AstroGrid-D: Enhancing Astronomic Science with Grid Technology

Harry Enke, M. Steinmetz, *Astrophysical Institute Potsdam (AIP), An der Sternwarte 16, 14482 Potsdam*
Thomas Radke, *Max-Planck-Insitute for Gravitational Physics, Albert-Einstein-Instiute (AEI), Am Mühlenberg 1, 14476 Golm*
Angelika Reiser, *Technische Universität München (TUM), Institute for Informatics, Boltzmannstraße 3, 85748 Garching*
Thomas Röblitz, Mikael Högqvist, *Zuse Institute Berlin (ZIB), Takusstraße 7, 14195 Berlin-Dahlem*

Abstract

We present AstroGrid-D, a project bringing together astronomers and experts in Grid technology to enhance astronomic science in many aspects. First, by sharing currently dispersed resources, scientists can calculate their models in more detail. Second, by developing new mechanisms to efficiently access and process existing datasets, scientific problems can be investigated that were until now impossible to solve. Third, by adopting Grid technology large instruments such as robotic telescopes and complex scientific workflows from data aquisition to analysis can be managed in an integrated manner. In this paper, we present prominent astronomic use cases, discuss requirements on a Grid middleware and present our approach to extend/augment existing middleware to facilitate the improvements mentioned above.

1 Introduction

AstroGrid-D [5] is a research and development project within the *German D-Grid Initiative* [7]. Five major German astronomy institutes as well as experts in Grid technology from computer science groups at supercomputing centers and universities have joined in an interdisciplinary partnership to pursue several strategic goals:

- build a nation-wide Grid-based infrastructure for astronomical and astrophysical research by embedding existing computational facilities at partner institutes,
- integrate this Grid infrastructure into a common D-Grid production environment,
- integrate astronomical data archives, instruments, and experiments into a common research infrastructure
- provide support for other astronomical institutes to join this infrastructure, thus enabling even small research groups from universities to benefit from

AstroGrid-D efforts
- strengthen the international collaboration of the German astronomy community

AstroGrid-D's overall goal is to establish a collaborative working environment which facilitates a wide variety of use cases described in Section 2. The AstroGrid-D middleware, as presented in Section 3, builds on existing Grid tools to integrate diverse types of resources. Due to the specific requirements of the AstroGrid-D community existing components have to be extended or substituted by newly developed components. However, to let other communities benefit from these developments, we aim at generic solutions wherever possible.

2 Astronomic Use Cases

In an extensive survey, we collected nearly twenty scientific use cases from the partners in AstroGrid-D. On an abstract level, the scenarios can be divided into three main classes: (a) large scale simulations, (b) analysis of data sets (derived through actual observations or simulations), and (c) integration of astronomical instruments such as robotic telescopes. In the following, we present use cases of these classes in more detail.

2.1 Large Scale Simulations

Simulations model physical processes in order to understand the building of the observed structures of the universe, the formation of galaxies and stars or the dynamics of our own sun and solar system. The type of simulations ranges from taskfarming, e.g. exploring parameter spaces for dynamical models of the solar dynamo, to large scale cosmological simulations running on supercomputers such as Mare Nostrum, that produce terabytes of data for postprocessing, e.g. for comparison with observational results.

The Grid environment can relieve two main obstacles that these huge simulations face. Today's high performance computing (HPC) centres require scientists to learn about the HPC resources in order to submit their jobs and manage their data. Grid infrastructure leverages this burden by providing a unified interface to grid-enabled resources of the HPC. Most of the HPC centers do not offer longterm storage and postprocessing facilities for the results. Postprocessing, which is mostly done on smaller compute facilities, requires access to the huge datasets from very different sites or machines. Today this requires copying the data to the different locations. The Grid infrastucture offers more efficient ways to manage and access these datasets. AstroGrid-D is working on file management services and metadata services, which offer these mechanisms. In the second phase of the project, inclusion of postprocessing facilities for simulation data, e.g. for visualisation of snapshots, is planned. In collaboration with the German Astrophysical Observatory (GAVO) [10], AstroGrid-D is working on grid-based access to databases, which store information from halos and other objects extracted from such simulations.

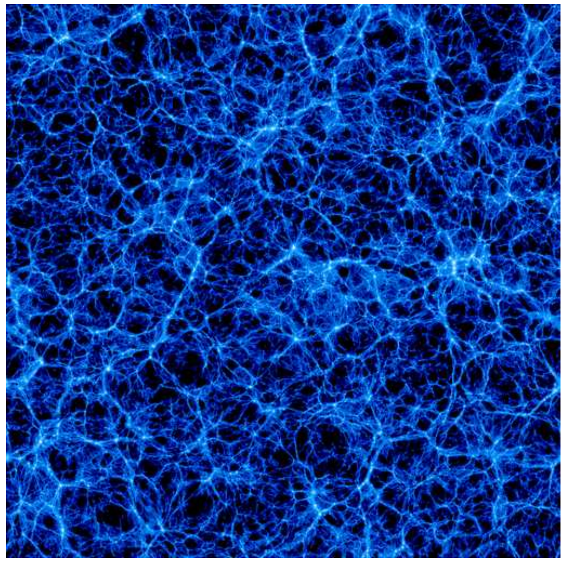

Figure 1: MareNostrum simulation [18], with 1024^3 dark matter and 1024^3 gas particles in a 500 Megaparsec/h box, simulating the formation of large scale structures in the universe. Current data size (spanning 1.34 GigaYear of simulation time) is 10 TByte. The picture shows the filaments and halos from a snapshot.

Taskfarming jobs are required for preparation of large scale simulations. They benefit from the Grid infrastructure since they now have a multitude of resources available, on which exploration of a given parameter space is possible in parallel. This kind of Grid job runs with the basic facilities of the Grid middleware and use cases such as Dynamo or Amiga taht are already working examples. Additionally, AstroGrid-D is developing a GridSphere [9] portlet for this type of Grid jobs.

2.2 Analysis of Data Sets

One of the most important challenges of e-science is the analysis of huge volumes of scientific data and the interconnection of various sciences and communities, thus enabling scientists to share scientific interests, data, and research results. In the area of astrophysics there are many application use cases which require a complex processing of distributed data with a volume of often more than 1 TB. AstroGrid-D use cases like e.g. Astrometric Matching, Clusterfinder, Virtual Telescope, Millennium Query, Millennium PostProc, Geo600 [5] either require access to data archives as SDSS or ROSAT, or to databases which store extracted information from simulations.

From those catalogues and databases which often have very different formats and interfaces, subsets need to be filtered out and combined using sophisticated algorithms. For the filtering database and/or partial file access is required. For the results, temporary storage is needed and must be accessible to the user. As the data volume is huge and the catalogues are widely distributed, techniques for discovering, early filtering, and pipelining the data (data streaming) have to be applied, as well as other mechanisms for parallelisation and load-balancing of the complex computing process.

In the context of the Grid we are able to overcome the limits from the centralized data processing approach where large volumes of data are shipped to the application. Two roads are currently followed: a) distributing the data processing within the Grid and use of storage facilities accessible via Grid methods and b) using mobile, dynamically loadable functions to bring the processing operators close to the data to be analyzed.

As an advanced scenario we work on the use cases Stella and PLANET [5] to process, also in conjunction with the robotic telescopes, event streams and event-driven follow-up observations in the Grid.

2.3 Integration of Robotic Telescopes

The integration of robotic telescopes into a worldwide network can provide important advantages for astronomical observations which require the coordinated participation of multiple telescopes. For example, multi-wavelength campaigns or the continuous monitoring of transient astronomical objects currently

Figure 2: Robotic telescopes STELLA-I and STELLA-II at Tenerife.

require a lot of mutual coordination between the participating groups. Despite the rather high effort these are more likely to be subject to failure due to e.g. adverse weather or technical problems. However, within a network of robotic telescopes those problems can be overcome: by combining distributed telescope resources the total efficiency increases and the likelihood of failure for observations of high priority can be reduced. The advantage of the telescope network can be illustrated by comparing the average probability (p) for the completion of a successful observation of a gamma ray burst. While for a single telescope p is limited to ≈ 0.12 by e.g. weather, daytime and altitude of the object, p can reach the theoretical limit of ≈ 0.95 in a network of telescopes, only limited by the minimal required distance of the object from the sun. The efforts within AstroGrid-D aim at the integration of robotic telescopes into the AstroGrid-D by adopting Grid technology. The goal is to develop a uniform interface to telescope resources with a powerful monitoring system and advanced scheduling mechanism, which can provide the basis for a global network of telescopes resources.

By adopting Grid technology a telescope network can immediately benefit from the accomplished Grid developments. Particularly useful for the operation of a robotic telescope network are the security layer, the information service and the Grid infrastructure including storage, databases and bandwidth. Moreover, global schedulers will automatically coordinate and optimize the observations, but these are still to be developed. Finally, the coordination and scheduling of an observation as well as the retrieval of the data becomes very efficient and convenient for the user.

3 Requirements Analysis

The AstroGrid-D use cases unveiled many requirements of a middleware. In this Section, we summarize these requirements and assess which of them are fulfilled by existing Grid middleware components.

Integration of Compute and Data Resources. The AstroGrid-D community owns or has access to a large diversity of compute resources – from personal workstations, over PC clusters to high-performance supercomputers. Some of them are already using Grid middleware such as Globus [8] or Unicore [19] and cannot replace them with AstroGrid-D specific implementations. Similarly, data storing resources provide different access methods such as HTTP or GridFTP [8]. Hence, the middleware must be very flexible with respect to the core layer of the resource management system.

Authentication, Authorization and VO-Management. To secure the access to resources – compute, storage or metadata – AstroGrid-D users and services need to authenticate themselves and to be authorized by the accessed resource. For authentication and authorization, typically, X.509 certificates [16] and admission control, e.g. by the Globus *grid-mapfile*, are used. The AstroGrid-D community consists of many groups which possess different rights to access resources, in particular, data resources, i.e. results from observation or large simulations. Each group may form a distinct virtual organization (VO). Hence, the middleware must support the management of several VOs within AstroGrid-D.

Handling of Metadata. In AstroGrid-D, we distinguish several types of metadata: (a) resource information, (b) state of activities, (c) application-specific information and (d) scientific metadata. All shall be supported by a single distributed component. The advantage of a single component is its uniform interface and the ability to deduce relations of the metadata of the different types more easily. Because, the schemes of these types are not fixed or subject to future changes, the metadata management must be very flexible. Additionally, it must facilitate enhanced access restrictions if many different VOs will use the same service instance. Resource information are typically managed by the Globus Toolkit Monitoring and Discovery System (MDS) [8].

Storing and Provisioning of Files. Most of the use cases process large input files and/or create large results. Input files need to be transferred from a data storage to the job's execution site, while output files must be copied to a permanent archive after the job has finished. For these basic tasks, existing tools such as GridFTP or Globus RFT already provide most of the needed functionality. An open issue, however, is to manage distributed data servers such that a user need not know where his files are stored. Thus, AstroGrid-D middleware shall provide a virtualization layer, which not only simplifies access to distributed data, but also facilitates the easy integration of additional data resources.

Access to Databases. Many use cases access data stored in relational databases. However, the current procedure often involves a pre-processing step which extracts the data and stores it into a file. Existing tools such as OGSA-DAI [3] may be used to directly access the data from within a Grid application. The volume of the data is often too big to be sent to the processing nodes without pre-filtering. Databases allow semantically-rich filtering operations. These filter operators have to be installed near the data, preferably dynamically.

Processing via Data Streams. Some use cases are very demanding with respect to the processed data volumes and consist of a complex workflow. Simply gathering all data from distributed (remote) databases into memory and processing it locally, may not be possible or too time consuming. Hence, the middleware shall support the intelligent distribution of the workflow onto the Grid and the streaming of the data through the nodes processing the single steps of the workflow. Herewith the amount of storage needed for intermediary results is reduced also. Advanced use cases like, e.g. Stella and PLANET, also demand the processing of realtime data streams coming from sensors. This can then smoothly be integrated into the middleware component for the streamed processing of the data.

Job Submission and Execution. Most AstroGrid-D jobs posess no specific requirements on job management, e.g. they specify the architecture of the processors, the version of the operating system, the number of processors, their estimated runtime, the input files, etc. However, some jobs require special hardware (GRAPE boards) or are dependent on the result of preceeding jobs, i.e. in a workflow application. Thus, the description and the execution of jobs must pay attention to these additional requirements. For these purposes many solutions do exist such as Globus RSL or the OGF's Job Submission Description Language (JSDL) [12] for describing jobs, and Grid brokers such as GridWay [14] for determining matching resources for a job.

Job and Application-Specific Monitoring. In addition to generic job monitoring, provided through the Globus GRAM interface, specific runtime progress information should be available especially for long-running astronomic simulations. Proprietary monitoring mechanisms already exist for some AstroGrid-D use cases. They need to be integrated into a common solution if such applications are executed on a heterogeneous Grid infrastructure.

Integration of Robotic Telescopes. Robotic telescopes are a unique resource within the astronomic community. The resources as well as the observation requests are described in the Remote Telescope Markup Language (RTML) [11]. Access to a telescope and to the results of observation must be secured. Unlike compute jobs, observation depend on external conditions such as daylight or weather, which must be taken into account if requests are distributed to a network of worldwide telescopes. Within the astronomic community, the Heterogeneous Telescope Network (HTN) initiative has developed the RTML and sketched a method for distributing observation requests [1].

User and Developer Interfaces. The AstroGrid-D middleware shall provide an easy way to use the Grid infrastructure, i.e. to find data sets or resources and execute applications on them. While few scientists are already using Grid technology, many of them do not feel very comfortable with the existing interfaces provided by standard toolkits such as Globus. Graphical user interfaces embedded into WWW servers, based on standard web portal frameworks such as GridSphere [9], should be offered as alternatives taylored to the specific needs of AstroGrid-D applications. Both, the application and the middleware component developer, are faced with a large variety of existing tools and APIs to program them. Hence, their burden to use the Grid and develop advanced components will be lowered if standardized protocols or generalized APIs can be used. In this respect, the Grid Application Toolkit [2] provides a uniform interface to basic management tasks such as job submission or copying files and maps these to the appropriate methods applicable in a specific Grid middleware enviroment.

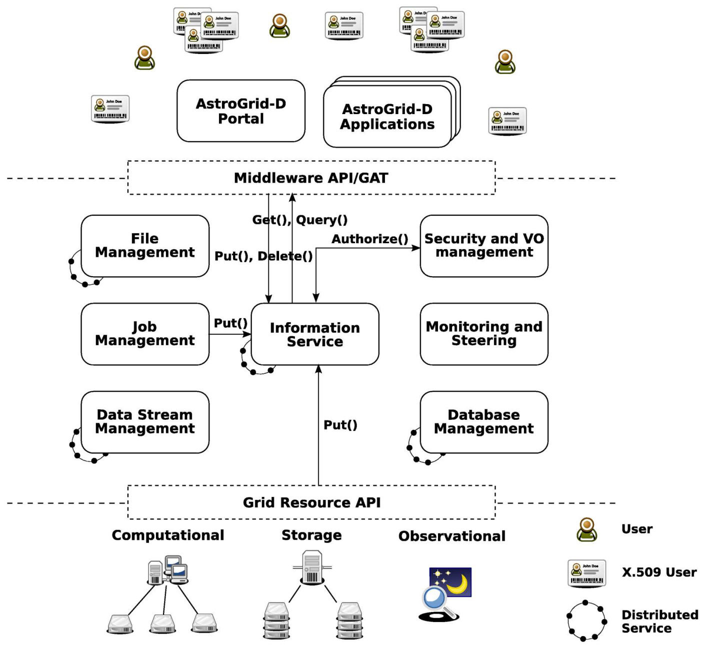

Figure 3: Architecture of AstroGrid-D, showing only component interactions already implemented with the information service. Other interactions between the components as described in this section are not shown.

4 Architecture of AstroGrid-D Middleware

4.1 Resource integration

AstroGrid-D early on decided to use Globus Toolkit 4.0.x (GT4) as Grid middleware [4]. GT4 provides GridFTP and RFT for data transport, remote execution of shell commands, via GRAM a job submission and management facility and via MDS a monitoring system for Grid resources. All connectivity of GT4 is built on X.509 certificates, both for ssh or ftp and webbased connections.

4.2 Security and VO Management

The GT4 security layer meets the moderate security requirements of the community. One of the core features of a Grid supported collaborative working environment, a VO-Management system, was designed using the Globus gridmap mechanism. With a combination of VOMRS[6] and java or perl based tools for local Grid user management, AstroGrid-D implemented a VO-Management, which handles multiple VOs and subVOs. [4].

4.3 Information Service

The AstroGrid-D Information Service (Stellaris) was designed and implemented to handle the four metadata classes defined in Section 3 (resource information, state of Grid activities, application and scientific-related metadata). In order to unify these classes, we use the RDF [13] information model in combination with the SPARQL query language [15]. An HTTP-based interface is used to create, retrieve, update, delete and query metadata. User authentication and authorization is done with X.509 proxy certificates in combination with the VOMRS service.

4.4 File Management

The file management is responsible for locating and transferring files between distributed storage resources in AstroGrid-D. A distributed index is used to manage replicas and to provide a unified view of files. In order to increase file retrieval efficieny an interface for partial file access is supported. Additionally, the file transfer throughput is increased by using parallel transfer techniques. Basic protocols such as GridFTP and HTTP are available as fallback [4].

4.5 Database Access and Datastream Management

Many data catalogues in astrophysics are stored in databases which must at the same time be publicly available and accessible in the Grid environment. We use the OGSA-DAI software to achieve this condition [4]. For enabling intelligent in-grid processing of data we develop a data stream management system which is implemented as a GT4 service. Herewith we are able to handle large data volumes efficiently using early filtering, parallelization, and pipelining. We are thereby able to return first results early on and to support domain-specific application logic in a flexible way by the use of mobile, user-defined operators.

4.6 Job Management

Because the job management must support a variety of job classes – basic jobs, jobs requiring special hardware or workflows – we decided to use an extended version of JSDL [12] to describe jobs. For submission the job description is translated into the format which is accepted by the middleware of the accessed resources, e.g. Globus RSL for resources using the Globus Toolkit. As Grid resource broker we use the GridWay Metascheduler [14]. We will adapt GridWay to integrate it with the AstroGrid-D information service, i.e. to retrieve the status of the resources and to update the status of the jobs.

4.7 Generic and Application-Specific Job Monitoring

Generic job management is based on standard mechanisms provided by existing middleware such as Globus. However, to integrate job information from the middleware with other components, for example the portal, we are developing a generic metadata schema to store basic runtime information about file staging, job start, execution and termination into Stellaris. Because we also store a timestamp with each job metadata record, we are able to to extract job histories for each user or to track execution problems of jobs.

Long-running, large scale simulations will benefit from application-specific job monitoring and steering mechanisms. Often, the applications already support proprietary protocols for accessing progress information and for controlling their ongoing processing [4]. These applications use the schema flexibility of AstroGrid's information service to add and update specific metadata. The metadata can easily be accessed by scientists via portlets tailored at the application requirements.

4.8 User and Developer Interfaces

GridSphere [4] is used as a de-facto standard framework to build the portlet-based AstroGrid-D web portal. For example, we are currently developing a portlet to provide a common interface for the task farming scenarios. Other portlets are being developed to build application-specific web frontends such as the Cactus user portal [17], which will provide customised access to application metadata stored in the AstroGrid-D information service and allow users to monitor and steer their astrophysical simulations. In addition, the GridSphere implementation is partially reworked to support the new GT4 WSRF [8] based services.

As common interface for developing Grid application and middleware components we will use the Grid Application Toolkit (GAT) [4]. In a first step we implemented an adaptor to submit jobs to PC clusters managed by Sun Grid Engine.

5 Conclusions

By building on Globus middleware AstroGrid-D stays compatible with the development lines of the Grid. An efficient VO-Management was achieved by a combination of tools from HEP-Grid and AstroGrid's own design. Adhering to the service-oriented architecture, Grid services are being developed to efficiently manage metadata and data (either stored in files or in databases). Both draw on specifications from standardisation initiatives such as OGF and W3C and use already implemented components like OGSA-DAI and GAT. This already enables us to cover a fair range of requirements derived from the uses cases of our community. By enhancing job submission and handling with JSDL and GridWay a first version of an AstroGrid-D production Grid will be ready by mid 2007.

Acknowledgments

This work is supported by the German Federal Ministry of Education and Research within the D-Grid initiative under contracts 01AK804[A-G].

References

1. A. Allan et al. A protocol standard for heterogeneous telescope networks. *Astronomische Nachrichten*, 327:744–+, September 2006.
2. G. Allen et al. The Grid Application Toolkit: Toward Generic and Easy Application Programming Interfaces for the Grid. *Procceedings of the IEEE*, 93(3):534–550, March 2005.
3. M. Antonioletti et al. The design and implementation of Grid database services in OGSA-DAI. *Concurrency and Computation: Practice and Experience*, 17(2-4):357–376, 2005.
4. AstroGrid-D. Project Deliverables. `http://www.gac-grid.org/project-documents/deliverables.html`.
5. AstroGrid-D. Project homepage. `http://www.gac-grid.org/`, December 2006.
6. L. Bauerdick et al. The virtual organization management registration service. In *Proceedings of the CHEP 06, Mumbai, India*, February 2006.
7. D-Grid Initiative. Project homepage. `http://www.d-grid.de/`, December 2006.
8. Ian T. Foster. Globus toolkit version 4: Software for service-oriented systems. In *Proceedings of the IFIP International Conference on Network and Parallel Computing (NPC 2005), Beijing, China*, pages 2–13, November 2005.
9. GridSphere Portal Framework. `http://www.gridsphere.org`, December 2006.
10. German Astrophyiscal Virtual Observatory. `http://www.g-vo.org`.
11. F. V. Hessman. Remote Telescope Markup Language (RTML). *Astronomische Nachrichten*, 327:751–+, September 2006.
12. Job Submission Description Language (JSDL), Specification, Version 1.0. Technical Report GFD-R.056, Global Grid Forum, November 2005.
13. F. Manola and E. Miller. RDF primer. `http://www.w3.org/TR/rdf-primer/`, February 2004.
14. GridWay Metascheduler. `http://www.gridway.org`, December 2006.
15. E. Prud'hommeaux and A. Seaborne. SPARQL query language for RDF. `http://www.w3.org/TR/rdf-sparql-query/`, February 2006.
16. Public-key and attribute certificate frameworks. Recommendation X.509 (08/05). Technical Report T-REC-X.509-200508-I, ITU, August 2005.
17. The Cactus Users Portal. `http://portal.cactuscode.org`, December 2006.
18. The MareNostrum Universe. `http://astro.ft.uam.es/~marenostrum/`.
19. UNICORE. Project Homepage. `http://www.unicore.eu/`, December 2006.