



IST-2001-32133

GridLab - A Grid Application Toolkit and Testbed

Technical Specifications

Author(s):	André Merzky, Florian Schintke and Thorsten Schütt
Document Filename:	GridLab-08-DATA-0002-TechSpec
Work package:	Data Management and Visualization
Partner(s):	ZIB
Lead Partner:	ZIB
Config ID:	GridLab-08-DATA-0002-1.0
Document classification:	PUBLIC

Abstract: This document outlines the design of components developed by WP8 *Data Management and Visualization* of the GridLab project. These components cover services and libraries. The interfaces as defined here are to be checked against all other GridLab WP-X design documents.



Contents

1	Introduction	2
2	General Aspects	2
2.1	Generic Service Architecture	2
2.2	Coding Conventions	2
3	Replica Service	4
3.1	Description	4
3.2	Interface	4
4	Data Movement Service	6
4.1	Description	6
4.1.1	Interface	6
5	Streaming Service	6
5.1	Description	6
5.2	Interface	7
6	Remote File Access	7
6.1	Description	7
6.1.1	Interface	7
7	Viz Services	7
7.1	Description	7
7.2	Interface	8
8	Online Recombination and Data Extraction Service	8
8.1	Description	8
8.2	Interface	8
9	Dependencies of Components within WP8	8
10	WP8 Dependencies on other Work Packages	9
10.1	WP6 Security	9
10.2	WP7 Adaptive Components	9
10.3	WP9 Resource Management	9
10.4	WP10 Information Services	10
11	Mapping between services and deliverables	10
11.1	D8.3: Data discovery and tracking services	10
11.2	D8.4: Low level data access services	10
11.3	D8.5: High level data access services	10
11.4	D8.6: Data management integration	10
11.5	D8.8: (Parallel) dynamic data filters	10

1 Introduction

This document contains the technical specification for Work Package VIII (WP8) of the GridLab project – “*Data Management and Visualization*”. The document bases on the requirement specification document of WP8 [10], and on the General GridLab Architecture as specified by the GridLab technical board [4]. The text also uses the definitions defined there.

The goal of this document is twofold: (i) it describes the capability providers as designed by WP8 in order to provide the functionality identified in the requirement specification document, and (ii) it provides information how these capability providers map to the deliverables from the GridLab project proposal in Section 11. This will allow other GridLab workpackages to estimate *which* functionality they can expect *when* from WP8.

The technical specification document is structured as follows: the chapter 2 describes the general structure of services implemented by WP8. Chapters 3 to chapter 8 describe the individual capability providers (services and libraries) in detail, listing their individual interfaces. Chapter 11 gives information about how these capability providers will map to the WP8 deliverables.

2 General Aspects

2.1 Generic Service Architecture

Most services implemented by WP8 will be based on a common generic service framework (see Figure 1). This framework is no substitution for OGSA but it will give us the possibility to develop our services independent of OGSA or WSDL. During development we can invoke our services via a GUI or command line and for production use the invocation mechanism can be easily changed to OGSA or WSDL.

We will derive classes from CallWrapper which implement the interface to consumers of our services. These classes will be specialized for OGSA, WSDL etc. They will hide all details of the particular Services Framework from the actual implementation of our services.

The Call Wrapper forwards the parameters of a method invocation to an Action object which will handle the execution. The Action object queries the ServiceFinder for Services which implements this method. The purpose of this mechanism becomes clear when we examine the data movement service. Data movement can be implemented in various ways. We can use custom protocols, a local `cp`, `scp` or `grid_url_copy`. For data movement the ServiceFinder returns a proxy (Service) for each implemented movement method. In the next step the Action object calls a Selector object which selects the most ‘suitable’ service for the actual parameters. After that the Action object will forward the parameters to the selected Service object which implements the service or fulfills the role of a proxy to the actual implementation.

For some services we will only provide one implementation so the ServiceFinder and the Selector can be rather dumb. But sometimes as for the data movement service we have to carefully select the best implementation. The Selector must distinguish between implementations which can fulfill the task and implementations which fail, for example for security reasons. Other parameters which have to be taken into account by the Selector are scalability, availability and cost / performance ratio. The development of the individual Selectors will be tightly coupled with WP-7 (Adaptive Application Components).

2.2 Coding Conventions

The function prototypes presented in this document follow a common schema which is explained in more detail in the GridLab Coding Guidelines.

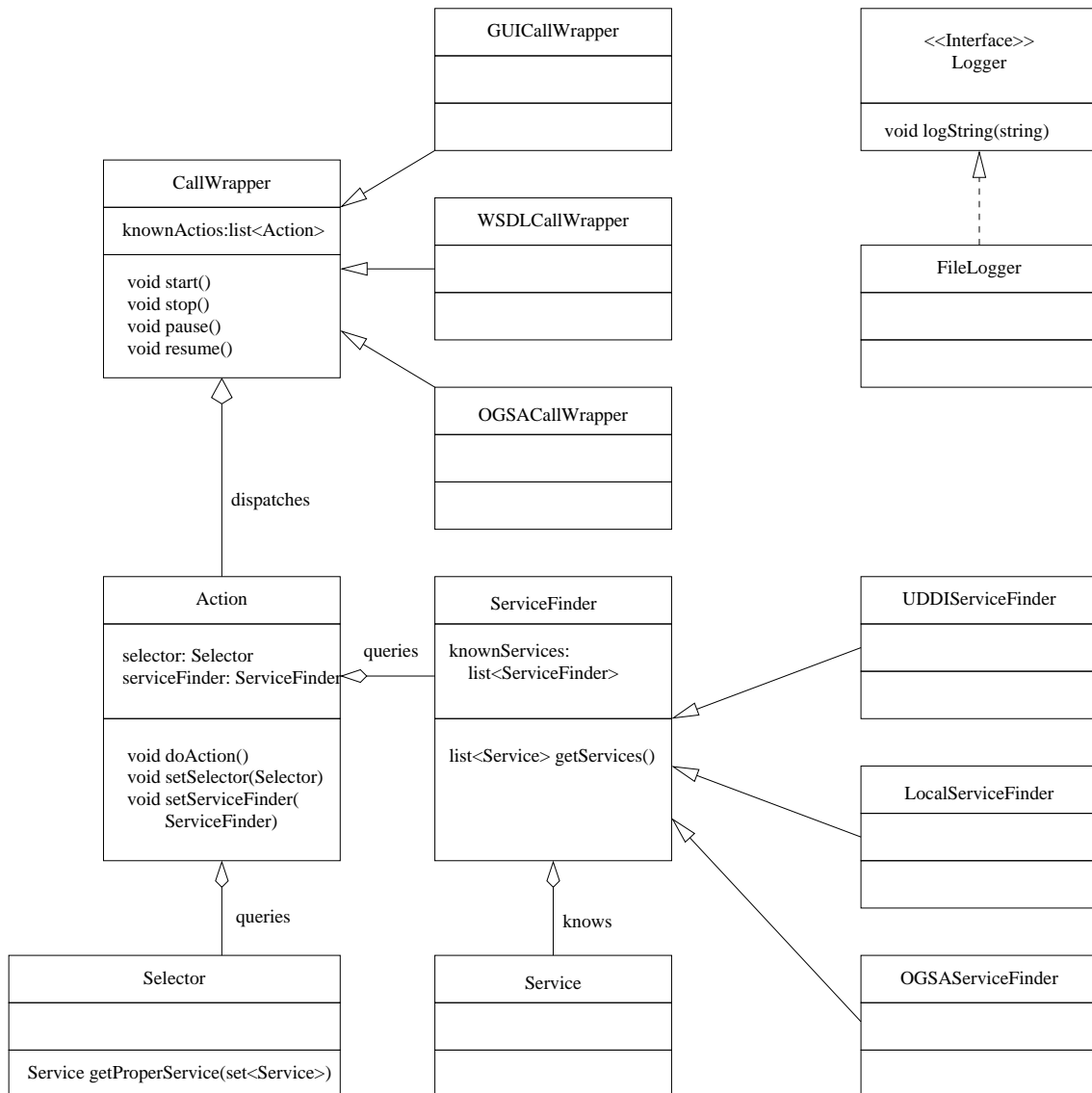


Figure 1: UML description of generic WP8 service

- Each functions returns a string. The strings will include a 3-digit number describing the general error/success class. After that follows a verbose description of the error. This convention is used by several internet protocols like HTTP or SMTP.

501 There was a serious security violation while accessing /foo/bar.

- Functions which return values use call-by-reference for these parameters. In the prototypes we use & to mark them. They will be the last parameters in the list. E.g. `string DATA_Foo(int in, string &out);`.
- According to the GridLab coding guidelines all data management functions have a common prefix (“DATA_”). For visualization we use “VIZ_”.
- The prototypes are written in C++-style.
- At the moment it is not clear how to handle user authentication so each functions has as its first parameter a credential. Maybe authentication and authorization will checked at an other level in this case we will remove the credential-parameter.

3 Replica Service

3.1 Description

The replication service manages sets of logical files called collections. For each file the service maintains a list of locations where instances of this file are saved (physical file). The user can add and delete instances for particular logical files.

The collections allow the user to link several related files into one entity. Thereby the user can structure his files, e.g. he can place all files of an experiment in one collection. On the other hand he can execute an operation on this bundle of files, e.g. move all files of an experiment to another site.

We will also associate a list of key-value pairs to each file set. So the user can maintain properties associated with file sets. He also can search for file sets by properties.

For performance reasons the replication service won't guarantee that all replicas of one file are identical. Therefore files have to be written once then they are published in the replica system. Files in the replica system should only be read.

Collection management: These functions allow the creation and deletion of collections of files. There are also functions for adding and removing logical files from/to collections.

Location management: With the functions of the collection management the user can add and remove physical locations of logical files. We also provide functions for retrieval of physical locations.

Attribute management: We will associate with collections a list of key-value pairs. Users can use them to store meta data about collections. The user can set and get properties of specified collections. He can also search for collections by giving a set of needed properties.

Replication: This group of functions provides the user means for creating new physical copies on a very abstract way. The replication service creates replicas of the selected files at the destination and updates the replica catalog accordingly.

3.2 Interface

Naming conventions for collection id's are not yet decided.

The **Properties** of a file or a collection are a list of key-value pairs. It is not clear whether the user is free to choose arbitrary keys.

```
// Collection management
string DATA_CreateCollection      (string credential, string collection_id);
string DATA_DeleteCollection     (string credential, string collection_id);
string DATA_AddFileToCollection  (string credential, string collection_id,
                                   string file_name);
string DATA_RemoveFileFromCollection (string credential, string collection_id,
                                   string file_name);
string DATA_ListCollection        (string credential, string collection_id,
                                   list<string> &file_names);

// Location management
string DATA_AddLocationOfFile    (string credential, string collection_id,
                                   string file_name,
```

```
string DATA_RemoveLocationOfFile (string credential, string collection_id,
                                   string file_name,
                                   string URL);
string DATA_GetLocationsOfFile (string credential, string collection_id,
                                 string file_name,
                                 list<string> &URLs);

// Attribute management

// value_type for example "String", "Integer", "Float", "Date", "File", ....
string DATA_SetPropertyOfFile (string credential, string collection_id,
                                string file_name, string attribute_name,
                                string value_type, string value);

// value_hint is not mandatory, only helps for faster removal
string DATA_RemovePropertyFromFile (string credential, string collection_id,
                                     string file_name, string attribute_name,
                                     string value_hint);

string DATA_GetPropertiesOfFile (string credential, string collection_id,
                                  string file_name,
                                  list<Property> &properties);

string DATA_FindFilesByProperties (string credential, string query,
                                   list<pair<string, string>>
                                   &collection_ids_and_file_names);

// value_type for example "String", "Integer", "Float", "Date", "File", ....
string DATA_SetPropertyOfCollection (string credential, string collection_id,
                                      string attribute_name, string value_type,
                                      string value);

// value_hint is not mandatory, only helps for faster removal
string DATA_RemovePropertyFromCollection(string credential,
                                         string collection_id,
                                         string attribute_name,
                                         string value_hint);

string DATA_GetPropertiesOfCollection(string credential, string collection_id,
                                       list<Property> &properties);

string DATA_FindCollectionsByProperties(string credential, string query,
                                       list<string> &collection_ids);

// High-level replication
string DATA_ReplicateCollectionTo (string credential,
```

```
string DATA_ReplicateFileTo(
    string collection_id,
    string URL);
(string credential,
 string collection_id,
 string file_name,
 string URL);
```

4 Data Movement Service

4.1 Description

This service provides to the user an abstract interface for data movement/copying. The actual file transfer protocol is hidden from the user. The protocol will be selected on availability and performance aspects. For this aspect we will work closely with WP-7.

For data transfer we will evaluate several protocols. The first release of this service will probably use `grid_url_copy`, `cp` and `scp`. During the lifetime of the GridLab project we will include support for further protocols/tools.

The Replica Catalog will not be updated after a successful move or copy. File movement and replica catalog update in one function is provided by `ReplicateCollectionTo` and `ReplicateFileTo`.

4.1.1 Interface

```
string DATA_MoveFile (string credential, string source_URL, string dest_URL);
string DATA_CopyFile (string credential, string source_URL, string dest_URL);
```

5 Streaming Service

5.1 Description

WP8 provides functionality for streaming. This includes a service for providers and consumers of data to find each other. We will also provide the actual streaming functionality. This last part will not be implemented as a service, but as a library to be used by providers and consumers directly. This library will utilize a service which will help to find and connect both streaming ends. Also, it should be possible to implement a service utilizing this library for advanced features like stream caching and multiplexing.

One end of a stream is specified via an URL containing information about

- protocol used by that endpoint
- host providing the endpoint
- port providing the endpoint
- path as additional information to endpoint (optional)

Example URL:

```
hdf5://kiwi.zib.de:3141/path/to/data/dataset_3.h5
```

5.2 Interface

It is not yet clear what will be stored inside `Metadata`. But the information stored in the URL is probably not sufficient for streaming data.

```
// hand-shake
string DATA_AnnounceURL      (string credential, string URL, MetaData metadata);
string DATA_SearchURL        (string credential, string URL, MetaData &metadata);
string DATA_RevokeAnnouncement (string credential, string URL);

// streaming
protocol dependent
```

6 Remote File Access

6.1 Description

WP8 will provide remote file access. There are two ways to provide this: (a) implementing a service which utilizes the remote streaming functionality to stream a remote file [3], or (b) map that functionality to GridFTP [2].

The solution (a) stays independent of any 3rd part software. The service establishes the connection of the respective streaming endpoints.

The solution (b) provides a much richer access functionality (eg. partial file access). The GridLab service will be reduced to the task of finding an appropriate GridFTP-Server, or, if not available, to stage (replicate) the file to an appropriate location, and to initiate the endpoint connection. The *access* of the data can be performed via low level GridFTP calls.

6.1.1 Interface

```
// remote file access
string DATA_StreamFile      (string credential, string src_URL, dest_URL);
string DATA_Open             (string credential, string URL, info &info);
string DATA_Close            (string credential, string URL, info info);
```

7 Viz Services

7.1 Description

Cactus and Triana will make use of various visualization environments, like Amira, OpenDX, IsoView and Triana-GUI. Some of these environments will be equipped with the respective data (file and stream) interfaces. Also, we plan to wrap some of these environments in Grid Services. Since it is not yet completely clear if this is technically possible (and sensible), the specification for the interfaces as defined in this document are preliminary.

Certainly, the interfaces will cover the connection from data source (application) to the viz service (eg. via data streaming or remote file access), the control of the viz process, and the control of the data source (steering)

7.2 Interface

```
// hand-shake
string VIZ_Announce      (string credential, string URL, MetaData metadata);
string VIZ_SearchURL     (string credential, string URL, MetaData *metadata);
string VIZ_RevokeAnnouncement (string credential, string URL);

// control viz process
string VIZ_ControlDescription (string credential, control_data_description descr);
string VIZ_Control         (string credential, control_data          ctrl );

// control application
string VIZ_SteerDescription  (string credential, steer_data_description  descr);
string VIZ_Steer            (string credential, steer_data            steer);
```

8 Online Recombination and Data Extraction Service

8.1 Description

- Recombination : combining several data files into one
- Data Extraction : extracting data from a set of files

Extraction and Recombination can be performed directly by this service. Resource demanding extraction or recombination processes will be submitted to the Grid as jobs.

If the resulting data should not be saved at the same site as the source data we have to decide where processing should take place [9].

Online recombination as well as data extraction is heavily dependent on the file format. Probably the main customer of this service will be Cactus. So the first implementation will support the HDF5 format which is used by Cactus.

For data extraction, it seems suitable to provide some functionality as a library, in order to link directly into that application and to perform data extraction (and hence reduction) on the fly and *before* creating any network traffic with the raw data. The design of this library is not finished, but will be based on the PaDEV¹ library from the DFN-Project “*Teleimmersion in Weitverkehrsnetzen*”, performed by ZIB and RRZN². It will accept pointers to structured memory data, and will convert these on the fly to a different set of structured data to be written to files or network streams. The output components may themselves act as application backends and utilize the GAT for further service invocation.

8.2 Interface

```
string DATA_RecombineFiles (string credential, list<string> source_URLs,
                             list<string> target_URL,
                             process_hint hint);
```

9 Dependencies of Components within WP8

As can be seen in Figure 2 most of our services rely on the data movement service. The data movement interface will be part of the GAT so we will use GAT for access to data movement.

¹Parallel Data Extraction and Visualization

²Regionales Rechenzentrum Hannover

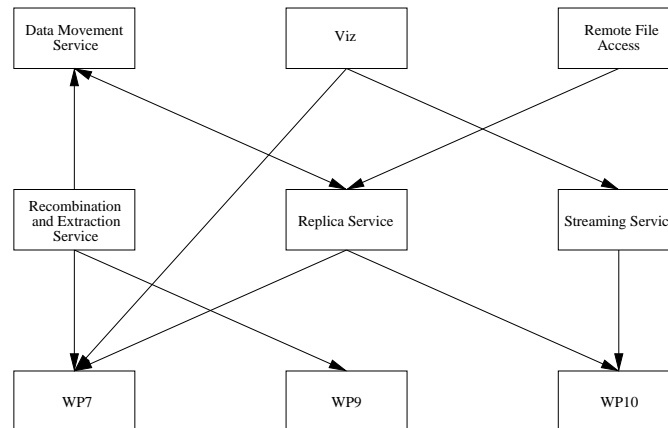


Figure 2: WP8 services and their dependencies

For calls to services in the core layer we will use their interface instead of the GAT. Those interfaces will be libraries, executables and/or WSDL/SOAP/UDDI based.

10 WP8 Dependencies on other Work Packages

In this Section we present the identified dependencies from services of other WPs. These dependencies are based on the requirement analysis and were further refined in the technical specification.

10.1 WP6 Security

GridLab services by definition follow any security specifications issued by the Security WP. This probably includes the requirement to communicate with security servers.

The WP8 services at various places need to utilize secure communication channels from host A to host B. We require from WP6 to establish these secure channels, or to provide means to establish those.

10.2 WP7 Adaptive Components

WP7 will provide Adaptive Components – Services which are able to react dynamically and adaptively to a changing Grid environment. WP7 will draw its information from the MDS (WP10) and Monitoring (WP11).

WP8 will utilize these Services for various decision processes:

- where to perform data extraction
- tuning of I/O parameters
- location of storage resources
- choice of communication channels and protocols

10.3 WP9 Resource Management

WP9 will provide means to execute jobs on remote resources - WP8 will utilize this functionality in order to schedule jobs for data recombination, data extraction and (possibly) visualization.

10.4 WP10 Information Services

We need an API to insert, update and retrieve arbitrary information in/from the GIS (Grid Information System) because we can't specify yet which types of information we want to store in the GIS in the future.

11 Mapping between services and deliverables

This Section will describe how the components defined in this document relate to the WP8 deliverables from the Annex1 of the GridLab project proposal. We list only those deliverables containing software component implementations, and omit all report and document deliverables.

11.1 D8.3: Data discovery and tracking services

Data discovery and tracking will be performed by the Replica Management Service (see Section 3). In particular, the Replica Management System will contain a Replica Catalog, storing location data for all files maintained by the Replica Management System. This catalog can be searched (discovery) and dynamically updated (tracking).

11.2 D8.4: Low level data access services

Low level data access covers various separate capability providers: streaming (Section 5), remote file access (Section 6), and low level data movement (Section 4). As low level techniques, these do not yet cooperate with the Replica Management System.

11.3 D8.5: High level data access services

As the low level services from D8.4 are put under the regime of the replica management system, higher level equivalents of these services will result (Section 3). Further functionality will be provided by more abstract or specialized services, as the recombination service (Section 8).

11.4 D8.6: Data management integration

The capability providers provided in D8.3, D8.4 and D8.5 need to be fully integrated into the GAT, and via GAT into the GridLab applications (Cactus, Triana, Portal and Visualization). This will also enable to realize the visualization service (Section 7).

11.5 D8.8: (Parallel) dynamic data filters

As the infrastructure for data management, movement and visualization got in place with the previous deliverables, D8.8 will focus on the efficient extraction and filtering of application data. This covers the utilization of the PaDEV data extraction as described in Section 8, and its incorporation into the GAT. Also, special remote file servers (from D8.4) may be equipped with dynamic filtering capabilities.

References

- [1] Annex 1 - description of work. GridLab - A Grid Application Toolkit and Testbed, September 2001.

- [2] W. Allcock, J. Bresnahan, I. Foster, L. Liming, J. Link, and P. Plaszczac. Gridftp update. Technical report, Argonne National Laboratory (ANL), January 2002.
- [3] Werner Benger, Hans-Christian Hege, Andre Merzky, Thomas Radke, and Edward Seidel. Efficient distributed file I/O for visualization in grid environments. In *PDC'99: Simulation and Visualization on the Grid*, Lecture Notes in Computational Science and Engineering. Springer, January 2000.
- [4] Technical Board. The gridlab general architecture. GridLab - A Grid Application Toolkit and Testbed, June 2002.
- [5] Ann Cervenak et al. Giggle: A framework for constructing scalable replica location services, 2002.
- [6] WP1. Gridlab requirement analysis: WP1 grid application toolkit. GridLab - A Grid Application Toolkit and Testbed, March 2002.
- [7] WP10. Report on the current gis capabilities and limitations. GridLab - A Grid Application Toolkit and Testbed, April 2002.
- [8] WP6. WP6 security: Initial requirements. GridLab - A Grid Application Toolkit and Testbed, April 2002.
- [9] WP7. Use cases for adaptive components. GridLab - A Grid Application Toolkit and Testbed, April 2002.
- [10] WP8. Requirement analysis. GridLab - A Grid Application Toolkit and Testbed, April 2002.
- [11] WP9. Requirement analysis and definition for the grid resource management system. GridLab - A Grid Application Toolkit and Testbed, April 2002.