IST-2001-32133
GridLab - A Grid Application Toolkit and Testbed

Deliverable D8.5 - High level data access services

Author(s):	Felix Hupfeld, Andrei Hutanu, Andre Merzky, Thorsten Schütt, Brygg Ullmer
Document Filename:	
Work package:	WP 8 - Data Handling and Visualization
Partner(s):	
Lead Partner:	ZIB
Config ID:	GridLab-08-DATA-0005-M24
Document classification:	Internal

Abstract: This document describes the release of the high level data access services and libraries, with prototypes deployed on the GridLab testbed.

Contents

1 Introduction **2**

2 Remote Data Access Architecture **2**

3 Remote Data Access API **2**

1 Introduction

This deliverable implements high level data access techniques including dynamic filtering, hierarchical data access, and partial data access. They are built on top of the lower level techniques from earlier deliverables.
High level data access is provided by a POSIX-like API. The implementation of this API is transparent to local and remote data access. The API additionally provides powerful extensions for expressing and evaluating access "patterns" within binary data. The implementation of our high level data access routines is based upon (a) the GridFTP service and protocol as provided by Globus, and (b) our own implementation of a high performance access library built upon FALLS (see below), as well as associated plugin and protocol extensions to GridFTP.
The next section includes the abstract of a publication describing the overall architecture and the concept of FALLS in more detail. We then outline the API that is provided by our solution. The software is tested and benchmarked, and is available (as a prototype) at `http://www.gridlab.org/WorkPackages/wp-8/software/D8.5.tgz`.

2 Remote Data Access Architecture

The design of the GridLab high level data access services is detailed in [1]. The abstract of this publication follows:

> *We present a method for the efficient partial access of remote files that significantly reduces the number of messages that have to be transferred via the network. Thereby the influence of network latencies on the achieved transmission rate can be reduced dramatically if regular subsets of a file are requested.*
>
> *Our approach uses a compact pattern description to read several parts of the file in a single operation. We implemented our remote data access mechanism as a plugin for GridFTP and measured remote partial file access with and without our pattern access and present the results.*

3 Remote Data Access API

```
// utility API for pattern related calls
off_t* _io_wrap_get_pat_sequence (const char *pat, off_t &elems);

// open close read write seek etc. - standard POSIX interface

// calls on file descriptors
int      io_wrap_open           (const char* url, int flags, ...);
int      io_wrap_close          (int fd);
ssize_t  io_wrap_read           (int fd,       void*  buf, size_t n);
ssize_t  io_wrap_write          (int fd, const void*  buf, size_t n);
off_t    io_wrap_lseek          (int fd, off_t  offset, int whence);

// additional calls
int      io_wrap_size           (int fd, off_t* size);
int      io_wrap_pread          (int fd, const char* pat, char** buf, size_t* n);
int      io_wrap_pread_buf      (int fd, const char* pat, char*  buf, size_t  n);
```

```
int      io_wrap_pread_dump     (int fd, const char* pat);

// calls on IW_FILE*
IW_FILE* io_wrap_fdopen         (int fd,          const char* mode);
IW_FILE* io_wrap_fopen          (const char* url, const char* mode);
IW_FILE* io_wrap_freopen        (const char* url, const char* mode, IW_FILE* fp);
int      io_wrap_fclose         (IW_FILE* fp);
ssize_t  io_wrap_fread          (      void* buf, size_t s, size_t n, IW_FILE* fp);
ssize_t  io_wrap_fwrite         (const void* buf, size_t s, size_t n, IW_FILE* fp);
void     io_wrap_clearerr       (IW_FILE* fp);
int      io_wrap_feof           (IW_FILE* fp);
int      io_wrap_ferror         (IW_FILE* fp);
int      io_wrap_fileno         (IW_FILE* fp);

int      io_wrap_fgetpos        (IW_FILE* fp, off_t* pos);
int      io_wrap_fsetpos        (IW_FILE* fp, off_t* pos);
int      io_wrap_fseek          (IW_FILE* fp, long   offset, int whence);
long     io_wrap_ftell          (IW_FILE* fp);
void     io_wrap_rewind         (IW_FILE* fp);
int      io_wrap_fflush         (IW_FILE* fp);
int      io_wrap_ferror         (IW_FILE* fp);


// additional calls
int      io_wrap_fsize          (IW_FILE* fp, off_t* size);
int      io_wrap_fpread         (IW_FILE* fp, const char* pat, char** buf, size_t* n);
int      io_wrap_fpread_buf     (IW_FILE* fp, const char* pat, char*  buf, size_t  n);

int      io_wrap_fprintf        (IW_FILE* fp, const char* format, ...);
int      io_wrap_vfprintf       (IW_FILE* fp, const char* format, va_list ap);

// not implemented calls
int      io_wrap_fscanf         (IW_FILE* fp, const char* format, ...);
int      io_wrap_vfscanf        (IW_FILE* fp, const char* format, va_list ap);

// config calls
void     io_wrap_set_driver     (driver_t driver);
void     io_wrap_set_cache_size (off_t    size);


// ATTENTION: pread and pread_buf move the file pointer by the
// size of the pattern, not by the number of bytes read
```

References

[1] T. Schütt, A. Merzky, A. Hutanu, and F. Schintke. Remote partial file access using compact pattern descriptions. Accepted for CCGrid 2004.