Enabling Grids for E-sciencE

# Gridification practices in gLite 3.x

***Vladimir Dimitrov***
*IPP-BAS*

*“gLite middleware Application Developers Course”,*
*Plovdiv, Bulgaria, 4.04.2008*

www.eu-egee.org

Information Society

# Outline

- **This talk gives a high-level view of application development in Grids**
- **Contents**
  - Review of concepts: grids and grid applications
  - Characteristics of VOs
  - Challenges to researchers who write applications
  - General steps of application gridification
  - Practical: Preparing and submitting a job based on a simple non-grid application.
- **Acknowledgements**
  - Gergely Sipos, SZTAKI, Hungary
  - Miklos Kozlovszky, SZTAKI, Hungary
  - Mike Mineter, University of Edinburgh, UK, "Application Development and Aspects of gLite 3.0", 2006
  - Douglas Thain, The University of Notre Dame
  - GILDA team

# What is a grid application?

- *Definition*

**Software that interacts with grid services to achieve requirements that are specific to a particular Virtual Oraganization (VO) or user.**

# Characteristics of VOs

- **What is being shared?**
  - resources of storage and/or compute cycles
  - software and/or data

- **Distinct groups of developers and of users?**
  - Some VOs have distinct groups of developers and users…
    - Biomedical applications used by clinicians,….
  - …. Some don't
    - Physics application developers who share data but write own analyses
  - Effect: need to
    - hide complexity from the 1st type of VOs
    - expose functionality to 2nd type of VOs
    - many security issues

# Basic tasks while Porting applications to the Grid

1. **Developing a non-grid application (or inheriting and updating an ancient one);**
2. **Go/no-Go decision about gridification**
   - Is it suitable for the Grid environment?
   - “Cost/profit” analysis / feasibility study

**Typical Questions Groups:**

- Current structure of the application
- Dependencies of the application
- Available resources (manpower, knowledge, etc.)
- Requirements for the gridified application
- Expected impact of gridification
- Requirements for the grid infrastructure

# Basic tasks while Porting applications to the Grid (contd.)

3. **Grid environment access**
   - Requesting Certificates / VO membership
   - Accessing Grid environment
     - Appropriate VO **UI** machine account for command line
     - Portal GUI account;

4. **Executing, Testing and Debugging the application;**
   - Testing the non-grid application (debugging in Grid environment is a hard task), creating use cases for single (non-grid) runs;

5. **Constructing the job suite – JDL (Job Description Language) files, executables, auxiliary scripts and input/output data files;**

6. **Submitting the job to the Grid as a small-scale pilot application;**

# Basic tasks while Porting applications to the Grid (contd.)

7. **Executing, Testing and Debugging the pilot application;**

8. ***IF* something goes wrong**
   ***THEN GOTO 4***;

9. ***IF* everything seems to work**
   ***THEN*** increase the scale of the application (increase problem size, amount of used resources);

10. **Optimizing and improving the grid application;**

# Porting legacy code applications

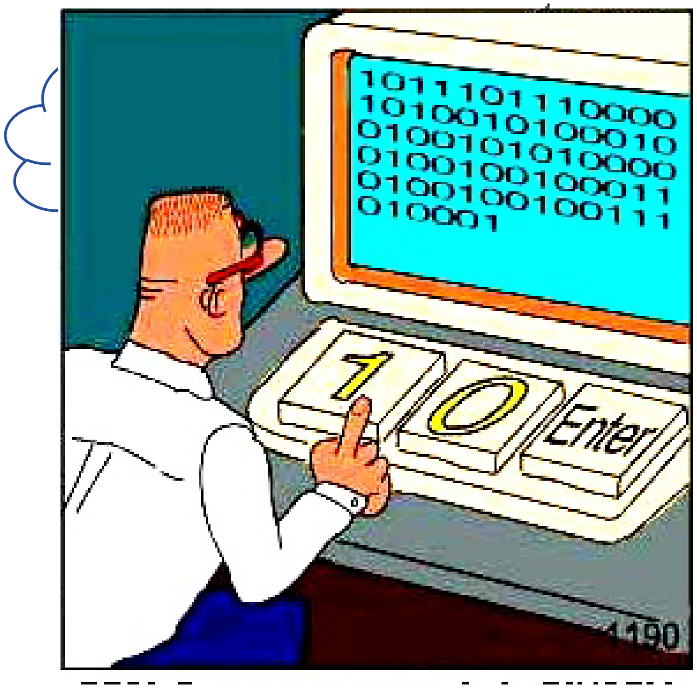
**Code from the past, maintained because it works**

Often supports business critical functions

Not Grid enabled

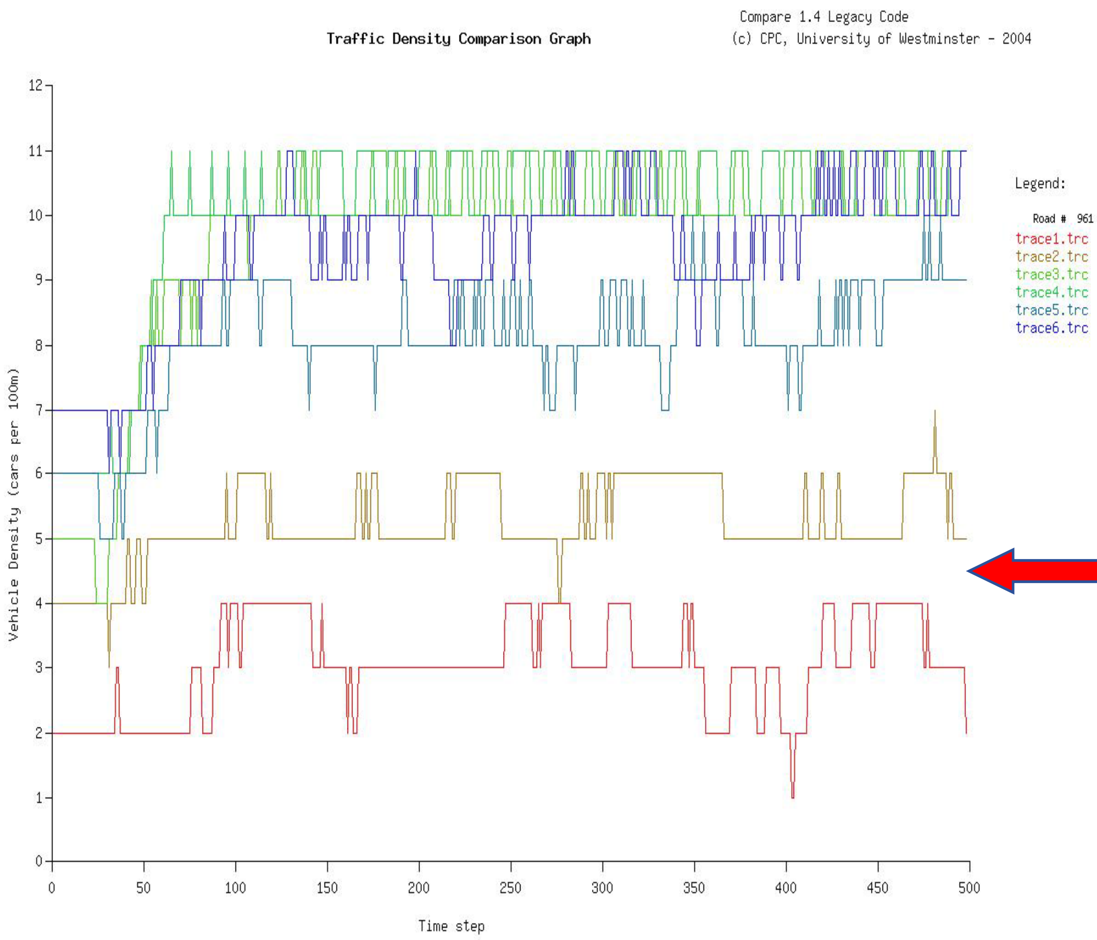
**What to do with legacy codes in service Grids?**

- Bin them and reimplement them as grid services
- Reengineer them → who knows the source code?
- **Port them onto the Grid with minimum user effort!**



# Legacy Application example

## Workflow to analyse road traffic




Manhattan road network generator

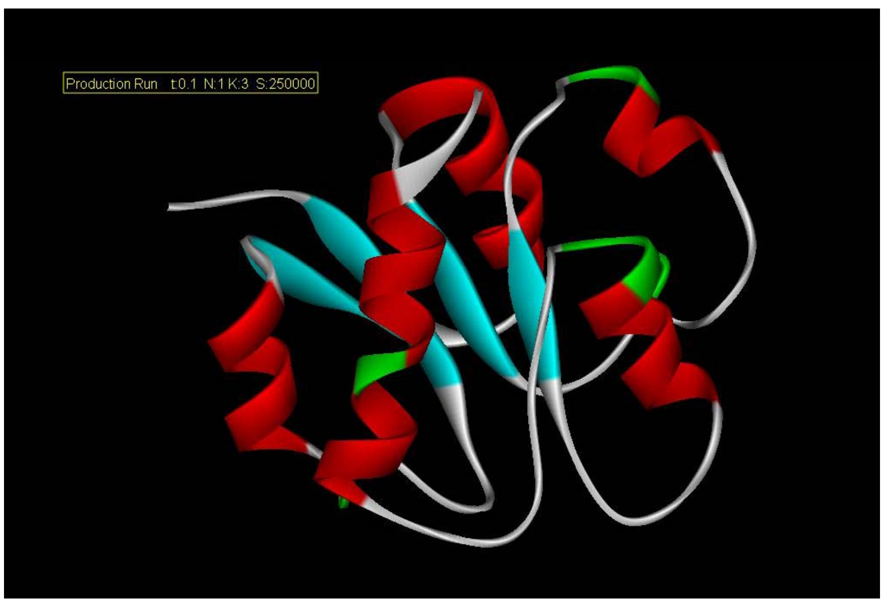
Traffic simulators

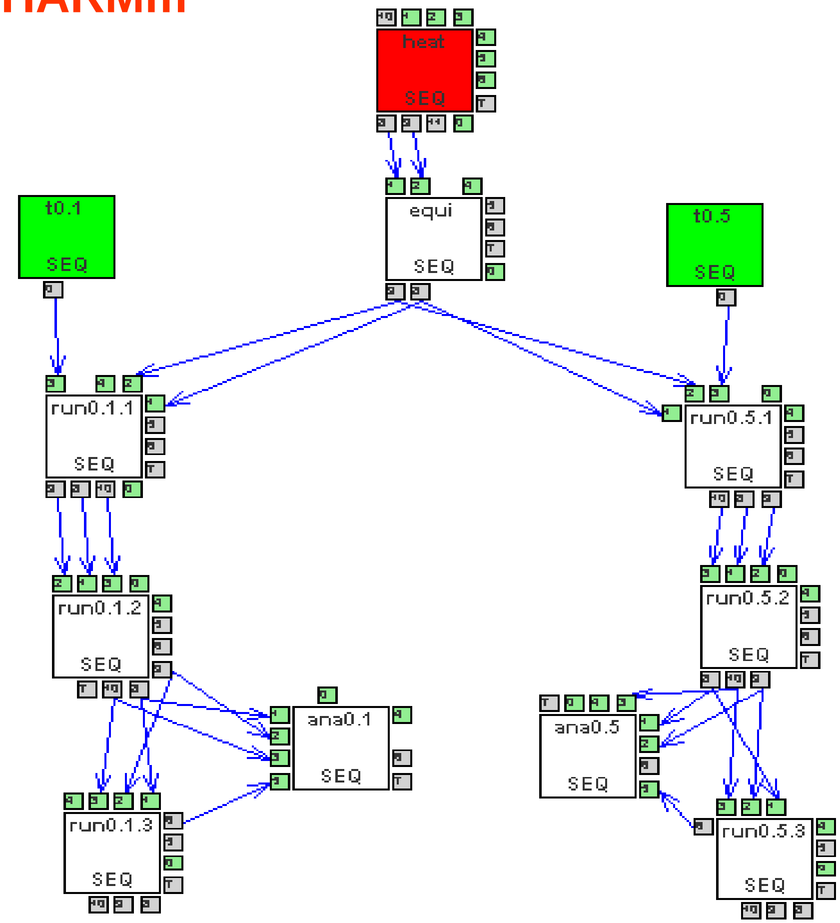
Analyser

# Another Legacy Application example

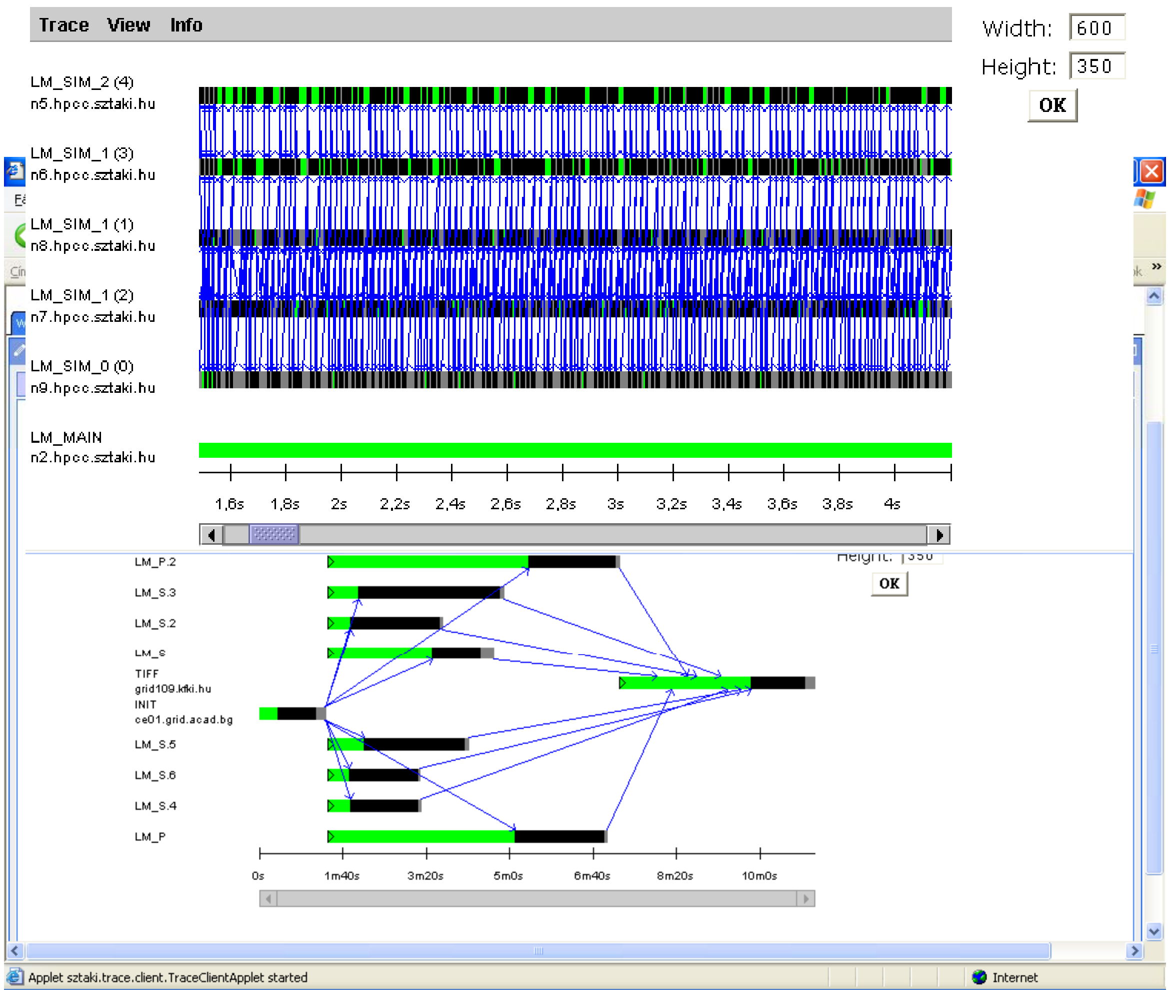
## Molecular Dynamics Study of Water Penetration in Staphylococcal Nuclease using CHARMm

- Analysis of several production runs with different parameters following a common heating and equilibrium phase



# On-Line Monitoring




## Mercury monitor

- to debug/optimize and visualize parallel jobs.
- both at the workflow and job levels

# Practical tools @ gridification (contd.)

## Hiding Grid remote storage system from a legacy application

- **Parrot**
  - a handy tool for attaching old programs to new storage systems
  - EGEE (gLite module) Data Access: GFAL, LFN, GUID, SRM, RFIO, DCAP, and LFC
  - does not require any special privileges, any recompiling to existing programs
  - For example, an anonymous FTP service is made available to **vi** like so:

```
parrot vi /anonftp/ftp.cs.wisc.edu/RoadMap
```

**Or example with gsiftp:**

```
$./parrot app_name /gsiftp/<gsiftp usr without gsiftp://>
```

*More info:*
http://www.cse.nd.edu/~ccl/software/manuals/parrot.html

# Common problems and obstacles

## Now some bad news:

- The candidate-applications for porting usually are huge and complex.
- Some of them use low-level network functions and/or parallel execution features of a specific non-grid environment.
- Usage of non-standard or proprietary communication protocols.
- The complete source code might not be available, might not be well documented or its “out-of-host” usage is  restricted by a license agreement.

# Common obstacles (continued)

- The application might be written in many different programming languages – C, C++, C#, Java, FORTRAN etc. or even mixture of them.
- Applications may depend on third-party libraries or executables which are not available by default on some Grid worker nodes.
- Some application features could cause unintentional violation of Grid Acceptable Use Policies (Grid AUP).
- Furthermore, the application can have hidden security weakness which will be very dangerous in case of remote Grid job execution.

# Common obstacles (continued)

- **Some applications are pre-compiled or optimized for using on a machine with particular processor(s) only – Intel, AMD, in 32-bit or 64-bit mode, etc. <u>But the Grid is heterogeneous</u>!**

- **The application may contain serious bugs, which have never been detected while running in a non-grid environment.**

- **Lack of convenient Grid-enabled program debuggers and profilers.**

- **Finally, the formal procedure for accepting a new application to be ported to a Grid for production or even for experimental purpose is not simple.**

**Therefore, the porting of an arbitrary application to Grid could be very long, difficult and expensive process!**

# Grid Application Development Support Model

- **8 steps support model**
  - Contact phase
  - Pre-selection phase
  - Analysis phase
  - Planning phase
  - Prototyping phase
  - Testing phase
  - Execution phase
  - Dissemination and feedback phase

# In the practical

- **The application called *MatrixDemo* will be ported and executed in GILDA grid environment.** (The program is borrowed from the *“EGEE summer school”* at MTA SZTAKI, 2006. The archive can be downloaded from:
  **`http://vgd.acad.bg/grid/MatrixDemo.zip`**)

  ***MatrixDemo* is written in C programming language**

GILDA environment (gLite based) is supporting C**,**
**so porting the C or C++ programs is easy … hopefully.**

# *MatrixDemo* program

- *MatrixDemo* program performs some matrix operations – inverting, multiplying, etc.

- **Usage:**
- *MatrixDemo* has command line interface which accepts several arguments. Starting the program without any argument will display a short help.
  - Example:

```
MatrixDemo I V
```

This will Invert (**`I`**) the matrix defined in the file named INPUT1 and will store the result in the file OUTPUT with verbose details (**`V`**).

# *MatrixDemo* program (continued)

- **Prerequisites:**
- **File *MatrixDemo.c*** – the source code of the program.
- **Files *INPUT1* and *INPUT2*** – they contain matrix data in the following text format:

  ***rows, columns, cell1, cell2, cell3 …***

  Where *rows* is an integer representing the number of rows. *columns* represents number of columns, and *cell1, cell2* etc. are the cells of the matrix, floating point numbers separated by commas (,).
- **A standard C compiler and linker.** In this case we will use GNU C (gcc) already installed.
- **File *MatrixDemo.jdl*** – a prepared JDL (Job Description Language) file.

# Questions?

?