



School of Computing, Computational PDEs Unit

<http://www.comp.leeds.ac.uk/cpde/>

# A Grid-Enabled Parallel PSE for Computational Engineering Design

Chris Goodyer, Martin Berzins, Peter Jimack

This work is funded by the UK e-Science programme in collaboration with Shell  
Global Solutions

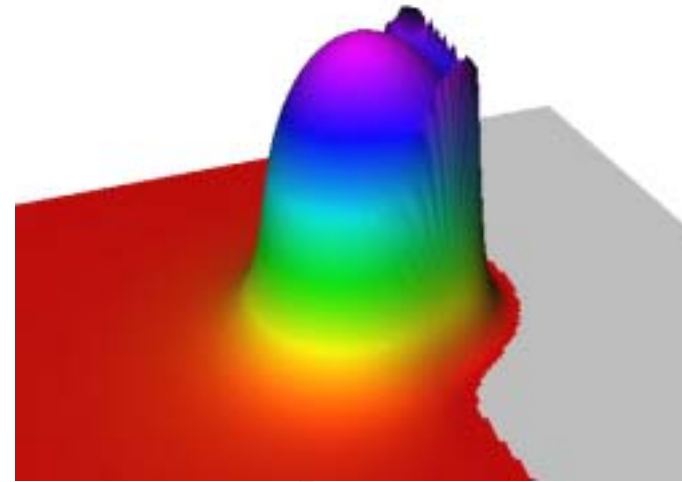
## Talk outline

- Description of the engineering problem
- Parallel solution methods
- A Problem Solving Environment for this case
- Using PSE with the Grid

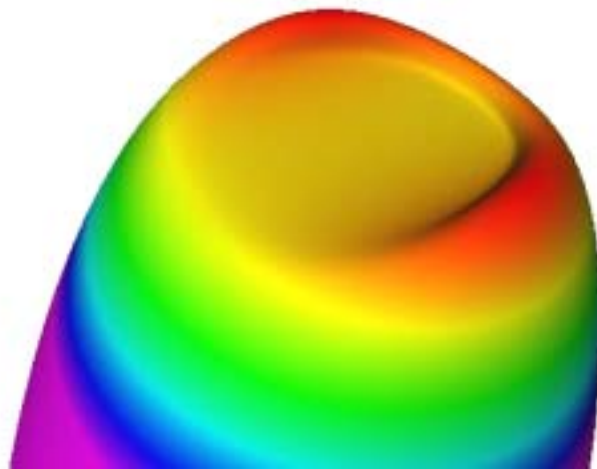
# Lubrication Modelling

Put gear movie here

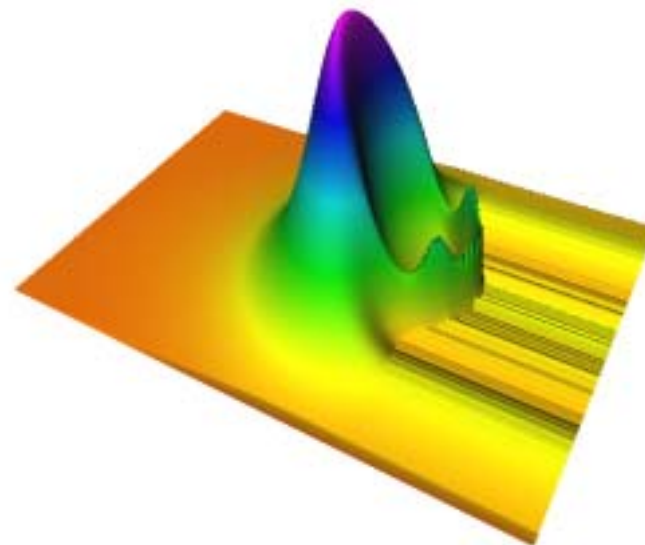
# Typical solutions



Pressure



Film thickness



Temperature

# Purpose of the Optimiser

- Wish to find set of lubricant model parameters (e.g. viscosity, pressure, temperature coefficients) which best match observed data
- Typically over 10 such parameters to optimise
- Physical observations already taken for different loadings (3), ambient temperatures (2) and slide:roll ratios (6)

Therefore must perform non-linear optimisation of

$$\mathcal{R}_F = \sum_{j=1}^{36} \left( F_j^{num} - F_j^{exp} \right)^2$$

⇒ Each  $\mathcal{R}_F$  evaluation needs 36 numerically expensive EHL calculations.

- Shell uses a simplex method to optimise

## Numerical solution of EHL cases

- Highly non-linear coupled integro-differential equation system to solve
- Multigrid is used to achieve rapid convergence on fine grids
- Most expensive part is the deformation calculation which is  $\mathcal{O}(N^4)$ 
  - This cost reduced to  $\mathcal{O}(N^2 \ln N^2)$  by using Multilevel Multi-integration [Brandt and Lubrecht]
  - MLMI similar to Fast Multipole Method
- Actual friction calculation is a post processing stage on the solution variables

# Parallelism of the solver

For 2-d cases there is the ability – and need – to parallelise the code

- Use strips to maintain convergence of solution method

Domain decomposition split between processors with each having an equal number of rows

Halos for each partition of 1 row for MG process, but much wider for MLMI

Number of grids used in MG/MLMI process currently governed by critical level

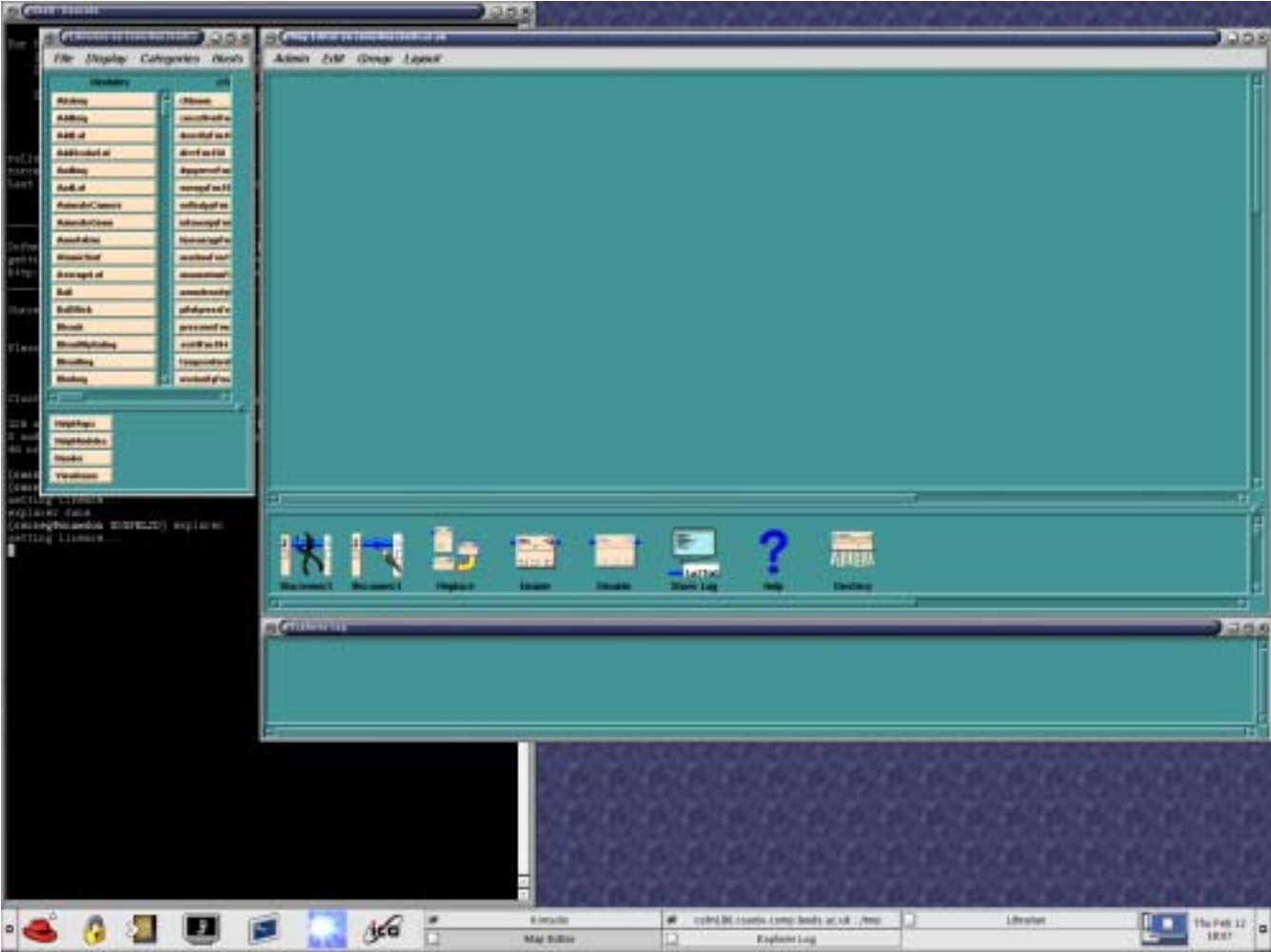
- Each processor must have some work to do for ease of communication pattern and load balancing
- Agglomeration another possibility

Isomemory is not easy due to global nature of deformation calculation

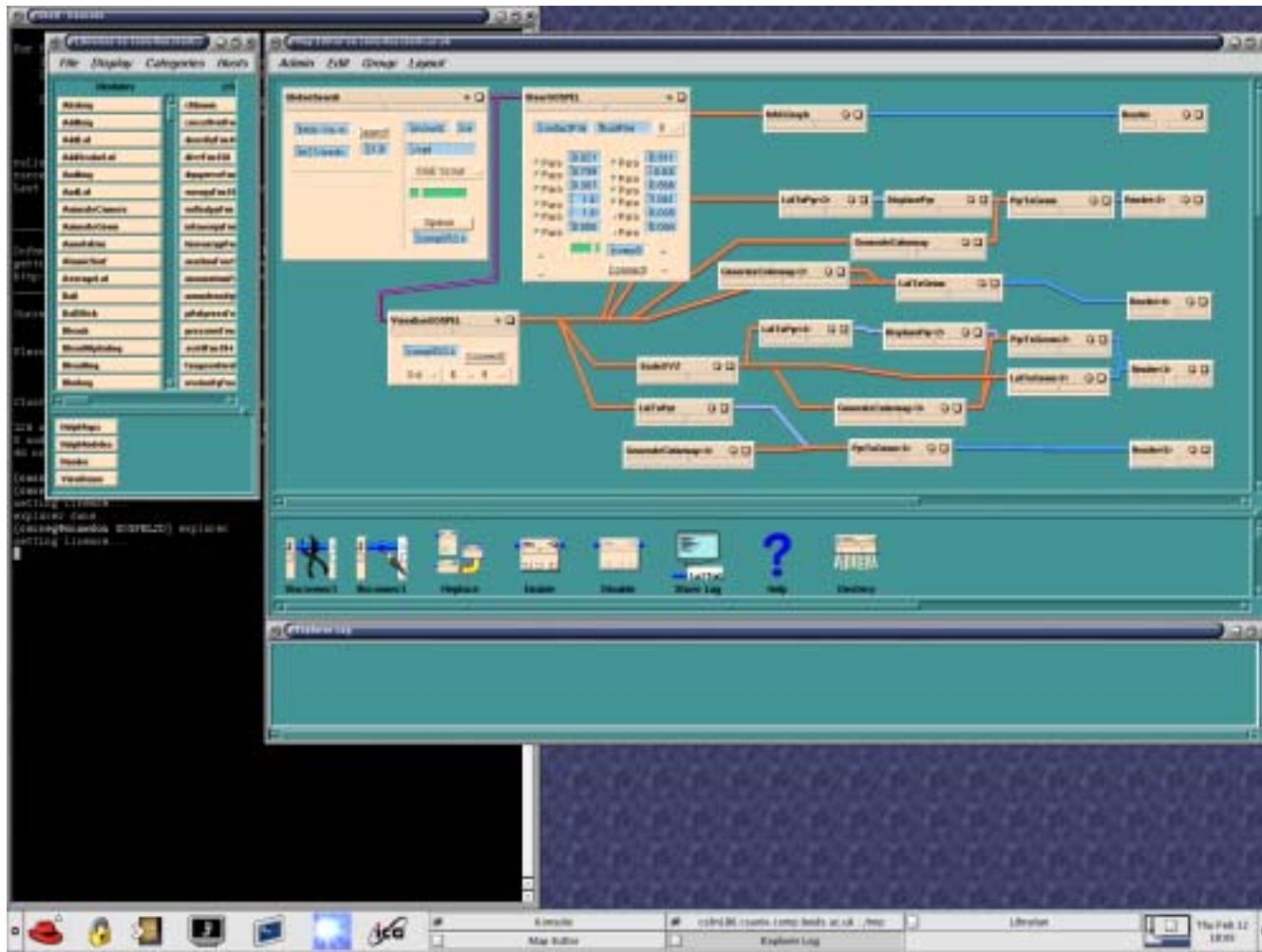
# Parallelising the Optimiser

- Each  $\mathcal{R}_F$  evaluation independent between all 36 cases
  - ∴ Could compute all 36 in parallel
- However increasing slide:roll ratios give better convergence rates
  - Hence use continuation along cases with fixed load and temperature
- Uses MPI Communication groups
  - Group for each EHL (continuation) case
  - Group for each of the head processes of the groups
- Communication between groups needs only be done at end of  $\mathcal{R}_F$  evaluation
  - could use different Grid resources for each parallel group with slower TCP/IP communication between continuation runs

# Problem Solving Environment built in IRIS Explorer



# Problem Solving Environment - Loading map



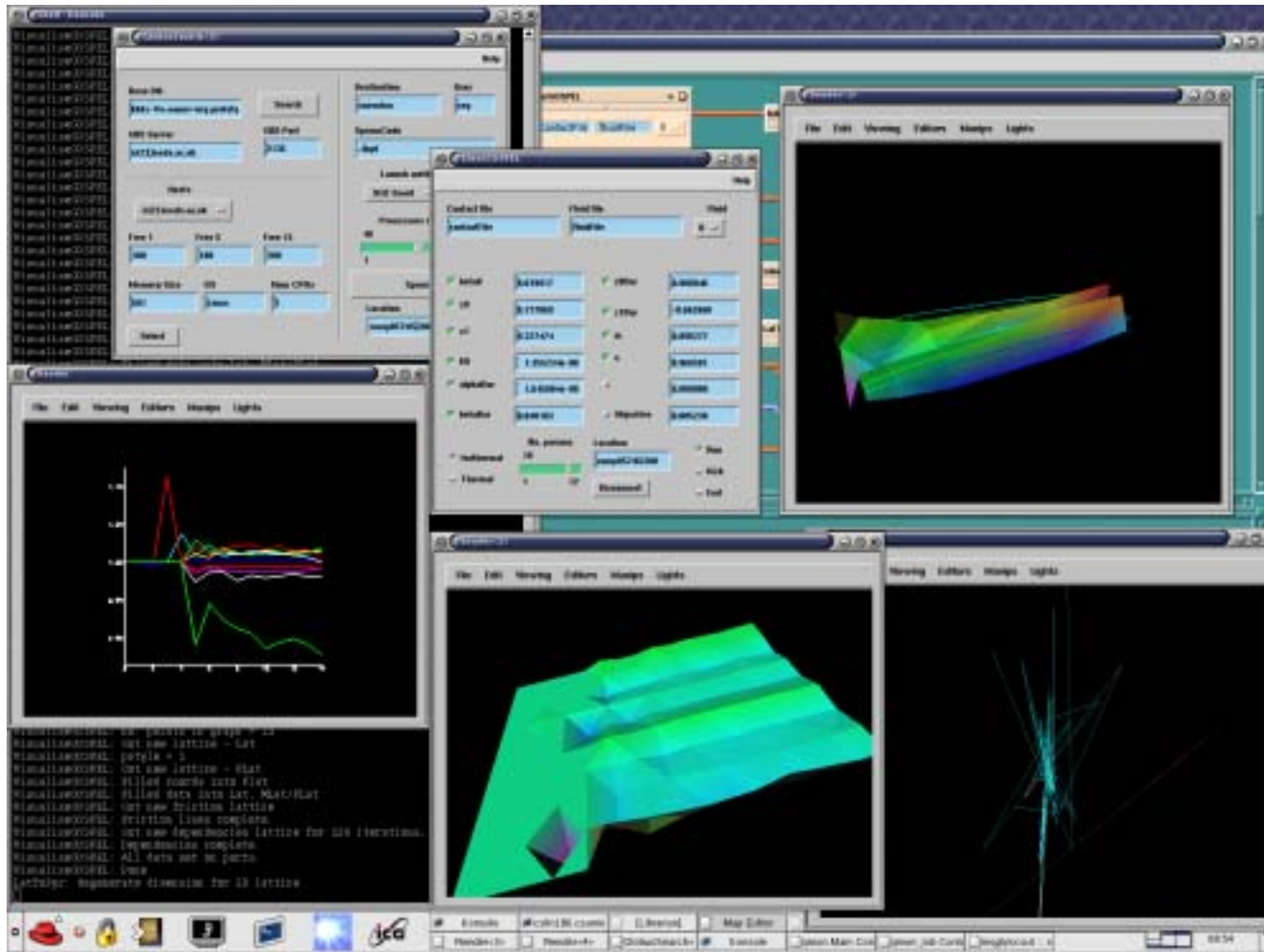
Grid certification already done

# Problem Solving Environment - Launching job

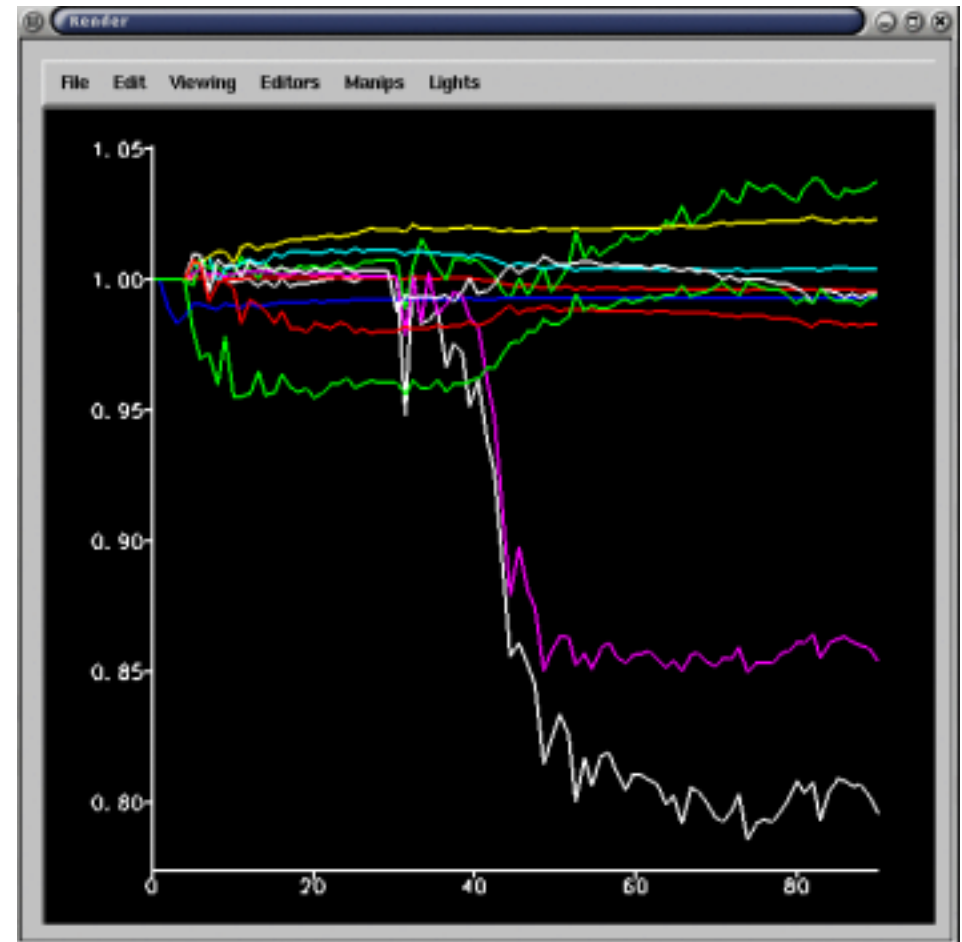
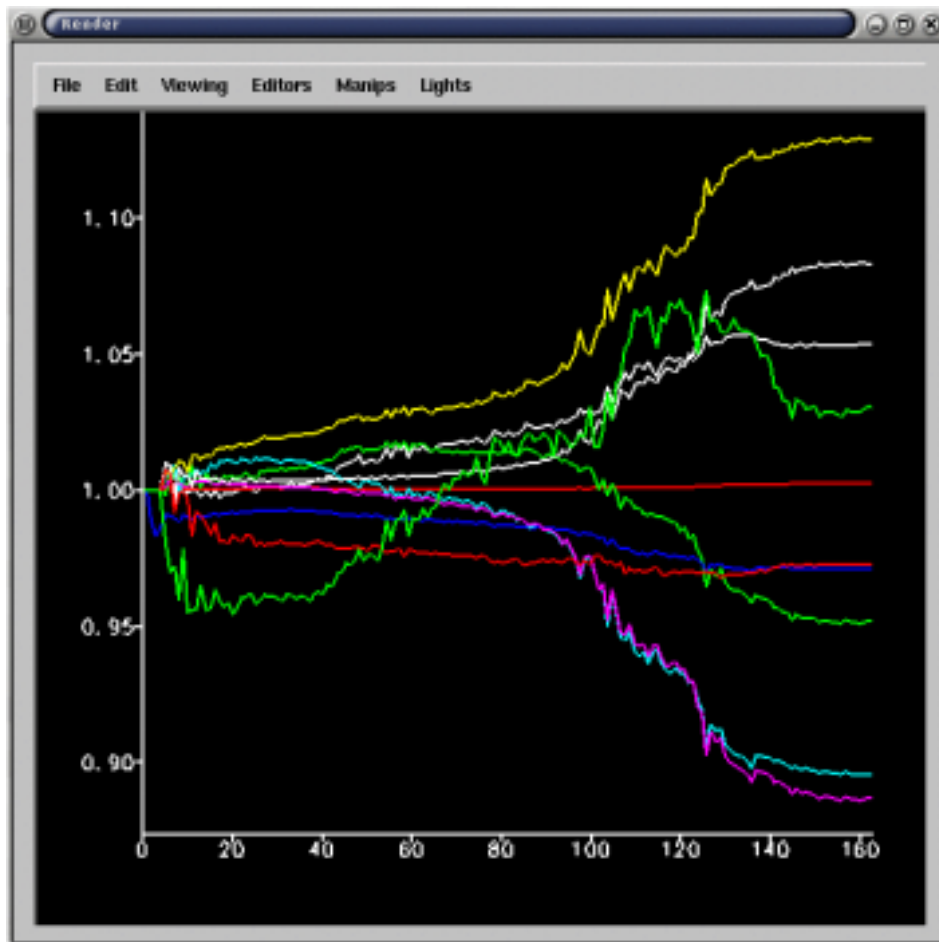
The screenshot shows a web-based interface for launching a job. It is divided into two main sections. The left section contains fields for Base DN (Mds-Vo-name=wrq prototy), GIS Server (iri23.leeds.ac.uk), and GIS Port (2135). Below these is a Hosts dropdown menu showing iri23.leeds.ac.uk. Further down are three columns for Free 1, Free 5, and Free 15, each with a value of 100. At the bottom of this section are fields for Memory Size (501), OS (Linux), and Num CPUs (1), along with a Select button. The right section contains fields for Destination (snowdon) and User (ceg). Below these is a SpawnCode field with the value ./opt. The Launch method is set to SGE Scout. A Progress bar for Processors required shows 48 out of 96, with a green bar extending to 48. A large Spawn button is located below the progress bar. At the bottom right is a Location field with the value comp057:65200. A Help button is located in the top right corner of the interface.

Search for available machines, choice of launching mechanism

# Problem Solving Environment - Visualisation



# Problem Solving Environment - Steering

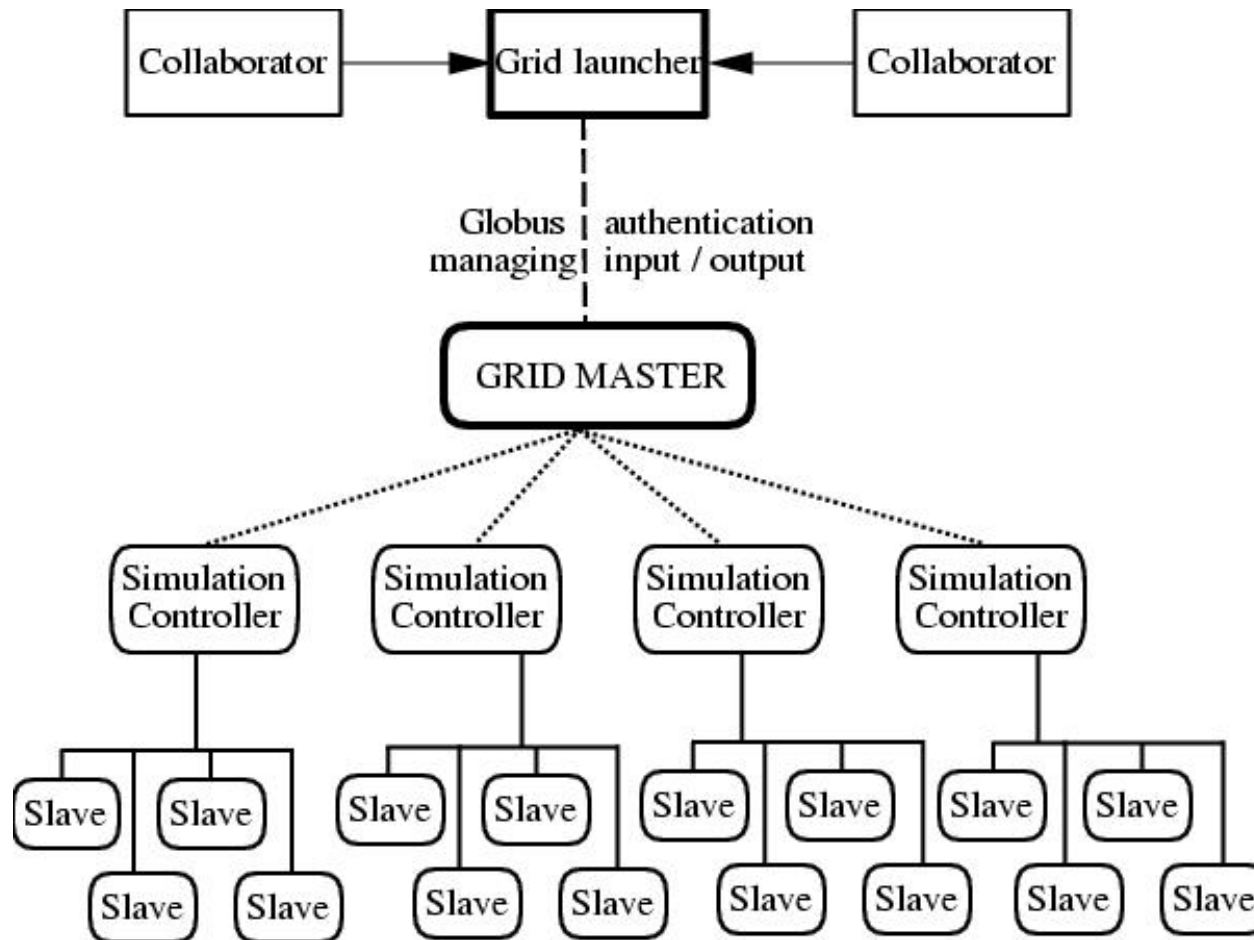


# PSE to Grid communication through gViz

**gViz** library is Grid middleware providing support for computational steering and visualization

- Insulates both simulation and visualisation developers from low level detail
- Connects distributed components of systems, transparently linking the desktop to the remote Grid resource
- Provides:
  - Functionality to securely launch processes onto the grid
  - Communications mechanisms between processes
  - Provides access to computed data for visualization and steering
  - Supports multiple, synchronised, user access

# Summary



# Summary

- A previously serial optimisation solver has been parallelised
- The numerical solver inside the optimiser has also been parallelised
- The optimiser has been made interactive through a PSE
- Various techniques used for representing high-dimensional data
- Grid enabling has added to the functionality without detracting from the interactivity of the steering