Grid File Access Library (GFAL) JAVA API

Valeria Ardizzone

INFN Catania – EGEE-II NA3/NA4

Health e-Child Application Developer Tutorial
Cern(Geneve), September 25-26, 2006

www.eu-egee.org

Information Society

GFAL: Introduction

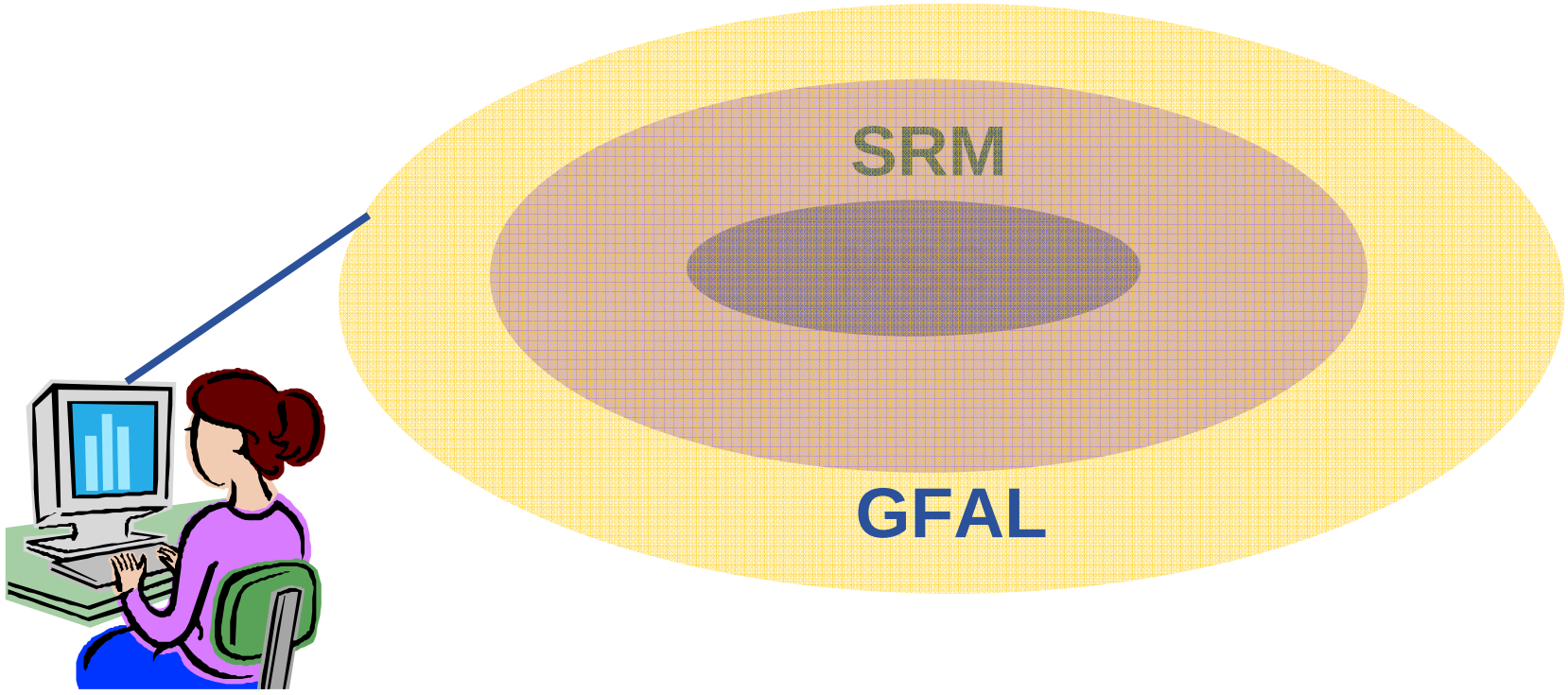
- **What is GFAL for?**
  - Grid storage interactions today require using some existing software components:
    - The catalog services to locate valid replicas of files in order to :
      - ***Download*** *them to the user local machine*
      - ***Move*** *them from a SE to another one*
      - *Make job running on the worker node able to* ***access and manage*** *files stored on remote storage element.*
    - The SRM software to ensure:
      - *Files existence on disk or disk pool (they are recalled from mass storage if necessary)*
      - *Space allocation on disk for new files (they are possibly migrated to mass storage later)*

GFAL: Main features

- **The GFAL Features**
  - Hides interactions to the SRM to the end user
  - Provides a Posix-like interface for File I/O Operation
  - Based on shared libraries (both threaded e unthreaded version)
  - Needs only one header file (gfal_api.h) to write C applications
  - Supports following protocols :
    - file for local access **nfs-like**
    - dcap, gsidcap and kdcap for dCache access protocol
    - rfio for CASTOR access protocol.
  - Access to SRMs in secure mode, i.e. using a valid Grid proxy obtained by voms-proxy-init command.

GFAL, SRM and MSS

1. **GFAL** will be the highest level interface
2. It will take care of SRM and Replica Managers and protocols (transparent for the user)
3. **SRM** will take care of the handling with **MSS** (not visible for the user)



GFAL: File Names

- **GFAL works with all Grid types name:**
  - Logical File Name (LFN)
    - lfn:baud/testgfal15
  - Grid Unique IDentifier (GUID)
    - guid:2cd59291-7ae7-4778-af6d-b1f423719441
  - File Replica (SURL)
    - srm://wacdr002d.cern.ch:8443/castor/cern.ch/user/b/baud/testgfal
  - Transport file name (TURL).
    - rfio:////castor/cern.ch/user/b/baud/testgfal15

GFAL: Environment

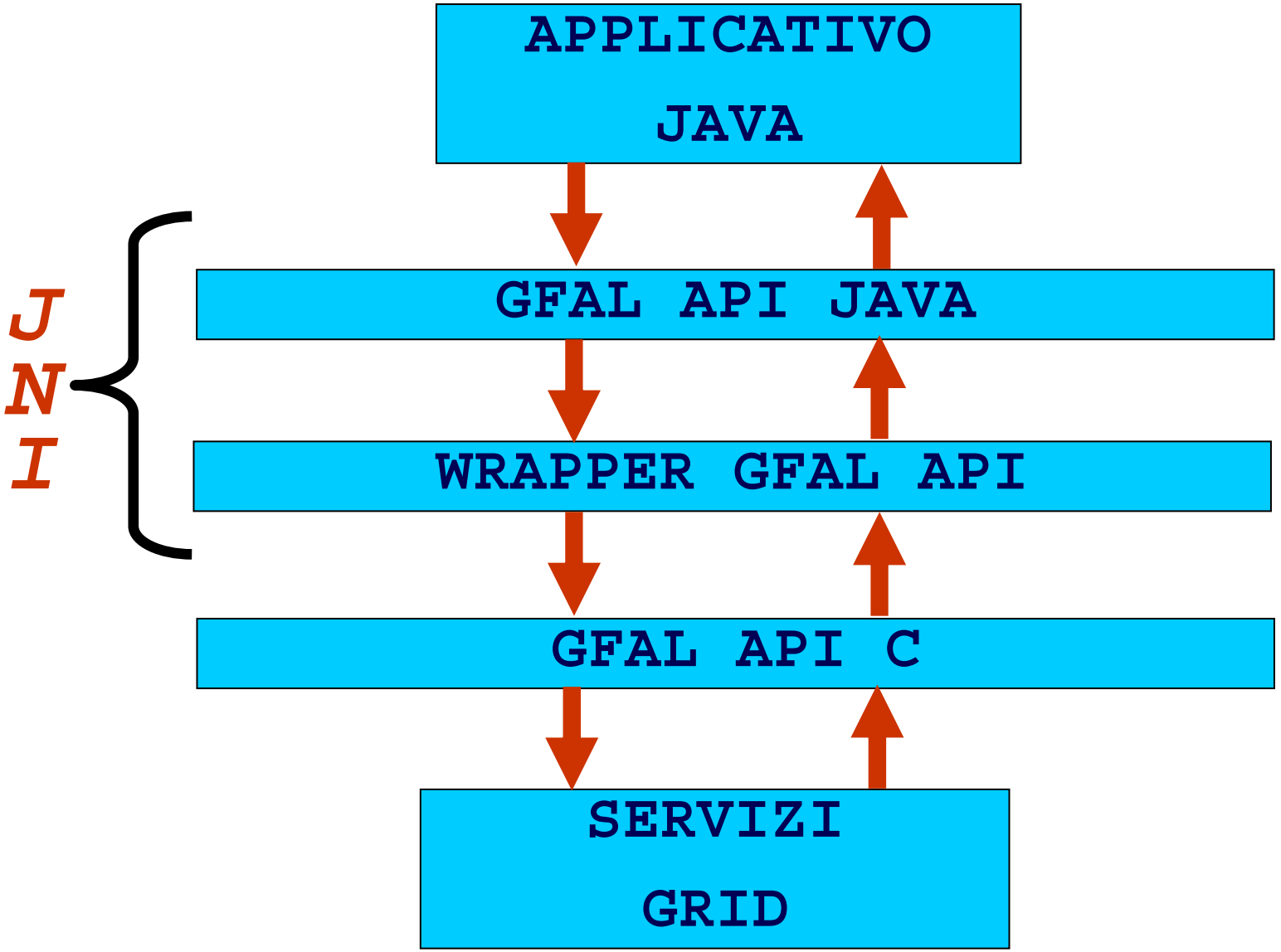
- **Auxiliary linked libraries**
  - libcgsi_plugin_gsoap_2.3
  - libglobus_gss
  - api_gsi_gcc32dbg
  - libglobus_gss_assist_gcc32dbg).

- **Environment Variables**
  - LCG_GFAL_VO (-> **gilda**)
  - LCG_GFAL_INFOSYS (-> *grid004.ct.infn.it:2170*)
  - LCG_CATALOG_TYPE (-> *lfc*)
  - LCG_RFIO_TYPE (-> *dpm*)
  - LFC_HOST (-> *lfc-gilda.ct.infn.it*)
  - LD_LIBRARY_PATH

GFAL JAVA API: an example

```
import it.infn.catania.gfal.*;
public class Sample {
   public static void main (String args[]) {
     String fileName = "aliserv6.ct.infn.it";
     try {
       GFalFile gFalFile= new GFalFile();
       gFalFile.createFile(fileName,644,false,false);
       byte[] dati = new byte[1024];
       int ret = gFalFile.writeFile(dati);
       if (ret==-1) {
         System.exit(1);
       }
       ret = gFalFile.closeFile();
       if (ret==-1) {
        System.exit(1);
       }
       ret = gFalFile.lfcRegisterFile("lfn:/grid/gilda/test/1.dat");
       if (ret==-1) {
        System.exit(1);
       }
     }
     catch (GFalFileException exc) {
       exc.printStackTrace();
     }
   }
}
```

GFAL JAVA API: Soluzione Tecnica



GFAL JAVA API: technical solution

Classe Java:

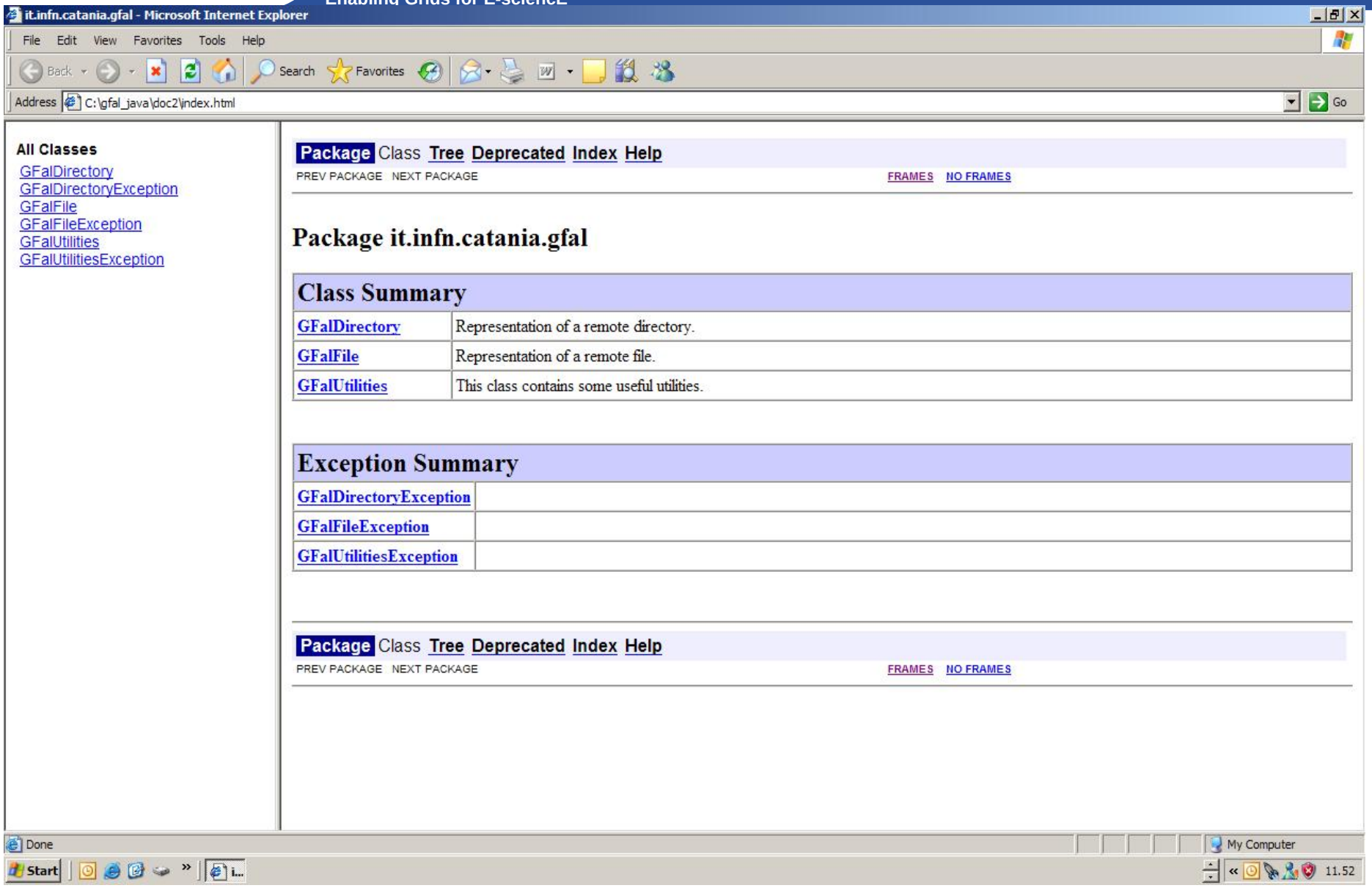
```
package it.infn.catania.gfal;
…
public class GFalFile
{
   static {
                System.loadLibrary("GFalFile");
   }
   …
   private native int openGFalFile(String fileName, int mode, int
   perm, boolean isLargeFile);
   private native int closeGFalFile(int fd);
   …
   public int openFile(String fileName, int flags, int mode, boolean
   isLargeFile) throws GFalFileException {
        …
        int ret = openGFalFile(fileName, flags, mode, isLargeFile);
        …
   }
}
```

GFAL JAVA API: technical solution

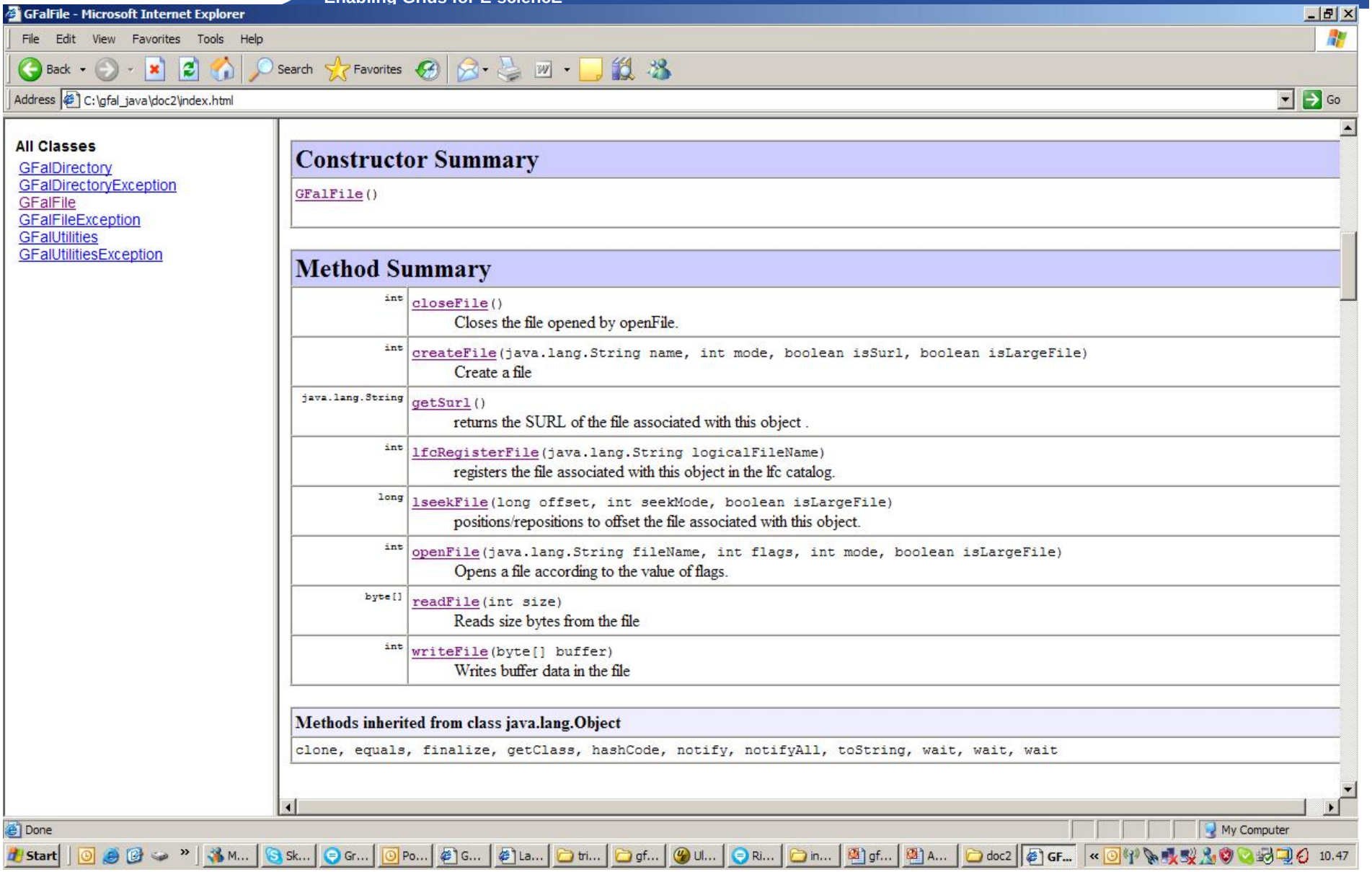
Wrapper C (libGFalFile.so):

```
#include <gfal_api.h>
#include <jni.h>
#include "it_infn_catania_gfal_GFalFile.h"
…
JNIEXPORT jint JNICALL
   Java_it_infn_catania_gfal_GFalFile_openGFalFile (JNIEnv *env,
   jobject obj, jstring fileName, jint mode, jint permission,
   jboolean largeFile)
{
   int fd;
   const char *str = (*env)->GetStringUTFChars(env, fileName, 0);
   if(largeFile)
        fd = gfal_open64(str, mode, permission);
   else
        fd = gfal_open(str, mode, permission);
   (*env)->ReleaseStringUTFChars(env, fileName, str);
   …//check errors
   return fd;
}
```

GFAL API: Javadoc



GFAL API: Javadoc



Introduction to the LFC Java API practicals

- **Browse to the agenda page:**
  http://agenda.cern.ch/fullAgenda.php?ida=a063217

- **Follow the link "more information" on the topic "Practicals on GFAL Java API usage"**