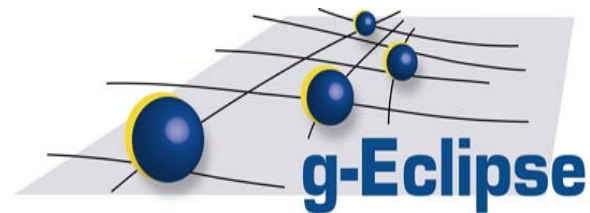




Enabling Grids for
E-science in Europe



g-Eclipse

A Framework for Accessing Grid Infrastructures



Nicholas Loulloudes
Trainer, University of Cyprus
(loulloudes.n_AT_cs.ucy.ac.cy)

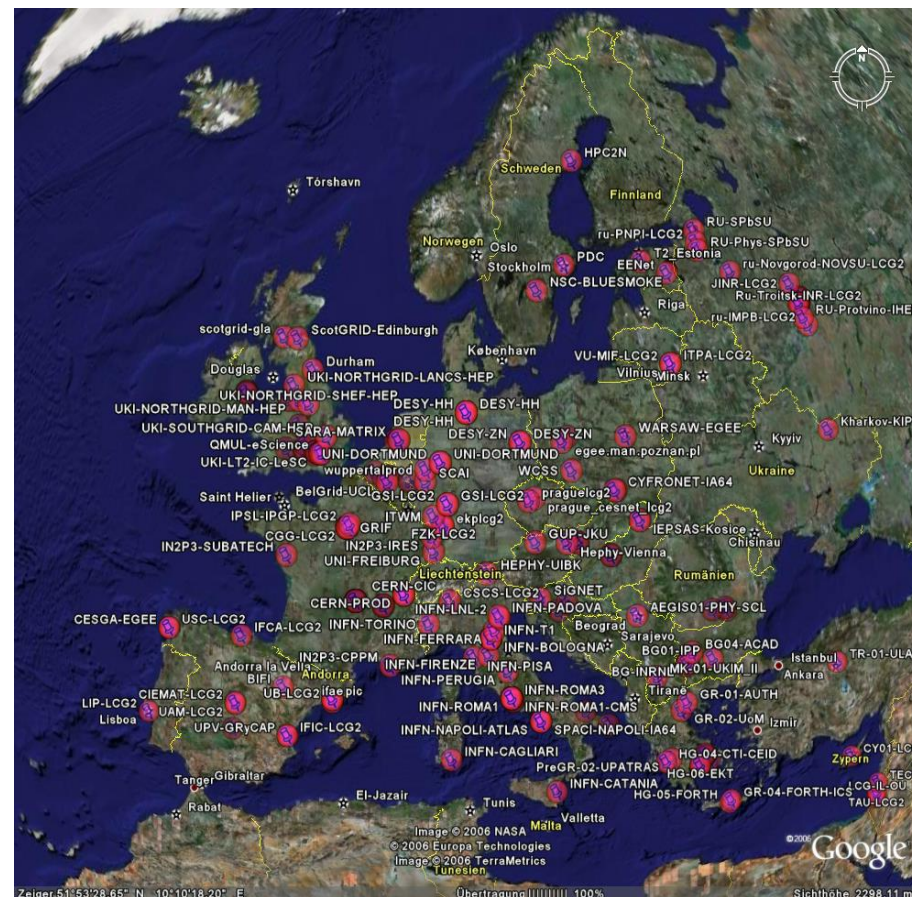
- Grid Reality – The Problem
- g-Eclipse - The Idea
- g-Eclipse Framework
- Technical Overview
- Abstracting Grid Middlewares
- gLite Support
- Conclusions

Grid Reality – The Problem

Grid Infrastructures: a huge collection of computational and storage resources.

The [EGEE](#)* infrastructure in Europe alone has available 24 / 7:

- 267 Sites (in 54 countries)
- ~114,000 CPUs
- 20 Petabytes of disk space.
- Supporting ~ 15 application domains

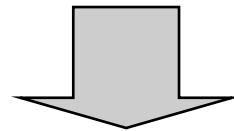


•Enabling Grids for E-science: www.eu-egee.org

Grid Reality – The Problem II

Grid Technology is **complex**:

- Different systems are used:
 - Middleware (gLite, GRIA, Globus, Unicore...)
 - Installation (rpm, tar, Quattor,...)
- Different Programming paradigms exist:
 - Batch Type systems Vs. Service Oriented Systems
 - Many programming Languages.
- Interaction with the Grid is mostly done via Command Line Interface (CLI).

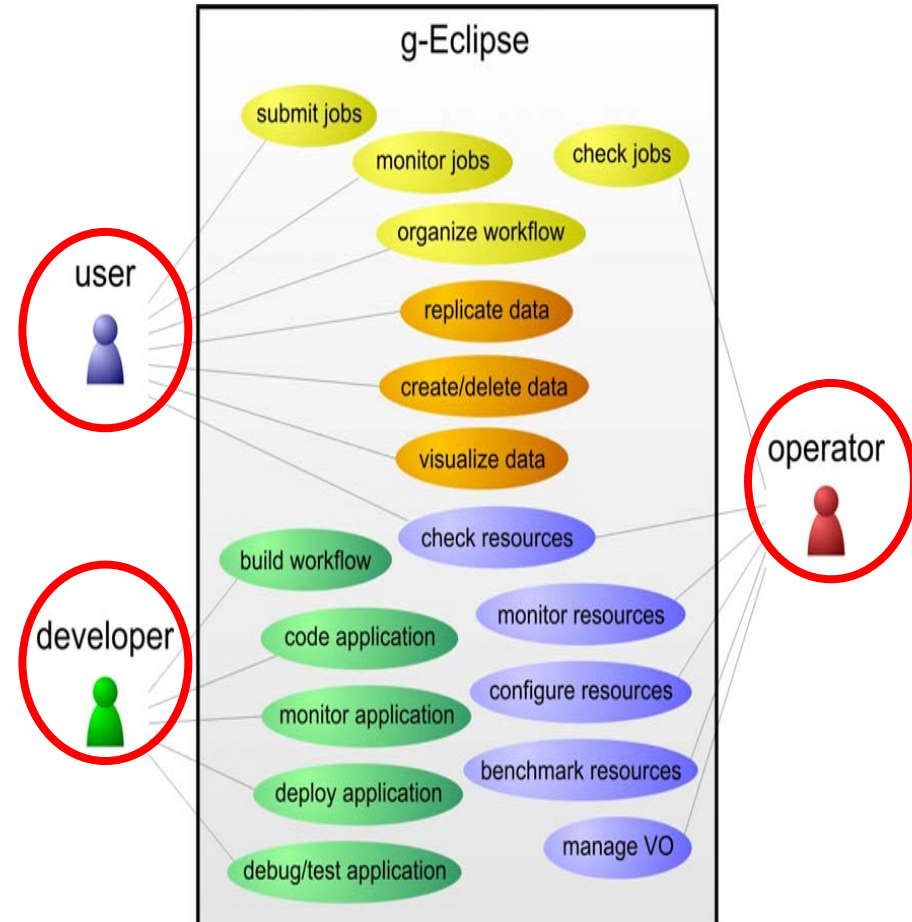


The Threshold is too “high” for the Standard user.

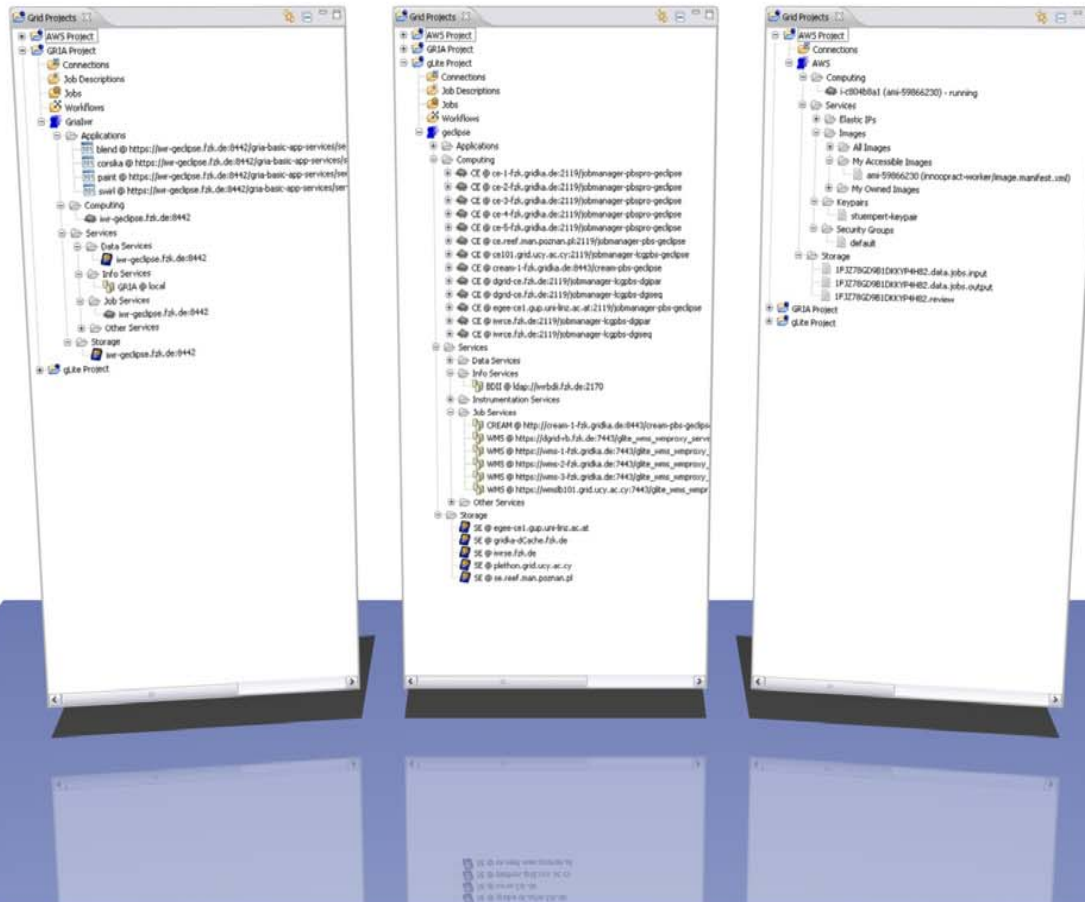
Users want easy access to Grids without knowing the details.

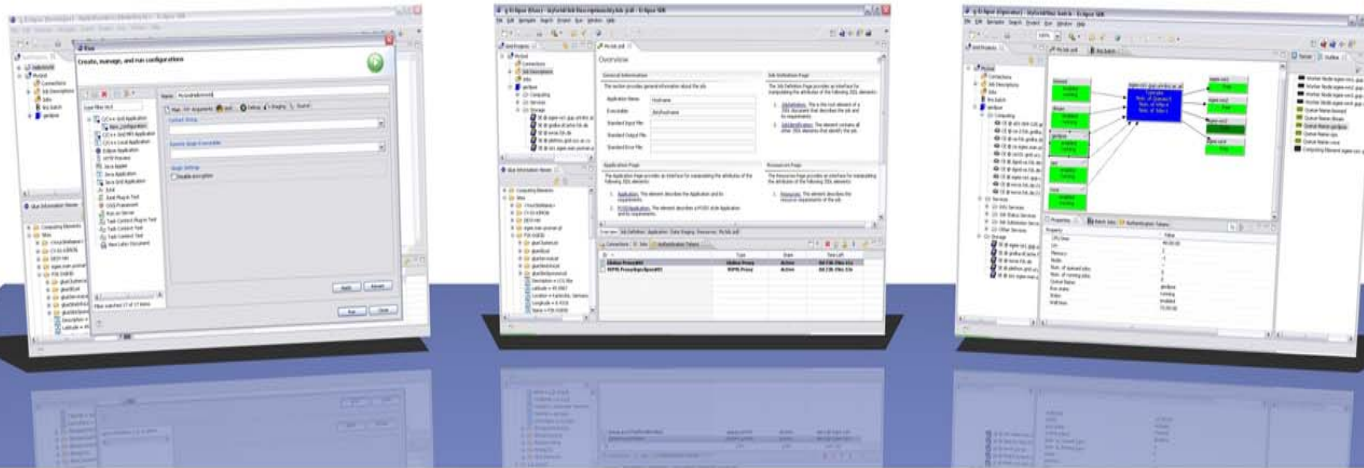
g-Eclipse – The Idea

- Provide a friendly UI for accessing Grids.
- Provide an extensible, middleware-independent, framework for accessing Grids.
- Support the roles of Grid **users**, **developers** and **operators**.
- Provide the necessary tooling to hide the complexity (**wizards, editors, views ...**).
- **Substitute** CLI with GUI.
- Conform to **Grid Standards**.



Single UI for all Middlewares





Perspective - An Eclipse term for the set of tools (views, editors wizards)

Grid Developer

- Code Applications
- Compile Apps.
- Debug Apps.
- etc.

Grid User

- Data Management
- Job Management
- Visualization
- etc.

Grid Site Operator

- Site Administration
- User Administration
- Job Management
- etc.

The g-Eclipse Project

Funded by the g-Eclipse (FP6) EU project

- Duration: July 2006 – December 2008
- Funding: 2 Million Euro
- Consortium members:

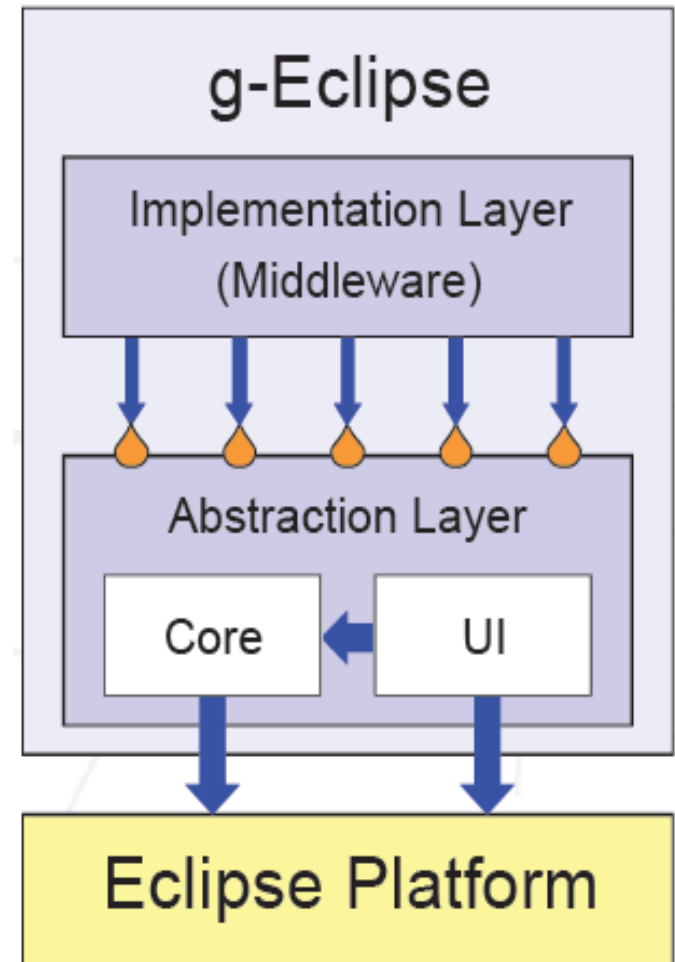




- **g-Eclipse – An official Eclipse Technology Project**
 - Around 20 developers among the 8 partners
 - 14 members with committer status at Eclipse.org
 - Community established and started to grow.
- **Source code released under Eclipse Public License**
 - Version 1.0 released in January 2009
 - Roughly 70 plug-ins, 4000 classes and interfaces aka 350.000 LOC
 - Available for free download
 - Supported platforms: Linux, Windows, Mac OS X

g-Eclipse - Technical Overview

- **Based on the Eclipse platform**
 - It is modular (OSGi) and extensible (Extension point mechanism of Eclipse).
- **Provide a middleware-independent architecture that:**
 - Abstracts common grid concepts.
 - Provides abstract core functionalities.
 - Is extensible by middleware-specific plug-ins.
- **Provide a graphical user interface that:**
 - Is based on the abstract core and therefore.
 - Looks and behaves (at least) the same for any middleware.



 Eclipse Extension Point

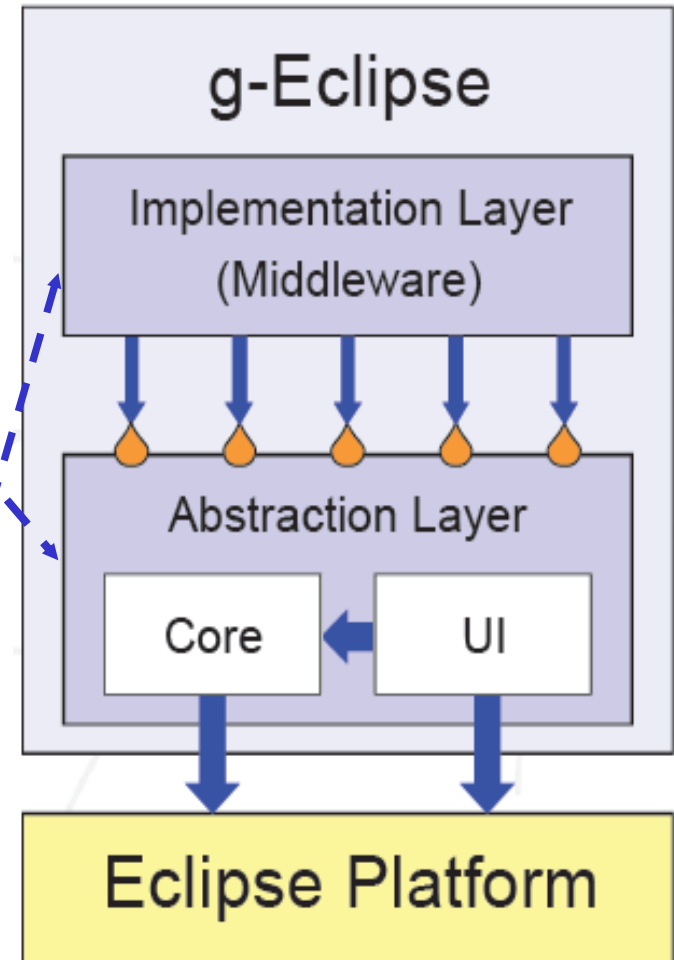
g-Eclipse - Technical Overview II

Abstraction Layer

- Core functionalities, e.g.
 - Grid Authentication / Authorization
 - Virtual Organization management
 - Data Management
 - Job submission
- Common User Interface
 - Views
 - Wizards
 - Editors
 - Dialogs
 - Preference Pages

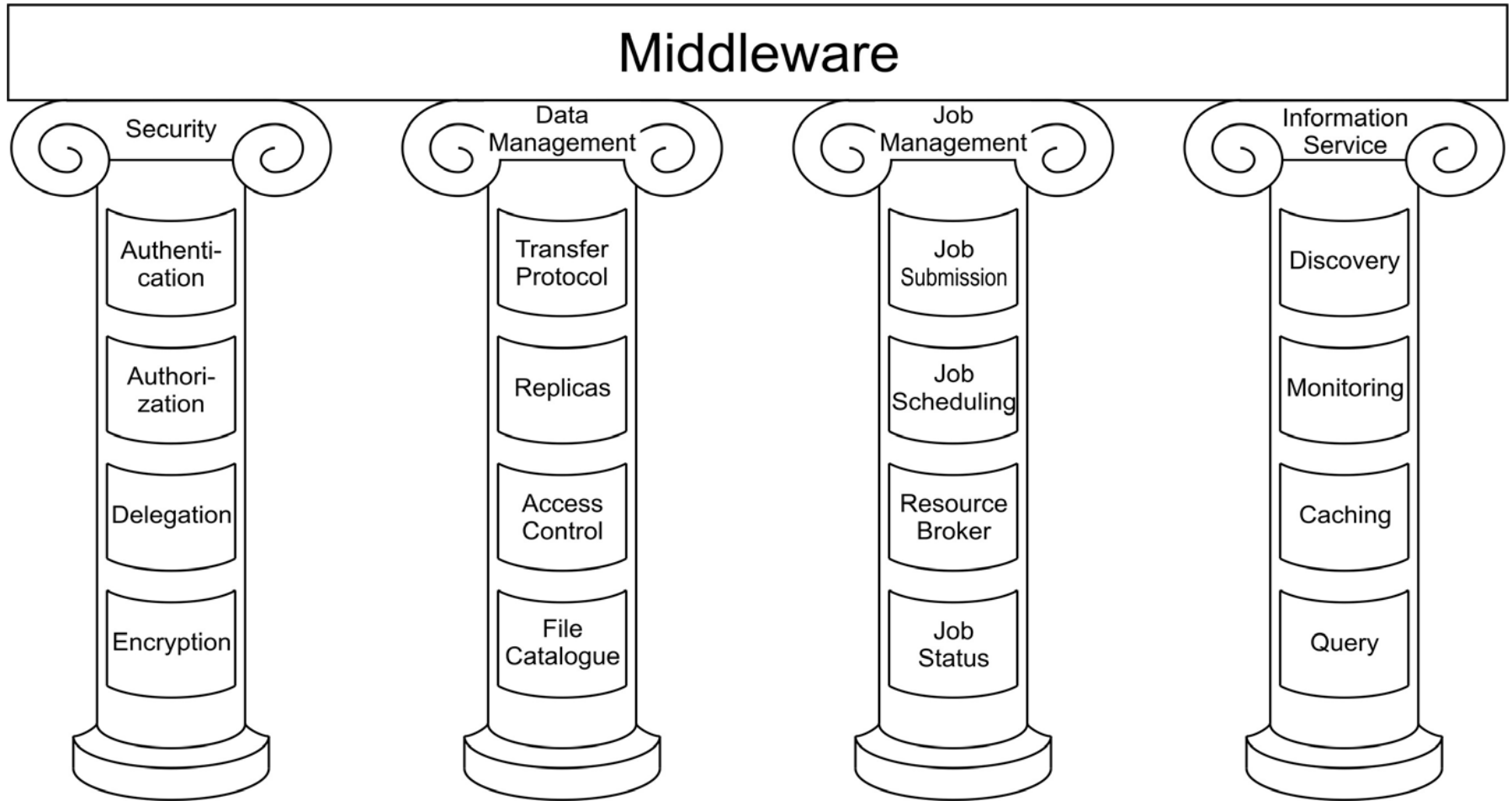
Implementation Layer

- Extended core functionalities
- Middleware specific functionalities

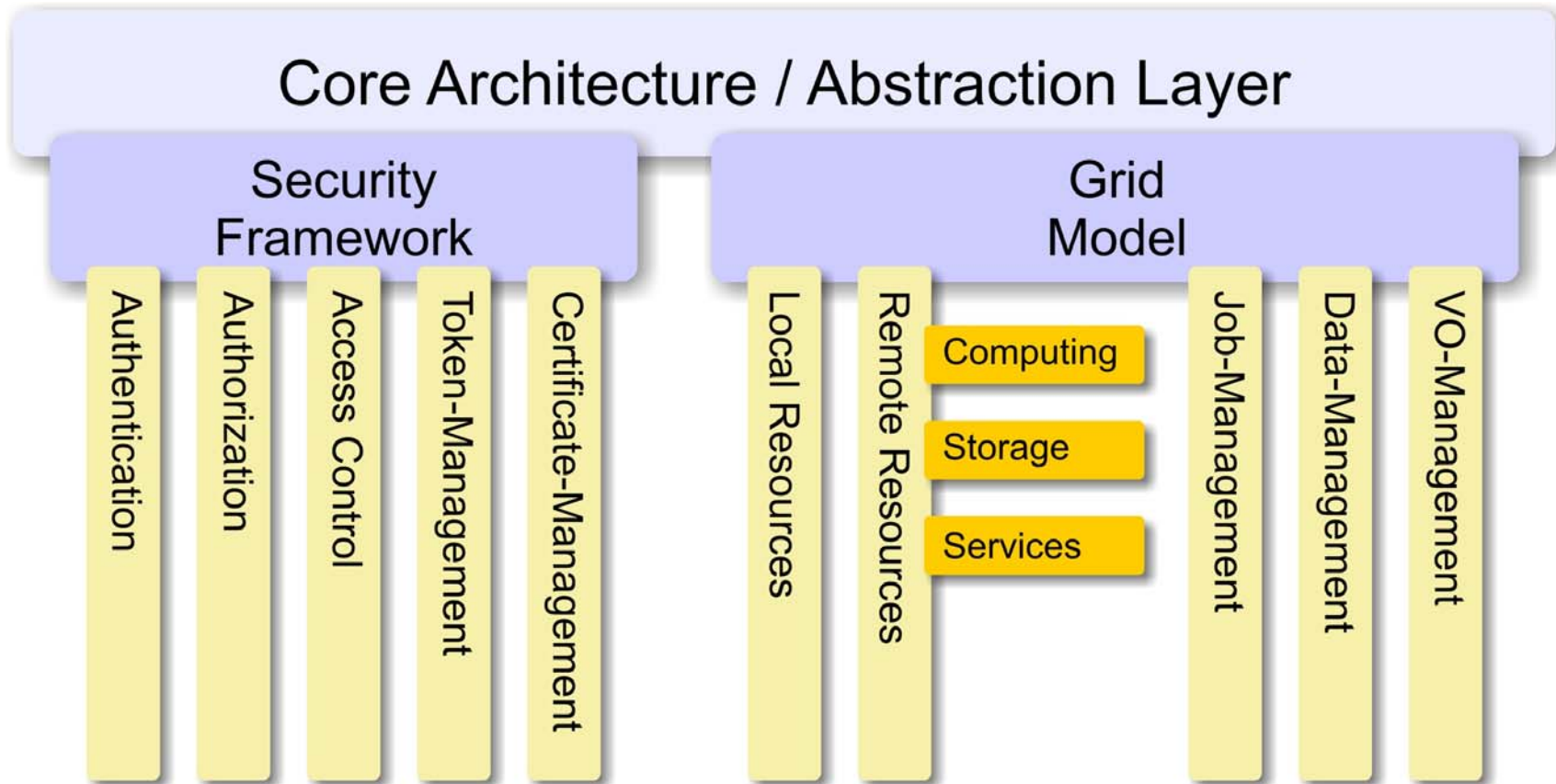


 Eclipse Extension Point

Four Pillars of Modern Grid Middleware



Abstracting Grid Middleware





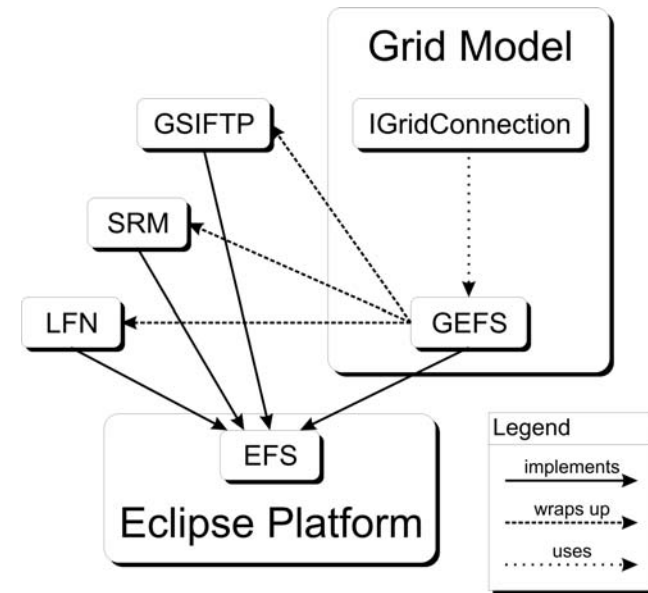
Supporting the gLite Middleware

- **AAI support**

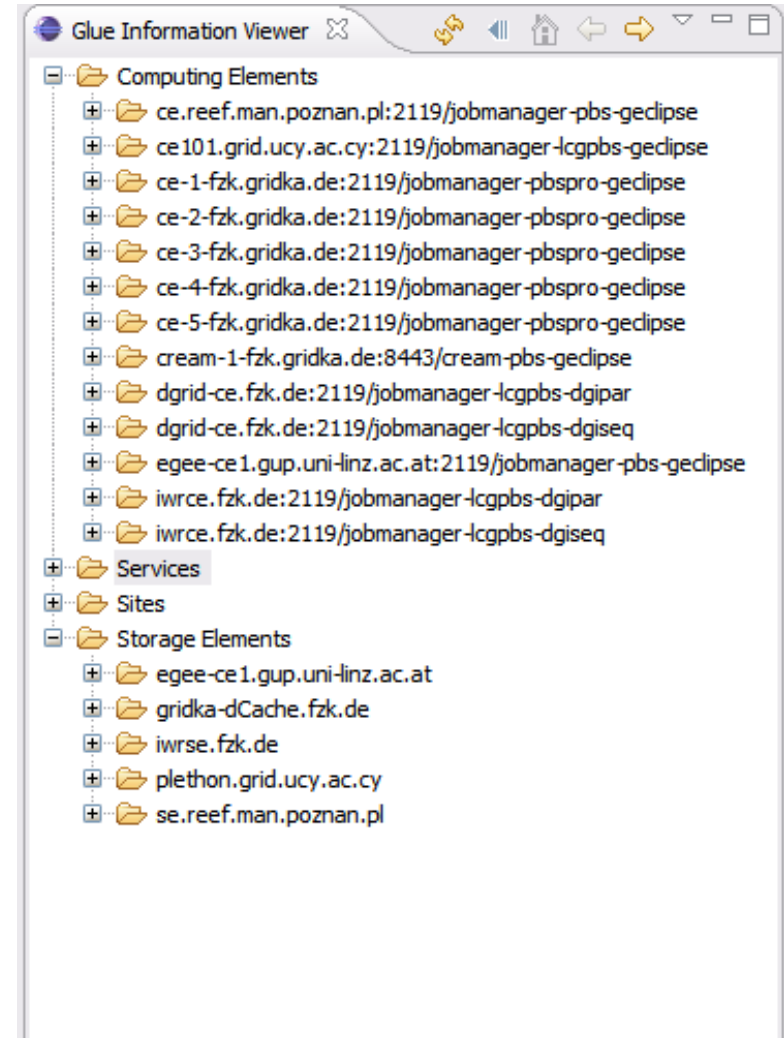
- Globus proxy creation and management
- VOMS proxy creation and management
- CA certificate management

- **Data Management**

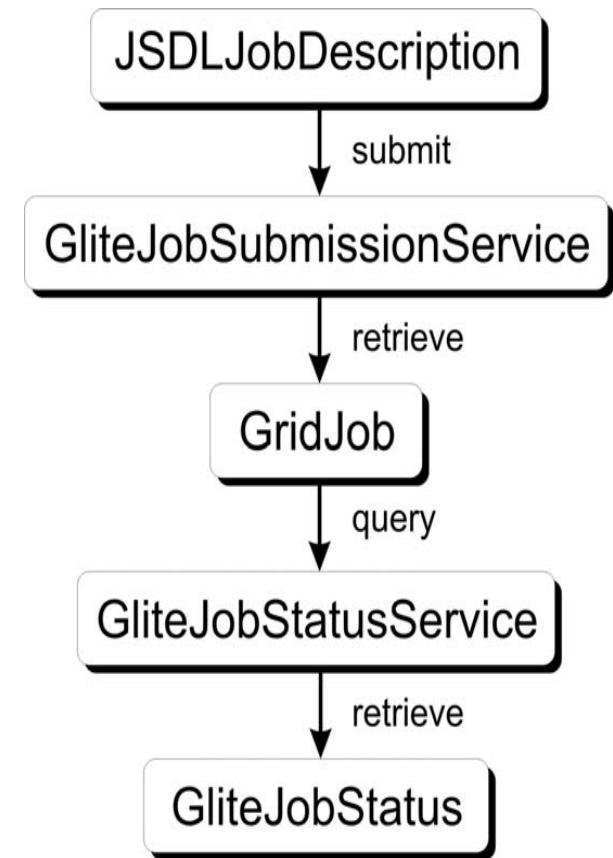
- GSIFTP
- SRM
- LFC
- Transfer manager (resume, restart, ... transfers)
- 3rd party transfers (server-2-server)



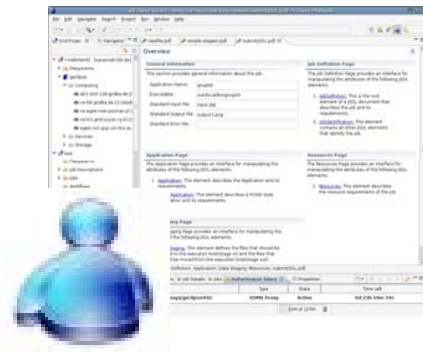
- **Information Services**
 - Support for BDII
 - Query available resources for a dedicated VO
 - Build the personalized Grid
 - Uses the GLUE schema to store information.
 - Provides methods to:
 - Fetch all available information or
 - Query the server side for specific information



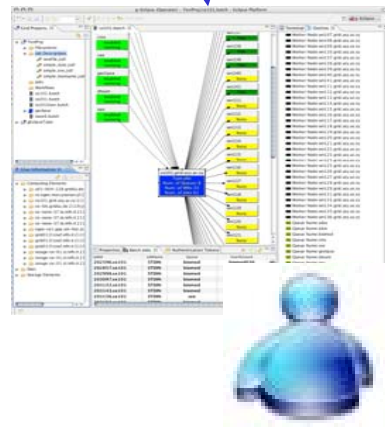
- **Job Description Languages**
 - JSDL and JDL supported
 - Parametric jobs supported
 - Workflows supported
- **Job Submission services**
 - WMS
 - Cream
- **Job Monitoring**
 - Logging & Bookeeping (L&B)



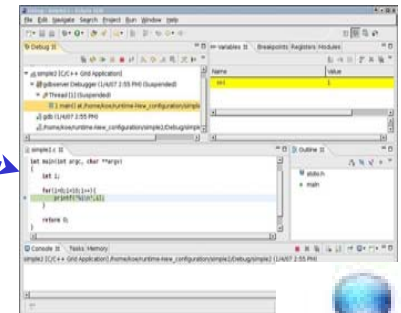
gLite - User Role Mappings



User
Perspective



Operator
Perspective



Developer
Perspective

The User Perspective

Grid Project
View

The screenshot shows the g-Eclipse IDE interface. The top menu bar includes File, Edit, Navigate, Search, Project, Run, Window, and Help. The main workspace is divided into several panes:

- Grid Project View:** Located on the left, it shows a tree structure of the project hierarchy, including 'HelloWorld', 'Filesystems', 'geclipse', 'Computing', 'Services', 'Storage', 'test', and 'Sites'.
- Information View:** Located at the bottom left, it displays 'Computing Elements' and 'Sites'.
- Authentication View:** Located at the bottom center, it shows a table of authentication tokens.
- JSDL Editor View:** The main central area, displaying the 'submitJSDL.jsdl' file with an 'Overview' section and various tabs for editing.

The 'Overview' section contains the following information:

- General Information:** Application Name: gnuplot, Executable: /usr/local/bin/gnuplot, Standard Input File: input.dat, Standard Output File: output1.png, Standard Error File: (empty).
- Application Page:** Lists 'Application' and 'POSIXApplication' elements.
- Data Staging Page:** Lists 'DataStaging' elements.
- Job Definition Page:** Lists 'JobDefinition' and 'JobIdentification' elements.
- Resources Page:** Lists 'Resources' elements.

The 'Authentication View' table shows:

ID	Type	State	Time Left
VOMS Proxy@geclipse#02	VOMS Proxy	Active	00 23h 59m 59s

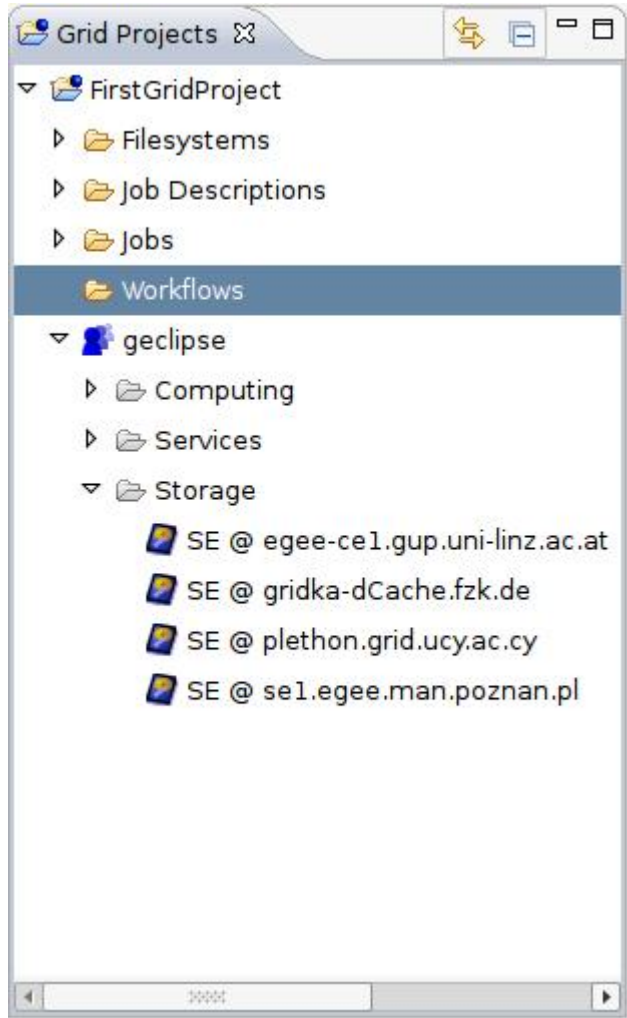
W
o
r
k
b
e
n
c
h

Information View

Authentication View

JSDL Editor View

The User Perspective II



- Everything revolves around the a **Grid Project**.
- A Grid Project is created as any other project in Eclipse.
- The Grid Project provides a view of:
 - Mounted Remote File Systems
 - Job Description
 - Submitted Jobs
 - Workflows
 - VO-specific Resource Browser.

The User Perspective III

The screenshot shows a web-based interface for editing JSDL files. The title bar indicates the file is 'newFile1.jsdl'. The main content area is divided into several sections:

- Overview**: A summary section with a help icon.
- General Information**: A form with fields for 'Application Name' (My First Grid Project), 'Executable' (/bin/hostname), 'Standard Input File', 'Standard Output File', and 'Standard Error File'.
- Job Definition Page**: A text area explaining the Job Definition Page and listing elements:
 - JobDefinition**: This is the root element of a JSDL document that describes the job and its requirements.
 - JobIdentification**: This element contains all other JSDL elements that identify the job.
- Application Page**: A text area explaining the Application Page and listing elements:
 - Application**: This element describes the Application and its requirements.
 - POSIXApplication**: This element describes a POSIX style Application and its requirements.
- Resources Page**: A text area explaining the Resources Page and listing elements:
 - Resources**: This element describes the resource requirements of the job.
- Data Staging Page**: A text area explaining the Data Staging Page and listing elements:
 - DataStaging**: This element defines the files that should be moved to the execution host(stage in) and the files that should be moved from the execution host(stage out).

At the bottom, there is a navigation bar with tabs for 'Overview', 'Job Definition', 'Application', 'Data Staging', 'Resources', and 'newFile1.jsdl'.

- The Job Description Wizard helps users to prepare a job description quickly.

- The Wizard can be enhanced with application defined pages

- Other job description languages can also be implemented (e.g. an RSL editor is available)

- The JSDL Editor allows users to edit JSDL files with a user-friendly, multipage, form-based editor

The Operator Perspective

- The **Operator Perspective** contains tools for Grid operator daily activities
 - Management of Computing Elements and Worker Nodes.
 - Queues management.
 - Virtual Organizations management.
 - Monitoring / Testing / Benchmarking various services.
 - CLI is also available.

The Operator Perspective II

The screenshot displays the Grid Engine Operator interface for a batch system. It is divided into several panes:

- Queues:** A grid of queue status indicators. Queues include 'ops', 'biomed', 'see', 'cms', 'geolife', 'steam', 'alice', and 'atlas'. Each queue has a green 'Enabled' box and a red 'Running' box.
- Nodes (sorted by state):** A grid of node status indicators. Nodes are identified by IDs like 'wn107' through 'wn135'. States include 'free' (green), 'job-exclusive' (yellow), and 'job-exclusive' (dark green).
- Terminal/Outline:** A list of nodes sorted by state. The selected node is 'Worker Node:wn112.grid.ucy.ac.cy'.
- Properties View:** A table showing properties for the selected node.

Property	Value
Kernel:	2.6.9-78.0.1.ELsmp
Name:	wn112.grid.ucy.ac.cy
Num. of CPUs:	2
RAM:	1033968kb
Running jobs	0
State:	free

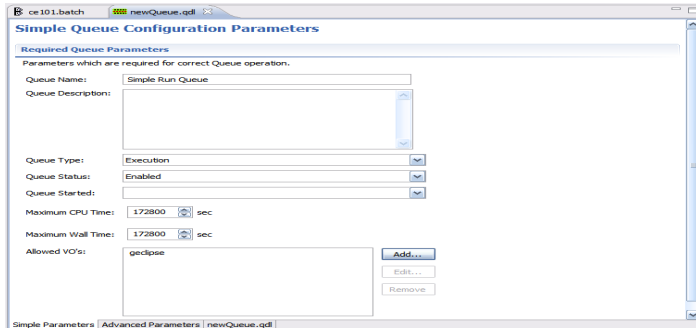
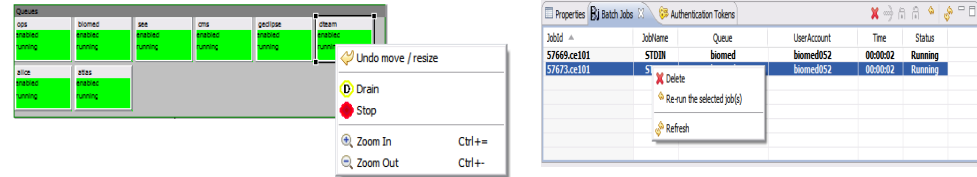
Batch Editor

Properties View

Outline View

The Operator Perspective III

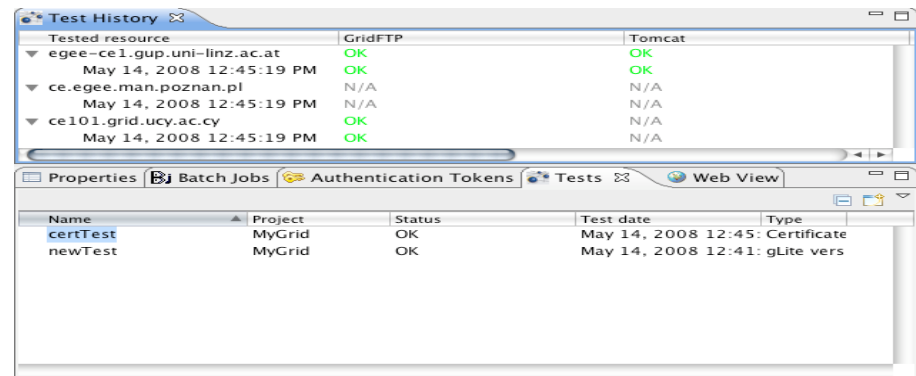
- **The Batch UI Framework**
 - Support for the **Portable Batch System (PBS)**
 - Start / Stop / Drain queues.
 - Hold / Un-hold / Move Jobs



- **Queue Configuration**
 - MPE for editing queue configuration files (QDL).
 - Wizard for applying QDL files to Batch services

- **Tests Framework**

- A Framework to **test physical** Grid Resources and **investigate** causes of failure.



The Developer Perspective

- The **Developer Perspective** contains tools for grid developer activities:
 - Develop C/C++ / Java applications
 - Debug applications locally or remotely on the Grid.
 - Application Deployment.
 - Application Monitoring.

The Developer Perspective II

The screenshot displays the Eclipse IDE interface for a C/C++ application. The main editor shows the source code of `simple.c`, which implements a simple MPI communication loop. The code is as follows:

```
// actual communication
int send=5;
int recv=0;
if (num_procs!=1) {
    for (j=0; j<num_procs; j++) {
        if (my_id==j) {
            for (i=0; i<num_procs; i++) {
                if (my_id!=i) {
                    MPI_Send(&send,1,MPI_INT,i,0,MPI_COMM_WORLD);
                } else {
                    MPI_Recv(&recv,1,MPI_INT,j,0,MPI_COMM_WORLD);
                }
            }
        }
    }
    printf("rank : %i value: %i\n", my_id, recv);
    MPI_Finalize();
    return 0;
}
```

The interface includes several key components:

- C / C++ Editor:** The central area where the source code is displayed and edited.
- C/C++ Debug View:** The top-right pane showing the current state of variables, such as `argc` (3), `argv` (0xbf8514e4), `my_id` (2), `num_procs` (4), `i` (0), `j` (2), and `buf` (0xbf851334).
- Process View:** The bottom-left pane showing a grid of process status indicators.
- Console View:** The bottom-center pane displaying the output of the application, showing the message `rank : 2 loop: 01234`.
- Trace Viewer:** The bottom-right pane showing a graph of the execution trace, with nodes representing process states and edges representing communication events.

Other Middleware support

- **GRIA (Grid Computing, Industry and Commerce)**
 - Fully integrated



- **Globus (Grid Computing, Scientific)**
 - GT2 partly integrated
 - GT4 integration initiated.

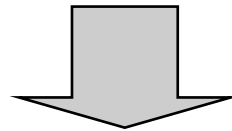


- **AWS (Cloud Computing, Amazon)**
 - S3 and EC2 integrated



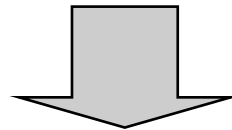
Other Middleware support

- Support of gLite, GRIA, Condor .



Prove of g-Eclipse Middleware Independence.

- With Amazon EC support:



Prove of g-Eclipse Grid Independence.

- g-Eclipse is and will stay an official Eclipse Project!
- Self-sustainability by the Eclipse Eco-system.
- Most developers are and will stay Committers on the Project!
- Contributions from other projects started (e.g DORII).

- **EVERYBODY** can contribute ...

YOU can contribute ...

- Report bugs or request features:
 - <http://bugs.eclipse.org>
- Contact the team:
 - geclipse-user@eclipse.org
 - geclipse-dev@eclipse.org
 - contact@geclipse.eu
- Contribute source code:
 - Bug fixes
 - Improvements
 - New features

!!! Thank you for your attention !!!

For more information visit:

<http://www.geclipse.eu>

<http://www.eclipse.org/geclipse>