Leiden **G**rid **I**nfrastructure
the future of computational chemistry

Hugo Meiland (LIC-ICT)
Mark Somers (LIC-THEOR)

Battle plan of this presentation:

- Some awarenesses.
- The true problems and challenges developers will be facing.
- Are computer GRIDs a possible solution?
- Grid middle ware.
- The 3 layer structure of LGI.
- Communication within LGI:
    XML
    HTTPS
- Where are we now.
- What's next?

Some awarenesses

- Research focus turns to results rather than methodology.
- Computers are more and more used as disposable objects.
- (PhD) Students are encouraged to spend more time on computing/research.
- The number of programs used is relatively low.

In contrast to:

- Computers are becoming more and more complex and diverse.
- Theories and algorithms are getting more advanced.
- Installing most of the (chemistry) software is not for the fainthearted.
- Using computers efficiently becomes harder to do.
- Doing both chemistry and computer science is no longer possible.

Some awarenesses

- Research focus turns to results rather than methodology.
- Computers are more and more used as disposable objects.
- (PhD) Students are encouraged to spend more time on computing/research.
- The number of programs used is relatively low.

More users...

In contrast to:

- Computers are becoming more and more complex and diverse.
- Theories and algorithms are getting more advanced.
- Installing most of the (chemistry) software is left for the fainthearted.
- Using computers efficiently becomes harder to do.
- Doing both chemistry and computer science is no longer possible.

Less developers !

The problems developers will be facing

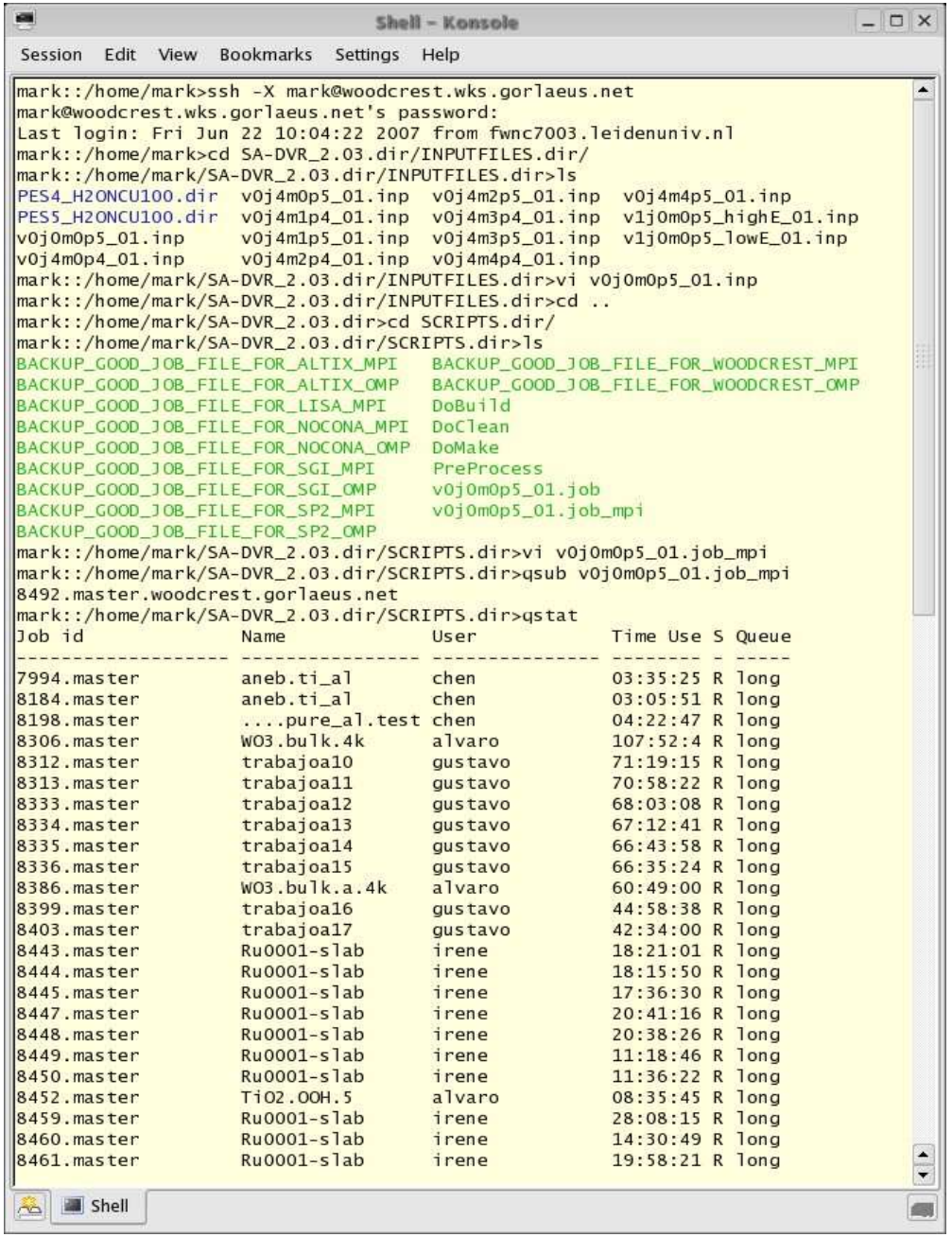
The program serves a **purpose** and **users** determine program's existence:

- Need to make programs more **intuitively** and **accessible** for the most common tasks performed by the program so that they are easier to use.

- Need to disentangle program interface from hardware details so user needn't worry about queues, nodes, security and all the complexity of setting up the program on each type of hardware.

- Need to provide **scalability** and **continuity** for the programs to support more users in future and to match the demand for more hardware again ***without interfering the users***.

For example... The usual way of starting a calculation...



1) Start with logging into cluster
2) Setup program
3) Edit input file
4) Edit PBS script file
5) Submit calculation
6) Regularly check job

Only input file is independent of cluster and the hardware used!

For example... wouldn't it be really cool if...

New job for m.somers at 22 Mar 2007 9:16:47 UTC

New job specifics

Application: Classical

Name of job (no spaces, quotes or slashes): test_sommeke

Input:

```
interaction bending f=0.175 deg=111.2   ( 3 4 400 )
interaction bending f=0.175 deg=111.2   ( 3 4 401 )
interaction bending f=0.175 deg=111.2   ( 3 4 402 )

# the bonding interactions of the first CH3
interaction harmonic f=1.195 r0=3.76    ( 4 400 )
interaction harmonic f=1.195 r0=3.76    ( 4 401 )
interaction harmonic f=1.195 r0=3.76    ( 4 402 )

# ----------------------------------------
# find stable situation first
conformation n=10000 error=1.0E-4 maxstep=2.5

# ----------------------------------------
# then setup a constant temperature
temperature k=3.166829379841521e-06 constant

# and finaly the dynamics statement, run 100
dynamics dt=4.13411 tend=4134110.5461 t=0.0
```

Estimated time to completion: 3 h.

Estimated memory usage: 4 Mb.

Estimated disk usage: 4 Mb.

Submit Job

Commands

Run Classical-Builder

Go back to your queue

Log out

... you could do that with:

- ADF
- Gaussian
- VASP

...

Personal job list of m.somers at 22 Mar 2007 9:08:45 UTC

You have 6 jobs listed !

Job #	Job submit time	Job status	Job name	Job ID
1	16 Mar 2006 12:22:58 UTC	finished	test_trajtou_pd110paw	88811
2	16 Mar 2006 12:22:01 UTC	finished	test_trajtou_pt111	88804
3	23 Oct 2006 13:05:57 UTC	finished	test_ABC	1534582
4	20 Feb 2007 7:27:46 UTC	finished	test_butaan	1854209
5	9 Feb 2007 14:21:34 UTC	finished	test_pd_again	1824217
6	26 Feb 2007 13:23:15 UTC	finished	test_pd_again	1868383

Commands

You can submit 84 more jobs: Submit a job

Your account

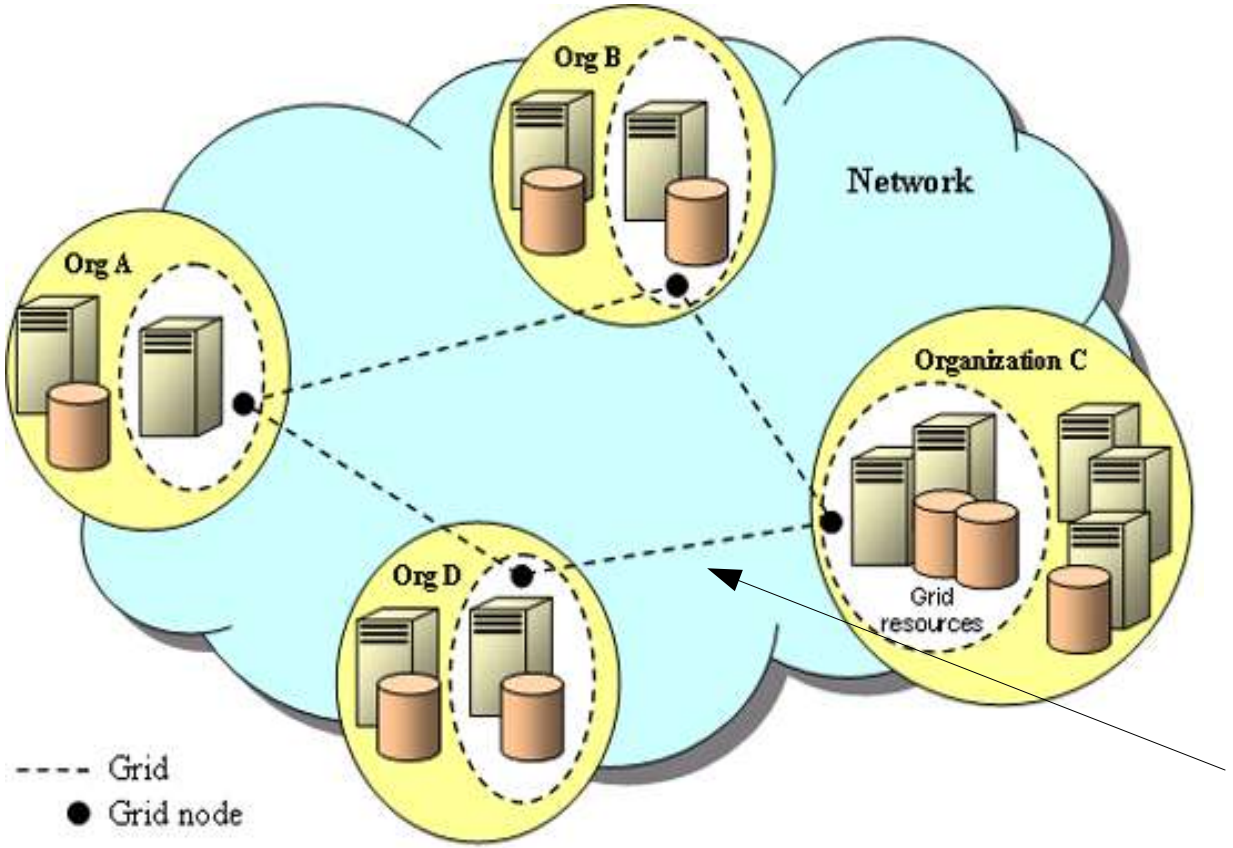
Your computers

Log out

Return to Leiden Classical main page

Are computer grids the answer?

A GRID is a collection of different types of computers, all connected to each other, that are owned and administrated by several people, that are capable of performing tasks and can be regarded as a whole.

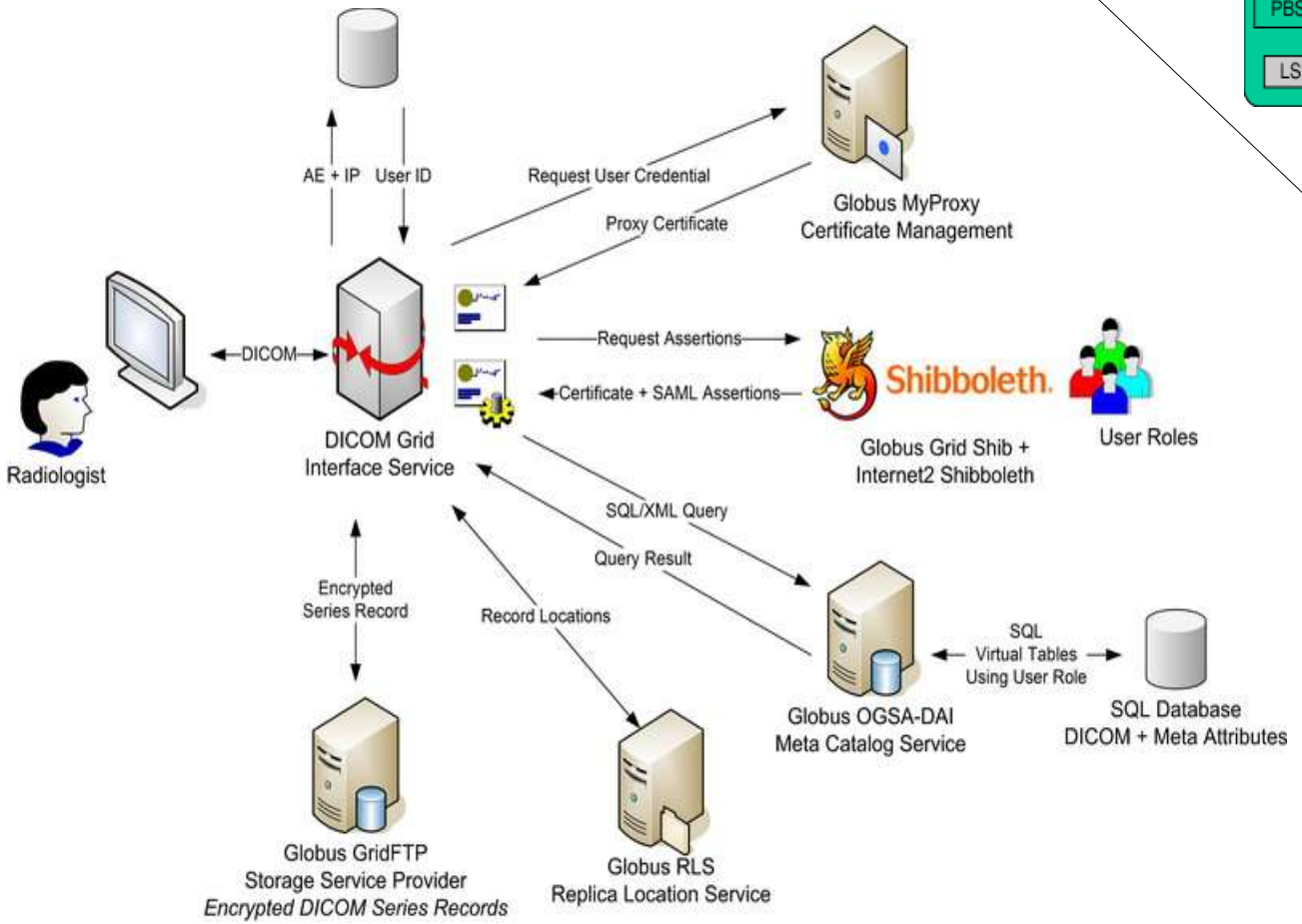


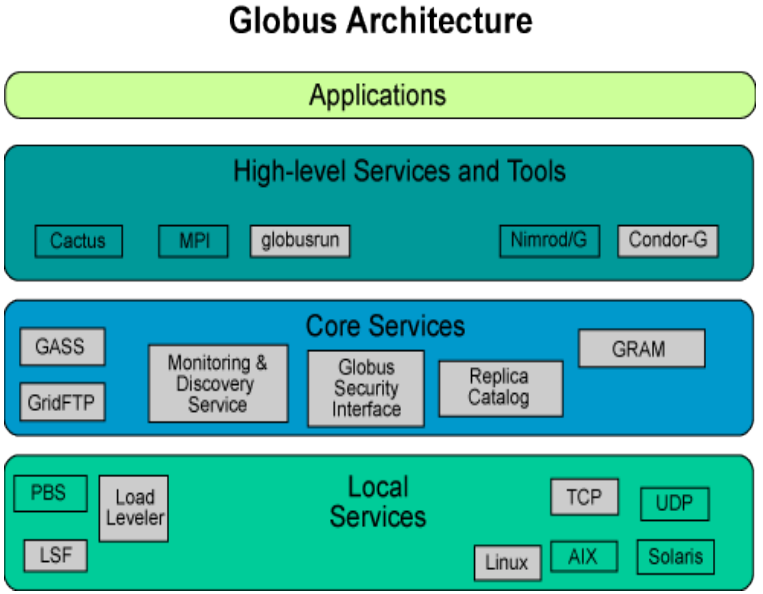
Grids do allow for **scalability** and **continuity** but need "**middle ware**"...

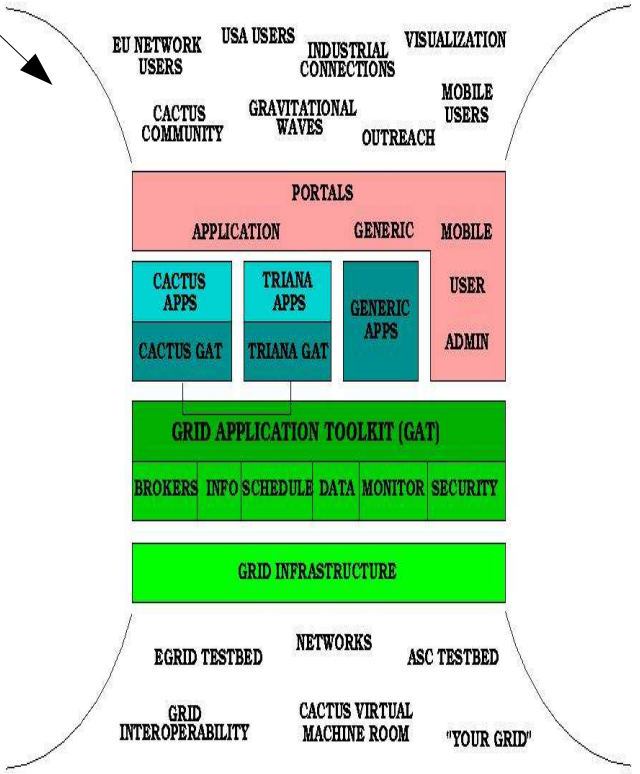
Grid middle warez...

like GLOBUS and GAT...

allows you to setup something like this...







Grid middle warez...

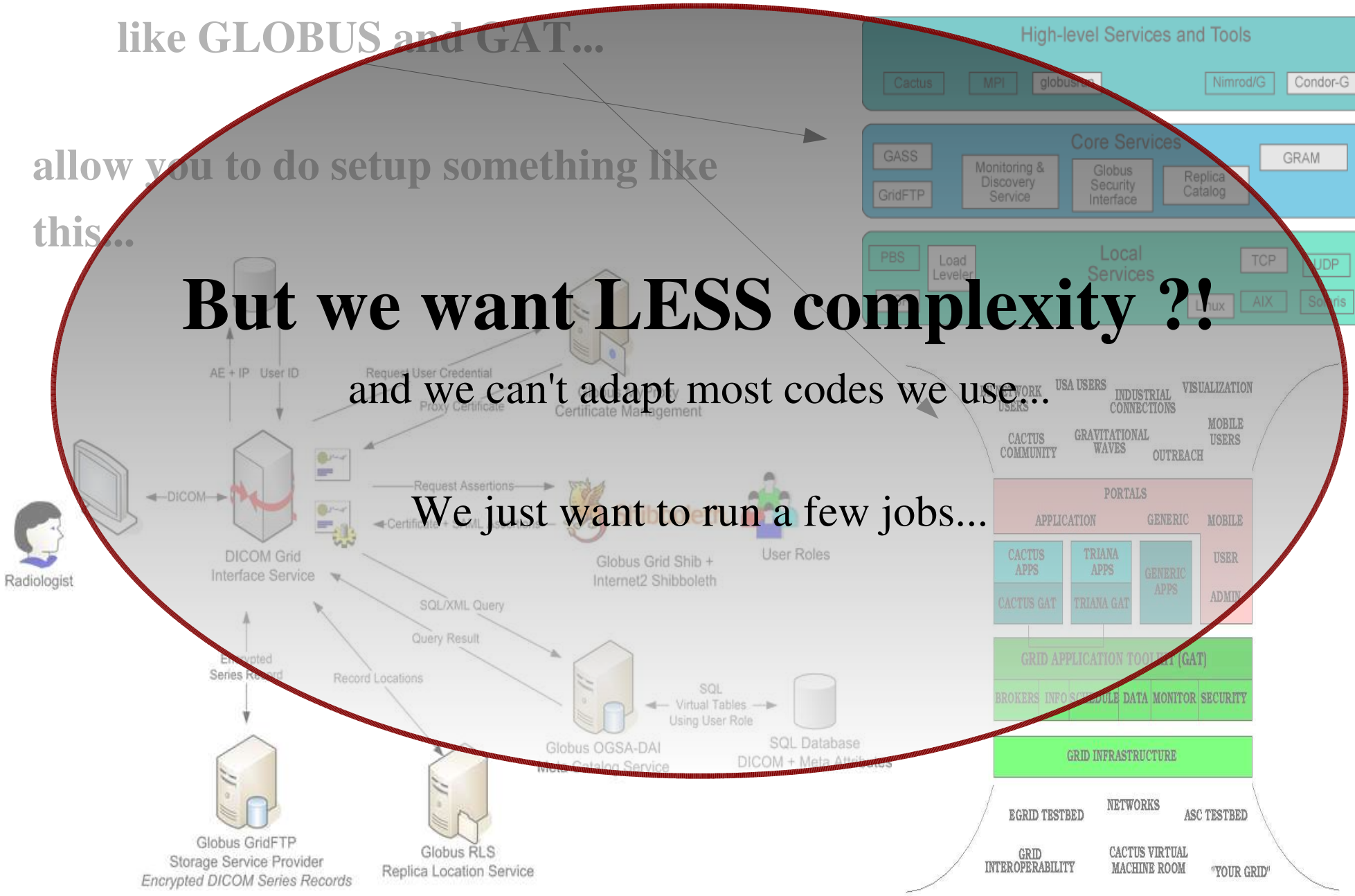
like GLOBUS and GAT...

allow you to do setup something like this...

But we want LESS complexity ?!

and we can't adapt most codes we use...

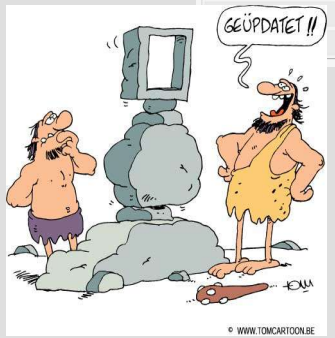
We just want to run a few jobs...



Enter LGI... The (lightweight) grid middle ware we need...

USER LAYER

How to bridge the gap?



HARDWARE LAYER

Enter LGI... The (lightweight) grid middle ware we need...

USER LAYER

COMMUNICATION LAYER

Insert a new layer that is:

- General
- Scalable
- Extendable
- Easy to administer and maintain

HARDWARE LAYER

Enter LGI... The (lightweight) grid middle ware we need...

USER LAYER

COMMUNICATION LAYER

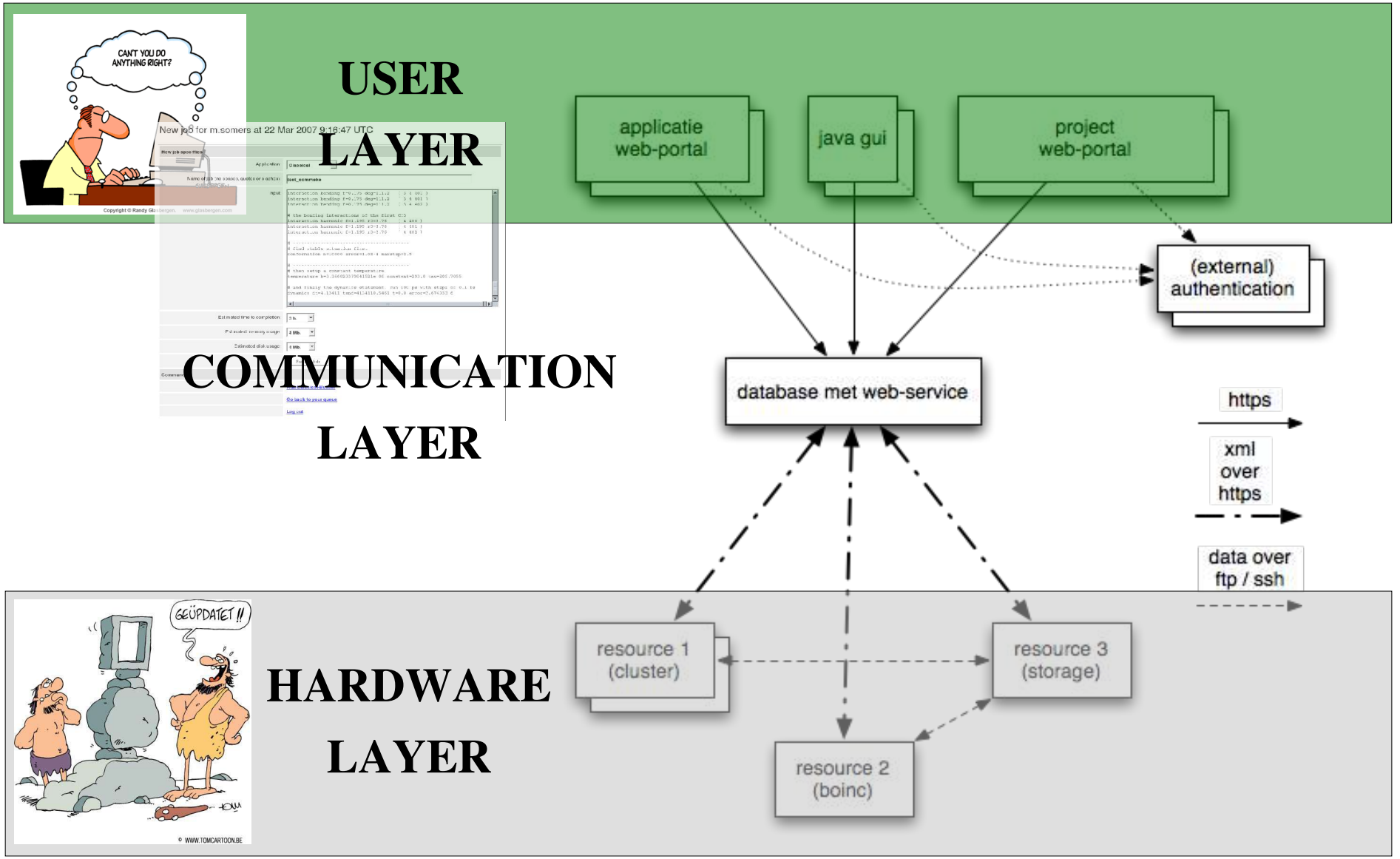
Insert a new layer that is:

- General
- Scalable
- Extendable
- Easy to administer and maintain

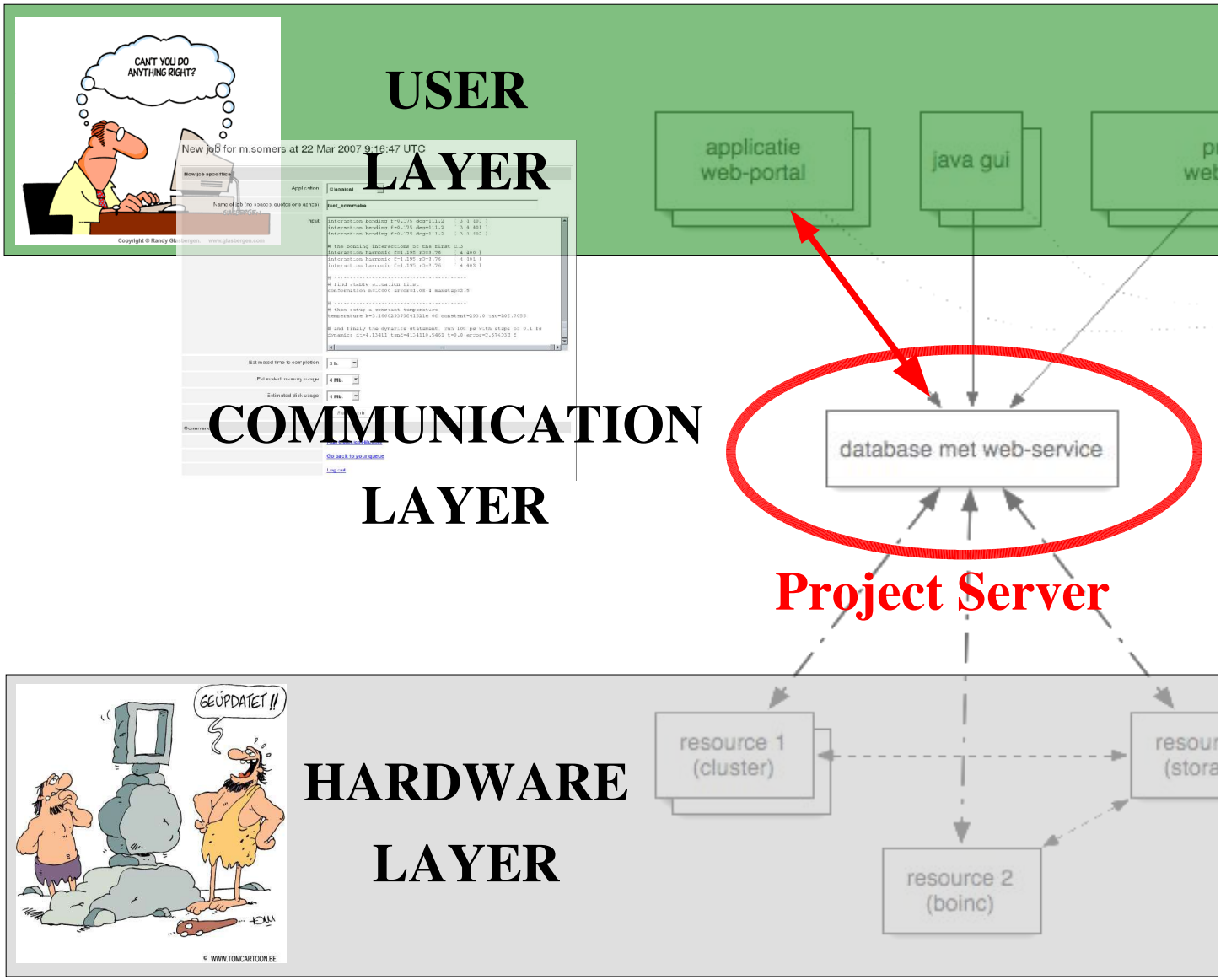
HARDWARE LAYER

Very important that standard UNIX administrators can work with this using techniques they are already familiar with!

Enter LGI... The (lightweight) grid middle ware we need...

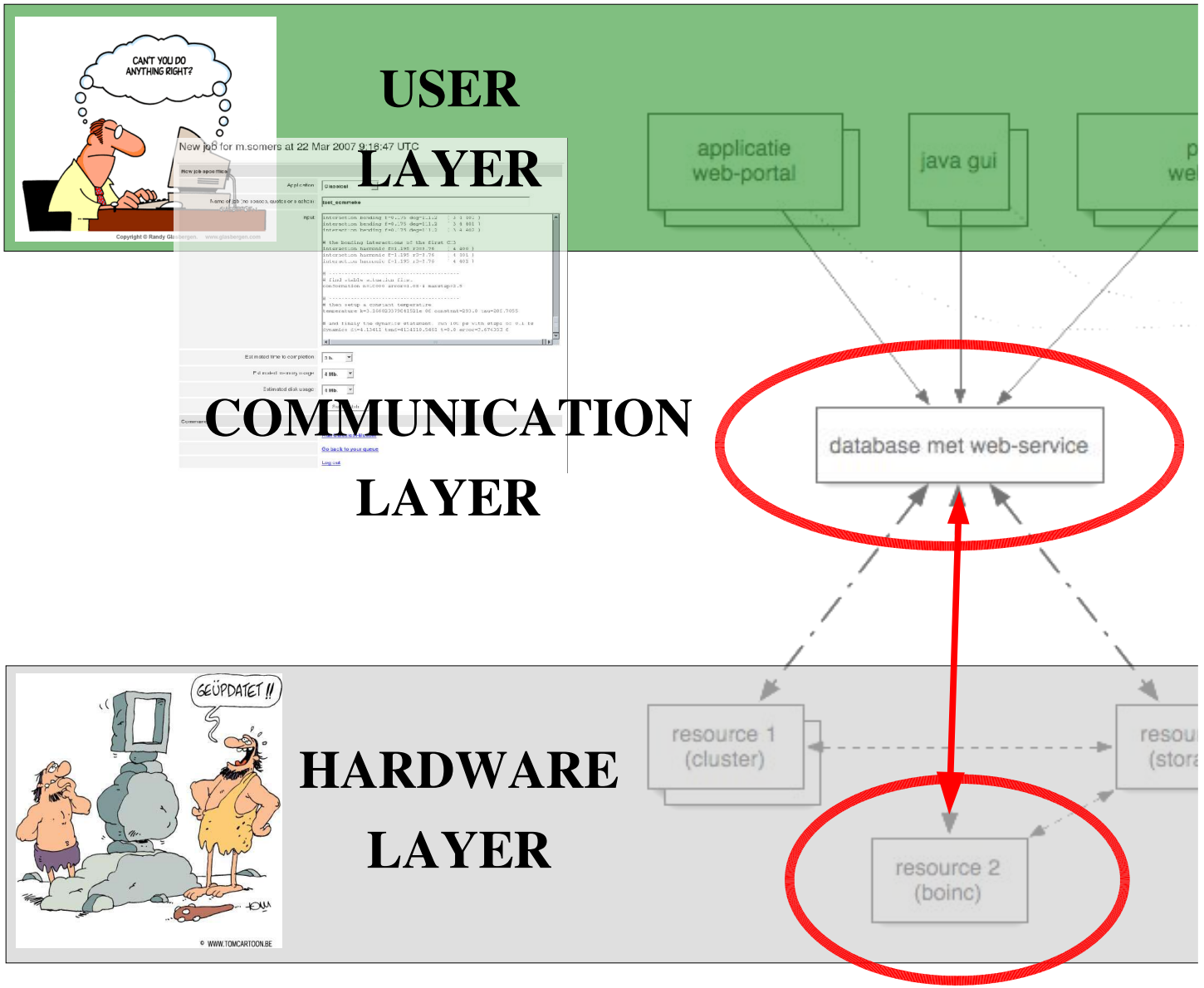


Enter LGI... The (lightweight) grid middle ware we need...



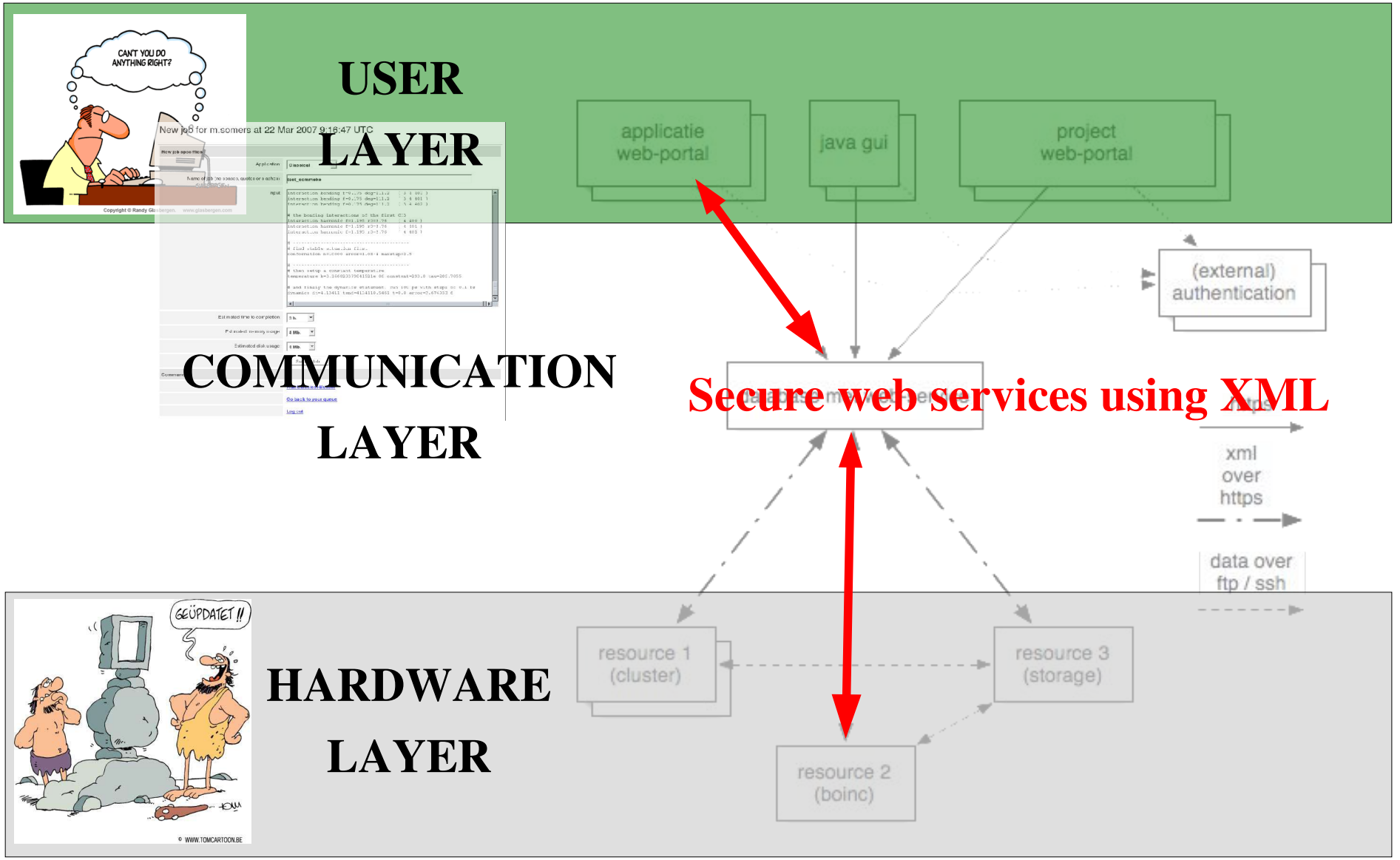
Users submit jobs to database on project server and check for results by using a browser...

Enter LGI... The (lightweight) grid middle ware we need...



Resources request work they can handle from database on project server and report back the results...

Communication within the LGI...



Communication within the LGI...

What is e**X**tensible **M**arkup **L**anguage?

A format that allows to markup or encapsulate data suitable for transferring and extending...

It uses tags:

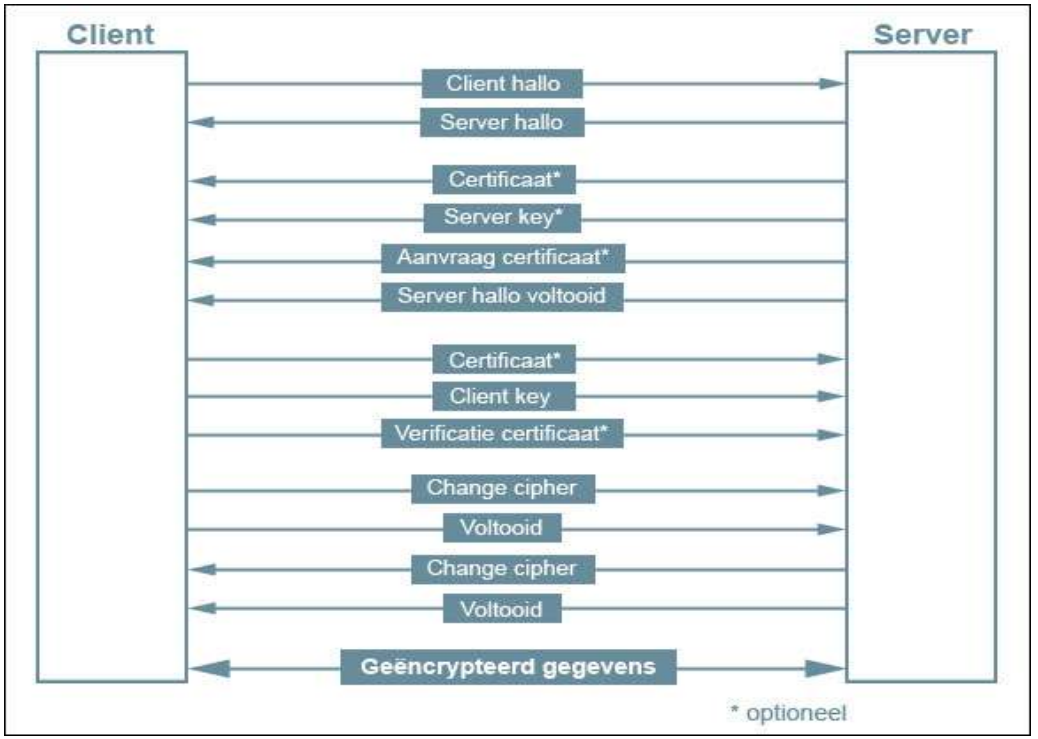
```
<LGI>
     <CA_certificate> http://www.LGI.org/LGI-CA.crt </CA_certificate>
     <response>
          <project> testproject </project>
          <project_master_sever> https://fwnc7003.leidenuniv.nl/testmasterserver </project_master_server>
          <this_project_sever> https://fwnc7003.leidenuniv.nl/testslaveserver </this_project_server>
          <user> mark </user>
          <groups> teras, cyttron </groups>
          <job>
               <job_id> 147 </job_id>
               <application> testapp </application>
               <state> queued </state>
               <target_resources> any </target_resources>
               <owners> sjoerd, cyttron </owners>
               <read_access> any, sjoerd, cyttron </read_access>
               <state_time_stamp> 1259936661 </state_time_stamp>
               <job_specifics> </job_specifics>
               <input> CDEF9021569C8787E </input>
          </job>
     </response>
</LGI>
```

Communication within the LGI...

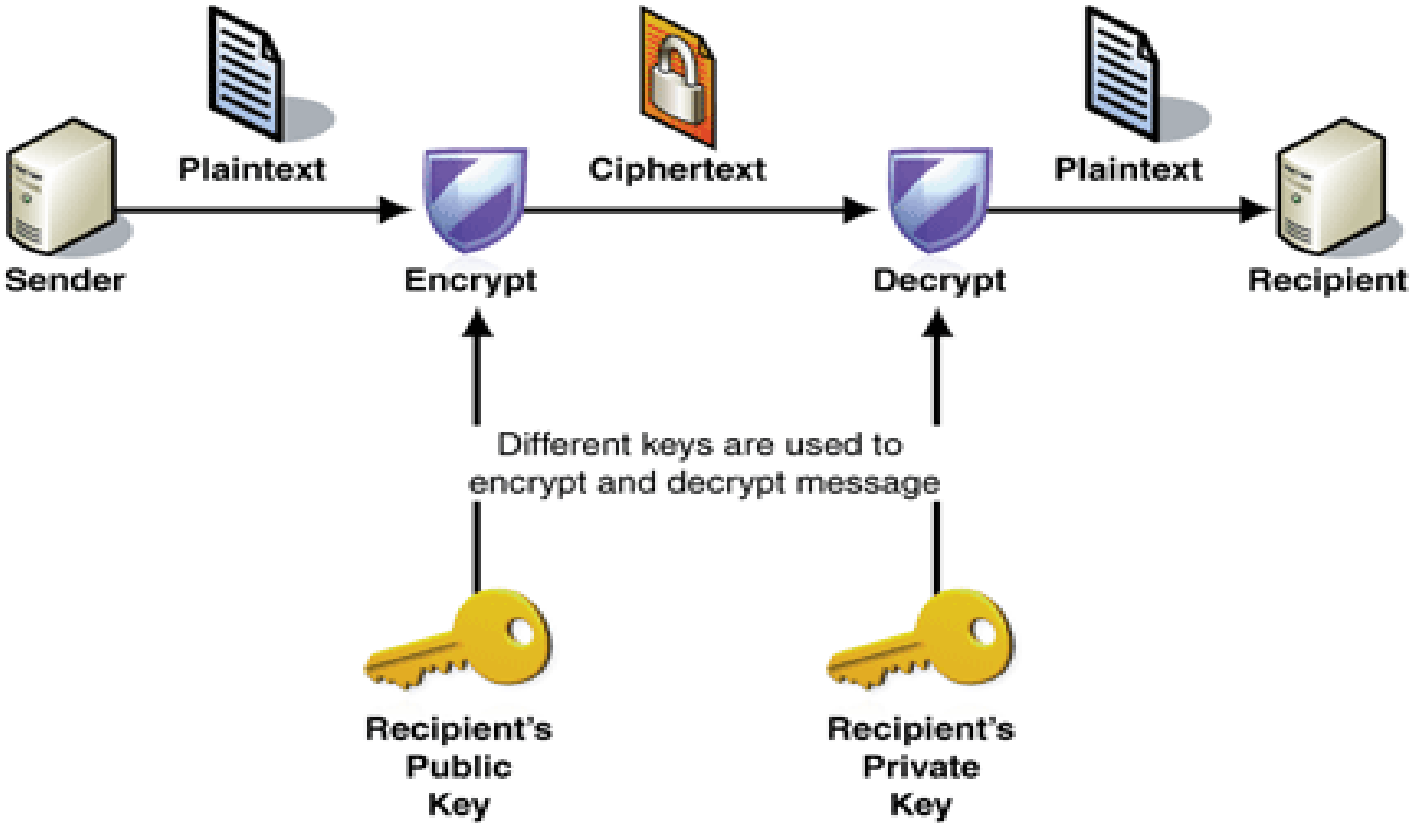
What is **H**yper **T**ext **T**ransport **P**rotocol **S**ecure?

A protocol to transfer data between computers securely using **S**ecure **S**ocket **L**ayer technology...

It uses **certificates** and **asymmetric** encryption:



Asymmetric encryption (RSA)...

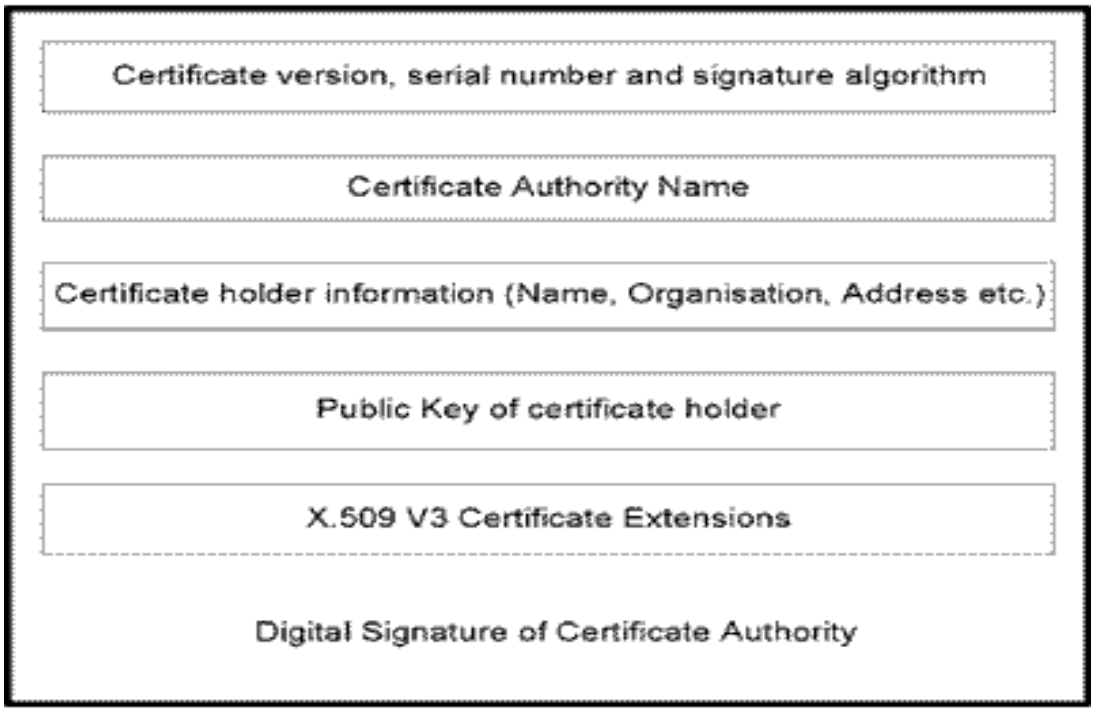


If we are sure that we use the recipients public key, the data can only be read by that recipient!

Assurance is guaranteed by using certificates!

x509 Certificates...

Piece of data that stores the public key and the identity of a recipient:



Certificate has been 'signed' by a trusted third party!

'Signing' by encrypting the fingerprint with the private key of trusted party.

Where are we now...

Project server software and communication has been implemented using:

- **Apache web server** using SSL (https) with client- and server-certificates.
- API for 'Interface'↔'Project server', 'Project server'↔'Resource' and 'Project server'↔'Project server' communication using XML in **PHP**:

interface_submit_job
interface_job_state
interface_delete_job
interface_project_server_list
interface_project_resource_list

resource_signup_resource
resource_request_work
resource_job_details
resource_lock_job
resource_unlock_job
resource_update_job
resource_submit_job
resource_job_state
resource_signoff_resource
resource_get_resource_details

server_get_update
server_run_update

- **MySQL database** as back-end on 'Project server' with user management tables.

Where are we now...

'**Resource daemon**' has been implemented for UNIX systems in **C++ using STL and libCURL**:

```
<LGI>
        <ca_certificate_file> ../certificates/LGI+CA.crt </ca_certificate_file>
        <resource>
                <resource_certificate_file> ../certificates/fwnc7003.crt </resource_certificate_file>
                <resource_key_file> ../certificates/fwnc7003.key </resource_key_file>
                <run_directory> ./runhere </run_directory>

                <owner_allow> <any> 10 </any> </owner_allow>
                <owner_deny> nobody </owner_deny>

                <number_of_projects> 1 </number_of_projects>

                <project number='1'>
                        <project_name> LGI </project_name>
                        <project_master_server> https://fwnc7003.leidenuniv.nl/LGI </project_master_server>

                        <owner_allow> <any> 5 </any> </owner_allow>
                        <owner_deny> nobody </owner_deny>

                        <number_of_applications> 1 </number_of_applications>

                        <application number='1'>
                                <application_name> hello_world </application_name>

                                <owner_allow> <any> 2 </any> </owner_allow>
                                <owner_deny> nobody </owner_deny>
                                <max_output_size> 4096 </max_output_size>

                                <check_system_limits_script> ./hello_world_scripts/check_system_limits_script </check_system_limits_script>
                                <job_check_limits_script> ./hello_world_scripts/job_check_limits_script </job_check_limits_script>
                                <job_check_running_script> ./hello_world_scripts/job_check_running_script </job_check_running_script>
                                <job_check_finished_script> ./hello_world_scripts/job_check_finished_script </job_check_finished_script>
                                <job_prologue_script> ./hello_world_scripts/job_prologue_script </job_prologue_script>
                                <job_run_script>./hello_world_scripts/job_run_script </job_run_script>
                                <job_epilogue_script> ./hello_world_scripts/job_epilogue_script </job_epilogue_script>
                                <job_abort_script> ./hello_world_scripts/job_abort_script </job_abort_script>
                        </application>
                </project>

        </resource>
</LGI>
```

http://fwnc7003.leidenuniv.nl/LGI/daemon

Where are we now...

'Resource daemon' already runs on:

Aster	(linux IA64)
Huygens	(linux IBM PPC)
Woodcrest	(linux EM64T)
Nocona	(linux EM64T)
My laptop	(linux IA32)
Hugo's laptop	(FreeBSD Mac OSX IA32)

https://fwnc7003.leidenuniv.nl/LGI/daemon

Basic interface in PHP has been built:

https://fwnc7003.leidenuniv.nl/LGI/basic_interface

What's next?

1) Implementing the secure 'file transfer programs'.

2) Testing phase and do more that just 'hello world' ;-).

like... run ADF and Gaussian ?!

Thank you for your attention... any further questions?

https://fwnc7003.leidenuniv.nl/LGI/docs/LGI.pdf