Grids and Clouds Interoperation: Development of e-Science Applications Data Manager on Grid Application Platform

Hsin-Yen Chen & Wei-Long Ueng
Academia Sinica Grid Computing

5th EGEE User Forum
Uppsala, Sweden

Outline

- Principles of the e-Science Distributed Data Management
- Introduction to GAP (Grid Application Platform).
- Putting it to Practice
- GAP Data Manager Design
- Summary

The e-Science Data Flood

- Instrument data
  - Satellites
  - Microscopes
  - Telescopes
  - Accelerators
  - ..
- Simulation data
  - Climate
  - Material science
  - Physics, Chemistry
  - ..
- Imaging Data
  - Medical imaging
  - Visualizations
  - Animations
  - ..
- Generic Metadata
  - Description data
  - Libraries
  - Publications
  - Knowledge base
  - ..

Data Intensive Sciences

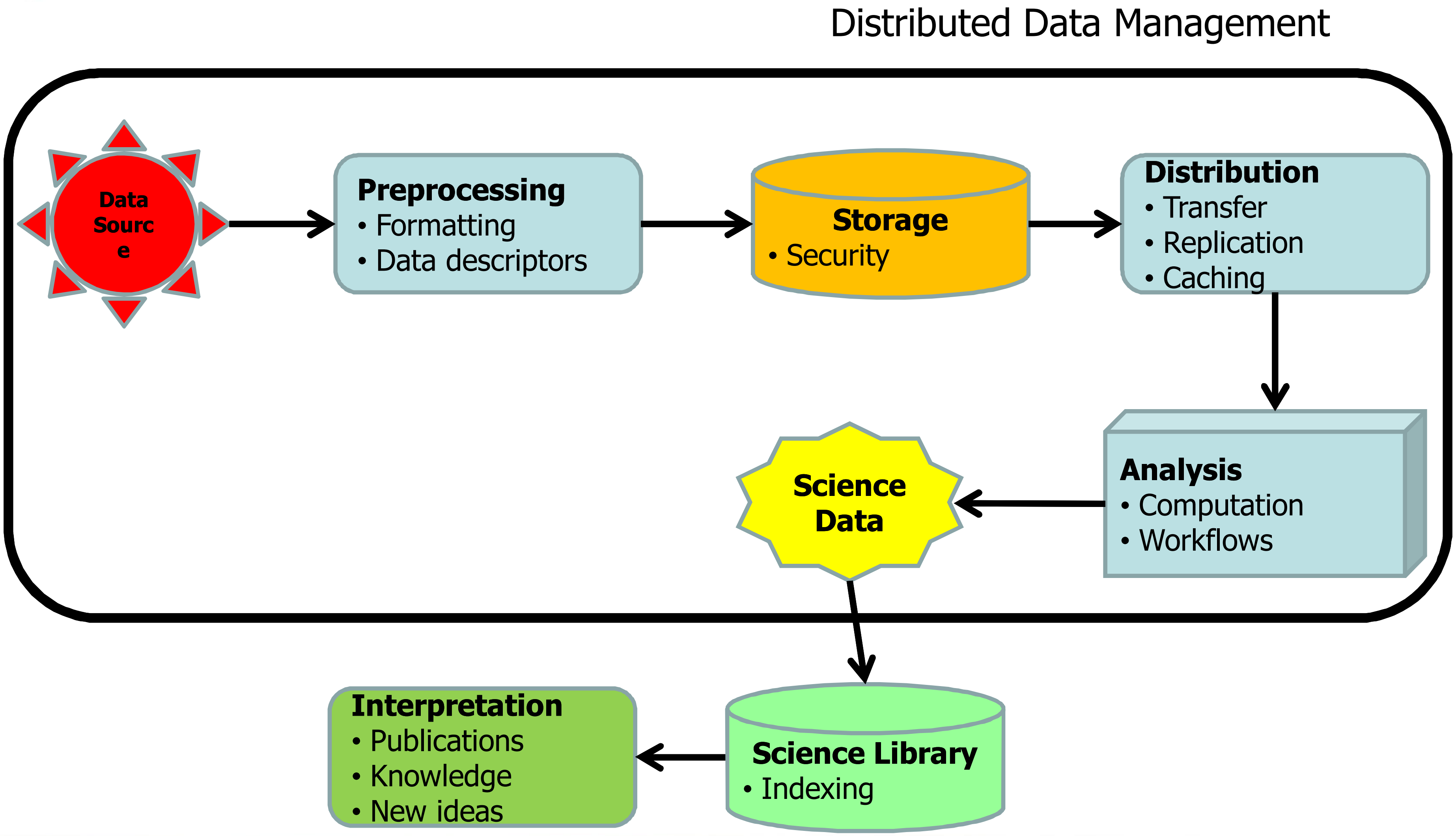
Data Intensive Sciences **depend** on Grid Infrastructures

Characteristics: any one of the following

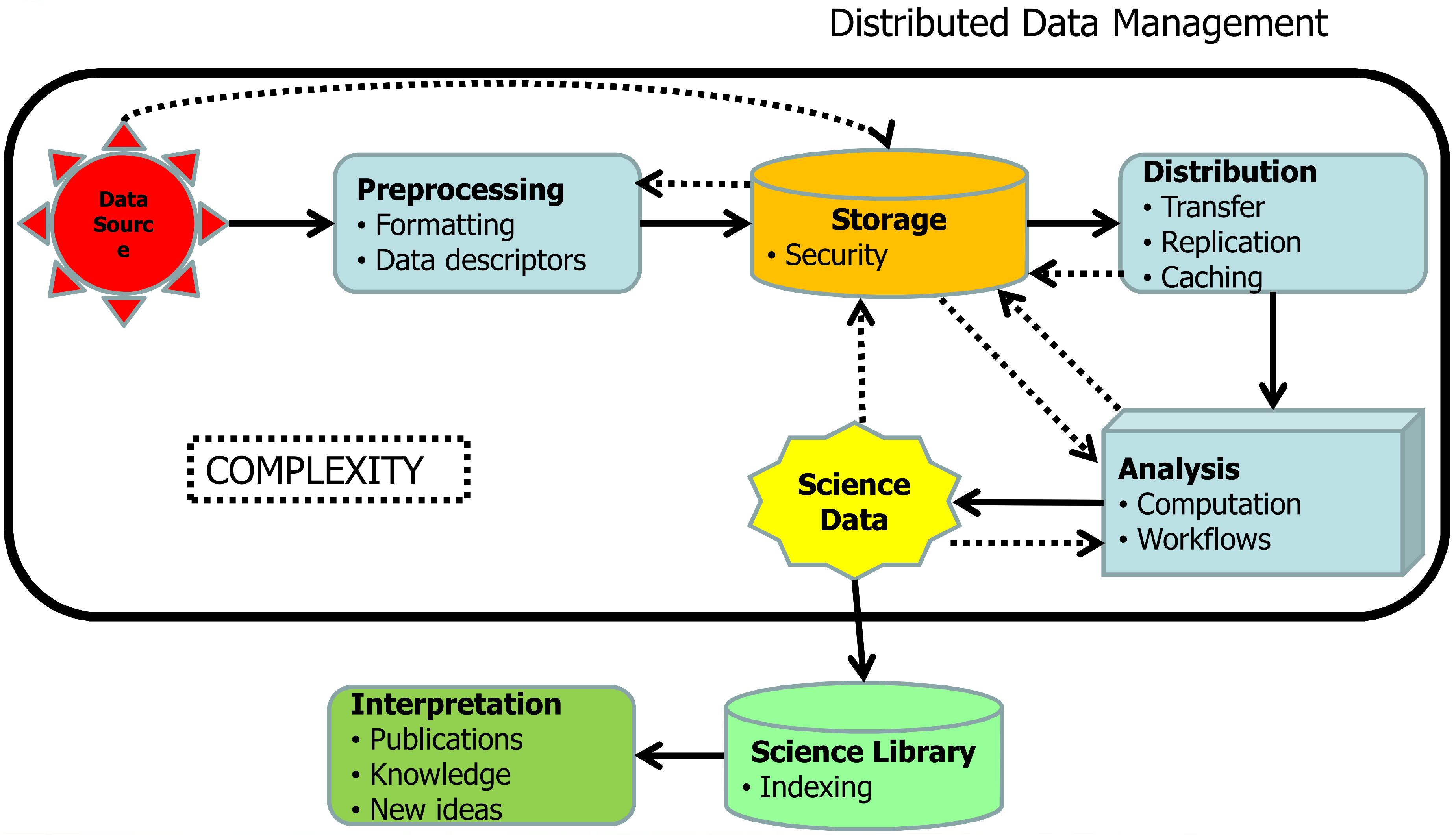
- Data is produced at a **very high rate**
- Data is inherently **distributed**
- Data is produced in **large quantities**
- Data is **needed/shared** by many people
- Data has **complex** interrelations
- Data has many free **parameters**

A single person / computer alone cannot do all the work
Several Groups Collaborating in Data Analysis

High-Level Data Processing Scenario



High-Level Data Processing Scenario



Principles of Distributed Data Management

- Data and computation co-scheduling
- Streaming
- Caching
- Replication

Co-Scheduling: Moving computation to the data

- Desirable for very large input data sets
- Conscious manual data location based on application access patterns

- Beware: Automatic data placement is domain specific!

Complexities

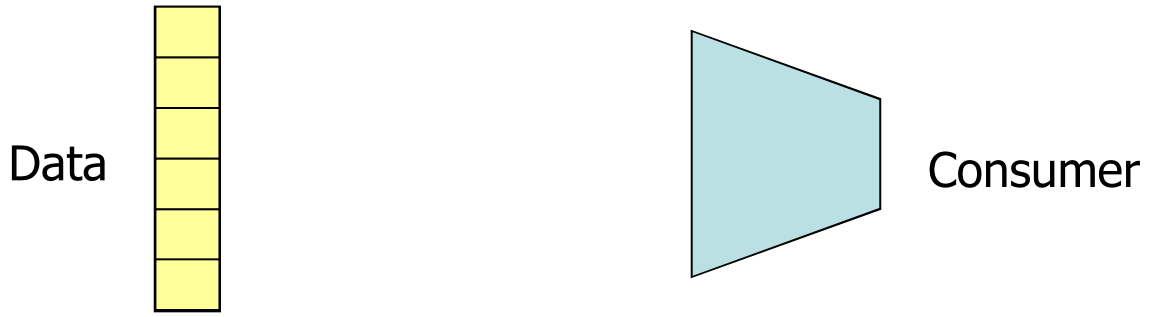
- It is a good idea to keep the large amounts of data locating in the computation
- Some data cannot be distributed
- Metadata stores are usually central

Combination of all of the above

Accessing Remote Data: Streaming

Streaming data across the wide area

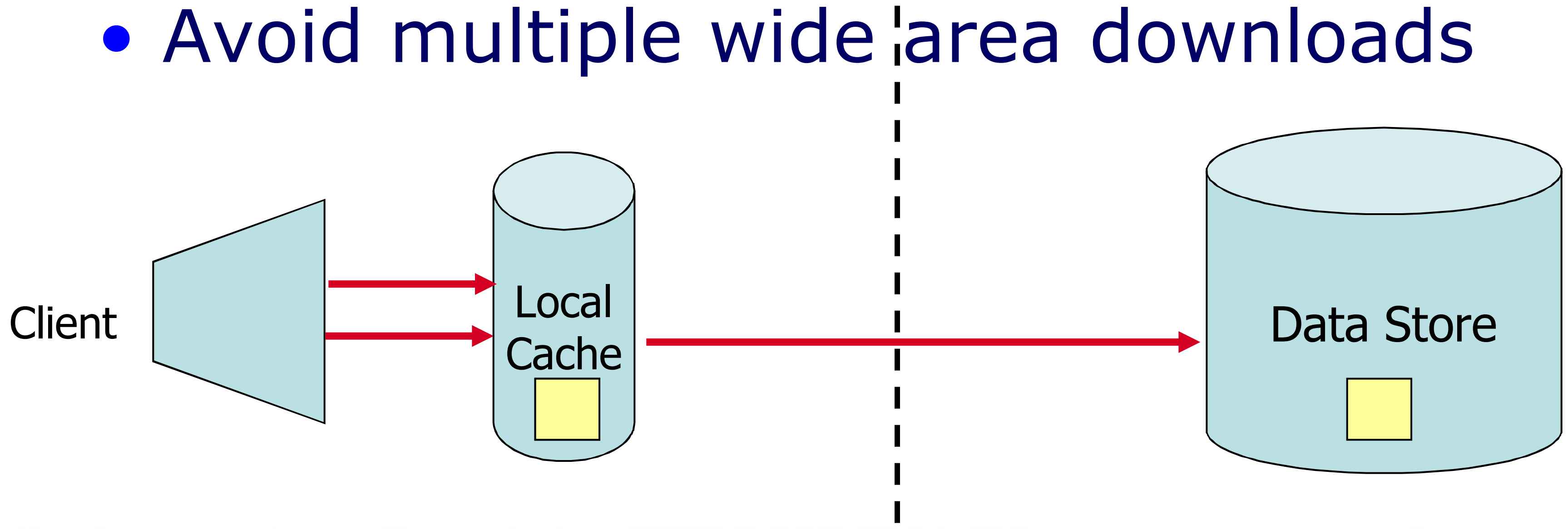
- Avoid intermediary storage issues
- Processing data as it comes
- Allow multiple consumers and producers
- Allow for computational steering and visualization



Accessing Remote Data: Caching

Caching data in local data caches

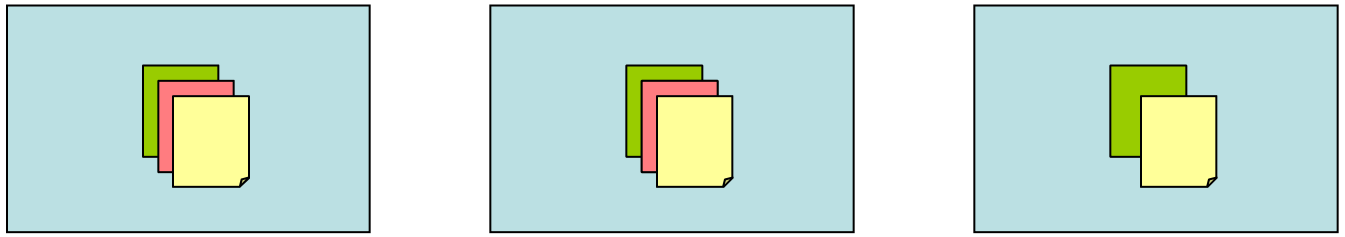
- Improve access rate for repeated access
- Avoid multiple wide area downloads



Distributing Data: Replication

Data is **replicated** across many sites in a Grid

- Keeping Data close to Computation
- Improving throughput and efficiency
- Reduce latencies

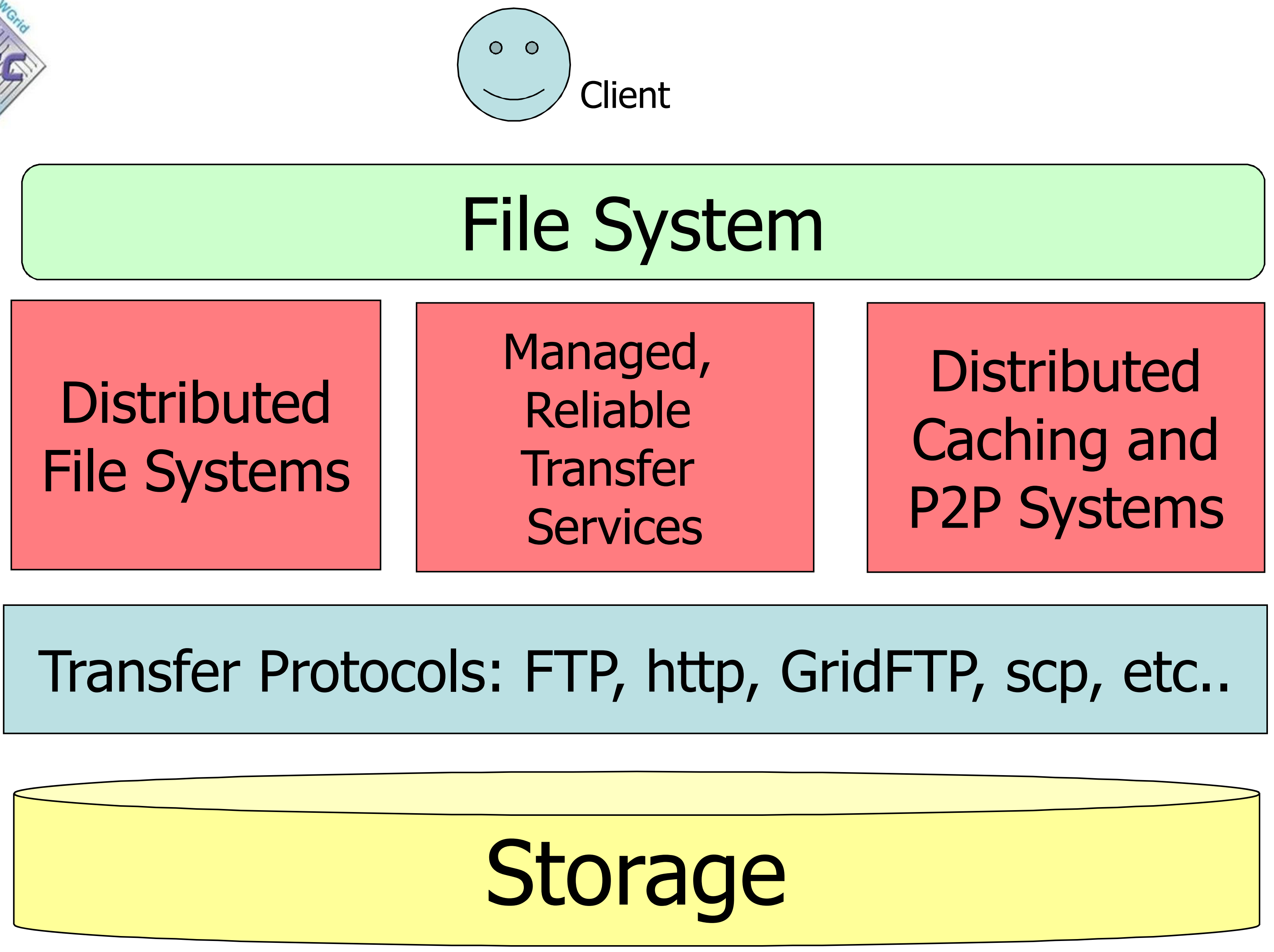


File Transfer

- Most Grid projects use GridFTP to transfer data over the wide area
- Managed transfer services on top:
  - Reliable GridFTP
  - gLite File Transfer Service
  - CERN CMS experiment’s Phedex service
  - SRM copy
- Management achieved by
  - Transfer Queues
  - Retry on failure
- Other Transfer Mechanisms (and example services):
  - http(s) (slashgrid, SRM)
  - UDP (SECTOR)
  - scp (UNICORE)
  - ..

Putting it to Practice

- Trust
- Distributed file management
  - Distributed Cluster File Systems
  - The Storage Resource Manager interface
    - dCache, SRB, NeST, SECTOR
  - Clouds File System
    - HDFS
- Distributed database management


Client
File System
Distributed File Systems
Managed, Reliable Transfer Services
Distributed Caching and P2P Systems
Transfer Protocols: FTP, http, GridFTP, scp, etc..
Storage

Trust

Trust goes both ways

- Site policies:
  - Trace what users accesses what data
  - Trace who belongs to what group
  - Trace where requests for access come from
  - Ability to block and ban users
- VO policies:
  - Store sensitive data in encrypted format
  - Managing user and group mappings at VO level

File Data Management

- Distributed Cluster File Systems
  - Andrew File System AFS, Distributed GPFS, Lustre
- Storage Resource Manager SRM interface to File Storage
  - Several implementations exist: dCache, BeStMan, CASTOR, DPM, StoRM, Jasmine, Storage Resource Broker SRB, Condor NeST..
- Other File Storage Systems
  - iRODS, SECTOR, .. (many many more)

Managed Storage Systems

- **Basics**
  - Stores data in the order of Petabytes
  - Total-throughput scales with the size of the installation
  - Supports several hundreds to thousands of clients
  - Adding / removing storage nodes w/o system interruption
  - Supports posix-like access protocols
  - Supports wide area data transfer protocols
- **Advanced**
  - Supports quotas or space reservation, data lifetime
  - Drives back-end tape systems (generates tape copies, retrieves non cached files)
  - Supports various storage semantics (temporary, permanent, *durable)*
  - System improves access speed by replicating 'hot spot‘ datasets, internal caching techniques, etc

Grid Application Platform

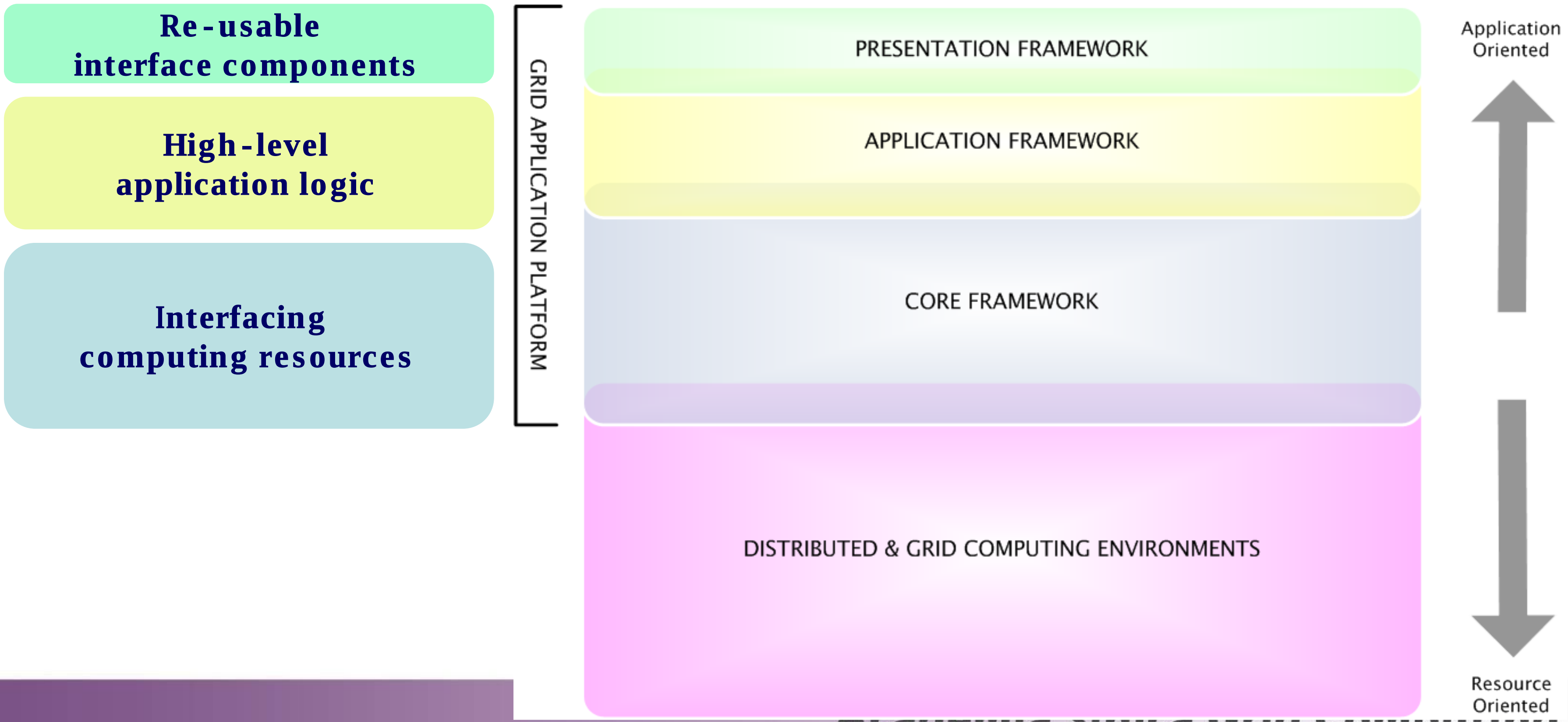
- Grid Application Platform (GAP) is a grid application framework developed by ASGC. It provides a vertical integration for developers and end-users
  - In our aspects, GAP should be
    - Easy to use for both end-users and developers.
    - Easy to extend for adopting new IT technologies, the adoption should be transparent to developers and users.
    - Light-weight in terms of the deployment effort and the system overhead.

The layered GAP architecture

Reduce the effort of developing application services

Reduce the effort of adapting new technologies

Concentrate efforts on applications


Re-usable interface components
High-level application logic
Interfacing computing resources
GRID APPLICATION PLATFORM
PRESENTATION FRAMEWORK
APPLICATION FRAMEWORK
CORE FRAMEWORK
DISTRIBUTED & GRID COMPUTING ENVIRONMENTS
Application Oriented
Resource Oriented

Advantages of GAP

- Through GAP, you can be a
  - Developer
    - Reduce the effort of developing application services.
    - Reduce the effort of adopting new distributed computing technologies.
    - Concentrate efforts on implementing application in their domain.
    - Client can be developed by any Java-based technologies.
  - End-user
    - Portable and light-weight client.
    - User can run their grid-enabled application as simple as using a desktop utility.

Features

- Application-oriented approach focuses developers effort on domain-specific implementations.
- Layered and modularized architecture reduces the effort of adopting new technology.
- Object-oriented (OO) design prevents repeating tedious but common works inbuilding application services.
- Service-oriented architecture (SOA) makes the whole system scalable.
- Portable thin client gives the possibility to access the grid from end-users desktop.

The GAP (V3.1.0)

- Can’s
  - simplify User and Job management as well as the access to the Utility Applications with a set of well-defined APIs
  - interface different computing environments with customizable plug-ins
- Cannot’s
  - simplify Data management

Why?

- Distributed data management is a hard problem
- There is no one-size-fits-all solution (otherwise Condor/Globus/gLite/grid would´ve done it!)
- Solutions exist for most individual problems (learn from RDBMS or P2P community)
- Integrating everything into an end-to-end solution for a specific domain is hard and ongoing work
- Many open problems!!

The GAP Data Manager Framework Objective

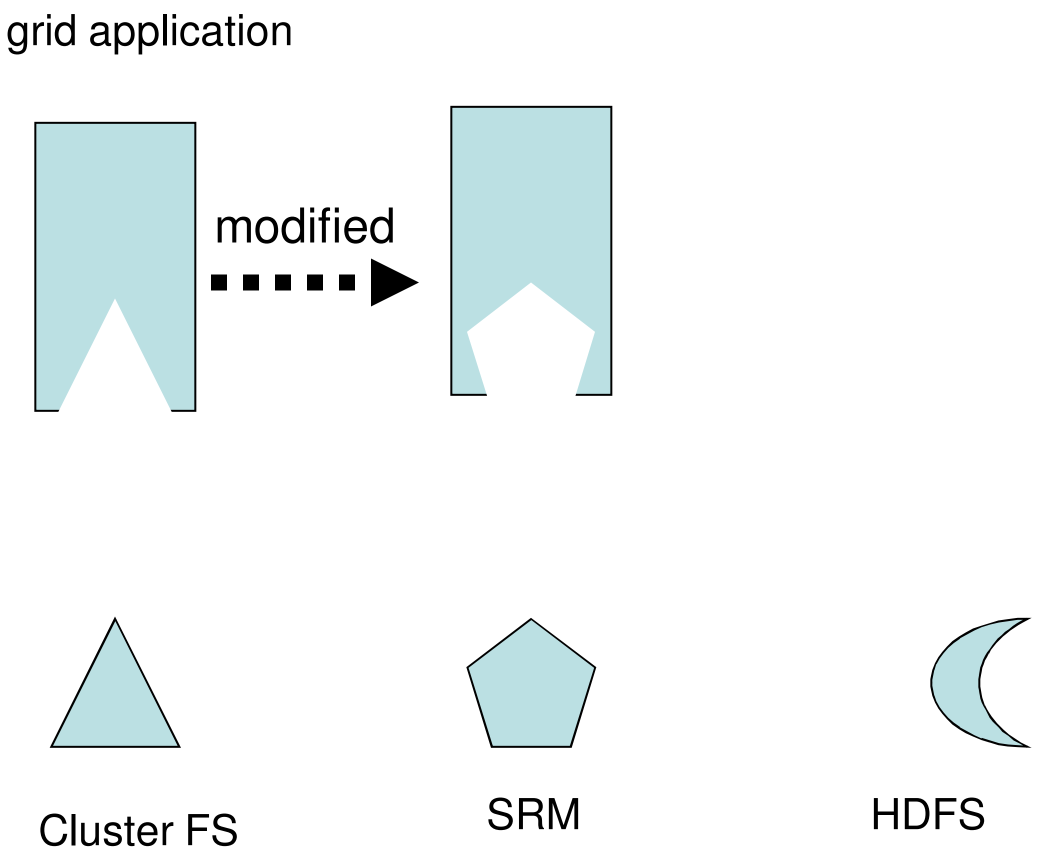
- Integrate different storage resources.
  - Cluster File System.
  - gLite / SRM / Storage Element.
  - Hadoop File System.
- Integrate different database resources
  - RDMS
  - HBase
- Hope to meet
  - Different user requirements

Data Manager Framework Development

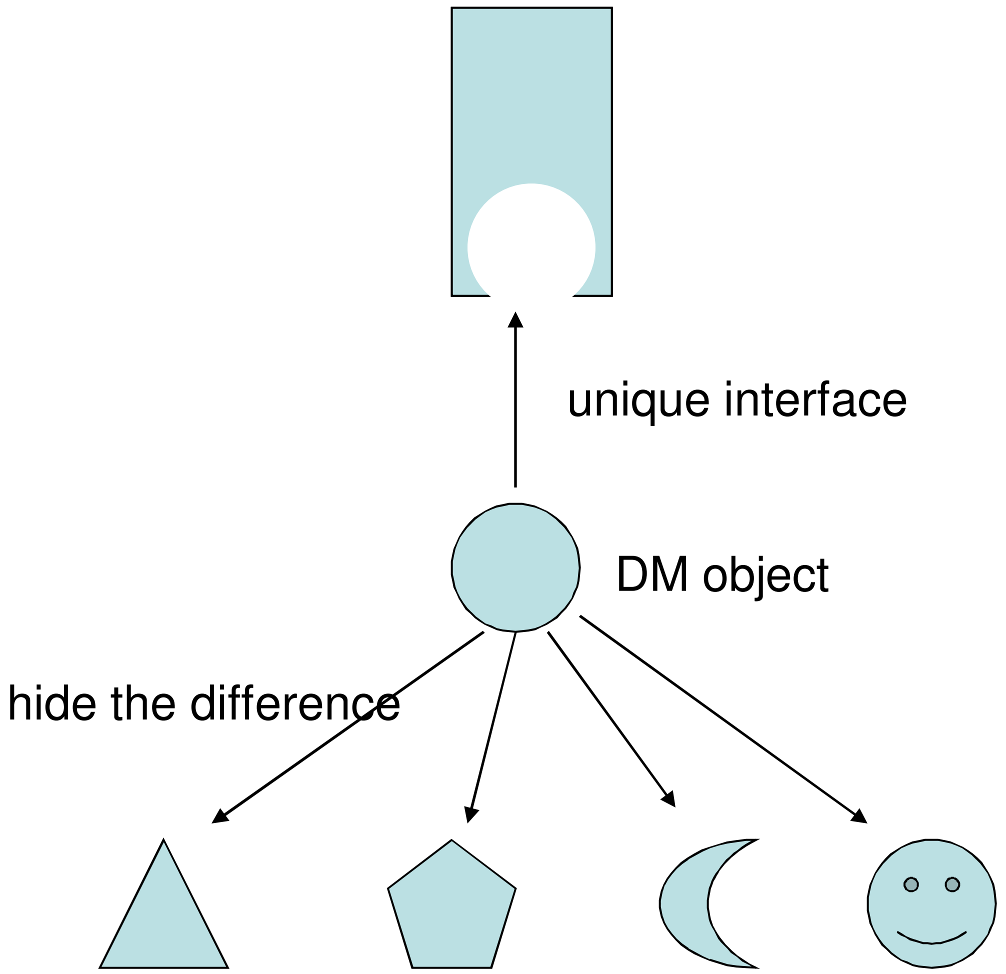
- Data Manager Framework development consists of
  - Interfacing underlying difference storage and database resources
  - Implementing Data Management logics
  - Designing Well-Define interfaces

Many efforts can be reused to speedup the development

How do I benefit from Data Manager Framework?



How do I benefit from Data Manager Framework?

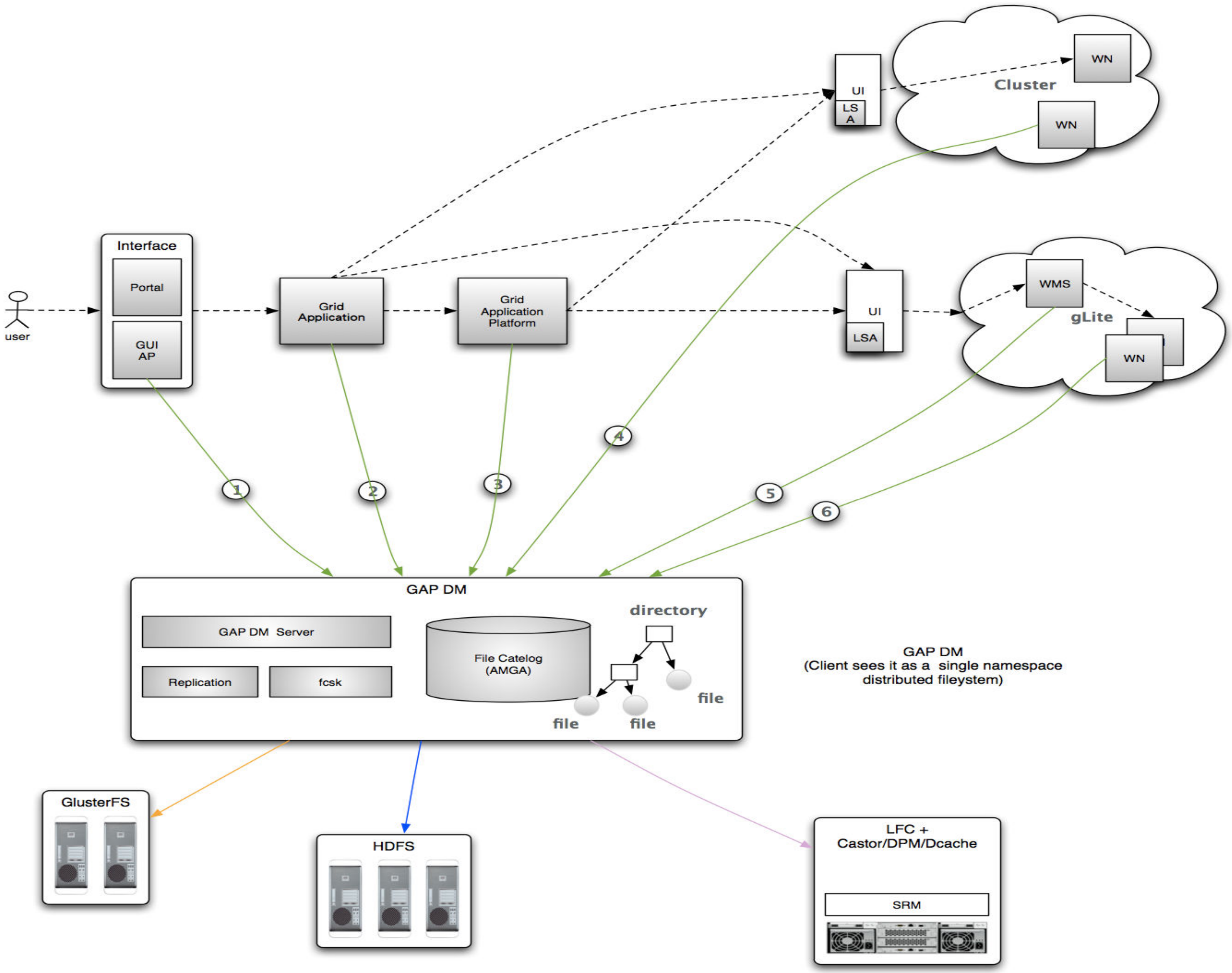


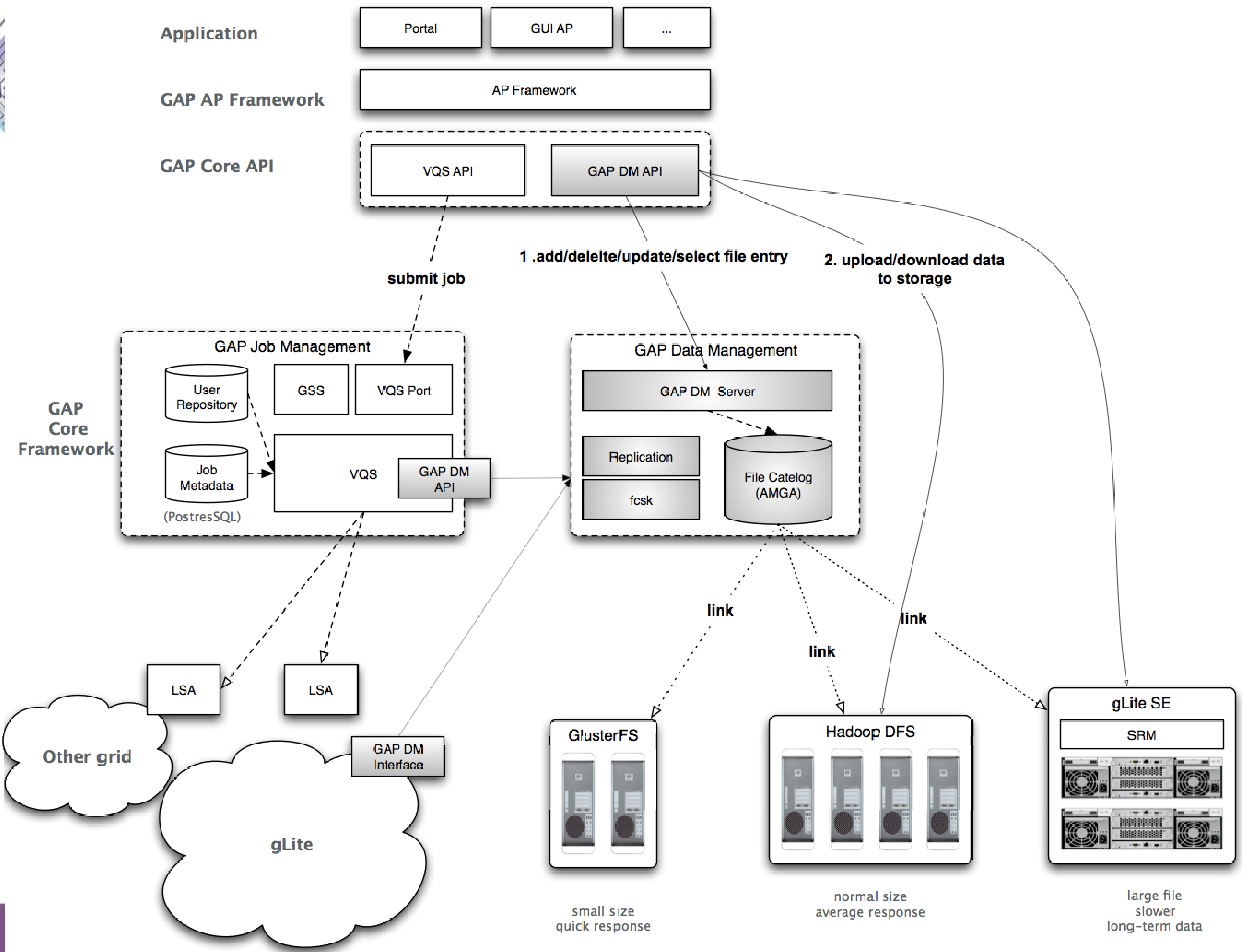
GAP Data Manager Design Goal

- Single namespace
- Single interface to difference DM solutions
- Support variety of storage types
  - Grids and Clouds
- Support non-structure and structure data
- Job management integration
- Authentication and Authorization
- Replication

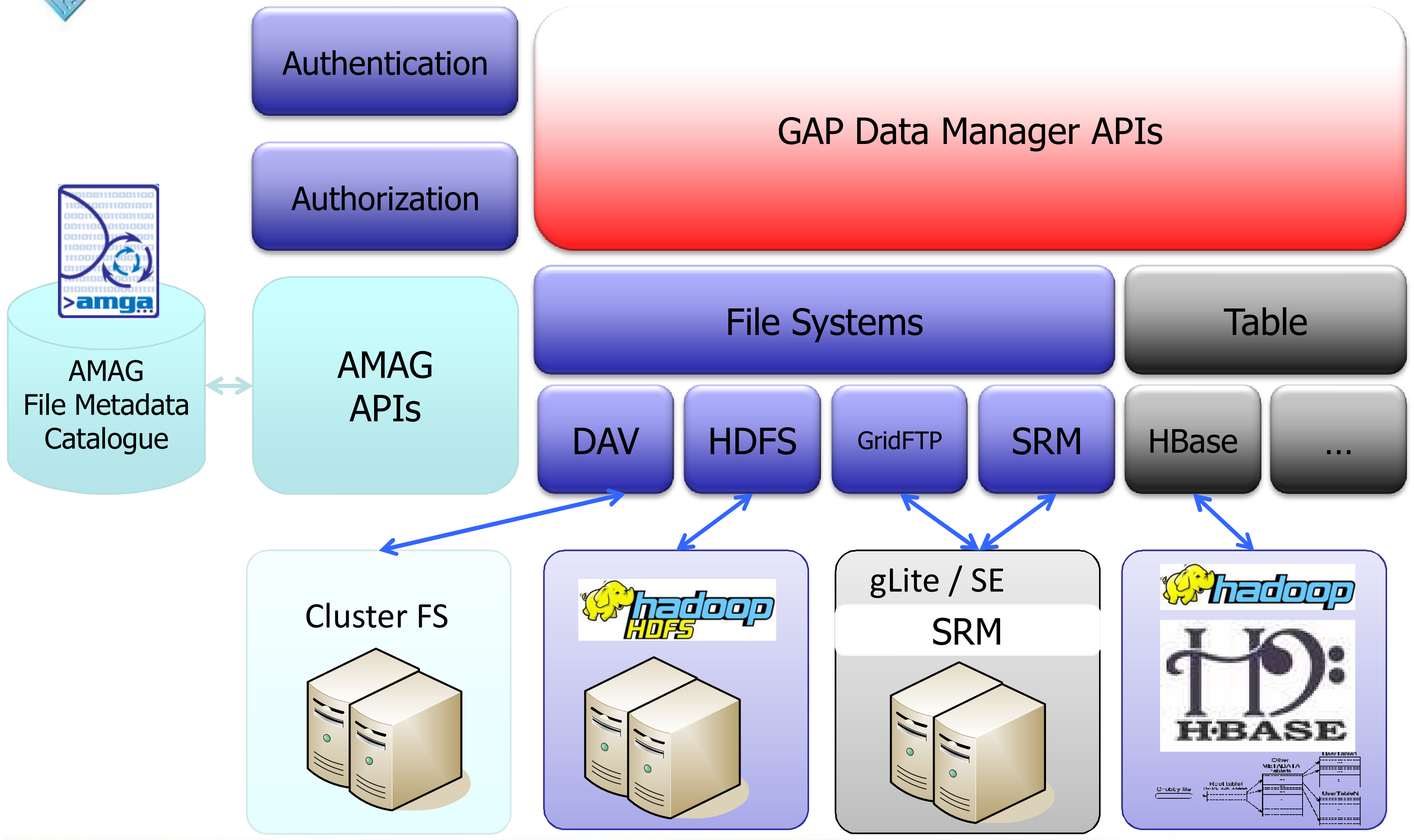
Hadoop File System (HDFS)

- HDFS is designed to store large files across the machines in a large cluster
- Highly fault-tolerant
- High throughput
- Suitable for applications with large data sets
- Streaming access to file system data
- Can be built out of commodity hardware

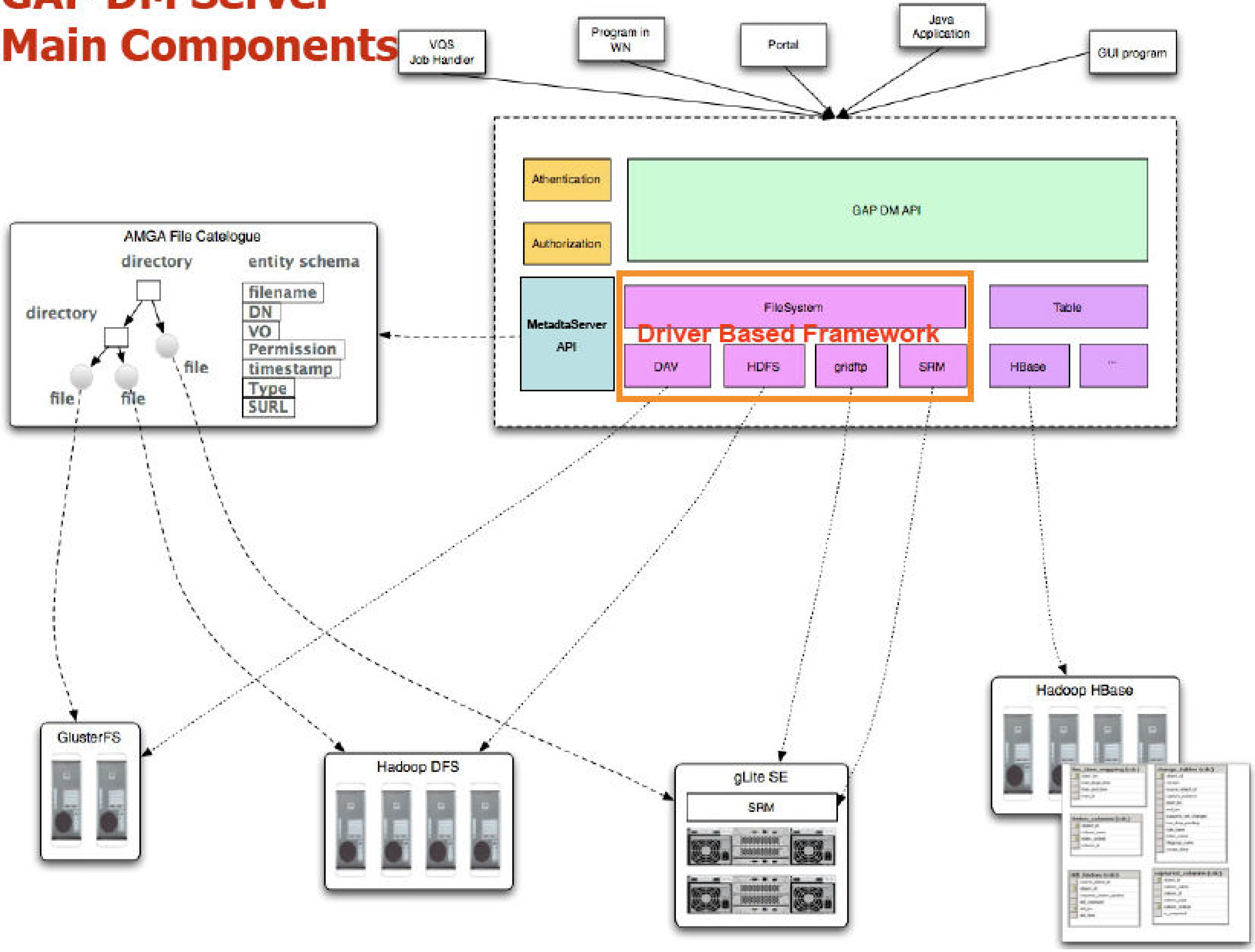

user
Interface
Portal
GUI AP
Grid Application
Grid Application Platform
UI
LSA
Cluster
WN
WN
UI
LSA
WMS
gLite
WN
1
2
3
4
5
6
GAP DM
GAP DM Server
Replication
fcsk
File Catelog (AMGA)
directory
file
file
file
GAP DM
(Client sees it as a single namespace distributed fileystem)
GlusterFS
HDFS
LFC + Castor/DPM/Dcache
SRM


Application
Portal
GUI AP
...
GAP AP Framework
AP Framework
GAP Core API
VQS API
GAP DM API
submit job
1 .add/delelte/update/select file entry
2. upload/download data to storage
GAP Core Framework
GAP Job Management
User Repository
GSS
VQS Port
Job Metadata
VQS
GAP DM API
(PostresSQL)
GAP Data Management
GAP DM Server
Replication
fcsk
File Catelog (AMGA)
link
link
link
LSA
LSA
Other grid
GAP DM Interface
gLite
GlusterFS
Hadoop DFS
gLite SE
SRM
small size quick response
normal size average response
large file slower long-term data

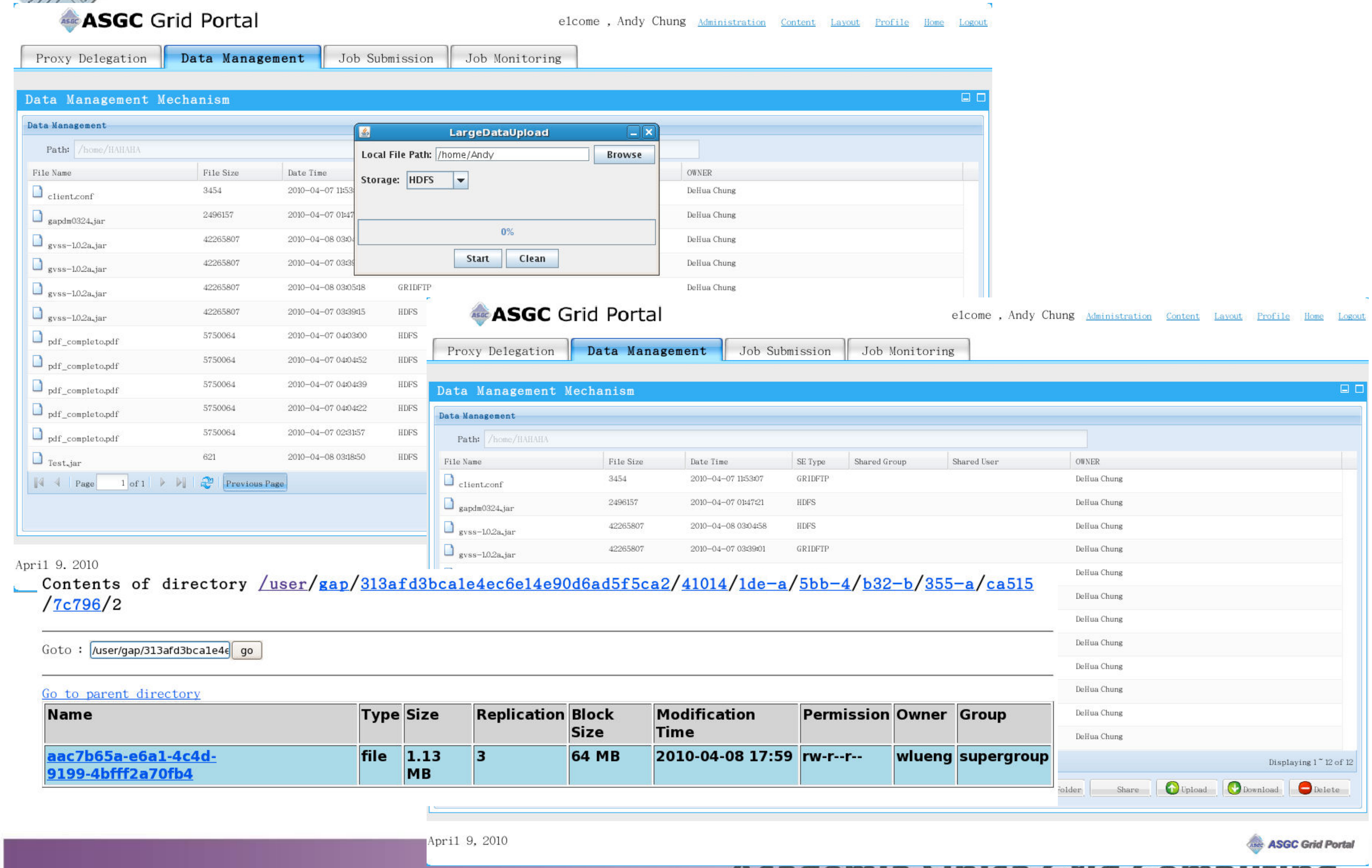
GAP Data Manager Archiecture



GAP DM Server Main Components



Demo



ASGC Grid Portal

elcome , Andy Chung Administration Content Layout Profile Home Logout

Proxy Delegation | Data Management | Job Submission | Job Monitoring

Data Management Mechanism

Data Management

Path: /home/HAHAHA

File Name	File Size	Date Time		OWNER
client.conf	3454	2010-04-07 11:53		DeHua Chung
gapdm0324.jar	2496157	2010-04-07 01:47		DeHua Chung
gvss-1.0.2a.jar	42265807	2010-04-08 03:04		DeHua Chung
gvss-1.0.2a.jar	42265807	2010-04-07 03:39		DeHua Chung
gvss-1.0.2a.jar	42265807	2010-04-08 03:05:18	GRIDFTP	DeHua Chung
gvss-1.0.2a.jar	42265807	2010-04-07 03:39:15	HDFS	
pdf_completo.pdf	5750064	2010-04-07 04:03:00	HDFS	
pdf_completo.pdf	5750064	2010-04-07 04:04:52	HDFS	
pdf_completo.pdf	5750064	2010-04-07 04:04:39	HDFS	
pdf_completo.pdf	5750064	2010-04-07 04:04:22	HDFS	
pdf_completo.pdf	5750064	2010-04-07 02:31:57	HDFS	
Test.jar	621	2010-04-08 03:18:50	HDFS	

Page 1 of 1 Previous Page

LargeDataUpload

Local File Path: /home/Andy Browse

Storage: HDFS

0%

Start Clean

April 9, 2010

ASGC Grid Portal

elcome , Andy Chung Administration Content Layout Profile Home Logout

Proxy Delegation | Data Management | Job Submission | Job Monitoring

Data Management Mechanism

Data Management

Path: /home/HAHAHA

File Name	File Size	Date Time	SE Type	Shared Group	Shared User	OWNER
client.conf	3454	2010-04-07 11:53:07	GRIDFTP			DeHua Chung
gapdm0324.jar	2496157	2010-04-07 01:47:21	HDFS			DeHua Chung
gvss-1.0.2a.jar	42265807	2010-04-08 03:04:58	HDFS			DeHua Chung
gvss-1.0.2a.jar	42265807	2010-04-07 03:39:01	GRIDFTP			DeHua Chung
						DeHua Chung
						DeHua Chung
						DeHua Chung
						DeHua Chung
						DeHua Chung
						DeHua Chung
						DeHua Chung
						DeHua Chung

Displaying 1 ~ 12 of 12

older Share Upload Download Delete

April 9, 2010

ASGC Grid Portal

Contents of directory /user/gap/313afd3bca1e4ec6e14e90d6ad5f5ca2/41014/1de-a/5bb-4/b32-b/355-a/ca515/7c796/2

Goto : /user/gap/313afd3bca1e4e go

Go to parent directory

Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
aac7b65a-e6a1-4c4d-9199-4bfff2a70fb4	file	1.13 MB	3	64 MB	2010-04-08 17:59	rw-r--r--	wlueng	supergroup

Academia Sinica Grid Computing

Summary

- Integrate different storage resources on GAP to provide more options of heterogeneous data management mechanisms.
- This work also demonstrated lots of viable alternatives to Grid Storage Element, especially in terms of scalability, reliability, and manageability.
- Enhances the capability of parallel processing and also versatile data management approaches for Grid.
- GAP could be a bridge between Cloud and Grid infrastructure and more computing framework from Cloud would be integrated in the future.

Thank you for your attention and great inputs!

Backup Slides

Storage Resource Manager Interface

- SRM is an **OGF interface standard**
- One of the few interfaces where **several** implementations exist (>5)

Main features

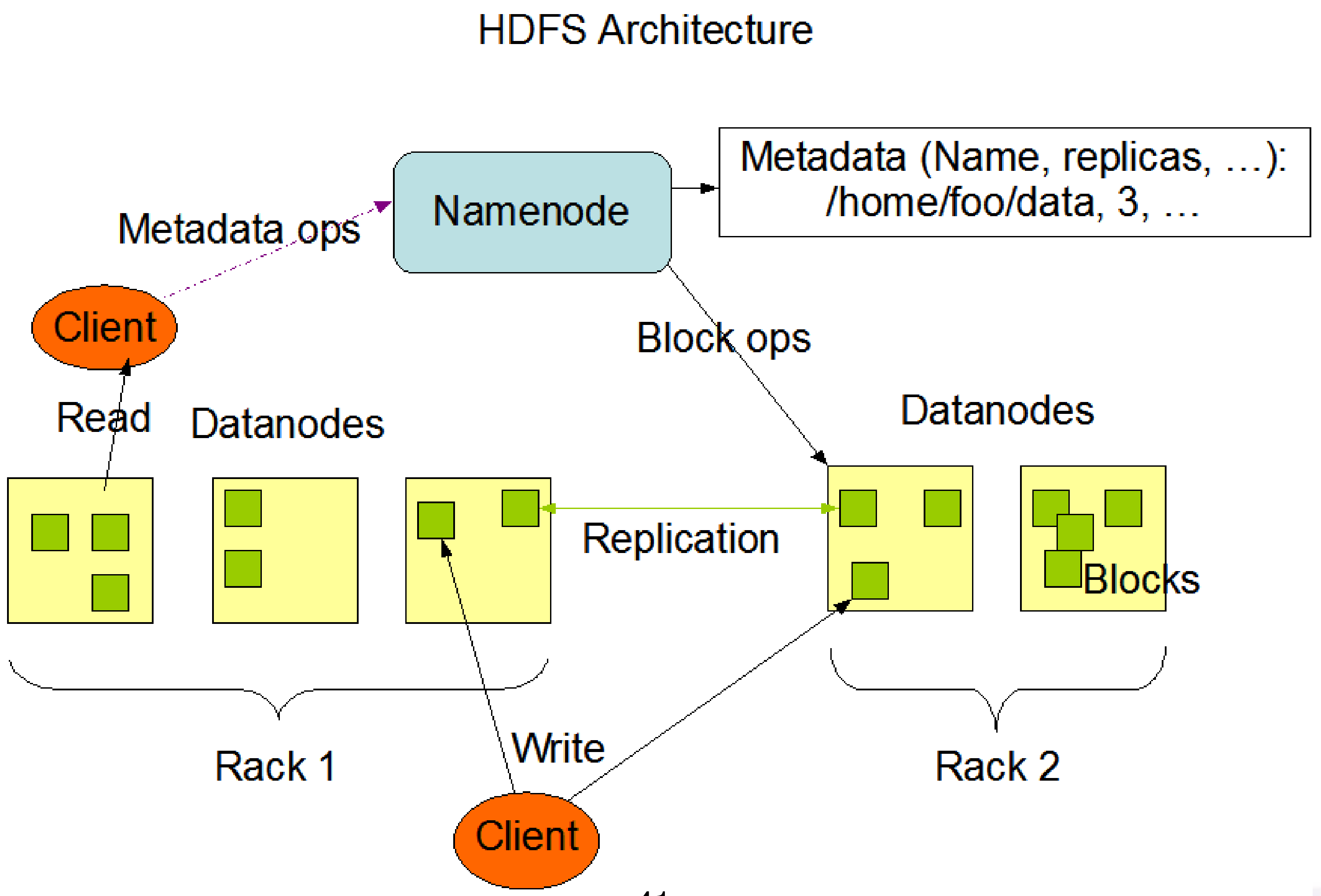
- **Prepares for data transfer** (not transfer itself)
  - Transparent management of hierarchical storage backends
  - Make sure data is accessible when needed: Initiate restore from *nearline* storage (tape) to *online* storage (disk)
- Transfer between SRMs as managed transfer (**SRM copy**)
- **Space reservation** functionality (implicit and explicit via space tokens)

Storage Resource Manager Interface

SRM v2.2 interface supports

- **Asynchronous** interaction
- **Temporary, permanent and durable** file and space semantics
  - Temporary: no guarantees are taken for the data (scratch space or /tmp)
  - Permanent: strong guarantees are taken for the data (tape backup, several copies)
  - Durable: guarantee until used: permanent for a limited time
- **Directory functions** including file listings.
- **Negotiation of** the actual data **transfer protocol**.

HDFS Architecture



File system Namespace

- Hierarchical file system with directories and files
- Create, remove, move, rename etc.
- Namenode maintains the file system
- Any meta information changes to the file system recorded by the Namenode.
- An application can specify the number of replicas of the file needed: replication factor of the file. This information is stored in the Namenode.

Data Replication

- HDFS is designed to store very large files across machines in a large cluster.
- Each file is a sequence of blocks.
- All blocks in the file except the last are of the same size.
- Blocks are replicated for fault tolerance.
- Block size and replicas are configurable per file.
- The Namenode receives a Heartbeat and a BlockReport from each DataNode in the cluster.
- BlockReport contains all the blocks on a Datanode.