

GSAF - Grid Storage Access Framework

S. Scifo, Consorzio COMETA - Catania, Italy (salvatore.scifo@ct.infn.it)
 G.Foti, Dipartimento di Fisica, Università di Catania, Italy (gaetano.foti@ct.infn.it)
 F. Portuese, IR&T engineering s.r.l. - Catania, Italy (f.portuese@irt-engineering.it)
 S.Parisi, IR&T engineering s.r.l. - Catania, Italy (s.parisi@irt-engineering.it)
 R. Barbera, Istituto Nazionale Fisica Nucleare - Catania, Italy (roberto.barbera@ct.infn.it)

GSAF (Grid Storage Access Framework) covers some aspects of the Data Access Management area, looking at the integration and development of data oriented applications interfaced with the EGEE gLite grid Middleware. It is an Object Oriented Framework (based on the black box model) developed to create Data Grid Application such as digital archives. It allows developers to extend applications with middleware data management functionalities. Moreover, GSAF introduces the Transaction Layer capability at client side permitting operation sequence in atomic mode on the same dataset. There are already several use cases that adopt GSAF: DM Web Manager, BM Portal, GRIDICOM and ADAT are some of them and they are treated in the present work. GSAF project is carried out by a cooperation between the Cometa Consortium and the IR&T engineering s.r.l. (SME located in Catania, Italy) in the context of the PI2S2 Project and the ADAT Project ("Archivi Digitali Antico Testò"), which aim at the implementation of a Digital Archive for Cultural Heritage Data (antique manuscripts) with the Digital Repository based on EGEE grid services.

Thinking of GRID as Digital Repository

Storage Virtualization

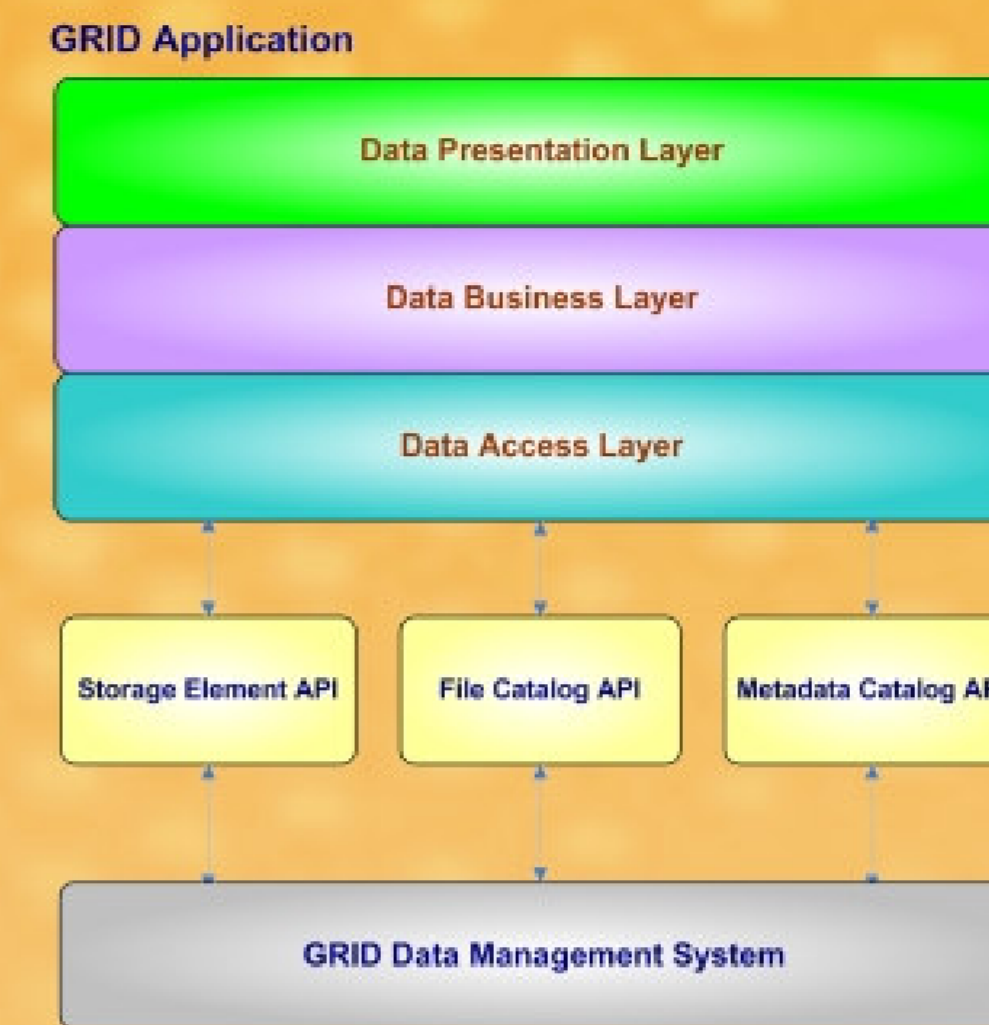
- Unique and uniform **interface to manage DATA** provided by the grid middleware
- Unique and uniform **interface to manage METADATA** provided by the grid middleware
- Large and numerous **file handling capability** also in a geographic distributed environment
- **Ubiquity**: data access independently by their location.

Security capabilities

- **Centralized** access control mechanism based on x.509 certificates and **user roles** according to **Virtual Organization policies** that users belong to.

Availability, Scalability, Fault Tolerance

Grid Enterprise Integration Model



Porting aspects

- Within the Grid environment files are stored inside a **Storage Element (SE)**
- Files can be replicated on several SEs for ubiquity, security and sharing needs; relationship among locations of files and replicas and their identifier are kept within a specific **File Catalogue**
- For each file is possible to associate descriptive metadata arranged into a specific **Metadata Catalogue**
- There are not tools neither services that help clients to maintain **semantically coherence** and **integrity consistency** among files stored within the SE and their entries inside the File Catalog neither inside the Metadata Catalog

Technical

Developing applications for Grid means substitute the traditional Data Access Layer with an appropriate interface that permits **business components** to manage data stored within the Grid Data Management Service and **presentation objects** to search and retrieve data from them.

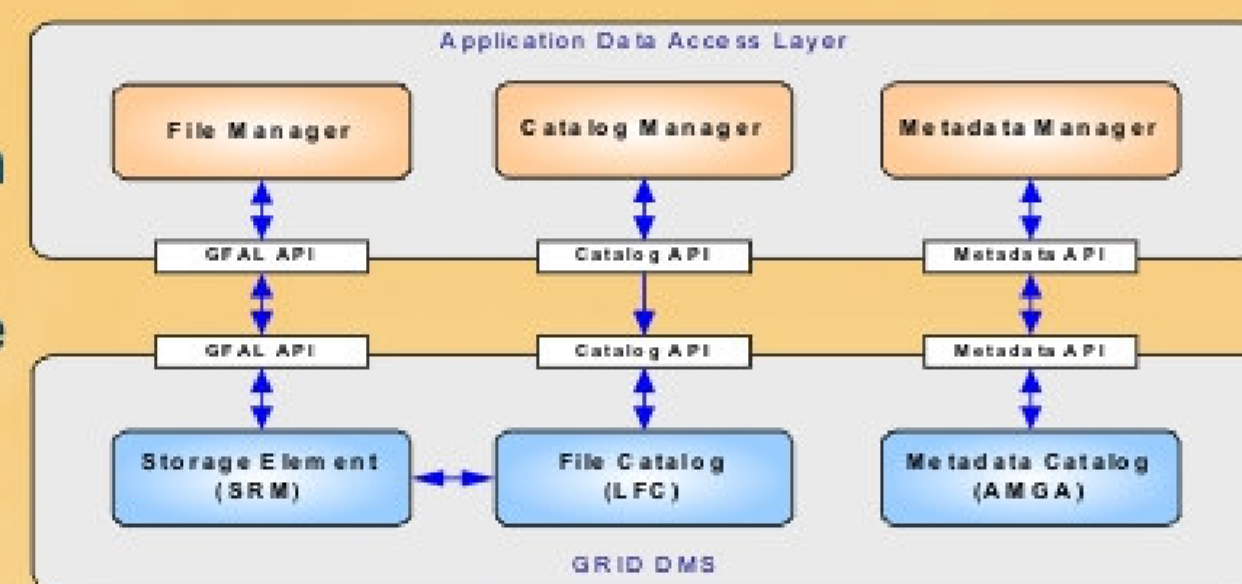
Designing tragedy ☹

Development troubles

- Grid Data Services are **independent** each from others (SOA)
- They work in a **"stand alone"** mode (API fragmentation)

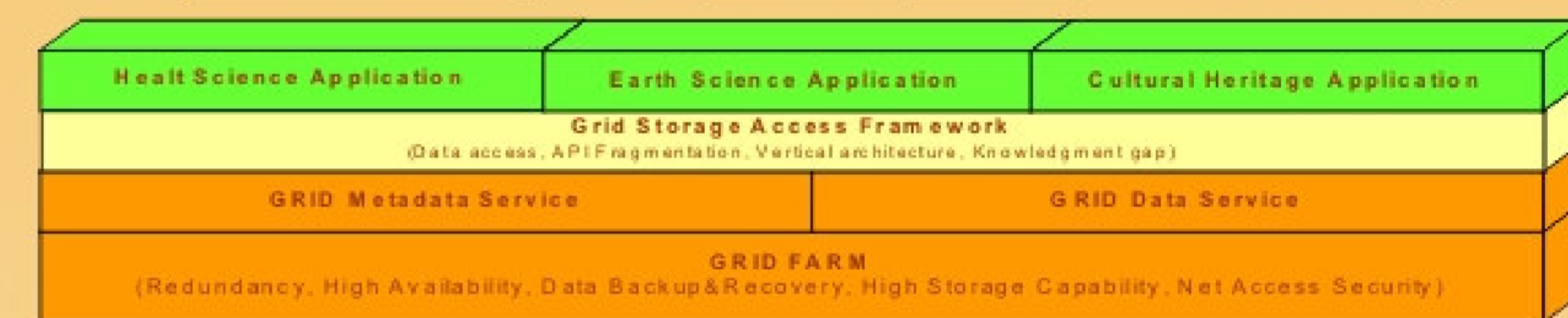
Fragmentation effects

- **No coherence** is ensured (uncontrollable resources)
- **software engineers** must consider a **vertical architecture** using specialized software components
- **applications** must take care themselves (at Business Logic Level), about the **atomicity, coherence** and **synchronization** of data manipulation (development effort)
- **knowledge Gap** (traditional development Vs grid development)
- **code replication** for different use cases



GSAF as Design Paradigm

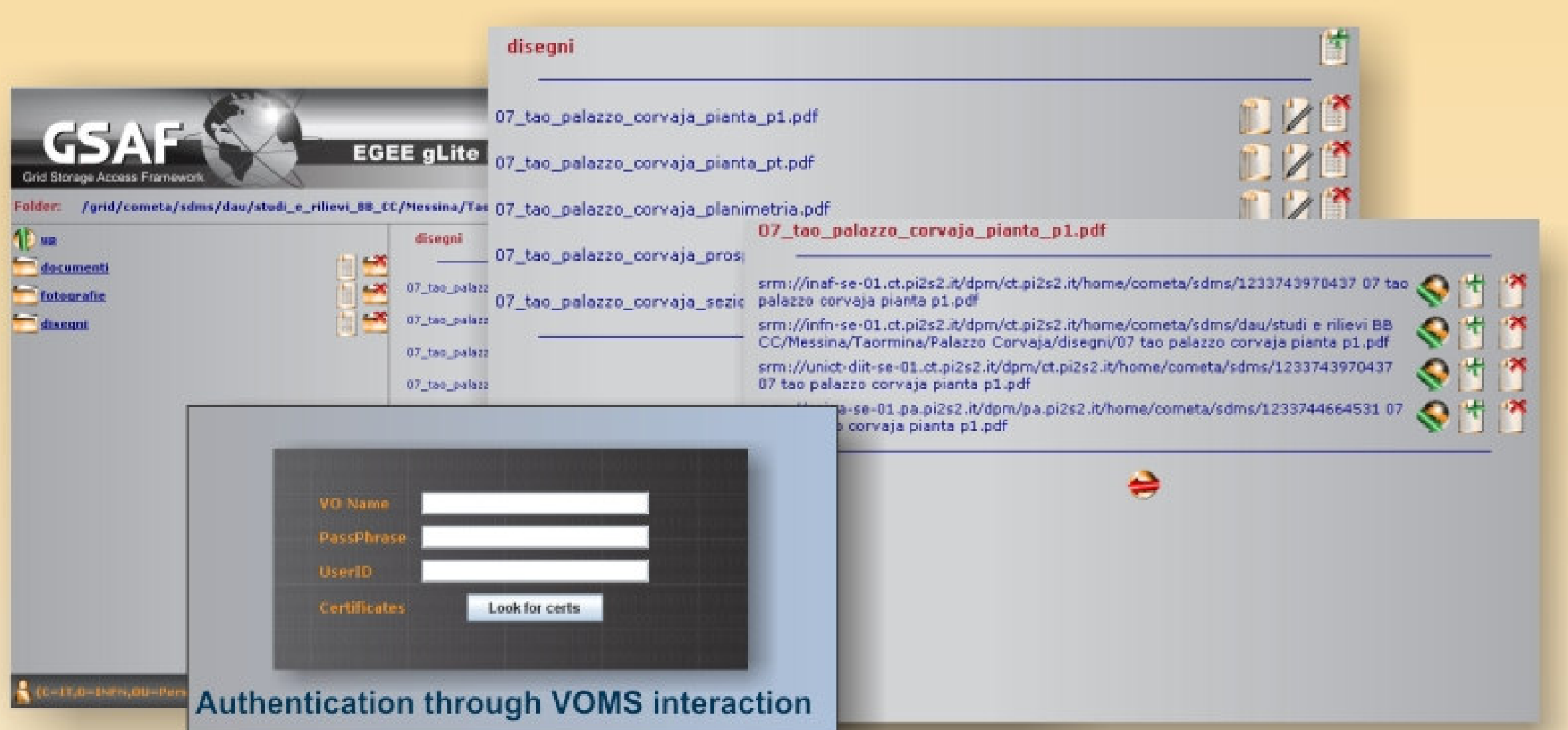
We have common features, we have common problems → we need a **Design Pattern**



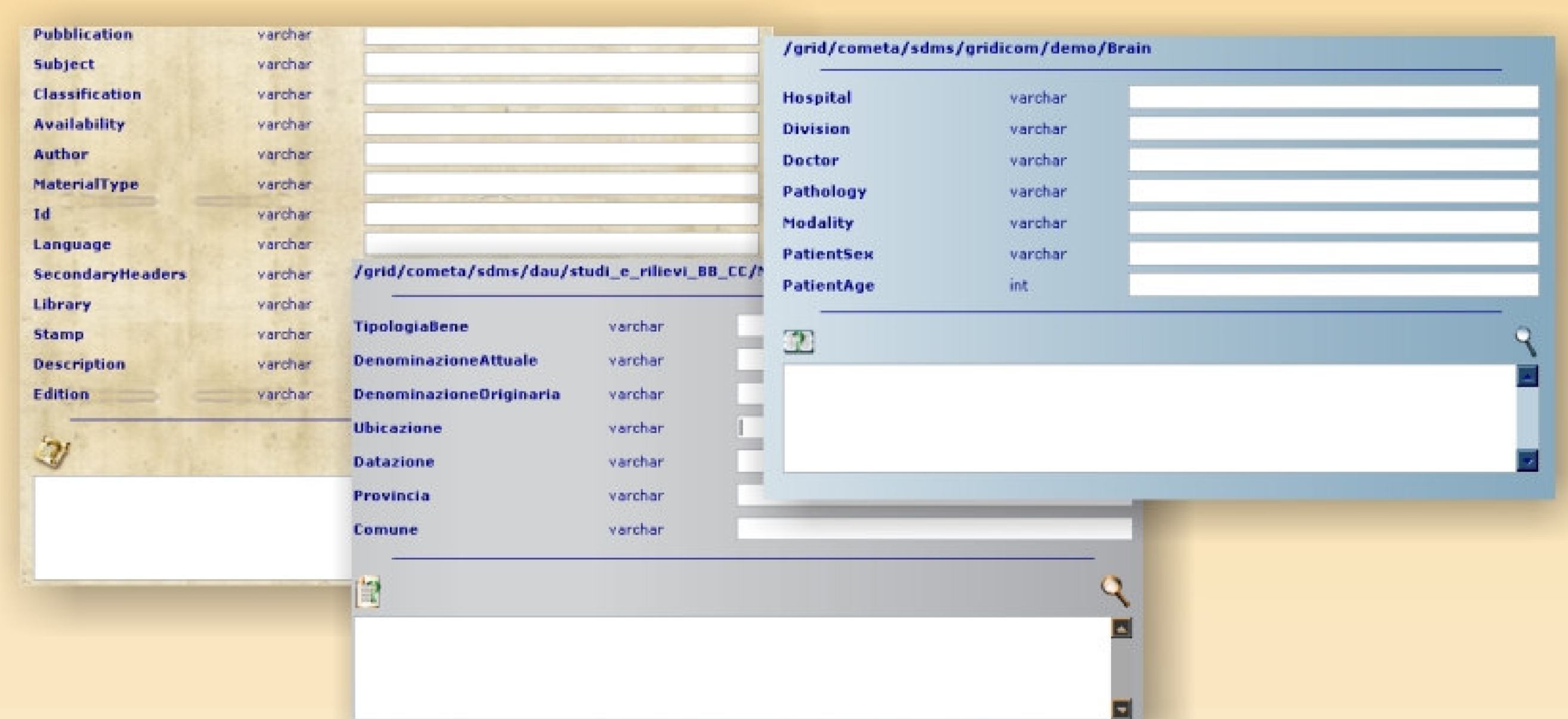
Built on top of the Grid Metadata Service and Grid Data Service

- collects and implements functionalities shared among applications according to **"write once use anywhere"** principle
- reduces the knowledge gap
 - hiding the **complexity** and the **fragmentation** of the several underlying APIs
 - exposing a unified interface more near to the **developer mind** (design patterns) rather than the **Grid stuff details** (API syntaxes)
- providing a transaction support to
 - perform several data manipulation in Atomic Mode
 - hold Runtime Exceptions to limit Data Inconsistence and Data Loss
- acts as a **black box** providing
 - **classes** and related **methods** for applications located above
 - **interfaces** to extend the implemented capabilities

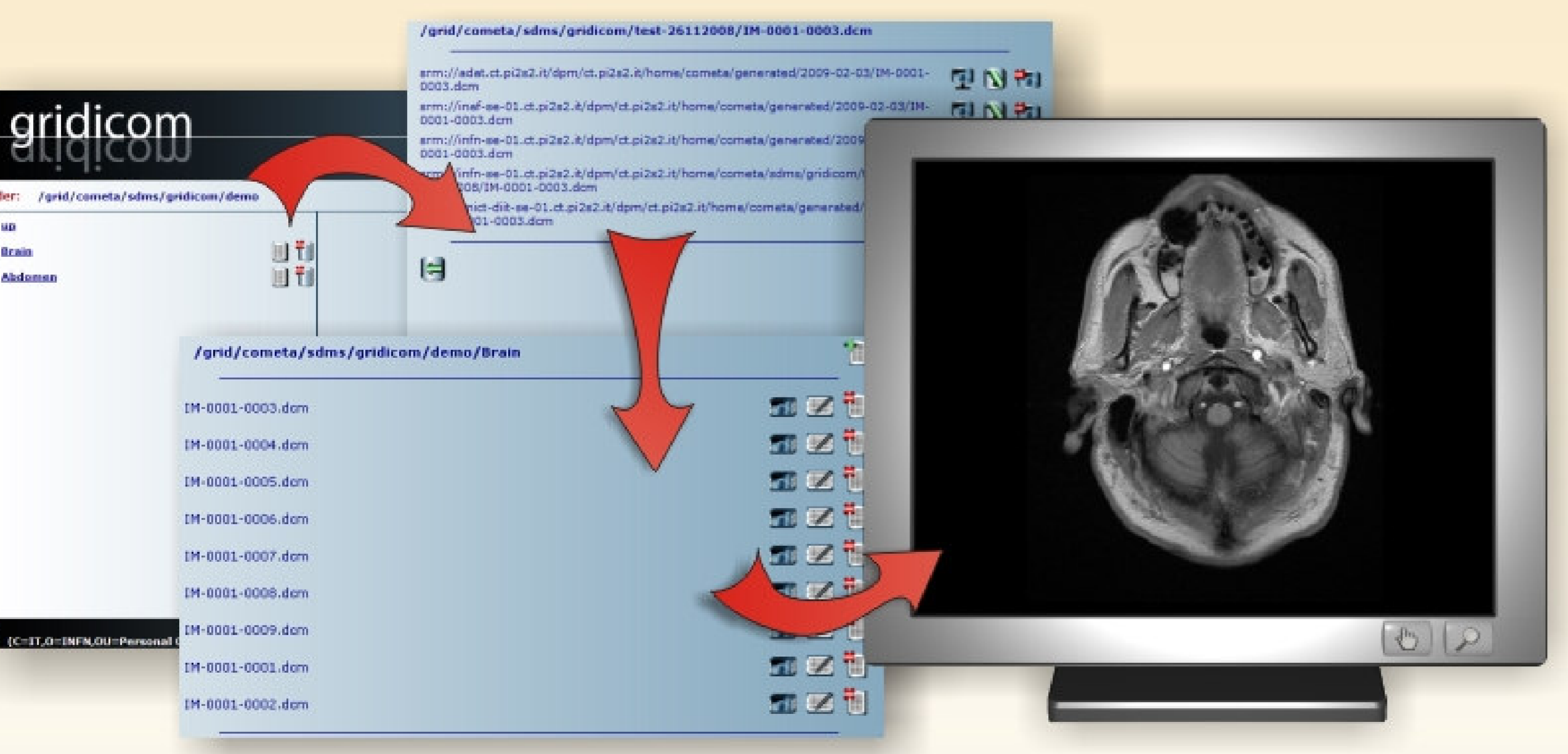
DM Web Manager - remote Data Grid management



Searching file by metadata



GRIDICOM - Grid Archive for DICOM images



Use Cases

