



IST-2001-32133

GridLab - A Grid Application Toolkit and Testbed

## Deliverable D8.3 - Data discovery and tracking services

---

Author(s):	Thorsten Schlütt, Andre Merzky, Andrei Hutanu, Felix Hupfeld, Florian Schintke
Document Filename:	
Work package:	WP 8 - Data Handling and Visualization
Partner(s):	
Lead Partner:	ZIB
Config ID:	GridLab-08-DATA-0003-M12
Document classification:	Internal

---

**Abstract:** This document describes the APIs for the first data tracking prototype. We also discuss our implementation of the data movement service, as we have already deployed it within the testbed.



## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Data Discovery and Tracking Services</b>	<b>2</b>
2.1	Limitations . . . . .	2
2.2	Return Codes . . . . .	2
2.3	Collection Management . . . . .	2
2.3.1	CreateCollection . . . . .	2
2.3.2	DeleteCollection . . . . .	3
2.3.3	AddFileToCollection . . . . .	3
2.3.4	RemoveFileFromCollection . . . . .	3
2.3.5	ListCollection . . . . .	3
2.4	Location Management . . . . .	3
2.4.1	AddLocationOfFile . . . . .	3
2.4.2	RemoveLocationOfFile . . . . .	3
2.4.3	GetLocationsOfFile . . . . .	3
<b>3</b>	<b>Data movement services</b>	<b>4</b>
3.1	Reliable Operations . . . . .	4
3.1.1	CopyFile . . . . .	4
3.1.2	MoveFile . . . . .	4
3.1.3	DeleteFile . . . . .	4
3.1.4	Parameters . . . . .	4
3.1.5	Return values . . . . .	5
3.2	Default Operations . . . . .	5
3.2.1	CopyFileDefaults . . . . .	5
3.2.2	MoveFileDefaults . . . . .	5
3.2.3	DeleteFileDefaults . . . . .	5
<b>4</b>	<b>Summary</b>	<b>5</b>

## 1 Introduction

This document describes the API of the GridLab data tracking service, in association with this service's first prototype release.

According to the Annex 1, we initially expected to make major enhancements to the GIS. However, the GIS did not seem appropriate for the kinds of information needed for data tracking. Instead, we developed a service which is optimized for storing replica related information. Our service is used for storing the locations of replicas and grouping replicas into collections. But we still use the GIS for information such as the size of file systems, which is needed (e.g.) for replication services.

We are ahead of our timeschedule in implementing the data movement service, and have already deployed this within the testbed. For this reason, we have included the documentation for this service within this deliverable report.

## 2 Data Discovery and Tracking Services

The Data Discovery and Tracking Services provide means to store the locations of input, output and intermediate files at a central place. Job chains can use this service to retrieve their input files and to publish their output files. This service can also be used for parameter sweeps to publish the location of the executable.

Communities which use a common set of files e.g. in high energy physics can use it to reduce data movements between sites by publishing all existing copies/replicas of their files.

The API of our first release is based on the Globus API. Calls to the service can be either done via `gsi-soap` or via the `GAT-Adaptor`.

As already described in D8.2 this prototype deals with logical files organised in collections. To each logical file a set of locations is associated. Further releases will provide additionally more sophisticated abstractions and APIs.

### 2.1 Limitations

Logical file names and collection names can be any legal string with one exception. They are not allowed to include a `"/`". Locations of files are allowed to have arbitrary form.

If two different collections contain logical files with the same name these files are treated as different entities.

### 2.2 Return Codes

Each function returns a string indicating the success or failure of the call. The strings are loosely based on common Internet protocol concepts for error codes.

On success, functions return `"200 OK"`. On failure e.g. `"300 file not found"` is returned. Here, `"file not found"` is a verbose description of the problem which caused the failure. The verbose string is not expected to be parsed by a program.

### 2.3 Collection Management

The following sections describe the functions for managing collections.

#### 2.3.1 CreateCollection

```
string CreateCollection(string collection_name);
```

Creates a collection with the name `collection_name`.

### 2.3.2 DeleteCollection

```
string DeleteCollection(string collection_name);
```

Deletes the collection with the name `collection_name`.

### 2.3.3 AddFileToCollection

```
string AddFileToCollection(string collection_name, string file_name);
```

Adds the file `file_name` to the collection `collection_name`.

### 2.3.4 RemoveFileFromCollection

```
string RemoveFileFromCollection(string collection_name, string file_name);
```

Removes the file `file_name` from the collection `collection_name`.

### 2.3.5 ListCollection

```
string ListCollection(string collection_name, list<string> &file_names);
```

Lists the files in the collection `collection_name`.

## 2.4 Location Management

This section describes the management of locations of files. Files in this context are identified by logical file names and their collection names.

### 2.4.1 AddLocationOfFile

```
string AddLocationOfFile(string collection_name, string file_name,  
                        string location);
```

Adds a location to a file.

### 2.4.2 RemoveLocationOfFile

```
string RemoteLocationOfFile(string collection_name, string file_name,  
                           string location);
```

Removes a location of a file.

### 2.4.3 GetLocationsOfFile

```
string GetLocationsOfFile(string collection_name, string file_name,  
                        list<string> &locations);
```

Lists the locations of file `file_name` in collection `collection_name`.

## 3 Data movement services

The file movement service is based on the Globus `gass_copy` library. It provides a high-level file movement API to the GAT using the current supported protocols in `gass_copy` (http, https, ftp and gsiftp). There are two sets of methods: a set of fully parametrized methods that provide reliable operations; and a set of default methods that do not use reliable operations.

### 3.1 Reliable Operations

Reliability functionality is provided through use of the ftp restart plugin. This plugin is configurable by three parameters: the maximum number of retries to attempt; the interval to wait between retries; and the deadline after which no further retries will be attempted. These parameters are found in all of the three methods that provide reliable operations as `max_retries`, `interval_sec` and `deadline_sec`, respectively.

#### 3.1.1 CopyFile

```
string CopyFile(string source_URL, string dest_URL, int max_retries,  
               long interval_sec, long_deadline_sec);
```

Copies the file specified by `source_URL` to `dest_URL`.

#### 3.1.2 MoveFile

```
string MoveFile(string source_URL, string dest_URL, int max_retries,  
               long interval_sec, long_deadline_sec);
```

Moves the file from `source_URL` to `dest_URL`.

#### 3.1.3 DeleteFile

```
string DeleteFile(string in_URL, int max_retries, long interval_sec,  
                 long_deadline_sec);
```

Deletes the `in_URL` file.

#### 3.1.4 Parameters

`max_retries`: The maximum number of times to retry the operation before giving up on the transfer. If this value is less than or equal to 0, the service will keep trying to restart the operation until it completes or the deadline is reached with an unsuccessful operation.

`interval_sec`: The interval (in seconds) to wait after a failure before retrying the transfer. If the interval is 0 seconds, then an exponential backoff is used.

`deadline_sec`: The absolute timeout (in seconds) before failing the operation.

The supported protocols are https, http, ftp and gsiftp. In the `DeleteFile` and `MoveFile` methods, deletion is not possible for http or https URLs.

### 3.1.5 Return values

Each function returns a string indicating the success or failure of the call. The strings are loosely based on common Internet protocol concepts for error codes.

On success, functions return “200 OK”. On failure e.g. “550 550 /home/vis/bzfhatan/test\_copy: No such file or directory” is returned. Here, “550 /home/vis/bzfhatan/test\_copy: No such file or directory” is a verbose description of the problem which caused the failure. The verbose string is not expected to be parsed by a program. Most of the error codes are received from the servers. When no error code is returned by the servers, “300” is returned.

## 3.2 Default Operations

For each of the above methods, there exists a paired method that takes only the URLs as parameters and uses default values (zeroes) for the restart parameters. These methods are:

### 3.2.1 CopyFileDefaults

```
string CopyFileDefaults(string source_URL, string dest_URL);
```

### 3.2.2 MoveFileDefaults

```
string MoveFileDefaults(string source_URL, string dest_URL);
```

### 3.2.3 DeleteFileDefaults

```
string DeleteFileDefaults(string in_URL);
```

## 4 Summary

We delivered all components as stated in D8.2. With the data movement service we are already ahead of our time schedule. The sources are provided on our web pages.