Virtual Grid Resource Attributes

Version 0.7

March 15, 2005

Jerry Chou, Andrew A. Chien, and Henri Casanova
vgrads@cs.ucsd.edu

1. Overview

Virtual Grid resource attributes are used to describe and characterize resources in a virtual grid while the application is using them. That is, when the virtual grid (VG) has already been created, and the application wishes to monitor how well the resources are working out, or if they are changing. The notion and capabilities of virtual grid resource attributes are quite general, and we use them to present a wide range of static (cpu speed, memory, IP address, architecture type) and dynamic (delivery cpu fraction, load factor, network weather service, disk space available) information to the application in convenient form.[1] For example, the current implementation integrates information from the MDS (the Globus LDAP-based information service) and the Network Weather Service, presenting them seamlessly as resource attributes for each VGNode (node which corresponds to a resource within the VG). Ganglia is likely to be another future information source. Note that the role of the virtual grid resource attributes are primarily to provide a simple, convenient, uniform source of information for dynamic resource management. That is to obviate the need for an application developer to use a distinct API and tool for each type of information (e.g. MDS, NWS, Ganglia, etc.).

However, the properties of the attribute will affect what style of implementation is most efficient. Slowly varying attributes, such as are often stored in the MDS are best updated periodically, or if used only rarely may be requested from the underlying information service on demand. A dynamic attribute, such as network bandwidth and latency provided by NWS provides interesting challenges both in compact presentation of information and rapid access.

The set of VG resource attributes is user-extensible, flexible and can even be resource-specific. Each VGNode has a set of default attributes associated with it. Any application (or any user-level program) can provide a resource attribute definition, extending the resource attribute set. Two mechanisms are provided to implement a resource attribute:

- A user program can traverse the VG, defining attribute values using the "setAttribute" method, or
- A user program can be built as a module of vgAgent, allowing the attribute to have a sophisticated implementation, which includes caching and computation on demand.

Figure 1 explains how the VG attributes relate to application controllers, vgES and information services.

[1] VG resource attributes should not be confused with vgDL attributes which are used for selection by the vgFAB. While the names and corresponding semantic values of such attributes may overlap significantly, they are distinct sets and implementations with the Virtual Grid system.

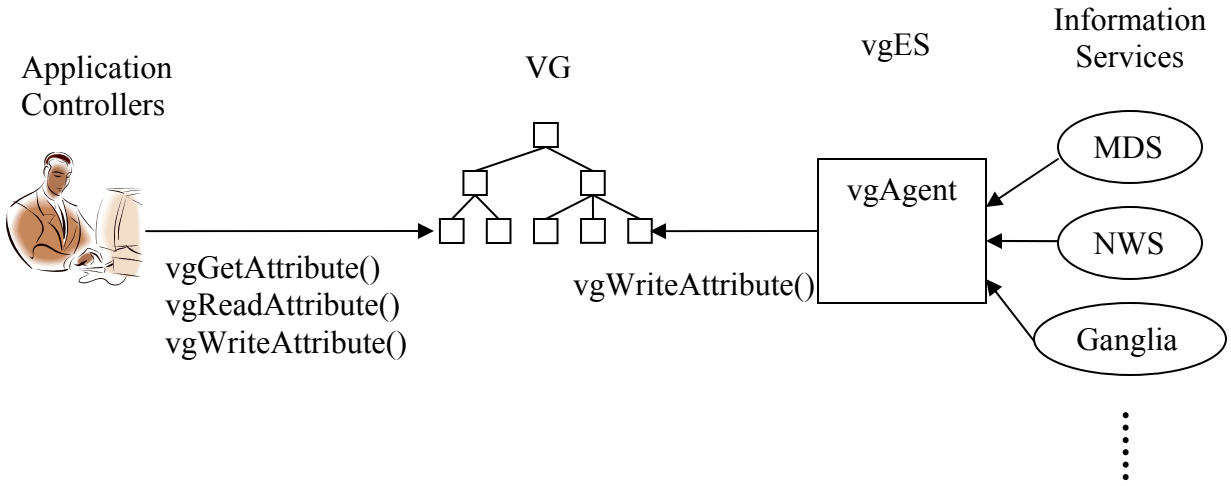

Figure 1: VG resource attributes connect application controllers (applications), VGs, and a collection of information services. Application controllers use vgReadAttribute() to access information. Additional attributes can be defined either by use of vgWriteAttribute() or vgAgent.

In Section 2, we explain extensibility and flexibility. Section 3 lists all current default attributes associated with different types of VGNode. Section 4 shows the VGNode API for adding, reading and getting attributes. Section 5 presents a few examples for using VG attributes. Finally, our future plans are described in Section 6.

2. Flexibility and Extensibility of VG Attributes

In VG, resource attributes are defined as key-value pairs. The key is the name of an attribute, and the values are strings (or in some cases single and multi-dimensional arrays of strings). Each VGNode maintains a list of keys to represent the attributes associated with it. Application controllers can check the existing keys by calling vgGetAttribute() and add a key into the list by calling vgWriteAttribute() on the VGNode. The vgAgent provides the capability to implement resource attributes which are periodically updated to the VGNodes. In general, a VG attribute can be any type of data, created at any time, and updated on any schedule (including on-demand). Applications controllers can access attribute values through vgReadAttribute(). However, if the attribute is not defined a null value is returned.

3. Basic Attributes

To represent all kind of attributes, we divide them into two types: intra-attributes (local) and inter-attributes. Intra-attributes are associated with a single VGNode, while inter-attributes are related to two VGNodes (or more). For example, memory, disk and CPU are intra-attributes, while network measurements of bandwidth and latency are inter-attributes. Since inter-attributes need two VGNodes as reference, inter-attributes are stored internally in a two-dimensional array of values. By contrast, intra-attributes are stored in a one-dimension key-value pair list. The default attributes associated with different types of VGNodes are listed below. For each attribute, we give the variable type, description and example value.

- ***Host***

Intra Attributes:

Attribute Name	Type	Description	Example
Hostname	String	Fully-qualified Public Hostname	Csag-226-248.ucsd.edu
Processor	String	CPU Model Name	Pentium 4
OS	String	OS or kernel version name in	Linux

		vendor-specific convention	
Clock	Integer	Clock speed of a CPU (MHz)	2000
Cache	Integer	Second-level unified cache size of a CPU (KB)	256
CPUs	Integer	Total number of CPUs	2
Memory	Integer	Total Memory Size (MB)	512
Avail_Memory	Integer	Available Memory Size (MB)	200
Disk	Integer	Total Disk Size (MB)	80000
Avail_Disk	Integer	Available Disk Size (MB)	12000
Avail_CPU	Integer	1-minute average processor availability for the host (%)	198
IP	String	A list of IP Addresses available on the host	132.239.226.248; 137.243.34.248; 10.0.0.25

- ***Cluster***

Intra Attributes:

Attribute Name	Type	Description	Example value
Processor	String	CPU model name.	Pentium
CPUs	Integer	Number of CPUs	64
Nodes	Integer	Number of Hosts	32
Memory	Integer	Total Memory Size MB	32768
Avail_Memory	Integer	Total Available Memory Size MB	20000
Disk	Integer	Total Disk Size MB	5120000
Avail_Disk	Integer	Total Available Disk Size MB	300000
Avail_CPU	Integer	1-minute average processor availability for all computing elements (%)	6370

Inter-Attributes:

Attribute Name	Type	Description
BW	Integer[][]	Bandwidth between any two Hosts within a Cluster.
Latency	Float[][]	Latency between any two Hosts within a Cluster.

- ***Bag***

Intra-Attributes:

Attribute Name	Type	Description	Example value
CPUs	Integer	Number of CPUs in a Host	2
Nodes	Integer	Number of Hosts	128
Memory	Integer	Total Memory Size MB	40000
Avail_Memory	Integer	Available Memory Size MB	20000
Disk	Integer	Total Disk Size MB	10000000
Avail_Disk	Integer	Available Disk Size MB	3000000

Inter Attribute:

Attribute Name	Type	Description
BW	Integer[][]	Bandwidth between any two Hosts within a Bag
Latency	Float[][]	Latency between any two Hosts within a Bag.

- ***Close/Far/highBW/lowBW***

Every VGNode with type *Close,Far,highBW* or *lowBW* has exactly two child nodes, because these are binary operations in vgDL. So the Inter attribute value can be simply represented by a single variable.

Inter Attribute:

Attribute Name	Type	Description
BW	Integer	Bandwidth between two Aggregates
Latency	Float	Latency between any two Aggregates

For a VGNode with type *Close*, the value for latency is set to '0'.
For a VGNode with type *highBW*, the value for bandwidth is set to '0'.
For a VGNode with type *Far* and *lowBW*, the values for both bandwidth and latency are set to '0'.
These attribute values are set to '0', because there can be no guarantee of any network connectivity.

4. Attribute JAVA API, VGNode Class

Intra Attribute

- `String[] vgGetAttributes ()`

=>Return the current set of Intra Attributes describing this VGNode as a null-terminated array of null-terminated strings.

- `String vgReadAttribute (String attribute)`

=>Returns the value associated with the Intra Attribute for this VGNode.

- `VGResult vgWriteAttribute (String attribute, String value)`

=>Writes the value associated with the Intra Attribute for this VGNode.

Inter Attribute

- `String[] vgGetInterAttributes ()`

=>Return the current set of Inter Attributes contained in this VGNode as a null-terminated array of null-terminated strings.

- `String[][] vgReadInterAttribute (String attribute)`

=>Returns the pairwise value associated with the Inter Attribute between the child VGNodes of this VGNode.

- `VGResult vgWriteInterAttribute (VGNode node, String attribute, String[][] value)`

=>Writes the value associated with the Intra Attribute between two child VGNodes of this VGNode.

5. Attribute Usage and Example

- Usage:
  1. The attributes can be examined by vgMON to detect a resource violation of the original vgDL requirements.
  2. The Inter Attributes under a Cluster or TightBag can be used by application controller to decide where the best place to locate and/or transfer the data is.

- Example:

//create a VG has a TightBag with four clusters

//choose the two clusters that have maximum bandwidth

//move data from one to the other

//terminate VG

6. Future Work

The VG resource attribute system is the focus of a wide range of innovative activities. Particular areas of focus in the future include:

- Additional information service interfaces (such as Ganglia) thru vgAgent
- Experimentation with caching and on-demand attributes
- More efficient group interfaces to dump inter attribute values