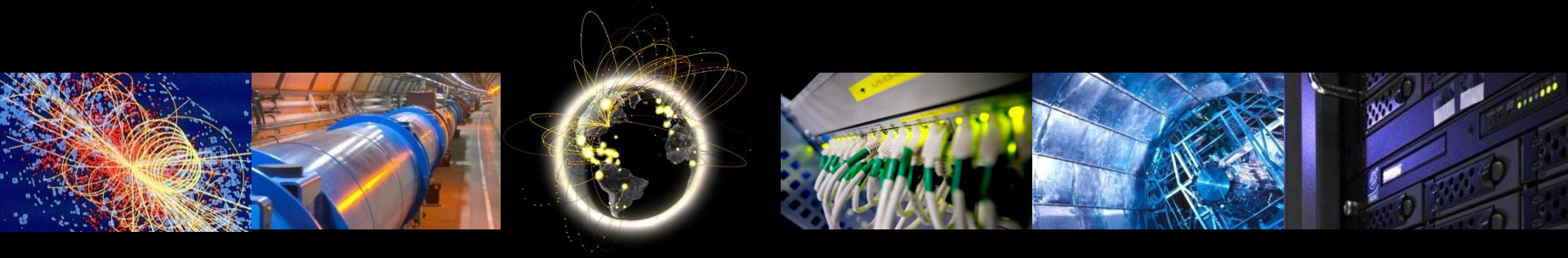
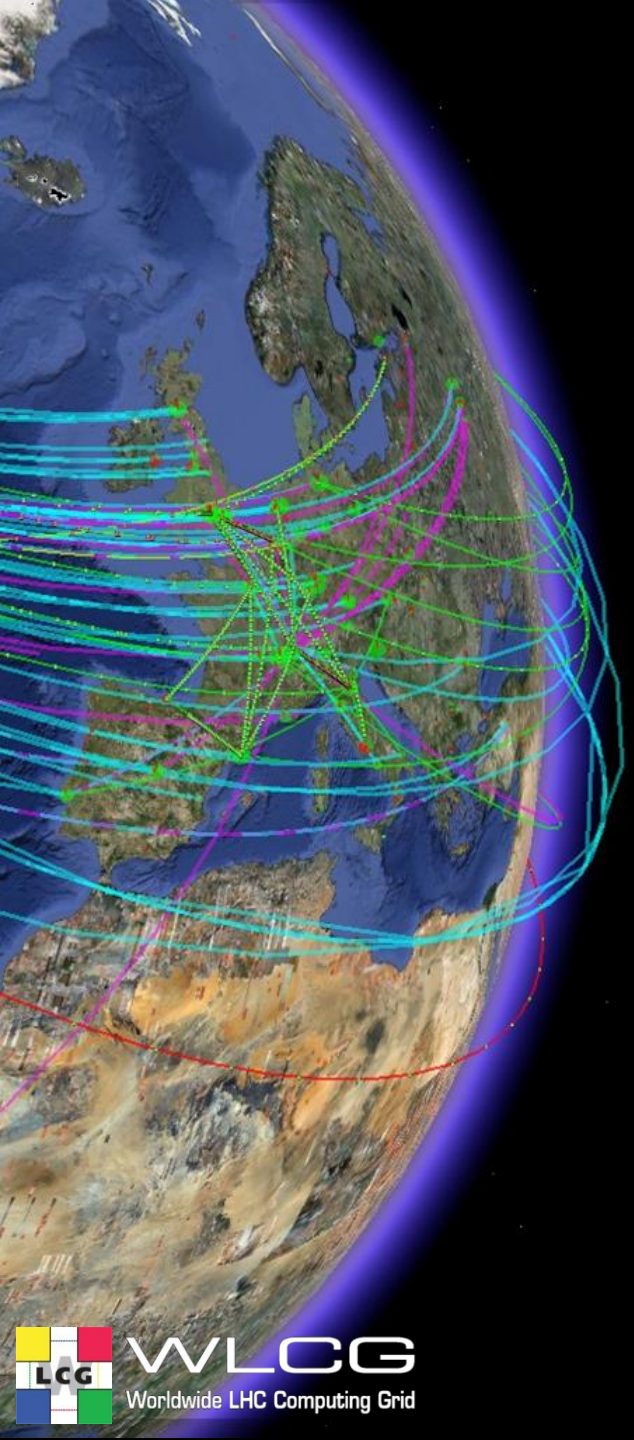
SRM-2.2
Lessons Learned

Markus Schulz

June 2010 DM Jamboree





Overview

- SRM-2.2
  - Motivation and Short History
  - Protocol
  - Implementation
  - Usage
- Main Issues
- All Bad?
- Lessons Learned
- More History

Motivation (for HEP)

- Distributed processing model
  - Data Management, Data Access, Storage Resource Management
  - User community is experiment centric
    - No longer institute centric
    - Requires radical change in Authentication/Authorisation technology
- But:
- Many existing and heavily used heterogeneous (local) storage systems
  - Different models and implementations for
    - local storage hierarchy
      - transparent/explicit
    - Synchronous/Asynchronous operations
    - Cluster file system based / disk server based
    - Plethora of Data Access Clients
    - Authorization and authentication
      - Often local, mostly UID/GID or AFS like ACLs, Kerberos, +++
    - Wide area transfers
      - FTP doors, proprietary
    - ……
  - Deeply integrated with local computing fabrics
  - Representing decade long, massive investment
  - Have to respect local policies and resource allocations

Approach

- An abstraction layer for storage and data access is necessary
  - Guiding principle:
  - Non-interference with local policies

- Providing all necessary user functionality and control
  - Data Management
  - Data Access
  - Storage management
  - Control:
    - Pinning files
    - Retention Policy
    - Space management and reservation
  - Data Transfers
- Grid enabled and based on current technology
  - Interface technology (gSOAP)
  - Security Model (gsi security)
  - To integrate with the grid infrastructure

Functional Model

- Catalogue(s) translate
  - Logical File Names (LFN) to GUID to Storage URLs (SURL)
    - LFNs can have aliases
    - GUIDs are unique and static
    - SURLS are static
- SRM interface translates
  - SURLs to Transport URLs which are mutable, but stable during operations
    - Decouples external name space from local
    - Allows for transparent Storage Management
    - Flexibility for users and providers
  - Data management and user commands to local operations
  - Flexible to allow for multiple protocols
  - Specify (many) commands as asynchronous
    - Allows the underlying system to schedule operations
      - Needed to implement local and project priorities
  - Translates grid credentials to local credentials (as far as possible)
- Translation is not necessary straight forward
  - storage system can have no corresponding functionality
  - follows a different architectural concept
    - Pinning, retention policy, AAA, async/sync
  - → many optional features

A Short History of SRM

- SRM is based on concepts that have been around for more than 10 years
- SRM-V2.2 requirement gathering and first implementations took for a standard a very short time
  - From proof of concept to "production" less than a year
    - Painful for early adopters….. But they had no choice.
  - Comparsion: NFS-4.1 started in 2005 and finished late 09
- Verification tests appeared late in the game
  - Helped massively to synchronize
- First large scale production experience required re-adjustments (requirements → implementations)
  - HEP Use Case Addendum
- See appendix for details

Methods

- Standard Document ~100 pages
  - Relative short (see NFS-4.1 ~260 pages)
- Space Management (11)
- Permission (3)
- Directory (6)
- Data Transfer (17)
- Discovery (2)
- Too many?
  - Not as many as a naïve count indicates
  - Most methods are asynchronous -> divide by two
    - Srm<method> and srmStatusOf<method>Request
- Very flexible behavior → complex verification/clients
- WLCG Addendum tried to simplify both dimensions

Implementation(s)

- WSDL to generate client stubs
- gSOAP
- GSI for A(A)

- External interface is the “easy” part
- Integration with the existing storage system is the challenge
- Different approaches concerning the accepted level of entanglement between SRM and storage
  - New storage systems have less problems
    - Designed for SRM

Usage

- Intended Use:
  - Agreement that users will access SRM via high level interfaces
    - FTS, gfal, LCGutils
    - Adding an extra layer to ease usage
  - Data management
    - Organized data movements between sites
    - Actively managing the space(s)
  - Data access
    - File by file basis, TURLs and protocols from SRMs
    - Space reservation etc.

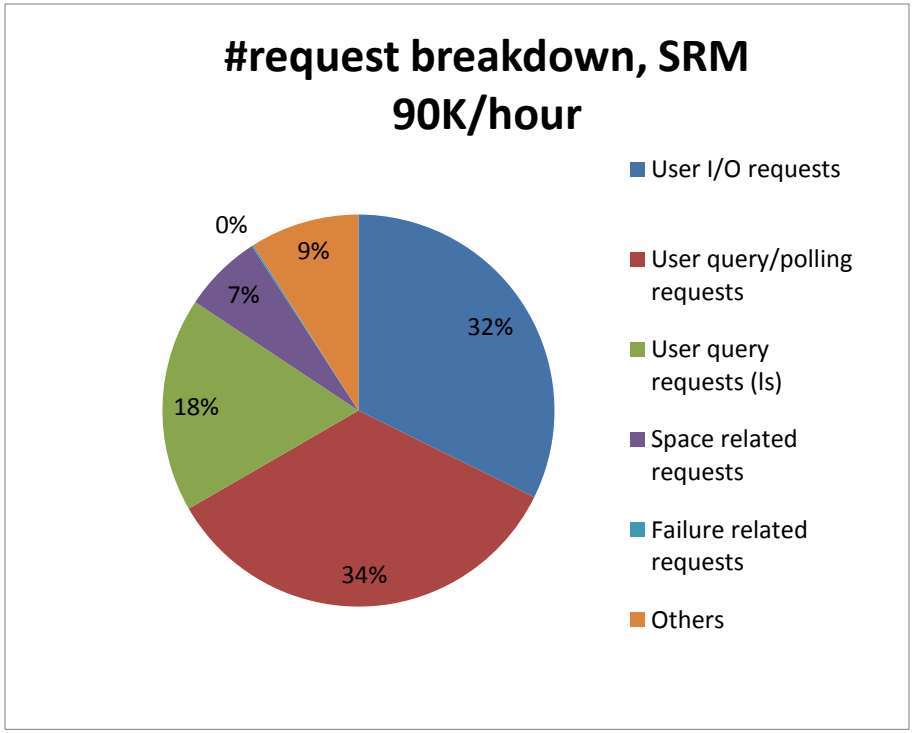
Actual Usage

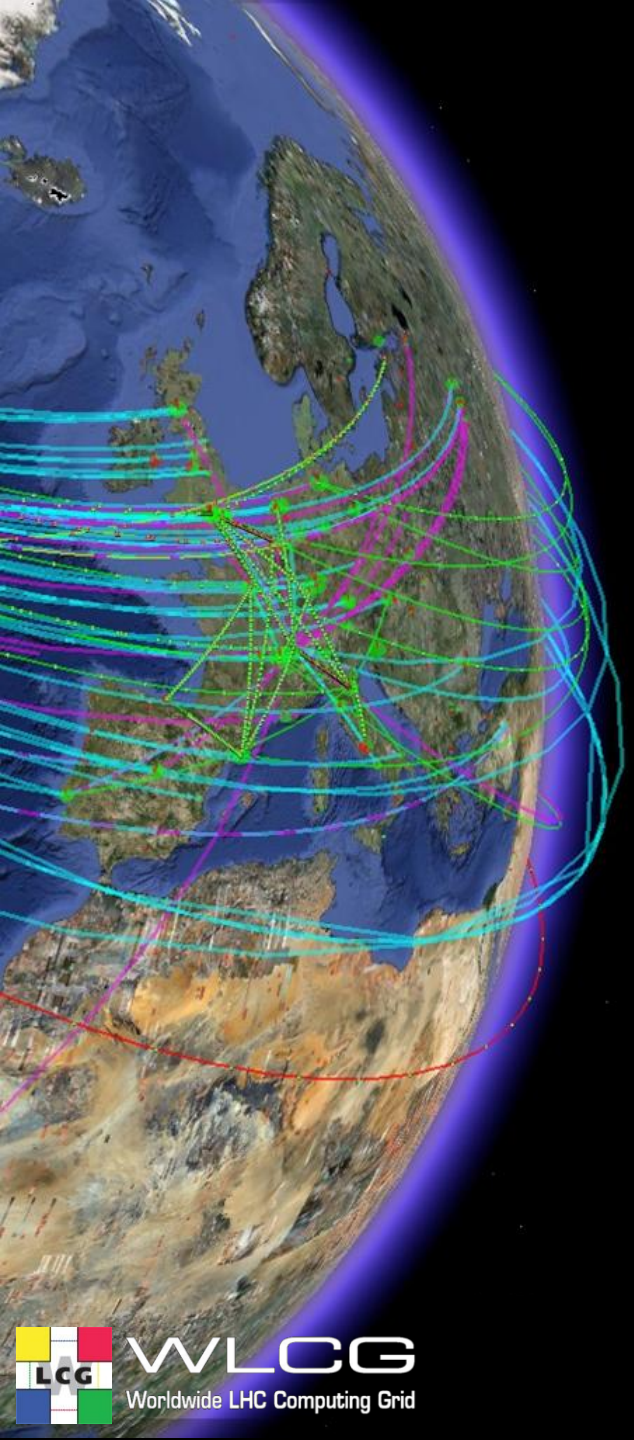
- Best by looking at some of the SRM endpoints
  - Some T0 and T1 statistical data is available
  - Certainly not representative for all sites
    - But at least some data…..
    - Thanks to Dirk and Giuseppe
- Grouping the methods for clarity
  - User I/O requests
    - srmPrepareTo, srmCopy, srmBringOnline, srmReleaseFiles ...
  - Failure related requests
    - srmAbortRequest, srmAbortFiles, ….
  - Polling/query requests
    - srmPing, srmStatusOf, srmLs
  - Space related requests
    - srmGetSpaceTokens, srmReserveSpace, ...
  - Others - >**10 more methods (not all used)**

Usage Statistic

- Example: ATLAS at CERN June 1st to 9th 2010
- Average SRM request rate: 25Hz (2.4 Hz 2009)

0	"srmAbortFiles"	f
1446	"srmAbortRequest"	f
6903	"srmBringOnline"	u
0	"srmCheckPermission"	o
0	"srmCopy"	u
258455	"srmGetSpaceMetaData"	s
376399	"srmGetSpaceTokens"	s
1721196	"srmLs"	l
136961	"srmMkdir"	o
739643	"srmPing"	o
719405	"srmPrepareToGet"	u
366364	"srmPrepareToPut"	u
727130	"srmPutDone"	u
1324662	"srmReleaseFiles"	u
12876	"srmRm"	f
3	"srmRmdir"	o
0	"srmStatusOfCopyRequest"	q
2036128	"srmStatusOfGetRequest"	q
1310692	"srmStatusOfPutRequest"	q





So… what went wrong with SRM?

Problems 0

Conceptual Problems:

- SRM tries to abstract and hide underlying storage management system differences
- AND
- Exposes internal storage management functionality
- AND
- Does it for two fundamentally different use cases

- Leads to complex interferences and problems

Problems 1

- We rushed the SRM specification and implementation
  - Not enough time for iterative improvements
  - NFS-4.1 (270 pages, 5 years)
    - Large community/ Reference clients +++
- S2 tests arrived late
  - Helped massively
- Large effort to align different implementations
  - Still not 100% coherent
  - Castor can't handle pinning, dCache only one copy…

Problems 2

- Initially the implementations had severe stability and performance problems
  - Thread exhaustion, authentication overhead (GSI)
  - **These problems have been solved**
    - **And there are plans for further improvements**
- ACLs and permissions are not well covered
  - Incoherence between implementations
  - Limited by the Storage Systems
- No standard client library
  - Several: FTS, gfal, dCache, S2 test clients …

Problems 3

- Space Management
  - Rich set of functionality
  - Overly complex for most uses
    - basically “offline” reservation at T0 and T1
  - Not properly used
    - Used to manage Quality of Service requests
- Quota management missing
- Some methods not meaningful for all implementations
  - srmCopy
  - Additional complexity for all

Problems 4

- Interface for data management (WAN transfers) and local access
  - T1 and T0 focus
    - Affects complexity and reliability of T2 data management
- Conflicting requirements:
  - Data Management
    - scheduling capabilities ,decoupling, overall throughput, space reservation, retention policy,…
  - Local Access
    - ease of use, responsiveness,
- Users find ways to solve their problems…
  - Status methods for asynchronous operations were initially not reliable
    - Users used srmLs
    - Expensive operation → Sync/Async implementations
  - Storage <-> catalogue (re) synchronization needed
    - SRM-LS not suitable (no cursor functionality to handle large output)
    - Users work with name server DB dumps…. Not a standardized interface
  - SRM far too expensive and complex for data access
    - **String manipulation bases SURL to TURL translation**
- And almost never come back to the official path

Problems 5

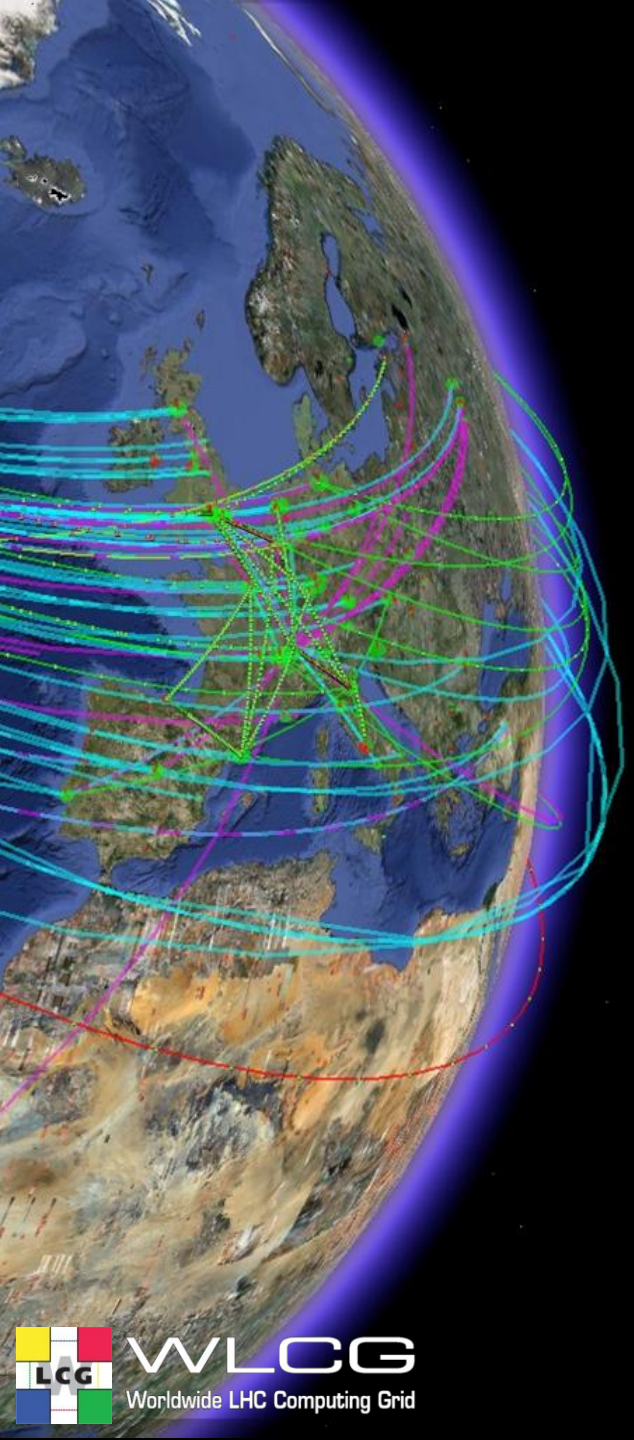
- FTS and other Data Management tools
  - Difficult error handling and propagation
    - Multi layer product
    - End to end problem analysis is very difficult
  - No good backpressure mechanism
    - SRM front end oversubscription
    - Storage backend oversubscription
    - Single hot files
    - All lead to (complex) timeouts…
  - Bulk method concept could be extended to sets
    - Saves on auth overhead
    - Iterator (next) instead of sequential processing

Problems 6

- Storage Classes and Space management link different concepts
  - Quality of Service
  - Pool allocation for different activities
  - Space reservation for operations
  - Retention Policy management
  - Problematic transition between classes
    - often an internal extra copy needed
- Monitoring and analysis for performance tuning
  - Individual efforts
    - Quite late and at small scale
  - Due to multi step, multi service nature a combined approach is needed
  - We drive blind folded
    - And complain that we can't see

Other Problems

- gSOAP didn't help implementation
  - Tool chain not really up to the task
  - Frequent duplication of code
    - Maintenance issue
  - Web Service like, but different
- GSI (modified openssl) made security complex
- Lack of a common local data access protocol
- Clash of concepts between Storage Systems and SRM
  - Protocol and Storage developed independently
  - "Impedance" mismatch
    - Some functionality notoriously difficult to implement (see: abort in Castor)
- Many smaller ones …..



So… SRM is useless?

SRM's Potential Role

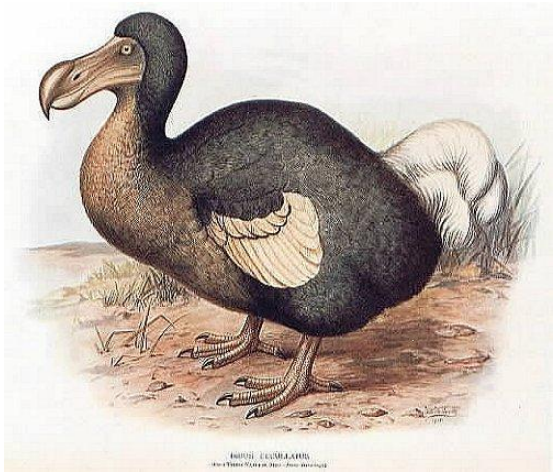
- Complex, large storage systems require a scheduling and storage management layer between users and the raw disks
  - For most of the data management tasks a subset of the current SRM seems adequate
  - Establishing a protocol is costly
  - Verification of different implementations is costly
  - Performance and scalability improvements have been costly
  - SRM is now of the same quality as the storage systems
    - Entering the domain of diminishing return on investment

SRM's Potential Role

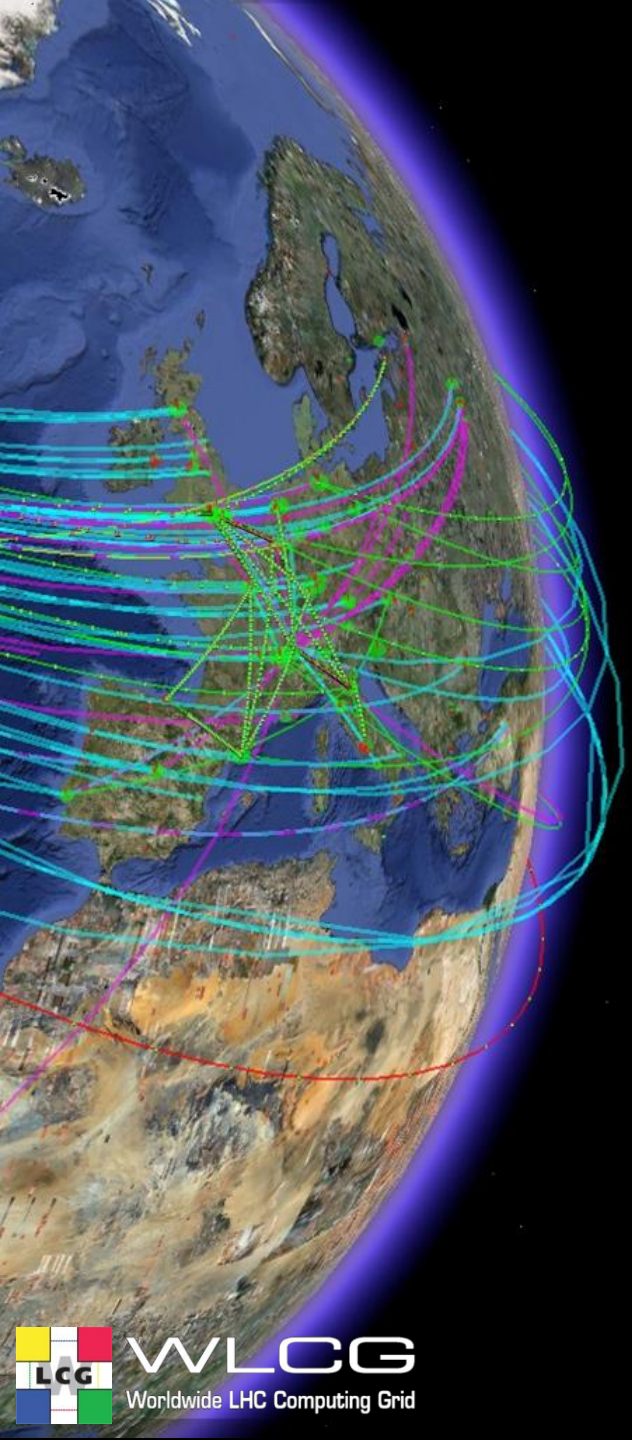
- The cost for a radical paradigm shift has to be realistically assessed
  - How much can be gained in end to end improvement?
    - 10%, 20% ? At what costs/savings?
- SRM and its implementations can be improved
  - Further reduction of scope
  - Investment in monitoring and performance analysis
  - Improving error handling
  - Some of the scalability issues can be addressed
  - Handling non core functionality outside SRM
    - Storage – Catalogue synchronization ….

SRM's Potential Role

- Data Access, especially analysis
- SRM will go the way of the Dodo
- Users bypass the system in different ways
  - leads to invasive implicit constraints on storage systems



- Need a common approach for data access
  - best not adding too many layers…….



Lessons Learned?

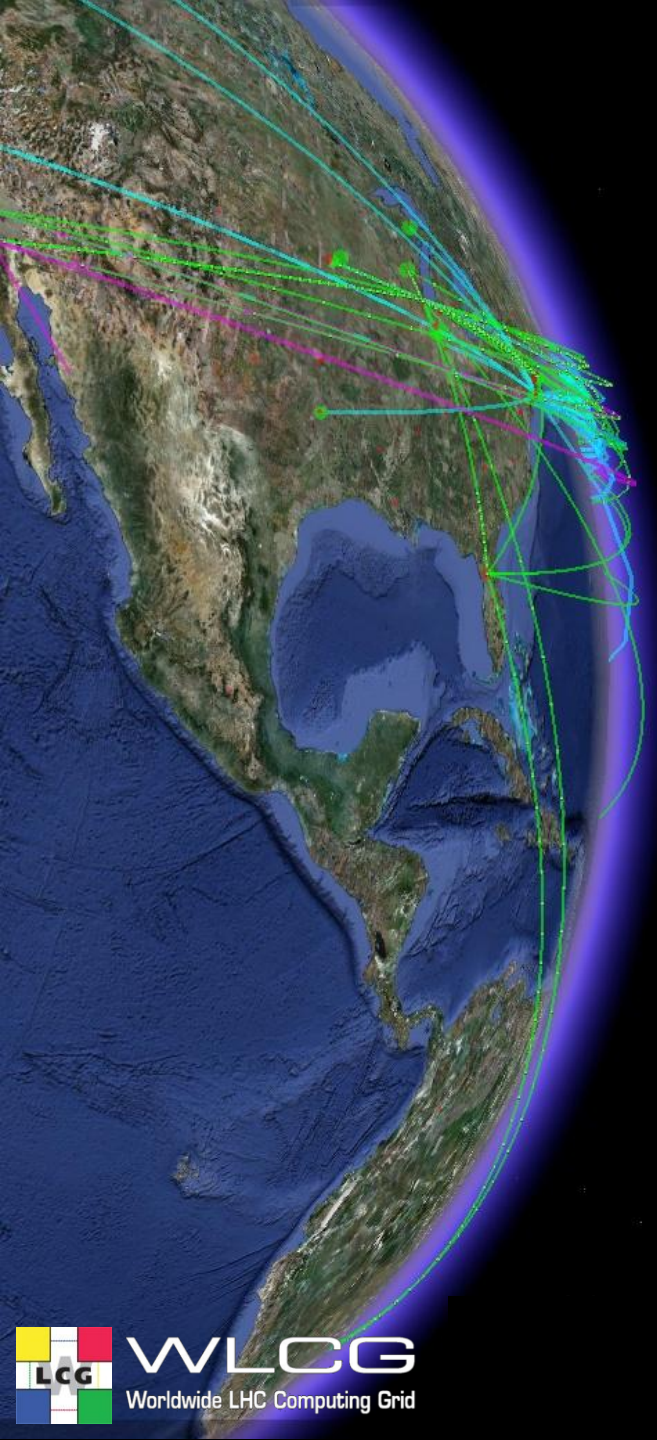
Lessons Learned

Mostly trivial…

- Development of a protocol requires time, resources and commitment form all involved
  - And realistic expectations
  - Timescale is years not months
- Only address functionality that is essential
  - Adding is is easier than removing
- Clear boundary between interface and underlying system
  - What is exposed, what is managed by the system
- Storage Systems/Use Cases/Protocols best go through a co-evolution
  - Difficult for systems that are already established
- AA and security have to be in the protocol from the start
  - Or will gum up any implementation when bolted on later
  - Difficult for systems that are already in use
- Use cases that are sufficiently different often require different approach
  - Data Management/data access

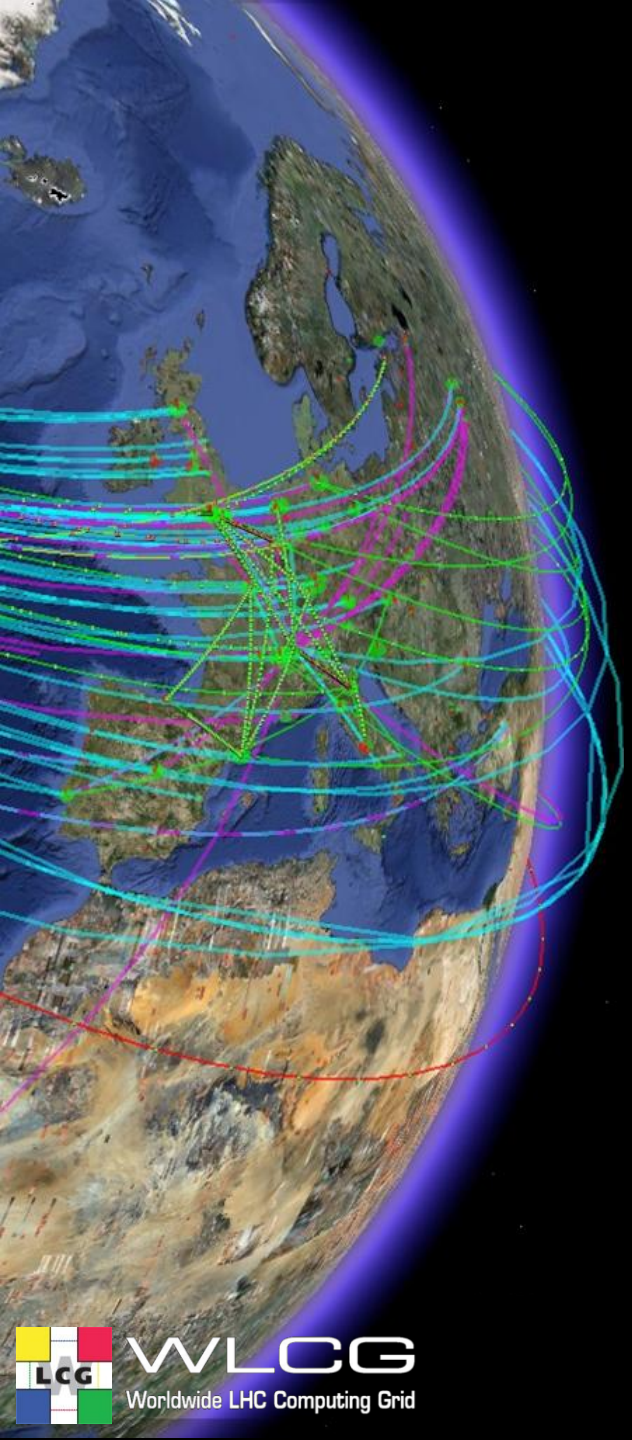
Lessons Learned

- Different aspects of a storage system should not be mixed
  - QoS, Space Management, Retention Policy,Quota
- Expect failure
  - define time behavior, retries, alternative paths
- Provide a functional reference implementation
  - Only for verification use
  - Provide verification tests
- Propagate error messages in a meaningful way
- Mandatory monitoring as part of the protocol
  - Or addressing performance and scalability issues requires black magic
- Choose implementation technology consciously



Special Thanks to

- Flavia Donno
- Jean-Philippe Baud
- Patrick Fuhrmann
- Gavin Mccance
- Andrea Sciaba
- Dirk Düllmann
- Giuseppe Lo Presti



Communities never learn from history

History

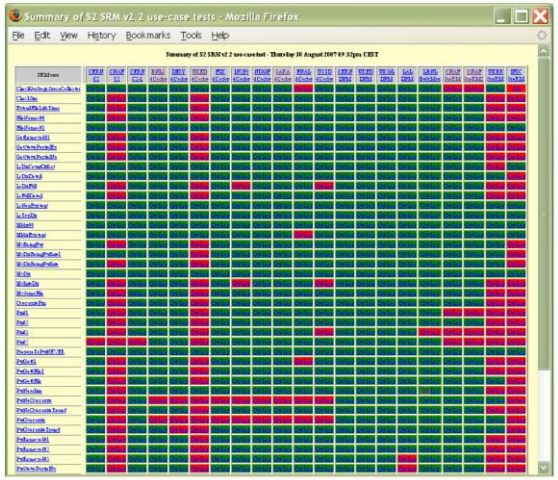
- Before 2000
  - Several ideas for data management and storage
    - HPSS, SRB etc.
  - 99 Super Computer Conference
    - GridStorage, first concepts of GSI + FTP
- 2000 GridFtp
  - WAN access + grid authentication
- 2001 proto SRM at LBNL (DoE funded)
  - HPSS + Cache Manager
- 2003 LBNL, FNAL, Jefferson Lab, CERN
  - OGF SRM WG
  - gridFTP, space reservation, pinning……..
- 2004 EGEE started, SRM-2.0
  - First, mainly disk based demonstrators

History

- 2004-2005
  - User, developer and site feedback
    - Lack of functionality
    - Ambiguities
  - WLCG input let to the definition of SRM-2.1
- June 2005 Baseline Service Working Group Report
  - Requirements stated (based on SRM-v1.1/2.1)
  - SRM usage via high level tools (gfal, LCGUtils, FTS,)
  - Target date for implementation: February 2006
- 2006 CHEP + WLCG Workshop in Mumbai
  - Collection of critique and suggestions
  - Space token, spaces, tape <-> disk transitions
  - New requirements (based on more experience)
- Meeting at FNAL in May led to SRM-2.2 agreement
  - CERN/DESY/FNAL/INFN/LGNL/RAL/TJNAF/ICTO
- May-December 2007 weekly meetings to track implementations
- WLCG defined a subset of the functionality
  - To ensure implementation in time for LHC start
  - Clarify some of the semantics

History

- 2007-2010
  - S2 test suite run daily
    - Interoperation between SRMs
    - Clarification of semantics
    - Stress testing
    - “The Flavia Tables”
    - About 30 open issues in January 2007
  - SRM-v2.2 rollout (GSSD WG)
    - Usage of instances in the process of implementation
    - Accelerated maturity ……
  - Service Challenges
  - CCRC08, STEP09
  - May 2008 finalization of of the specification
  - Addendum to the SRM Usage Agreement
    - Different behavior for different backends (Spaces, Storage Classses)
  - September 2009 minor revision
  - Tuning and fixing of implementations….



Links

- https://sdm.lbl.gov/srm-wg/
- http://cdsweb.cern.ch/record/1083653/files/CERN-THESIS-2008-008.pdf
- http://s-2.sourceforge.net/
- http://lcg.web.cern.ch/LCG/peb/bs/BSReport-v1.0.pdf
- https://twiki.cern.ch/twiki/bin/view/LCG/GSSD