Storage Resource Managers: Interface and SRM services

Riccardo Zappi, INFN-CNAF

1st INFN Grid User School

26-29 November 2007

Outline

- Storage resource and Data resource in Grid
- SRM paradigm and SRM interface
- A brief history of SRM services in WLCG
- Storage Model: foundation for SRMv2.2
- SRM v2.2 interface description
- Some SRM v2.2 functions in detail
- Conclusion

Motivation (1/2)

- Suppose you want to run a job on your local machine
  - Need to allocate space
  - Need to bring all input files
  - Need to ensure correctness of files transferred
  - Need to monitor and recover from errors
  - What if files dont fit space? Need to manage file streaming
  - Need to remove files to make space for more files
- Now, suppose that the machine and storage space is a shared resource
  - Need to do the above for many users
  - Need to enforce quotas
  - Need to ensure fairness of space allocation and scheduling

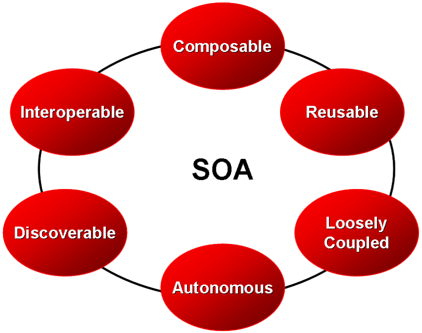
Motivation (2/2)

- Now, suppose you want to do that on a Grid
  - Need to access a variety of storage systems
  - mostly remote systems, need to have access permission
  - Need to have special software to access mass storage systems
- Now, suppose you want to run distributed jobs on the Grid
  - Need to allocate remote spaces
  - Need to move (stream) files to remote sites
  - Need to manage file outputs and their movement to destination site(s)

Service Oriented Architecture and Web Service

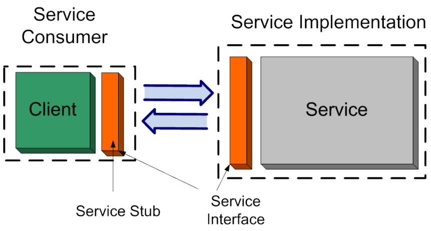
Definition

Service Oriented Architecture is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains.



Service Oriented Architecture and Web Service

Web Service:



- a software system designed to support inter-operable machine-to-machine interaction over a network
- It has an interface described in a machine processable format (specifically WSDL)
- Other systems interact with the Web service in a manner prescribed by its description using SOAP messages

Web Service Description Language (WSDL)

Definition

Web Services Description Language (WSDL): XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information

- Developed by W3C
- Current version 2.0 (2007)
- We refer to version 1.1 because it is currently used in Grid middleware

Storage Resource

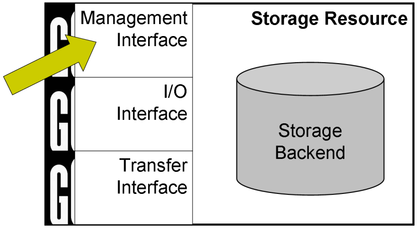
- All data is ultimately stored on a data storage resource (memory stick to a multi-petabyte tape silo)
- Different storage resources offer different levels of Quality of Service (QoS)
- Data is ultimately stored in storage hardware
- The storage hardware will typically be controlled by software

Definition

storage resource is the combination of the storage hardware and the controlling software (eg. file system)

Storage Resource Interfaces

- To make the storage available in the data architecture, the storage resource is wrapped by a storage service
- Three categories of interfaces: Data Access (eg. ByteIO, DAI), Data Transfer (eg. DMIS, GridFTP), Storage Management (eg. SRM)



Storage Resource Manager (SRM) Services

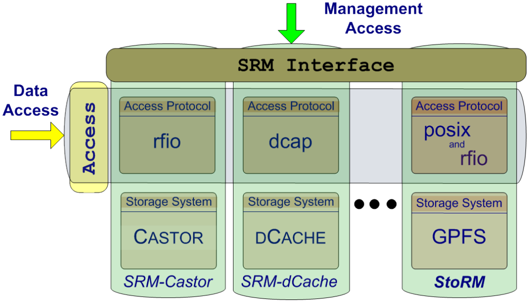
Definition

SRM services are middleware components whose function is to provide dynamic space allocation file management in spaces for shared storage components on the Grid. SRM services implement SRM interface.

- Files are **no longer permanent entities** on the storage, but **dynamic** ones that can appear or disappear according to the user's specification.
- SRMs **do not perform file transfers**, but can invoke middleware components that perform this job (such as GridFTP)

Storage Resource Manager (SRM) Interface

- **Storage Resource Manager (SRM) Interface** describe the management service of a storage element in Grid.
- SRM services agree on a standard interface (**the SRM interface**) to hide storage characteristics and to allow interoperability.



Storage Element and SRM Service

Definition

A **Storage Element** (SE) is a Grid services that allows Grid users to store and manage files together with the space assigned to them

A Storage Element has properties such as:

- *Globally Unique Identifier* that univocally identifies the SE
- *Implementation*: Castor, dCache, DPM, StoRM, ...
- *OnlineSizeTotal and NearlineSizeTotal*.
- ...

A Storage Element expose a set of **Access Protocol** and a **Control Protocol**
The Control Protocol is exposed by the **SRM interface** and it is implemented by a **SRM service** (SRM-dCache, SRM-Castor, StoRM,..)

The origin: Classic Element

The Storage Resource is exposed without a management interface

- The first Storage Server in the Grid based on Globus GridFTP
- Protocols supported: NFS/file, rfio, root, gsiftp

Limitations:

- No possibility to query the service itself about its status, space available, etc. (one has to rely on the Information System)
- Hard to manage the growing of space managed
- No explicit support for tape backend (pre-staging, pool selection, etc.)

The past: SRM v1.1 (1/2)

The need for a standard interface to manage the storage resource in Grid was recognized.

- International collaboration: LBNL, FNAL, CERN, JLAB
- Started in 2001
- Provide basic functionality required
- SRM v1.1 implemented by all major storage providers: CASTOR, dCache, DPM

The main functions:

- Get, getRequestStatus, pin, unpin
- Put, setFileStatus, Copy
- getProtocols, AdvisoryDelete, FileMetaData

The past: SRM v1.1 (2/2)

Main features:

- Asynchronous operations
- Support for bulk requests
- Protocol negotiation

Main problems:

- Missing reference implementation/No clear specs
- No space management
- No explicit quality of storage management
- No abort operations
- No staging operations
- ...

The present: SRM v2.2

- SRM V2.2 enhancements introduced after WLCG (the World-wide LHC Computing Grid) adopted SRM standard
  - Several implementations of v2.2 completed or in-progress
  - extensive compatibility testing
- Open Grid Forum (OGF)
  - Started Grid Storage Management (GSM-WG): GGF8 June 2003
  - Last SRM collaboration meeting Sept. 2006
  - SRM v2.2 spec (for OGF) submitted August 2007

The future: SRM v3.x

Status:

- Development is within OGF GSM-WG
- Functional design is under development

Some considerations:

- When SRM v3.0 will be complete?
- Too complex?
- Waiting for feedback from you.

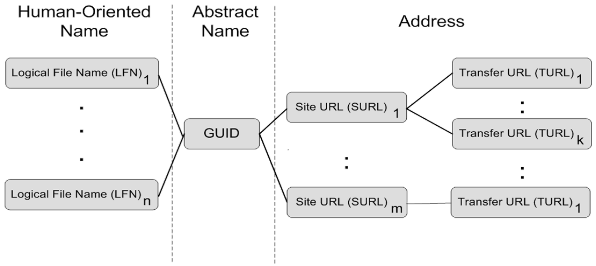
Data Resource and File

Files in the Grid can be referred by different names:

- **Logical File Name (LFN)** : An alias created by a user to refer to some item of data
- **Grid Unique IDentifier (GUID)** : A non-human-readable unique identifier for an item of data
- **Site URL (SURL)** : The location of an actual piece of data on a storage system
- **Transport URL (TURL)** : Temporary locator of a replica + access protocol understood by a SE

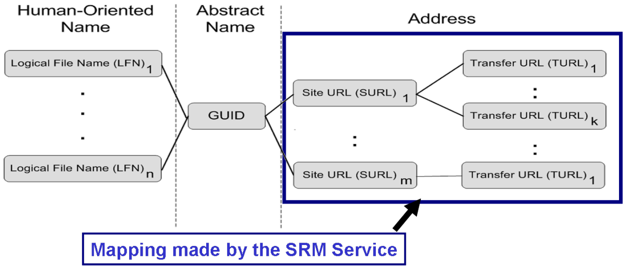
Files in Grid: LFN, GUID, SURL and TURL

While the GUIDs and LFNs identify a file irrespective of its location, the SURLs and TURLs contain information about where a physical replica is located, and how it can be accessed.

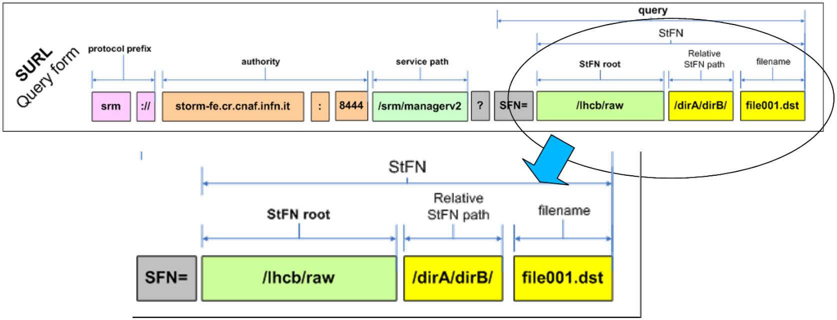


Site URL (SURL) and Transfer URL (TURL)

SURL - TURL : This mapping is managed by Storage Resource Management (SRM) service.



SURL and TURL Structure: SFN e PFN



Storage Model

A definition of the model for a Grid Storage Element represent the foundation to clarify the behavior of the SRM interface implementations.

- F. Donno et al., "**Storage Element Model for SRM 2.2 and GLUE schema description, v3.5**"
- available at: http://glueschema.forge.cnaf.infn.it/uploads/Spec/V13/SE-Model-3.5.pdf

Storage Area and Storage Component (1/2)

A Storage Element can have multiple Storage Areas

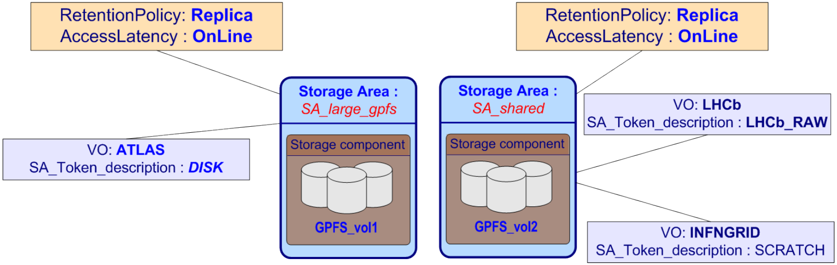
Definition

A **Storage Area** defines a portion of the total available space that can span different kinds of storage devices within a Storage Element and in case of WLCG it implements a Storage Class instance.

Definition

A **Storage Component** identifies a specific storage with certain quality properties.

Storage Area and Storage Component (2/2)



Recalls about Storage Quality

- **Retention Policy**, probability of data loss once stored.
    - **custodial** (High Quality): Custodial quality provides low probability of loss.
    - **output** (Middle Quality): Output quality is an intermediate level and refers to the data which can be replaced by lengthy or effort-full processes.
    - emph**replica** (Low Quality): it has the highest probability of loss, but is appropriate for data that can be replaced because other copies can be accessed in a timely fashion
- **Access Latency**,
    - *On-line*: Storage where files are moved to before their use
    - *Near-line*: Requires latency before files can be accessed

Recalls about Storage Quality in WLCG

Definition

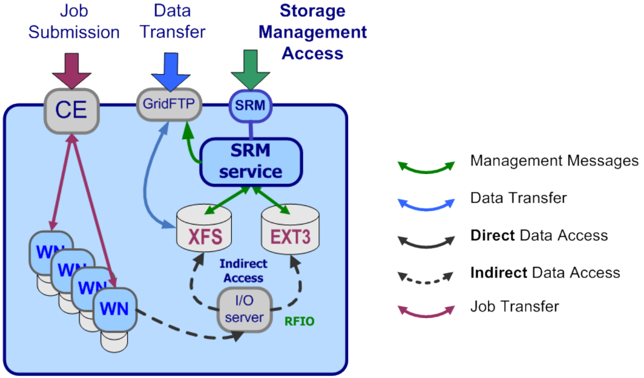
storage class is a classification of quality of storage system defined in terms of the *Retention Policy* and *Access Latency*

Asking for a certain storage class users can specify the desired storage system characteristics to store data.
In WLCG has been decided mapping the quality of storage together with the combinations of storage devices.

- **Custodial-Nearline**, In WLCG **T1D0**
- **Custodial-Online**, In WLCG **T1D1**
- **Replica-Online**,In WLCG **T0D1**

SRM role in a site

In a **Grid Storage Element**, the SRM service provides the functionality to manage file and spaces.



SRM v2.2 in a snapshot

The SRM v2.2 is based on these concepts:

- **lifetime** of a file (volatile with a fixed lifetime, durable or permanent).
- **file pinning** (to ensure a file is not canceled during operation).
- **space pre-allocation** (to ensure the request space is available for the whole life of the application since the beginning).
- **storage classes** to identify different quality of storage resources.

File Storage Type

- **ReleaseWhenExpired (volatile)**:
  - Temporary files with a lifetime guarantee
  - Files are *pinned* for a lifetime and can be *released* by client
  - Files can be removed by SRM when released or when lifetime expires
- **NeverExpired (permanent)**
  - No lifetime
  - Files can only be removed by creator (owner)
- **WarnWhenExpired (durable)**
  - files with a lifetime that CANNOT be removed by SRM
  - Files are *pinned* and can be *released*
  - Files can only be removed by the replica creator
  - If lifetime expires invoke administrative action (e.g. notify owner, archive and release space)

About Tokens

In SRM v2.2 there are different tokens.

Definition

Request Token: Represent a unique identifier of a request. Only asynchronous request holds a request token. It is used by user to retrieve the status of asynchronous request.

Definition

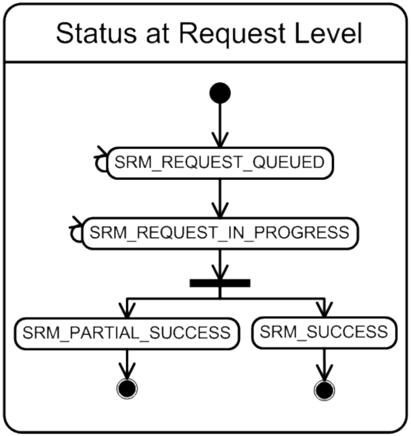
Space Token: Represent a unique identifier of a reserved space.

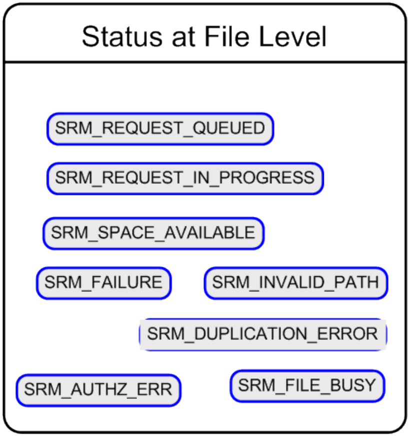
Definition

User Space Token Description: Represents a human-readable alias of reserved space. It is case-sensitive and it is chosen by the user.

Status Code and Request Status example

An example of Return Status Code for *srmPrepareToPut* methods.





Protocol Negotiation

- Client provides an ordered list of protocols
- SRM return: first protocol from the list it supports
    - Protocols list: `gsiftp`, `file`, `rfio`
    - SRM returns: `gsiftp`
    - Example: `gsiftp://storm.cnaf.infn.it:2811/ss/atlas/file.txt`
- Advantages:
    - Easy to introduce new protocols
    - User controls which protocol to use

Space Definitions

Definition

Space: A space is a chunk of disk storage. A file lives within a space.

Definition

Static Space Reservation: It represent a Storage Area within a SE. Typically controlled by production managers of big VOs.

Definition

Dynamic Space Reservation: It represent a space reserved by a user through SRM functions.

SRM v2.2 functionalities

The SRM functionalities can be grouped in the following categories:

- **Data management** provides capabilities to manage the data stored in a SRM, preparing envinronment for access operation (staging, etc.) and for receiving new data.
- **Space management** space can be reserved by user to have the guarantee of availability when the transfer operation takes place.
- **Directory management**, this is a set of UNIX-like directory operation used to manage the SRM namespace for creating, moving and deleting directories and files.
- **Query functions**, this set of function is used to discover information on a specific SRM endpoint.

Synchronous and Asynchronous functionality

SRM service provides provides two classes of methods:

- **Asynchronous methods**, **non blocking call**. Return a token corresponding to the request, the client can retrieve at any time the status of the request by addressing it through such a token. This is the case for data management functionalities.
- **Synchronous methods**, **blocking call**. The control is returned to the client only when the request is completed. This is the case of directory management and space management functions.

Directory Management functions

- Usual unix semantics
  - srmLs, srmMkdir, srmMv, srmRm, srmRmdir, srmCp
- A single directory for all spaces
  - No directories for each file type
  - File assignment to spaces is virtual
- Access control services : Support owner/group/world permission

Data Transfer functions

- srmPrepareToGet
- srmStatusOfGetRequest
- srmPrepareToPut
- srmStatusOfPutRequest
- srmCopy
- srmStatusOfCopyRequest
- srmBringOnLine
- srmStatusOfBringOnline
- srmReleaseFile
- srmPutDone
- srmAbortRequest
- srmSuspendRequest
- srmResumeRequest
- srmGetRequestSummary
- srmExtendFileLifeTime
- srmGetRequestTokens

Discovery functions

- srmGetTransferProtocol
- srmPing

Managing Spaces

- **Default spaces**:
  - Files can be put into an SRM without explicit reservation
  - Defaults are not visible to client
- Files already in the SRM can be moved in other spaces by
  - `srmChangeSpaceForFiles`
- Files already in the SRM can be pinned in spaces
  - By requesting specific files (srmPrepareToGet)
  - By pre-loading them into online space (srmBringOnline)
- Updating space
  - Resize to request more space or to Release all unused space
  - Extend or Shorten the lifetime of a space

Space Management functions

- srmReserveSpace
- srmStatusOfReserveSpaceRequest
- srmStatusOfUpdateSpaceRequest
- srmGetSpaceMetaData
- srmChangeSpaceForFiles
- srmExtendFileLifeTimeInSpace
- srmStatusOfChangeSpaceForFileRequest

- srmReleaseSpace
- srmUpdateSpace
- srmPurgeFromSpace
- getGetSpaceTokens

srmPing

srmPing : This function is to discover what transfer protocols are supported by the SRM.

srmPrepareToPut

srmPrepareToPut() is used to write files into the storage.
Upon the clients request, SRM prepares a TURL so that client can write data into the TURL.
Lifetime (pinning expiration time) is assigned on the TURL.

- target space token (Non mandatory)
- asynchronous operation
- Request token returned by SRM service
- Status may be checked through `srmStatusOfPutRequest` with the returned request token

srmPrepareToGet

srmPrepareToGet is used to bring files online upon the clients request.
It assigns TURL so that client can access the file.

- target space token (Non mandatory)
- asynchronous operation
- Request token returned by SRM service
- Status may be checked through `srmStatusOfGetRequest` with the returned request token

srmStatusOfGetRequest and srmStatusOfPutRequest

- **srmStatusOfGetRequest** : This function is used to check the status of the previously requested srmPrepareToGet. Request token from srmPrepareToGet must be provided.
- **srmStatusOfPutRequest** : This function is used to check the status of the previously requested srmPrepareToPut. Request token from srmPrepareToPut must be provided.

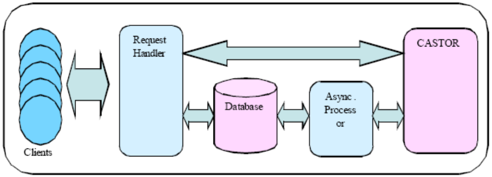
srmPutDone and srmReleaseFiles

- **srmPutDone** : srmPutDone() is used to notify the SRM that the client completed a file transfer to the TransferURL in the allocated space. This call should normally follow srmPrepareToPut.
- **srmReleaseFiles** : srmReleaseFiles() is used to release pins on the previously requested "copies" (or "state") of the SURL. This function normally follows srmPrepareToGet or srmBringOnline functions.

srmReserveSpace and srmGetSpaceMetadata

- **srmReserveSpace** : srmReserveSpace() is used to reserve a space in advance for the upcoming requests to get some guarantee on the file management. Asynchronous space reservation may be necessary for some SRMs to serve many concurrent requests.
- **srmGetSpaceMetadata** : srmGetSpaceMetadata() is used to get information of a space. Space token must be provided, and space tokens are returned upon a completion of a space reservation through srmReserveSpace or srmStatusOfReserveSpaceRequest.

CASTOR-SRM [CERN and RAL]

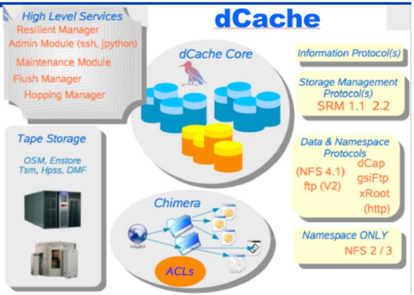


- **CASTOR is the HSM in production at CERN**
- **Support for multiple tape robots**
  - Support for Disk-only storage recently added
- **Designed to meet Large Hadron Collider Computing requirements**
  - Maximize throughput from clients to tape (e.g. LHC experiments data taking)
  - ***More on: G. Lo Presti et al., CASTOR, MSST Poster Session***

- **C++ Implementation**
- **Reuse of CASTOR software infrastructure**
  - Derived SRM specific classes
- **Configurable number of thread pools for both front- and back-ends**
- **ORACLE centric**
- **Front and back ends can be distributed on multiple hosts**

dCache-SRM, [Fermilab and DESY]

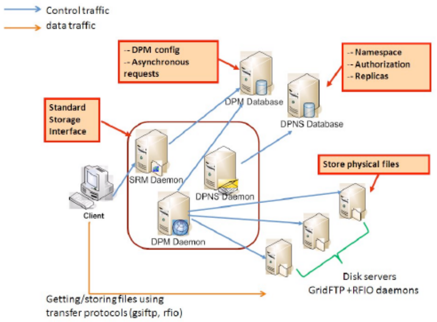
- Strict name space and data storage separation
- Automatic file replication on based on access patterns
- HSM Connectivity (Enstore, OSM, TSM, HPSS, DMF)
- Automated HSM migration and restore
- Scales to Peta-byte range on 1000's of disks
- Supported protocols:
- (gsi/krb)FTP, (gsi/krb)dCap, xRoot, NFS 2/3
- Separate IO queues per protocol
- Resilient dataset management
- Command line and graphical admin interface
- Variety of Authorization mechanisms including VOMS
- Deployed in a large number of institutions worldwide



- SRM 1.1 and SRM 2.2
- Dynamic Space Management
- Request queuing and scheduling
- Load balancing
- Robust replication using SrmCopy functionality via SRM, (gsi)FTP and http protocols

Disk Pool Manager (DPM), [CERN]

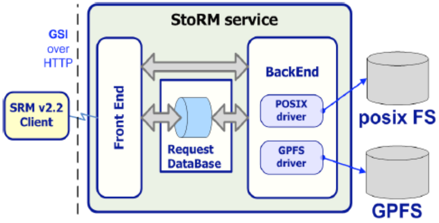
- **Manages storage on disks only**
- **Security**
  - GSI for authentication
  - VOMS for authorization
  - Standard POSIX permissions + ACLs based on user's DN and VOMS roles
- **Virtual ids**
  - Accounts created on the fly
- **Full SRMv2.2 implementation (srmCopy being done)**
- **Standard disk pool manager capabilities**
  - Garbage collector
  - Replication of hot files
- **Transfer protocols**
  - GridFTP
  - Secure RFIO
- **Easy to install and administer**



- **Supported database backends**
  - MySQL/Oracle
- **High availability**
  - All servers can be load balanced (except the DPM one)
  - Resiliency: all states are kept in the DB at all times

Storage Resource Manager (StoRM), [INFN, ICTP-EGRID]

- It's designed to leverage the advantages of high performing parallel file systems in Grid.
- Different file systems supported through a driver mechanism:
  - generic POSIX FS
  - GPFS
  - Lustre
  - XFS
- It provides the capability to perform local and secure access to storage resources (*file://* access protocol + ACLs on data).



StoRM architecture:

- Frontends: C/C++ based, expose the SRM interface
- Backends: Java based, execute SRM requests.
- DB: based on MySQL DBMS, stores requests data and StoRM metadata.
- Each component can be replicated and instantiated on a dedicated machine.

Conclusion

- Storage Resource Management (SRM) is essential for Grid
- Multiple implementations interoperate
- Permits interchanging one SRM product by another
- Multiple SRM implementations exist and are in production use
- Storage Element Model
- SRM v2.2 functions

Next session

- Analisi di un servizio SRM: StoRM
- Presenter: Luca Magnoni

Question

StoRM
Storage Resource Manager

http://storm.forge.cnaf.infn.it

INFN GRID

Antonia Ghiselli
Alberto Forti
Luca Magnoni
Riccardo Zappi

Ezio Corso

References

- SRM Spec
- Storage Model
- StoRM home page