CSGrid Overview

CSGrid is a system for management and integration of applications in a distributed and heterogeneous computing environment. It provides a collaborative and extensible environment to abstract the use of the computational resources, and providing to end-users and domain-specific applications functionalities for creating, sharing and accessing distributed resources, such as projects' data and executable programs. Figure 1 shows an overview of the CSGrid middleware.

Figure 1 - CSGrid middleware overview

CSGrid Server and Middleware Services

The CSGrid Server is a component that intermediates requests from users and client applications to data repositories and execution nodes. The CSGrid data repository is an area where the CSGrid server stores the user database, user projects and algorithms execution code. The user database keeps data about users and their permissions. The projects area has a directory for the projects of each user and keeps descriptor files that hold information about projects and the projects files. The algorithm repository has a hierarchical structure that uses directories to represent each algorithm. Each directory contains a configuration file, the binaries and subfolders containing the algorithm versions.

The CSGrid server provides a set of services for the interaction of users and client applications, which are instantiated and exposed by the CSGrid Server through RMI (Remote Method Invocation) interfaces, for Java applications that are built using CSGrid Development Kit and through CORBA, by the OpenBus middleware, that allows external applications and services to access CSGrid. The main functionalities provided by CSGrid Middleware services are:

- **Algorithm Management**

  Provides facilities for users to manage, share and make algorithms available for submission and execution on the environment's computing resources. A user with adequate permissions can create and alter algorithms versions, registering the required information for their execution. This information includes the documentation, binary code, the platform (operating system version) where the binary code shall be executed, and a configuration file describing the type of input and output parameters of the algorithm.

- **Project Data Management**

  Provides facilities for users to create and share projects, which have hierarchical directory structures on which algorithms' execution input and output data are stored. The CSGrid server provides a storage area for user projects and mechanisms for permission and access control to project data.

- **Computing Resources Monitoring**

  Collects and provides information of available resources. These resources are the execution nodes that can be used to submit the execution of algorithms, and data areas used for the storage. The server defines a set of attributes to the computing resources that are presented to the users. These resources can be nodes or clusters of nodes.

- **Management and Monitoring of Executions**

  Allows the management and monitoring of algorithms executions. The user can submit algorithm executions on specific resources or leave the scheduling of these executions to be performed automatically by the system. Currently, CSGrid uses the CPU usage information from execution nodes to schedule the execution. The user can monitor the progress and interrupt the processing of running algorithms. When the execution of an algorithm terminates, CSGrid notifies the user and updates the user's project tree with the files created by the execution.

- **Intermediation of Resources Access to Projects Data**

  Provides mechanisms that allow execution nodes to access projects' and algorithms' data. The server commands the transfer of input and output data to the data areas configured as sandboxes of the computing resources.

- **Logs and Notifications**

  Offers facilities for monitoring the system status, including the log registering and notifications of events indicating the usage of the available functionalities.

- **Security and Authentication**

  Provides facilities for user management and authentication. Validates user authorization for the execution of the provided functionalities and access to resources.

SGA

The SGA is a component that exposes interfaces to allow remote execution of algorithms on execution host machines, and the monitoring of their status by the middleware (e.g. current or historical processing and storage usage). It also allows querying the machines' configurations, their

execution platforms, environment attributes and their current status. The SGA interface is flexible and can be instantiated in different scenarios. It can be deployed as a daemon in host machines to provide an execution interface to these hosts individually, or it can be set as a gateway to 3rd p arty Resource Management Systems. In the former case, each host machine becomes a server for the execution of algorithms. In the latter case, a module inside the SGA implementing the integration with a specific system is used, and the SGA intermediates execution requests from CSGrid to the other systems and manages these executions. Currently, there are SGA gateways implementations for the Sun Grid Engine, TORQUE/PBS, OurGrid, SLURM and Amazon EC2.

The main functionalities exposed by the SGA interfaces are:

- **Resource Configuration Information**

Provides mechanisms for querying the configuration of available computational resources for the execution of algorithms, identifying the attributes and requirements of the platform. When the SGA initiates, it registers itself on the CSGrid server as a new available resource.

- **Specific Platforms Information**

Collects and provides to the CSGrid server information about the computational environment, which is dependent of specific platforms. This information is consolidated and provided to the CSGrid server, together with a standard set of attributes.

- **Commands to Specific Platforms**

Provides mechanisms for the execution, monitoring and control of commands that are dependent of specific platforms.

- **Sandbox to Algorithms**

Provides mechanisms for the configuration of an optional sandbox area used for the input and output data of algorithms executions.

OpenBus

OpenBus is a CORBA-based middleware used to integrate multi-platform and multi-language systems based on a service-oriented architecture. It offers a service directory, access control mechanisms and peer-to-peer communication. External applications use the OpenBus development kit (OpenBus SDK) to authenticate, publish, discover and access services components, based on the usage policies configured by the OpenBus administrator.

The CSGrid server publishes the following CORBA services on OpenBus middleware:

- **OpenDreams (OpenBus Distributed Resource and Algorithms Management Service)**: offers a set of operations for the submission, monitoring and control of jobs on SGA's remote computational resources. It also provides access to the CSGrid user's algorithms configuration, abstracting their parameters and requirements. The OpenDreams interface is based on OGF's DRMAA 1.0 specification.
- **Hierarchical Data Service**: offers a set of operations for accessing and manipulating files in the CSGrid user's project area. User files are organized by projects, and users can share their files in a specific project with other users. Each project represents a data area, organized as hierarchical structures that store related files. All data files needed as input to programs submission should be previously available in a selected user project. Submitted programs also write output files in the associated project area.
- **SGA Monitor Service:** offers a set of operations for monitoring the SGA's execution nodes.