Comparison of European Grid Projects

Version 1.0, 16 May 2003

Project:

GEMSS

Area:

Resource Management and Scheduling

Table of Contents

1. Introduction

1.1. Objective and Structure

This document is one of thirteen templates that have common goal to gather information related to main European Grid Projects in order to make their accurate comparison in the framework of GRIDSTART initiative. We believe that the participation of particular projects members in preparation of this document will allow comparing all activities in a credible and exhaustive way.

The proposed structure of the description consists of two parts. The former is concerned with the general overview and architecture together with the contents of layers (the first template). The latter includes the main components of the Grid infrastructure (remaining 12 templates). Since information regarding the project architecture is to be quite general, more detailed description should be provided in the review of the main aspects of the Grid infrastructure. In order to prepare uniform description for each project, we identify the important issues that have to, should or can be included into particular components. Common issues for all components and these specific for this component are briefly described in the next section.

We ask you to proceed according to this schema. However, a feedback is obviously welcome. For some projects the document has been partially completed on the basis of descriptions found at the official web pages. In this case, we ask you to revise already filled in sections, correct and complete them if necessary.

You should take into consideration future plans while you fill in particular sections. Actually they are even more important then the current state of the project components. If you are not going to design some elements in the scope of the project at all, please, note it in the proper section.

1.2. Uniform description

All the descriptions of the Grid infrastructure components are divided into three parts: **General** section includes main requirements and functionality, **Details** section relates to the issues specific for particular component and **External** defines its connections with other components and users.

As it was mentioned above, some of the issues are common for all components or at least repeat for many of them. Such issues, appearing for many or even all areas are shortly characterized below.

In **General** section:

Main requirements determine the objectives and requirements of the workpackage or the software module responsible for the design of functionality related to the particular domain of the Grid infrastructure.

Functionality contains a set of operations provided by the project in the given area.

In **External** section:

Interfaces define services, SDKs, APIs and so forth which can be used in order to access the functionality of the component.

Low level Grid middleware is the middleware providing basic Grid functionality as for example Globus or UNICORE.

Relations with other components determine components that utilize or are utilized by component being described as well as data and information flow between them.

Issues that are specific for this particular domain of the Grid infrastructure are presented in the sequel. Some of them, which we consider to be clear, have been skipped, however, if they turn out to be vague, please, contact the authors of this document (ariel@man.poznan.pl).

The **Details** section deals with all issues related to the resource management. The whole process generally is divided into three parts: information, selection and running phase.

User information should cover all data exchange with a user during the process of resource selection and allocation. This section includes four subsections. **Level of automation** specifies whether a user gets the list of resources or just a single one, whether he makes the final choice etc. **Preferences gathering** issue describes a method for getting user's preferences concerning the time of execution, costs of resource usage and other criteria. **Quality of service negotiation and service level agreement**, in turn, provides methods for description and negotiation of user's requirements specification regarding dynamic resource usage, contract violation, etc.

Architecture description in **Application information** paragraph presents the mechanisms that allow a user to describe the architecture of application, for example, as a workflow.

Multischeduling provides information about applied global scheduling algorithms and coordination of multiple schedulers of various types (e.g. interactions between Grid scheduler and Local Resource Management System should be considered).

Adaptability concerns an adaptability in the context of the resource management. It includes issues such as rescheduling, checkpointing and job migration.

Economy-based brokering and scheduling takes into consideration costs of resource usage. Thus, information whether the cost is one of the criteria involved in the resource selection process (in **Scheduling and resource matching taking costs into consideration** paragraph) and whether there are the techniques facing the problem of cost exceeding during running a job (in **Adaptability in the context of costs**) is needed.

2. Resource Management and Scheduling

2.1. General

The GEMSS design is still being discussed by the project and not all issues have been finalized yet.

- **Main requirements**

The GEMSS environment provides medical simulation services on PC clusters that may be controlled by different queuing systems like Condor, PBS, or Cosy. In order to . address quality of service (QoS) aspects, we instrument the NEC scheduler Cosy for the corresponding kinds of interaction.
However it should be clearly stated that users do not explicitly deal with the scheduling system, they only select services advertised by specific providers under certain conditions (Qos contracts). Internal GEMSS modules are in charge of interacting with the queuing system.

- **Functionality**

(a) Submitting, releasing, and querying compute requests.
(b) Interaction with QoS module

2.2.Details

- **Supported application types**

  Independent tasks

INFORMATION PHASE

- **User information**

  Level of automation
  Internal GEMSS modules get a list of providers for the selected service. This could be interpreted as a list of resources, since each service provider employs a well known type of architecture. The architecture type itself could also define a selection criterion for a service.
  Preferences gathering

  Users can specify their preferences via a GUI. These settings will be interpreted by the QoS module.
  Quality of service negotiation and service level agreement
  In essence, it is possible to negotiate a price for the execution of a service on the particular machine. This is done by an iterative process which involves the QoS module and the scheduling system as determining components. Due to the support of advanced reservation schemes along with the usage of priority levels, the implementation of a non-trivial cost function is feasible. Once the user (QoS module) confirms the reservation, a contract is established. This contract is supposed to contain, among others, information about the costs of the service execution on the one hand and about penalty fees the provider has to pay in the case of contract violation.

- **Application information**

  Methods of architecture description

  Techniques for description of resource requirements

  Performance model

- **Resource information**

  Methods of resource description

Cosy provides functionality to retrieve dynamic information about the status of the compute system.

Preliminary filtering

SELECTION PHASE

- **Main objective: system throughput vs. application performance**
  The system focuses more on availability (-> advance reservation).
- **Mapping techniques**

  Matching tasks to resources and scheduling strategies

  Strategies
  Advanced reservation and FIFO oriented
  Optimization techniques

  Taking data location and transfer into consideration

  Multischeduling

- **Metrics for evaluation of mapping**

  Metrics involved

  Methods for aggregation of multiple metrics

RUNNING PHASE

- **Resource reservation**

  Cosy supports advanced reservation techniques. Thus users can request the reservation of resources at a fixed date and time. If resources are available at that time, the user gets a "ticket" for this slot, which has to be acknowledged within a certain time frame.

- **Submitting jobs to resources**

  The submission of jobs will be handled via the regular command line interface. The caller is the corresponding service of the requested application.

- **Adaptability**

  Rescheduling
  Jobs get rescheduled if hardware problems cause the system to crash. During the re-start phase Cosy scans submitted jobs that were scheduled and didn't terminate properly. Those are started once again and the user acknowledged. However, due to possible data losses further actions have to taken at user side.
  Checkpointing

Opaque user level (code modifications required)

Transparent user level (no code modifications required)

System level

Migration

Other

- **Economy-based brokering and scheduling**
  Cosy delivers information of (artificial) costs for a request. This data can be taken by other modules to decide, whether this provider should get the order for the requested service or not.
  Scheduling and resource matching taking costs into consideration

  Adaptability in the context of costs

2.3.External

- **Interfaces**
  Interface to the Qos module
  .
  Interface to the corresponding services

- **Low level grid middleware**

- **Relations with other components**