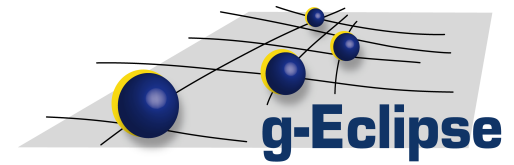


g-Eclipse - User friendly access to Grid and Cloud infrastructures

Ariel Garcia
KIT



Overview



■ Introduction

- The idea
- Some facts

■ Demonstrations

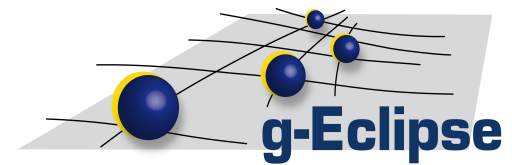
- The user perspective
 - Job management
 - Data management
 - Cloud computing on AWS
- The operator perspective
- The developer perspective

■ Developing with g-Eclipse

■ Outlook

- Ongoing activities and future plans

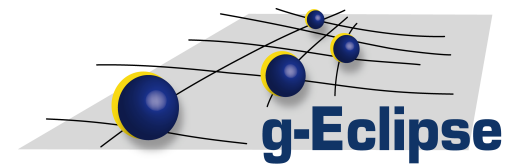
The idea



- Accessing a Grid is difficult
 - provide **user-friendly** UI for accessing Grids
- Many different middlewares are out there
 - provide **extensible middleware-independent framework** for accessing Grids → mw-independent UI!
- Currently supported middlewares:
 - **gLite** - Batch oriented Grid for the scientific user
 - **GRIA** - Service-oriented infrastructure for industry and commerce, b2b, SLA's...
 - **AWS** elastic compute cloud **EC2** and **S3** storage – Cloud computing



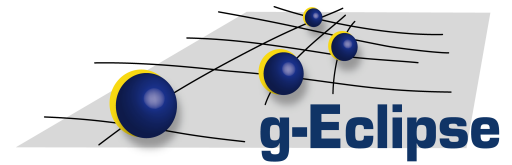
The g-Eclipse EU project



- Funded by European Commission, FP6
 - <http://www.geclipse.eu>
- 2 ½ years
- 8 partners in 5 countries
- 20+ developers
- Monthly release cycle
 - ~ 20 releases
- Final EU-project release in January '09

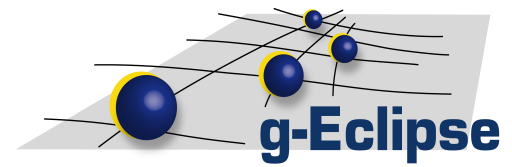


The open source project



- Eclipse Foundation project
 - <http://www.eclipse.org/geclipse>
 - Eclipse: Java IDE → whole framework, ecosystem
- Since end '06
- Open source
- Currently at 1.0 release track
 - **intellectual property** issues cleared (AWS support)
 - “official” 1.0 release soon

The g-Eclipse client



Support for

Grid User

- job management
- application workflows
- data management
- data visualisation

Grid Operator

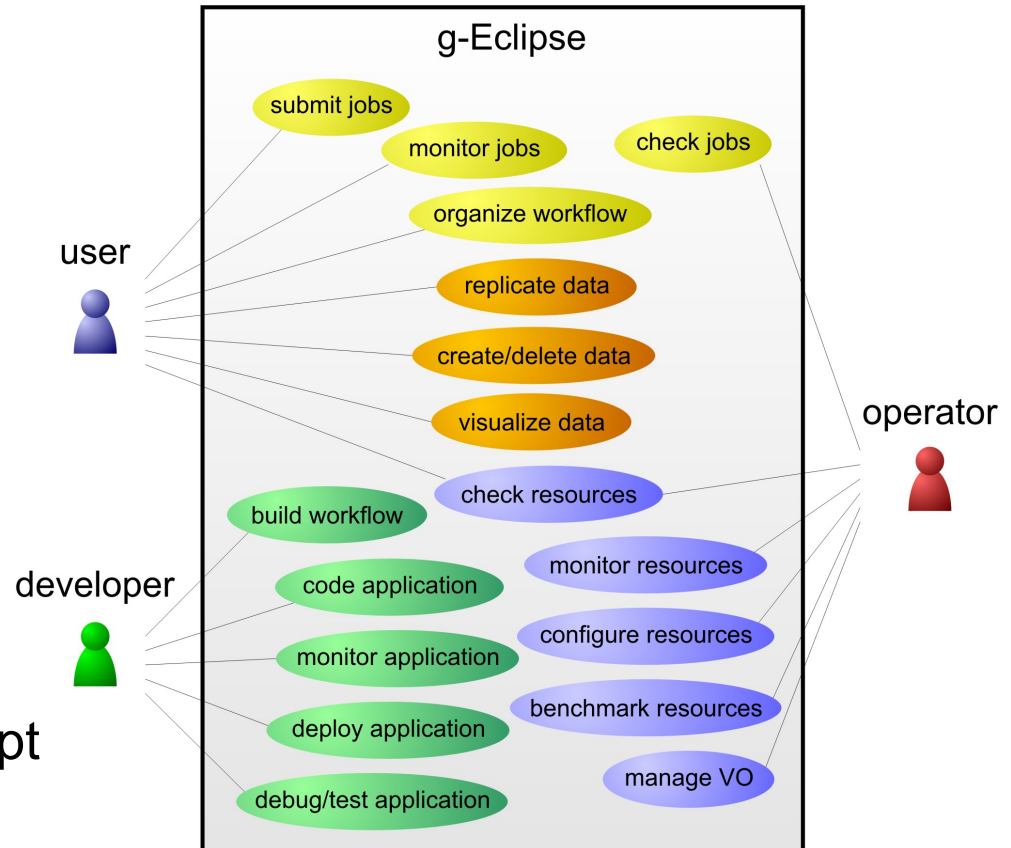
- site administration

Grid Developer

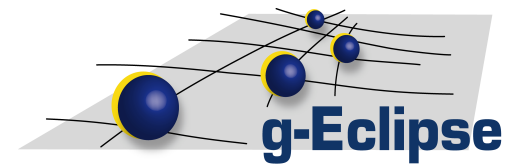
- compile/debug apps.
- deploy apps.

Eclipse's perspective concept

- Arrangement of “views”
 - view = panel



How does it look like?



Mounted File Systems →

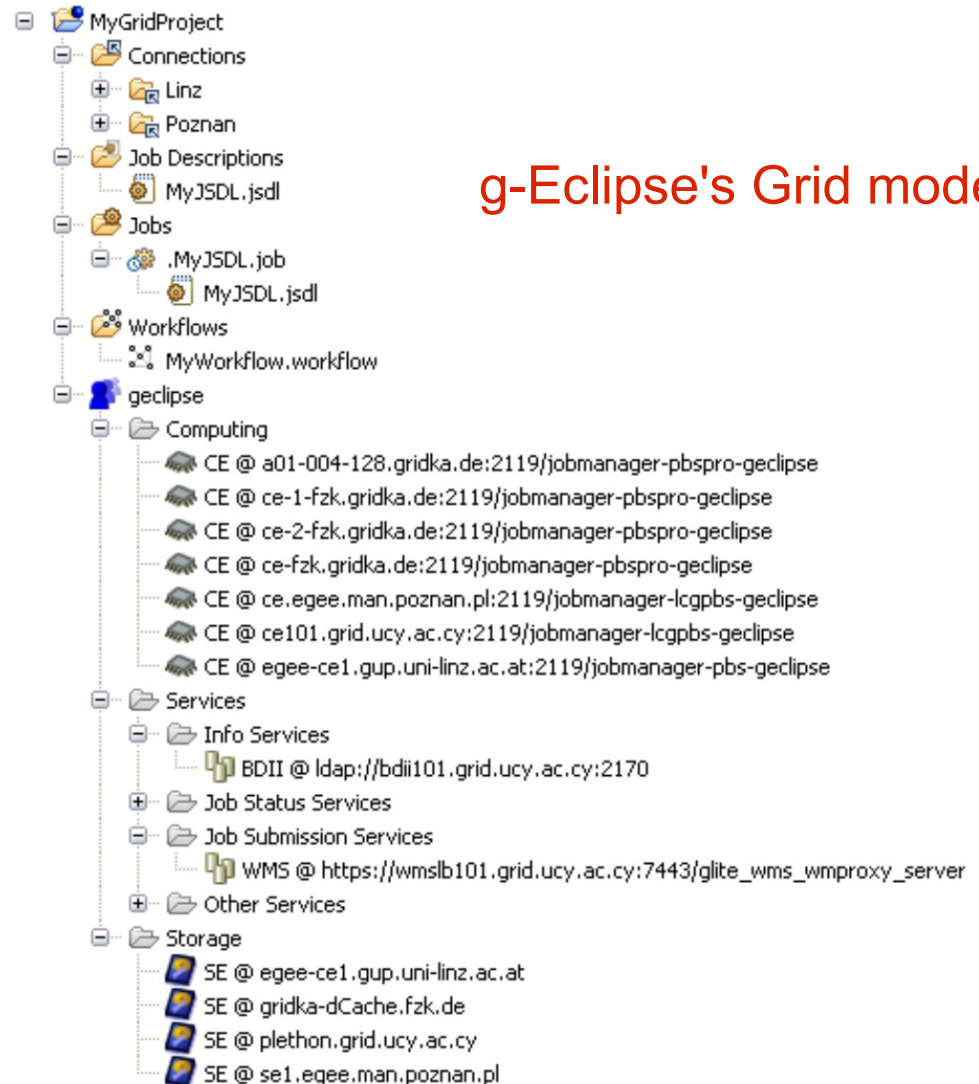
User's view of
the resources

Virtual Organisation →

Computing Elements →

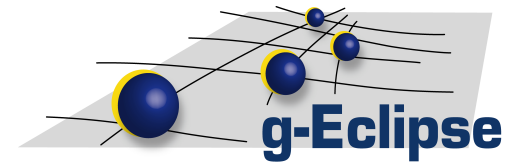
Services →

Storage Elements →



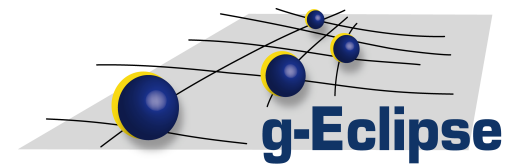
g-Eclipse's Grid model

The user perspective



- Data management
 - Files/folders create/save/copy/move/delete
 - GridFTP
 - SRM (WS standard)
 - LFC, own native Java implementation
 - GRIA data stagers
 - AWS S3
 - 3rd party transfers
 - Transfer manager
 - can restart unfinished transfers

The user perspective



■ Job management

- Job description creation and editing
 - JSDL standard compliant editor, full support
 - JDL supported (gLite)
- Job submission, status monitoring
 - WMS/Cream (gLite)
 - JobService (GRIA)
- Parametric jobs support

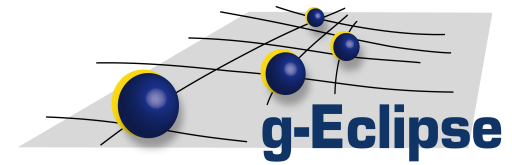
■ Workflows

- Dedicated workflow editor
- Submission and status, just like ordinary job!

■ Data visualisation

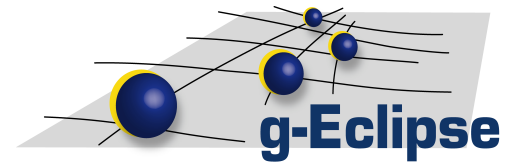
- Using VTK and SRS3D

Cloud computing



- User's point of view:
 - no big difference between Cloud, Grid
- Similar uses cases: users want to
 - process data (manage applications...)
 - manage data
 - view/monitor the available resources
- **Amazon Web Services**
 - EC2 – Elastic Compute Cloud
 - S3 – Simple Storage Service

The operator perspective

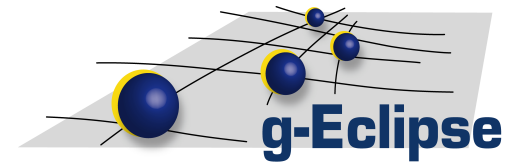


- Site administration
 - Queue management
 - PBS/Torque
 - Job management
 - Infrastructure monitoring
 - Infrastructure testing
 - Infrastructure benchmarking
 - Service level agreement editor
- User administration
 - VO management (in preparation)

The screenshot shows the operator perspective for a batch system named 'ce101.batch'. The interface is divided into several sections:

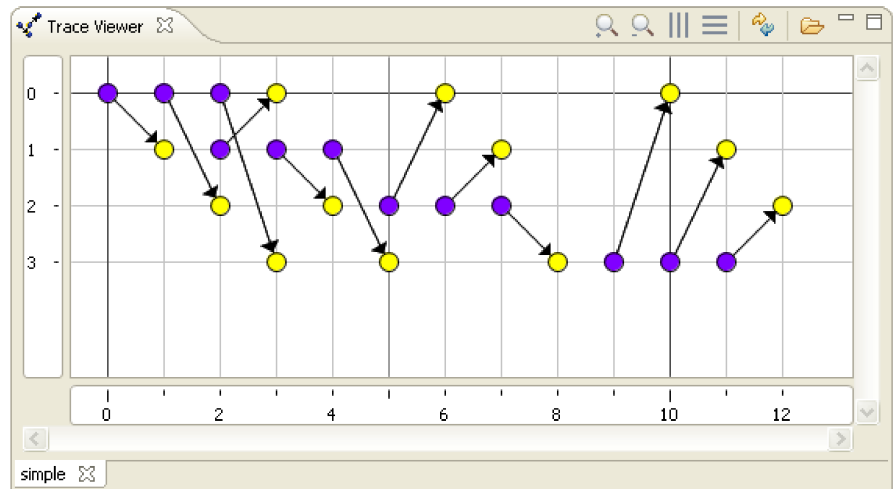
- Queues:** A grid of queue status boxes. Each box contains the queue name, a status indicator (e.g., 'enabled'), and a small icon. The queues shown are atlas, alice, biomed, cms, see, geclipse, dteam, and ops. All are currently 'enabled'.
- Summary:** A central box displaying system information: 'ce101.grid.ucy.ac.cy', 'Type: pbs', 'Num. of Queues: 8', 'Num. of WNs: 35', and 'Num. of Jobs: 11'.
- Nodes:** A grid of node status boxes. Each box contains a node ID (e.g., wn107) and a status indicator (e.g., 'free'). The nodes shown are wn107 through wn136. Most are 'free', but nodes wn132 through wn136 are marked as 'job-exclusive'.

The developer perspective

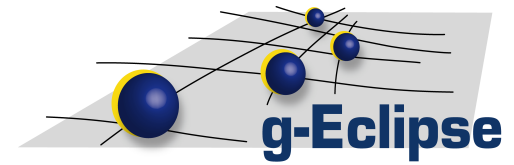


- Application development
 - Remote compiling
 - Remote debugging
 - normal Eclipse debugging perspective!
 - Analyzing MPI applications
 - traceviewer

- Application deployment

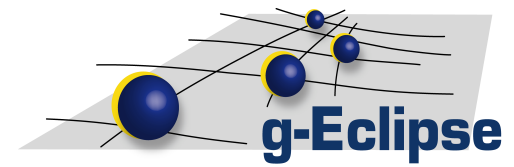


Overview



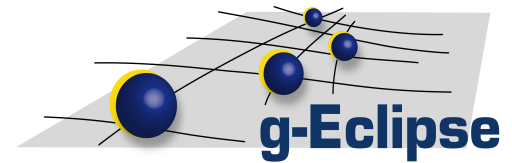
- Introduction
 - The idea
 - Some facts
- Demonstrations
 - The user perspective
 - Job management
 - Data management
 - Cloud computing on AWS
 - The operator perspective
 - The developer perspective
- **Developing with g-Eclipse**
- Outlook
 - Ongoing activities and future plans

g-Eclipse domains



- User interface / Grid client
 - graphical user interface for accessing Grid infrastructures
- Framework / API
 - collection of pure Java classes for developing client- and server-side applications for the Grid

Developing on g-Eclipse

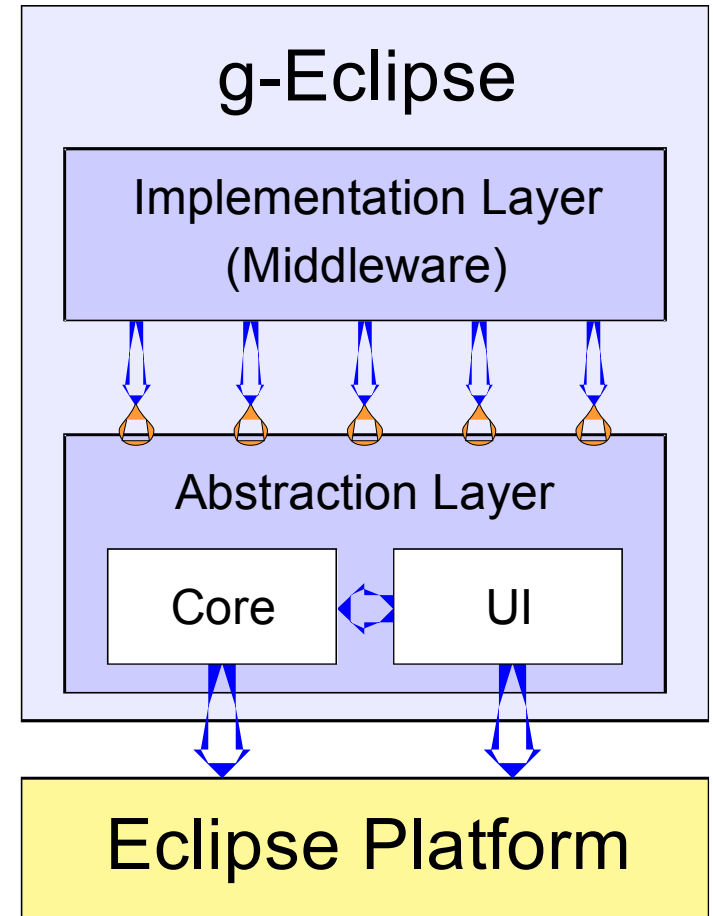


■ Abstraction layer

- core functionality, e.g.
 - authentication/authorization
 - VO management
 - data management
 - job submission
- common user interface, e.g.
 - views
 - wizards
 - dialogs
 - preference pages

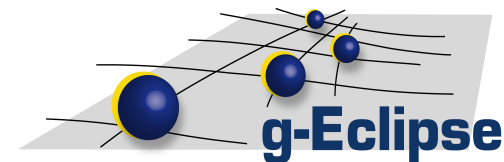
■ Implementation layer

- implementing core functionality
- middleware specific functionality

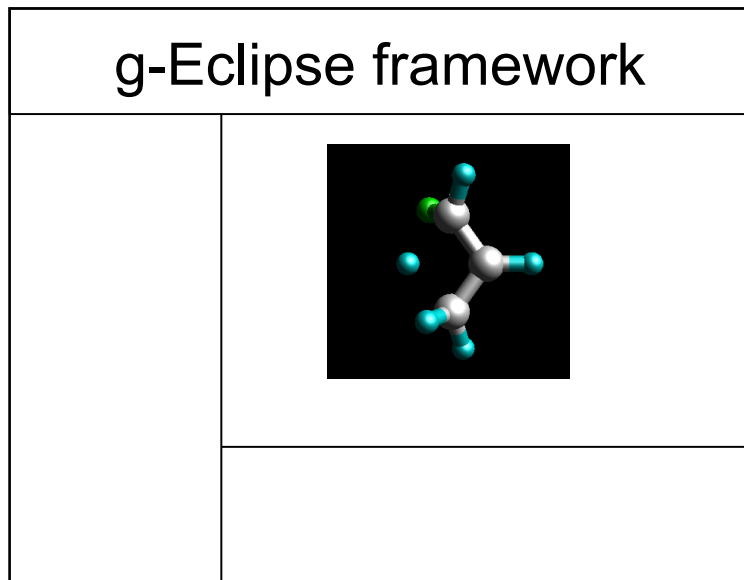


 Eclipse Extension Point

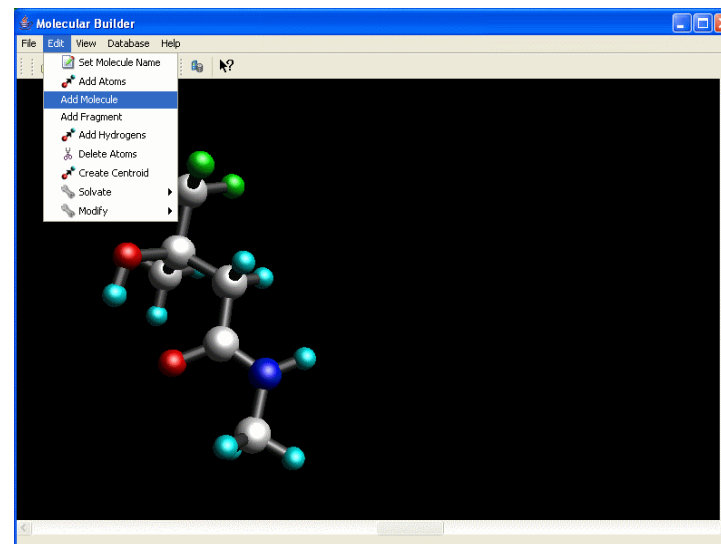
Applications with g-Eclipse



- Application inside g-Eclipse



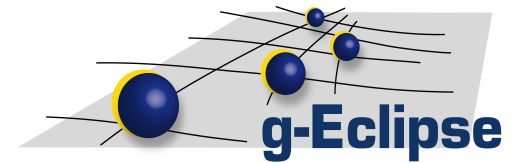
- Application on top of gEclipse



g-Eclipse API

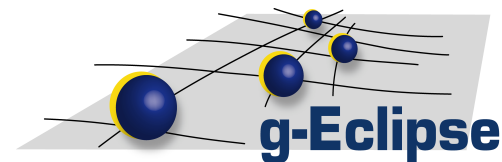
GRID

Application inside g-Eclipse



- Application is plugged into g-Eclipse framework
- Benefit from the user-friendly graphical interface for accessing Grid infrastructures
- Only few optional elements should be added:
 - editor for input files/parameters
 - submission support
 - data visualisation

Application inside g-Eclipse



The screenshot displays the g-Eclipse application window. The top menu bar includes 'File', 'Edit', 'Navigate', 'Search', 'Project', 'Run', 'Window', and 'Help'. The left sidebar shows a project tree with 'Gaussian' and 'Jobs' sections. The main window contains a text editor for 'CH2F.gjf' with the following content:

```
#T RHF/6-31G(d) Opt(TS,CalcFC, noeigentest) Freq Test  
  
CH2F-CH=CH2 <- -> CHF=CH-CH3 TS  
  
0 1  
C  
H, 1, R2  
C, 1, R3, 2, A3  
F, 3, R5, 1, A5, 2, D5, 0  
H, 3, R6, 1, A6, 5, D6, 0  
H, 4, R7, 1, A7, 2, D7, 0  
H, 4, R8, 1, A8, 7, D8, 0  
H, 8, R9, 4, A9, 1, D9, 0  
Variables:  
D7-1 075
```

Below the text editor is a toolbar with icons for file operations and a 'Save G03 input...' button. At the bottom, a 3D ball-and-stick model of a molecule is shown against a black background. Three red arrows point from text boxes to specific parts of the interface: one to the menu bar, one to the text editor, and one to the 3D model.

G-Eclipse framework

GJF text editor

3D visualisation

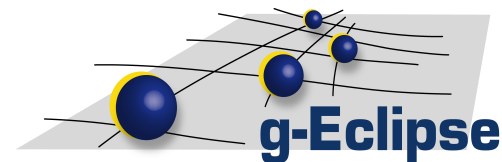
- Grid job description is created from GJF file on the fly

Application on top of g-Eclipse



- Enhance existing applications with Grid support
- Application has its own GUI and calls g-Eclipse API for accessing Grid resources.
- Application is started as Eclipse Rich Client Application
- Can access other bundles provided by g-Eclipse

Application on top of g-Eclipse

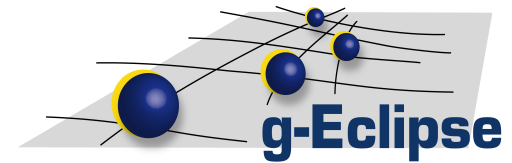


JMOL application

Data provided by G-Eclipse libraries

Action will be delegated to g-Eclipse libraries

Authn. and authz.



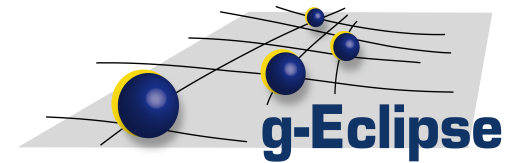
■ AAI support

- Globus proxies (X509)
- **VOMS proxies**
- GRIA keystores
- AWS tokens
- trusted certificates management

```
VomsProxyDescription desc = new VomsProxyDescription();  
desc.setVo( vo );  
desc.setCertFile( "/home/user/.globus/usercert.pem" );  
desc.setKeyFile( "/home/user/.globus/userkey.pem" );  
desc.setLifetime( 86400 );
```

```
AuthenticationTokenManager manager =  
    AuthenticationTokenManager.getManager();  
IAuthenticationToken token = manager.createToken( desc );  
token.validate();  
token.setActive( true );
```

VOs and jobs



■ VO support

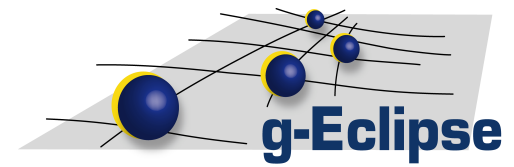
```
VomsVoCreator creator = new VomsVoCreator();
creator.setVoName( "geclipse" );
creator.setVoHost( "dgrid-voms.fzk.de" );
creator.setVoPort( 15009 );
creator.setVoHostDN(
    "/O=GermanGrid/OU=FZK/CN=host/dgrid-voms.fzk.de" );
creator.setVoInfoService(
    URI.create( "ldap://iwrbdii.fzk.de:2170" ) );

IVoManager manager = GridModel.getVoManager();
IVirtualOrganization vo
    = ( IVirtualOrganization ) manager.create( creator );
```

■ Job management

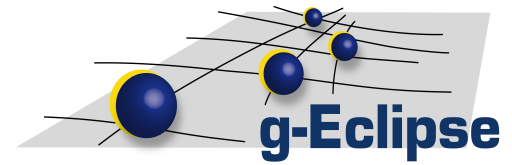
```
WMSClient wmsClient = WMSClient.getClient( wmsClientUri );
JobIdStructType jobId = wmsClient.submitJob( my_jsdl, null );
LBClient lbClient = LBClient.getLBClient( lbClientUri );
JobStatus status = lbClient.getJobStatus( jobId.getId() );
```

Overview



- Introduction
 - The idea
 - Some facts
- Demonstrations
 - The user perspective
 - Job management
 - Data management
 - Cloud computing on AWS
 - The operator perspective
 - The developer perspective
- Developing with g-Eclipse
- Outlook
 - Ongoing activities and future plans

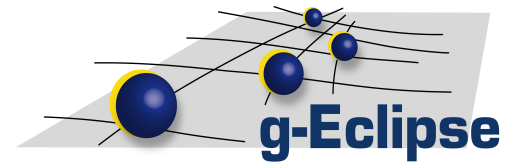
Outlook



- g-Eclipse **keeps going as an Eclipse project**
 - ongoing support
 - **open source**, anybody can join
 - continue gathering community, users & developers

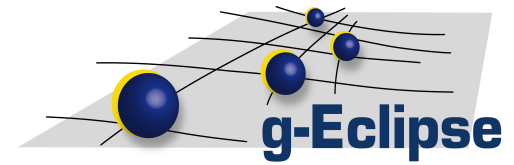
- We welcome reuse and new developments
 - **use of g-Eclipse as a library**
 - for RCP applications
 - for server-side services which need Grid access
 - ...
 - **new middleware implementations**
 - new components

Ongoing activities & future plans



- **GT4 implementation**
 - ongoing, initial MDS access available
- **Eucalyptus interoperability**
 - tests and bugfixes required
- **Lots of bugfixes... :-)**
 - usability improvements
 - performance/scalability improvements
- **Google AppEngine?**
- **See some effort in better integration with**
 - Eclipse runtime framework (Equinox, ECF, RAP, ...)?
 - Cloud services, XaaS?

The end



Thanks for listening!