a client to GridFTP, providing management of a large number of transfer jobs
  - Same as Condor to GRAM
- Capabilities
  - Keep track of state of each job
  - Run several transfers at once
  - Deal with connection failure, network failure, or failure of any of the servers involved
- RFT vs Condor
  - In job submission, we have a facility GRAM to submit jobs, and Condor to provide reliability on top of that
  - In data transfer context, we have RFT as a part of Globus Toolkit




  - RFT acts as a client to GridFTP to add the following
    - Provide management of a large number of transfer jobs
    - Keep track of the state of each job
    - Run several transfers at once
    - Deal with connection failure, network failure, or failure of any of the servers involved
- Reliability on top of high performance provided by GridFTP
  - Fire and forget
  - Integrated automatic failure recovery
    - Network level failures
    - System level failures
- Example
  - Create a `.xfr` transfer job file to list some RFT parameters and all of the files to transfer

    - On `osg-grid1`, look at the file `/client/globus/share/globus_wsrf_rft_client/transfer.xfr`
    - Parameters to be specified in transfer file include
      - Use of different URLs
      - Transfer between two remote sites
      - Use of parallel streams
      - Increased transfer concurrency
  - Use `rft` command with a `.xfr` file
    - `rft -h terminable.ci.uchicago.edu -f ./rft.xfr`
- Working with RFT
  - Create a transfer job file to list some RFT parameters and all of files to transfer
  - Transfer the large file that was created to Wisconsin machine
  - Experiment with the following
    - Add more URLs to transfer
    - Transfer between two remote sites
    - Use parallel streams
    - Increase the transfer concurrency

### RLS – Replica location service

- Component of the data grid architecture (Globus component)
- Provides access to mapping information from logical names to physical names of items
  - Logical filename (LFN)
    - Names a file containing interesting data
    - Does not refer to a location (which host, or where in a host)
  - Physical filename (PFN)
    - Refers to a file on some filesystem somewhere
    - Often uses `gsiftp://` URL for specification
- Aims to reduce access latency, improve data locality, improve robustness, scalability, and performance for distributed applications
- Produces Local Replica Catalogs (LRCs) to represent mappings between logical and physical files scattered across the storage system
  - LRC can be indexed for better performance
- Two RLS catalogs
  1. Local Replica Catalog (LRC)
     - Stores mappings from LFNs to PFNs
     - Local catalog of maps from LFNs to PFNs
       - `H-R-792845521-16.gfw` → `gsiftp://dataserver.phys.uwm.edu/LIGO/H-R-792845521-16.gfw`
     - Informs RLIs about known mappings
     - Interaction

       **Q:** Where can I get filename `experiment_result_1`?

**A:** You can get it from `gsiftp://gridlab1.ci.uchicago.edu/home/benc/r.txt`

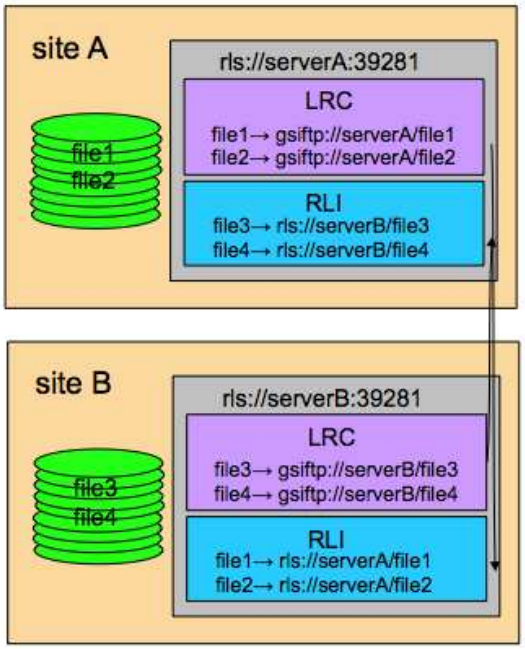
– Undesirable to have one of these for whole grid
  * Lots of data
  * Single point of failure

2. Replica Location Index (RLI)
   – Stores mappings from LFNs to LRCs
   – Local catalog of maps from LFNs to LRCs
     * `H-R-792845521-16.gfw` → LRCs at MIT, PSU, Caltech, and UW-M
   – Interaction

     **Q:** Who can tell me about filename `experiment_result_1`?

     **A:** You can get more information from the LRC at `gridlab1`
   – Failure of one RLI or LRC does not break everything
   – RLI stores reduced set of information; so can cope with many more mappings

- Globus RLS




  – Can query for files in a two-step process: find files on the grid by
    * Querying RLIs to get LRCs
    * Then, query LRCs to get URLs

- Server perspective
  – Mappings LFNs → PFNs kept in database
    * Uses generic ODBC interface to talk to any [good] relational database
    * MySQL, PostgreSQL, Oracle, DB2, . . .
    * All relational database details hidden from administrator and user
      · Database may need to be tuned for performance
  – Mappings LFNs → LRCs stored in one of two ways
    1. Table in database

        * Full, complete listing from LRCs that update your RLI
        * Requires each LRC to send your RLI full, complete list
            · As number of LFNs in catalog grows, this becomes substantial
            · $10^8$ filenames at 64 bytes per filename $\Rightarrow \approx 6$ GB
    2. In memory in a special hash called Bloom filter
        * $10^8$ filenames stored in as little as 256 MB
            · Easy for LRC to create Bloom filter and send over network to RLIs
        * May cause RLI to lie when asked if it knows about an LFN
            · Only false positives
            · Tunable error rate
            · Acceptable in many contexts
        * Not possible to use wild cards with Bloom filters

- RLS command line tools
    - `globus-rls-admin`
        * Administrative tasks
        * Ping server
        * Connect RLIs and LRCs together
    - `globus-rls-cli`
        * End-user tasks
        * Query LRC and RLI
        * Add mappings to LRC

- Client perspective
    - Two ways for clients to interact with RLS server
        1. `globus-rls-cli` simple command line tool
            * Query
            * Create new mappings
        2. Create your own client by creating against API in Java, C, or Python

- `globus-rls-cli`
    - Simple query to LRC to find a PFN for LFN

```
$ globus-rls-cli query lrc lfn some-file.jpg rls://dataserver:39281
some-file.jpg : file://localhost/netdata/s001/S1/R/H/714023808-714029599/
    some-file.jpg
some-file.jpg : file://medusa-slave001.medusa.phys.uwm.edu/data/S1/R/H/
    714023808-714029599/some-file.jpg
some-file.jpg : gsiftp://dataserver.phys.uwm.edu:15000/data/gsiftp_root/
    cluster_storage/data/s001/S1/R/H/714023808-714029599/some-file.jpg
```

    - The command returns 3 entries for the LFN `some-file.jpg`
    - `rls://dataserver:39281` designates the RLS server location with port number
    - Server and client sane if LFN not found

```
$ globus-rls-cli query lrc lfn foo rls://dataserver
LFN doesn't exist: foo
$ echo $?
1
```

    - Wildcard searches of LRC

```
$ globus-rls-cli query wildcard lrc lfn "H-R-7140242*-16.gwf" \
    rls://dataserver:39281
H-R-714024208-16.gwf: gsiftp://dataserver.phys.uwm.edu:15000/data/gsiftp_root/
    cluster_storage/data/s001/S1/R/H/714023808-714029599/H-R-714024208-16.gwf
H-R-714024224-16.gwf: gsiftp://dataserver.phys.uwm.edu:15000/data/gsiftp_root/
    cluster_storage/data/s001/S1/R/H/714023808-714029599/H-R-714024224-16.gwf
```

– Bulk queries – obtain PFNs for more than one LFN at a time

```
$ globus-rls-cli bulk query lrc lfn H-R-714024224-16.gwf H-R-71402408-16.gwf \
    rls://dataserver:39281
H-R-714024208-16.gwf: gsiftp://dataserver.phys.uwm.edu:15000/data/gsiftp_root/
    cluster_storage/data/s001/S1/R/H/714023808-714029599/H-R-714024208-16.gwf
H-R-714024224-16.gwf: gsiftp://dataserver.phys.uwm.edu:15000/data/gsiftp_root/
    cluster_storage/data/s001/S1/R/H/714023808-714029599/H-R-714024224-16.gwf
```

– Two step process to query RLI to locate LFN→LRC mapping and then, uerying that LRC for the PFN

```
$ globus-rls-cli query rli lfn example-file.gwf rls://dataserver
example-file.gwf: rls://ldas-cit.ligo.caltech.edu:39281
$ globus-rls-cli query lrc lfn example-file.gwf \
    rls://ldas-cit.ligo.caltech.edu:39281
example-file.gwf:gsiftp://ldas-cit.ligo.caltech.edu:15000/archive/S1/L0/LH0/
    H-R-7140/H-R-714024224-16.gwf
```

– Bulk queries to RLI

```
$ globus-rls-cli bulk query rli lfn H-R-714024224-16.gwf H-R-714024320-16.gwf \
    rls://dataserver
H-R-714024320-16.gwf: rls://ldas-cit.ligo.caltech.edu:39281
H-R-714024224-16.gwf: rls://ldas-cit.ligo.caltech.edu:39281
```

– Wildcard queries to RLI may not be supported

```
$ globus-rls-cli query wildcard lrc lfn "H-R-7140242*-16.gwf" rls://dataserver
Operation is unsupported: Wildcard searches with Bloom filters
```

– Create new LFN→PFN mappings

  * Use `create` to create first mapping for an LFN
    `globus-rls-cli create file1 gsiftp://dataserver/file1 rls://dataserver`
  * Use `add` to add more mappings for an LFN
    `globus-rls-cli add file1 file://dataserver/file1 rls://dataserver`
  * Use `delete` to remove a mapping for an LFN
    `globus-rls-cli delete file1 file://dataserver/file1 rls://dataserver`
    - When last mapping is deleted for an LFN, the LFN is also deleted
    - Cannot have LFN in LRC without a mapping

– LRC can also store attributes about LFNs and PFNs

  * Size of LFN in bytes
  * md5 checksum for an LFN
  * Ranking for a PFN or URL
  * Extensible ... user defined attributes to create and add
  * Can search catalog on the attributes
  * Attributes limited to strings, integers, double, and date/time

– First, create the attributes and then add values for LFNs

```
$ globus-rls-cli attribute define md5checksum lfn string rls://dataserver
$ globus-rls-cli attribute add file1 md5checksum lfn string \
42947c86b8a08f067b178d56a77b2650 rls://dataserver
```

  - Query on the attribute

```
$ globus-rls-cli attribute query file1 md5checksum lfn rls://dataserver
md5checksum: string: 42947c86b8a08f067b178d56a77b2650
```

- Bloom filters
  - LRC to RLI can happen in two ways
    1. LRC sends list of all its LFNs (but not PFNs) to the RLI
       * RLI stores whole list
       * Answers accurately (I know/I don't know)
       * Expensive to move and store large lists
    2. Bloom filters
       * LRC generates a Bloom filter of all of its LFNs
       * Bitmap that is much smaller than the whole list of LFNs
       * Answers less accurately (May be I know/I don't know)
       * Might end up querying LRCs unnecessarily but we won't ever get wrong answers)
       * Can't do a wildcard search
- Working with RLS
  - Check that we can connect to an RLS server

```
$ globus-rls-admin -p rls://communicado.ci.uchicago.edu
```

  - Querying an RLS server

```
$ globus-rls-cli rls://communicado.ci.uchicago.edu
rls> query lrc lfn example
```

**Related work**

- Storage resource manager (SRM)
  - Equivalent of a job scheduler for storage
  - Allocates space, makes sure it does not get swapped out before you are done (pinning), handles staging to/from tape
  - `http://sdm.lbl.gov/indexproj.php?ProjectID=SRM`
- dCache
  - Provides a system to store/retrieve a huge amount of data
  - Data may be distributed among a large number of heterogeneous server nodes
  - The nodes are under a single virtual filesystem tree with a variety of standard access methods
  - `http://www.dcache.org`
- BeSTMan
- Globus metadata catalog
  - Standalone metadata catalog service with an OSGA service interface
  - Associates application-specific descriptions with data files, tables, or objects

  - Descriptions encoded in structured ways defined by schemaor community standards
  - Makes it easier for users and applications to locate data relevant to specific problems
  - *Get temperature, barometric pressure, and CO2 concentrations for a specific geographic area*
  - item `http://www.globus.org/grid_software/data/mcs.php`
- Stork
  - Cross between RFT and Condor DAGman
  - Make data placement activities "first class citizens" in the Grid just like computational jobs
  - Provides facilities to queue, schedule, monitor, manage, and check-point data placement activities
  - Makes sure that data placement activities complete successfully and without any human interaction
  - `http://www.cs.wisc.edu/condor/stork`
- Storage resource broker
  - Supports shared collections that can be distributed across multiple organizations and heterogeneous storage systems
  - Used as a data grid management system to provide a hierarchical logical namespace to manage the organization of data (files)
  - `http://www.sdsc.edu/srb/index.php/Main_Page`

#### OSG and data management

- OSG relies on GridFTP protocol for the raw transport of data
  - Uses Globus GridFTP in all cases
  - Exception where interfaces to storage management systems (rather than file systems) dictate individual implementations
- OSG supports the SRM interface to storage resources
  - Enables management of space and data transfers
  - Prevents unexpected errors due to running out of space
  - Prevents overload of GridFTP services
  - Provides capabilities for pre-staging, pinning, and retention of data files
- OSG provides reference implementation of two storage systems: BeSTMan and dCache