CMST 13(2) 131-141 (2007)

DOI:10.12921/cmst.2007.13.02.131-141

COMPUTATIONAL METHODS IN SCIENCE AND TECHNOLOGY **13(2)**, 131-141 (2007)

Accessing Grid Computing Resources with g-Eclipse Platform

Paweł Wolniewicz[1], Norbert Meyer[1], Maciej Stroiński[1], Mathias Stuempert[2]
Harald Kornmayer[3], Martin Polak[4], Harald Gjermundrød[5]

[1]*Poznan Supercomputing and Networking Center*
ul. Noskowskiego 10, 61-704 Poznań, Poland
e-mail: {pawel.wolniewicz/ meyer/stroinsk}@man.poznan.pl
[2]*Forschungszentrum Karlsruhe*
Postfach 3640, 76021 Karlsruhe, Germany
e-mail: mathias.stuempert@iwr.fzk.de
[3]*NEC Laboratories Europe, IT Research Division*
Rathausallee 10, D-53757 St. Augustin, Germany
e-mail: harald.kornmayer@it.neclab.eu
[4]*Institute for Graphics and Parallel Computing (GUP) University Linz*
Altenbergerstraße 69, A-4040 Linz, Austria
e-mail: mpolak@gup.jku.at
[5]*University of Cyprus*
PO Box 20537, 75 Kallipoleos Str., 1678 Nicosia, Cyprus
e-mail: harald@cs.ucy.ac.cy

(Rec: December 7, 2007)

Abstract: In the paper we present how g-Eclipse can be used for easy running computation on Grid resources. The g-Eclipse project is an EU-founded project that aims to build an integrated workbench framework to access the power of existing Grid infrastructures. The g-Eclipse framework provides a general, integrated workbench toolset for Grid users, operators and developers. It is very useful for inexperienced users to interact with Grid resources independently of the underlying Grid middleware. The Grid abstraction enables Grid users to access the Grid in a desktop-like manner with wizards specified for common use cases.

Keywords: Grid, Eclipse, g-Eclipse, user interface, grid tool, middleware independent

I. INTRODUCTION

The growing computational requirements for modern applications are approching the limit where it very often cannot be fulfilled by a single organization. In Europe there are numerous Grid projects focusing on different applications, communities and technologies. The majority of the efforts were put on the preparation of working Grid infrastructure; implying simplicity and user friendliness were not the priority. As a result, Grids were not widely accepted, because the command line interface to the Grid required some knowledge about the underlying Grid mechanisms. It was not a problem in the beginning of the Grid era when most users of the Grid were involved in its creation and configuration. The Grid proved its usefulness for many application areas and the number of potential users is increasing rapidly. To support new, inexperienced users, user-friendly interfaces to the Grid are required.

The most popular Grid infrastructure in Europe is EGEE [1], which brings together scientists and engineers from more than 240 institutions in 45 countries world-wide to provide a seamless Grid infrastructure for e-Science that is available to scientists 24 hours-a-day. EGEE is using gLite middleware which provides a bleeding-edge, best-of-breed framework for building Grid applications tapping into the power of distributed computing and storage resources across the Internet. The most popular way of using gLite [2] middleware is to use command lines from the dedicated access machine. This way of using Grid is not simple and some effort was put to prepare the Genius Grid portal [3]. Other projects like GridLab [4], Deisa [5], UNICORE [6] also started with command line interfaces and in the later phase Grid portals were prepared.

Currently, portals tend to be the standard way of accessing Grid resources. However, usually they are targeted to the specific Grid infrastructure or even the specific Grid application. To simplify the process of preparing portals for specific usage (e.g. for specific scientific community), some portal development kits vere devised, like GridSphere [7]. Using the same portal technology has a positive effect that many portals have similar user interface and users do not need to learn it when they start to use new portal. Recently some effort was put to specify a standard for building the Grid portals.

g-Eclipse is more than a tool for accessing Grid and has much more functionality that those provided by Grid portals. g-Eclipse is an integrated Grid environment for Grid users, Grid operators, and Grid developers. In the later part of the paper we present the general overview of g-Eclipse and its architecture. We focus on g-Eclipse parts related to Grid computing, which is accessing Grid resources and developing Grid applications.

II. g-ECLIPSE OVERVIEW

The g-Eclipse is an integrated workbench framework to access the power of existing Grid infrastructures. The framework is built on top of the reliable eco-system of the Eclipse community to enable a sustainable development and to benefit from synergy effects arising from the use of the widely spread Eclipse [8] platform. The product allows to access Grid resources, to manage Grid resources and to support the development cycle of new Grid applications.

The g-Eclipse framework is based on a middleware-independent model and provides a graphical user interface build upon this model. By extensively using the Eclipse extension mechanism in combination with object-oriented design patterns, the framework can be easily extended by middleware-specific implementations. The result of this approach is a common user interface for all potential middleware in order to lower the threshold for scientific and industrial applications to be used on a Grid.

In its first project year, g-Eclipse provided the middleware-independent architecture and an exemplary implementation for the gLite middleware based on it. This implementation directly enables the use of already existing large Grid infrastructures such as EGEE or int.eu.grid. In the second project year another implementation of this architecture for the GRIA [9] middleware will be provided in order to prove the middleware-independent concept of the g-Eclipse model.

The g-Eclipse project directly supports three different user roles, Grid application users, Grid operators and Grid application developers. Grid application users are able to access the Grid with standardized but customizable user-friendly interfaces. Grid operators can reduce the cost of operation as the complexity of the Grid will be reduced with the supporting tools. Grid application developers are empowered to speed up the development cycle of new Grid applications. g-Eclipse tools are organized in so called perspectives, and the three provided perspectives correspond to the three Grid roles. Each perspective contains a predefined set of tools, but can be customised for specific users' needs.

In the context of Grid computing it seems probable that the majority of end-users will use g-Eclipse as a graphical tool within the Eclipse workbench. These users only get in direct contact with the UI contributions of g-Eclipse that hides the complexity of subjacent core features. Therefore, g-Eclipse provides a reliable and intuitive graphical interface that is conformant to the Eclipse user interface guidelines. The second group of end-users may use g-Eclipse as a framework in the sense of an API. They want to create their own applications based on the core functionalities rather than on the UI components. For g-Eclipse it means that we have to design a sophisticated, but at the same time manageable, API.

g-Eclipse is build on the Eclipse Platform, which is probably the most successful integration platform currently available. The development of Eclipse was started by IBM in the late 1990s and then handed over to the nonprofit Eclipse Foundation [8], to be managed as an open-source platform. Its design follows the standards set by the Object Management Group [10] that supports the interoperability between enterprise applications. The Eclipse platform is freely licensed and open source. The power of Eclipse lies in the common platform that it provides into which different multi-vendor tools can be integrated. Eclipse was designed for extensions from the very beginning and all Eclipse components and plug-ins are built for re-use. Anyone can write plug-ins for Eclipse and can have them work directly with any other plug-in for the platform. Eclipse's success is attributable to this capability and to the Eclipse open-source license, which allows developers to have easy and free-of-charge access to the source code. This will allow them to modify it and innovate quickly to meet user needs. Eclipse is also experiencing strong adoption in the research area as an ideal platform for research, as it allows the user to concentrate on the research subject, instead of creating the basic infrastructure. The rich set of open source extensions (>500 at the time of writing) available from the Eclipse community can provide an additional benefit to research projects. With all these features, Eclipse is a perfect base on which to build an integrated Grid environment.

The most popular usage of Eclipse is Java and C++ Integrated Development Environment. It is commonly used by programmers around the world. But Eclipse is used as integration platform also for many other products like personal information managers, stock exchange analysis and other. A lot of plug-ins are available for Eclipse and they can be installed individually depending on users' needs. Example plug-ins include collaboration tools, development support tools, mail clients and many other. Eclipse-based products are already used by many people, and installing g-Eclipse is just updating the current Eclipse with g-Eclipse plug-ins by pointing the Eclipse Update Manager to g-Eclipse update site. As g-Eclipse keeps the Eclipse style and look-and-feel, working with the Grid is easy and intuitive for people already familiar with Eclipse-based products.

III. General Architecture

In order to provide a common interface for accessing local and remote resources, the g-Eclipse framework offers a mechanism for accessing and managing both local elements, i.e. local files and folders, and Grid elements, i.e. either local or remote Grid resources. This abstraction layer is called the Grid Model. It is responsible for providing basic interfaces and classes that may be extended by specific middleware implementations. Moreover, it provides standard and abstract implementations for common types of resources, for instance, local files or standard containers that may contain other resources. This makes it easy for developers to build their specific implementations upon the Grid model. To add support for specific Grid middleware it is enough to implement some interfaces. The g-Eclipse architecture is presented in Fig. 1.

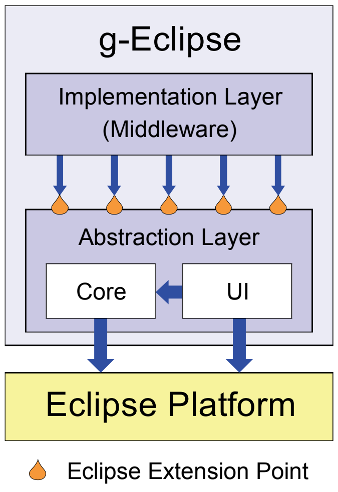
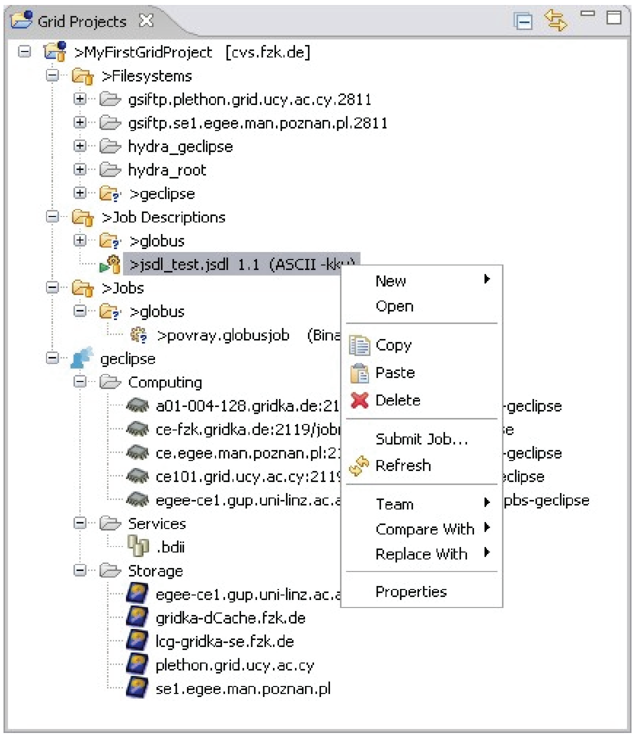


Fig. 1. g-Eclipse architecture

The central point for a user to access Grid resources from within g-Eclipse is the Grid project (see Fig. 2). Every time a user wants to work on the Grid, a Grid project has to be created. Each Grid Project has similar structure and groups all currently used resources. There can be user-defined resources like job descriptions, submitted jobs or open connections to remote file systems. The Grid project also allows for easy access to Grid services connected with this project, like resource brokers, information systems or storage systems. The set of available resources is determined by the Virtual Organization chosen for the project.

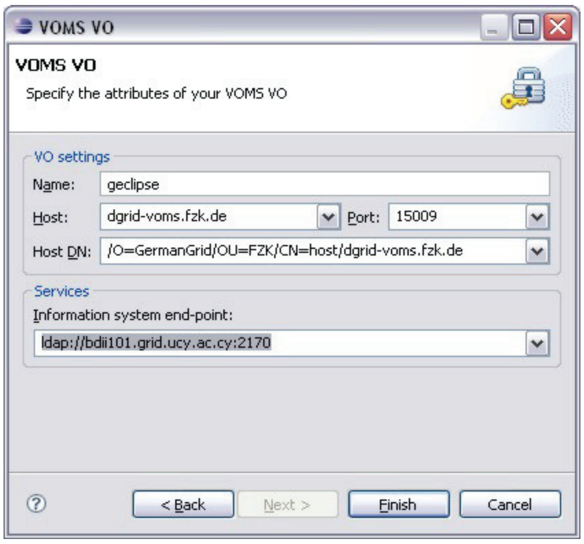
In the Grid world Virtual Organization is the central authorization point for accessing Grid resources. g-Eclipse uses an even more enhanced concept of VO which is not only the authorization point, but also the central information point providing information about all resources avail-



```
>MyFirstGridProject  [cvs.fzk.de]
  >Filesystems
    gsiftp.plethon.grid.ucy.ac.cy.2811
    gsiftp.se1.egee.man.poznan.pl.2811
    hydra_geclipse
    hydra_root
    >geclipse
  >Job Descriptions
    >globus
    >jsdl_test.jsdl  1.1  (ASCII -kkv)
  >Jobs
    >globus
      >povray.globusjob   (Bina
geclipse
  Computing
    a01-004-128.gridka.de:21          -geclipse
    ce-fzk.gridka.de:2119/jobi        e
    ce.egee.man.poznan.pl:2           -geclipse
    ce101.grid.ucy.ac.cy:2119         clipse
    egee-ce1.gup.uni-linz.ac.a        pbs-geclipse
  Services
    .bdii
  Storage
    egee-ce1.gup.uni-linz.ac.a
    gridka-dCache.fzk.de
    lcg-gridka-se.fzk.de
    plethon.grid.ucy.ac.cy
    se1.egee.man.poznan.pl
```

Fig. 2. Example contents of Grid Project View

able for this VO. g-Eclipse can support many Grid infrastructures by the set of middleware dependent plug-ins. VO implementation plug-in is one of the central plug-ins. It provides middleware-dependent authentication to the Grid using specific implementation of Abstract Authentication Tokens, and provides the list of available services. For simple Grid infrastructures which do not use the concept of VO we defined generic VO which does not provide user authentication. It provides only a manually added list of services. A sample Dialog defining VO is shown in Fig. 3.



VOMS VO

Specify the attributes of your VOMS VO

VO settings

Name: geclipse

Host: dgrid-voms.fzk.de Port: 15009

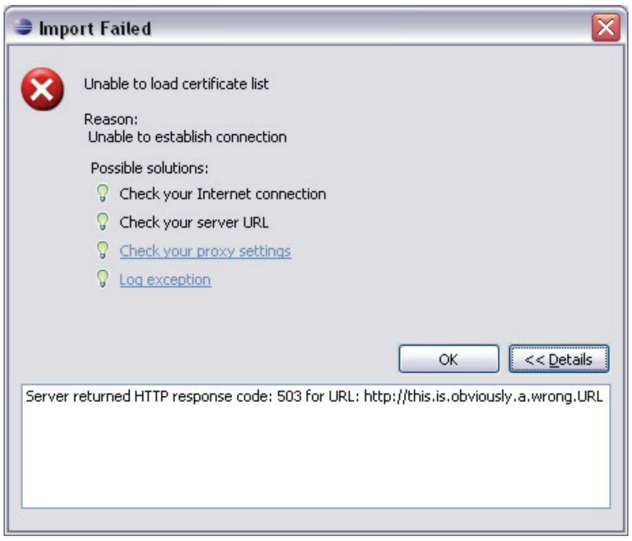
Host DN: /O=GermanGrid/OU=FZK/CN=host/dgrid-voms.fzk.de

Services

Information system end-point: ldap://bdii101.grid.ucy.ac.cy:2170

Fig. 3. VO Definition Dialog

The g-Eclipse architecture is designed to simplify the way of interacting with Grid resources. Users interact with Grid using user-friendly graphical wizards and editors. They should not need to deal with details like the syntax of the job description language, specific steps needed to do some complex actions or details of the authentication process. g-Eclipse prompts users if an action is necessary (e.g. authentication is required) and in case of errors presents possible hints and solutions for the problem, independently of reporting problem of the underlying middleware (see Fig. 4).


Import Failed

Unable to load certificate list

Reason:
Unable to establish connection

Possible solutions:
- Check your Internet connection
- Check your server URL
- Check your proxy settings
- Log exception

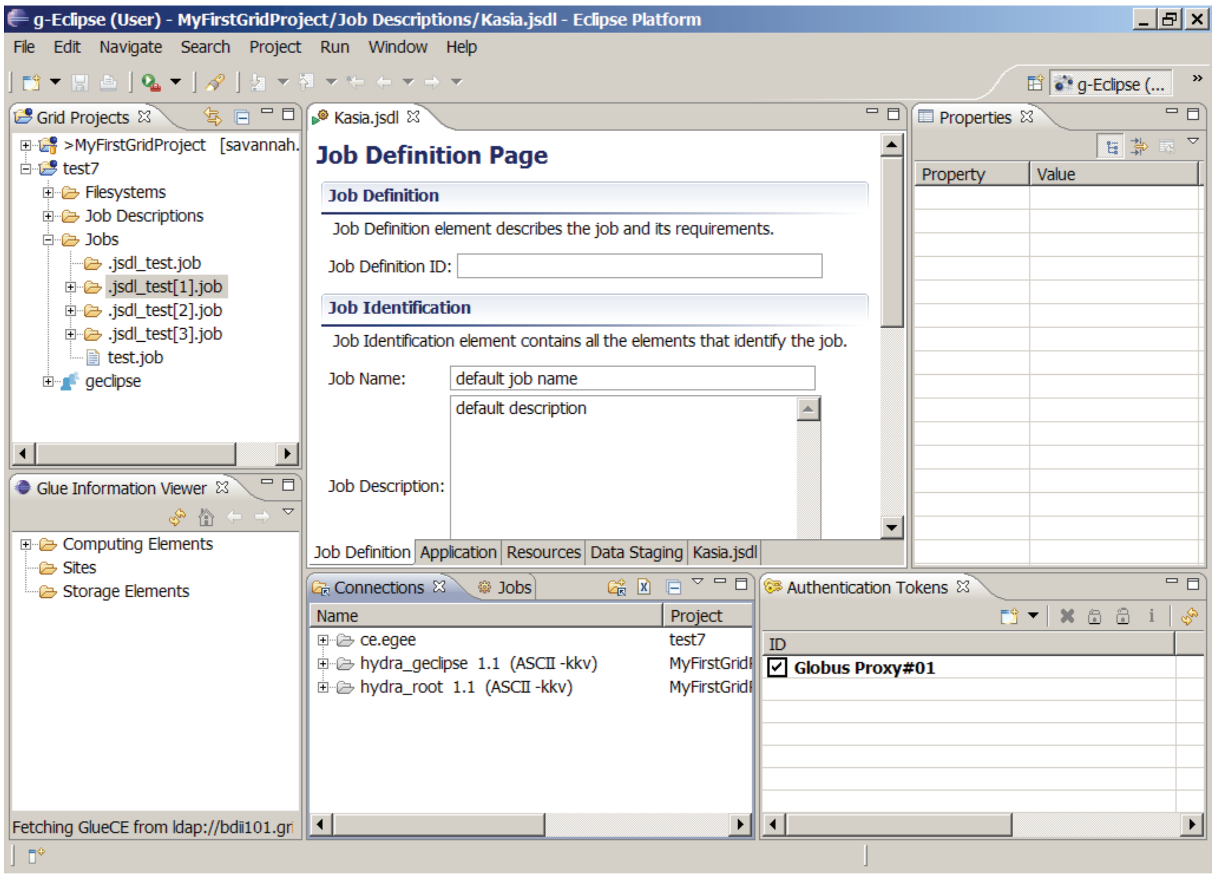
```
Server returned HTTP response code: 503 for URL: http://this.is.obviously.a.wrong.URL
```

Fig. 4. Example Error dialog

IV. GRID-USER PERSPECTIVE

The Grid User Perspective (see Fig. 5) is one of the g-Eclipse perspectives that groups tools for the specific needs of Grid users. It offers a simple way for Grid users to interact with Grid resources. The most common interactions of a Grid user are the submission and monitoring of computational jobs and the management of data within a Grid infrastructure.

The User Perspective provides extensions on top of Eclipse that respect the common Eclipse guidelines; this means it contains separate views and editors for different functionalities. These functionalities are served by a set of tools, implemented as component-orientated Eclipse plug-ins that integrate seamlessly with other views and editors which will be integrated in the g-Eclipse Grid User Perspective. The elements of the User Perspective will now be presented.



Job Definition Page

Job Definition

Job Definition element describes the job and its requirements.

Job Definition ID:

Job Identification

Job Identification element contains all the elements that identify the job.

Job Name: default job name

Job Description: default description

Name	Project
ce.egee	test7
hydra_geclipse 1.1 (ASCII -kkv)	MyFirstGridI
hydra_root 1.1 (ASCII -kkv)	MyFirstGridI

Fetching GlueCE from ldap://bdii101.gr

Fig. 5. General view of g-Eclipse User Perspective

IV.1. Grid Job Submission Management

The submission of Grid jobs includes some standard steps such as the selection of the executable, the definition of the input parameters and input data and the definition of the output data. The g-Eclipse User Perspective provides tools that support the Grid user through the various steps of the job submission procedure. The Grid job submission tool can be re-used and customized by any particular Grid application to execute a computational job on Grid resources. To submit a job to the Grid, first a job description must be created. The most general and middleware-independent job description format is currently the Job Submission Definition Language (JSDL) [11] which is defined by the OGF group. JSDL is the main job description language used in g-Eclipse, although other job description formats can be used as well. An example of this is the Resource Specification Language (RSL) [12] which is used to define Globus jobs. If the middleware does not support JSDL job description, then the responsible plug-ins transparently transforms a JSDL job description to a middleware native description. This is now done for gLite which supports the Job Definition Language (JDL) [13] only. JSDL is transformed to JDL automatically before the job is submitted.

The JSDL wizard is used to create JSDL files. The wizard asks for the most important JSDL parameters and creates the JSDL file. The file can later be edited using the JSDL multi-page editor.

In this way users can submit a simple executable and pass all necessary parameters for it. For more advanced applications the standard wizard can be enhanced with application-defined wizard pages. This can be used to ask the user about application-specific parameters in a more user-friendly way than just command-line arguments, e.g. check for parameter correctness, warn about erroneous values, use lists, combo boxes and sliders, etc. Such an advanced wizard should be prepared by an application developer or integrator, and be provided as an additional plug-in or XML file. Resulting JSDL file can be than edited using multi-page editor that assists the user in editing a document by grouping together common parts of the document in one tab and also provide instruction and information about the fields/properties/text that can be edited in that specific tab. Again, this provide a higher level of abstraction for the user compared to editing a pure xml file. In addition to simplify the task of creating and editing a JSDL file, the JSDL multi-page editor also helps in creating a well-formed document with a structure that is verified. As a result, the user is given error messages when she has entered invalid data into fields.

IV.2. Grid Job Monitoring

Access to a large number of Grid resources allows the Grid user to execute many, but slightly different jobs at the same time to compare their results. The g-Eclipse Grid User Perspective provides a user-friendly and intuitive tool to monitor and control these jobs on-line. A list of jobs is presented to the user containing fundamental information about each of them, such as the date of submission, current status and other common values. More details will be displayed on-demand for selected jobs.

The Grid job monitoring view shows a list of all currently existing jobs with some properties about each of them, such as the job name and status. The list can be filtered by status, date, job directory, hosts etc. Another view shows all information about a specific job. It includes all the status information that is available from the Grid middleware. e.g. status history, job description etc.

Another way of presenting standard job information is via the usage of the standard PropertyDialog, which shows properties for any GridElement in the Grid project tree.

IV.3. Grid File Management

Files containing data are distributed in the Grid and replica management layer controls access to them. The users of Grid resources would prefer a similar method of accessing the files to the way they access files on their personal computer. Moreover, there are various replica managers for different middleware (some, like gLite, have more than one). g-Eclipse's task is to present a consistent `look and feel' to the user. The g-Eclipse User Perspective provides methods that enable Grid users to access an abstract storage space. It contains a unified view to different types of storage, such as Storage Elements, GridFTP, DPM implementation of GridFTP or Storage Resource Manager (SRM) resources and local media, such as hard disk and removable media (CD, DVD, USB memory sticks).

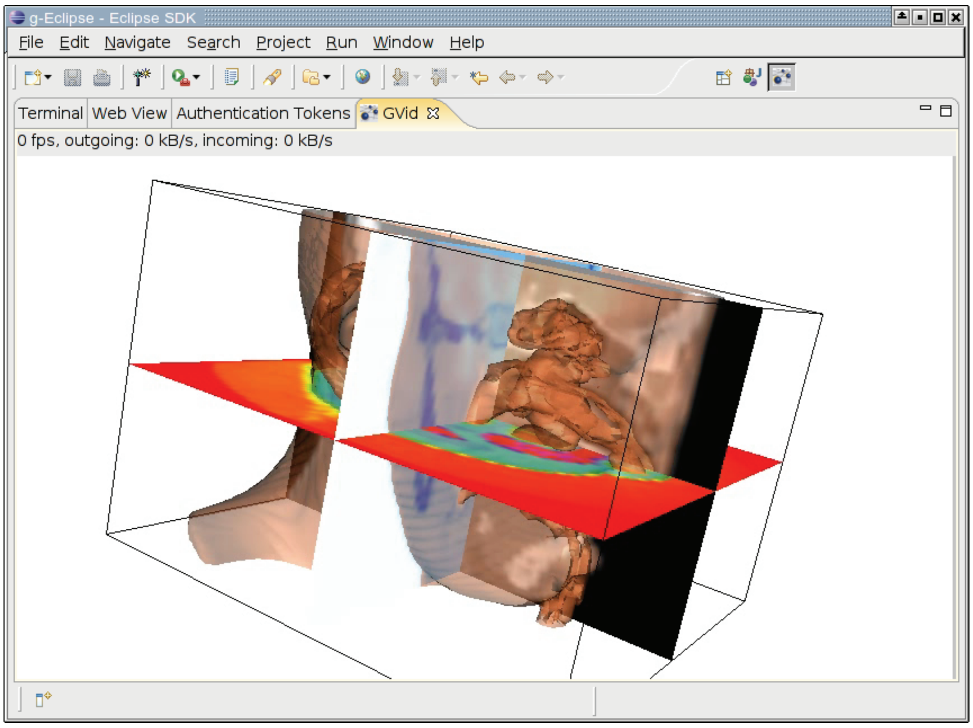
The g-Eclipse data management subsystem hides low-level details about the file systems. The user deals with high level operations only – create directory, copy file, remove file, copy directory etc.

Common user actions such as drag & drop or copy & paste are supported as well. File transfers that would take a long time can be put into the background to avoid blocking the user interface. Except for a small time delay, users should notice no difference between remote and local file systems.

Because of good integration with the Eclipse platform, g-Eclipse is able to utilize third-party implementations of file systems. g-Eclipse exploits the core Eclipse File System (EFS) API and therefore, any file system which was developed to work with the Eclipse Platform will also work with g-Eclipse.

The Grid is composed of distributed, heterogeneous resources. Transferring data between different nodes involves the use of several tools. In g-Eclipse, these tools are selected automatically. Scientists and engineers can focus on their task rather than on which tools to use.

Easy use is the highest priority for g-Eclipse. Earlier, users had to remember not only a lot of information, but also the location of the information. g-Eclipse utilizes the BDII information system in the background, so that the user


g-Eclipse - Eclipse SDK
File Edit Navigate Search Project Run Window Help
Terminal Web View Authentication Tokens GVid

0 fps, outgoing: 0 kB/s, incoming: 0 kB/s

Fig. 6. Example visualisation

is able to browse the Grid and its storage elements without knowledge about its addresses or configuration. But advanced or experienced users can work with this raw data, if they find it more useful.

IV.4. Grid Visualization

The Grid allows complex calculations of any kind for Grid users and scientists. The presentation of the results of such calculations is another challenging task. In many scientific domains, the graphical visualisation of these results will provide the scientists and engineers with a deeper insight into the problem and its solutions. Therefore the integration of a user-friendly and interactive visualisation tool in a general integrated Grid environment is addressed by the g-Eclipse Grid User Perspective.

For this task, the g-Eclipse project builds upon the well known Open Source Visualization Toolkit (vtk) by Kitware [14]. This has the advantage that scientists who are already used to doing their visualization with this widely used toolkit only need very little learning efforts to perform their visualization tasks interactively on the Grid using g-Eclipse.

Depending on the trade-off between the size of the data to be visualized and the computing rendering power of a user's desktop, g-Eclipse offers two different ways of performing the visualization, which should be usable transparently in the future. One is to move the data to the client host running g-Eclipse using the integrated data management of the platform and performing local rendering with the g-Eclipse vtk client. The other is to find a machine close to the data on the Grid offering remote rendering functionality being advertised in the information system and the usage of the native vtk-gvid applications. The former makes sense for small datasets and more powerful machines, the latter for large datasets where powerful hardware-based remote rendering is available.

In the first release, remote rendering with GVID[15] is available. The application needs to implement the vtk-based visualisation and needs to be linked with the GVID vtk libraries. Being submitted to a Grid site being prepared appropriately, offering off-screen hardware rendering functionality, the user can then transparently use an application being rendered remotely, only interacting with a video stream using the g-Eclipse GVID client. Fig. 6. shows a screenshot of a vtk demo application.

IV.5. Grid Command Console

Sometimes experienced users would like to access Grid computers directly. The g-Eclipse Grid command console provides a low-level interface to Grid-based resources. Similarly to the command line interface on standard desktop workstations, a user may need to access, inspect and steer resources in the Grid environment using a normal shell access. A command console is now implemented with two protocols: SSH and gLogin. The SSH console is just the standard secure shell using standard authorization. gLogin offers ssh-like shell access to remote machines based on X.509 user credentials for authentication and the same features such as port-forwarding and X11 tunneling. Even if the idea of having such functionality in the Grid looks very straightforward, it has not been foreseen from the beginning and therefore had to be developed by tricking the first middleware. The same functionality offering interactive shell access is also used to offer a generic interactive tunnel into the Grid being used by many parts of g-Eclipse, e.g. remote debugging or the GVID component.

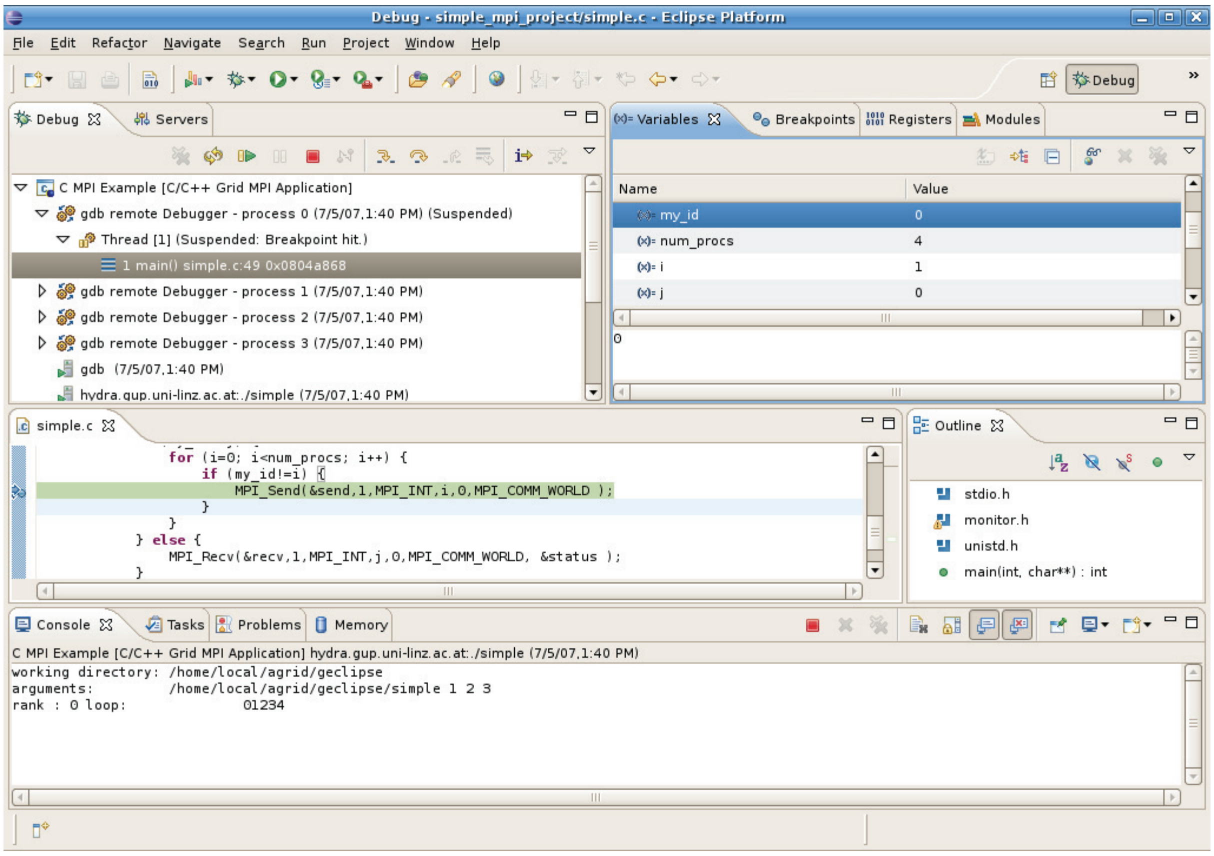
V. GRID DEVELOPER PERSPECTIVE

The g-Eclipse Developer Perspective contains very high-level features to support developers developing for Grids working through the full development cycle of their applications. This includes support for preparing Grid applications, and debugging applications remotely.

V.1. Grid Application Development and Debugging Tools for C++ and Java

Supporting the Application Development Cycle on the Grid using g-Eclipse is projected in two parts. The first idea is to support developers using g-Eclipse with the same type of tooling they are used to when developing for their desktop using their favourite Integrated Development Environment (IDE). The second idea is to support their work by providing wizards for creating stubs and templates as a basis for developing applications that can utilize Grid middleware functionality on an API basis or even more sophisticated – developing extensions to the g-Eclipse platform itself.

The first set of tooling is already quite complete and usable. It is integrated in the first release of g-Eclipse and outlined briefly in the following paragraphs. The project has identified support for developing C/C++ and Java applications in its scope of work, both of which are already very well supported within the Eclipse Platform through CDT (the C/C++ Development Tooling) and JDT (the Java Development Tooling). This includes editing, building and graphical debugging. So the main idea has not been to re-invent a full set of tools for developing for the Grid, but instead to find mechanisms to interface the existing tooling to operate locally on a user's machine with the Grid. This is currently achieved by creating application-transparent


Debug - simple_mpi_project/simple.c - Eclipse Platform
File Edit Refactor Navigate Search Run Project Window Help
Debug
Debug
Servers
Variables
Breakpoints
Registers
Modules
Outline
Console
Tasks
Problems
Memory

```
C MPI Example [C/C++ Grid MPI Application]
  gdb remote Debugger - process 0 (7/5/07,1:40 PM) (Suspended)
    Thread [1] (Suspended: Breakpoint hit.)
      1 main() simple.c:49 0x0804a868
  gdb remote Debugger - process 1 (7/5/07,1:40 PM)
  gdb remote Debugger - process 2 (7/5/07,1:40 PM)
  gdb remote Debugger - process 3 (7/5/07,1:40 PM)
  gdb  (7/5/07,1:40 PM)
  hydra.gup.uni-linz.ac.at:./simple (7/5/07,1:40 PM)
```

Name	Value
my_id	0
num_procs	4
i	1
j	0

```
0
```

simple.c

```
        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, &status );
    }
```

Outline: stdio.h, monitor.h, unistd.h, main(int, char**) : int

```
C MPI Example [C/C++ Grid MPI Application] hydra.gup.uni-linz.ac.at:./simple (7/5/07,1:40 PM)
working directory: /home/local/agrid/geclipse
arguments:         /home/local/agrid/geclipse/simple 1 2 3
rank : 0 loop:              01234
```

Fig. 7. Remote debugging of MPI application on Grid

interactive communication channels using the g-Eclipse gLogin plugin for accessing remote resources and interfacing locally built & remotely deployed binaries with the local workbench and its debugging facility.

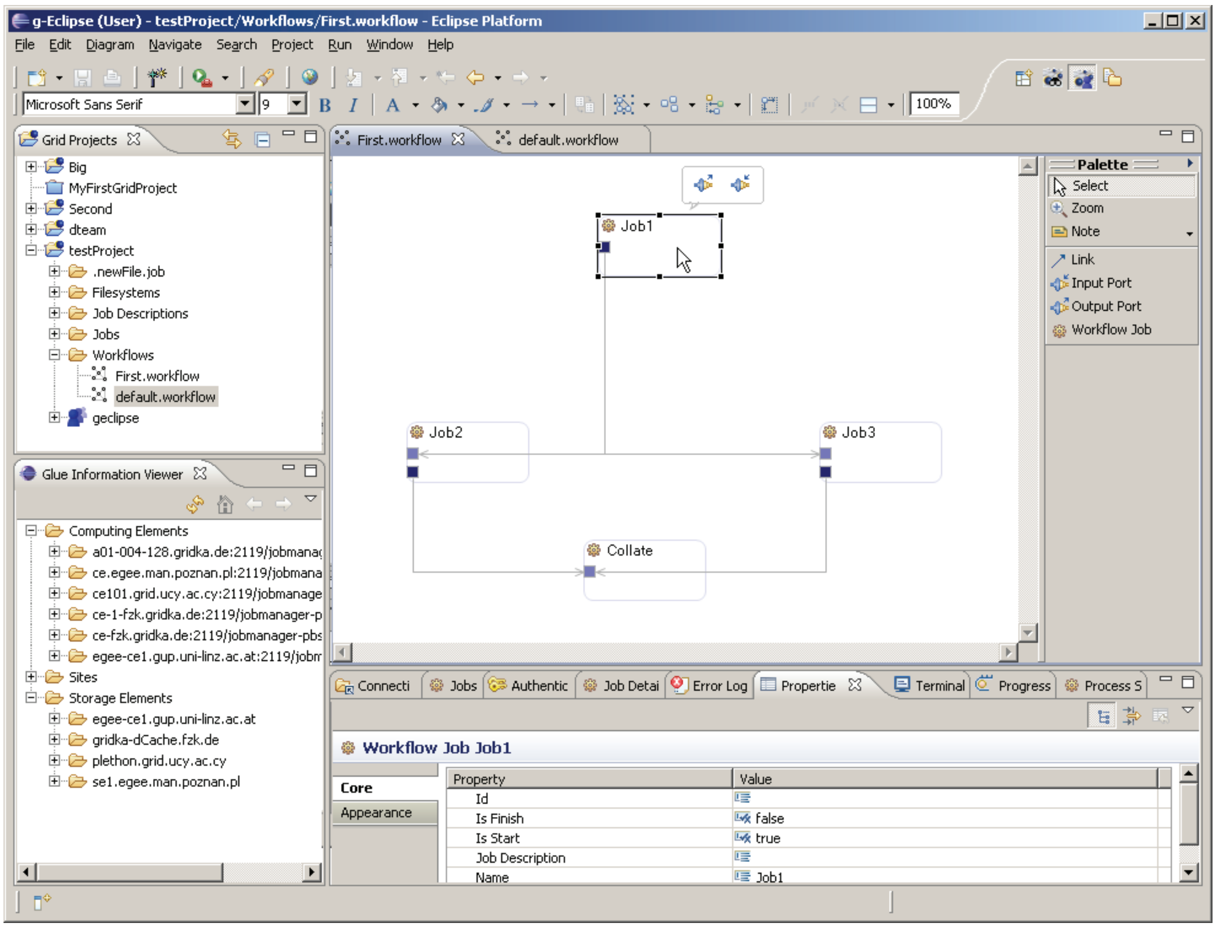
The g-Eclipse way to build and run or debug applications with CDT or JDT is to edit the source files in the corresponding development perspective of Eclipse and then to create so called ”launch configurations”. These automatically deploy the locally built binaries to the respective Grid resource, launch them and connect the local workbench to the remotely running instance through the interactive connection to the Grid. The same functionality is available for Java projects as well and is handled for that case in a similar way. In the case of C/C++ applications, g-Eclipse additionally supports the development of parallel applications using the message passing interface (MPI) [16] programming paradigm and the same concept of having special launch configuration for running on the Grid utilizing the interactive gLogin communication channel.

Figure 7 shows a simple parallel (MPI) application being debugged, running on a Grid computing node. The only way this view differs from a normal Eclipse CDT debugging view is the way processes are launched through the g-Eclipse Launch Configuration and how the graphical debugging frontend is connected through gLogin to the debugging backends on the Grid.

V.2. Grid Application Deployment Tools

When the application is prepared, compiled and tested locally, it must be deployed on the Grid to share it with other users. For some Grid infrastructures it may require a special permission. Application deployment using g-Eclipse is split into two separate parts. The first one is the definition of a generalised User Interface (UI) frontend that enables the deployer to select the resources to deploy, the targets where these resources should be deployed and an optional deployment tag. As soon as these deployment parameters are fixed, the second part becomes operative. This is the actual deployment process.

The deployment process itself can be either represented by a shallow middleware-independent operation or by a high-



Property	Value
Id	
Is Finish	false
Is Start	true
Job Description	
Name	Job1

Fig. 8. Example workflow definition

ly specialised middleware dependent operation. The first one makes use of the core functionalities for mounting remote filesystems and for transfering data between the local installation and these remote targets. In that sense the deployment process is equivalent to a simple copy of the resources that have to be deployed to one or more target systems.

In order to make a clean application deployment to the Grid, one has to use the middleware-specific mechanisms to upload files and to register these files as application(s). This is then made available for the users of the Grid on the corresponding computing elements by the middleware. Within the g-Eclipse framework, these deployment techniques may not be implemented by the core itself but by a specific middleware extension. The framework itself again provides a common UI to the deployer and the possibility to choose among all registered deployment tools in order to start a deployment process.

V.3. Graphical Grid Workflow Builder

The part of g-Eclipse providing support for workflows has two distinct components - a graphical editor to create & edit workflows and the execution management part. A workflow model was defined and a graphical editor was created on the basis of this model. Currently, this editor allows the creation of workflows by the addition of workflow jobs and links between them. The description of a workflow composed in the g-Eclipse Worklow Editor (See Fig. 8) can be saved to a file. At the moment it is in an XML Metadata Interchange (XMI) format. In order to support other workflow description languages, it is envisaged to provide an `Export to' and `Import from' functionality. This would allow users to export their workflows created in g-Eclipse to formats supported by various workflow engines. As the Grid middleware chosen for the first year of the g-Eclipse Project is gLite, work on an export funtionality to the Job Description Language (JDL) is ongoing. In JDL, a workflow is represented as a Directed Acyclic Graph (DAG). Hence, the current workflow model used in g-Eclipse does not support loops.

V.4. Grid Application Monitoring

From the Developers' Perspective, Application Monitoring is split into two different levels of granularity. Coarse-grained Application Monitoring concerns the tracking of submitted jobs' status which is already addressed through the g-Eclipse job-status plug-in. Fine-grained Application Monitoring, on the other hand, involves fetching of the status of remotely running process(-es) at the Operating System level. The latter allows selecting processes and resources from the context menu in the Grid Project View and is implemented in a separate plug-in.

VI. CONCLUSIONS

g-Eclipse is a user-friendly environment for accessing Grid resources. It can support a wide range of potential Grid users, from Grid newbies to application developer experts. It has proved to be a very useful tool for accessing Grid resources. Currently the supported Grid infrastructures are gLite and Globus. Support for GRIA is started and this will be the first usage of g-Eclipse in industry. g-Eclipse is based on the open Eclipse platform founded by IBM. The usefulness and significance of g-Eclipse were noticed by the Eclipse Foundation and terefore became an official Eclipse technology project. As a result g-Eclipse is an open source project and the sources are public and available for the g-Eclipse community, which can easily add support for other Grid middleware.

g-Eclipse is a valuable tool that can be used by all Grid users for their daily work. As it does not require deep Grid knowledge, it is really suitable for inexperienced users that would just like to run their computations on the Grid without lerning its complexity.

Acknowledgments

g-Eclipse project is funded by the European Commission's 6th Framework Programme, Contract number IST-034327.

References

[1] F. Gagliardi, *The EGEE European Grid Infrastructure Project*, LNCS, Volume 3402/2005, p. 194-203.

[2] F. Hemmer, E. Laure, M. Barroso Lopez, A. Di Meglio, S. Fisher, L. Guy, P. Kunst and F. Prelz, *Middleware for the Next Generation Grid Infrastructure*, Proceedings of CHEP 2004, Intelaken, Switzerland, 2004.

[3] G. Andronico, R. Barbera, A. Falzone, G. Lo Re, A. Pulvirenti and A. Rodolico, *The GENIUS web portal: grid computing made easy*, International Conference on Information Technology: Computers and Communications, 2003, p. 425-431.

[4] G. Allen, K. Davis, K. Dolkas, N. Doulamis, T. Goodale, T. Kielmann, A. Merzky, J. Nabrzyski, J. Pukacki, T. Radke, M. Russell, E. Seidel, J. Shalf and I. Taylor, *Enabling Applications on the Grid: A GridLab Overview*, International Journal of High Performance Computing Applications, Aug. 2003, p. 449-466.

[5] *Distributed european infrastructure for supercomputing applications – DEISA*, http://www.deisa.org

[6] D. W. Erwin and D. F. Snelling, *UNICORE: A Grid Computing Environment.* Euro-Par 2001, LNCS Volume 2150/2001: p. 825-834.

[7] J. Novotny, M. Russell and O. Wehrens, *GridSphere: An Advanced Portal Framework.* EUROMICRO 2004: p. 412-419.

[8] Eclipse platform – www.eclipse.org

[9] M. Surridge, S. Taylor, D. De Roure and E. Zaluska, *Experiences with GRIA – Industrial Applications on a Web Services Grid*, Proceedings of the First International Conference on e-Science and Grid Computing, 2005, p. 98-105.

[10] Object Management Group, www.omg.org

[11] Job Submission Description Language (JSDL) Specification, Version 1.0, http://www.gridforum.org/documents/GFD.56.pdfl

[12] The Globus Resource Specification Language RSL v1.0, http://www.globus.org/toolkit/docs/2.4/gram/rsl_spec1.html

[13] Job Description Language Attributes Specification, https://edms.cern.ch/file/590869/1/EGEE-JRA1-TEC-590869-JDL-Attributes-v0-8.pdf

[14] The Visualisation Toolkit, http://www.vtk.org/

[15] T. Stütz, and M. Polak, *Gvid – Video Coding and Encryption for Advanced Grid Visualization*, 1st Austrian Grid Symposium, Hagenberg, Austria, December 1, 2005.

[16] M. Snir, S. Otto, S. Huss-Lederman, D. Walker and J. Dongarra, *MPI: The Complete Reference*, Massachusetts Institute of Technology, 1996.

Paweł Wolniewicz graduated from Poznań University of Technology and received M.Sc. in computer Science in 1997. In 2003 he presented and defended his Ph.D. thesis in computer science at the Institute of Computing Science, Poznań University of Technology. Currently he works for Poznań Supercomputing and Networking Center, Poznań, Poland. His research interests include Grids, distributed environments and scheduling. Since 2002 he has been working on several international research projects in the field of Grid Computing.

Norbert Meyer is currently the head of the Supercomputing Department in Poznań Supercomputing and Networking Center. His research interests concern resource management in GRID environment, GRID accounting, data management, technology of development graphical user interfaces and network security, mainly in the aspects of connecting independent, geographically distant Grid domains. NM conceived the idea of connecting Polish supercomputing centres, vision of dedicated application servers and distributed storage infrastructure. He is the author and co-author of 40+ conference papers and articles in international journals, member of programme committees of international conferences related high performace computing and grid computing.

Maciej Stroiński received the Ph. D. degree in Computer Science from the Technical University of Gdańsk in 1987. Currently he is Technical Director of the Poznań Supercomputing and Networking Center. He is also lecturer in the Institute of Computing Science of the Poznań University of Technology. His research interests concern computer network protocols and management. He is author or co-author of over 100 papers in major professional journals and conference proceedings.

Mathias Stümpert is the project coordinator of the g-Eclipse project. He is working as a principle researcher in the Institute for Scientific Computing at the Forschungszentrum Karlsruhe. His background is astro-particle physics where he made his PhD in November 2007 by investigating anisotropies in cosmic-ray arrival directions using data of the KASCADE-Grande experiment. In August 2006 he joined the g-Eclipse project as responsible person for the architecture of the framework, the quality of the source code and the integration of the different components. He became project coordinator in October 2007 and is still responsible for large parts of the g-Eclipse core

DR. HARALD KORNMAYER works as senior researcher in the IT Division of NEC Laboratories Europe in St. Augustin, Germany. His research agenda focus on distributed wide scale IT systems including Grid and SOA technologies. Before he joined NEC Laboratories Europe, he worked in different European and national Grid projects (i.e. CrossGrid, EGEE, D-Grid). He is Eclipse project lead of the g-Eclipse project since 2006 and coordinated the European project until September 2007 as scientific coordinator.

MARTIN POLAK is currently doing his Ph.D. under the supervision of Univ. Prof. Dr. Jens Volkert at the Institute of Graphics and Parallel Processing (GUP), University Linz. His interest and focus of is work is studying the applicability of more efficient encryption methods to interactive video streams for usage in grid based high performance visualization scenarios. Since 2002 he has been working on several international research projects in the field of Grid Computing and he also is an official Eclipse.org committer.

HARALD GJERMUNDRØD is currently a postdoctorate associate at the High-Performance Computing Systems Laboratory in the Computer Science Department of the University of Cyprus. His research interests include middleware, distributed computing systems, and Grid computing. Gjermundrød received his PhD, MS, and BS degrees in computer science from Washington State University in 2006, 2001, 1999 respectively and Dipl.-Ing degree from Oslo University College in 1998 (including a year as an Socrates/Erasmus student at the Robert Gordon University). Gjermundrød has worked on projects funded by the National Institute of Technology and the National Science Foundation in the USA and served as a reviewer for several scientific conferences.