int.eu.grid

http://www.interactive-grid.eu

GridSolve: A nice tool for distributed computing

Marcus Hardt - Karlsruhe Institute of Technology

distributed computing is

when a computer

that you have never heard of

keeps yours from working correctly

Outline

- The Grid
  - Great, big, old
  - Problems
- Improvement 1: GridSolve
- Improvement 2: BURN

Grid Computing...

Idea: **Computer power <=> Electrical power**

From Electrical power grid => computational grid

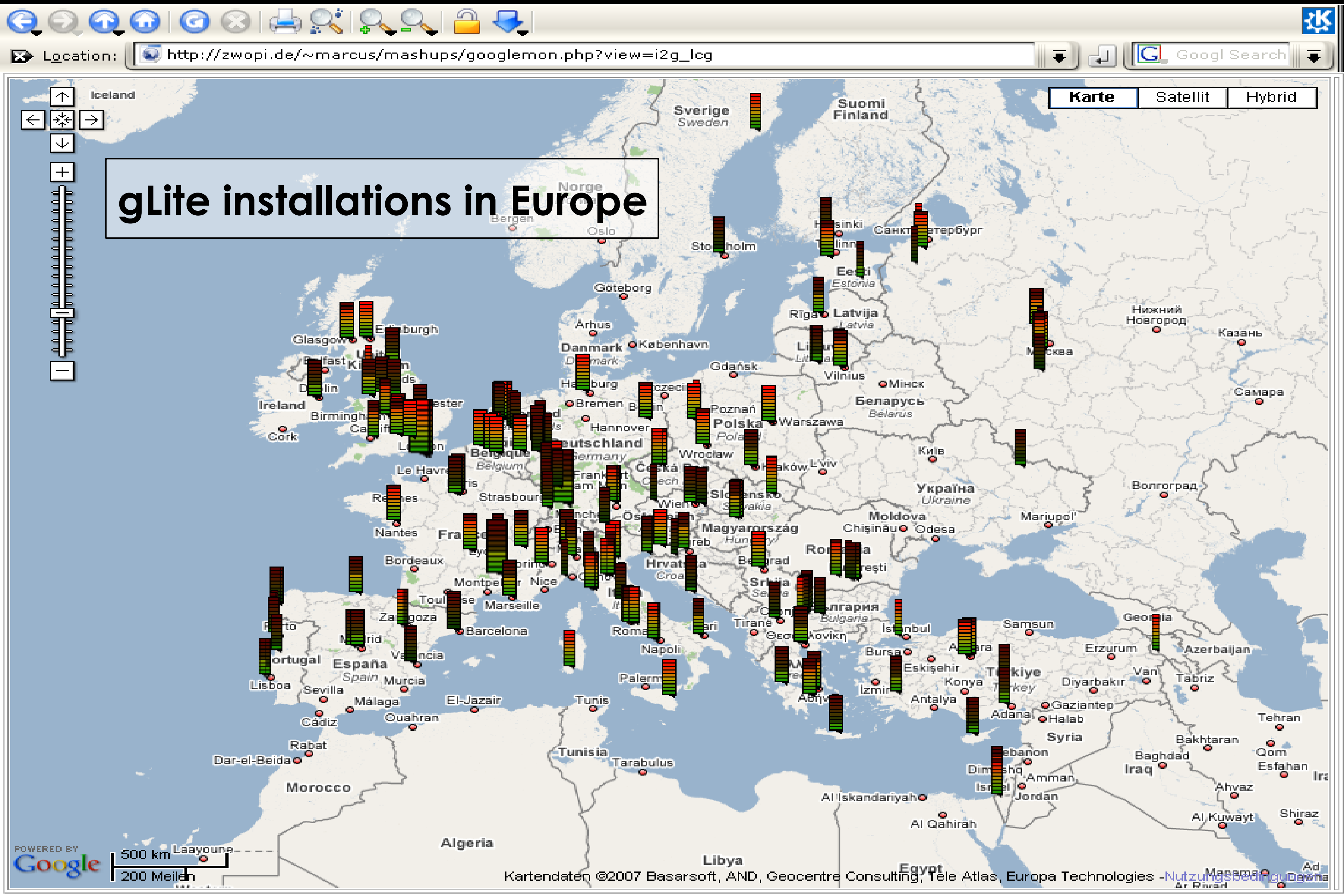
- Across organisationsal domains / countries
- Transparent access to
  - Computing
  - Data
  - Network
- Large scale installations

Grid middleware

- Middleware

  := Layer between application and operating system

- **gLite:** <u>one</u> grid middleware
  - Development driven by CERN
  - Tools for data+computing of new accelerator
  - 10 PB/year * 20 years, random access
- Paradigm: **Send job to where the data is**
- The trouble with jobs:
  - Self contained application to be sent
  - Unclear what software installed at destination
  - Long (3min) overhead for startup
  - No API-style access to remove resources (RPC)


Location: http://zwopi.de/~marcus/mashups/googlemon.php?view=i2g_lcg
Googl Search
Karte
Satellit
Hybrid
500 km
200 Meilen
Kartendaten ©2007 Basarsoft, AND, Geocentre Consulting, Tele Atlas, Europa Technologies - Nutzungsbedingungen

gLite installations in Europe

- Sites: 243 (in 49 countries)
- CPUS: 42798 (176 per site)
- RAM: 19TB
- RAM/CPU: 468MB
- DISK [Tot / Avail]: [8042TB / 5408TB] ([33892GB / 22792GB] per site)

0-9 CPUs --- 10-99 CPUs --- 100-999 CPUs --- > 1000 CPUs

Using a lightbulb in the glite world

- Describe the lightbulb

  Voltage, Watts, Amount
  Lighting_time, ...

- Submit request for electricity to broker

  => Powerplant automatically chosen for you

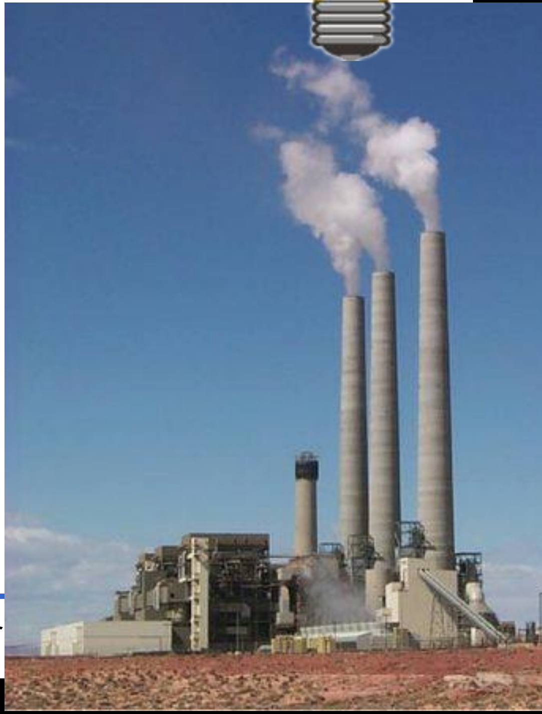
  => Send lightbulb to powerplant

  => Wait for electricity

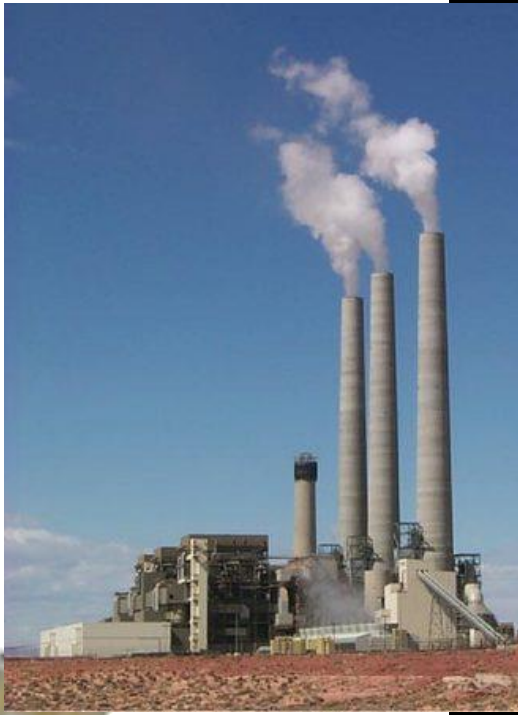
  => Lightbulb glows

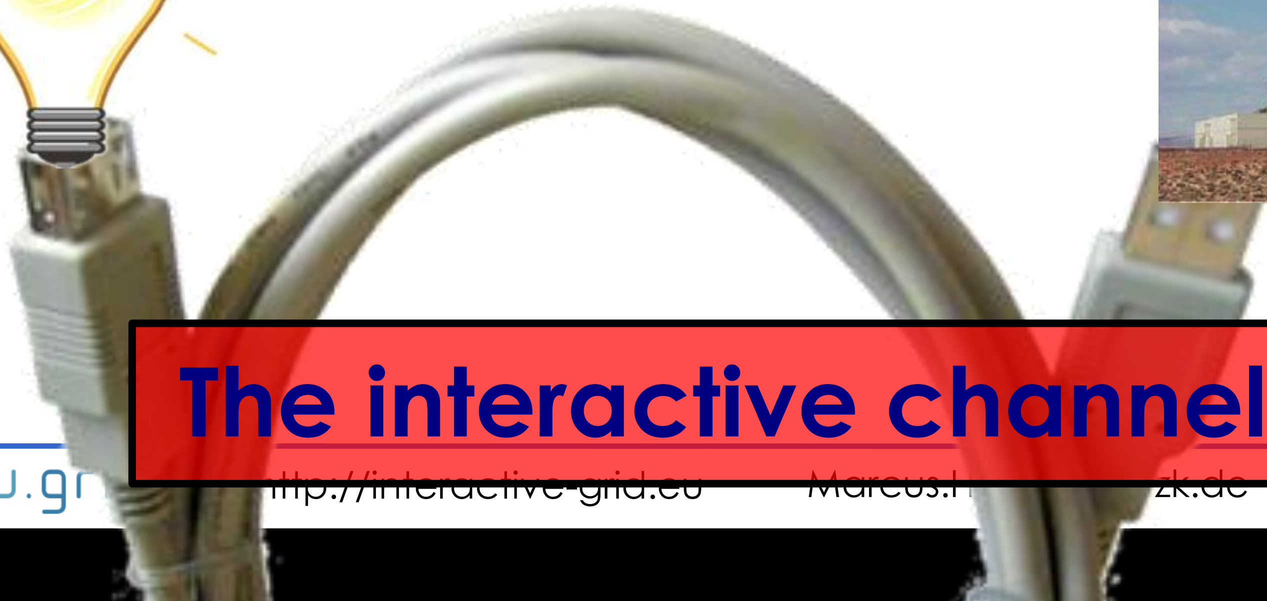
- Results come back

=> Too Slow



An idea for a solution



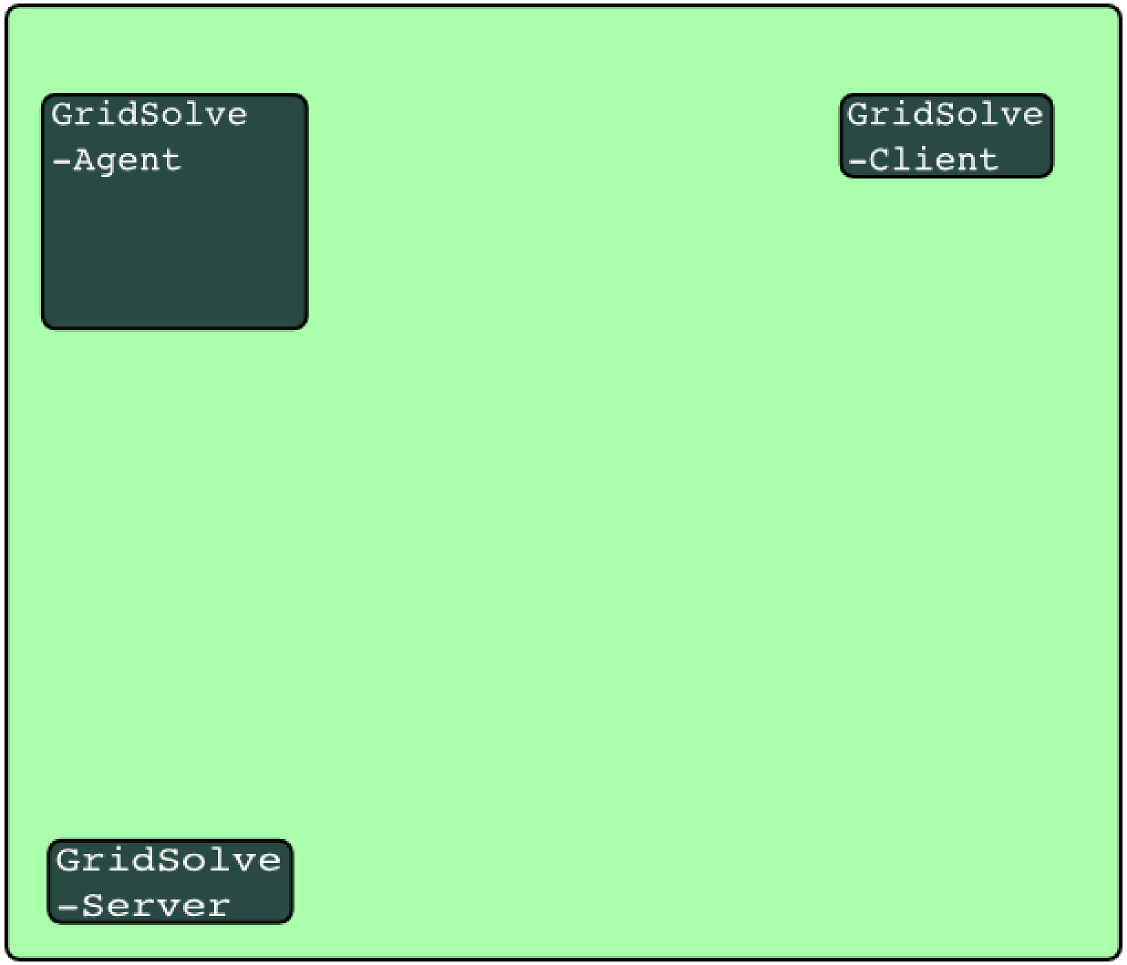


The interactive channel

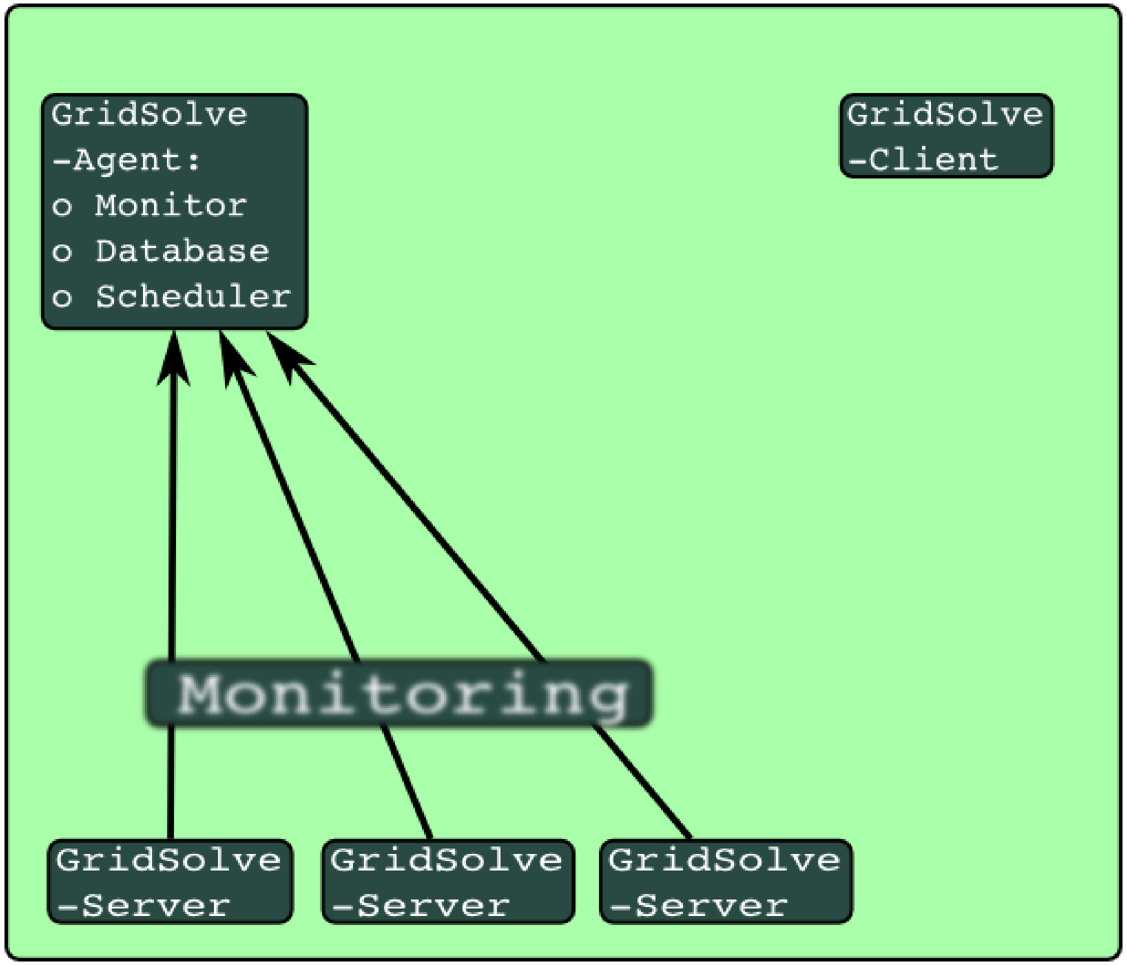
One possible Cable

- **GridSolve**
  - Developed at ICL, UTK, USA
  - A tool for RPC calls
  - Architecture-style:
    - Client <=> Agent <=> Server
  - Proxy support
    - To reach hosts behind a NAT

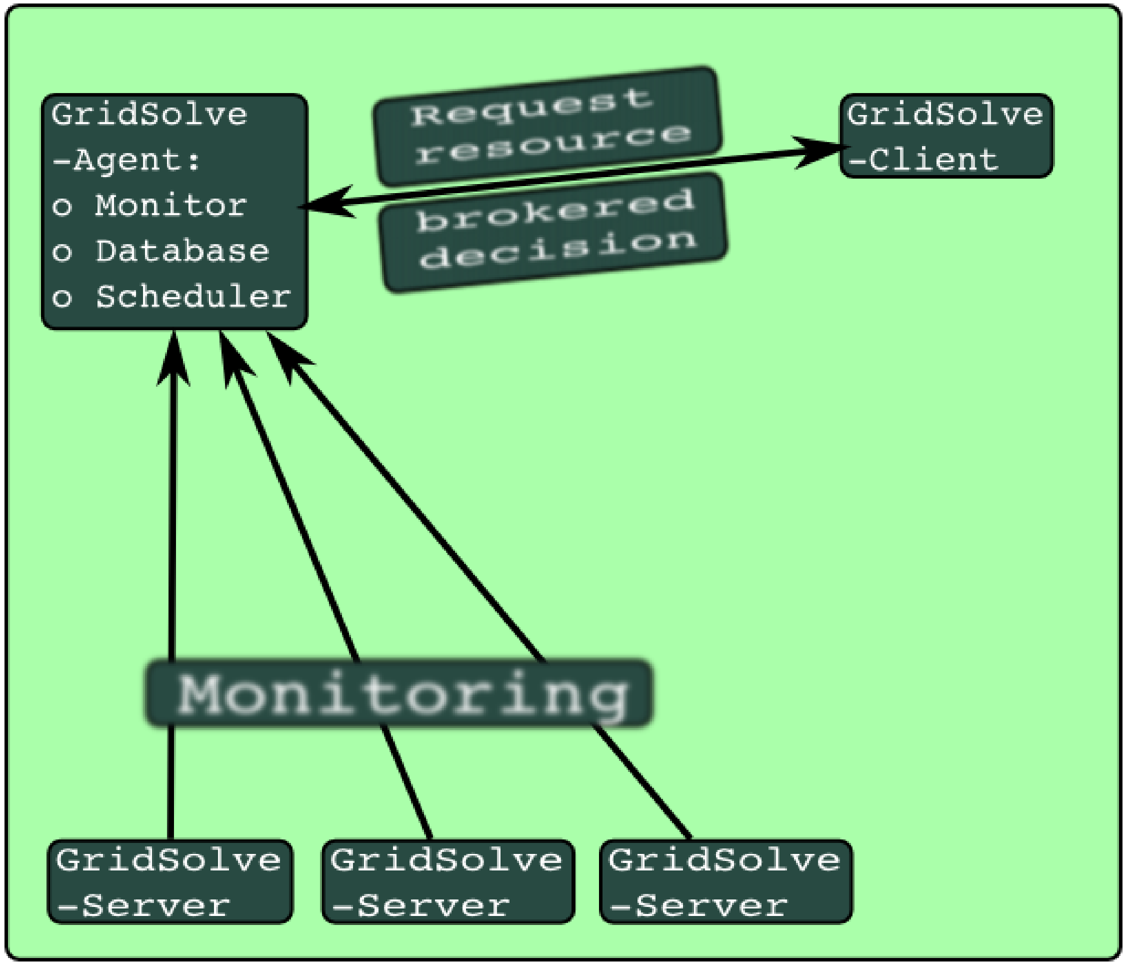
Gridsolve Architecture



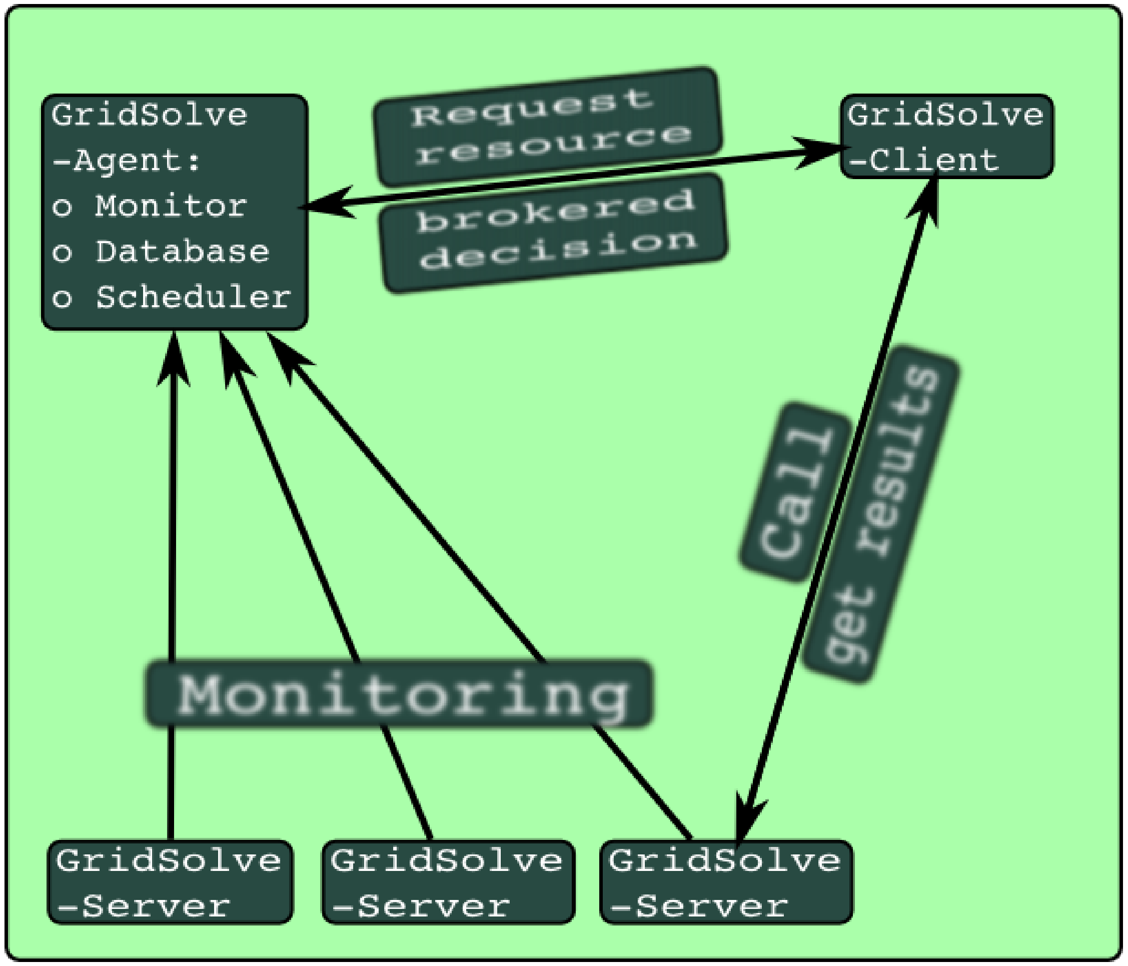
Gridsolve Architecture



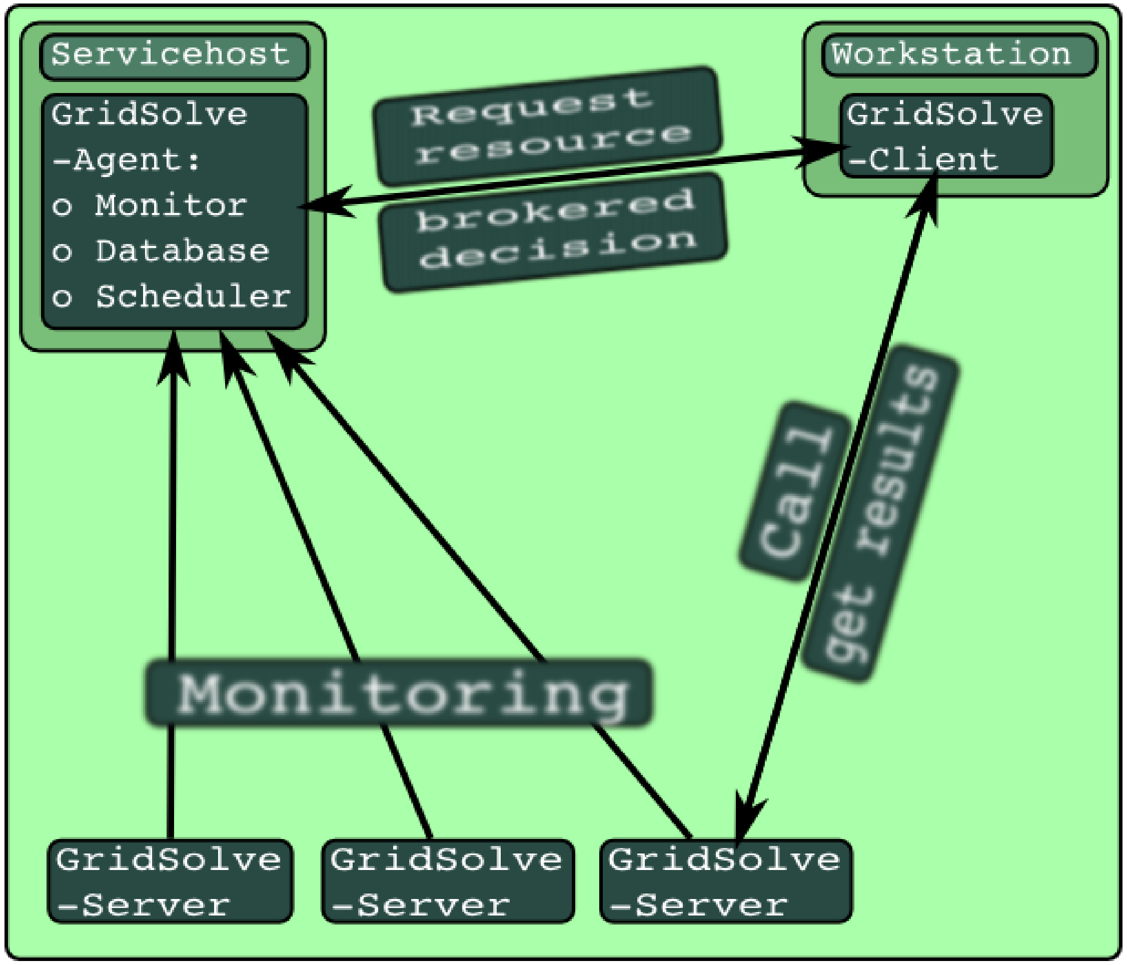
Gridsolve Architecture



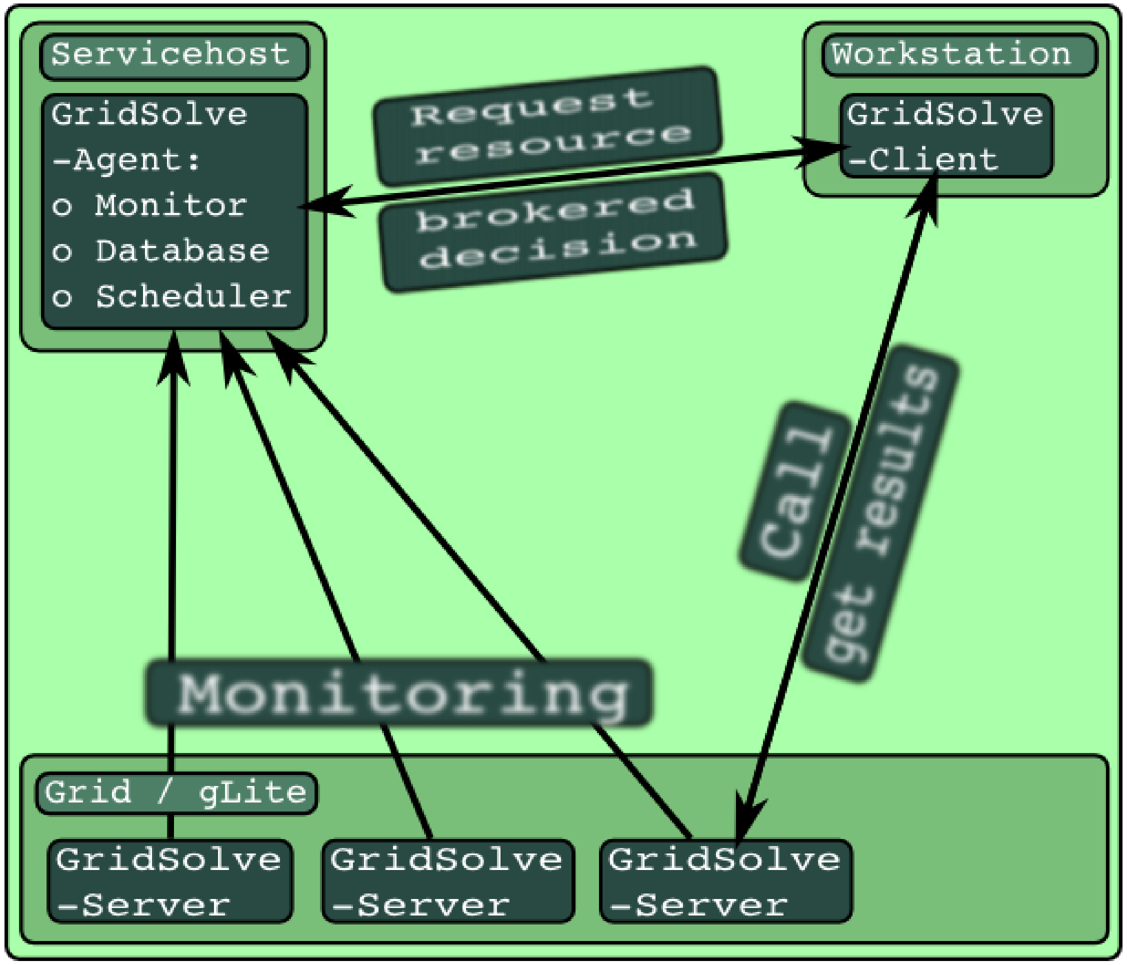
Gridsolve Architecture



Gridsolve Architecture



Gridsolve Architecture



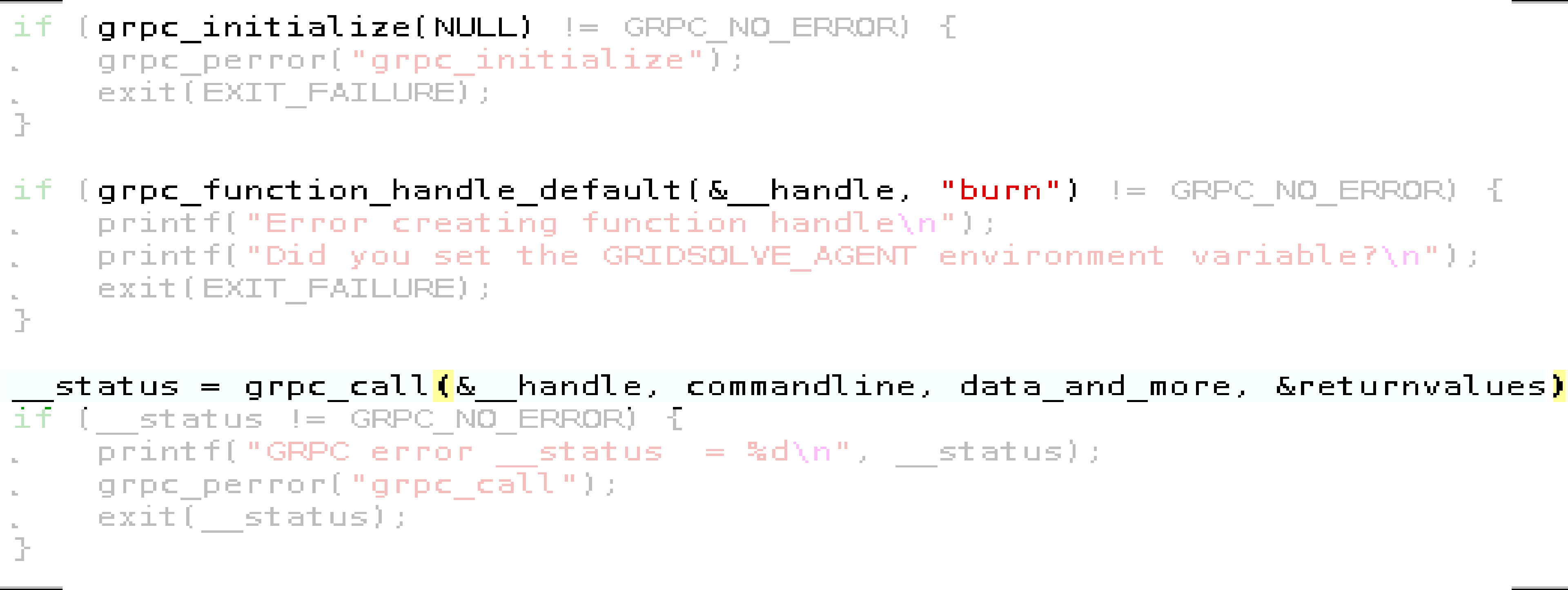
GridSolve interfaces

- Client interface for **C**, Fortran, **Matlab**, Octave
- Easy to use: (Example in Matlab code)

  y=problem(x) <=> y=gs_call('problem', x)
  - Transport input parameters to remote side
  - Execute “problem”
  - Transport result back
- Server executes C and Fortran libraries
  - Can be extended by the C-function **system**

=> Reduce complexity of the grid to one function call

Source code (C)



```c
if (grpc_initialize(NULL) != GRPC_NO_ERROR) {
    grpc_perror("grpc_initialize");
    exit(EXIT_FAILURE);
}

if (grpc_function_handle_default(&__handle, "burn") != GRPC_NO_ERROR) {
    printf("Error creating function handle\n");
    printf("Did you set the GRIDSOLVE_AGENT environment variable?\n");
    exit(EXIT_FAILURE);
}

__status = grpc_call(&__handle, commandline, data_and_more, &returnvalues);
if (__status != GRPC_NO_ERROR) {
    printf("GRPC error __status  = %d\n", __status);
    grpc_perror("grpc_call");
    exit(__status);
}
```

GridSolve Demo

- Application "**backpropagation**"
  - Analyses data
  - Returns an image
  - Part of my work
    - PhD about diffraction tomographic reconstruction for Ultra Sound Computer Tomography ;-)
  - Using gridsolve, I compute parts of the image at different computers
  - Development Environment: Matlab

=>Demo

Everyone Happy?

- Why not?
  - Modifications of RPC source involves
    - Recompilation of GridSolve
    - Recompilation of Backpropagation
      => Adds about 5 min to compilation time
  - Re start servers on the grid
    => Adds another 3-5 min of overhead
- ~10 mins ist just too much

Solution

- BURN
  - Bash Universal Remote Nurturer
  - RPC service for GridSolve that
    - Uses **system** to execute arbitrary shell commands
    - Downloads installation package on remote machine
    - Executes self-installed packages

BurnDemo

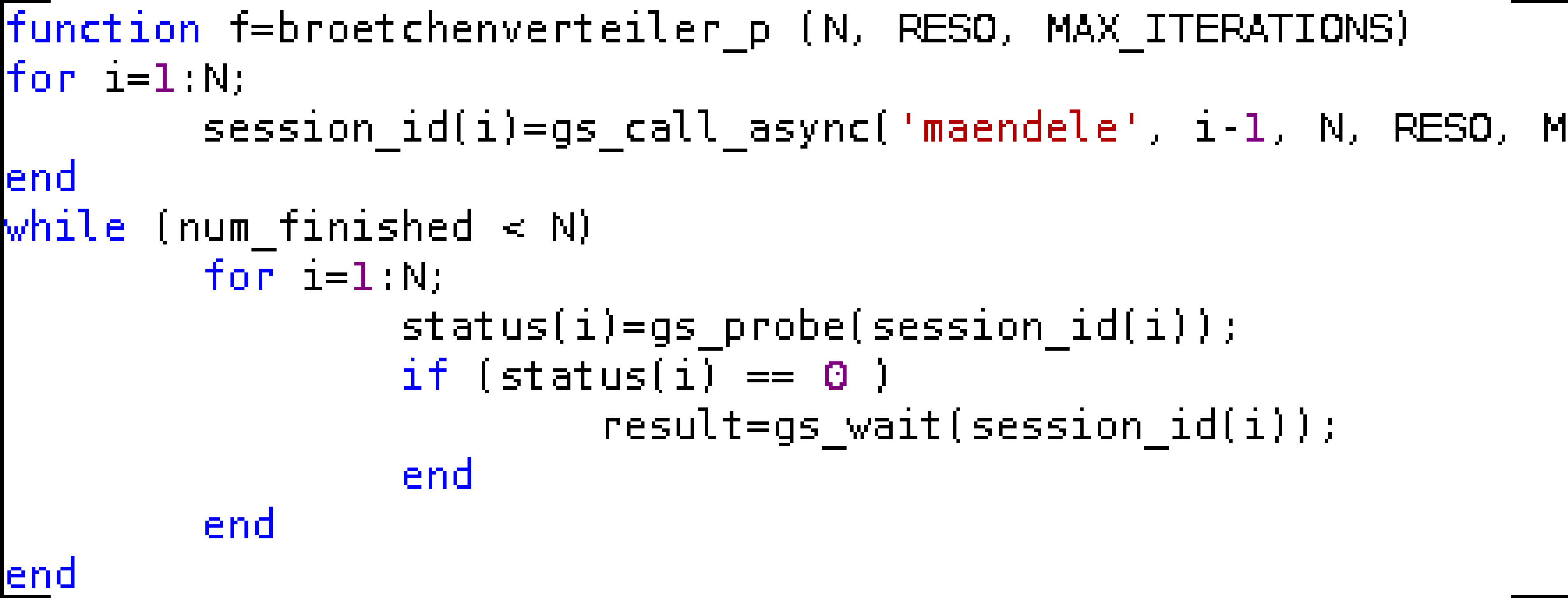
- Demo on int.eu.grid

Conclusion

- "The Grid" ("The Cloud")
  - A source for resources – not more!
  - Slow allocation, much overhead
- GridSolve
  - Provides RPC access to pool of resources
  - Bridges the NAT border
  - Long compile cycles
- BURN
  - Provides easy + generic access to resource pool
  - Reduces deployment time

?

Source code (Matlab)



```
function f=broetchenverteiler_p (N, RESO, MAX_ITERATIONS)
for i=1:N;
        session_id(i)=gs_call_async('maendele', i-1, N, RESO, M
end
while (num_finished < N)
        for i=1:N;
                status(i)=gs_probe(session_id(i));
                if (status(i) == 0 )
                        result=gs_wait(session_id(i));
                end
        end
end
```