



# Adaption of User and Program Interfaces and Testing<sup>1</sup>

Deliverable	WG-5, D5.4
Authors	Workgroup 5
Editors	Robert Tucker, Frank Breitling, Jürgen Steinacker, Rainer Spurzem
Date	18.02.08
Document Version	1.1.0
Current Version	1.1.0
Previous Versions	1.0.1, 0.x.y

### **A: Status of this Document**

Deliverable 4 of Workgroup 5.

### **B: Reference to project plan**

Deliverable 5.4: AstroGrid-D - Adaption of User and Program Interfaces and Testing

---

<sup>1</sup>This work is part of the AstroGrid-D project and D-Grid. The project is funded by the German Federal Ministry of Education and Resarch (BMBF).

## C: Abstract

This document summarizes the state of testing and implementation of User Interfaces and Programming Interfaces in relation to job submission within the AstroGrid-D community. This is done by means of a specific set of Use Cases spread across the participating sites of the AstroGrid-D. The testing is in reference to deliverables 5.2 (Scheduler/Broker) and 5.3 (Robotic Telescopes). A summary of testing of Use Cases run independently of 5.2 is also provided.

## D: Changes History

Version	Date	Name	Brief summary
1.1.0	15.4.08	Rainer Spurzem	Published in Astrogrid-D
1.1.0	18.2.08	Rainer Spurzem	Small corr. 3.5, 5.4; Update Table in 4.1
1.0.1	11.2.08	Rainer Spurzem	Finalising content
1.0.0	22.12.07	Rainer Spurzem	Adding input by TRo, HMA, finalising content
0.3.5	20.12.07	Jürgen Steinacker	Adding input MPE, adding text
0.3.4	16.12.07	Rainer Spurzem	Adding general text 1
0.3.0	07.12.07	Jürgen Steinacker	Adding input MPA
0.2.0	03.08.07	Frank Breitling	Adding sections for robotic telescopes; adding references
0.1.0	25.07.07	Robert Tucker	Working draft creation

E:

## Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
<b>2</b>	<b>Descriptions of Use Cases Tested</b>	<b>7</b>
2.1	GEO600 . . . . .	7
2.2	ClusterFinder . . . . .	7
2.3	ProC . . . . .	7
2.4	NBODY6++ . . . . .	7
2.5	Dynamo . . . . .	8
2.6	Amiga . . . . .	8
2.7	Nirvana . . . . .	8
2.8	Robotic Telescopes . . . . .	8
<b>3</b>	<b>Job Submission Mechanisms</b>	<b>10</b>
3.1	GEO600 . . . . .	10
3.2	Atomic Use Cases - Dynamo . . . . .	10
3.2.1	A first mechanism . . . . .	10
3.2.2	A more general mechanism . . . . .	10
3.2.3	Summary . . . . .	11
3.3	ProC . . . . .	11
3.4	Cluster Finder . . . . .	11
3.5	NBODY6++ . . . . .	11
3.6	Robotic Telescopes . . . . .	12
<b>4</b>	<b>Resources</b>	<b>13</b>
4.1	Resources on which use cases have been tested . . . . .	13
4.2	Live List of Resources . . . . .	14
<b>5</b>	<b>Results of Use Case Testing</b>	<b>15</b>
5.1	GEO600 Testing . . . . .	15
5.1.1	Command Line Results . . . . .	15

---

5.1.2	Scheduler Results . . . . .	15
5.2	ProC Testing . . . . .	16
5.2.1	Command Line Results . . . . .	16
5.2.2	Scheduler Results . . . . .	16
5.3	ClusterFinder Testing . . . . .	17
5.3.1	Command Line Results . . . . .	17
5.3.2	Scheduler Results . . . . .	17
5.4	NBOD6++ Testing . . . . .	18
5.4.1	Command Line Results . . . . .	18
5.4.2	Scheduler Results . . . . .	18
5.5	Dynamo, Amiga, Nirvana Testing . . . . .	19
5.5.1	Command Line Results . . . . .	19
5.5.2	Scheduler Results . . . . .	19
5.6	Robotic Telescopes . . . . .	19
<b>6</b>	<b>Summary</b>	<b>20</b>
6.1	Collation of Use Case Testing . . . . .	20
<b>A</b>	<b>Definitions/Concepts</b>	<b>21</b>
A.1	Use Cases . . . . .	21
A.2	Atomic job . . . . .	21
A.3	Data Staging . . . . .	21
A.4	MPI Job . . . . .	21
	<b>References</b>	<b>22</b>

## 1 Introduction

The AstroGrid-D is committed to develop software and services such that the seamless access of science users to distributed resources through standardized interfaces in the grid is provided for their scientific research. The present set of resources and interfaces is summarized as follows:

- a compute resource such as a cluster for high performance computing
- a workstation for smaller tasks or job farming
- a data storage server
- a robotic telescope

A set of so called 'Use Cases' based on research topics and associated software were developed to help identify what components are required in the AstroGrid-D. These were selected to be the pioneering applications to be supported and used in the Astrogrid-D. In fact progress in the deployment and support of our 'Use Cases' in Astrogrid-D is very diverse. Some of them are very advanced and use all available facilities more or less right from the moment they are available. Others have only been tested less deeply and for some no information at all is available. As compared to the previous deliverables 5.1 and 5.2 we concentrate this deliverable 5.4 therefore on those 'Use Cases' which have made considerable progress and could be considered as 'close to production' for their users.

This document presents in particular:

- developments and testing done with respect to deliverables 5.2 and 5.3:
- Use Case description and requirements for those Use Cases tested.
- description of interfaces developed to facilitate Use Case testing.
- results of the Use Case testing on the command line.
- results of the Use Case testing via the scheduler/broker 'GridWay'.
- results of the Use Case testing via the GRAM Adaptor 'GridGateWay'.

This deliverable is also intended to identify as much as possible generic features in the running and deployment procedure, which are required to run and support a certain class of our present use cases, and thus also presumably a larger number of future applications similar in their structure. This is complementary to the approach in D5.1 and D5.2, where we tried to anticipate requirements, here the main focus is to summarize from the actual implementation (as far as it exists) the common properties a posteriori. In order to provide an environment, which a typical user would consider worth while to access in and via the grid certain elements need to be present:

- a generic user portal with sub-menu's providing more specific functionalities for use cases or classes of use cases.

- a facility which translates the request of the user, through the portal, into a job description, using a proper language – in Astrogrid-D we have decided to support in principle JSDL (see D5.2), however, our present implementations unfortunately still only support RSL.
- the task should be executed on grid resources selected by requirements stated by the user; special requirements need to be supported by special parameters known in the middleware. The way how to specify these requirements has not yet converged in Astrogrid-D - some specifics such as special hardware or parallel jobs can be specified by certain job submission parameters, while more complex software requirements should be specified e.g. with modules.
- the user should have information about the job status, receive output logs and data, and have control about the handling of larger job data.
- the user should be able to use job monitoring and data streaming capabilities as far as they are required.
- the user should be able to concatenate atomic jobs to workflows and describe them alternatively in a proper workflow language by oneself or create the workflow description using proper service interfaces.

This is an ambitious list of requirements, but it should be always clearly remembered as a goal when working on individual deliverables and software components. It is not sufficient to provide isolated functionalities which are not able to work hand in hand with other components to finally fulfil the aforementioned needs. We also note that these requirements transgress the borders of WG-5 (job submission and job scheduling), but since WG-5 is providing one of the core facilities of middleware we need to be aware of it. File management is provided by WG-3, resource description by WG-0, data stream management by WG-4, and portal development by WG-7, to mention only a few important links.

The possibility for users (and users of our use cases) to exploit the grid works presently in a number of levels of different complexity (and different degree of matching the more general criteria mentioned above):

- using the grid for gsissh and authorization, but otherwise submitting all jobs manually on certain execution hosts; file and data management manually.
- using globusrun-ws for job submission of a JSDL or RSL job; a target host needs to be specified manually, data are staged and returned through globus. Some more or less use-case specific shell script might be used to facilitate the procedure for the user.
- using scripts developed for specific Use Cases or a set of Use Cases
- supplying a template of fields to the GridWay Server which is installed locally
- submission via Globus of an RSL job to the GridGateWay on the GridWay Server host

The GridWay Server and GridGateWay mechanisms are both described in Deliverable 5.2. The function of the GridGateWay is to avoid having to install a GridWay Server at each site and to circumvent issues with directory-staging encountered with the GridWay template mechanism. This also avoids a requirement for scientists or scripts to generate/create another different set of template tags.

## 2 Descriptions of Use Cases Tested

The following gives a brief description of the Use Cases for which testing results are reported in section 5. More details on how they are being run is given in section 3 (Submission mechanisms).

### 2.1 GEO600

The GEO600 use case is named after the German-British gravitational wave detector located in Hannover. It contributes as one out of four world-wide detectors to the Laser Interferometric Gravitational Wave Observatory (LIGO). Their common goal is to detect and prove the existence of gravitational waves. The four detectors collect a large amount of data, which is later being pre-processed offline. Starting from simple workstations to supercomputers the application is actively supported on a wide range of hardware and software platforms, thus making it most suitable to be run on and truly benefit from a grid.

### 2.2 ClusterFinder

Clusterfinder processes data from galaxy catalogs and x-ray telescopes to produce a map of the likelihood of the presence of a galaxy cluster at every point in space. The job is computationally intensive but can be easily parallelized by treating each patch of sky separately, so that grid technology is a promising approach to reduce the turn-around time of a calculation. The input data needed by a host for a particular patch of sky is obtained from public data bases using grid technology. The program is coded in Fortran, which is in common use in the astronomy community but whose application on the grid faces issues which other languages do not.

### 2.3 ProC

ProC uses the Process Coordinator (ProC) scientific workflow engine to organize several workflows. One of the workflows produces a simulation of colliding galaxies. Another workflow uses a snapshot and produces a number of fly-around videos in real-time. The latter workflow has been demonstrated on occasion of the AstroGrid-D mid-term review which took place in Berlin in early March 2007.

### 2.4 NBODY6++

NBODY6++ is a member of a family of high accuracy direct N-body integrators used for simulations of dense star clusters, galactic nuclei, and problems of planet formation.

More information can be found on <http://nb6mpi.pbwiki.com> (see FAQ).

NBODY6++ simulations run as medium- to large-size parallel MPI jobs on supercomputers or clusters, using 32 ... 512 processors. Due to the communication scheme used within the application, NBODY6++ jobs require high network bandwidth and low latency (supercomputers with proprietary interconnect, or clusters with high-speed interconnect such as Myrinet or InfiniBand). NBODY6++ simulations are also very compute-intensive; if available the code can make use of special GRAPE boards as hardware accelerator (this will greatly speed up a run). At present there exists a simplified version of NBODY6++, which still bears its basic features. This variant is called phi-GRAPE (Harfst

et al. 2007, *New Astron.* 12, p. 357), it has a simpler algorithmic structure for benchmarking and testing, but is considered to belong to the same use case as NBODY6++.

An experimental version of the NBODY6++ code provides integrated real time application monitoring, steering, and visualisation functionality via IP sockets through SSH tunnels (cooperation with FZ Jülich, see also <http://www.fz-juelich.de/zam/xnbody/>).

## 2.5 Dynamo

The Dynamo Use Case comes from the field of Magneto-Hydro-Dynamics (MHD) Rotation and turbulence in stars, accretion disks, and galaxies cause a “Dynamo” effect. Computer simulations of Dynamos solve the induction equation with turbulent electromotive force (alpha tensor). The Dynamo code is written in Fortran. Currently a statically built executable is used for testing. An MPI version is under development. Dynamo requires an ASCII input file which gives the start field conditions for the simulation. The input file is staged with the executable to the remote host and needs to be in the same directory as the executable. For more details of running this Use Case see section 3 on submissions mechanism.

## 2.6 Amiga

AMIGA is a code for cosmological  $n$ -body simulations, It is purely grid-based and uses an arbitrary number of particles (each having its individual mass). The inter-particle forces due to gravitational interactions are computed by solving Poisson’s equation on a hierarchy of grids: while the whole computational domain is covered by a regular mesh, in high density regions the code places finer grids in a recursive manner that freely adapts to the actual particle distribution. These “refinement grids” can have arbitrary shape and entail an adaptive force softening during the course of the simulation. The Amiga executable requires an ASCII input file passed as standard input.

## 2.7 Nirvana

NIRVANA is a grid-based computer code for the solution of the equations of magnetohydrodynamics and for the Poisson equation. It uses adaptive mesh refinement techniques to describe systems with large spatial scale variations.

The Nirvana job is an MPI job. For the purposes of porting to the grid the use-case owner provided a static executable. Nirvana requires an ASCII input file called `nirvana.in` to be present in the same directory as the executable.

## 2.8 Robotic Telescopes

The grid integration of robotic telescopes aims at the development of a global network for new types of observations which cannot be accomplished with individual instruments. Important examples include multi-wavelength campaigns, rapid response to transient objects such as GRBs and supernovas as well as continuous long-term monitoring of objects independent of day and night cycles and weather. Advantages of the grid of telescopes include common interfaces for scheduling of observations, monitoring the network activity and access to data.

An observation request can be submitted to a robotic telescope where the Robotic Telescope Markup Language (RTML, [?]) serves as the job description, where it is scheduled by the telescope controller. Further details can be found in [?].

---

## 3 Job Submission Mechanisms

### 3.1 GEO600

The end user checks out the GEO600 use case from SVN (`svn://svn.gac-grid.org/software/GEO600`), and can then either choose to submit single jobs for debugging purposes using the provided command line tool or specify a list of resources and their usage policies in a configuration file.

The configuration file forms the basis for a simple grid scheduler that checks the status of running jobs and automatically submits new jobs to grid resources using GT4 web services. The deployment process is integrated in the job submission process and so frees the user from logging into resources to carry out a manual deployment.

GEO600 supports job submissions to simple workstations ( GT4/fork ) as well as cluster resources ( GT4/PBS/SGE). The majority of jobs is currently being submitted to cluster resources.

### 3.2 Atomic Use Cases - Dynamo

#### 3.2.1 A first mechanism

Initially a bash script was developed to have a preliminary auto-submission process for the Dynamo Use Case.

- a set of input files were provided by the use-case owner
- an execution host is taken from a predefined list
- several jobs could run in parallel although actual job submission was done in quick sequence.
- a static executable was staged to all of the Linux based platforms available in the AstroGrid-D.
- an input file is staged with the executable to the remote machine.

This was a very effective mechanism and was adapted to allow real-time fetching of output files for visualization using IDL during a demo.

For script details see `svn//:software/`

#### 3.2.2 A more general mechanism

A second development was to generalize the job-submission mechanism to cater to more Use Cases (Dynamo, Amiga, Nirvana) thus allowing atomic jobs to be automatically submitted to a predefined list of resources. This was achieved using a Perl script `jsdl_submit.pl` which creates a JSDL (Job Submission Description Language - XML based) document.

- a user specifies in the simple template the job input data, executable name and location and the output location for the results.

- multiple input sets are allowed and produce independent jobs running on different resources
- the template is parsed by the Perl script `jsdl_submit.pl`.
- this produces a JSDL document for each required job input set.
- the JSDL document is further translated to the RSL language which is submitted to Globus.
- jobs are submitted to a resource taken from a preset list.
- the user does not know which resource will be selected unless specified in advance
- the availability of the resource is verified before submission.
- results are copied to the required output location

For `jsdl_submit.pl` details and source see `svn//:software/auto_submit`.

### 3.2.3 Summary

Using `jsdl_submit.pl` the AIP use-cases Dynamo, Amiga and Nirvana could be submitted automatically to AstroGrid-D resources by means of filling in the simple template.

## 3.3 ProC

The video-production workflow uses several job-submission mechanisms. In the original version native Globus 2 (pre-WS) commands are being used for data staging and job submission. In the next, more modern version the Grid Application Toolkit (GAT) is used for job submission. Internally, again, Globus 2 protocols are being used for job submission and file staging.

## 3.4 Cluster Finder

Clusterfinder is not yet in production on the grid. A system has been developed to distribute the source code and related files using grid technology to grid nodes and to ensure the ability to compile and run on those nodes. The workflow is expressed in a makefile deployed on the nodes, and the calculation is controlled by a parameter file. Because the current mode is still development and not yet production, job submission is usually done by logging in to a host as a grid user with `gssh`, editing the parameter file as required, and calling the makefile with appropriate targets.

## 3.5 NBODY6++

Job submission follows exactly the general mechanism described above. In fact, to ease general user access a submission script has been written by TB, that automatizes the steps and allows a book-keeping of user jobs. Every user specifies the name of input files and will receive the output and can check the status under a certain unique job number. The capabilities of the GridWay

broker are fully used, see general features above. It is possible meanwhile to access parallel queues and GRAPE clusters (for phi-GRAPE).

The job submission script is invoked by

```
submit.sh
```

where the call without parameters provides help output; it is possible to specify input data files, different queues, parallel or serial jobs. The submission target could be specified as a particular globus host or gridway host. It is planned to allow as a parameter as well certain special hardware (GRAPE). Parameters not provided will be substituted by suitable defaults. The script compiles the entire source code once more at the target host and runs the executable using the specified input file. Every job gets a job id. More details can be found on

<http://www.ari.uni-heidelberg.de/mitarbeiter/bruesemeister/agdguide/x105.htm>

Presently a problem is the creation of job chains, because for a follow-up job typically large data files need to be accessed and their staging back and forth through the gridway job submission process is considered as inefficient and inelegant. We are currently working in the context of WG-3 on an algorithm to allow it to the user to specify files in the Astrogrid-D transparently. The job broker should in the future be able to take into account file locations when sending jobs to execution hosts.

### 3.6 Robotic Telescopes

Before an observation request can be submitted an RTML description has to be created. Therefore a template is provided. Moreover, XML editors provide efficient user interfaces for editing RTML templates. A good example is the XML editor included in Eclipse ([?]).

Submission is done through the GSI-OpenSSH command. For example, the syntax for submitting the observation request `G1586A.rtml` to the telescope server `photon.aip.de` is

```
cat G1586A.rtml | gsissh photon.aip.de tel -submit
```

The observation is canceled analogously using the observation unique identifier through

```
gsissh photon.aip.de tel -cancel de.aip.STELLA-I/G1586A
```

Details regarding the submission of observation requests can be found in [?].

## 4 Resources

### 4.1 Resources on which use cases have been tested

Resource Name	Location	Type	Scheduler	Use Cases
astar.aip.de	AIP	workstation	globusrun-ws	atomic <sup>1</sup>
astrodata01-10.aip.de	AIP	Storage Servers		large data sets
cashmere.aip.de	AIP	workstation	globusrun-ws	atomic
dublin.aip.de	AIP	workstation	globusrun-ws	atomic
photon.aip.de	AIP	workstation	globusrun-ws	atomic
gavo2.aip.de	AIP	workstation	globusrun-ws GAT	atomic
gavo3.aip.de	AIP	workstation	globusrun-ws	atomic
STELLA I	Tenerife	Telescope	gsissh	Observation
STELLA II	Tenerife	Telescope	gsissh	Observation
RoboTel	AIP	Telescope	gsissh	Observation
a01.hlr2.lrz-muenchen.de	LRZ	cluster	SGE	GEO600
astrodata01-10.gac-grid.org	AIP	workstation	Fork	GEO600, atomic
bellatrix.ari.uni-heidelberg.de	ARI	workstation	Fork	GEO600, atomic
beteigeuze.ari.uni-heidelberg.de	ARI	workstation	Fork	GEO600, atomic
bladekemper <sup>2</sup>	TUM	workstation	Fork	GEO600, atomic, GAT, ProC
buran.aei.mpg.de	AEI	workstation	Fork, GAT	GEO600, atomic, ProC
dgrid-globus.rz.rwth-aachen.de	RZA	cluster	PBE	GEO600
gramd1.d-grid.uni-hannover.de	UH	cluster	PBE	GEO600
hector.zih.tu-dresden.de	TUD	cluster	PBE	GEO600
hydra.ari.uni-heidelberg.de	ARI	cluster	PBE, GAT	GEO600, atomic, ProC
iwr4.fzk.de	FZK	cluster	PBE	GEO600
lx32ia1.lrz-muenchen.de	LRZ	cluster	SGE, GAT atomic, MPI	GEO600, ProC
lx64ia2.lrz-muenchen.de	LRZ	cluster	SGE, GAT atomic, MPI	GEO600, ProC
mardschana.zib.de	ZIB	cluster	PBE	GEO600, atomic
othello.zih.tu-dresden.de	TUD	cluster	PBE	GEO600
rigel.ari.uni-heidelberg.de	ARI	workstation	Fork	GEO600, atomic
saiph.ari.uni-heidelberg.de	ARI	workstation	Fork	GEO600, atomic
srvgrid01.offis.uni-oldenburg.de	UO	cluster	PBE	GEO600
supergrid.aei.mpg.de	AEI	workstation	Fork	GEO600
titan.ari.uni-heidelberg.de	ARI	workstation	Fork	GEO600, atomic, MPI
udo-gt01.grid.uni-dortmund.de	UD	cluster	PBE, GAT	GEO600, ProC
astrogrid.aei.mpg.de	AEI	workstation	GAT	ProC
gavosrv1.mpe.mpg.de	MPE	workstation	globusrun-ws CoG-Ki	ClusterFinder <sup>3</sup>

<sup>1</sup> atomic jobs are e.g. Dynamo or NBODY6++, which run with single jsdl-scripts via globusrun-ws.

<sup>2</sup> bladekemper21.informatik.tu-muenchen.de

<sup>3</sup> Deployment and compilation has been verified on about 20 hosts in AstroGrid-D.

## 4.2 Live List of Resources

Currently active compute and storage resources in the AstroGrid-D can be seen at:  
**<http://www.gac-grid.org/project-products/grid-status/astrogrid-mds.html>**

Robotic Telescopes can be seen at:  
**<http://mintaka.aip.de:24050/telescopemap.html>**

## 5 Results of Use Case Testing

The GEO600 use case started to provide production level service on the grid in Summer 2007. A peak grid utilization of 60.000 CPU hours within one week was reached in July 2007, where a number of several thousand jobs was submitted to all major grid resources available at the time. The jobs were submitted by three users at the AEI of whom two can be considered end users of the use case. The current grid utilization is only limited by the availability and reliability of grid resources and grid services. It is actively pushing the integration of resources into AstroGrid in close cooperation with site administrators.

### 5.1 GEO600 Testing

All sources are available in SVN ( `svn://svn.gac-grid.org/software/GEO600` ) together with documentation that enables the end-user to submit GEO600 jobs to the grid. The use case was handed to two end users who successfully are submitting jobs to all major grid resources in AstroGrid.

GEO600 integrates with the Stellaris Information Service, which provides a list of resources available in AstroGrid. Grid service monitoring is being used to detect the availability of grid services to the end user. The AstroGrid storage resource hosted at the AIP is being used to collect all use case data. A built-in accounting provides the use case with statistical data that documents the success of GEO600 utilizing the grid.

#### 5.1.1 Command Line Results

A command line interface to allow job submission, administrative tasks and display of use-case statistics is available together with documentation in the AstroGrid SVN. It can be used from any resource that has Globus client tools installed. It is actively being tested and used by two end users at the AEI.

#### 5.1.2 Scheduler Results

GEO600 can utilize any resource that supports GT4 web services for job submission. The resource specific factory-type ( Fork/PBS/SGE ) can be specified at the command line or is read in from a configuration file. Several thousand jobs were sent to grid resources utilizing queueing systems such as SGE and PBS, where the use case was running on cluster internal nodes ranging from minutes to days depending on the user specified runtime settings. GridWay is being tested as a possible grid scheduler.

## 5.2 ProC Testing

### 5.2.1 Command Line Results

It turned out to be difficult to keep this use case up and running over stretches of weeks. As soon as one did not use it for a while, it stopped working. Since at least three Grid hosts had to be fully operational for the video workflow to produce the desired result (a merged video), we created two workflow variants with completely different hosts. Usually we got one workflow to work, but rarely both of them.

### 5.2.2 Scheduler Results

We did not yet get around to use the GridGateWay resource broker for submitting the video workflows to feasible resources within AstroGrid-D and the core D-GRID. But we are certainly aiming for it. (A beta-version of the wsgram adapter for GAT is now available, and shall be used in the future.)

## **5.3 ClusterFinder Testing**

### **5.3.1 Command Line Results**

Only preliminary testing of job submission using grid technology, as opposed to manual submission after logging in with grid technology, has been performed for Clusterfinder. Both globusrun-ws and CoG-Kit have been used.

### **5.3.2 Scheduler Results**

No attempt has yet been made to submit Clusterfinder jobs through a scheduler-broker.

## 5.4 NBODY6++ Testing

### 5.4.1 Command Line Results

A JSDL/RSL job template of NBODY6++ has been created and thoroughly tested via globusjobs. Since a few months all testing is solely provided through the Gridway- GridGateWay schedulers.

### 5.4.2 Scheduler Results

The present standard way to test and run NBODY6++ and phi-GRAPE jobs is through the Gridway- or Gridgateway scheduling programs. The submission script developed is used to deploy and execute the code. An execution host normally needs not to be specified, but it is possible. Testing by application users and Astrogrid-D members has been done, life and poster demonstrations also at various national and international conferences and workshops, including the SciDac2007 in Boston, the Astronomische Gesellschaft Meeting in Würzburg, the International Supercomputing ISC2007 in Baden-Baden. The standard way of demonstration is to submit a number of 10 or 20 jobs single CPU atomic jobs to the gridway scheduler and observe how it distributes them across the resources. This has been extensively tested for single-CPU jobs so far.

Recently, parallel jobs became possible, the submission script allows to specify this as well as if a GRAPE job is required. This has not yet extensively been tested, mainly because the resource definitions have yet to be done manually (which queues are suitable, which machines have special hardware).

It is required for a real production to work on the Astrogrid-D, as well as a proper mechanism to access files, in particular for restarts. Practical production jobs require a stable and sustained mechanism to access file replicas from previous jobs, and to define long-running parallel jobs for certain queues and hardware (order of weeks). In the near future we hope to be able to port the monitoring, steering, and visualization functionality xtbody from Unicore/visit to globus/gridway. Since there are not many GRAPE cluster like titan in the world, it is desirable for the future as well to have an interoperability of the job submission and file handling mechanisms with other such clusters, e.g. the golowood cluster in Kiev (which is already member in Astrogrid-D) or new clusters coming up in Astrogrid-PL. Since there are not many GRAPE cluster like titan in the world, it is desirable for the future as well to have an interoperability of the job submission and file handling mechanisms with other such clusters, e.g. the golowood cluster in Kiev (which is already member in Astrogrid-D) or new clusters coming up in Astrogrid-PL.

## 5.5 Dynamo, Amiga, Nirvana Testing

### 5.5.1 Command Line Results

The Dynamo Use Case has been tested extensively on AstroGrid-D compute resource and workstations in a static deployment method using command line scripts `dynamo.sh` and subsequently the more general `jsdl_submit.pl`. For details of these scripts see section 3 on submission mechanisms.

Specifically for Dynamo:

- a precompiled static executable is staged along with the input data to the remote resource.
- all attempted jobs sent to active resources were able to launch the executable on the remote resource.
- the input file contains field information used by the executable for field start values
- a running job produces field files.
- the length of the job varies greatly depending on the parameter values in the input file.
- short jobs as well as overnight jobs were run

### 5.5.2 Scheduler Results

The Dynamo Use Case was tested against the GridWay Scheduler:

- the GridWay server was installed on the workstation `dublin.aip.de`
- a GridWay template file was created matching the Dynamo job requirements.
- staging elements had to be listed individually in the template
- the template was submitted to the GridWay instance using `gwsuubmit`
- using `gwps` the job appeared as a process and a host was selected
- the job completed correctly

At this point some difficulties were encountered with staging of directories. Staging directories are useful in the case where many job input or output files need to be staged. Work on implementing a stable installation of GridGateWay is in progress and test submission of Dynamo to a GridGateWay will be attempted.

## 5.6 Robotic Telescopes

The submission and cancellation methods have been tested with the STELLA controller software run in installation mode. It was installed at the telescope server along with the Globus Toolkit and the components of the telescope network software. Observations were scheduled successfully and as the related metadata was automatically uploaded to the central information service.

## 6 Summary

### 6.1 Collation of Use Case Testing

The following table gives a brief summary of the status of the use case tests.

Use Case	Owner	Type	Tested Command-Line	via Scheduler	Comments
GEO600	AEI	atomic	comprehensive	comprehensive	in production
ClusterFinder	MPE	Workflow	comprehensive	No	-
ProC	MPA	Workflow	widely	No	-
NBODY6++	ZAH	Atomic	comprehensive	comprehensive	GRAPE with phiGRAPE
Dynamo	AIP	Atomic	comprehensive	yes, basic	not with GridGateWay
Amiga	AIP	Atomic	widely	No	-
Nirvana	AIP	MPI	as atomic job	No	full MPI testing required
Robotic Telescopes	AIP	Scheduled RTML requests	successful	No	-

For the atomic jobs, the test results indicate that a submission through GridWay/GridGateWay will be possible in the near future. A full testing of a submission of MPI jobs is still pending. The workflow submission is not yet able to use the Scheduler.

## Appendix

### A Definitions/Concepts

A set of common definitions and concepts used in the document are listed here.

#### A.1 Use Cases

A 'Use Case' is based on actual research topics and associated software. A Use Case is developed to help identify what components are required in the AstroGrid-D. This facilitates making resources available to scientists and the testing of implemented components.

#### A.2 Atomic job

Atomic Use Cases are grid jobs with simple submission requirements (no workflows). Examples of Atomic Use Cases from the AstroGrid-D Community are NBODY6, GEO600, DYNAMO

#### A.3 Data Staging

Data Staging is the process of transferring required data for job start to the remote resource (Stagein) and returning required data back to a specified location after job completion (Stageout)

#### A.4 MPI Job

An MPI job is one which passes messages between concurrently running processes which can be running on different nodes within a cluster. Special libraries need to be linked into the application. These libraries depend on the platform implementation.

**F: References / Bibliography**