Overview

- GridWay, on top of Globus Toolkit services, enables large-scale, reliable and efficient sharing of computing resources (clusters, computing farms, servers, supercomputers...), managed by different LRM (Local Resource Management) systems, such as PBS, SGE, LSF or Condor, within a single organization (such as an enterprise grid) or scattered across several administrative domains (partner or supply-chain grid).
- GridWay is an open-source component for meta-scheduling in the Grid Ecosystem, released under Apache license version 2.0, that gives end users, application developers and managers of Globus infrastructures a scheduling functionality similar to that found on LRM systems.
- The GridWay Metascheduler is a Globus project, so it adheres to Globus philosophy and guidelines for collaborative development.

Highlights

- Flexible and extensible architecture
- High efficiency and reliability
- State-of-the-art scheduling functionality
- Information drivers to interface MDS2 and MDS4
- Execution drivers to interface pre-WS GRAM and WS GRAM
- Transfer drivers to interface GridFTP and RFT
- Support for OGF standards: DRMAA and JSDL
- LRM-like commands to use and manage the Grid
- Interoperability between different grid infrastructures and middlewares.

# GridWay Metascheduler 5.2
## Metascheduling Technologies for the Grid

## Features and Benefits

<table>
<thead>
<tr>
<th>Feature</th>
<th>Function</th>
<th>Benefits</th>
</tr>
</thead>
</table>
| **Advanced scheduling capabilities on a grid consisting of distinct computing platforms** | Dynamic scheduling, opportunistic migration support, performance slowdown detection, self-adaptive application support and checkpointing support on heterogeneous and dynamic grids managed by Globus Toolkit services | • Decoupling between applications and the underlying local management systems (PBS, SGE...)  
• Non-intrusive execution  
• Integration of non-interoperable independent computational platforms (vertical silos)  
• Increased application throughput  
• Uniform environment and flexible infrastructure  
• Greater utilization of underlying resources |
| **Support for array jobs** | Array job capability provides parameterized and repeated execution of the same task. | • Efficient execution of high throughput computing and parameter sweep applications |
| **Support for job dependencies** | Job dependency capability allows the execution of a submitted job depending on the completion of other jobs submitted in the grid | • Efficient execution of abstract workflows involving branching and looping |
| **Scheduling policy module** | State-of-the-art scheduling policies, comprising job and resource prioritization policies. Support for the definition of new scheduling policies | • Allocation of grid resources according to management specified policies |
| **Scheduling reporting and accounting** | Support for the development of scheduling reporting and accounting facilities that provide detailed statistics of usage on the grid | • Analysis of resource utilization, determining trends in usage and monitoring user behavior  
• Performance tuning  
• Troubleshooting configuration problems |
| **Fault detection & recovery capabilities** | The meta-scheduler is able to detect and recover from the remote failure situations, such as remote job cancellation, remote system crash or outage and, network disconnection; and to recover from local failure | • Reliable and unattended execution of jobs |
| **Application compatibility** | The meta-scheduler is not bounded to a specific class of application generated by a given programming environment and does not require application deployment on remote hosts | • Wide application range  
• Reusing of existing software |
| **LRM Command Line Interface** | The CLI interface allows users to submit, kill, migrate, monitor and synchronize jobs, including MPI jobs Support for OGF standard JSDL | • CLI similar to that found on Unix and DRM systems such as PBS or SGE  
• Standard definition of jobs |
| **Standard Applications API (DRMAA)** | The scheduler provides full support for OGF standard DRMAA (C and JAVA bindings) to develop distributed applications | • Integration of ISV’s applications to GridWay  
• Compatibility of applications with DRM systems that implements the standard, such as SGE, Torque... |
| **Support for multiple-users** | The installation and configuration of GridWay is performed by the system manager and the users access GridWay from a front-end or from submission hosts, which do not require GridWay and Globus installation | • Globus installation is not required in each end-user system  
• Reduction in Firewall requirements  
• The administrators have full control of meta-scheduling deployment |
| **Flexible and extensible architecture** | The scheduler provides a modular architecture to allow communication with different resource management, file management and information services | • The meta-scheduler can be extended or used as a building block for more complex architectures  
• Easy development of drivers to access new computing services |
| **Straightforward deployment** | The scheduler is installed on a client system and does not require the installation of new services in the remote resources, apart from Globus services Installation based on auto-tools | • Easy and fast installation |
| **Interoperability** | The meta-scheduler provides support for the development of drivers that interface to distinct middlewares | • Interoperability between different grid infrastructures and middlewares (Globus, EGEE, UNICORE...) |
| **Supported remote services** | • Information drivers to interface MDS2 (MDS schema), MDS2 (Gluu schema) and MDS4  
• Execution drivers to interface pre-WS GRAM and WS GRAM (even both simultaneously)  
• Transfer drivers to interface GridFTP and RFT | • Support the existing platforms and resource managers (fork, PBS, SGE, LSF, LoadLeveler, Condor...). |