Grid-Based Galaxy Morphology Analysis for the National Virtual Observatory

Ewa Deelman
Information Sciences Institute, University of Southern California, Marina Del Rey, CA 90202 (ISI),
deelman@isi.edu

Raymond Plante
National Center for Supercomputing Applications, Champaign, IL 61820
rplante@ncsa.uiuc.edu

Carl Kesselman, Gurmeet Singh, Mei-Hui Su
Information Sciences Institute, University of Southern California, Marina Del Rey, CA 90292 (ISI),
{carl, gurmeet, mei}@isi.edu

Gretchen Greene, Robert Hanisch, Niall Gaffney, Antonio Volpicelli,
Space Telescope Science Institute, Baltimore, MD 21218
{greene,hanisch,gaffney,volpicelli}@stsci.edu

James Annis, Vijay Sekhri
Fermi National Accelerator Laboratory, Batavia. IL 60510
{annis,sekhri}@fnal.gov

Tamas Budavari, Maria Nieto-Santisteban, William O'Mullane
Johns Hopkins University, Baltimore, MD 21218
{budavari,nieto,womullan}@pha.jhu.edu

David Bohlender
Canadian Astrophysical Data Center, Victoria, British Columbia
David.Bohlender@nrc.ca

Tom McGlynn
NASA High Energy Astrophysics Science Archive Research Center, Greenbelt, MD
tam@lheapop.gsfc.nasa.gov

Arnold Rots
Smithsonian Astrophysical Observatory, Cambridge, MA
arots@head-cfa.harvard.edu

Olga Pevunova
NASA Infrared Processing and Analysis Center, Pasadena, CA
olga@ipac.caltech.edu

Abstract
As part of the development of the National Virtual Observatory (NVO), a Data Grid for astronomy, we have developed a prototype science application to explore the dynamical history of galaxy clusters by analyzing the galaxies' morphologies. The purpose of the prototype is to investigate how Grid-based technologies can be used to provide specialized computational services within the NVO environment. In this paper we focus on the key enabling technology components, particularly Chimera and Pegasus which are used to create and manage the computational workflow that must be present to deal with the challenging application requirements. We illustrate how the components interplay with each other and can be driven from a special purpose application portal.

1 Introduction

The National Virtual Observatory (NVO) initiative can be concisely described as a Data Grid for astronomy. Its goal is to provide integrated access to distributed databases, archives, and computational services in order to enable efficient and previously impractical, labor-intensive research in astronomy [Brunner 2001; Hanisch 2001a; us-vo]. The NVO will benefit from the large and growing collections of digital astronomical imagery, spectra, and time series data being collected by ground-based observatories and NASA space missions. By implementing metadata standards and standard access protocols, the NVO will allow these diverse and distributed information resources to be utilized by researchers as a single entity. Moreover, analysis and intercomparison of terabyte to Petabyte scale databases will be enabled by closely linking the data resources to the computational power of the emerging Grid. The Grid is an attractive infrastructure, because NVO applications will perform computationally complex analysis on large, heterogeneous and distributed datasets.

Today, the NVO is a project in its early stages. Currently, the astronomical research community has public, on-line access (primarily via the Web) to thousands of data collections (images, catalogs, spectra, literature, and other various datasets) through a large number of different sites [Hanisch 2000; Hanisch 2001a]. These collections range in size from a few kilobytes to terabytes. In the early phases of NVO growth, registries, data information services, and data access services will help users locate and retrieve data in a uniform way without having to visit many sites and learn their specialized interfaces. As the project evolves, we see general-purpose services becoming available. These services can perform common data manipulation and integration tasks, such as one that can join two tables or convert astronomical coordinates from one system to another. In its full maturity, there will also be many highly-specialized services dedicated to specific forms of analysis, such as those that classify objects in an image by their morphologies. While advanced users of the NVO will provide processing algorithms and create new services, the vast majority of the NVO user community will be non-programming astronomers using portals to connect existing services to conduct research.

In pursuing these goals, the NVO team is utilizing a number of scientific prototypes as demonstration projects [us-vo]. Our initial set of prototypes attempts to capture various aspects of the different levels of complexity described above. Their purpose is to help to validate the standards and protocols and show the user community part of the promise of the Virtual Observatory. The prototypes allow us to test our designs and implementations and to improve them prior to general release. With the help of the end-to-end prototypes, we have been able to discover a number of deficiencies in our system.

This paper describes one of the NVO science prototypes aimed at studying the morphological properties of galaxies in rich clusters. The Galaxy Morphology prototype is an example of a highly-specialized analysis service. It has many of the key technical elements of the astronomical problem (data to be found at many data centers, data representing different wavelengths, integrating heterogeneous tabular data) and combines them with the need to perform dynamic computations on the data. The prototype needs to support the following operations: find online catalogs of galaxies in clusters, obtain images of the many hundreds of those galaxies, compute a set of morphological parameters on those images using Grid computing, and integrate the new results into the catalogs. The prototype is illustrative of a large class of NVO science challenges, where information available from generally available, precomputed image properties are insufficient to address new research questions.

The remainder of the paper is structured as follows: Section 2 explains the scientific goals of the galaxy morphology prototype, Section 3 describes the technologies used in our work, Section 4 details the end-to-end prototype, including the portal, and finally Section 5 provides the results and conclusions.

2 Science Goals: The Dynamical State of Galaxy Clusters

Our goal is to investigate the dynamical state of galaxy clusters and to explore galaxy evolution inside the context of large-scale structure. We use galaxy morphologies as a probe of the star formation and stellar distribution history of the galaxies inside the clusters. The hypothesis is that recent falling of matter into the cluster, be it in the form of single galaxies or cluster mass groupings, will show the effects of the merging into the main cluster mass through star formation events [Ellingson 2003].

The morphology of a galaxy is related to the past star formation, as well as the dynamical environment the stars find themselves in. By mapping morphologies one might be able to see large scale events in the history of the galaxy cluster, those that have occurred more recently than the few Giga-year mixing time of a cluster. Galaxy morphologies are known to be related to the local density of galaxies [Dressler 1980]; the morphologies are also possibly related to the distance of the galaxy from the center of the cluster it inhabits. Our science model examines the distribution of star formation indicators, both spectral and morphological, as a function of cluster radius, local density, and x-ray surface brightness.

In our demonstration, we characterize a galaxy's morphology in terms three parameters that can be calculated directly from an image of the galaxy [Conselice 2003]:

- **Average Surface Brightness** – a measure of the total amount of detected light (per area) from the galaxy.

- **Concentration Index** – a measure that differentiates between galaxies with a uniform distribution of brightness and those dominated by a bright core.
- **Asymmetry Index** – a measure that differentiates between spiral galaxies (most asymmetric) and elliptical galaxies (most symmetric).

The computational requirements for calculating these parameters for a single galaxy are fairly light; however, to statistically characterize a cluster well, we need to calculate our parameters for the hundreds or thousands of galaxies that constitute the galaxy cluster. Ultimately, we would like to apply this technique to hundreds of clusters, analyzing hundreds of thousands of galaxies; at that level, the statistical sampling would be sufficient to gain insight into the evolution of our Universe as a whole.

There is also a significant amount of knowledge that we already have about many of the galaxies that make up clusters, such as their colors, their brightness at different frequencies, and their motions within the cluster. This information is available from the many, various catalogs accessible on-line. In addition, we can also make use of other image sources that contain images that reveal important large-scale characteristics of the galaxy cluster, such as the x-ray emission that traces the hot inter-galactic gas. We can assemble a picture of the dynamical state of a cluster by gathering together this existing information and comparing it with our new calculations. Clearly, as we extend our study to many galaxy clusters, efficient access to the data becomes important.

Our strategy, for the single galaxy cluster, is as follows:

1. Choose a cluster.
2. Obtain optical and x-ray images revealing the large-scale structure of the cluster.
3. Create a catalog of the galaxies making up the cluster, including any interesting properties from existing catalogs.
4. Obtain a "cutout" image for each individual galaxy—that is, an image extracted from a larger one but which contains only that galaxy.
5. Calculate the morphology parameters from the cutout images
6. Merge the calculated values into the galaxy catalog.
7. Visualize the results.

The goal, then, of the NVO Grid is to make the process simple and scalable.

3 Enabling Standards and Technologies

An important purpose behind our science prototypes is to understand the important technical components of our application, particularly those that might be common to many NVO applications [us-vo] (e.g. data access, interchange, and transformation; common metadata schemas; distributed queries; and workflow management). In this section, we highlight some of the core technologies important for accessing data and performing the computations.

3.1 Data Formats and Data Access Interfaces

A study that involves integrating large amounts of diverse data has the challenge of dealing with the problem of formats and access interfaces. Thus, a major effort of the

NVO project is to develop the standard formats and interfaces that allow one to handle various types of data in a uniform way.

The astronomy community is fortunate to have the well-established, internationally accepted standard format for exchanging astronomical images and tables known as FITS [Hanisch 2001b]; we use this standard in the all our NVO demonstrations to transport images. Augmenting this format is VOTable, an XML schema for transmitting astronomical tables [VOTable]. Developed through an international collaboration of Virtual Observatory projects, this format can be more convenient than FITS within a web context: by virtue of being XML, VOTable is readily created and manipulated with off-the-shelf tools. As described in Section 4, this proved useful for integrating with the Chimera and Pegasus software [Foster 2002, Deelman 2003, Blythe 2003].

Two standard interfaces provided by the data resources of the NVO project allowed us to access data from the various astronomy catalogs in a uniform way. The *Cone Search* protocol [cone] defines an interface for searching and retrieving records from an astronomical catalog over the web. The *Simple Image Access* protocol (SIA) [sia] is a similar interface for images. This latter interface is general enough to provide access to both simple static images from an image archive (as is needed for obtaining large-scale images of a cluster) and custom cutout images from an image cutout service. It is worth noting that both of these interfaces use position in the sky as the primary data selection criterion; that is, the interface includes domain-specific features. How this selection is handled in detail may vary greatly among the repositories that implement the interface. It is also important to recognize the prototype status of these interfaces: like the demonstration as a whole, they are defined primarily to help us understand the problem of building an astronomy Data Grid. Based on HTTP Get operations, the interface is simple, highly-specialized, and meant to be easy for data providers to connect to their existing services in the language of their choice. Successors to these interfaces (now in development) will likely leverage emerging standards such as Web Services and OGSA-DAI and will employ more sophisticated techniques for accessing large amounts of data efficiently.

3.2 Chimera and Pegasus

The NVO applications are characterized by complex queries, which refer to data in many distributed, heterogeneous data catalogs. Additional requirements are to support a large problem solution space and a large user community. In response to the queries, the initial data may be processed in many intermediate steps, those results are less costly to store than re-derive. As a result, the prototype needs to be able to derive data on demand.

Chimera [Foster 2002] and Pegasus [Deelman 2003, Deelman 2003b, Blythe 2003, Blythe 2003b, Blythe 2003b] are part of the GriPhyN Virtual Data System (VDS) which enables efficient on-demand data derivation. They allow users and applications to describe data products in terms of abstract workflows and to execute these workflows on the Grid. Unlike other workflow mapping systems (described in Section 3.3), Pegasus allows for easy reuse of existing, intermediate data products.

GriPhyN (Grid Physics Network) [griphyn] is an NSF-funded project that aims to support large-scale data management in physics experiments such as high-energy physics, astronomy and gravitational wave physics. GriPhyN puts data both raw and derived under the umbrella of Virtual Data. A user or application can ask for data using application-specific metadata without needing to know whether the data is available on some storage system or if it needs to be computed. To satisfy the request, GriPhyN will schedule the necessary data movements and computations to produce the requested results.

Using the Chimera *Virtual Data Language* (VDL), the user can describe *transformations,* which are general descriptions of the transformation (e.g. an executable program) applied to data, and *derivations,* which are instantiations of these transformations on specific datasets. The transformations enumerate the input and output files as well as define the parameters used by the program. The derivations name the particular input files and provide the actual parameters used to produce desired files. An example of the galaxy morphology transformations described in VDL is shown below.

```
TR      galMorph( in redshift, in pixScale, in zeroPoint, in Ho, in om, in flat,
                in image, out galMorph ) {
                …  }
```

The "in" prefix in front of the argument name signifies that it is an input argument and the "out" prefix signifies that it is an output argument. The transformation is a template for the program and its parameters. In order to be able to construct the desired workflow, the user also needs to define derivations, where the transformation template parameters are instantiated and the files are indicated by their logical names. Below is an example derivation (arbitrarily named d1), for a single invocation of the transformation.

```
DV d1->galMorph(
     redshift="0.027886",
     image=@{in:"NGP9_F323-0927589.fit"},
     pixScale="2.831933107035062E-4",
     zeroPoint="0",
     Ho="100",
     om="0.3",
     flat="1",
     galMorph=@{out:"NGP9_F323-0927589.txt"}  );
```

When a user or application requests a particular logical file name, Chimera composes an *abstract* workflow based on the previously defined derivations (if that composition is possible). For example, assume there are two derivations, d_1 and d_2, where d_1 takes an input file *a* and produces an input file *b,* and d_2 takes an input file *b* and produces an output file *c*. If a user requests file *c*, Chimera will produce the workflow in Figure 1. This workflow is termed abstract, because it describes the desired data product in terms of logical filenames and logical transformations without specifying the resources that will be used to execute the workflow.

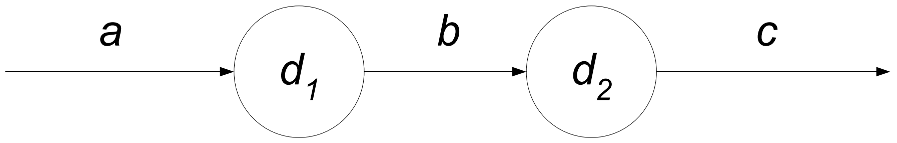

Figure 1: Abstract Workflow.

Pegasus, which stands for Planning for Execution in Grids [Deelman 2003, Deelman 2003b, Blythe 2003, Blythe 2003b, Blythe 2003c], was developed at ISI as part of the GriPhyN project. Pegasus is a configurable system that can map and execute workflows on the Grid. In particular, Pegasus can map an abstract workflow onto the available Grid resources.

In the configuration described here, Pegasus receives an abstract workflow (AW) description from Chimera, produces a concrete workflow (CW), and submits it to Condor-G/DAGMan [Frey 2001] for execution.

The workflows are represented as Directed Acyclic Graphs (DAGs). The AW describes the transformations and data in terms of their logical names; it has information about all the jobs that need to be executed to "materialize" the required data. The Pegasus Request Manager sends this workflow (labeled "Abstract DAG" in Figure 2) to the Concrete Workflow Generator. In order to locate the input data in the Grid environment, Pegasus uses services such as the Globus Replica Location Service (RLS) [Chervenak 2002]. The input files are specified by their logical filenames in the DAG. Using RLS, Pegasus finds the list of physical locations for these files.

The information about the available data is then used to optimize the concrete workflow from the point of view of Virtual Data. The optimization is performed by the Abstract DAG Reduction component of Pegasus. If data products described within the AW already exist, Pegasus reuses them and thus reduces the complexity of the CW. In general, the reduction component of Pegasus assumes that it is more costly to execute a component (a job) than to access the results of the component if that data is available. For example, some other user may have already materialized (available on some storage system) part of the entire required dataset. If this information is published into the RLS, Pegasus can utilize this knowledge and obtain the data, thus avoiding possibly costly computation. As a result, some components that appear in the abstract workflow do not appear in the concrete workflow.

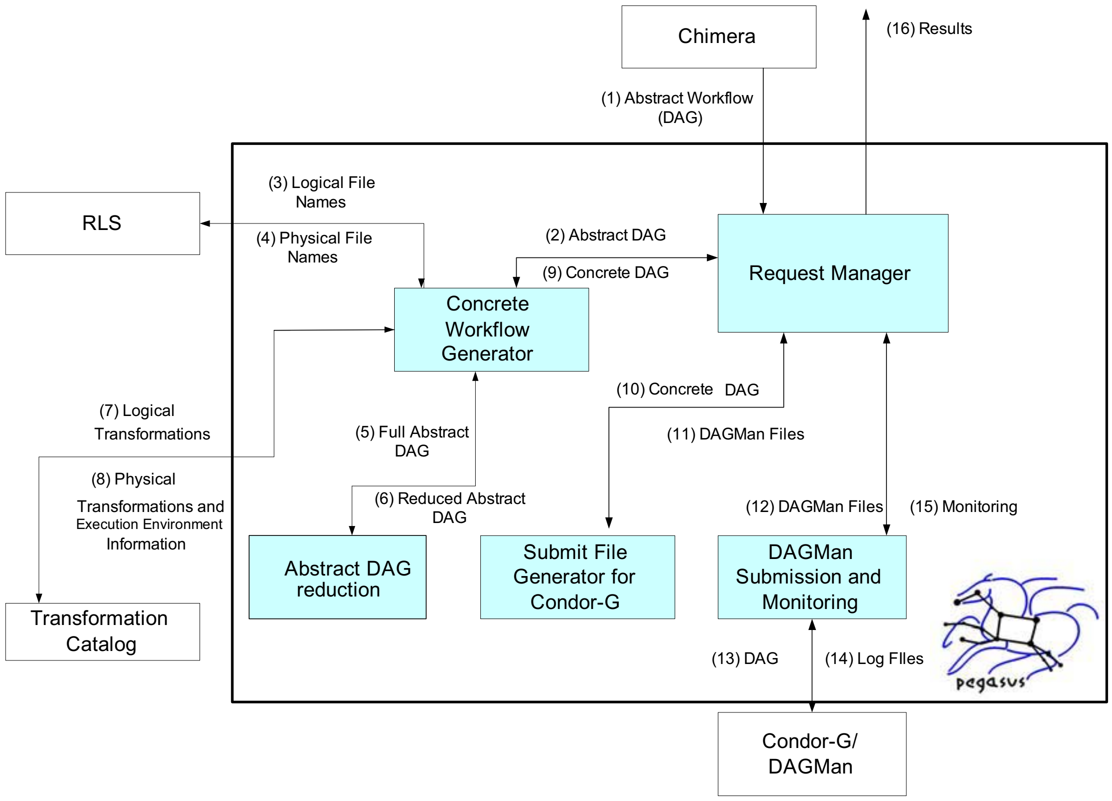

Figure 2: Chimera-driven Pegasus. Pegasus generates the concrete workflow.

Continuing with the example shown in Figure 2, if the intermediate file *b* exists at some location identified by the RLS, then the workflow will be reduced to that shown in Figure 3.

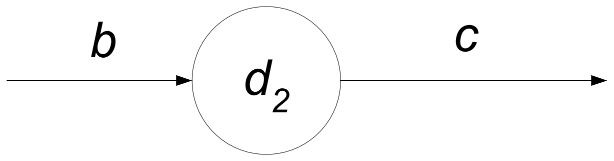

Figure 3: Reduced Abstract Workflow.

Next, the Concrete Workflow Generator component of Pegasus maps the remaining abstract workflow onto the available resources. Currently the information about the available resources is statically configured. In the near future, we plan to include dynamic information provided by Globus Monitoring and Discovery Service (MDS) [Czajkowski 2001].

Pegasus also checks for the feasibility of the abstract workflow. It determines the root nodes for the abstract workflow and queries the RLS for the existence of the input files for these components. The workflow can only be executed if the input files for these components can be found to exist somewhere in the Grid and are accessible via a data transport protocol.

The transformations in the abstract workflow are only identified by their logical name. In order to be able to find the actual executables, we have developed a Transformation Catalog [Deelman 2001]. The Transformation Catalog performs the mapping between a logical component name and the location of the corresponding executables on specific compute resources. The Transformation Catalog can also be used to annotate the components with the creation information. Pegasus queries the catalog to determine if the components are available in the execution environment and to identify their locations. Currently, the Concrete Workflow Generator picks a random location to execute from among the returned locations.

Transfer nodes are added for any input files that need to be staged in, so that each component and its input files are at the same physical location. If the input files are replicated at several locations, Pegasus currently picks the source location at random.

Finally, transfer nodes and registration nodes, which publish the resulting data products in the RLS, are added if the user requested that all the data be published and sent to a particular storage location.

The concrete workflow for the example in Figure 3 is shown in Figure 4. The workflow now specifies the resources to be used, performs the data movement, stages the data in and out of the computation, delivers it to the user-specified location *U* and registers the newly created data product in the RLS.

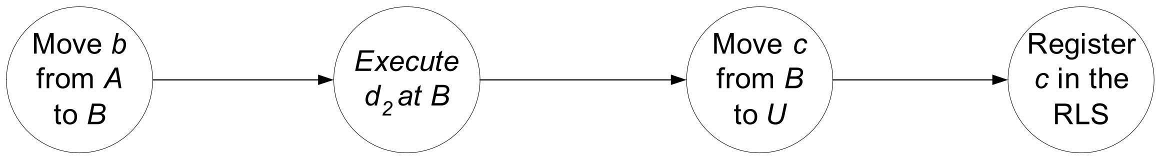

Figure 4: Concrete, Executable Workflow.

In order to be able to execute the workflow, Pegasus' Submit File Generator generates submit files which are given to Condor-G and the associated DAGMan for execution. These files contain the actual commands used to execute the workflow as well as the path for the executables and data.

3.3 Related Technologies

There have been a number of efforts within the Grid community to develop general-purpose workflow management solutions. While it was not intended that our prototype compare the applicability or effectiveness these various solutions, it is worth identifying them and commenting on how they differ from Chimera and Pegasus in their approach.

WebFlow [Fox 1998] is a multi-leveled system for high performance distributed computing. It consists of three layers. The top layer consists of a web-based tool for visual programming and monitoring. It provides the user the ability to compose new applications with existing components using a drag and drop capability. The middle layer consists of distributed web flow server implemented using java extensions to httpd servers. The lower layer uses the Java CoG Kit to interface with the Grid [Laszewski 2001] for high performance computing. WebFlow uses GRAM as the interface between itself and the Globus Toolkit. Thus, WebFlow also provides a visual programming aid for the Globus toolkit.

GridFlow [Cao 2003] has a two-tiered architecture with global Grid workflow management and local Grid sub workflow scheduling. GridAnt [gridant] uses the Ant [ant] workflow processing engine. Nimrod-G [Buyya 2000] is a cost and deadline based resource management and scheduling system. The Accelerated Strategic Computing Initiative Grid [Beiriger 2000] distributed resource manager includes a desktop submission tool, a workflow manager and a resource broker. In the ASCI Grid, software components are registered so that the user can ask "run code X" and the system finds out an appropriate resource to run the code. Pegasus uses a similar concept of virtual data where the user can ask "get Y" where Y is a data product and the system figures out how to compute Y. Almost all the systems mentioned above except GridFlow use the Globus Toolkit [globus] for resource discovery and job submission. The GridFlow project will apply the OGSA [ogsa] standards and protocols when their system becomes more mature. Both ASCI Grid and Nimrod-G uses the Globus MDS [Fitzgerald 1997] service for resource discovery and a similar interface is being developed for Pegasus. GridAnt, Nimrod-G and Pegasus use GRAM [Czajkowski 1998] for remote job submission and GSI [Welch 2003] for authentication purposes. GridAnt has predefined tasks for authentication, file transfer and job execution, while reusing the XML-based workflow specification implicitly included in ant, which also makes it possible to describe parallel and sequential executions.

The ASCI Grid is the only system which uses Kerberos. In GridAnt the user specifies the remote resource on which to submit the job. In ASCI Grid the system tries to schedule the job on the least loaded resource. The main emphasis of Nimrod-G is deadline and cost based scheduling. Pegasus can work with any local scheduling system such as condor [Litzkow 1988] or pbs [pbs] for which a GRAM jobmanager interface is available. The scheduling of jobs within a condor pool is left to the condor matchmaking system. The GridFlow system does task scheduling using a performance prediction-based mechanism.

The main difference between Pegasus and the above systems is that while most of the above system focus on resource brokerage and scheduling strategies Pegasus uses the concept of virtual data and provenance to generate and reduce the workflow based on data products which have already been computed earlier. It prunes the workflow based on the assumption that it is always more costly to compute the data product than to fetch it from an existing location. Pegasus also automates the job of replica selection so that the user does not have to specify the location of the input data files.

4 The End-to-End System

Another important purpose of this prototype is to gain experience using the core components described above, having them interact with each other, and evaluating them in a representative Grid. In this prototype we focused on the major software components which support data access, computation and a friendly user interface. In the future, we plan to expand the capabilities to dynamic resource discovery. Furthermore, we were less concerned with evaluating all the applicable technologies, rather choosing those that provided the necessary functionality.

4.1 Data Interfaces

As described in section 3.1, we have two standard interfaces for searching and retrieving data. Data repositories providing data for our application had to implement either or both. Table 1 outlines which datasets were involved in the demonstration:

Table 1: Data and Interfaces used by the Galaxy Morphology Application.

Data Center	Data Collection	Interface used
Chandra X-ray Center	Chandra Data Archive	SIA
NASA High-Energy Astrophysical Science Archive (HEASARC)	ROSAT X-ray data	SIA
NASA Infrared Processing and Analysis Center (IPAC)	NASA Extragalactic Database (NED)	Cone Search
Canadian Astrophysical Data Center (CADC)	Canadian Network for Cosmology (CNOC) Survey	SIA Cone Search
Multimission Archive at Space Telescope (MAST)	Digitized Sky Survey (DSS)	SIA Cone Search

4.2 The User Portal

To facilitate user access, a web portal was constructed as a high-level user interface to the Galaxy Morphology Grid analysis. The portal was also used to manage the integration of multiple NVO services for carrying out the procedure outlined in Section 2 and returning the results to the user for visualization. This framework provides an efficient means of integrating multiple remote and disparate systems, facilitated by the VOTable uniform transport format.

The general information flow of the portal operation is illustrated in Figure 5. The portal first allows a user to select from a list of galaxy clusters. For the demonstration, we restrict the clusters to those for which we know all the necessary data exist and are accessible through the appropriate standard interfaces (see Section 3.1). Selection of a galaxy cluster causes the portal to look up the cluster's spherical position in an internal catalog. With that position, the portal searches three image archives, one containing optical images (DSS) and two others containing x-ray images (ROSAT, Chandra), for images of the large-scale structure using the SIA interface. Links to these images are returned to the user. The user can then request to begin analysis. This triggers the construction of a catalog of the galaxies in the cluster; this is done by retrieving records from catalogs from two other data centers implementing the Cone Search interface. Next, references to the galaxy "cutout" images are requested, again using the SIA interface, and the descriptions and URLs pointing to the cutouts are merged into the catalog. The combined catalog is then sent to a web service designed to carry out the calculations on the Grid (see section 4.2). Also passed (for reasons explained below) is the desired output name of the table returned by the service. This output table contains the calculated values which the portal merges into the galaxy catalog.

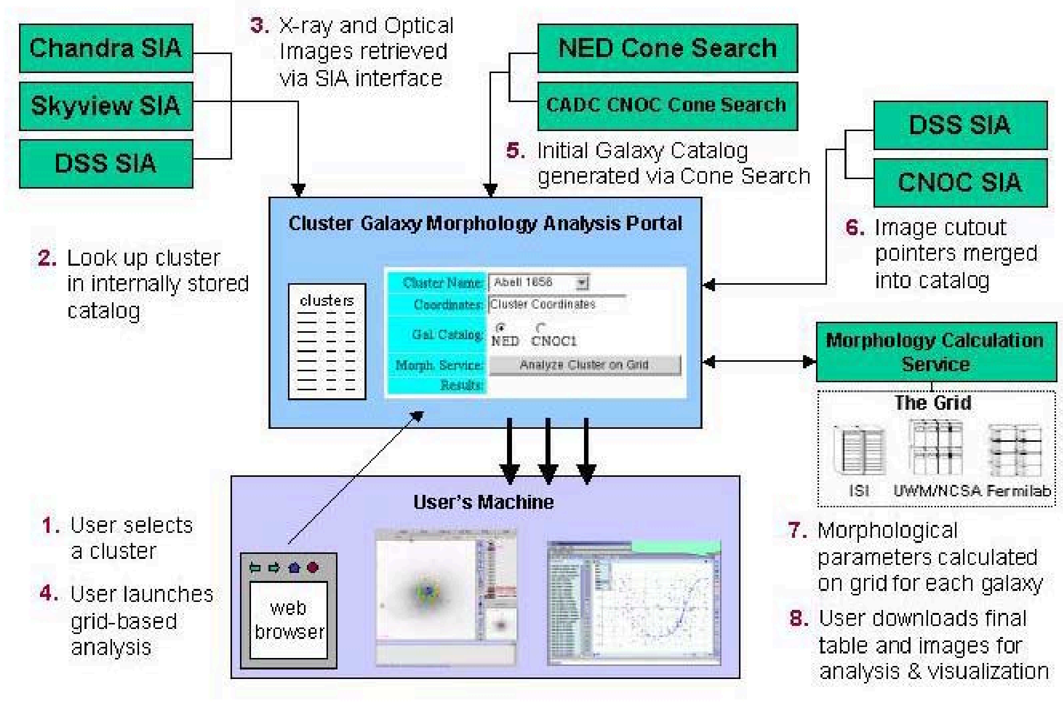
Chandra SIA
Skyview SIA
DSS SIA
3. X-ray and Optical Images retrieved via SIA interface
NED Cone Search
CADC CNOC Cone Search
5. Initial Galaxy Catalog generated via Cone Search
DSS SIA
CNOC SIA
6. Image cutout pointers merged into catalog
Cluster Galaxy Morphology Analysis Portal
2. Look up cluster in internally stored catalog
clusters
Cluster Name: Abell 1656
Coordinates: Cluster Coordinates
Gal. Catalog: NED CNOC1
Morph. Service: Analyze Cluster on Grid
Results:
Morphology Calculation Service
The Grid
ISI
UWM/NCSA
Fermilab
User's Machine
1. User selects a cluster
4. User launches grid-based analysis
web browser
7. Morphological parameters calculated on grid for each galaxy
8. User downloads final table and images for analysis & visualization

Figure 5: A schematic of the portal operation

The portal was implemented at the Space Telescope Science Institute (STScI) in Baltimore and is accessible from a web browser via a server there. The actual analysis of the galaxy morphologies, on the other hand, is managed via a web service hosted at ISI. The portal was implemented in .NET, a technology that provides an easy portal development environment. Integration of the morphology analysis web service into the portal was trivial and clearly demonstrating the value of high level application building tools provided by web services toolkits. From the WSDL, .NET provided automatic generation of the corresponding web client (C#) components. The inclusion of the service into the portal involved adding the client class to the project and then writing these two lines of code which passed in the input VOTable, *vot,* and returned *url* where the output VOTable (called *outVOTName)* containing morphological parameters was posted:

```
NVOGalMorphCompute comp = new NVOGalMorphCompute();
string url = comp.galMorphCompute(vot,outVOTName);
```

The portal operates in real-time with the multiple NVO services, waiting until all processing is done before returning the results page to the user. This synchronous behavior demonstrates a limitation of the portal as this processing can take up to a few hours; clearly an asynchronous response would be helpful. To facilitate ease of demonstration, the user has the ability to use cached results of the image search.

The major bottleneck in the application's operation is the querying of image servers (on the portal side) and collecting of the image cutouts (on the compute service side). This is due to some inherent inefficiencies in the SIA protocol: an image query and download for each galaxy must be done separately. This could be sped up tremendously if one could query for all images at once.

In keeping with the prototype nature of the application, there were a few capabilities that we did not include in the portal implementation that would be important for a production version. Two, in particular, are important for the NVO in general. The first is the ability to join VOTables in a general way. Joining is one of a few general-purpose VOTable manipulations that should be implemented as a generic, external service that could be used by a number of different NVO applications. In lieu of such a service, our portal combines data from different VOTables in a simple way using a local software library it calls internally. Another important capability we did not implement is a general registry of image and catalog services. Having such a registry would allow the user to discover and choose the appropriate data resources rather than being limited to the ones that were hard-coded into the portal. This capability could lead to a richer set of scientific results; for example, galaxy images from different frequency bands could yield different results. Obviously, providing this flexibility would require a higher level of fault tolerance and recovery.

4.3 Pegasus as a Web service

The Galaxy morphological analysis web service represents the type of highly-specialized service that we expect to see when the NVO environment reaches its most mature state. The service takes as its input a VOTable containing the collected data of all the galaxies in the cluster including the URLs of the image files of the galaxies. The functionality of the web service is to fetch the image files for all the galaxies, compute the parameters of all the galaxies, and finally concatenate all the results into an output VOTable. Carrying out these first two steps requires making use of information from the VOTable. XSLT [xslt], an XML standard for stylesheets that transform XML data into other formats, is a convenient and effective tool for feeding XML data into heterogeneous software systems [Plante 2002]. In the case of this particular web service, we used two stylesheets to process the input VOTable: the first simply created a URL list for loading the images into the RLS, and a second stylesheet converted the catalog directly into a derivation file containing the Virtual Data Language markup that described the necessary processing.

Upon receiving the input VOTable, the Pegasus web service immediately returns a URL where the status of the computation is published. The web service creates a unique identifier for each request which is included as a part of the returned URL. This identifier is used to match requests with replies. The portal polls the returned URL until it finds a "job completed" status message accompanied by a URL pointing to the location of the

VOTable containing the computed results. Each request from the portal also contains the name of the galaxy cluster for which the computations are being done. The computed VOTable is logically named after the galaxy cluster for which the computation is being performed.

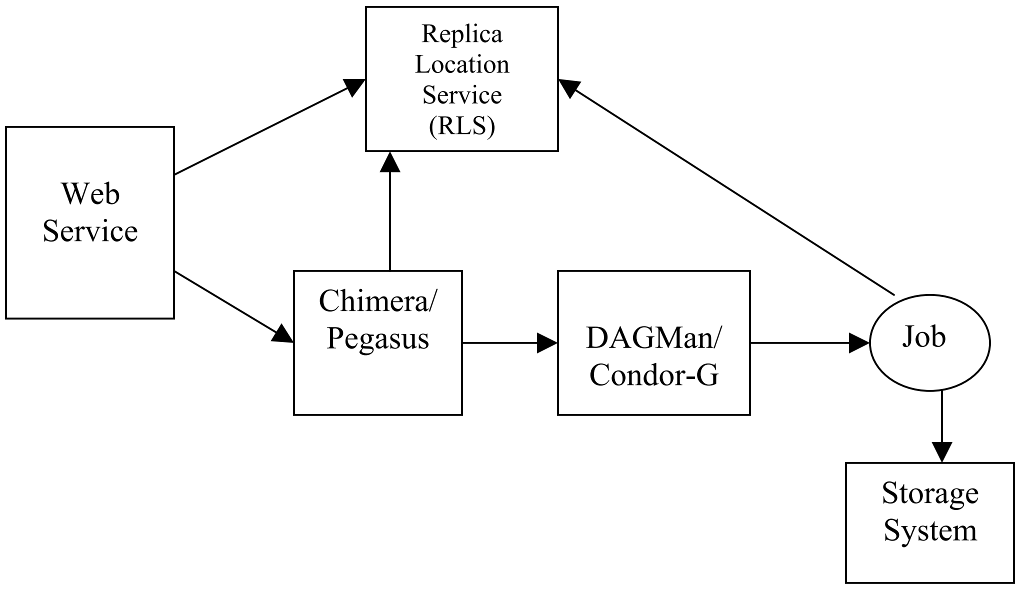

Figure 6: Design of the Web Service.

The web service, seen in Figure 6 can be described as follows:

1. The web service receives a request from the portal with an input VOTable and the name of the galaxy cluster. The web service assigns a unique identifier for the request. The service returns a URL to the portal where all the messages regarding the status of the computation are posted.
2. The service queries the RLS for the output VOTable. If the RLS contains the mapping for the output VOTable, its physical location is published on the URL returned to the portal, and the request is completed. If the VOTable is not found the computation proceeds.
3. The input VOTable is transformed into a simple list of URLs that point to the locations of the image files for each galaxy. Each image file is downloaded via its URL into a local directory and registered in the RLS. Thus the web service creates a local cache of image files. This feature is useful for avoiding the performance bottleneck of retrieving the data from the image servers via SIA should they be requested a second time.
4. In case the output VOTable is not registered in the RLS, the service transforms the input VOTable into the Chimera derivation file. A Chimera transformation file is also created; this file need only be created once the first time the service is called as the function prototypes remain the same.

5. Chimera creates an abstract workflow for computing the output VOTable.
6. Pegasus creates a concrete workflow based on the locations of the executables and image files. It adds nodes for registering the computed parameters for each galaxy and the final output VOTable. Pegasus also reduces the concrete workflow based on pre-computed data products. The reduced workflow is submitted to DAGMan/Condor-G for execution.
7. Whenever the portal polls the URL returned to it by the web service, a java servlet queries the RLS for the final VOTable associated with that request. This mapping would be present if the concrete workflow submitted to condor in the previous step has run to completion. The last job in this workflow is the registration of the final VOTable in the RLS.

4.3.1 Design Issues

While developing the galaxy morphology service, we needed to consider the following issues:

1. *Web Service vs. Web Application*: One of the first issues considered was whether to structure the service as a web service with a WSDL description or a web application using HTTP get/post to initiate the computations. The general advantage of creating a web service instead of a web application is the ability to formally describe and publish the service using WSDL and discover it via a registry service (e.g. UDDI). For us, the important advantage was the platform and programming language independence. It allowed the Grid-based service and demonstration portal to be developed at different sites. The automated web service tools provided by the Apache Axis (used at the service side) and the .Net platform (on the portal side) made interfacing the two components straightforward. Finally, as Grid services [ogsa] mature, the transition from a web service will be easier.
2. *Asynchronous/Synchronous interface*: The next design decision was whether to use synchronous or asynchronous operations in the web service. In the synchronous mode, the client blocks until the computation is completed and the output VOTable is returned to the client. In the asynchronous mode, the service immediately returns an URL to the client and the client keeps polling the URL for status messages. We decided to use an asynchronous interface because the computations can take a long time to get executed for bigger clusters. It also provides a mean for the portal to get intermediate status messages.
3. *Data caching*: We decided to cache the galaxy image files in the web server and register them in the RLS. This allows the service to be used even when the image services like MAST and CADC are down. Additionally, the data is then available via GridFTP [Allcock 2001], which provides much better performance than the SIA.
4. *Fault tolerance*: Often, the computation for calculating parameters of individual galaxies would fail because of the bad quality of galaxy images or some other reasons. One option was not to let the portal time out after some time and resubmit the request. The other option, which we implemented, was recover from the failed computations during the final concatenation of results: we added a

validity flag to the set of returned values to indicate whether the computations for a given galaxy completed successfully. Thus, this prevented a few failures from taking down the entire experiment.

5. *Authentication*: This prototype web service submits jobs onto the Grid using the credentials stored at the web server. However, for a more general solution, we are planning to use MyProxy [Novotny 2001] as a solution for authentication of users.

4.4 Analysis tools used to view the results

We used existing software to analyze the data. Again, because the results were returned as a XML-based VOTable, it was very easy to import the data into existing tools. One example, shown in Figure 7, depicts a Java application from the Strasbourg Data Center called Aladin, capable of visualizing image and catalog data together. Because it is open-source, it was straightforward to plug in a VOTable parser. We also made use of another visualization tool from IBM called Mirage [mirage] which can create various plots of tabular data; this tool allowed to use scatter plots to look for correlations between our morphology parameters and other galaxy characteristics returned in the VOTable. We were able to support Mirage by creating an XSL stylesheet that transformed the VOTable into the tool's native format.

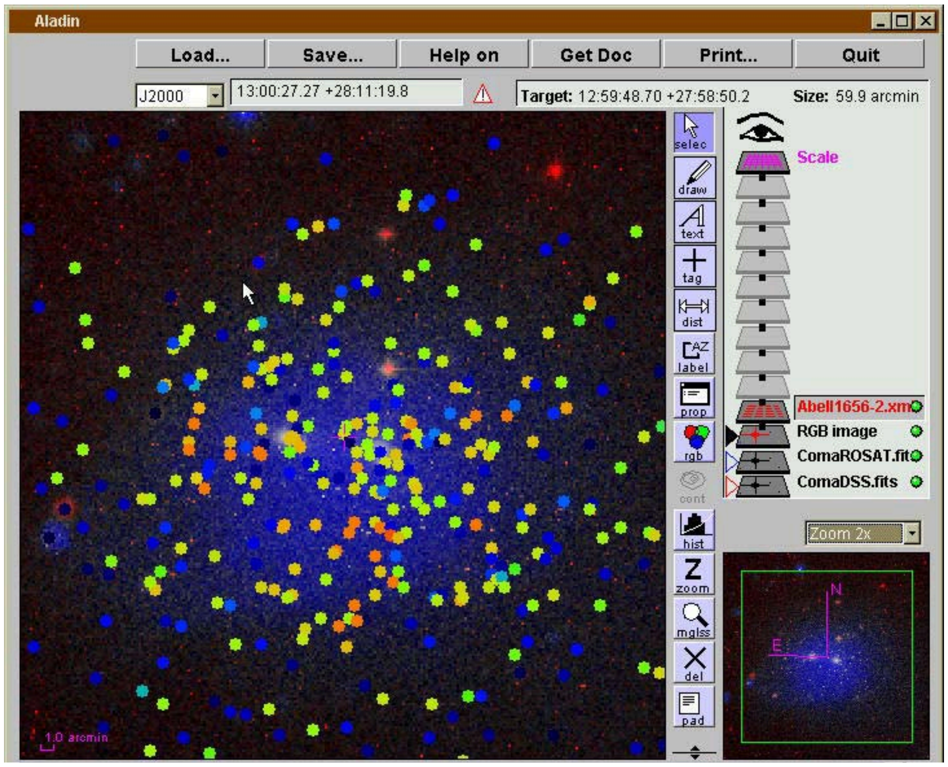

Figure 7: Aladin image and catalog viewer. The x-ray emission is shown in blue, and the optical mission is in red. The colored dots are located at the positions of the galaxies within the cluster; the dot color represents the value of the asymmetry index. Blue dots represent the most asymmetric galaxies (i.e. spiral galaxies) and are scattered throughout the image, while orange are the most symmetric, indicative of elliptical galaxies, are concentrated more toward the center.

5 Results and Conclusions

We used our prototype to separately analyze eight different galaxy clusters. The number of galaxies processed for each cluster ranged from 37 to 561. To carry out the computations, we used three Condor pools, one each at University of Southern California, University of Wisconsin, and Fermilab. During the analysis phases, there were a total of 1152 compute jobs executed. The computations were performed on a total of 1525 images, corresponding to 30MB of data. Staging the data in and out of the computations involved the transfer of 2295 files.

Analysis of our results (illustrated in Figure 7) indicates that we have "rediscovered" the Dressler density-morphology relation which showed that elliptical galaxies are concentrated more towards a cluster's center. We note that Dressler's method (carried out carefully "by hand") was different from ours, pointing out the value of the Grid for applying new analysis techniques on existing data. This demonstration can serve as a model for how generalized computational services could be exposed in a Data Grid for general experimentation.

Much of our motivation for building this prototype application focused on understanding the technical issues in building a Grid-based application. Collecting heterogeneous, distributed datasets to perform complex analysis typifies an advance type of application we expect to become commonplace in the NVO. We believe that the successful deployment and use of various technologies lays a foundation for the development of a general NVO application development framework. We identified the important components and necessary protocols and interfaces. We also identified the needed capabilities yet to be developed.

In particular, domain-specific interfaces allowed us to access the image and catalog data simply in uniform ways. Standard formats not only made it easier to exchange data between parts of the system, it allowed us to leverage existing software for visualization. Our XML format, VOTable, proved particularly flexible for data exchange when we made use of XSLT to convert the data to other formats. Chimera and Pegasus provided the core infrastructure for managing the data and computations within the Grid environment. The ability built into Pegasus to reuse previously calculated data was particularly useful. Our prototype was also successful in connecting Web Services and clients developed in different languages.

This effort shed light on several items of infrastructure that we still need to address when laying the real foundations of the NVO. Most obvious is the need for a registry of data and service resources. This would allow users to discover the relevant data and tools necessary for the study. In particular, different choices of images to be analyzed or computation algorithms to be employed could reveal different things about the dynamical state of the galaxy clusters. We also discovered the general utility of a service that could join two VOTables on an arbitrary column or manipulate tables in other ways.

Ultimately, with an environment with a rich set of data sources and services as well as robust tools for connecting them together, the astronomer would have great flexibility in harnessing the large and diverse datasets now available for on-line research.

Finally, we have gained insight into how distributed, parallel processing might be exposed as a generic service within a Data Grid. Chimera and Pegasus can provide the glue between input data, a specific set of processing modules, and general-purpose computing nodes and can ensure efficient use of those resources. The key to flexible, "on-the-fly" use of such a service is generating the derivation descriptions in VDL automatically, in part, from the descriptions of the input datasets. With this level of automation, we can envision a user using a portal to select data and algorithms and connect them in a new way, all without necessitating new programming.

Acknowledgments

We would like to thank Gregor von Laszewski for his helpful comments and suggestions on this version of the paper. This work was supported by the National Science Foundation's Information Technology Research Program under Cooperative Agreement AST0122449 with The Johns Hopkins University and by the NSF ITR-0086044 (GriPhyN) Grant. We would also like to acknowledge the participation of our other collaborators in the NVO project that provided the data used in our demonstration, including the Canadian Astronomy Data Center, the Chandra Science Archive at the Smithsonian Astrophysical Observatory, the NASA High Energy Science Archive, and the NASA Infrared Processing and Analysis Center.

References

[Allcock 2001] W. Allcock, J. Bester, et al., "Secure, Efficient Data Transport and Replica Management for High-Performance Data-Intensive Computing," Proceedings of the IEEE Mass Storage Conference, pp. 13-28 April 2001.

[ant] http://ant.apache.org

[Beiriger 2000] Beiriger, J., Johnson, W., Bivens, H., Humphreys, S. and Rhea, R., Constructing the ASCI Grid. In Proc. 9th IEEE Symposium on High Performance Distributed Computing, 2000, pp. 193-200. IEEE Press.

[Blythe 2003] Blythe, J., E. Deelman, Y. Gil, C. Kesselman. "Transparent Grid Computing: A Knowledge-Based Approach", In Proc. of the Conference on Innovative Applications of Artificial Intelligence (IAAI) 2003. (*to appear*)

[Blythe 2003b] Blythe, J., E. Deelman, Y. Gil, C. Kesselman, A. Agarwal, G. Mehta, K. Vahi, "The Role of Planning in Grid Computing" in Proc. International Conference on Automated Planning and Scheduling (ICAPS), p. 154-163 2003.

[Blythe 2003c] Blythe, J., E. Deelman, Y. Gil. "Planning for workflow construction and maintenance on the grid" in ICAPS 03 Workshop on Web Services Composition, pp. 8 – 14, 2003.

[Brunner 2001] R. J. Brunner, S. G. Djorgovski, A. S. Szalay, eds., "Virtual Observatories of the Future," Astronomical Society of the Pacific Conference Series Vol 225, 2001.

[Buyya 2000] Buyya, R., Abramson, D. and Giddy, J. "Nimrod/G: An Architecture of a Resource Management and Scheduling System in a Global Computational Grid", HPC Asia 2000, May 14-17, 2000, pp 283 - 289, Beijing, China.

[Cao 2003] J. Cao, S. A. Jarvis, S. Saini, and G. R. Nudd. GridFlow: WorkFlow Management for Grid Computing. In Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'03), pp. 198-205, 2003.

[Chervenak 2002] A. Chervenak, E. Deelman, et al., "Giggle: A Framework for Constructing Scalable Replica Location Services," Proceedings of Supercomputing 2002 (SC2002), Baltimore, MD. 2002.

[cone] www.us-vo.org/metadata/conesearch

[Conselice 2003] C. J. Conselice, "The Relationship between Stellar Light Distributions of Galaxies and Their Formation Histories", Astrophysical Journal Supplement Series, vol. 147, pages 1-28, 2003.

[Czajkowski 1998] K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W. Smith, S. Tuecke. A Resource Management Architecture for Metacomputing Systems. Proc. IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing, pg. 62-82, 1998.

[Czajkowski 2001] K. Czajkowski, S. Fitzgerald, et al., "Grid Information Services for Distributed Resource Sharing," Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing, pages: 181 -194, 2001.

[Deelman 2001] E. Deelman, C. Kesselman, et al., "Transformation Catalog Design for GriPhyN," Technical Report GriPhyN-2001-17, 2001.

[Deelman 2003] E. Deelman, J. Blythe, et al., "Mapping Abstract Complex Workflows onto Grid Environments," Journal of Grid Computing, vol 1, nr 1, pages 25-29, 2003.

[Deelman 2003b] E. Deelman, J. Blythe, Y. Gil, Carl Kesselman "Workflow Management in GriPhyN", Chapter in "The Grid Resource Management", to appear 2003.

[Dressler 1980] A. Dressler, "Galaxy Morphology in Rich Clusters-Implications for the Formation and Evolution of Galaxies", Astrophysical Journal, vol. 236, p. 351, 1980.

[Ellingson 2003] E. Ellingson, "Galaxy Evolution in Clusters", Astrophysics and Space Science, vol. 285, issue 1, pages 9-18.

[Fitzgerald 1997] S. Fitzgerald, I. Foster, C. Kesselman, G. von Laszewski, W. Smith, S. Tuecke. A Directory Service for Configuring High-Performance Distributed Computations. Proc. 6th IEEE Symposium on High-Performance Distributed Computing, pp. 365-375, 1997.

[Foster 2002] I. Foster, J. Voeckler, et al., "Chimera: A Virtual Data System for Representing, Querying, and Automating Data Derivation," Proceedings of the Scientific and Statistical Database Management Conference, Edinburgh, Scotland, pp. 37-46, July 2002.

[Fox 1998] http://www.supercomp.org/sc98/TechPapers/sc98_FullAbstracts/Akarsu809

[Frey 2001] J. Frey, T. Tannenbaum, et al., "Condor-G: A Computation Management Agent for Multi-Institutional Grids," Proceedings of 10th International Symposium on High Performance Distributed Computing, pages 55-63, 7-9 Aug. 2001

[Hanisch 2000] R. J. Hanisch, “Distributed Data Systems and Services for Astronomy and the Space Sciences”, Astronomical Data Analysis Software and Systems IX, ASP Conf. Ser., vol. 216, eds. N. Manset, C. Veillet, & D. Crabtree (San Francisco:ASP), p. 201.

[Hanisch 2001a] R. J. Hanisch, "Building the Infrastructure for the Virtual Observatory," Proceedings of the SPIE, vol. 4477, 191, 2001.

[Hanisch 2001b] R. J. Hanisch, A. Farris, E. W. Greisen, W. D. Pence, B. M. Schlesinger, P. J. Teuben, R. W. Thompson, and A. Warnock III, "Definition of the Flexible Image Transport System (FITS)", Astronomy and Astrophysics, vol. 376, p. 359, 2001.

[globus] http://www.globus.org

[gridant] http://www-unix.globus.org/cog/projects/gridant

[griphyn] www.griphyn.org

[Laszewski 2001] von Laszewski, G. I. Foster, J. Gawor, P. Lane, “A Java Commodity Grid Kit,” *Concurrency and Computation: Practice and Experience*, pages 643-662, Volume 13, Issue 8-9, 2001.

[Litzkow 1988] M. Litzkow, M. Livny, and M. Mukta, "Condor - A Hunter of Idle Workstations", Proceedings of the 8th International Conference of Distributed Computing Systems, pages 104-111, June, 1988.

[mirage] www.bell-labs.com/project/mirage/

[Novotny 2001] Novotny, J., S. Tuecke, and V. Welch, "An Online Credential Repository for the Grid: MyProxy," Proc. of the 10th IEEE International Symposium on High Performance Distributed Computing, pp. 104 -111, 2001.

[ogsa] www.globus.org/ogsa

[pbs] http://www.openpbs.org

[Plante 2002] R. L. Plante, D. Guillaume, D. Mehringer, and R. M. Crutcher, “Metadata-driven Processing in the BIMA Image Library”, Astronomical Data Analysis Software and Systems XI, ASP Conf. Ser., vol. 281, eds. D. A. Bohlender, D. Durand & T. H. Handley (San Francisco:ASP), p. 346.

[sia] www.us-vo.org/news/simspec.html

[us-vo] us-vo.org

[VOTable] us-vo.org/VOTable

[Welch 2003] V. Welch, F. Siebenlist, I. Foster, J. Bresnahan, K. Cajkowski, J. Gawor, C. Kesselman, S. Meder, L. Pearlman, S. Tuecke. Security for Grid Services. Twelfth International Symposium on High Performance Distributed Computing (HPDC-12), pp. 48 -57, June 2003, IEEE Press.

[xslt] www.w3c.org/TR/xslt