Making the Utility Data Center a Power Station for the Enterprise Grid

Sven Graupner, Jim Pruyne, Sharad Singhal
Internet Systems and Storage Laboratory
HP Laboratories Palo Alto
HPL-2003-53
March 17th , 2003*

E-mail: {sven.graupner, jim.pruyne, sharad.singhal}@hp.com

grid, utility computing, utility data center, enterprise systems, infrastructure, resource provisioning

Grids are seen as the 'next big thing' in the computing industry [1], [2]. Governments and industry are currently making significant investments in grid technologies [3] that allow transparent sharing of resources and foster collaboration within and across large organizations. While grid technologies have their roots in scientific computing, it is anticipated that they will also provide significant benefits in enterprise environments. Specifically, grids will enable adaptive resource sharing and collaboration within enterprises. However, there are significant differences between scientific applications and applications typically found in enterprises that must be accounted for when establishing grids within enterprises. The Utility Data Center (UDC) [4] provides an adaptive resource provisioning system targeted at enterprise applications. The UDC and grid provide complementary sets of technologies making the UDC an effective power station for an enterprise grid. This paper outlines the strengths of the technologies and demonstrates an approach how a UDC can be connected to an enterprise grid. This paper documents experimental work done to connect the UDC to the Grid using the Globus toolkit 2.0 [5].

* Internal Accession Date Only
Approved for External Publication

Making the Utility Data Center A Power Station for the Enterprise Grid

Sven Graupner, Jim Pruyne, Sharad Singhal
{sven.graupner, jim.pruyne, sharad.singhal}@hp.com
Hewlett-Packard Laboratories
Internet and Computing Platforms Research Center
1501 Page Mill Rd, Palo Alto, CA 94304

Abstract

Grids are seen as the 'next big thing' in the computing industry [1], [2]. Governments and industry are currently making significant investments in grid technologies [3] that allow transparent sharing of resources and foster collaboration within and across large organizations. While grid technologies have their roots in scientific computing, it is anticipated that they will also provide significant benefits in enterprise environments. Specifically, grids will enable adaptive resource sharing and collaboration within enterprises. However, there are significant differences between scientific applications and applications typically found in enterprises that must be accounted for when establishing grids within enterprises.

The Utility Data Center (UDC) [4] provides an adaptive resource provisioning system targeted at enterprise applications. The UDC and grid provide complementary sets of technologies making the UDC an effective power station for an enterprise grid. This paper outlines the strengths of the technologies and demonstrates an approach how a UDC can be connected to an enterprise grid.

This paper documents experimental work done to connect the UDC to the Grid using the Globus toolkit 2.0 [5].

Introduction

Three objectives are driving technology in the industry for enterprise IT environments:

- *Agility* – adapt IT infrastructures faster to changing business needs and environments,
- *Collaboration* – enable collaboration across organizations, and
- *Return-on-Investment* –increase asset utilization and lower operational costs.

These objectives translate into the following requirements for emerging technologies:

- *Virtualization* of resource infrastructures for application systems. The Utility Data Center (UDC) is an example of such virtualization.
- *Federating* virtualized resource pools within and across organizational domains within or across enterprises. Grid technology is merging into enterprise environments to enable federation.
- *Automation* of system operation and management.

Organizing IT assets in enterprise data centers around grids for providing federated resource and service environments seamlessly across organizational boundaries is just

emerging. Grid technology, originally focused the job-submission or batch models, is progressing rapidly towards a general service-oriented infrastructure where resources and applications are combined in one uniform abstraction of services. This vision is outlined in the Open Grid Services Architecture (OGSA) [6].

HP's Utility Data Center solution provides unique and powerful virtualization capabilities that allow an operator to "virtually wire" complex resource topologies in an easy-to-use manner. Because the UDC maintains logical separation between application environments (farms), it offers a secure and controlled environment for running applications that belong to multiple customers. Furthermore, resources are scrubbed when they are released, thus insuring that subsequent users cannot retrieve any lingering state. Thus, the UDC provides a unique offering for building an enterprise grid.

Currently, UDC customers design and deploy their resource configurations using a web-based portal. Grid protocols, however, are based on resource discovery and on-demand allocation and configuration of resources. The UDC requires that accounts pre-exist for different users, and does not offer matchmaking capabilities between resource users and resource providers. Interoperating with the automated matchmaking capability offered by the grid allows the UDC to be a "power station" within the emerging grid paradigm.

Connecting to the grid requires that the UDC advertise its resources to and accept requests from the grid. Additionally, the XML-based resource description language used within the UDC differs from the resource specification language (RSL) used by the popular Globus toolkit for the grid, and standard resource descriptions from the Global Grid Forum [7] have not been completed. This paper addresses issues in coupling the UDC to open (inter-) grids as well as to enterprise-wide (intra-) grids [8].

UDC in Enterprise Grids

Applications in Enterprise Grids

Applications in traditional, compute- and job-oriented grids and in enterprises are different in term of their workloads, lifetime, resources, and schedulable units. Major differences are summarized in the following table.

Table 1: Major differences between scientific applications in grids and enterprise applications.

	Scientific Grid applications	***Enterprise applications***
Workload	Compute-, job-oriented	Transactional
Lifetime	Duration of job, typically hours or days	Duration of application deployment, typically months, years, or indefinite
Resources	Homogeneous, clusters	Diverse resource topologies
Schedulable unit	(compute) job	Application deployment

Caused by these differences, different sets of technologies have evolved in the grid and in enterprises. Consequently, simply bringing grid technologies and tools into enterprise environments will not match enterprise needs. Pieces need to be identified where technologies overlap or extend the current capabilities in either environment. The following section outlines those areas when comparing grid technology and the UDC.

The goal is to identify where technologies provided by one side are mutually beneficial to the other side.

Grid and UDC – Mutually Beneficial Technologies

The following tables show areas where technology brought by the UDC is beneficial, and partially even unique, to grids, and vice versa.

Table 2: UDC and Grid: mutually beneficial sets of technology (UDC → Grid).

What can the UDC bring to the Grid?	*Why the Grid would benefit?*
Capability to describe and provision complex resource topologies;	The Grid typically only considers one type of resource: CPU (in clusters).
Capability to dynamically flex resources during operation of an application;	Grid applications typically use a fixed set of resources over the duration of an application (job).
System-enforced application isolation and protection;	Grid applications running in a shared cluster cannot be protected from one another for malicious or accidental interference. All grid systems entirely rely on software, and the degree of hardware-enabled isolation and protection cannot be achieved by those systems. Techniques of software-based application sandboxing [9] are complicated and insufficient.
Virtualized resource environments; the UDC provides storage and network virtualization;	Grids currently operate in resource environments that tightly bind applications to resources. Changes in resources require application configuration, and vice versa. In virtual resource environments, applications can maintain their configuration regardless of changes in the

[1] An example is that applications running grid jobs need to be configured before job execution for a particular set of machines (IP addresses, hostnames) assigned to this job by the scheduler. If the job could run in virtual network environment, the job could define its own IP address space regardless what machines will be assigned by the scheduler. This avoids the need to reconfigure the job before execution. This is an example of how configuration complexity for applications disappears when executed in private, virtualized resource environments.

	underlying resources.[1]
Programmable resource deployment and configuration[2];	Resources in grids need to be configured and managed by the operator; changes in configurations have to be communicated to the operator who manually performs them. There is no programmatic interface for changing resource configurations in grid deployments.

Table 3: UDC and Grid: mutually beneficial sets of technology (Grid → UDC).

What can the Grid bring to the UDC?	***Why the UDC would benefit?***
Uniform (OGSA) service model embracing resources and applications with the potential to become industry standard;	The UDC provides resources for applications. It is of value to incorporate the entire variety of physical resources, constructions of those resources (virtual, logical resources) including lower- or higher ordered applications into a uniform model such as proposed by OGSA.
Enable federations of UDC, cross-UDC interoperability;	Grid provides the "glue" middleware to connect resource pools located and managed in large geographically disperse environments making them available to requestors of resources.
(Enterprise) Grid-wide resource discovery, availability and use;	Federations rely on metadata about what resources and capabilities (services) are offered where and under which terms. Grid provides registries and description formats for this purpose. Resource customers will have more choice in which (utility) data center they can deploy their applications. Asset utilization can be improved by providing information what is available where.
PKI-based identification and authentication allowing single sign in into grid resource pools;	UDC relies on its own user control. Users must maintain different accounts in different UDCs. Grid accounts, based on X.509 PKI certificates, are unique and certified globally. Grid certificates can be used across organizations as when needed for establishing partner collaborations between corporations or enterprises.

[2] Programmability is a prerequisite for automation.

Scenario of UDC connected to a Grid

Figure 1 illustrates the scenario how multiple UDCs can be connected through the grid. Each UDC acts as a provider of resources from which resource requestors can chose resources for deploying their applications. In such a grid, the following major functions must be provided:

- Advertisement in registries of available resource capabilities offered by a participating UDC.
- Queries on those registries by resource requestors to identify a candidate UDC offering desired resources.
- Intermediary services such as matchmakers or brokers that may be (but not have to be) involved in identifying or classifying matches between resource requests and resource offerings.
- Routing descriptions of requested resources to the selected resource pool (UDC) with resource pools having the capability to provision these resources at the times and amounts as needed.
- Secure identification and authentication of resource requestors against resource providers and vice versa.

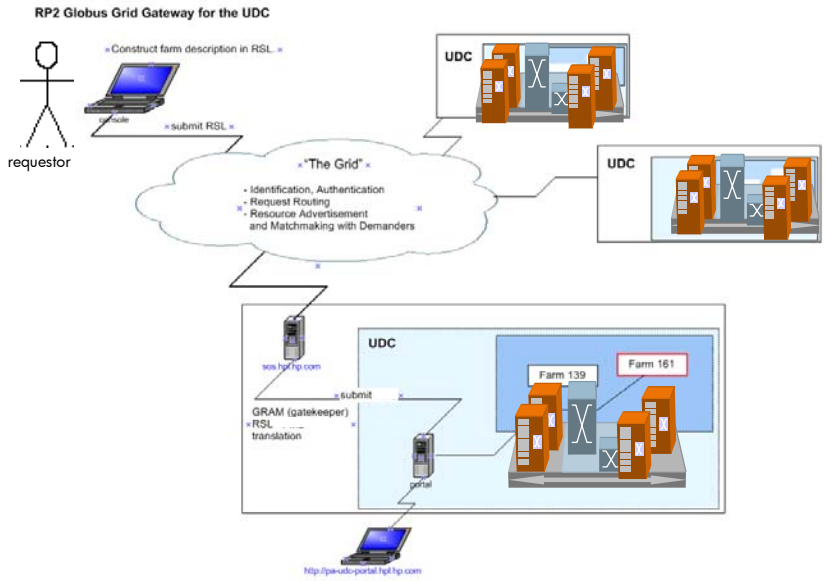

Figure 1: Federation of UDC connected through a Grid.

Figure 1 shows the access path of a user being connected through a grid to one or more UDCs. In the UDC product, a user defines its resource environment (called farm) though a portal that is directly connected to resource pools within the same UDC. Although multiple resource pools may be accessible through the portal, they are still within the same administrative domain.

In a grid scenario, the user constructs its resource environment independently from any particular resource pool (or UDC). Once the resource environment has been constructed, a resource description of the resource environment is sent "to the grid" with the goal of identifying the appropriate place for allocating the user's resources and subsequently deploying applications. This process of matchmaking the description of requested resources with resource offerings can be performed in many ways. The grid, for instance, provides an information service [10] where all resource offerings are listed and can be inspected by the user (manual match-making) or a broker (automated match-making) defined in the grid architecture. Today, manual matchmaking still dominates grid environments.

Once the destination location of resources has been identified, the resource description is routed through the grid to the location where resources are provided. All participating UDCs provide a grid protocol handler (Globus calls it gatekeeper) that receives the request and instructs the underlying resource pools to allocate and/or assign the specified resources and to return the information about the access path to resources to the user.

Advantages of accessing resources through a grid infrastructure rather than accessing local UDC resources directly include:

- Resource requestors (users, customers) have choice among multiple UDCs for deploying their applications;
- Choice enables policy for achieving objectives such as consolidating resources in fewer data centers;
- Resource consolidation leads to improved asset utilization and lower costs in purchases and maintenance;
- Resources from outside can more easily be incorporated in enterprise landscapes when the underlying operational model (grid) is already prepared for this; improved incorporation of external resources also leads to simplifying outsourcing and business opportunities for resource providers;
- Collaboration within and among enterprises resulting from business objectives can more easily be supported by the IT infrastructure establishing virtual organizations (VO) using the single sign-in mechanisms of the grid and accessibility of resources across organizational boundaries;
- Management becomes easier by uniformity achieved through the same (grid, OGSA) abstractions underpinning all resource pools in an enterprise and maintaining an enterprise-wide information system where resource capabilities are provided throughout the enterprise supporting asset management.

Open Grid Services Architecture (OGSA)

The Open Grid Services Architecture (OGSA) [6] was introduced in early 2002 by the leaders of the Globus team along with their sponsors from IBM. The intent was to integrate diverse grid-related projects, applications, and protocols, with the hope of providing greater inter-operability and greater collaboration across teams. For this purpose, they suggested the use of Web Services technologies as a foundation for building an infrastructure suitable for grid computing.

Comparing the Traditional Grid with Emerging OGSA Grids

The Grid, prior to the future envisioned by OGSA, was characterized as a collection of silos, each solving particular problems, but with little commonality or interaction. These silos included operations such as bulk data movement, resource allocation, registration and directory management, and so on. Additionally, each grid-scale application or application suite typically employed its own communication and distribution infrastructure (e.g. HTTP/Web, CORBA, Java RMI, raw sockets, etc.). In short, the Grid was a collection of related but non-interoperable applications and protocols.

Resource management was the most widely tackled problem in the pre-OGSA grid, and is an example of the types of problem OGSA attempts to solve. A variety of projects and products have addressed this problem, including LSF [11], Condor [12], NQE [13], Platform [14], and the Sun Grid Engine [15]. To provide inter-operability between these diverse systems, Globus [5] defined a standard protocol for making resource requests. While not universal, this has enabled resource sharing to occur across some of these otherwise incompatible resource management systems. Sharing leads to the belief that the common OGSA operating infrastructure may lead to better interoperation in a variety of grid computing domains.

OGSA and OGSI

The foundation of the OGSA is the Open Grid Services Infrastructure (OGSI) OGSI [17]. OGSI builds on Web Services protocols and languages, and provides a set of standard interfaces that make up a "Grid service." A Grid service therefore is any web service that implements these interfaces. Roughly, the grid service interfaces can be broken into two logical groups. Those that expose a service's state, and those that deal with service lifetimes.

OGSI introduces `serviceData` as a method of exposing the state of a service. Service data may contain arbitrary attributes for the service, and OGSI defines methods for defining the set of service data elements associated with a service as well as their mutability and other attributes. Additionally, queries over the service data are supported, though the specific query language to be used is left as an area for extension by particular implementations. Updates to service data can be propagated to other interested services using a notification mechanism.

It is assumed that grid services, unlike other web services, will often have short life-cycles. For this reason, OGSI contains life-cycle management services, including factories for creating services, service time-to-live definitions, as well as service termination operations. Additionally, the life-cycle model includes the notion that a single logical service lifetime may be spread over many distinct instantiations of the service. A common difficulty with this separation is that communication identifiers (e.g. IP addresses and ports) cannot be maintained over multiple instantiations. OGSI, therefore, uses a two-level naming scheme. A logical service has a single Grid Service Handle (GSH) for its entire lifetime. The GSH is in the form of a Uniform Resource Identifier (URI) though the interpretation of the URI is outside the scope of OGSI. OGSI does, however, require the notion that a GSH will be mapped into a Grid Service Reference (GSR) with one of a variety of methods. The GSR provides specific information for communicating with the service at a point in time. The GSH to GSR

mapping may change over time, allowing the logical service to have different communication end-points at different times.

OGSA builds upon OGSI, and uses it as the common infrastructure for all the services defined by it. At this point, OGSA is largely an organizational structure; it does not stand for any particular specification.

Grid-Enabling the UDC

Grid enabling the UDC means that the UDC will be connected to the grid, and users will be able to make requests for UDC resources over the grid using manual selection from a Grid information service, or automatic resource discovery as provided by resource brokers.

We break the problem of Grid and UDC integration into the following steps which are described further in the following sections:

- Advertising UDC Resources to the Grid,
- Accepting Resource requests from the Grid, and
- Compliance with Grid Identification and Authentication.

Advertising UDC Resources to the Grid

Resources provided by a UDC must be advertised or registered to the information service[3] that can be queried by resource requestors or brokers. Descriptions of resources have to be presented in form of specifications in RSL (Resource Specification Language, in the traditional Grid). We anticipate that these registries will evolve to an XML based representation which will be accessed in the same method as other Grid services within the context of OGSA.

Traditionally, LDAP directories have been used for GIS. Resources are described as physical resources with named attributes. A common schema for machine resource types includes platform, operating system, CPU information, memory, network interface information, and file system size. In the Globus toolkit, GRIS daemon processes automatically determine this information on hosts that are offered to the grid and populate the GRIS server.

In OGSA, with the goal to uniform resources and applications into services, service registries aim to provide more general means to advertise and query resources and applications as services.

[3] In the traditional grid, this service is called the Grid Information Service (GIS). The Globus toolkit implements a GIS in form of its Monitoring and Discovery Service (MDS) [10] that also provides current utilization data of resources. In OGSA, the service has been generalized to service registries.

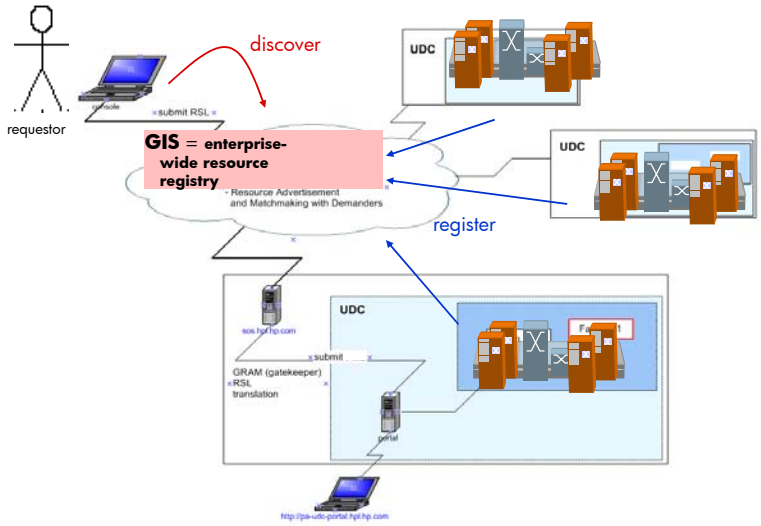

Figure 2: Registration and discovery of resources with the Grid Information Service (GIS).

Further challenges that are desirable in enterprise environments but have not been addressed in OGSA service registries include:

- We need to be able to advertise resource capabilities rather than simple enumeration of physical resource instances (such as machines, disks). This is particularly useful in an environment such as the UDC farms including resources and their topologies can be constructed, but need not exist at all times. The OGSA notion of a service factory may be used for this purpose allowing farms to be instantiated in the same way other Grid service instances are created.
- Another challenge is how queries for entire (wired, configured) resource topologies can be formulated and executed against registered resource capabilities.
- A third challenge is how to adapt resource advertisements to be appropriate to different requestors looking for resources at different levels of abstraction. Some requestors may specify their resources in very detailed terms while others may be satisfied with basic terms (one requestor may ask for 4 machines of type lp2000r running Debian Linux 2.4 being connected through the same network switch for performance reasons, another requestor may just ask for 4 Linux machines; or one requestor is asking for 6 disk devices of a certain type and certain properties, another requestor is simply asking for 1TB storage for 4 months, etc.). The appropriate level of resource abstraction, and potentially multiple levels of abstractions upon the same set of resources are desirable for advertisement and matching resource requests.

Accepting Resource Requests from the Grid

Requests submitted and received in a grid and in a UDC are different. In the grid, jobs are the units that are submitted and executed by a Grid Resource Allocations Manager (GRAM). Job descriptions specify an application to be launched on a specified number of

machines at a particular time. A requestor will be returned a reference to the result of the computation.

In the UDC, a description of a resource topology is submitted to the UDC controller through the UDC portal. A *resource topology* consists of a set of resources with properties including resource connection (logical wiring) relationships. The requestor will be returned a reference and access path to the specified resources.

Grid job submissions are formulated in RSL (Resource Specification language). UDC resource requests are internally described in an XML-based control language that is interpreted by UDC's internal farm controller.

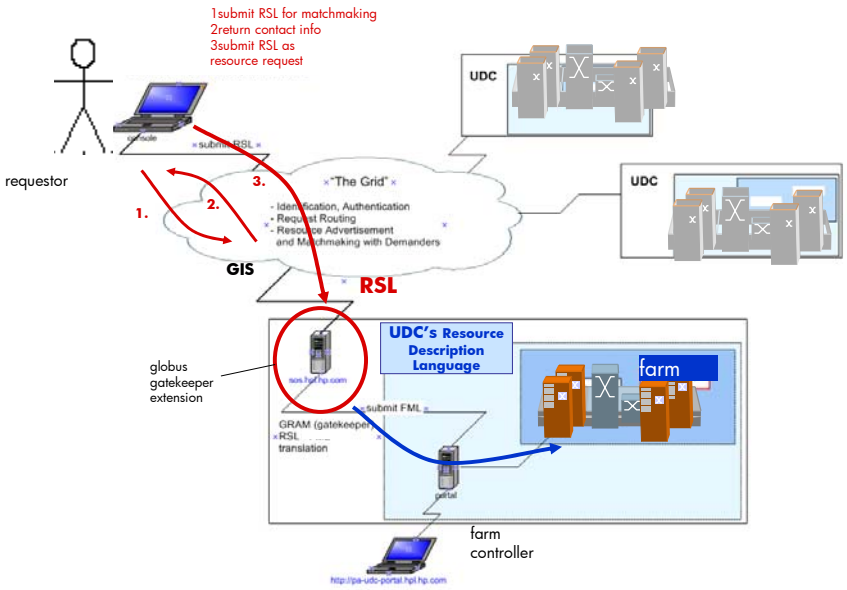

Figure 3: Accepting resource requests from the Grid and rendering a farm.

To receive resource requests from the grid each of the following problems must be solved:

- Define an encoding for UDC resource topologies in RSL. Only RSL can be routed through a grid infrastructure.
- Provide a construction tool for UDC farms that runs independently from any particular UDC instance.
- Build a grid protocol handler as front-end to the UDC that can connect to the grid (and receive resource requests). For Globus, this is a gatekeeper process running in front of the UDC.
- Define a translation of resource requests received in RSL from the grid into UDC's internal control language for further processing within the UDC.
- Provide the tunnel inside the UDC to forward the translated document from the front-end system to the UDC farm controller where the specified resource topology finally can be rendered.

- Provide the information for the access path to resources back to the requestor (such as IP addresses of the farm machine(s) that are exposed to external connections).

The requestor constructs a resource topology for an application using the design tool described below. The example application shown in Figure 4 is a three-tier application. The design tool generates a RSL description from the graphical representation the user has constructed. The RSL description of is submitted to the Grid Information Service (GIS) in order to determine a match with a UDC offering the desired resources[4]. The result of matchmaking is a contact string that specifies the address of a gatekeeper process front-ending the resource pool where the resources finally will be provided. An example of such a contact string is

sos.hpl.hp.com:4322/udc:/O=Grid/O=Globus/OU=hpl.hp.com/CN=Sven Graupner.
|----gatekeeper address----|----------Distinguished Name (DN)----------|

The requestor then will submit the constructed resource topology to the specified gatekeeper process. The contact string consists of two parts, the gatekeeper address where the resource request will be routed to, and the so-called Distinguished Name (DN) of the user. The DN is used for "grid login" and verified by the destination gatekeeper process.

Once the gatekeeper process has verified the requestor's DN, it performs a translation of the resource topology received in RSL into UDC's resource description language and tunnels it to the farm controller where the described resource topology will be rendered.

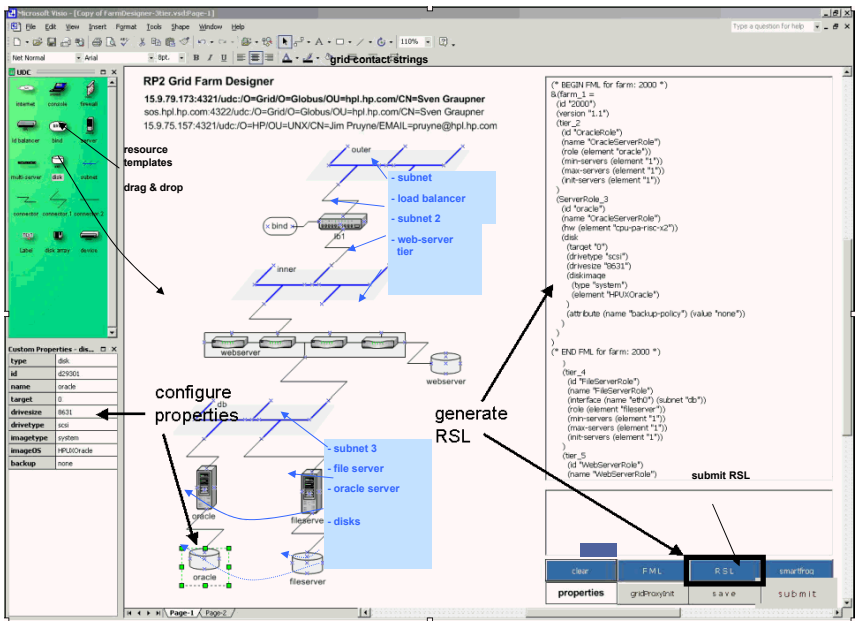

Figure 4: Farm design tool for constructing resource topologies (with an example of a resource topology for a three-tier application).

[4] Matchmaking has not yet been implemented.

Appendix A shows a listing of the resource topology description in RSL for the three-tier resource topology used in the example in Figure 4. The descriptions shown in the Appendix also illustrate that resource descriptions for resource topologies are typically more complex than RSL descriptions for job submissions in traditional grids. For this reason, tools such as the farm design tool need to be provided to create resource descriptions for more complex resource topologies.

Compliance with Grid Identification and Authentication

The Grid Security Infrastructure (GSI) [18] enables secure authentication and communication over an open network. GSI provides a number of useful services for Grids, including mutual authentication and single sign-on.

The primary motivations behind the GSI are:

- The need for secure communication (authenticated and perhaps confidential) between elements of a computational Grid.
- The need to support security across organizational boundaries, thus ruling out a centrally-managed security system.
- The need to support "single sign-on" for users of the Grid, including delegation of credentials for computations that involve multiple resources and/or sites.

GSI is based on public key encryption, X.509 certificates [19], and the Secure Sockets Layer (SSL) communication protocol. Extensions to these standards have been added for single sign-on and delegation. The Globus Toolkit's implementation of the GSI adheres to the Generic Security Service API (GSS-API), which is a standard API for security systems promoted by the Internet Engineering Task Force (IETF).

Assuming that enterprise grids will leverage GSI, the UDC must be compliant with that infrastructure. The gatekeeper front-end process (see Figure 3) implements the GSI protocols allowing only access through the grid identification and authentication process. Associated certificates and user mappings are maintained in the gatekeeper's configuration directory (/etc/grid-security on Unix systems).

User identify themselves by their Distinguished Names (DN)

/O=Grid/O=Globus/OU=hpl.hp.com/CN=Sven Graupner

Initial request are accompanied with respective X.509 user certificates issued by one of grid's Certification Authorities (CA).

```
Certificate:
  Data:
   Version: 1 (0x0)
   Serial Number: 176 (0xb0)
   Signature Algorithm: md5WithRSAEncryption
   Issuer: C=US, O=Hewlett-Packard, OU=UNX, CN=HP (UNX) Grid
Certificate Authority
   Validity
     Not Before: Aug 28 18:12:20 2002 GMT
     Not After : Aug 27 18:12:20 2004 GMT
   Subject: O=Grid, O=Globus, OU=hpl.hp.com, CN=Sven Graupner
   Subject Public Key Info:
     Public Key Algorithm: rsaEncryption
     RSA Public Key: (1024 bit)
       Modulus (1024 bit):
         00:d4:f6:d2:f2:c2:d1:58:0c:e2:e3:52:2d:d2:1e:
         3b:11:63:9e:39:2a:45:fe:12:a0:33:eb:5f:c4:27:
         86:be:a3:66:3e:30:96:9a:a9:c4:7f:03:52:a9:92:
         33:22:1f:9c:7b:5b:ef:f3:6b:e8:24:81:d0:5b:ce:
         e7:9d:da:1d:12:35:6d:18:0c:d3:cd:76:86:87:58:
         01:85:09:0c:67:d8:b1:4d:f7
       Exponent: 65537 (0x10001)
   Signature Algorithm: md5WithRSAEncryption
     7d:21:80:b0:a6:2c:4e:7d:a4:d4:85:aa:61:b1:0d:35:43:19:
     11:bf:01:12:c8:8b:89:46:40:02:0c:3e:73:b5:0b:72:ac:24:
     17:fc:1c:d7:b5:9e:b4:c9:fe:ef:c8:b0:7c:1c:a4:7e:30:9e:
     53:9a:7b:f3:39:bb:ab:20:70:25:ba:f1:43:93:fa:a8:03:48:
     74:38:3d:c2:82:01:8a:76:e3:d3:8e:a4:a1:65:e2:5d:52:91:
     8a:c2:d3:68:83:68:8c:d6:02:0f:e9:c4:4c:0b:fb:e4:7e:af:
     3f:28:09:74:5f:37:b0:c3:d2:6a:b8:8f:dd:3a:0d:c2:f7:42:
     3a:c2
-----BEGIN CERTIFICATE-----
MIICIDCCAYkCAgCwMA0GCSqGSIb3DQEBBAUAMGMx
VQQKEw9IZXdsZXR0LVBhY2thcmQxDDAKBgNVBAsTA1
KFVOWCkgR3JpZCBDZXJ0aWZpY2F0ZSBBdXRob3JpdH
WhcNMDQwODI3MTgxMjIwWjBNMQ0wCwYDVQQKEwRH
```

3d:49:0d:69:7d:85:75:ac:bc:0e:e8:6b:ef:da:26: 31:bc:7e:aa:bd:32:c4:13:b0:2b:6d:95:3d:60:06: 46:9c:f8:22:1b:32:ee:09:0e:e4:5a:f9:86:02:a5:	dXMxEzARBgNVBAsTCmhwbC5ocC5jb20xFjAUBgNVBAM gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBANT20vL… -----END CERTIFICATE-----

Figure 5: X.509 user certificate for user identification and authentication.

OGSA will continue user identification and authentication based on X.509. The Grid's global single sign-in based on DN and certificates is essential for cross-organizational user identification and authentication.

Related Work and Competitive Approaches

Capabilities offered by the grid and the UDC form the foundation for our work. We utilize Globus, the most widely used grid-computing infrastructure as a starting platform. We leverage both concepts and specifications, most notably the RSL and the certificate-based authentication mechanisms, and implementations. Similarly, we leverage the capability offered by the UDC for secure resource partitioning and virtual wiring to create complex application environments. Our work extends the Grid, because it provides no method for describing complex topologies. Similarly, we extend the applicability of the UDC by providing Grid interfaces to it.

Many of HP's chief competitors as well as other smaller providers are converging toward an enterprise grid or resource on demand paradigm. These include high-profile activities such as IBM's autonomous computing [20] and Sun's N1 [21] as well as software only offerings such as those from Platform [14], Grid Engine [15] and Avaki [16]. In this context, grid-enabling the UDC works to level the playing field with these competitors. It allows HP to position the UDC as a preferred resource platform for the Grid over "software-only" solutions, and leverage the "market buzz" created by the Grid. It also offers differentiation for other UDC-like offerings (e.g., Sun's N1) by offering heterogeneous resource pools to grid users and resource providers.

Status and Summary

In this paper, we describe a solution for connecting the UDC to the Grid. The solution consists of the following components:

- A *graphical resource farm designer:* Since the application writer (or user) does not know ahead of time which resource pool will be used by the Grid to deploy the application, the UDC portal, which is tied to a single UDC installation, cannot be used by the user to design farms. We have used Microsoft Visio® to build a drag-and-drop interface for designing farms. The farm can be exported from this designer in a number of syntaxes, including RSL and UDC's resource description language. By providing standard Visio templates, new resource descriptions can be easily made available to users. Because Visio provides a drag-and-drop interface, and it is widely available and used, the farm designer offers a familiar and easy to use method of designing complex resource topologies.
- *RSL translation:* RSL provides a generic way of describing attribute-value hierarchies. However, it does not define semantics that can describe topologies often found in farm descriptions. We have identified a small number of tags that enable us to uniquely map UDC's resource description language (or, indeed any arbitrary XML-based syntax) to RSL. This permits intuitive and simple translations to occur

between both languages. We have used these translations to convert the RSL specifications obtained from the Grid to UDC's resource description language for allocating resources within the UDC.

- A *front-end gateway for the UDC (gatekeeper):* We have used the Globus toolkit (2.0) [5] to implement a gateway between the Grid and the UDC. Globus is the de facto standard in today's Grid, so its use allows us to leverage tools that are in wide use. The UDC gateway can advertise resources to the grid using the Grid Information Service (GIS), and accept requests for those resources as a UDC Grid Resource Allocation Manager (GRAM). Users submit the resource specifications from the farm designer to the UDC using the Grid's resource request submission mechanisms. Inside the gateway, submissions received from the Grid (in RSL) are translated into UDC's resource description language and from then on treated as regular UDC farm deployments. In our current implementation, the gateway provides an alternative channel to the existing UDC portal for farm requests. This approach avoids major changes in the current UDC implementation.

De-coupling the front-end designer from the UDC makes the UDC model of resource provisioning much more attractive to users. Utilizing the gird submission mechanism has the advantage that existing Grid-based mechanisms for discovery, negotiation, and resource brokering can be leveraged. By translating RSL, it is possible to permit independent development of both languages, while maintaining interoperability.

Future Work

We have deployed our solution in the HP Labs, Palo Alto UDC. This solution makes resources within the HP-Labs UDC available to the HP Intra-Grid [8]. We plan to use this solution to offer UDC resources to the Grid community and create collaborative partnerships with others working in this space. A working demonstration of the system has been shown in December 2002.

A number of issues must be resolved to realize the vision of an enterprise grid, and making the UDC a preferred power station on this grid. Among them are:

- Standardization: The grid, like any interoperable solution, relies on a shared core specification. This work is largely happening within the Global Grid Forum (GGF) and is based on the Open Grid Services Architecture. Along with academics and other scientists, we are participating in these efforts, particularly as it applies to core resource description and allocation protocols.
- Arbitration: Another difficult task is to determine what makes for intelligent resource allocation decisions. These must take into account the characteristics of enterprise applications as well as the IT departments and businesses that operate a UDC.
- Production: At present, the enterprise grid concept, and our implementations are in a heavy research phase. A roll-out of this functionality requires further development both at the technological level, and at the mind share level to convince administrators and consumers that this model is sound and worthy of investment.

- OGSA-compliance: The demonstrated version that has been described in this report is based on the Globus toolkit 2.0. With the release of the first version of the Globus toolkit 3.0, which is OGSA compliant, upgrading the solution towards toolkit 3.0 is necessary.

Our vision of a resource utility is broader than the UDC and the open Grid as described here. Our overall research goals encompass these areas:

- Seamless integration of resource pools, applications, and further automation of management tasks remain important fields for research, accelerated with grid technology.

  Our research addresses this by developing an architecture and framework for federated resource environments [22].
- Vast amount of information is involved in federated resource environments. Components participating in the federation must communicate and collaborate with one another. Information must be structured properly in order to reduce complexity. Identifying a common information model that underpins all data maintained throughout the system is another research goal.

  Our research addresses this by developing an information model with abstractions and structures that can underpin all information throughout the system in a uniform manner reducing complexity and supporting interoperability [22].
- Integration of system management tasks (from sensing, data gathering, event detection to decision making and actuation) is vital for automation operation of applications, which is a third goal.

  Our research addresses this by an integrated management overlay.

We plan to experiment with these and other issues in our future work. Our approach is to partner with both the HP product teams as well as external researchers and HP partners to make the vision of the enterprise grid for enterprise applications a reality, with HP offering key products in that realization.

References

[1] The Next Big Thing: Computing Fabrics and Grids, http://www.infomaniacs.com/ComputingFabrics/Computing-Fabrics-And-Grids.htm.

[2] Greg Papadopoulos: The Next Big Thing, http://wwws.sun.com/software/solutions/n1/essays/papadopoulos.html.

[3] Grid Technology Partner, The Global Grid Computing Report 2002, Technology and Market Assessment, 150 Pages, ISBN #0-9721848-0-5, June 2002, http://www.gridpartners.com/publications.html (summary in http://www.hoise.com/primeur/02/articles/live/AE-PL-06-02-4.html).

[4] HP: Utility Data Center, http://www.hp.com/solutions1/infrastructure/solutions/utilitydata, http://www.google.com/search?hl=en&lr=&ie=ISO-8859-1&q=Utility+Data+Center.

[5] The Globus Toolkit, http://www.Globus.org/toolkit/.

[6] Ian Foster, Carl Kesselman, Jeffrey M. Nick, Steven Tuecke, The Physiology of the Grid - An Open Grid Services Architecture for Distributed Systems Integration, DRAFT, http://www.Globus.org/research/papers/ogsa.pdf, May 2002.

[7] The Global Grid Forum (GGF), http://www.gridforum.org/.

[8] HP Intra Grid, http://www.ingrid.hp.com.

[9] David A. Wagner, Janus: an Approach for Confinement of Untrusted Applications, CSD-99-1056, University of Berkeley, http://sunsite.berkeley.edu/TechRepPages/CSD-99-1056, August 1999.

[10] MDS2.2 User's Guide, http://www.Globus.org/mds/mdsusersguide.pdf.

[11] LSF, Platform, Inc., Platform LSF 5, http://www.platform.com/products/wm/LSF/index.asp.

[12] D. H. J Epema, Miron Livny, R. van Dantzig, X. Evers, and Jim Pruyne, "A Worldwide Flock of Condors : Load Sharing among Workstation Clusters" Journal on Future Generations of Computer Systems, Volume 12, 1996 The Condor Project, http://www.cs.wisc.edu/condor/publications.html.

[13] NQE Information from SDSC, http://www.sdsc.edu/projects/production/NQE.

[14] Platform, Inc., http://www.platform.com.

[15] Sun Microsystems, Grid Engine, http://wwws.sun.com/software/gridware/.

[16] Avaki, Inc., http://www.avaki.com.

[17] GGF: Open Grid Services Infrastructure Working Group, http://www.gridforum.org/ogsi-wg/.

[18] Grid Security Infrastructure (GSI), http://www.Globus.org/security.

[19] IETF, Public-Key Infrastructure (X.509), http://www.ietf.org/html.charters/pkix-charter.html.

[20] IBM, Autonomic Computing, Manifesto, http://www.research.ibm.com/autonomic/manifesto, Oct 2001.

[21] Sun Microsystems, N1 - Introducing Just In Time Computing, Whitepaper, 30 pages, http://wwws.sun.com/software/solutions/n1/wp-n1.pdf, http://wwws.sun.com/software/solutions/n1/essays/N1booklet.pdf.

[22] Graupner, S., Pruyne, J., Singhal, S.: Architecture of the Utility Resource Allocation Manager (URAM), (document in progress), HP Labs, 2003.

Appendix A: RSL description for the three-tier resource environment in **Figure 4**.

```
& (farm_1 =                RSL
 (id "496")
 (name "demo")
 (version "1.1")
)
(subnet_2 = (id "db") (name "db") (ip "internal") (vlan "db-vlan"))
(subnet_3 = (id "inner") (name "inner") (ip "internal") (vlan "inner-
vlan"))
(subnet_4 = (id "outer") (name "outer") (ip "external") (vlan "outer-
vlan"))
(lb_5 =
 (id "lb1")
 (name "lb1")
 (type "lb")
 (interface (name "eth0") (subnet "inner"))
 (interface (name "eth1") (subnet "outer"))
 (policy (element "round-robin"))
 (vip
  (name "vip0")
  (subnet "outer")
  (bind
   (id "bind1")
   (name "WebTier:et0")
   (virtualport "8080")
   (realport "8081")
  )
 )
)
(ServerRole_6 =
 (id "OracleRole")
 (name "OracleServerRole")
 (disk
  (target "0")
  (drivesize "8631")
  (drivetype "scsi")
  (diskimage
   (type "system")
   (element "HPUXOracle")
 ) }
 (hw (element "cpu-pa-risc-x2"))
)
(ServerRole_7 =
 (id "FileServerRole")
 (name "FileServerRole")
 (disk
  (target "0")
  (drivesize "8631")
  (drivetype "scsi")
  (diskimage
   (type "system")
   (element "Redhat7.2")
 ) }
 (hw (element "cpu-IA32"))
)
(ServerRole_8 =
 (id "WebServerRole")
 (name "WebServerRole")
 (disk
  (target "0")
  (drivesize "8631")
  (drivetype "scsi")
  (diskimage
   (type "system")
   (element "NT_IIS")
 ) }
 (hw (element "cpu-IA32"))
}
(tier_9 =
 (id "OracleServerRole")
 (name "OracleRole")
 (interface (name "eth0") (subnet "db"))
 (role (element "OracleServerRole"))
 (min-servers (element "1"))
 (max-servers (element "1"))
 (init-servers (element "1"))
)
(tier_10 =
 (id "FileServerRole")
 (name "FileServerRole")
 (interface (name "eth0") (subnet "db"))
 (role (element "FileServerRole"))
 (min-servers (element "1"))
 (max-servers (element "1"))
 (init-servers (element "1"))
)
(tier_11 =
 (id "WebServerRole")
 (name "WebServerRole")
 (interface (name "eth0") (subnet "db"))
 (interface (name "eth1") (subnet "inner"))
 (role (element "WebServerRole"))
 (min-servers (element "5"))
 (max-servers (element "40"))
 (init-servers (element "10"))
)
```