Integrating various Grid resource managers with GT4

the early experience

Pawel Plaszczak
Dominik Lupinski

www.gridwisetech.com
{pawel,dominik}@gridwisetech.com

Note

This project is a work-in-progress.

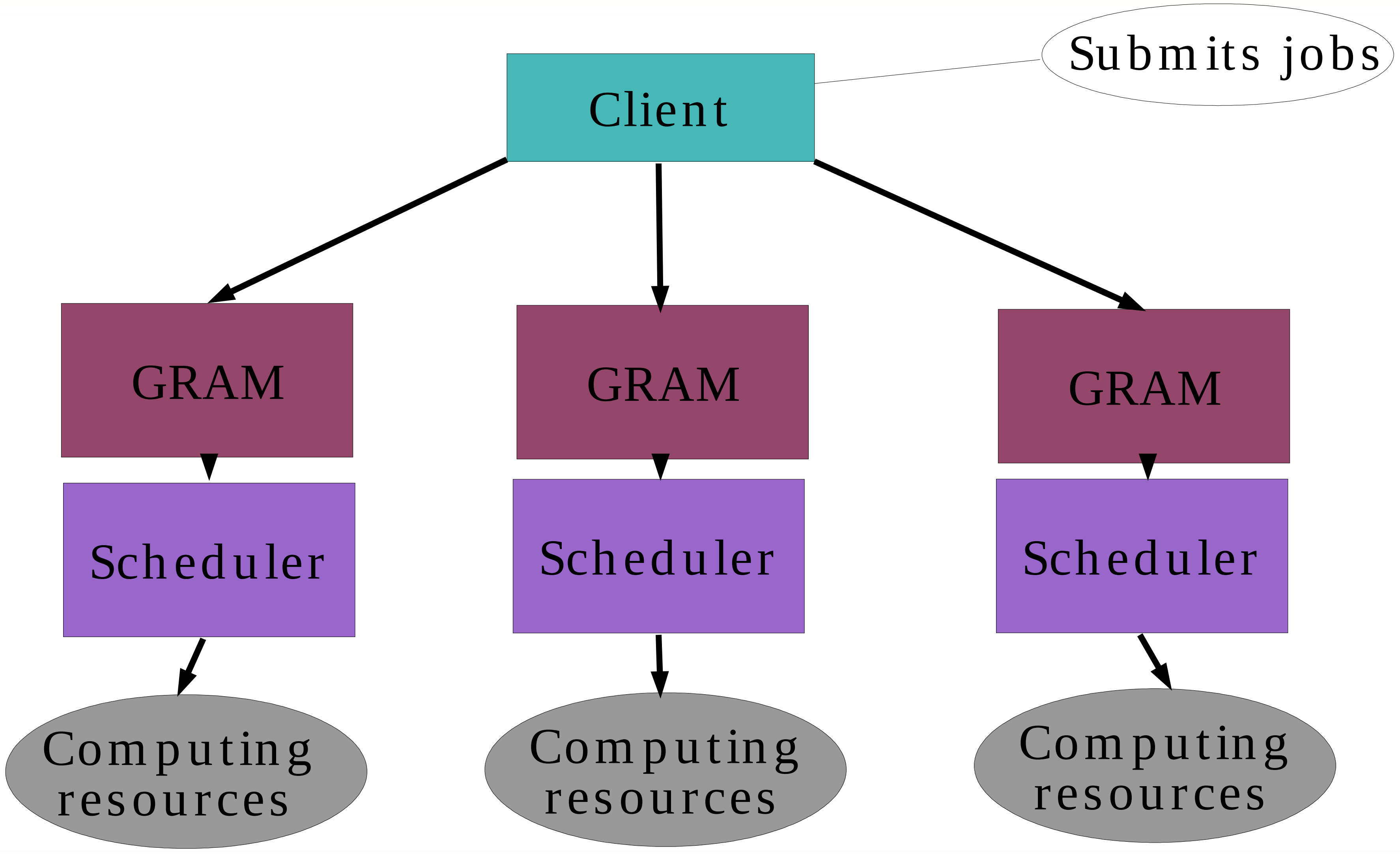
Please download the most recent version of the presentation from:

http://gridwisetech.com/resources

Intro

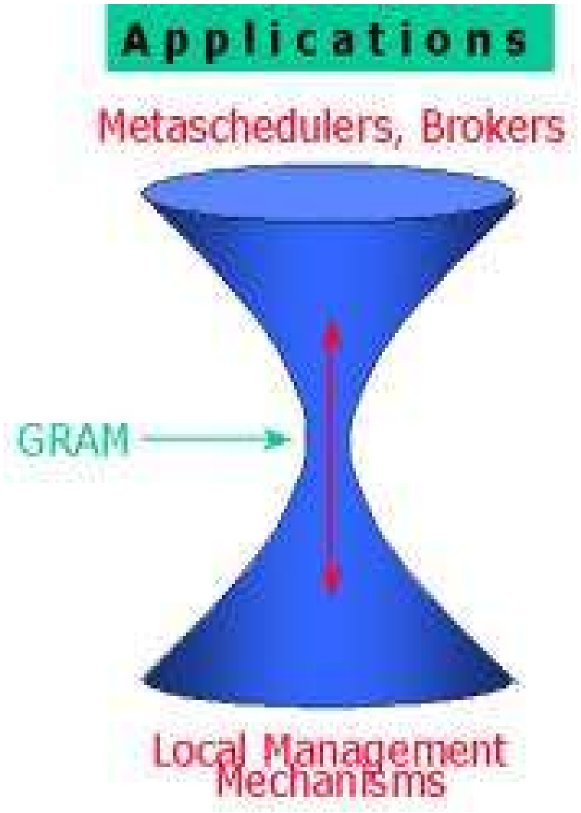
- In this presentation we are going to take a look at integration procedure of local resource management solutions with the new, fourth release of Globus Toolkit.
- We are also going to set up background information needed to understand this set of technologies and concepts.
- One of the presentation's points is to show why this integration can be valuable.

Overview


Submits jobs
Client
GRAM
GRAM
GRAM
Scheduler
Scheduler
Scheduler
Computing resources
Computing resources
Computing resources

What is GRAM?

- A resource manager.
- Grid Resource Allocation and Management.
- One of Globus Toolkit's components.
- Provides one-point access to remote resources.
- Allows to submit, monitor, cancel and get results of jobs on computing resources.
- Designed as a single, uniform interface to job scheduling systems.



Source: www.globus.org

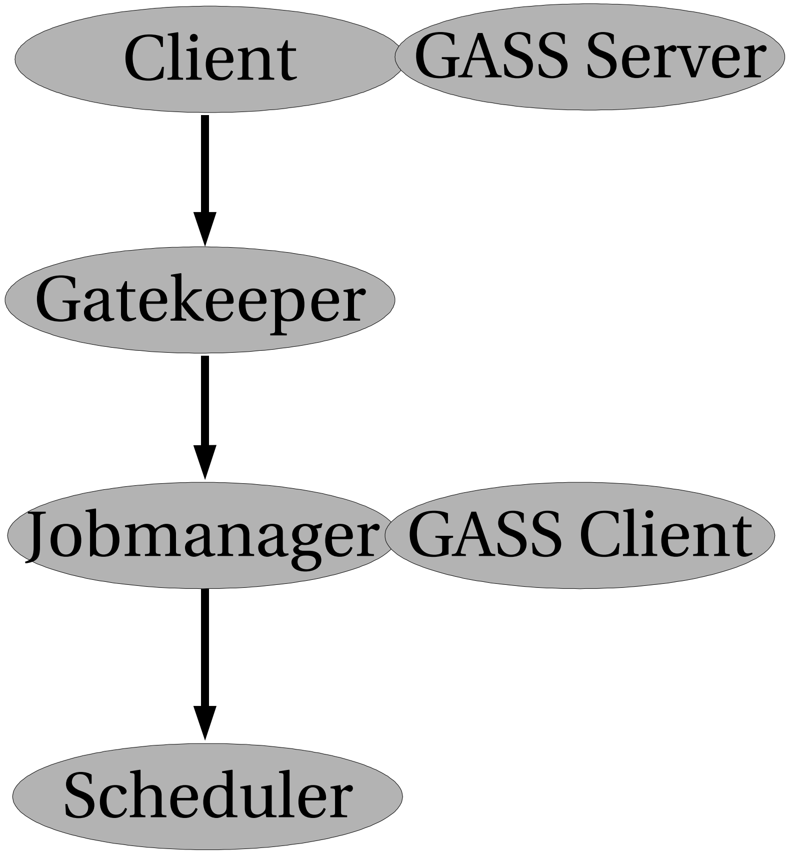
GRAM implementations in GT4

- Pre-WS GRAM
  - Based on proprietary Pre-WebServices protocol.
  - First introduced in Globus Toolkit 2.
- WS GRAM
  - Based on Web Services Resource Framework (WSRF).
  - New implementation in GT4.

Pre-WS GRAM

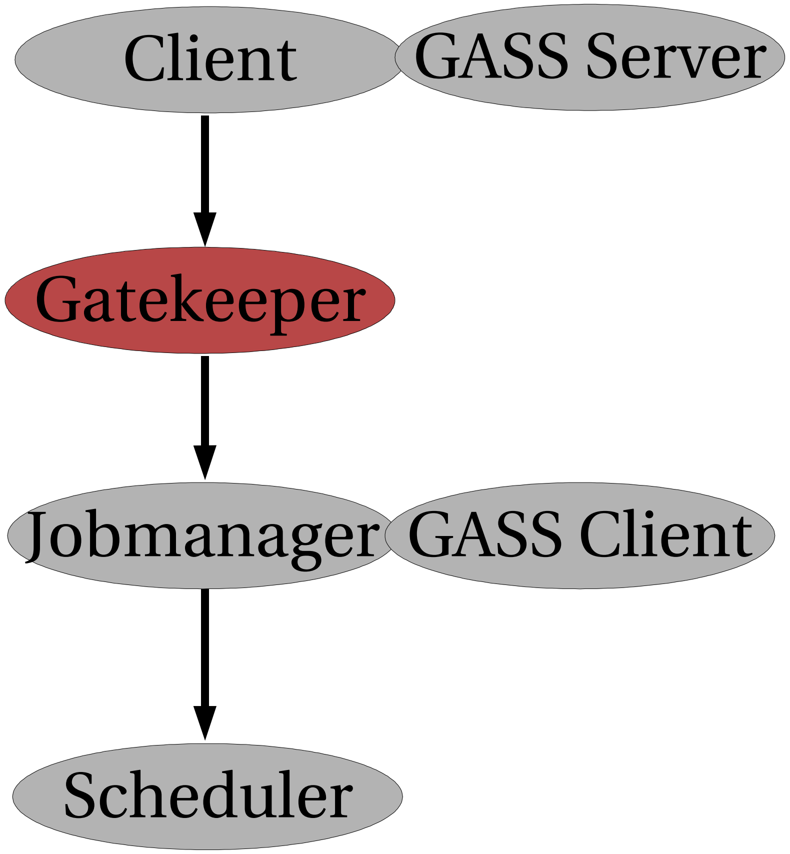
Pre-WS GRAM

- Main components:
    - Gatekeeper
    - Jobmanager
    - GASS



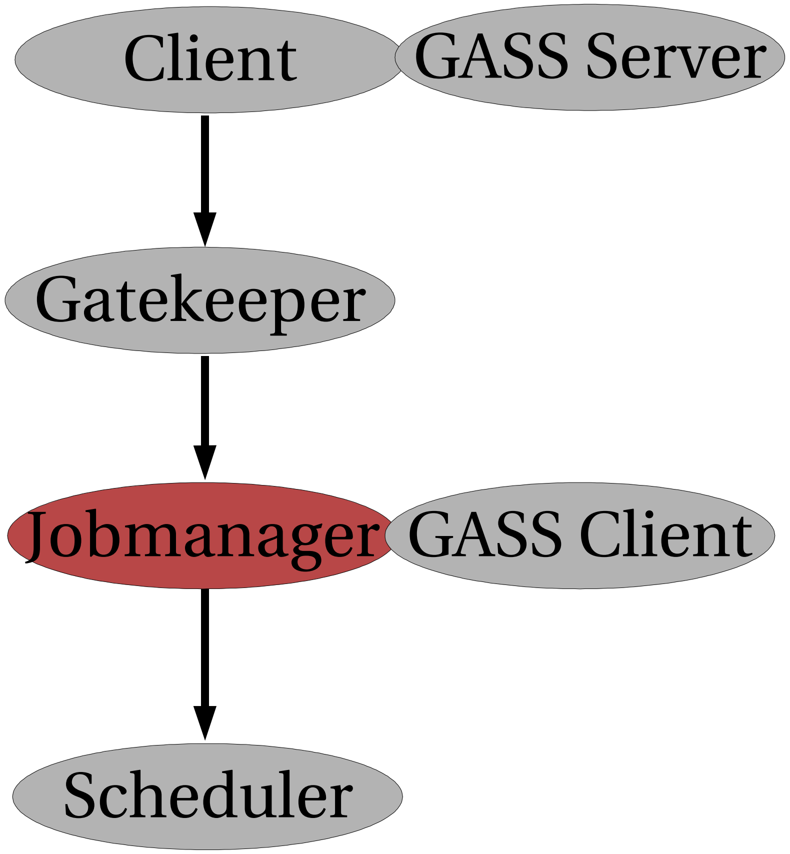
Pre-WS GRAM - Gatekeeper

- Gatekeeper
  - Acts as a secure equivalent of inetd daemon.
  - Remotely submitted jobs maps trough it to local accounts‘ priviledges.
  - Starts job manager on a local host with user's priviledges.



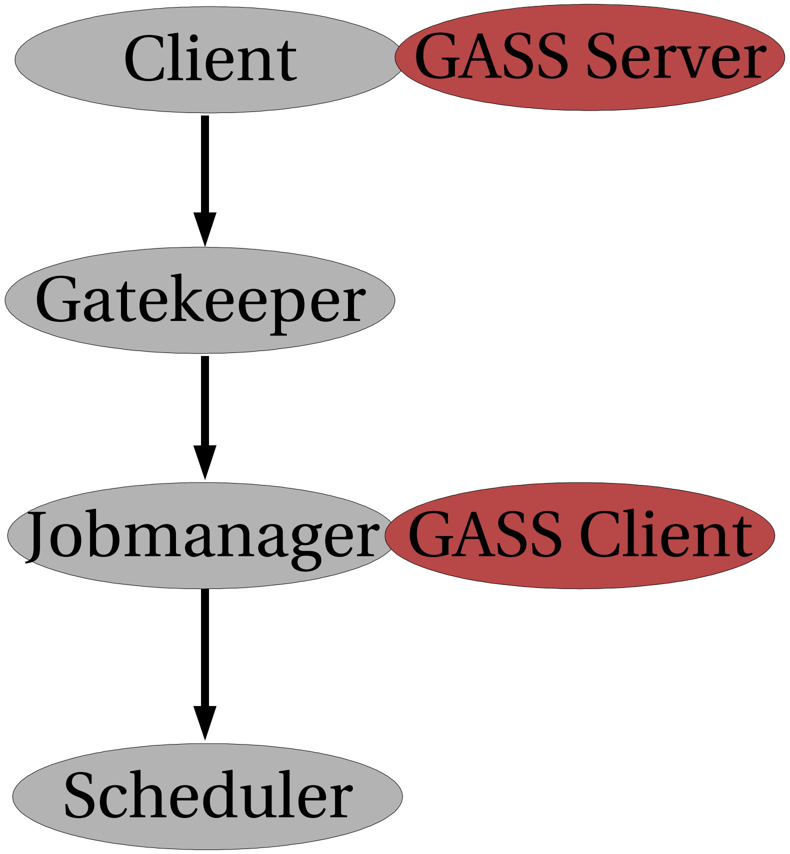
Pre-WS GRAM - Jobmanager

- Jobmanager:
  - Starts and monitors submited jobs on behalf of GRAM client.
  - Started by the Gatekeeper after successful authentication.
  - Communicates directly with local job schedulers to start requested jobs.



Pre-WS GRAM - GASS

- GASS
  - Global Access to Secondary Storage.
  - Integrated into GRAM.
  - Simple multi-protocol file transfer tools.
  - Used for:
    - ⇒ moving executables between storage servers and the execution hosts.
    - ⇒ moving input data to the execution hosts.
    - ⇒ retrieving results to the submission host.
  - GridFTP is used instead for more heavyweight data transfers.



WS GRAM

WS GRAM overview

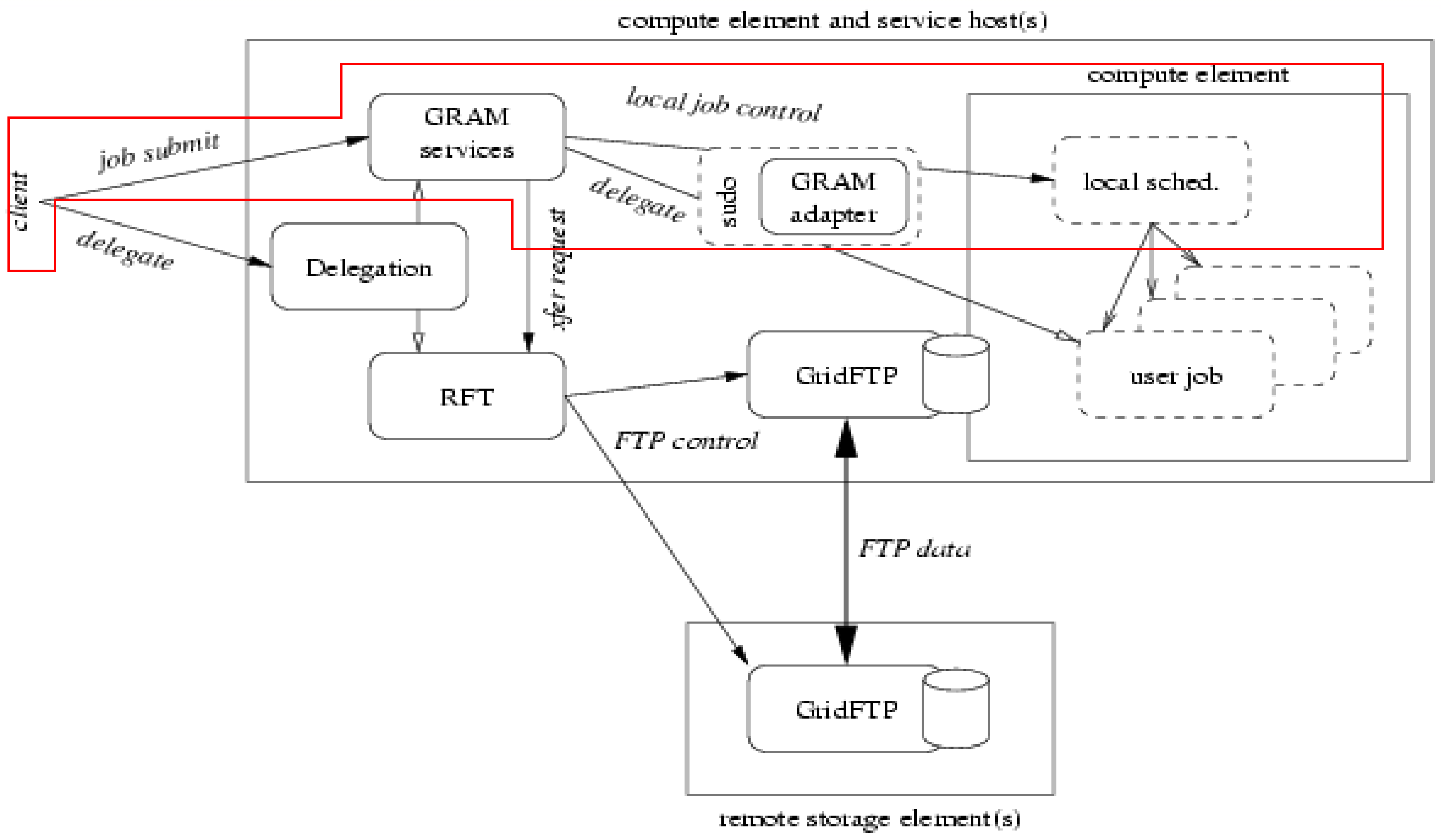
- WS GRAM is used for the same purposes as Pre-WS GRAM.
- The underlying core environment has changed (WSRF).
- WS GRAM provides a set of services that allows to access computing resources via Web services conforming to WSRF model.

What does WS GRAM change?

- WSRF compliant.
- Heading towards better stability, scalability and performance.
- GASS removed – only GridFTP and Reliable File Transfer (RFT) - *less overhead when not needed.*
- Sudo-*used when credentials of the submitter and the service account differ .*

  ⇒ *Replaces root-privileged Gatekeeper.*
  ⇒ *Avoids running Globus services container as root.*

How the pieces fit



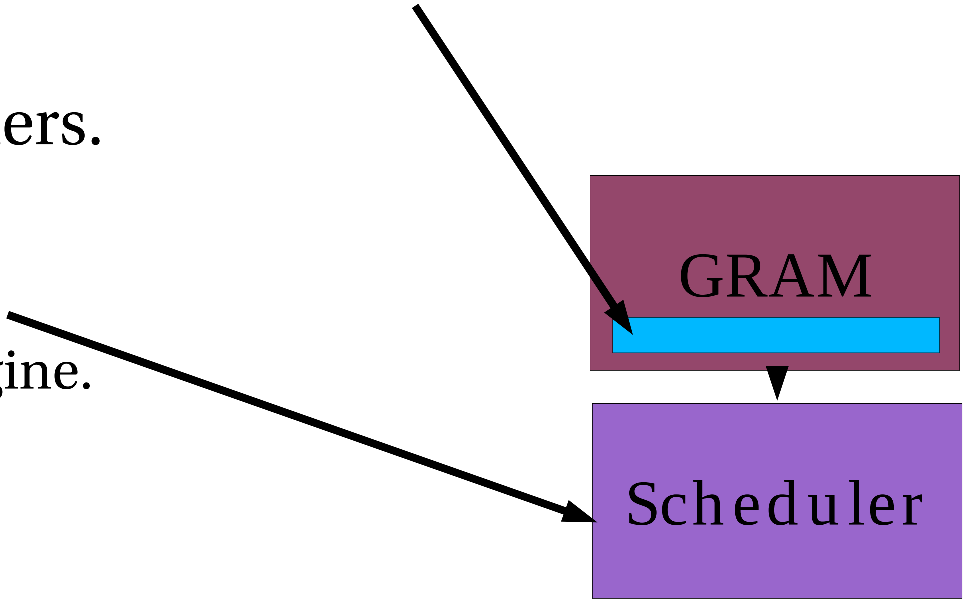
Source: www.globus.org

GRAM – Schedulers interface implemetation

gridwise technologies

What do we need to cope with?

- Job Manager Scheduler Interface.
  - Set of Perl modules that implement scheduler-specific interfaces.
- Local job schedulers.
  - PBS/TORQUE.
  - SUN N1 Grid Engine.
  - etc.



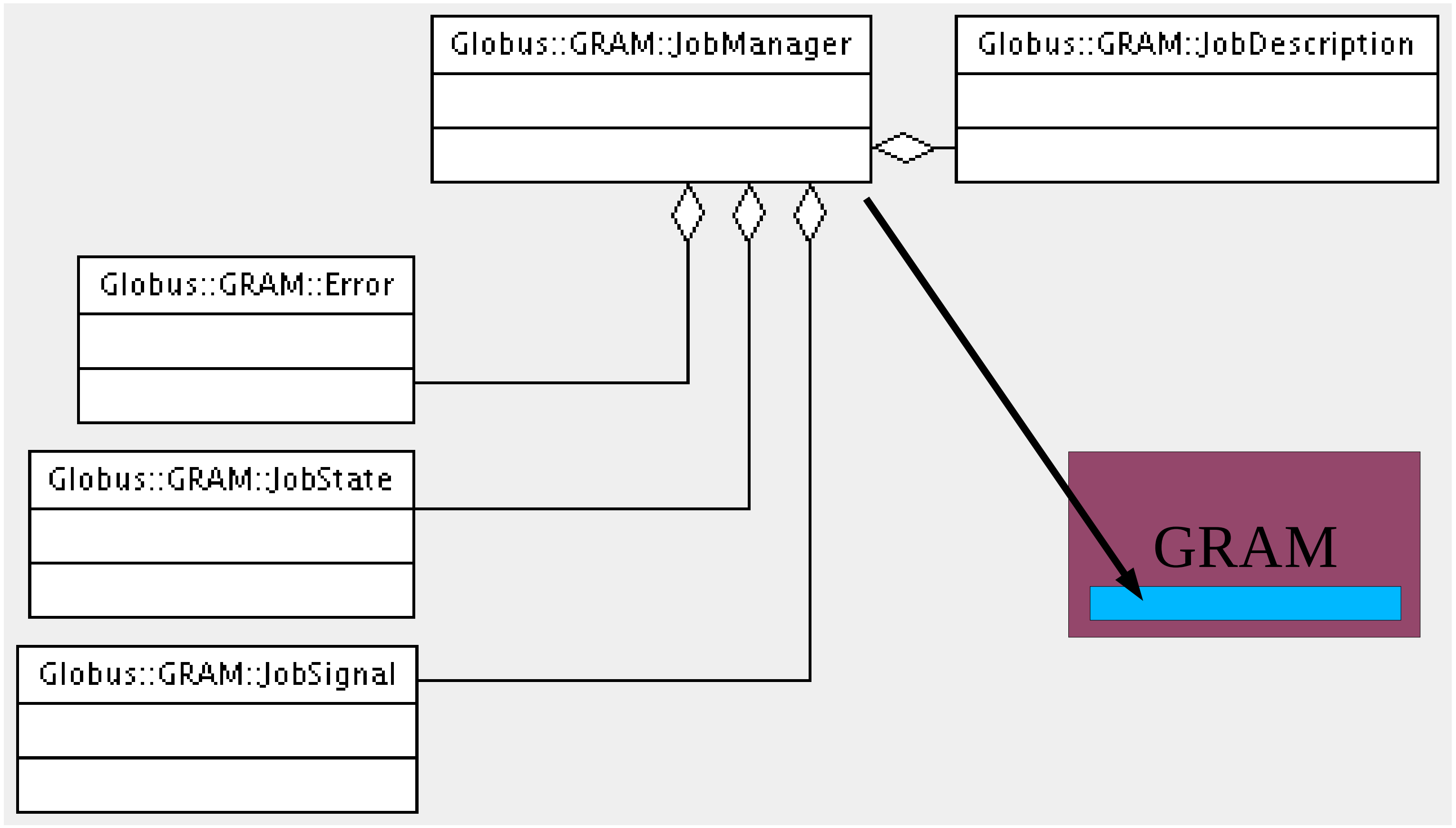
Good news

- Job Manager Scheduler Interfaces are compatible with both existing versions of GRAM.
  - Pre-WS GRAM
    - ⇒ Uses the whole implementation of the interface.
  - WS GRAM
    - ⇒ Uses a subset of the Pre-WS GRAM methods.

Job Manager Scheduler Interface

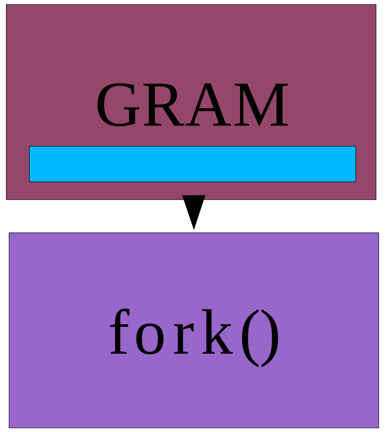
- There are a few files containing Perl modules needed to setup Job Manager Scheduler Interface (files are always named as the modules with .pm extension).
  - Globus::GRAM::JobManager – *Base class for all JobManager scripts.*
  - Globus::GRAM::Error – *GRAM Protocol Error Constants.*
  - Globus::GRAM::JobState – *GRAM Protocol JobState Constants.*
  - Globus::GRAM::JobSignal – *GRAM Protocol JobSignal Constants.*
  - Globus::GRAM::JobDescription – *GRAM Job Description.*

Schedulers' interface class diagram



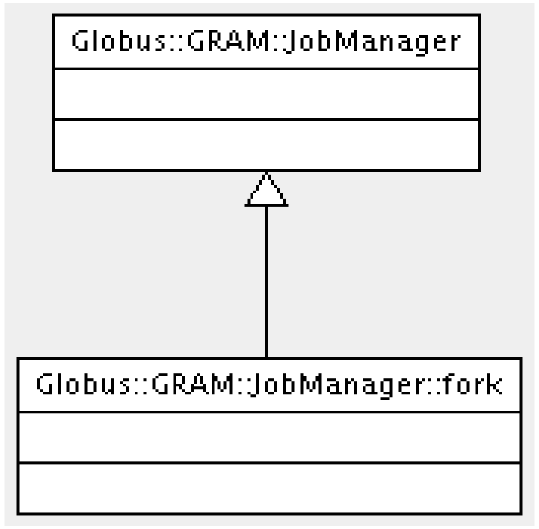
Simplifying the model

- Globus Toolkit contains basic interface to job scheduler – Fork.
- Not really a job manager scheduler interface, just the ability to spawn new jobs using fork() function.
- Fork is the only preinstalled interface and is a default one.
- Helps to test the environment and will help us to understand integrating GRAM with schedulers.



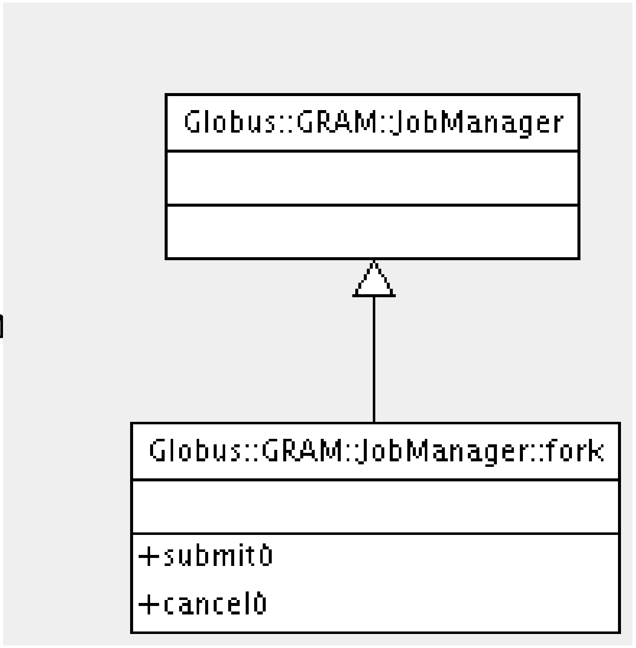
Specific job scheduler interface

- In order to use/write an interface for the specific job scheduler we need to care only for one module.
  - Globus::GRAM::JobManager::name_of_the_scheduler
- There is one such subclass of JobManager for each job scheduler. In case of Fork there is:
  - Globus::GRAM::JobManager::fork
  - Contained in fork.pm file.



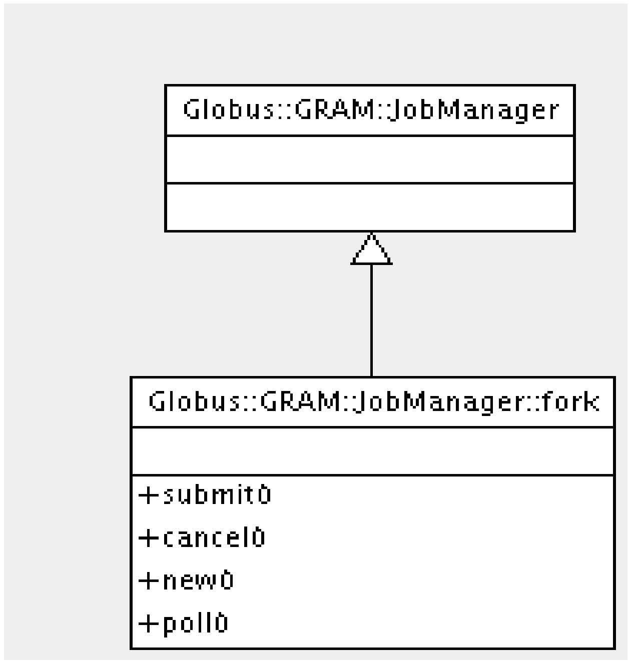
Module internals overview

- Each Job Schedulers' interface is implemented as a subclass of the Globus::GRAM::JobManager module.
- The most important methods that must be implemented are:
  - "submit"
    - ⇒ This method is called when job manager submits the job to the scheduler.
    - ⇒ "submit" method recieves the information of the original job request through the JobDescription data member.
  - "cancel"
    - ⇒ This method allows to cancel a scheduled job while it's running or waiting in a queue.



fork.pm

- Our sample Job Scheduler's interface additionally consists of:
  - A constructor.
    - ⇒ “new” method acts as a constructor
    - ⇒ If there is nothing specific to setup the default Globus::GRAM::JobManager::new will do the job.
    - ⇒ Otherwise we can overload “new” method as fork does.
  - “poll” method.
    - ⇒ “poll” method is used only by Pre-WS GRAM implementation.
    - ⇒ The purpose of this method is to check for updates of the job's status.



Scheduler Event Generator

- New, WS GRAM, job manager uses Scheduler Event Generator module for receiving events from schedulers.
- It is used instead of constantly polling schedulers with “poll” method (less overhead – performance improved).
- SEG module is implemented as C shared library.
- At the time of this presentation SEG module parses schedulers' logs to generate new events about job state changes.

Integration procedure

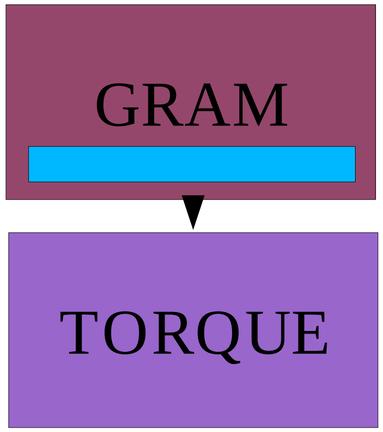
gridwise technologies

Job Schedulers' Adapters

- Job Schedulers' interface implementations come in prepared packages in the form of tarballs.
- Packages are prepared using Grid Packaging Toolkit (GPT) used in Globus Toolkit.
- Implementations available in GT4 includes:
  - Portable Batch System interface *(gt4-gram-pbs-3.9-src_bundle.tar.gz)*
  - Platform LSF *(gt4-gram-lsf-3.9-src_bundle.tar.gz)*
  - Condor *(gt4-gram-condor-3.9-src_bundle.tar.gz)*

TORQUE example

- TORQUE *(Tera-scale Open-source Resource and QUEue manager) is a resource manager.*
- It is based on the Portable Batch System (PBS) implementations such as OpenPBS.
- It is also a job scheduler (basic but with possibility to alter it by other, specialized schedulers).
- The fact that it is based on *PBS products makes it a good choice for integrating with Globus Toolkit's PBS implementation of Job Scheduler's Adapter.



Installation process

- Installation process using GT4 PBS adapter is straightforward.
  - First, we need to go to the *schedulers* directory in GT4 source distribution.
  - Next, the following commands need to be issued:
    - ⇒ *gpt-build gt4-gram-pbs-3.9-src_bundle.tar.gz gcc32dbg*
    - ⇒ *gpt-postinstall*
  - This will install the adapter using GPT and register the new functionality in Globus Toolkit installation.

Configuration process

- The only thing left is to associate local resource managers with GridFTP servers. This is done by mapping file systems paths to enable staging of files
- It is done by editing *$GLOBUS_LOCATION/etc/gram-service/globus_gram_fs_map_config.xml*
- The complete example of the file for PBS is located in WS_GRAM_Public_Interfaces.html in the Globus Toolkit documentation.

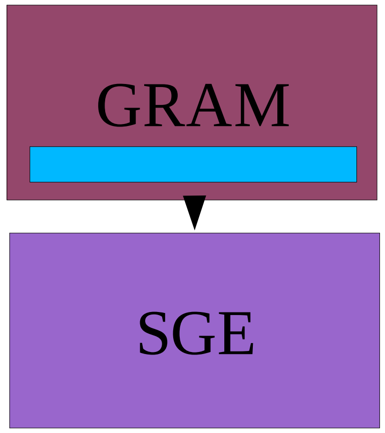
Configuration process

- The PBS adapter will be installed as *jobmanager-pbs*.
- The default jobmanager is fork and is called just *jobmanager*.
- If PBS is to be the default one, we need to change it by issuing:

  -
    ```
    setup-globus-job-manager-pbs  --service-name jobmanager
    ```

SUN N1 SGE example

- SUN N1 Grid Engine is a complete solution for resource management and scheduling.
- It is a commercial Sun Microsystems product that started as a community project.
- Globus Toolkit does not contain implementation of Job Scheduler's Adapter for SGE.
- There exists some unofficial implementations of the adapter for older versions of the Globus Toolkit.
- We are going to look into details of this existing solutions and/or come up with our own implementation.



Conclusions

- There are variety of resource managers used to create homogeneous environments.
- All those environments can be used for grid-wide computations, but we always need to know how to talk to them.
- By the use of GRAM we can create heterogeneous grid environment with one, uniform interface to various resources located and governed at different sites.

Comments, questions

dominik@gridwisetech.com

pawel@gridwisetech.com

Sources

- http://www.globus.org/toolkit/
- http://www.clusterresources.com/products/torque/
- http://wwws.sun.com/software/gridware/

Thank you.