Fermilab FERMILAB-Conf-03/027 March 2003

Uniformity on the Grid via a Configuration Framework

A. Baranovski, L. Lueking, G. Garzoglio, I. Terekhov

March 11, 2003

Abstract

As Grid permeates modern computing, Grid solutions continue to emerge and take shape. The actual Grid development projects continue to provide higher-level services that evolve in functionality and operate with application-level concepts which are often specific to the virtual organizations that use them. Physically, however, grids are comprised of sites whose resources are diverse and seldom project readily onto a grid's set of concepts. In practice, this also creates problems for site administrators who actually instantiate grid services. In this paper, we present a flexible, uniform framework to configure a grid site and its facilities, and otherwise describe the resources and services it offers. We start from a site configuration and instantiate services for resource advertisement, monitoring and data handling; we also apply our framework to hosting environment creation. We use our ideas in the Information Management part of the SAM-Grid project, a grid system which will deliver petabyte-scale data to the hundreds of users. Our users are High Energy Physics experimenters who are scattered worldwide across dozens oF institutions and always use facilities that are shared with other experiments as well as other grids. Our implementation represents information in the XML format and includes tools written in XQuery and XSLT.

1 Introduction

Grid [1] has emerged as a modern trend in computing, aiming to support the sharing and coordinated use of diverse resources by Virtual Organizations (VO's) in order to solve their common problems[2]. It was originally driven by by scientific, and especially High-Energy Physics (HEP) communities. HEP experiments are a classic example of large, globally distributed VO's whose participants are scattered over many institutions and collaborate on studies of experimental data, primarily on data processing and analysis.

Our background is specifically in the development of large-scale, globally distributed systems for HEP experiments. We apply grid technologies to our systems and develop higher-level, community-specific grid services (generally defined in [3]), currently for the two collider experiments at Fermilab, D0 and CDF. These two experiments are actually the largest currently running HEP experiments, each having over half a thousand users and planning to analyze repeatedly peta-byte scale data. In recent years, we have been working on the SAMGrid project [4], which addresses the grid needs of the experiments; our current focus is in the Jobs and Information Management (JIM), which is to complement the SAM grid data handling system [5] with services for job sub-

mission, brokering and execution as well as distributed monitoring. The grid research issues we discuss in this paper have been observed in the HEP realm which remains a typical and principal Grid domain.

It is well known that the computing and other resources shared by scientific collaborations are diverse, distributively owned and are rarely available all at the same time . We further note that the physical owners of resources are the real organizations (universities and HEP research centers) who seldom, if ever, partition the resources by the VO's in which their scientists collaborate. Moreover, the HEP scientists, while usually affiliated with a single real institution, are almost always collaborating in more than one HEP experiment. Consequently, physical resource providers need to share the resources among the various VO's, who choose different computing paradigms and policies and have legacy meta-computing systems. These VO's – HEP experiments – use various grid solutions for reasons of both technical and sociological nature. Obviously, these grid solutions carry the high-level, community-specific concepts, which translates into different views onto the resources and services and the way they are managed within the community. These differences among the grid communities are most prominent at the application level, but exist even at the middleware realm, because even with standards in middleware there will still be multiple software implementations. [1]

[1] In fact, we believe that a standard is a true success when it has more than one implementation!

2 The Grid Instantiation and Configuration Problem

For a site to join a grid, it's administrator must perform what we call *grid instantiation*. It entails mapping of the site's abilities as a grid player onto the design of that grid, and then actually instantiating that grid's services.

Each instantiation typically proceeds in a way that is specific to the grid at hand, which is already a challenge for the administrators who naturally think in terms of the physical resources they own or manage, e.g. computing clusters, LAN interconnections, their favorite storage systems, batch systems, etc.. The problem is aggravated by the aforementioned multiplicity of grid solutions; it re-surfaces every time when re-instantiation is necessary. Grid re-instantiation may be mandated by either grid software design evolution (which may change the cardinality or type of physically running servers) or physical site reconfiguration such as reassignment of disks among network-attached servers, firewall settings change, or addition of a new generation storage system. Note that we are talking not in terms of routine version upgrades (a challenge in itself), but rather *structural* changes in the grid software instance.

Further in the course of a grid instantiation, a site administrator as a physical resource provider needs to describe the existing resources to the grid at hand, in a way that is again specific to the grid. In the most general case, a resource is anything that has identity [6] and may include everything that is configured at the site, e.g. a rather uniquely configured cluster with heterogenous archi-

tecture or the location of a hole in the firewall provided for well-defined use or the acceptable use policies text file. In real life, the administrator wants to be able to describe selectively these resources to the grid at hand, in a language that's not specific to that grid, whether or not that grid can take advantage of these particular resource today. The Globus MDS-2 [7] software and the Grid Monitoring Architecture [8] do allow extension of the design so it is theoretically feasible to map a site configuration onto the community's concepts, yet we are not aware of any consistent, uniform way to do so.

Fundamentally, we see the problem in the lack of a coherent framework to define a grid site configuration and then use it to publish selected resources to a grid, instantiate the appropriate grid services, and otherwise enable the site for Grid computing, possibly in contexts of multiple grid communities.

In the next Section, we attempt such a framework. We present useful applications for Condor ClassAd framework, MDS-based grid activity monitoring infrastructure, the SAM grid data handling system, as well as generic physical hosting environment creation problem. Later, in Section 4 we discuss instantiation of services of multiple Grids at the same site and how this facilitates interoperability.

3 The Management of Configuration and Information in JIM

Our main proposal is that grid sites be configured using a site-oriented schema, and that grid instantiation at the sites be derived from these site configurations. We are not proposing any particular site schema at this time, although we hope for the Grid community as a whole to arrive at a common schema in the future which will allow reasonable variations such that various grids are still instantiatable.

Figure 1 shows configuration derivation in the course of instantiation of a grid at a site. The site configuration is created using a meta-configurator similar to one we propose below.

3.1 The Core Meta-Configurator and the Family of Configurators

In our framework, we create site and *all* other configurations by a universal tool which we call a *meta-configurator*, or configurator of configurators. The idea is to separate the process of querying the user for values of attributes from the *schema* that describes what those attributes are, how they should be queried, how to guess the default values, and how to derive values of attributes from those of other attributes. Any concrete configurator uses a concrete schema to ask the relevant questions to the end user (site administrator) in order to produce that site's configuration. Any particular schema is in turn derived from a meta-schema. Thus, the end configuration can be represented as:

$$C = c(S_d, I_u) = c(c(S_0, I_d), I_u),$$

where C is a particular configuration, c is the configuration operation, S_d is a particular schema reflecting certain design, S_0 is the meta-schema, I_d and I_u are the inputs of the designer and the user, respectively.

In our framework, configurations and

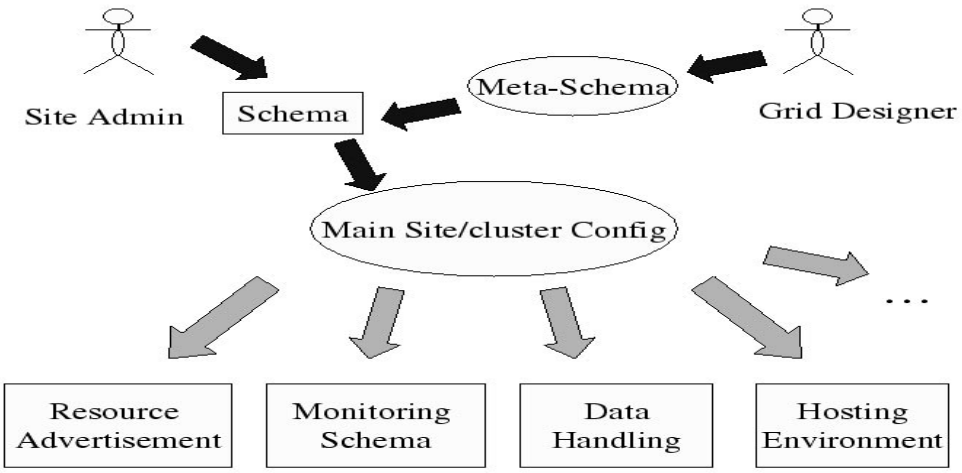

Figure 1: Configuration creation and derivation in our framework. Service collections are typical of the SAMGrid project.

schemas are structures of the same type, which we choose to be trees of nodes each containing a set of distinct attributes. Our choice has been influenced by the successes of the XML technologies and, naturally, we use XML for representing these objects.

To exemplify, assume that in our *present* design, a grid **site** consists of one or more **clusters** each having a name and an architecture (homogenous), as well as exactly one **gatekeeper** for Grid access. Example configuration is:

```
<?xml version='1.0'?>
<site name='FNAL'
      schema_version='v0_3'>
  <cluster name='samadams'
           architecture='Linux'>
    <gatekeeper ...>
  </cluster>
</site>
```

This configuration was produced by the following schema

```
<?xml version='1.0'?>
<site cardinalityMin='1'
      cardinalityMax='1'
      name='inquire-default,FNAL' >
   <cluster  cardinalityMin='1'
     name='set,CLUSTERNAME,inquire'
     architecture=
      'inquire-default,exec,uname'/>
</site>
```

in an interactive session with the site administrator as follows:

```
What is the name of the site ? [FNAL]:
 <return>
What is the name of cluster
  at the site 'FNAL'?  samadams
What is the architecture
  of cluster 'samadams' [Linux]?
```

...

When the schema changes or a new cluster is created at the site, the administrator merely needs to re-run the tool and answer the simple questions again.

3.2 Projection onto the Condor ClassAd Framework

In the JIM project, we have designed the grid job management as follows. We advertise the participating grid clusters to an information collector and grid jobs are matched [9] with clusters (resources) based on certain criteria primarily having to do with the data availability at the sites. We have implemented this job management using Condor-G [10] with extensions that we have designed together with the Condor team [4]. As far as the participating sites are concerned, we need to advertise the clusters together with the **gatekeepers** as the means for Condor to actually schedule and execute the grid job at the remote site. Thus, our *present* design requires that each advertisement contain a cluster, a gatekeeper, a SAM station (for jobs actually intending to process data) and a few other attributes that we omit here. Our advertisement software then selects from the configuration tree all patterns containing these attributes and then applies a ClassAd generation algorithm to each pattern.

The selection of the subtrees that are ClassAd candidates is based on the XQuery language. Our queries are generic enough as to allow for *design evolution*, i.e. to be resilient to some modifications in the schema. When new attributes are added to an element in the schema, or when the very structure of the tree changes due to insertion of a new element, our advertisement service will continue to advertise these clusters with or without the new information (depending on how the advertiser itself is configured) but the important factor is that this site will continue to be available to our grid.

For example, assume that one cluster at the site from subsection 3.1 now has a new grid gatekeeper mechanism from Globus Toolkit 3, in addition to the old one:

```
<?xml version='1.0'?>
<site name='FNAL'
      schema_version='v0_3'>
  <cluster name='samadams'
           architecture='Linux'>
    <grid_accesses>
       <gatekeeper ...>
       <gatekeeper-gtk3 ...>
    </grid_accesses>
    ...
```

Assume further that our particular grid is not yet capable of taking advantage of the new middleware and we continue to be interested in the old **gatekeeper** from each cluster. Our pattern was such that a **gatekeeper** is a descendant of the **cluster** so we continue to generate meaningful ClassAds and match jobs with this site's cluster(s).

3.3 Application to MDS Configuration

In addition to advertising (pushing) of resource information for the purpose of job matching, we deploy Globus MDS-2 for pull-based retrieval of information about the clusters and activities (jobs and more, such as data access requests) associated with them. This allows us to enable web-based monitoring, primarily by humans, for performance

and troubleshooting [4]. We introduce (or redefine in the context of our project) concepts of **cluster**, **station** etc, and map them onto the LDAP attributes in the OID space assigned to our project (the FNAL organization, to be exact) by the IANA[11]. We also create additional branches for the MDS information tree as to represent our concepts and their relations.

We derive the values of the *dn*'s on the information tree from the site configuration. In this framework, it is truly straightforward to use XSLT (or a straight XML-parsing library) to select the names and other attributes of the relevant pieces of configuration. For example, if the site has two clusters defined in the configuration file, our software will automatically instantiate two branches for the information tree. Note that the resulting tree may of course be distributed within the site as we decide e.g. to run an MDS server at each cluster, which is a separate degree of freedom, see Section 3.5.

3.4 Instantiation of the SAM Grid Data Handling System

For a system like SAMGrid where large-scale distributed datasets are managed and served to the applications, an important category of grid services has to do with the data handling[12]. The SAM system is an example production system that delivers petabyte scale data to the hundreds of our users scattered worldwide. Physically, it is implemented with a collection of servers of several types, whose "density" varies from several per machine to one per an entire VO. The exact arrangement of physical servers depends on the site configuration. For example, if there is a site firewall and we need to enable servers both inside and outside the firewall, a border naming service has to be started (The SAM software uses CORBA). Other examples include database server proxies [2] and, in the future, HTTP and FTP proxies.

Furthermore, this data handling system uses local storage element(s) for caching of data. To be installed at the site, the system needs to be communicated the locations and capacities of the storage elements, as well as the allocations of this local storage space to the VO(s) that use them via this data handling system.

The SAM services are first configured from the site configuration and then from additional specifications by the administrator, inquired by a SAM configurator.

3.5 Physical Server Instantiation

In the previous sections, we have described how some high-level services can be configured and instantiated logically based on the site configuration. With a few exceptions of stateless servers like the Globus Gatekeeper which can conveniently be started by the system daemon like **inetd**, most of the services in the SAMGrid project are physically created by means of native UNIX processes. There are about fifteen different types of server processes in the system, and we needed a tool to control these, including common actions taken upon failure such as detecting exit status, notifying the (local) administrators by email, saving core and/or diagnostic files, and automatically restart the server after a certain interval. These and other features are implemented in the **server_run** package, which uses XML con-

[2] Our HEP applications use a database to retrieve a collection of calibration data as parameters for data analysis.

figuration to create an arbitrary environment (by executing shells and sourcing specified scripts) and run arbitrary programs as child processes.

The SAMGrid project's services are therefore instantiated within a specific execution environment or *hosting environment* [13] by means of (XQuery and XSLT-based XML) transformations of the site configuration, with subsequent invocation of an (XML) configurator that queries the administrator for additional information, Section 3.1. For example, when the software products are installed using a tool such as UPS [3] the location of the products database is a parameter configured for the site's cluster, or tailored for this particular instance of **server_run**.

4 Multiple Grid Instantiation and Interoperability

We have been mentioning that there are in fact several other grid projects developing high-level Grid solutions; some of the most noteworthy include the European Datagrid [14], the Crossgrid[15], and The NorduGrid [16]. Interoperability of grids (or of solutions on The Grid if you prefer) is a well-recognized issue in the community. The High Energy and Nuclear Physics InterGrid [17] and Grid Interoperability [18] projects are some of the most prominent efforts in this area. As we have pointed out in the Introduction, we believe that interoperability must include *the ability to instantiate and maintain multiple grid service suites at sites.*

A good example of interoperability in this sense is given by various cooperating Web browsers which all understand the user's bookmarks, mail preferences etc.. Of course, each browser may give a different look and feel to its "bookmarks" menu, and otherwise treat them in entirely different ways, yet most browsers tend to save the bookmarks in the common HTML format, which has *de facto* become the standard for bookmarks. Our framework, proposed and described in Section 3, is a concrete means to facilitate this aspect of interoperability. Multiple grid solutions can be instantiated using a grid-neutral, site-oriented configuration in an XML-based format.

We can go one step further and envisage that the various grids instantiated at a site have additional, separate configuration spaces that can easily be conglomerated into a *grid instantiation database*. In practice, this will allow the administrators e.g., to list all the Globus gatekeepers with one simple query.

5 Status and Plans

One of our main strategies, for both the SAMGrid project in general and the configuration services in particular, is to embrace the Open Grid Services Architecture. We plan to investigate and participate in the development of the service for instantiation of other grid services. We believe that our proposed framework and the tools such as those described in Sections 3.1,3.5 form a prototype grid instantiation service. In practice, the OGSA services are likely to be described in WSDL[19], which is based on the XML technology – the same technology that our tools use – which should facilitate the development.

[3] A UNIX product management system from FNAL, similar to RPM.

We plan to bring true database semantics to the site configurations. As of the time of writing this paper, we manipulate with XML files directly. We are beginning to explore XML databases for configuration manipulation. (Incidentally, the database servers are themselves services instantiated using our framework so we'll face the famous poultry problem at some point). These will also deliver the highly desirable feature of querying the site configuration databases remotely, with possibilities for resource management and monitoring design based on distributed XQuery mechanisms. Obviously, security is an immediate challenge so we will be researching into GSI-enabled XML databases.

6 Summary

We have proposed a uniform and flexible framework for describing a site to the Grid and instantiate grid services from such a site description. Our framework uses XML configuration databases and generic configuration tools. We have used this framework in the SAMGrid project to configure our grid sites and instantiate at them, logically and physically, the actual services comprising the SAMGrid architecture. Such a common framework, if adopted by other grid project, would allow for interoperability of the various grids in terms of co-existence of services. In the future, we plan to instantiate actual OGSA services and develop a distributed database semantics for our configuration files.

7 Acknowledgements

This work has been sponsored by DOE contract No. DE-AC02-76CH03000, additional funding comes through the DOE Particle Physics Data Grid (PPDG) [20] Collaboratory SciDAC project. We thank Ruth Pordes of FNAL Computing Division, also a PPDG executive, for stimulating discussions concerning interoperability on the Grid. We have benefitted from ideas and work of Sinisa Veseli and other members of the SAM team and FNAL Computing Division.

References

[1] I. Foster and C. Kesselman (eds.) *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufman, 1999.

[2] I. Foster, C. Kesselman, S. Tuecke. The Anatomy of the Grid: Enabling Scalable Virtual Organizations, *International Journal of High Performance Computing Applications*, 15(3), 200-222, 2001.

[3] I. Foster, C. Kesselman, J. Nick, S. Tuecke, Grid Services for Distributed System Integration, *IEEE Computer* 35(6), 2002.

[4] G. Garzoglio, "The SAM-GRID Project: Architecture and Plan", in Proceedings of *The VIII International Workshop on Advanced Computing and Analysis Techniques in Physics Research (ACAT-2002)*, June 2002, Moscow, Russia.

[5] I. Terekhov *et al.*, "Distributed Data Access and Resource Management in the D0 SAM System" in Proceedings of 10-th International Symposium on

High Performance Distributed Computing (HPDC-10), IEEE Press, July 2001, San-Fransisco, CA

[6] Uniform Resource Identifiers (URI): Generic Syntax (RFC 2396) T. Berners-Lee, R. Fielding, L. Masinter.

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

[8] K. Czajkowski, S. Fitzgerald, I. Foster, C. Kesselman. "Grid Information Services for Distributed Resource Sharing", in the same proceedings as [5].

[9] R. Raman, M. Livny and M. Solomon, "Matchmaking: Distributed Resource Management for High Throughput Computing", in Proceedings of the Seventh IEEE International Symposium on High Performance Distributed Computing, July 28-31, 1998, Chicago, IL.

[10] J. Frey, T. Tannenbaum, M. Livny, I. Foster, S. Tuecke, "Condor-G: A Computation Management Agent for Multi-institutional Grids", in the same proceedings as [5].

[11] Internet Assigned Numbers Authority, http://www.iana.org.

[12] I. Terekhov, "Meta-Computing at D0", plenary talk at ACAT-2002, see [4], in Proceedings.

[13] I. Foster, C. Kesselman, J. Nick, S. Tuecke. "The Physiology of the Grid", Work in Progress.

[14] P. Kunszt, "Status of the EU DataGrid Project", in the Proceedings of ACAT-2002 The project home page is at http://www.eu-datagrid.org.

[15] M. Kunze, "The CrossGrid Project", in the Proceedings of ACAT-2002.

[16] A. Konstantinov, "The NorduGrid Project: Using Globus Toolkit for Building Grid Infrastructure", in the Proceedings of ACAT-2002.

[17] High Energy and Nuclear Physics InterGrid, http://www.hisb.org/

[18] Grid Interoperability, http://www.grid-interoperability.org/.

[19] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana, Web Services Description Architecture (WSDL) 1.1, W3C Note 15, 2001, http://www.w3.org/TR/wsdl.

[20] The Particle Physics Data Grid, http://www.ppdg.net.