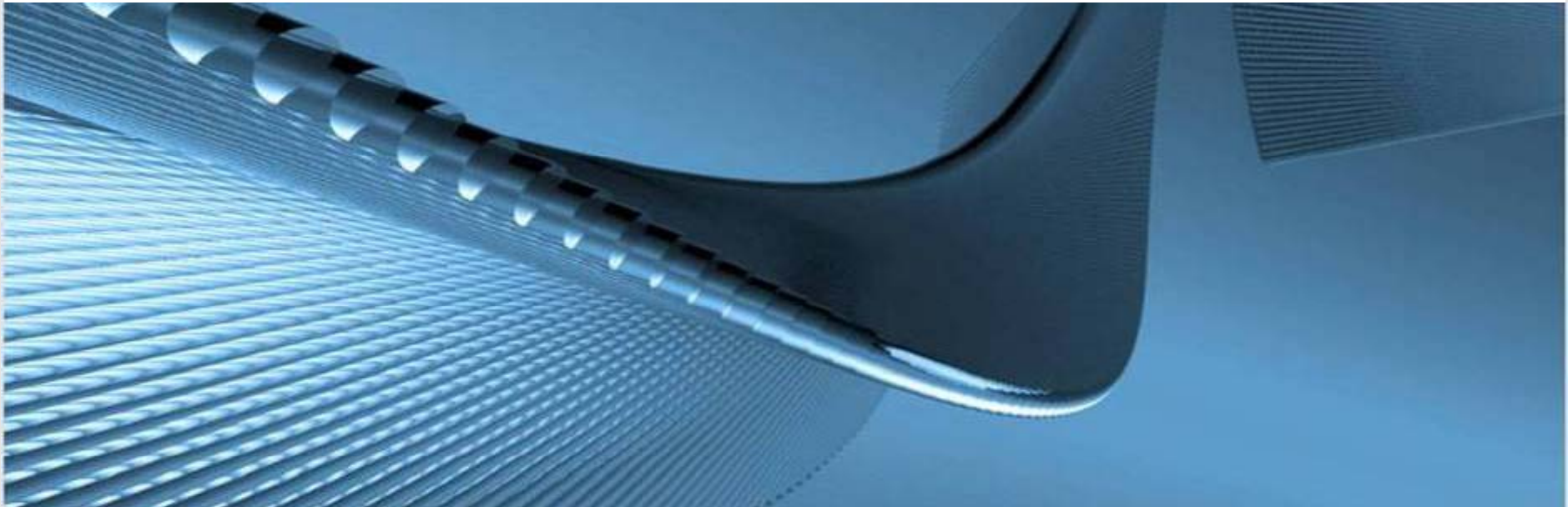


# Assuring Grid Accesses in the g-Eclipse Project

Jie Tao  
Karlsruhe Institute of Technology  
jie.tao@kit.edu



# Outline

- Introduction to g-Eclipse
- Quality Control in g-Eclipse
- Metrics Examples
- Hierarchical Testing Concept
- Integration in the Nightly Build System
- Conclusion

# g-Eclipse: A General Framework for Accessing the Grid

- Many application domains start using Grid infrastructures
- But...
  - Grid technology is complex
    - Different systems are used
      - Middleware (Unicore, gLite, Globus, GRIA, ...)
      - Many separate tools (i.e for installation, monitoring, ...)
    - Different programming paradigms
      - Batch type systems vs. service oriented systems
      - Many programming languages
- e-Users want to interact with the grid infrastructure
  - Without knowing all details (development, deployment, testing, management, ...)
- → **Tooling is necessary!!**
  - **Wizards, Editors, ...**
  - **Hide the complexity!!**

# g-Eclipse: the Idea

- Provides a general UI framework/eco system for the different grid actors
  - Grid applications users
  - Grid resources providers and operators
  - Grid application developers
- (Re-)use Eclipse and contribute
  - Eclipse is an eco system
    - Build for extension
    - More than a JAVA IDE
    - The biggest “coordinated” Open Source project
  - Gain OS independence (by using JAVA)

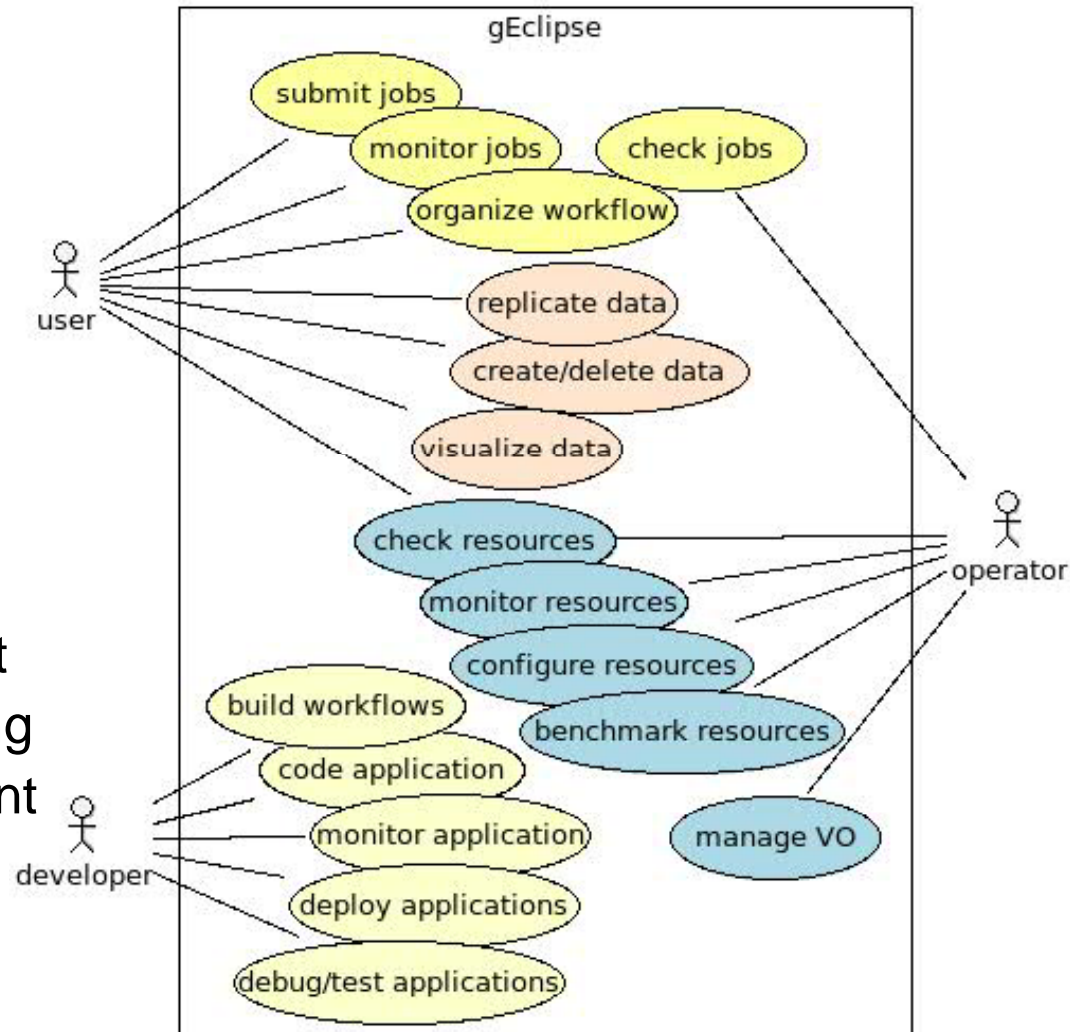
# Grid user roles & use cases

- 3 different roles

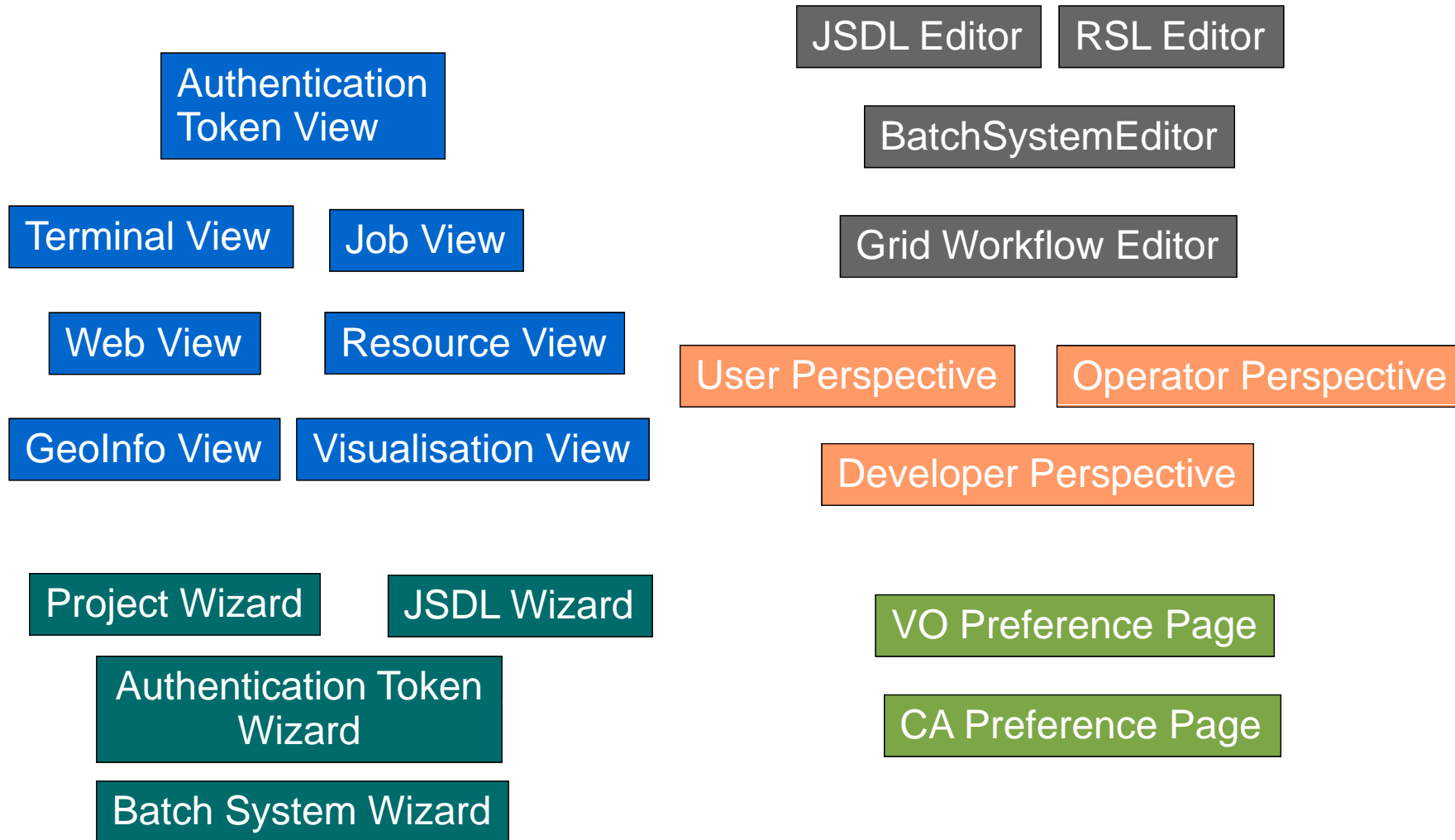
- User
- Operator
- Developer

- In general...

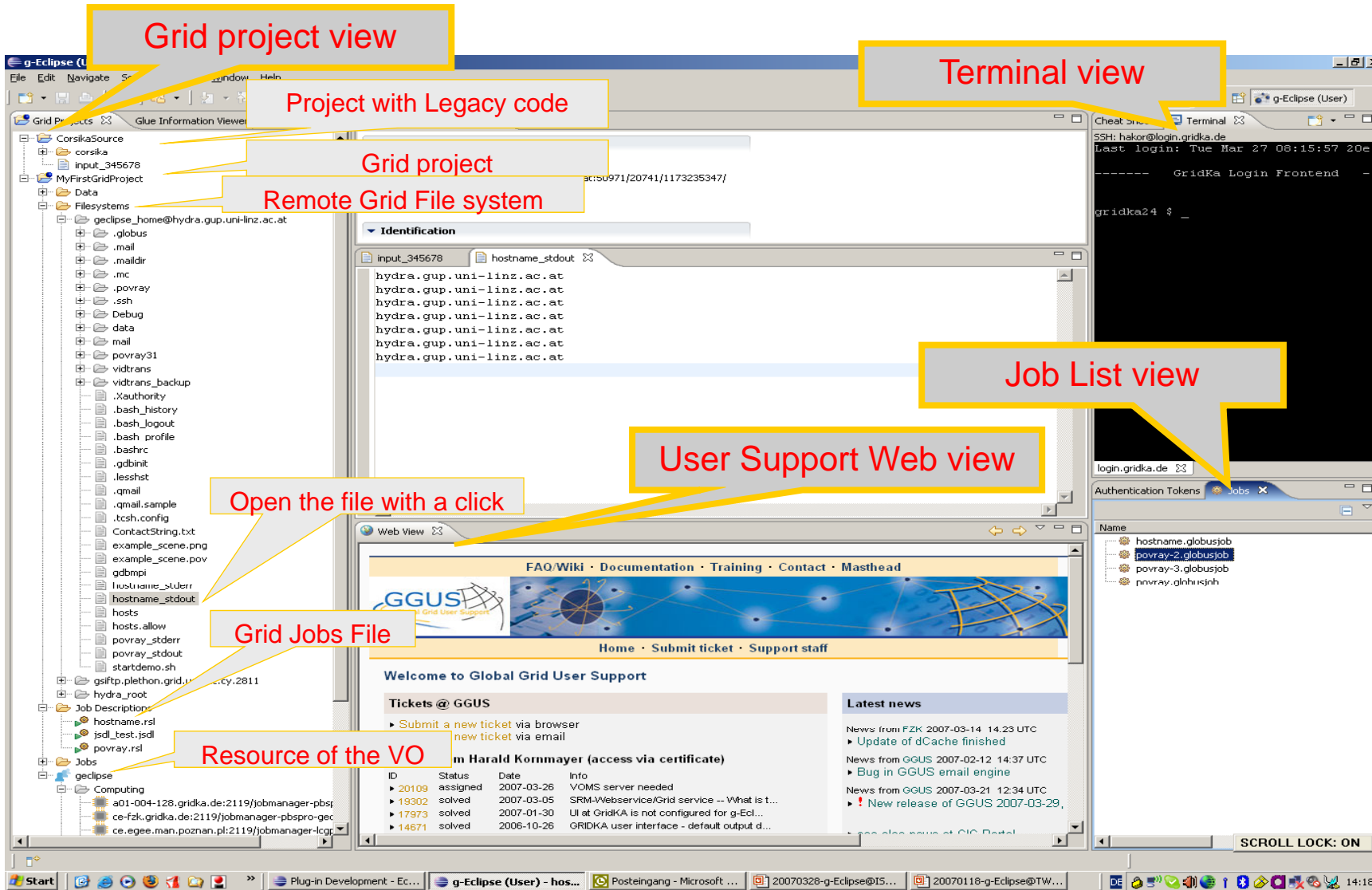
- Job management
- Resource management
  - Files
  - Applications
  - Hardware
- Application deployment
- Infrastructure monitoring
- Application development
- Visualization tools



# User Interface contributions



# Screenshots



The screenshot shows the gEclipse IDE interface with several callouts pointing to specific features:

- Grid project view**: Points to the Project Explorer on the left.
- Project with Legacy code**: Points to the 'MyFirstGridProject' in the Project Explorer.
- Grid project**: Points to the 'Data' folder within 'MyFirstGridProject'.
- Remote Grid File system**: Points to the 'Filesystems' folder in the Project Explorer.
- Terminal view**: Points to the Terminal window showing SSH login details.
- Job List view**: Points to the 'Jobs' tab in the Authentication Tokens window.
- User Support Web view**: Points to the Web View window displaying the GGUS website.
- Open the file with a click**: Points to a file in the Project Explorer.
- Grid Jobs File**: Points to the 'Jobs' folder in the Project Explorer.
- Resource of the VO**: Points to the 'Computing' folder in the Project Explorer.

The Web View window displays the GGUS website with the following content:

FAQ/Wiki · Documentation · Training · Contact · Masthead

Home · Submit ticket · Support staff

Welcome to Global Grid User Support

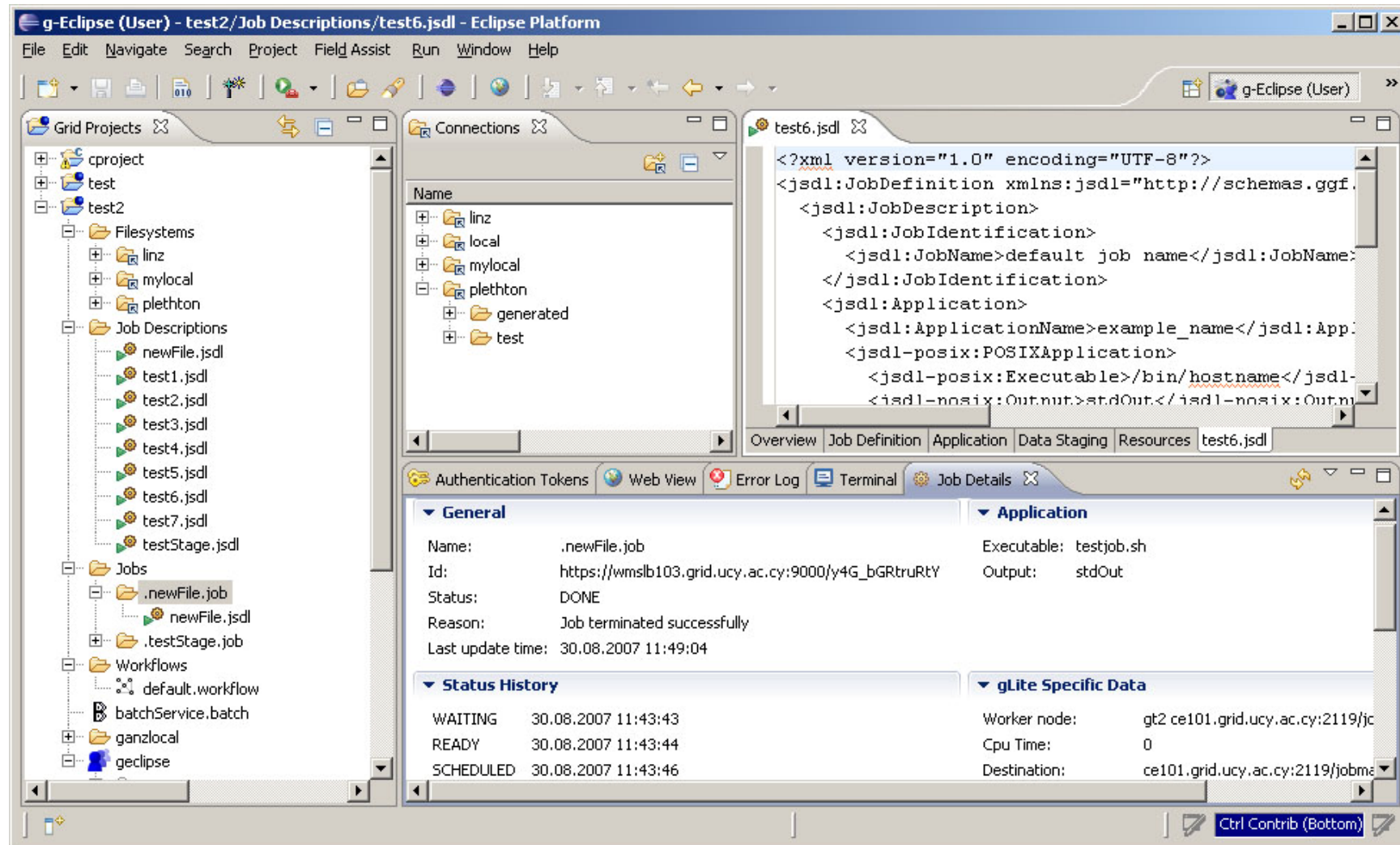
Tickets @ GGUS

ID	Status	Date	Info
20109	assigned	2007-03-26	VOMS server needed
19302	solved	2007-03-05	SRM-Webservice/Grid service -- What is t...
17973	solved	2007-01-30	UI at GridKA is not configured for g-Ecl...
14671	solved	2006-10-26	GRIDKA user interface - default output d...

Latest news

- News from FZK 2007-03-14 14:23 UTC
  - Update of dCache finished
- News from GGUS 2007-02-12 14:37 UTC
  - Bug in GGUS email engine
- News from GGUS 2007-03-21 12:34 UTC
  - New release of GGUS 2007-03-29.

# Screenshot



The screenshot shows the g-Eclipse IDE interface. The main window displays the XML content of a job definition file, `test6.jsdl`. The XML is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<jsd1:JobDefinition xmlns:jsdl="http://schemas.ggf.
<jsd1:JobDescription>
  <jsd1:JobIdentification>
    <jsd1:JobName>default job name</jsdl:JobName>
  </jsdl:JobIdentification>
  <jsd1:Application>
    <jsd1:ApplicationName>example_name</jsdl:App
    <jsd1-posix:POSIXApplication>
      <jsd1-posix:Executable>/bin/hostname</jsdl-
      <jsdl-posix:Output>stdout</jsdl-posix:Outm
```

The Job Details panel at the bottom shows the following information:

General		Application	
Name:	.newFile.job	Executable:	testjob.sh
Id:	https://wmslb103.grid.ucy.ac.cy:9000/y4G_bGRtruRtY	Output:	stdout
Status:	DONE		
Reason:	Job terminated successfully		
Last update time:	30.08.2007 11:49:04		

Status History		gLite Specific Data	
WAITING	30.08.2007 11:43:43	Worker node:	gt2 ce101.grid.ucy.ac.cy:2119/jc
READY	30.08.2007 11:43:44	Cpu Time:	0
SCHEDULED	30.08.2007 11:43:46	Destination:	ce101.grid.ucy.ac.cy:2119/jobme



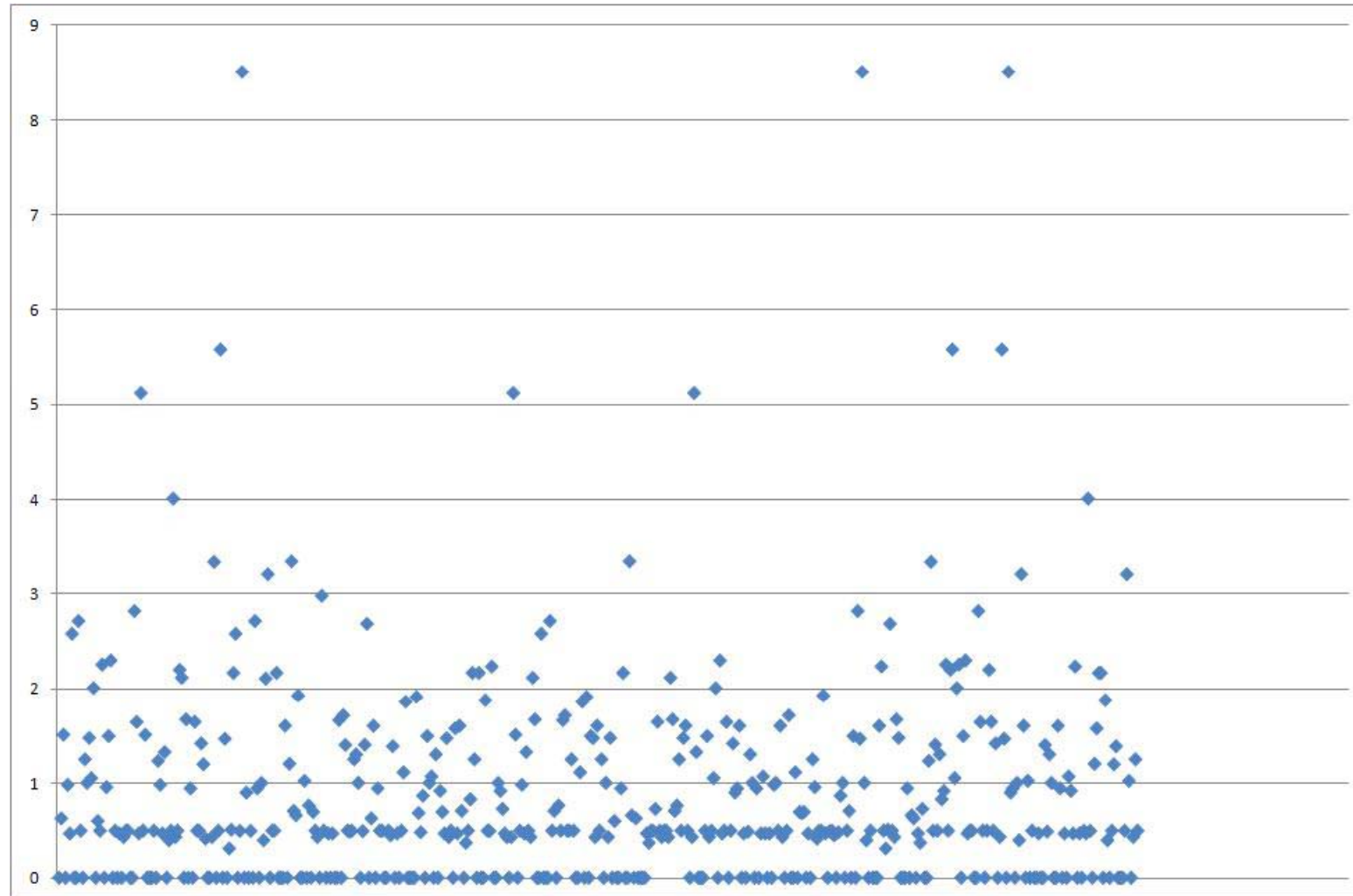
# Quality Control in g-Eclipse

- Code quality
  - Functionality
  - Error/exception handling
  - Efficiency (short code, optimal solution)
  - Simplicity
  - Structure
- Documentation quality
  - User manual (for users)
  - Development roadmap (for developers)
- Programming style
  - Identical view
- Approach: metrics and testing
  - Unit Test
  - Manual test

# Evaluating Code Quality with Metrics

- Cyclomatic Complexity (CC)
  - Evaluates the complexity of a method. A low CC is generally better (<10)
- Size: size of a method
  - Evaluates the understandability of the code
- Weighted Methods per Class (WMC)
  - Number of methods in a class or sum of the complexities of the methods
  - The larger the number of methods in a class, the greater the potential impact on children due to the inheritance
- Response for a Class (RFC)
  - Number of methods that can be invoked in response to a message or by some method in the class
  - Measures the amount of communications with other classes
  - The larger the RFC, the greater the complexity (testing, debugging)

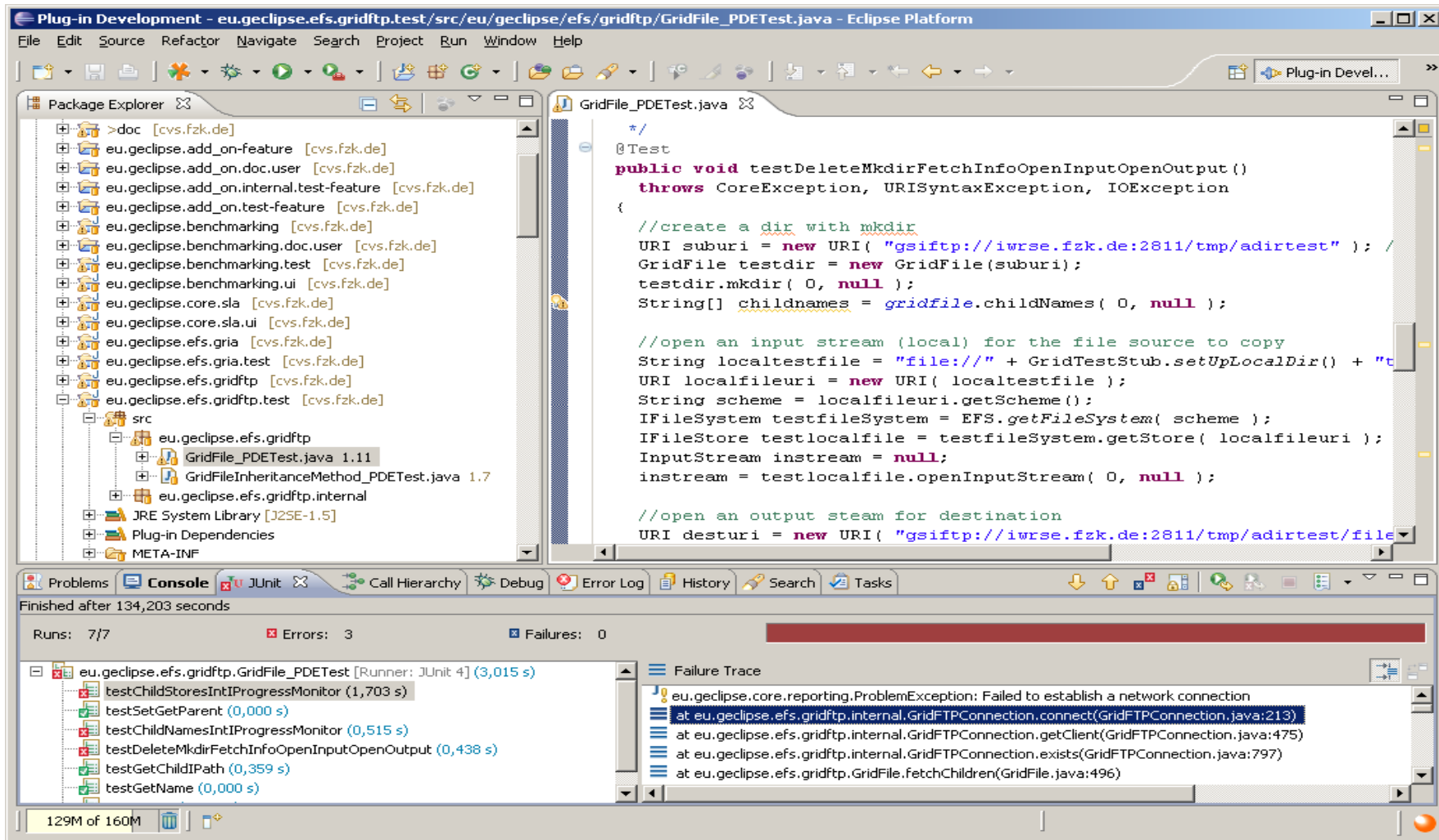
# Sample Results (CC)



# JUnit Testing

- Junit: a simple testing tool for JAVA
- Junit test method
  - An ordinary method without any parameters
  - Assertion statements indicate success or failure
  - Run using a test runner
- Eclipse unit test
  - Junit integrated
    - Runtime: Java runtime + Junit test classes
  - PDE extension
    - Runtime: Java runtime + Junit test classes +Eclipse runtime
    - Required for testing grid functionality

# PDE-Test Example



The screenshot displays the Eclipse IDE interface. The top window shows the source code of `GridFile_PDETest.java`. The code includes a test method `testDeleteMkdirFetchInfoOpenInputOpenOutput()` that performs several operations: creating a directory via `GridFile.mkdir()`, opening an input stream to copy a file, and opening an output stream for a destination file. The Package Explorer on the left shows the project structure, including the `src` folder and the `GridFile_PDETest.java` file. The bottom console window shows the test results, indicating that the test `testDeleteMkdirFetchInfoOpenInputOpenOutput` passed successfully. The failure trace on the right shows a `ProblemException` related to a network connection failure during the `testChildStoresIntIPProgressMonitor` test.

```
GridFile_PDETest.java
/*
 *
 * @Test
 * public void testDeleteMkdirFetchInfoOpenInputOpenOutput()
 *     throws CoreException, URISyntaxException, IOException
 * {
 *     //create a dir with mkdir
 *     URI suburi = new URI( "gsiftp://iwrse.fzk.de:2811/tmp/adirtest" ); /
 *     GridFile testdir = new GridFile(suburi);
 *     testdir.mkdir( 0, null );
 *     String[] childnames = gridfile.childNames( 0, null );
 *
 *     //open an input stream (local) for the file source to copy
 *     String localtestfile = "file://" + GridTestStub.setUpLocalDir() + "t
 *     URI localfileuri = new URI( localtestfile );
 *     String scheme = localfileuri.getScheme();
 *     IFileSystem testfileSystem = EFS.getFileSystem( scheme );
 *     IFileStore testlocalfile = testfileSystem.getStore( localfileuri );
 *     InputStream instream = null;
 *     instream = testlocalfile.openInputStream( 0, null );
 *
 *     //open an output steam for destination
 *     URI desturi = new URI( "gsiftp://iwrse.fzk.de:2811/tmp/adirtest/file
```

Finished after 134,203 seconds

Runs: 7/7      Errors: 3      Failures: 0

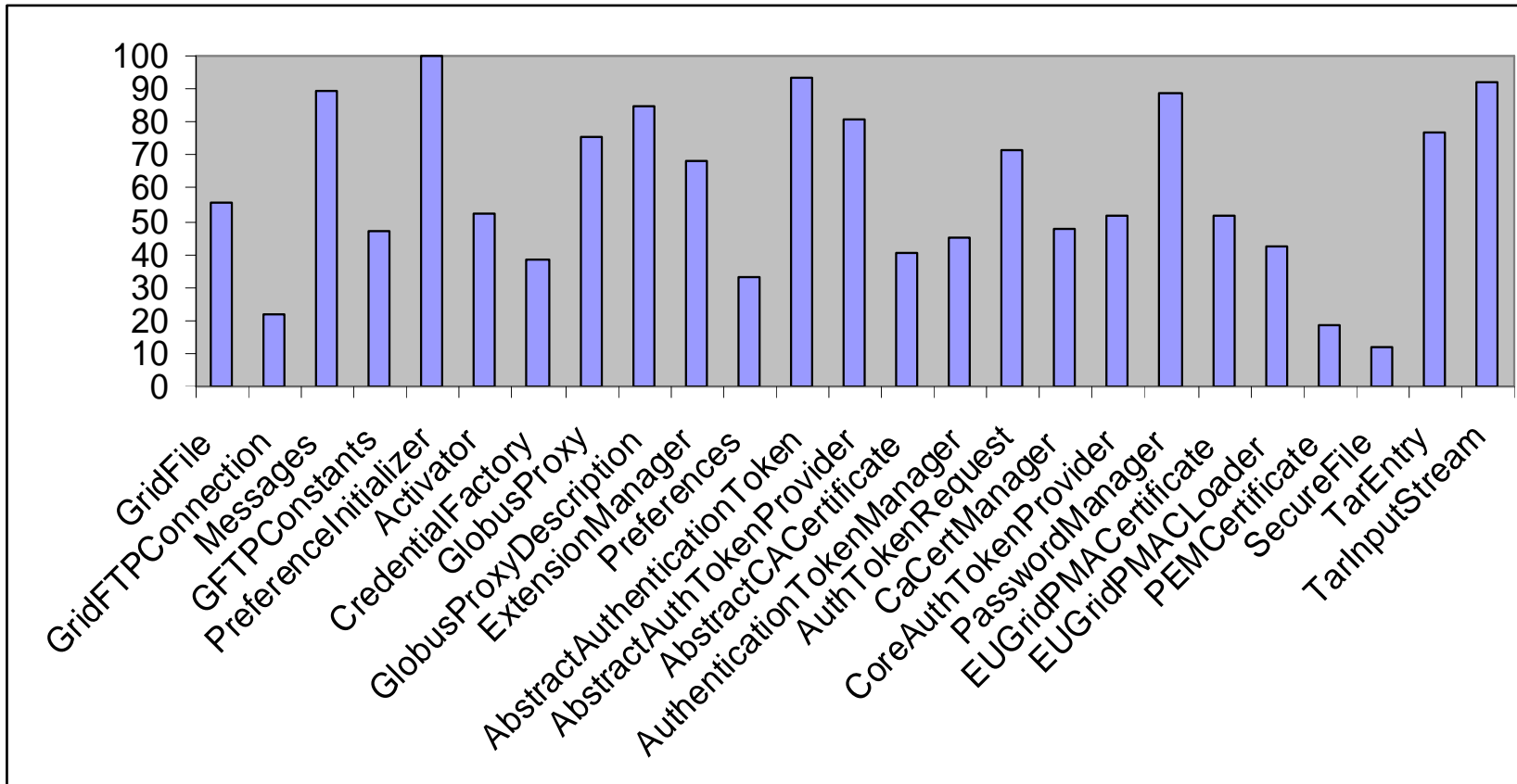
eu.geclipse.efs.gridftp.GridFile\_PDETest [Runner: JUnit 4] (3,015 s)

- testChildStoresIntIPProgressMonitor (1,703 s)
- testSetGetParent (0,000 s)
- testChildNamesIntIPProgressMonitor (0,515 s)
- testDeleteMkdirFetchInfoOpenInputOpenOutput (0,438 s)
- testGetChildIPath (0,359 s)
- testGetName (0,000 s)

Failure Trace

- eu.geclipse.core.reporting.ProblemException: Failed to establish a network connection
- at eu.geclipse.efs.gridftp.internal.GridFTPConnection.connect(GridFTPConnection.java:213)
- at eu.geclipse.efs.gridftp.internal.GridFTPConnection.getClient(GridFTPConnection.java:475)
- at eu.geclipse.efs.gridftp.internal.GridFTPConnection.exists(GridFTPConnection.java:797)
- at eu.geclipse.efs.gridftp.GridFile.fetchChildren(GridFile.java:496)

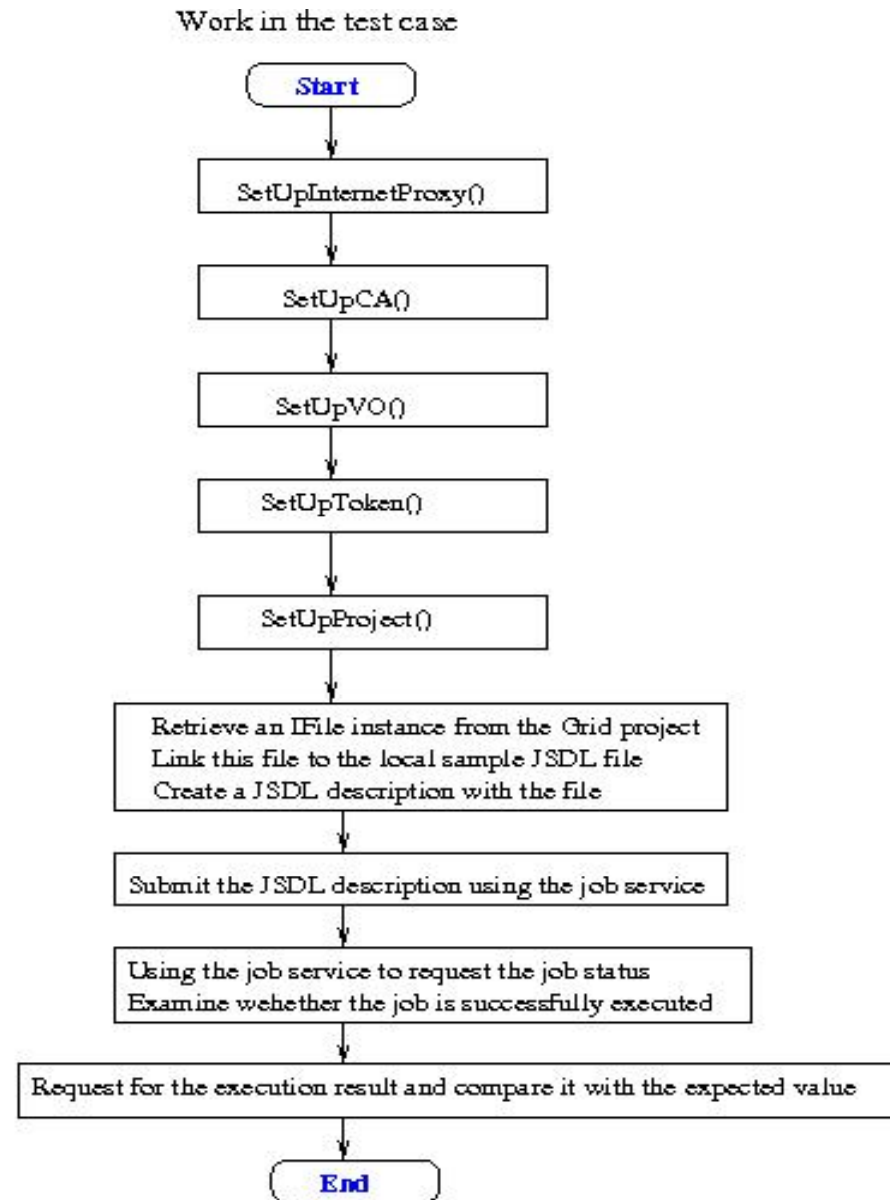
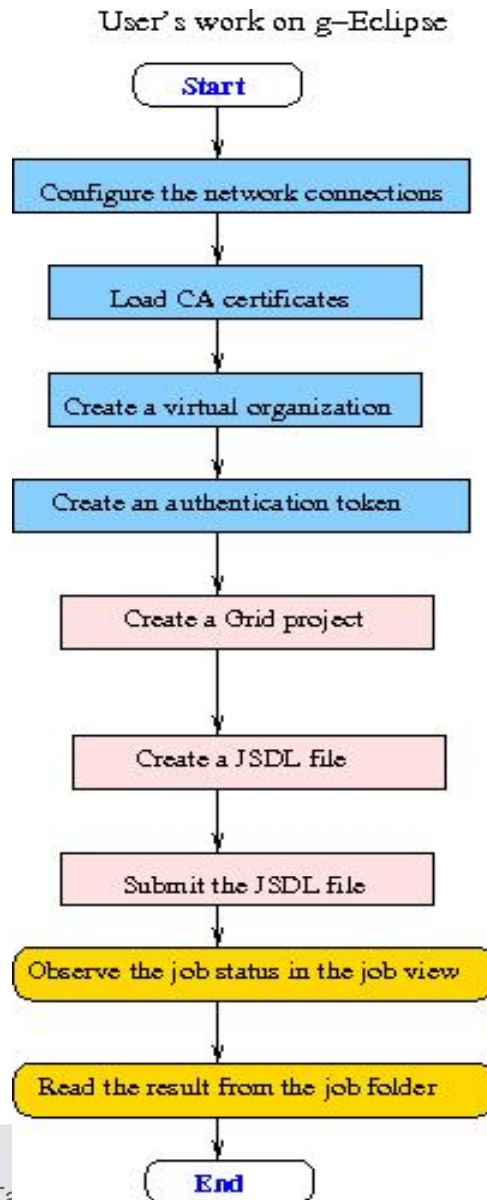
# Test Coverage



# Hierarchical Test Concept

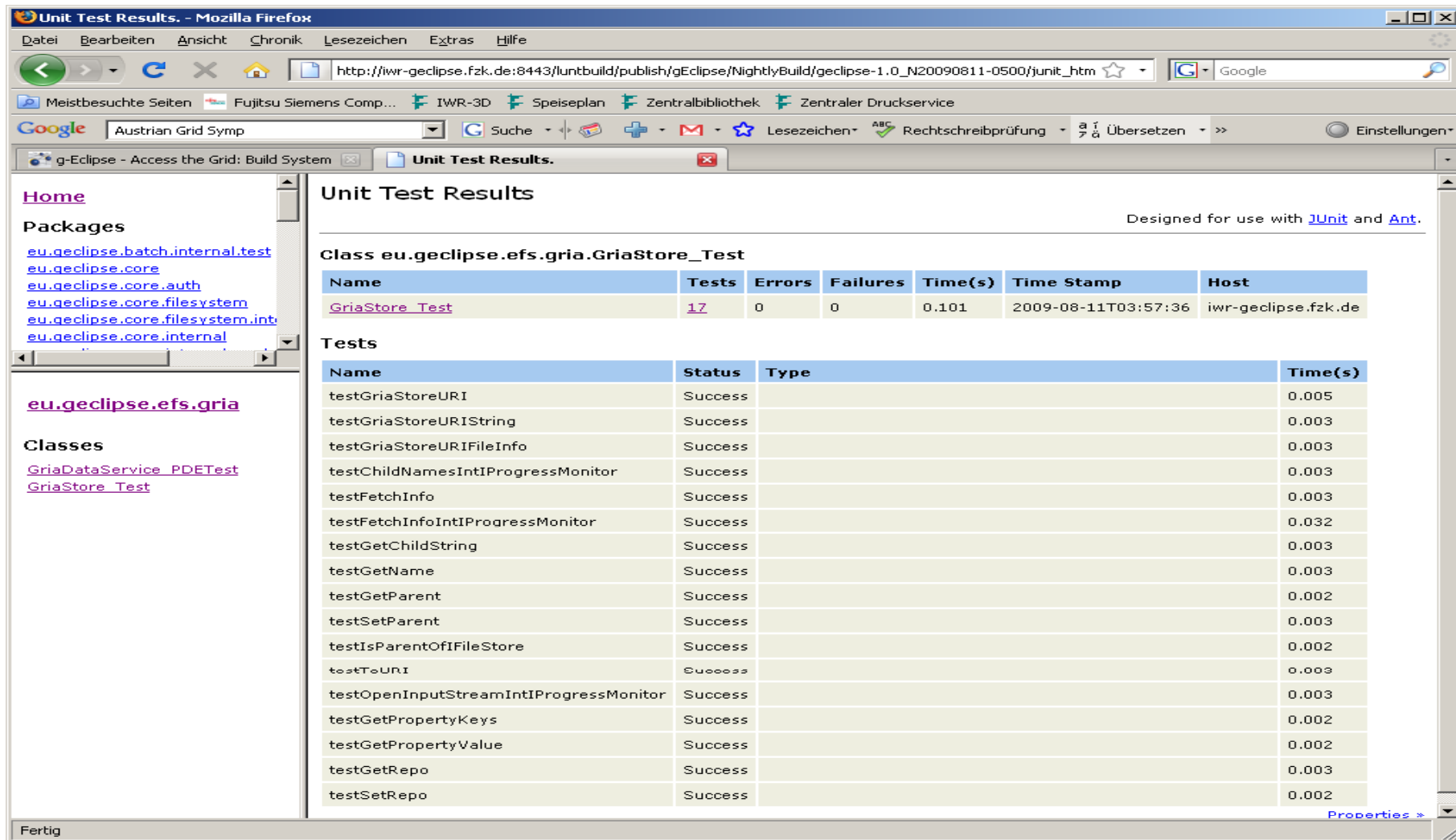
- Basis tests
  - Testing the functionality of individual methods
  - Could be simple Junit or PDE test
  - Need sometime a combined test with other methods
- Grid tests
  - Require initial setup of
    - token
    - VO
    - Internet proxy
    - CA
    - .....
  - Could not rely on wizards
  - Automation
- Integration tests
  - Using real workflow

# Sample Test Workflow





# Integration in the Nightly Build System (I)



Unit Test Results - Mozilla Firefox

http://iwr-geclipse.fzk.de:8443/luntbuild/publish/gEclipse/NightlyBuild/geclipse-1.0\_N20090811-0500/junit.htm

Unit Test Results

Designed for use with [JUnit](#) and [Ant](#).

Class eu.geclipse.efs.gria.GriaStore\_Test

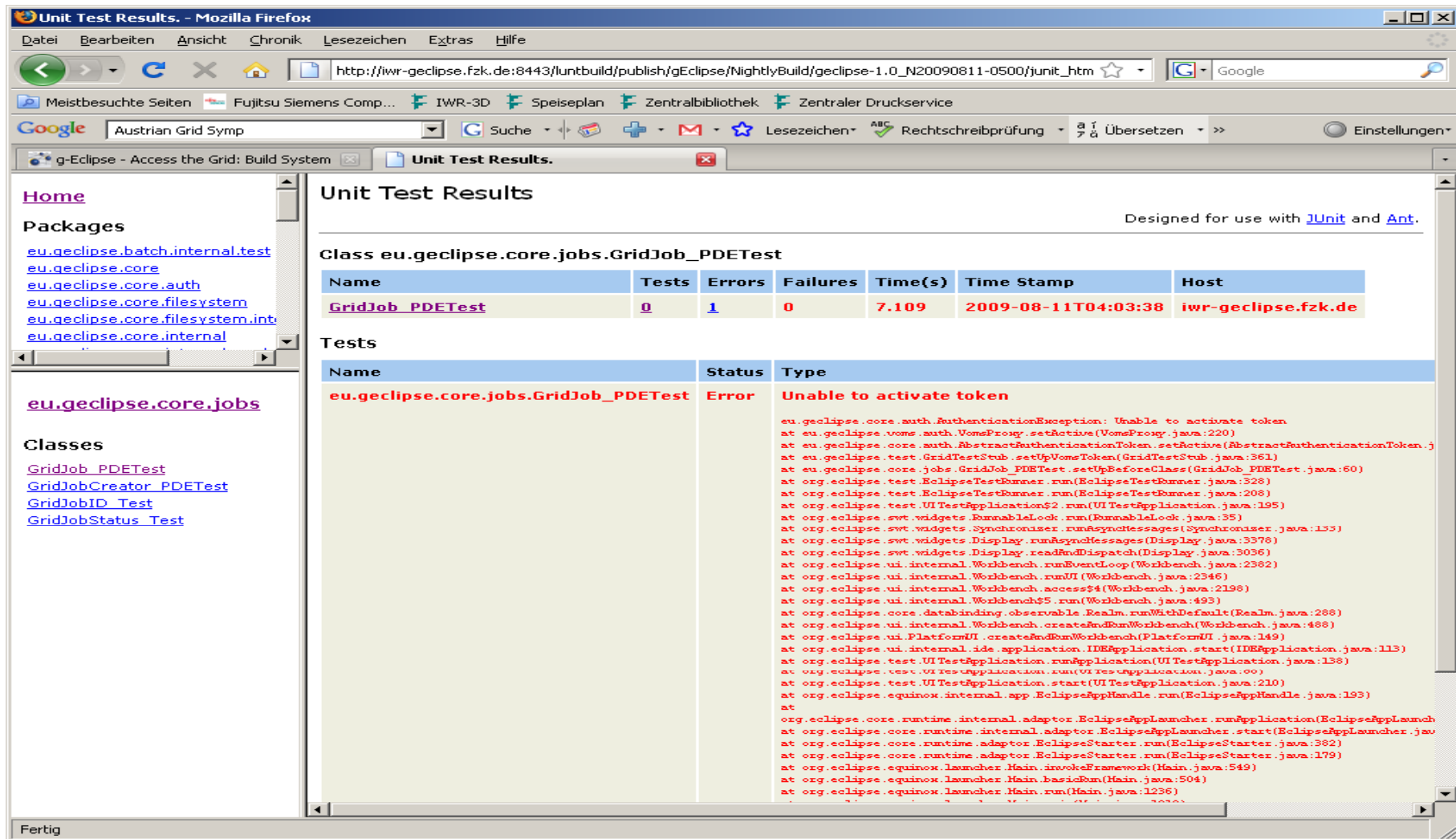
Name	Tests	Errors	Failures	Time(s)	Time Stamp	Host
GriaStore_Test	17	0	0	0.101	2009-08-11T03:57:36	iwr-geclipse.fzk.de

Tests

Name	Status	Type	Time(s)
testGriaStoreURI	Success		0.005
testGriaStoreURIStrng	Success		0.003
testGriaStoreURIFileInfo	Success		0.003
testChildNamesIntIPProgressMonitor	Success		0.003
testFetchInfo	Success		0.003
testFetchInfoIntIPProgressMonitor	Success		0.032
testGetChildString	Success		0.003
testGetName	Success		0.003
testGetParent	Success		0.002
testSetParent	Success		0.003
testIsParentOfIFileStore	Success		0.002
testToURI	Success		0.003
testOpenInputStreamIntIPProgressMonitor	Success		0.003
testGetPropertyKeys	Success		0.002
testGetPropertyValue	Success		0.002
testGetRepo	Success		0.003
testSetRepo	Success		0.002

Fertig

# Integration in the Nightly Build System (II)



Unit Test Results

Designed for use with [JUnit](#) and [Ant](#).

Class eu.geclipse.core.jobs.GridJob\_PDETest

Name	Tests	Errors	Failures	Time(s)	Time Stamp	Host
<a href="#">GridJob_PDETest</a>	0	1	0	7.109	2009-08-11T04:03:38	iwr-geclipse.fzk.de

Tests

Name	Status	Type
<a href="#">eu.geclipse.core.jobs.GridJob_PDETest</a>	Error	Unable to activate token

```
eu.geclipse.core.auth.AuthenticationException: Unable to activate token
at eu.geclipse.voms.auth.VomsProxy.setActive(VomsProxy.java:220)
at eu.geclipse.core.auth.AbstractAuthenticationToken.setActive(AbstractAuthenticationToken.java:361)
at eu.geclipse.test.GridTestStub.setUpVomsToken(GridTestStub.java:361)
at eu.geclipse.core.jobs.GridJob_PDETest.setUpBeforeClass(GridJob_PDETest.java:60)
at org.eclipse.test.EclipseTestRunner.run(EclipseTestRunner.java:328)
at org.eclipse.test.EclipseTestRunner.run(EclipseTestRunner.java:208)
at org.eclipse.test.UITestApplication$2.run(UITestApplication.java:195)
at org.eclipse.swt.widgets.RunnableLock.run(RunnableLock.java:35)
at org.eclipse.swt.widgets.Synchronizer.runAsyncMessages(Synchronizer.java:155)
at org.eclipse.swt.widgets.Display.runAsyncMessages(Display.java:3378)
at org.eclipse.swt.widgets.Display.readAndDispatch(Display.java:3036)
at org.eclipse.ui.internal.Workbench.runEventLoop(Workbench.java:2382)
at org.eclipse.ui.internal.Workbench.runUI(Workbench.java:2346)
at org.eclipse.ui.internal.Workbench.access$4(Workbench.java:2198)
at org.eclipse.ui.internal.Workbench$5.run(Workbench.java:493)
at org.eclipse.core.databinding.observable.Realm.runWithDefault(Realm.java:288)
at org.eclipse.ui.internal.Workbench.createAndRunWorkbench(Workbench.java:488)
at org.eclipse.ui.PlatformUI.createAndRunWorkbench(PlatformUI.java:149)
at org.eclipse.ui.internal.ide.application.IDEApplication.start(IDEApplication.java:113)
at org.eclipse.test.UITestApplication.runApplication(UITestApplication.java:138)
at org.eclipse.test.UITestApplication.run(UITestApplication.java:0)
at org.eclipse.test.UITestApplication.start(UITestApplication.java:210)
at org.eclipse.equinox.internal.app.EclipseAppHandle.run(EclipseAppHandle.java:193)
at
at org.eclipse.core.runtime.internal.adaptor.EclipseAppLauncher.runApplication(EclipseAppLauncher.java:179)
at org.eclipse.core.runtime.internal.adaptor.EclipseAppLauncher.start(EclipseAppLauncher.java:382)
at org.eclipse.core.runtime.adaptor.EclipseStarter.run(EclipseStarter.java:382)
at org.eclipse.core.runtime.adaptor.EclipseStarter.run(EclipseStarter.java:179)
at org.eclipse.equinox.launcher.Main.invokeFramework(Main.java:549)
at org.eclipse.equinox.launcher.Main.basicRun(Main.java:504)
at org.eclipse.equinox.launcher.Main.run(Main.java:1236)
```

# Conclusion

- g-Eclipse is a useful tool for simplifying the access to the grids
- Quality assurance is necessary for guaranteeing the functionality
- The approach
  - Coding quality evaluated with metrics
  - Unit tests
- Further work
  - Performance profiling
  - Site availability testing