

# jLite

A Lightweight Java API for gLite

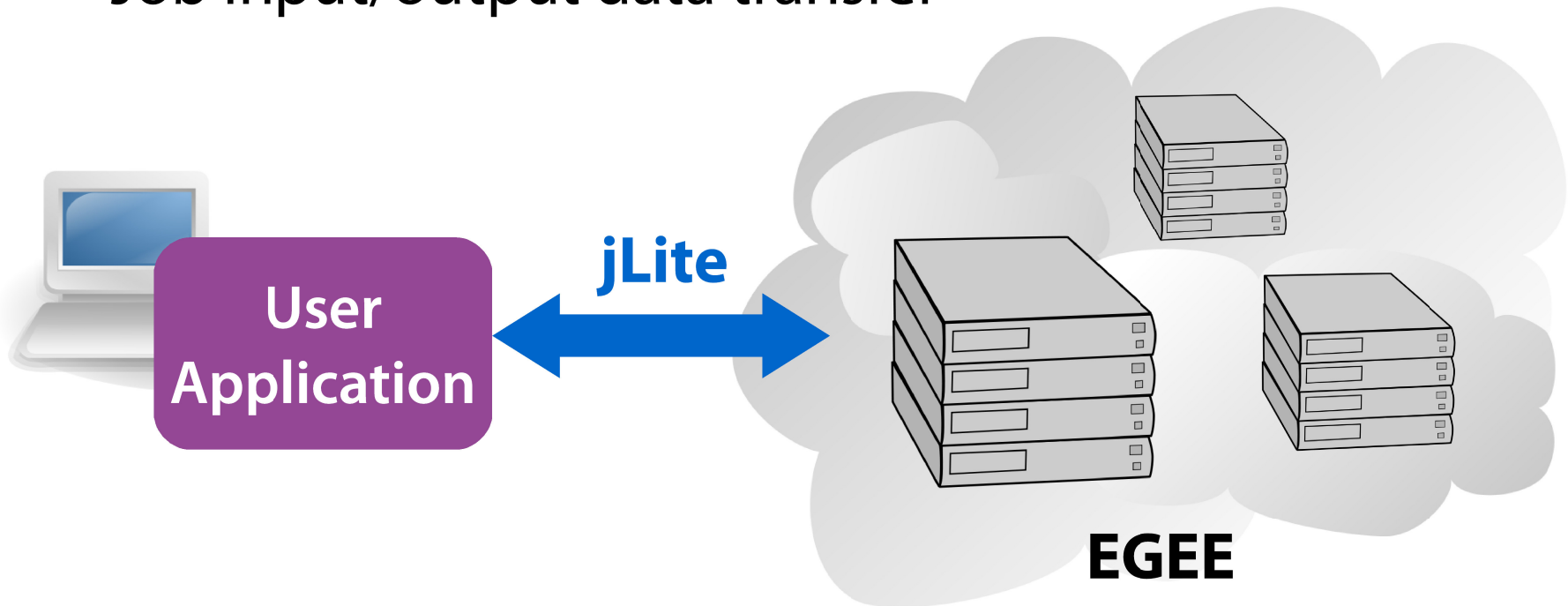
**Oleg Sukhoroslov**

Centre for Grid Technologies  
and Distributed Computing, ISA RAS

[os@isa.ru](mailto:os@isa.ru)

# jLite

- Minimalistic Java API for gLite based grids
  - VOMS proxy management
  - Job submission
  - Job input/output data transfer



# Problems with gLite Java APIs

- Expose low-level operations
- Scattered among several packages
- Complex external dependencies
- Lack of documentation
- Implicitly installed gLite User Interface

# jLite Goals

- Simple high-level API
  - 20% functionality for 80% cases
- Easy to install
  - All external dependencies included
  - No need to install gLite User Interface
- Platform-independent
  - Forget about Scientific Linux
- Well-documented

# Current Implementation

- Early "preview" release is available for download
  - <http://j-lite.googlecode.com/>
- Apache License 2.0
- Work in progress...
- This presentation is current documentation

# Dependencies

## gLite

- glite-jdl-api-java
- glite-security-delegation-java
- glite-security-trustmanager
- glite-security-util-java
- glite-voms
- glite-wms-wmproxy-api-java

## Other

- Apache Axis
- Apache Commons CLI, Discovery, Logging, Log4J
- BouncyCastle
- Condor ClassAd
- Cryptix
- Java CoG Kit
- Pure TLS
- SAAJ
- Servlet
- WSDL4J

# Installation and Configuration

- Unpack to any directory (JLITE\_HOME)
- Copy your Grid credentials to \$HOME/.globus
  - Windows "C:\Documents and Settings\User\.globus"
  - Linux "/home/user/.globus"
- Configure VO services (pre-configured for GILDA VO)
  - Copy CA certificate to JLITE\_HOME/etc/certs/ca
  - Add VOMS server file to JLITE\_HOME/etc/vomses
  - Copy VOMS certificate to JLITE\_HOME/etc/certs/voms
  - Add WMPProxy server file to JLITE\_HOME/etc/wms

# Usage

- Add all jars from JLite\_HOME/lib to your classpath
- Configure, create and use `jlite.GridSession`

```
// configure Grid session
GridSessionConfig config = new GridSessionConfig();
config.setUserKeyPasswd("*****");
...
// create Grid session
GridSession session =
    GridSessionFactory.create(config);
// use Grid session
session.createProxy("gilda", 12*3600);
...
```



# GridSessionConfig

- **setUserCert**(String userCert) // path to user certificate
- **setUserKey**(String userKey) // path to user private key
- **setUserKeyPasswd**(String pwd) // private key passphrase
- **setProxyPath**(String proxyFile) // path to user proxy file
- **setCADir**(String dir) // path to CA certificates dir
- **setVOMSDir**(String dir) // path to VOMS servers dir
- **setVOMSCertDir**(String dir) // path to VOMS certificates dir
- **addVOMSServer**(VOMSServerInfo info) // add VOMS server
- **setWMSDir**(String dir) // path to WMPProxy servers dir
- **addWMPProxy**(String vo, String url) // add WMPProxy server
- **setDelegationId**(String id) // set delegation id

# GridSession

- GlobusCredential **createProxy**(String vo, int lifetime)
- GlobusCredential **getProxy**()
- void **delegateProxy**(String delegationId)
- List<String> **listMatchedCE**(String jdl)
- String **submitJob**(String jdl, String inboxDir)
- JobStatus **getJobStatus**(String jobId)
- String **getJobState**(String jobId)
- void **getJobOutput**(String jobId, String outboxDir)
- void **cancelJob**(String jobId)
- void **destroyProxy**() ...

# Running Grid Job in 8 Steps

(src/jlite/examples/Demo.java)

1. Configure and create Grid session
2. Create user proxy
3. Delegate user proxy to WMPProxy server
4. Load job description
5. List matched resources
6. Submit job to Grid
7. Monitor job status
8. Download job output

# Step 1. Configure and Create Grid Session

```
GridSessionConfig config = new GridSessionConfig();  
  
// user's private key passphrase  
config.setUserKeyPasswd("*****");  
  
// path to CA certificates  
config.setCADir("etc/certs/ca");  
  
// paths to VOMS configuration files and certificates  
config.setVOMSDir("etc/vomses");  
config.setVOMSCertDir("etc/certs/voms");  
  
// path to WMPProxy configuration files  
config.setWMSDir("etc/wms");  
  
// create Grid session  
GridSession session = GridSessionFactory.create(config);
```

## Step 2. Create User Proxy

```
// GILDA VO, 12 hours
```

```
GlobusCredential proxy =  
    session.createProxy("gilda", 12*3600);
```

## Step 3. Delegate User Proxy to WMPProxy Server

```
session.delegateProxy( "myId" );
```

## Step 4. Load Job Description (org.glite.jdl.JobAd)

```
JobAd jad = new JobAd();  
jad.fromFile("test/test.jdl");  
String jdl = jad.toString();
```

## Step 5. List Matched Resources (optional)

```
List<String> ces =  
    session.listMatchedCE(jdl);  
  
System.out.println(  
    "Matched Computing Elements:");  
  
for (String ce : ces) {  
    System.out.println("\t" + ce);  
}
```



## Step 6. Submit Job to Grid

```
String jobId =  
    session.submitJob(jdl, "test/input");
```

- Only normal jobs are supported now

# Inside submitJob()

- Register job in WMPProxy server
- Get job input sandbox destination URI
- Upload input directory to sandbox via GridFTP
- Start job

## Step 7. Monitor Job Status

```
String jobState = "";  
do {  
    Thread.sleep(10000);  
    jobState = session.getJobState(jobId);  
    System.out.println(jobState);  
} while ( !jobState.equals("DONE") &&  
          !jobState.equals("ABORTED") );
```

## Step 8. Download Job Output

```
if (jobState.equals("DONE")) {  
    String outputDir =  
        "test/" + Util.getShortJobId(jobId);  
    session.getJobOutput(jobId, outputDir);  
}
```

Created user proxy: C=IT,O=GILDA,OU=Personal  
Certificate,L=Institute for Systems Analysis RAS,CN=Oleg  
Sukhoroslov,CN=proxy

Matched Computing Elements:

gn0.hpcc.szaki.hu:2119/jobmanager-lcgpbs-gilda  
grid010.ct.infn.it:2119/jobmanager-lcgpbs-infinite  
grid010.ct.infn.it:2119/jobmanager-lcgpbs-long  
grid010.ct.infn.it:2119/jobmanager-lcgpbs-short  
iceage-ce-01.ct.infn.it:2119/jobmanager-lcgpbs-infinite  
iceage-ce-01.ct.infn.it:2119/jobmanager-lcgpbs-long  
iceage-ce-01.ct.infn.it:2119/jobmanager-lcgpbs-short  
gilda-01.pd.infn.it:2119/jobmanager-lcgpbs-gilda

Started job:

[https://glite-rb3.ct.infn.it:9000/u\\_bHTUEEIpgY\\_-QH2fMuIw](https://glite-rb3.ct.infn.it:9000/u_bHTUEEIpgY_-QH2fMuIw)

Job status: SCHEDULED

Job status: RUNNING

Job status: RUNNING

Job status: RUNNING

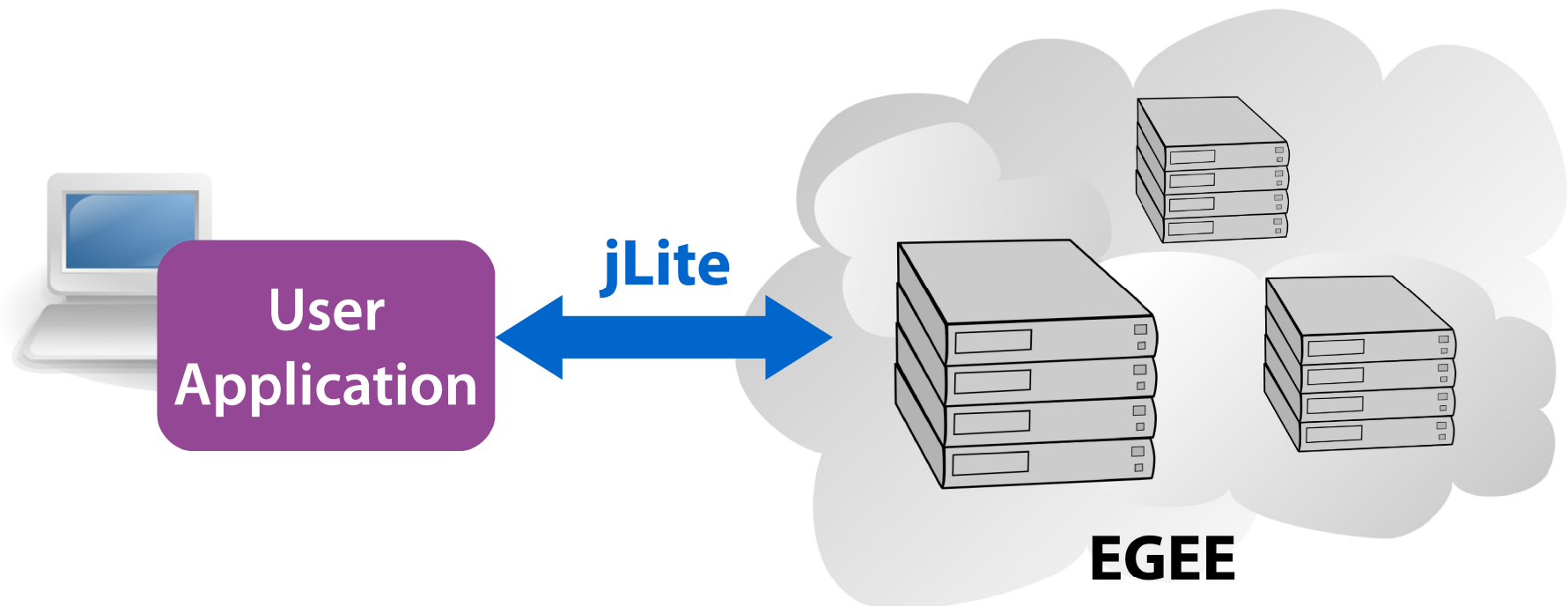
Job status: RUNNING

Job status: DONE

Job output is downloaded to: test/u\_bHTUEEIpgY\_-QH2fMuIw

# Applications

- "Grid-enabled" Java applications
- Grid portals and services
- Cross-platform command-line or GUI tools



# Command Line Interface (jLite CLI)

- **proxy-init** -vo <vo-name> -time <lifetime>
- **proxy-info**
- **proxy-delegate** -id <deleg-id>
- **job-match** -jdl <jdl-file> -id <deleg-id>
- **job-submit** -jdl <jdl-file> -in <input-dir> ...
- **job-status** -job <job-id>
- **job-output** -job <job-id> -out <output-dir>
- **job-cancel** -job <job-id>
- **proxy-destroy**

# Test Job Description

(JLITE\_HOME/test/test.jdl)

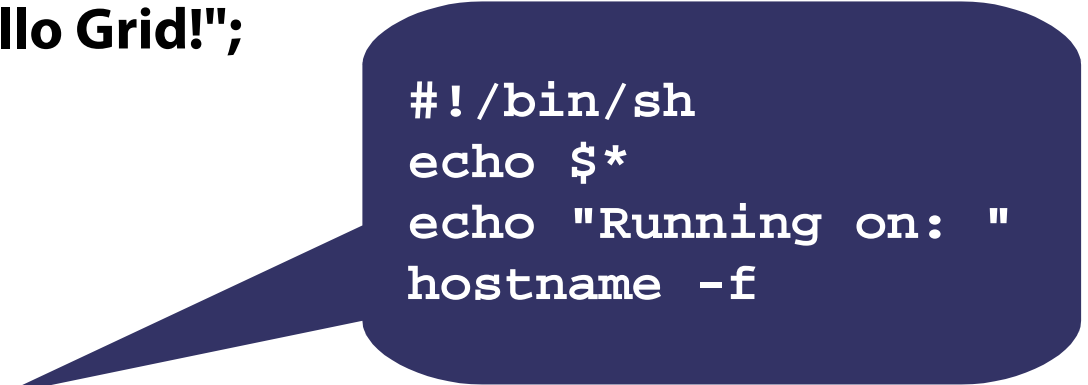
**Type** = "Job";  
**JobType** = "Normal";

**Executable** = "/bin/sh";  
**Arguments** = "test.sh Hello Grid!";

JLITE\_HOME/test/input/test.sh

**StdOutput** = "std.out";  
**StdError** = "std.err";

**InputSandbox** = {"test.sh"};  
**OutputSandbox** = {"std.out","std.err"};



```
#!/bin/sh  
echo $*  
echo "Running on: "  
hostname -f
```

**RetryCount** = 3;

**Requirements** = (other.GlueCEStateStatus == "Production");  
**Rank** = (-other.GlueCEStateEstimatedResponseTime);



# CLI User Session Example

(all commands should run from JLITE\_HOME/cli directory)

```
proxy-init -vo gilda
```

```
proxy-delegate
```

```
job-match -jdl test/test.jdl
```

```
job-submit -jdl test/test.jdl -in test/input
```

```
job-status -job jobId...
```

```
job-output -job jobId...
```

```
proxy-destroy
```

# Related Work

- Simple API for Grid Applications, <http://saga.cct.lsu.edu/>
  - Unified API (like MPI) for several Grid middleware
  - jLite focuses on gLite middleware
- gEclipse, <http://www.geclipse.eu/>
  - Cross-platorm Grid user workbench (Java+Eclipse)
  - No API, jLite inspiration
- Portable gLite User Interface
  - GILDA VM UI, <https://gilda.ct.infn.it/VirtualServices.html>
  - Grid2Win, <http://grid2win.gilda-forge.ct.infn.it/>

# Conclusion

- A minimalistic Java API for gLite based grid
  - Draft implementation is available
- A cross-platform command-line interface
  - Runs on Windows
- Work in progress...
  - Different job types
  - Information services
  - Data management

**Thank you!**

os@isa.ru

<http://j-lite.googlecode.com/>