International Journal of Parallel, Emergent and Distributed Systems
Vol. 23, No. 2, April 2008, 153–170

Taylor & Francis
Taylor & Francis Group

Gridscape II: An extensible grid monitoring portal architecture and its integration with Google Maps

Hussein Gibbins and Rajkumar Buyya*

Grid Computing and Distributed Systems (GRIDS) Laboratory, Department of Computer Science and Software Engineering, The University of Melbourne, Melbourne, Vic., Australia

(*Received 31 March 2007; final version received 19 June 2007*)

A pervasive problem seen in information management is information fragmentation, where information may be fragmented by physical location, device or even by the various tools designed to help manage it. Coupled with the explosion of information we see today, identifying information that is really important to us becomes difficult. In areas such as Network and Grid system management, this problem hinders our ability to plan and make intelligent decisions. Grid computing is particularly affected due to the inherent difficulty in dealing with distributed heterogeneous resources over various administrative domains. In this paper we present Gridscape II, a customisable portal component that can be used on its own or plugged-in to compliment existing Grid portals. Gridscape II manages the gathering of information from arbitrary, heterogeneous and distributed sources and presents them together seamlessly within a single interface. To provide an interactive user interface we leverage the Google Maps API, which has recently been adopted extensively as a highly-effective and innovative means of presenting information with geographic location. Gridscape II is simple and easy to use, providing a solution to those users who do not wish to invest heavily in developing their own monitoring portal from scratch, and also for those users who want something easy to customise and extend for their specific needs. Its simple and generic design means it can also directly be used in other, non-Grid related applications.

Keywords: grid computing; Google Maps; monitoring; grid portal

1. Introduction

A pervasive problem seen in personal information management is information fragmentation [11], where information may be fragmented by physical location, device or even by the various tools designed to help manage it. Coupled with the current overabundance of information, this problem severely hinders people's ability to make intelligent decisions and take appropriate actions, due to the inability to easily locate and interpret information. This problem affects many situations where information is important, including Network and Grid system management. Web portals, such as Google's personalised homepages have become a popular means for bringing together independent sources of information into a single web page, and also allowing us to customise what content is displayed and how it is presented, making it easier to access information that is important to us. Figure 1, provides an example where a user has access their calendar, email, weather report and to-do list from a single screen, helping them to better plan their day.

*Corresponding author. Email: raj@csse.unimelb.edu.au

ISSN 1744-5760 print/ISSN 1744-5779 online
© 2008 Taylor & Francis
DOI: 10.1080/17445760701659274
http://www.informaworld.com

Figure 1. Personalised Google homepage portal.

Grid computing [15] has recently emerged as a new paradigm for sharing distributed heterogeneous resources, facilitating truly global collaboration for both enterprises [6] and research communities [7]. The resources that make up these Grids are diverse and include scientific instruments, computational resources, application services and data stores. Each resource has its own set of information relating to its operation or current status and each has a different mechanism for accessing that information. They are likely to also adhere to different standards and provide information based on various schemas and policies. Information services, such as Globus' Monitoring and Discovery System (MDS; [2]) for compute resources or the Storage Resource Broker's Meta Information Catalog (SRB MCAT; [30]), from the San Diego Supercomputer Centre (SDSC), for data stores, are two such examples. These services are a key part of Grid middleware, providing fundamental mechanisms for monitoring resources, which is essential for accurate planning and for adapting application behaviour. In a simple scenario, resource information can be gathered by directly querying these resource or middleware specific information services. However, this may often require manually querying a number of different systems one-by-one in order to collect required pieces of information, as shown in Figure 2. Clearly, being able to manage resources, services and computations is challenging due to the heterogeneous nature, large number, dynamic behaviour, and geographical distribution of these resources. This presents the need for information management tools to assist in gathering and aggregating information and simplifying the task of extracting required information. Users need a customisable tool that presents concise information about available resources from a single interface without requiring them to switch between many tools.

With the introduction of freely-available and easy-to-use APIs such as the Google Maps API [13,18,29], integrating multi-source content and location data with a map into a single

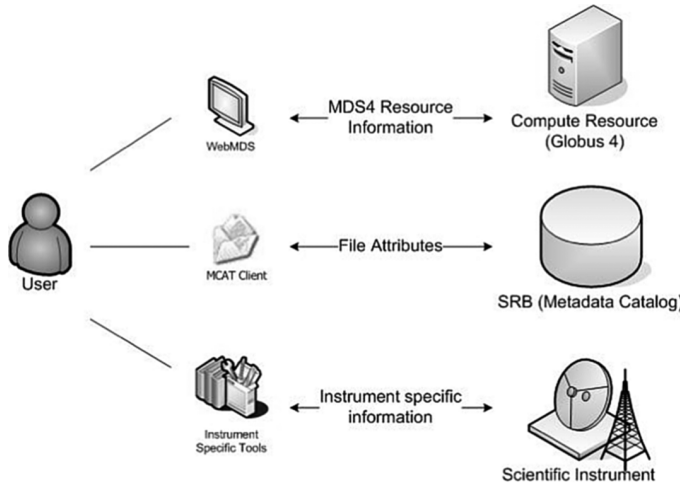

Figure 2. User needs to use a number of different tools to access information from various resources.

interface has become an increasing phenomenon and have been termed 'mashups' [25]. An endless stream of these 'mashups' have been developed, showcasing effective new ways to present data, enhancing existing applications and inspiring completely new ones. Postal companies are now providing visual shipment tracking information to customers so they can see where their package currently is or where it has been, and real-estate agents are providing geographic views of property information allowing house hunters to more easily identify houses in a particular area and visually assess the location and other property information.

Gridscape II leverages the Google Maps API in addressing the information management problem in Grid networks, providing a solution for the creation and management of customised Web portals as a means of providing a single interface to multiple different systems or fragments of information. Since, the ways in which individuals and groups organise and work with data may also differ [28], Gridscape II aims to be flexible and adaptable so that each organisation or individual need not invest heavily in developing or deploying Grid monitoring solutions for their own sets of resources.

Gridscape II incorporates the following key features.

- The ability to manage diverse forms of resource information from various types of information sources;
- A simple and flexible mechanism to support the introduction of new resource types and arbitrary information schemas using a plug-in architecture.
- A flexible mechanism for presenting and formatting information that supports customisation, making the introduction of new types of information easy and allowing different portals to display the same information in a different ways.
- User friendly portal administration to make it easy to manage and configure the portal online.
- Provide a clear and intuitive presentation of resource information in an interactive and dynamic portal.
- A flexible design and implementation such that core components can be reused in building new components and a high level of portability and accessibility can be provided.

The remainder of this paper is structured as follows. Section 2, looks at related work and presents a discussion on the strengths and weaknesses of some currently existing Grid resource monitoring solutions, identifying some unsatisfied requirements and areas of improvement to be addressed by Gridscape II. Sections 3 and 4 discuss how Gridscape II fits into the Grid architecture and provide details of its design and implementation. Section 5, looks at Gridscape II in practice and walks through the key features of Gridscape II illustrating how it is used in practice. Section 6, identifies a number of directions for future work. Finally, we end the paper in Section 7 with our concluding remarks on the work.

2. Related work

Currently, there are a number of solutions that aim to assist in the monitoring of Grid systems. Some of these are successful in their ability to provide detailed low-level system information while the strengths of others are their flexibility and adaptability. Here, we discuss a few representative implementations and see how each addresses or fails to address certain issues.

WebMDS [31] provides a web-based presentation of Globus monitoring information and is distributed with the Globus middleware. Globus provides a flexible mechanism for aggregating and publishing this information into Index Services called MDS. WebMDS allows users to browse this published information of the different components that make up the Globus middleware. The XML information is formatted using XSL transforms for presentation on the web to improve its readability. Some issues with this system are that each WebMDS installation only presents information from one Index Service and it only supports monitoring resources running the Globus middleware. Although, the default configuration aims to simply provide a user readable presentation of all information published through the Index Service, the use of XSLT stylesheets affords the possibility of customisation to filter out any unnecessary information and alter the look and feel.

The Grid monitor of the advanced resource connector (ARC), formally known as the NorduGrid Grid monitor [21], addresses the need for providing useful information by processing and interpreting information provided by MDS and other services and presenting that in a user-friendly interface. One downside to this implementation however, is that it is tailored specifically to the ARC project and is not available to be used to monitor Grid resources other than those that are part of the ARC project and are publishing information to the ARC Information System.

Ganglia [16], is a distributed monitoring system that provides detailed information about resources. Unlike the ARC Grid monitor, Ganglia are a flexible framework that can be utilised to monitor many types of high-performance computing systems including clusters and Grids. However, there is a caveat that in order to achieve high levels of scalability and to provide such detailed resource information, Ganglia relies on Ganglia-specific daemons that run on cluster nodes. These daemons collect information from a local resource and send multicast packets to other entities in the system in order to disseminate the information. The problem with this approach is that considerable effort and administrative overhead is required to set up such systems ensuring all resources are configured with the same software. There are management issues with resources across various administrative domains where resource owners may resist installation based on certain policies or have preference for other monitoring solutions. It may also be the case that existing information services are being integrated, where it is not possible to install such monitoring software or it simply does not make sense.

MonALISA [33] follows a similar approach to Ganglia and provides a comprehensive monitoring solution with a distributed architecture; however, it also suffers from the problem of needing to install monitoring agents on each resource. One of the strengths that MonALISA possesses that the above-mentioned systems ignore is an extremely informative user interface that includes a geographical view of resources, making it easy to observe the statuses of large collections of resources at a glance. This improves the user experience and the ability to quickly identify and navigate for required information. A problem with the MonALISA client is that its comprehensive client is a stand-alone Java application that needs to be downloaded and run on the client computer, somewhat reducing its accessibility.

GridCat [34] is a Web portal based monitoring system that includes a static map to visualise resources and includes detailed Grid information including job status, however, it does not offer a simple means for extending its functionality beyond its current capability, which includes support for Globus, LCG and TerraGrid resources.

More flexible and adaptable solutions are toolkits that assist in the development of Grid portals and work with various types of middleware and information sources. Examples of such toolkits are GridPort [35] and GridSphere Grid Portlets [12]. These toolkits use portlet technology for reusability and aim to support all aspects of Grid systems including job execution, management and resource monitoring. GridPort, which at the time of writing is no longer being actively developed, focuses on providing a toolkit to simplify access to heterogeneous Grid services through a single API. Using their Grid Portal Information Repository (GPIR) architecture, all information is restricted to a common set of attributes of specific schemas in a number of categories such as: jobs, load, status, and network latency. This limitation leads to a loss of richness of information through generalisation. Grid Portlets, an extension of the GridSphere portlet framework, provides a service for monitoring Grid resources and supports customisation for multiple different information sources to coexist. The limitations of this implementation are that there is no single collective view of all resources, no geographical visualisation (map interface), and the process of gathering resource information can only be run within a web server thus restricting its flexibility. Also, while these two approaches use portlet technology, they are small components dependant on larger systems.

MapCenter [22] is another tool to help create portals with a similar approach and design methodology to Gridscape II. MapCenter however is not portlet based and instead generates static HTML at certain poll intervals. Also, Gridscape II has the ability to query any resource type and customise the presentation of that information within a single view, which is unavailable in MapCenter.

3. System overview

The previous implementation of Gridscape [4] and its successor, the subject of this paper, both aim to provide a high-level, user-friendly and highly customisable portal interface in order to present the status of Grid resources. Both leverage existing technology and interact with existing software on Grid resources so no additional installation or configuration on these resources is required. Major improvements over the previous version of Gridscape are that it supports the integration of multiple arbitrary information sources through an extensible design; it provides a simple customisation mechanism to allow it to be enhanced to meet the specific needs of each individual Grid portal. Other improvements are integration with Google Maps, simplified portal administration and the use of portlet-based web components, which means it can be plugged into existing Grid

portals. The new design and integration with Google Maps makes it a lot more flexible and the base framework can be easily adapted to suit various applications, even those that are not related to Grid Computing.

3.1 Architecture

The architecture of Gridscape II is shown in Figure 3. The figure identifies the components that make up Gridscape II and their interactions with each other. The main components are the *Gridscape II Portal* (GSP) which provides the user interface, the *Gridscape II Resource Monitor* (GSRM) which gathers information about Grid resources and the *Gridscape II Core* (GSC) which is the data model for the system. The other important component is the *Interactive Client-Side Map* provided through the Google Maps API. These high-level interactions are described below:

1. The client begins interaction by accessing the GSP web interface and initiating a request.
2. Resource information is queried by the Gridscape portal via the Gridscape core and formatted as a response to the client's browser. This includes the geographic location of points and all other resource information.
3. Map tiles containing satellite images for the geographic area being viewed are downloaded to the client from the Google Map Server.
4. In parallel, and independently of the rest of the system, the Gridscape resource monitor takes care of collecting updated resource information from the Grid and publishing that through the Gridscape core. This updated information then becomes immediately available for display to users.

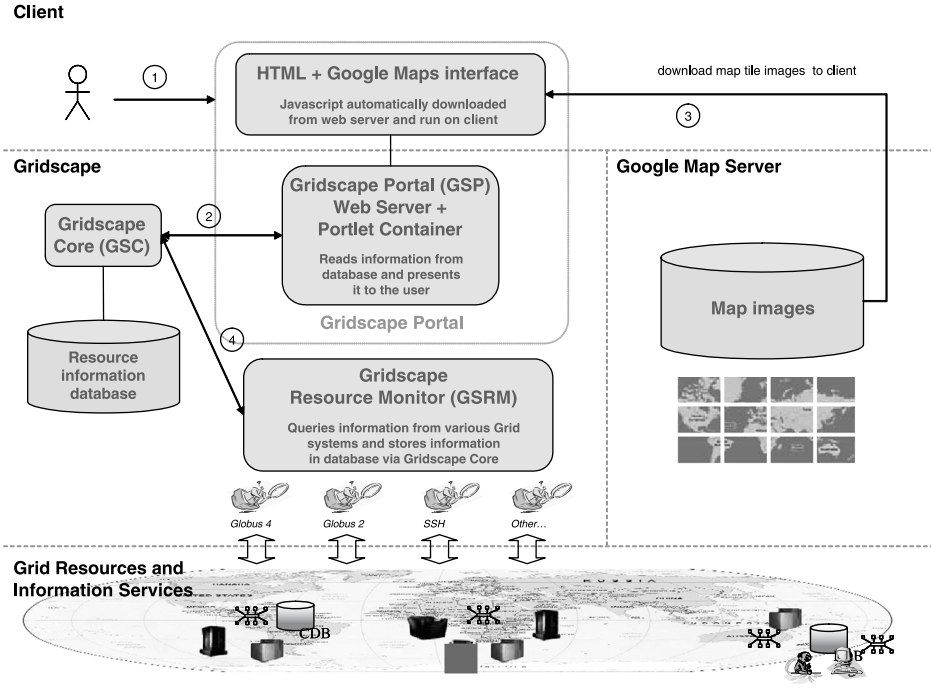

Figure 3. Gridscape II architecture.

The key functions of the system are (a) querying for information from various resource types, (b) storing and retrieving resource information, and (c) presentation of resource information.

3.1.1 Querying information

Information collection is the responsibility of the GSRM. Its role is to gather information from each of the various resources entered into the system and to publish that information for use by other components. The collective set of information becomes the current state of the Grid as seen by Gridscape II and becomes available as a simple set of information to be utilised by a lightweight presentation layer. The importance of having this independent entity gathering information is that it alleviates the burden from the user interface and makes it easy to extend or create new user interfaces that are only required to present information. It was decided not to create an information reporting agent to be installed on each resource, as this makes initial setup and adding of resources to be monitored more difficult and often site administrators do not want to install such agents or in cases where it may not be possible. As a result, Gridscape II can be used in a variety of applications and integrate with various information sources. A plug-in architecture has been adopted so that custom adapters can be created and plugged into the system to enable the gathering of information from any type of information provider.

3.1.2 Storing and retrieving information

The GSC provides a data abstraction layer for accessing and managing Grid resource information. It encapsulates the system state and is the data model representing the set of common entities and the rules governing access and updates to their data. The overall structure of Gridscape II was aimed towards being integrated with a persistent backend so that the updating of resource information and the presentation of that information could be performed by entirely independent entities or separate applications, and so that all state and information is not kept in memory, which was an issue with the original version of Gridscape. The GSC is a reusable component, which can be used to develop other applications or views that need to work with Gridscape II data. The core is used by both the portal and the resource monitor.

3.1.3 Presenting information

The GSP component provides the user interface for the Gridscape II system. Using portlet technology, Grid resource information is presented to the user via their web browser. To improve the interactivity of the otherwise static HTML interface, the Google Maps API is used to provide a highly interactive view of the globally distributed Grid resources and their positioning around the world. It is also important that the information is made available through a Web environment for increased accessibility and availability, as well as providing an intuitive user interface to enable easy navigation. The interface copes with presenting various types of information in a single view.

4. Design and implementation

The overall design of Gridscape II is aimed towards enabling a persistent representation of resource information that can be updated independently from the portal and without having to be run within a web server. Another key consideration is to allow the

architecture to be flexible enough to support multiple different information sources and allow for easy integration of new source types and also provide a simple way to customise the presentation of the information without requiring changes to source code.

JSR168 [9] compliant portlets are used to implement the GSP as a portable and pluggable component, which can be integrated into other portals. The Gridscape II web application follows the Model-View-Controller (MVC) Model-2 type architecture [14]. This architecture provides a means of decoupling the logic and data-structures of the application (server-side business logic code) from the presentation components (web pages). The components are kept as modular and general purpose as possible to make the system easy to extend and adapt to various scenarios.

4.1 Gridscape II Core

The GSC consists of a number of classes. The Resource class represents a Grid resource and contains attributes common to most resources, such as hostname, online status, location and resource specific information. The resource specific information is represented by the ResourceInfo class. This class holds the information, which has been collected by the GSRM. The location class represents the geographic location of that resource. Figure 4 shows the class diagram for this set of components. Both the Resource and ResourceInfo classes implement the Displayable interface. This interface requires the implementation of the toXML() method which returns an XML representation of the current object and is called during information presentation. This mechanism allows us to add new displayable components in the future without significant change to the code. The DBQuerier class has a small number of methods used as the interface to the database.

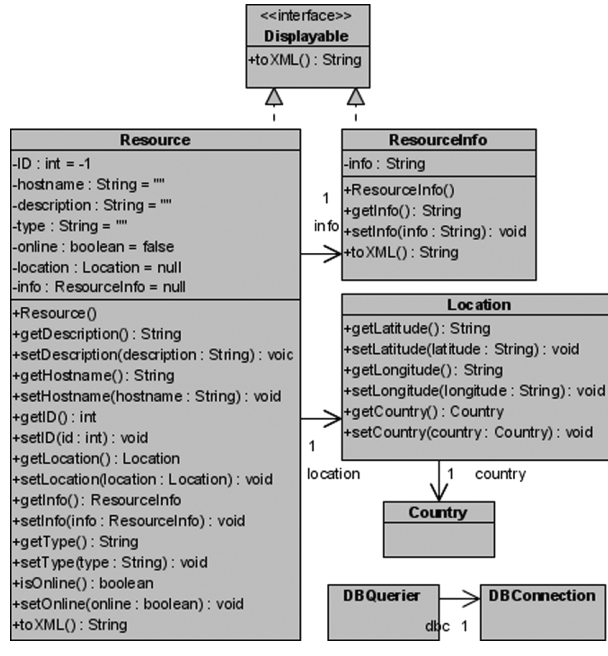

Figure 4. GSC class diagram.

4.2 Gridscape II Resource Monitor

With the plug-in architecture, the GSRM has the flexibility to query various types of information sources, including Grid middleware such as Globus 2 and 4 [23], Alchemi [27], Unicore [26], XGrid [20], SGE [10] and PBS [17]. Therefore, it is able to support different protocols simultaneously, in order for it to gather information from these diverse information sources. The GSRM, shown in Figure 5, is implemented as a standalone Java application. It consists of the ResourceMonitor class, which is the entry point to the GSRM, and the InfoGatherer plug-in interface along with implementations for each currently supported information service type. The GSRM is run on the command line, independently of the GSP in order to gather, and publish via the GSC data abstraction layer, information from the various information service resources. When run, the ResourceMonitor will periodically query each resource that has been entered into the system. Each query for information is started in a new thread to avoid having a query with an excessive response time preventing the following queries from executing. Using the resources *type* information, reflection is used to load the correct plug-in for querying that type of resource. Introducing a new Information Gatherer is simple and only requires implementing the InformationGatherer interface which consists of a single method: gatherInfo(), specific to that resource type. The gatherInfo() method of the selected plug-in is called to execute the actual query. Information received during the gatherInfo() call needs to then be converted into an XML format and is returned in a ResourceInfo object. The resource information in the ResourceInfo object is then stored in the database in its entirety as text. This design allows new types of information services to be introduced and

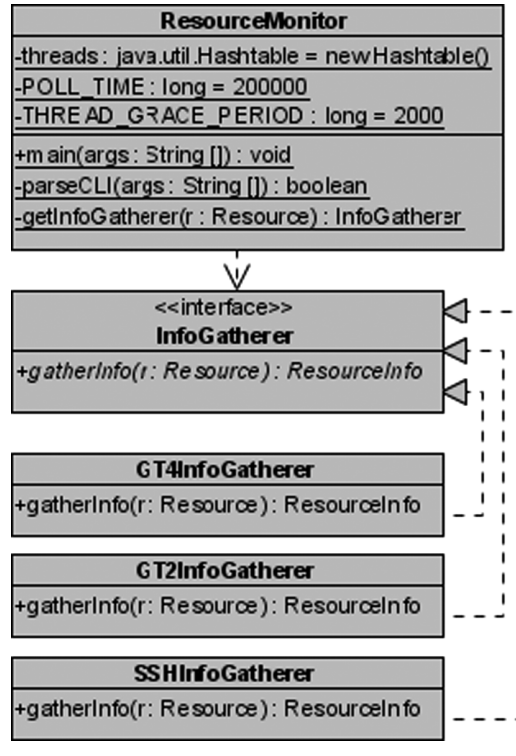

Figure 5. Gridscape II resource manager class diagram.

used with Gridscape II by simply implementing a new plug-in (Information Gatherer) for that service. Security issues for the various information services needs to be handled by the implementation of the relevant Information Gatherer. For example, with the current Globus implementation, a proxy generated local to the web server via Globus tools is used. If SSH is used, an implementation may require a username and password to be supplied or it may use a key to access the resource.

4.3 Gridscape II Portal

The GSP is a web portal developed based on JSR168 compliant portlets. Two individual portlets were created, one for administrating the Grid resources and the portal appearance and another to present the information to users. JSR168 compliant portlets were used to implement the Gridscape portal as a portable and pluggable component, which could be integrated into other portals. Figure 6 shows the two individual portlets that make up the GSP – GridscapeEdit for administrating the Grid resources and portal appearance, and GridscapeDisplay for presenting the resource information to users. The portlets control the program flow and query the database and prepare the information to be presented by the Java Server Pages (JSPs). Each JSP utilises the Google Maps API and contains Javascript code to embed the interactive map interface. They also both use the StyleUtil class to format objects for display at runtime.

4.3.1 Presenting resource information using XSLT

eXtensible Stylesheet Language Transformations (XSLT; [3]) is a way of transforming XML documents into other XML documents via stylesheets. In the case of Gridscape II, we are transforming the XML data gathered by the GSRM into HTML at runtime for presentation in the Web portal based on the appropriate stylesheet. The stylesheet describes how to access nodes of the source XML document using search patterns in the XPath language [19], and also how that should be transformed based on template tags to form the resulting document. The appropriate stylesheet is selected based on the resource type. The benefit of this approach is the flexibility it provides. It means that we can allow the data for each information service to be stored based on any XML schema that is most suitable, and then simply requiring that an XSL document is available to transform that data into HTML that can be presented to the user.

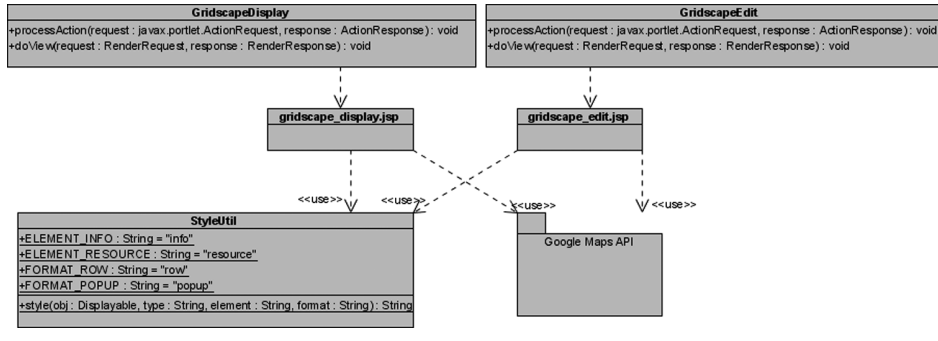

Figure 6. GSP class diagram.

In Gridscape II information is presented in two different ways and therefore requires that two stylesheets are available for each resource type, one for formatting the data to be displayed in an information window popup in the interactive map and another for displaying the information in a list view. The process of presenting resource information is shown in Figure 7. When a request comes in to render the portlet, the list of resources is queried and passed to the presentation layer – a JSP [24]. The JSP iterates through the list of resources to display each resource and makes a call to the helper class StyleUtil in order to perform the XSL Transform and format the information to be presented.

4.3.2 Google Maps API and AJAX

The Google Maps API is a free service offered by Google which allows developers to embed interactive maps (Google Maps) into their web sites. These maps display detailed mapping information and satellite imagery and allows interactivity such as panning and zooming of the map. The implementation of Google Maps is based on a web development technique that has recently become popular and has been termed Asynchronous JavaScript and XML (AJAX; [8]). AJAX is used for making interactive web applications and aims to make web sites feel more responsive by sending asynchronous HTTP requests to a server in the background to retrieve small amounts of data rather than refreshing the entire web page each time content needs to be updated. The way this works for Google Maps, as shown in Figure 8, is that when the map is displayed to the user, a request is made behind the scenes to a map tile server to retrieve the necessary tiles to be displayed. While the user drags the mouse on the map to pan the view, requests are made to the map server to fetch new tiles for the new region of the map to be displayed.

The Google Maps API also allows developers to draw overlays over their maps. These overlays include markers and paths. Markers are used to show points of interest in the form of an icon on the map. These icons are positioned based on latitude and longitude coordinates. They allow for the ability to pop up an information window above them upon user interaction. This capability is used in Gridscape II to indicate the position of Grid resources and detailed information about each resource is made available in each marker's information window. Gridscape II dynamically generates JavaScript at runtime to produce markers for each resource.

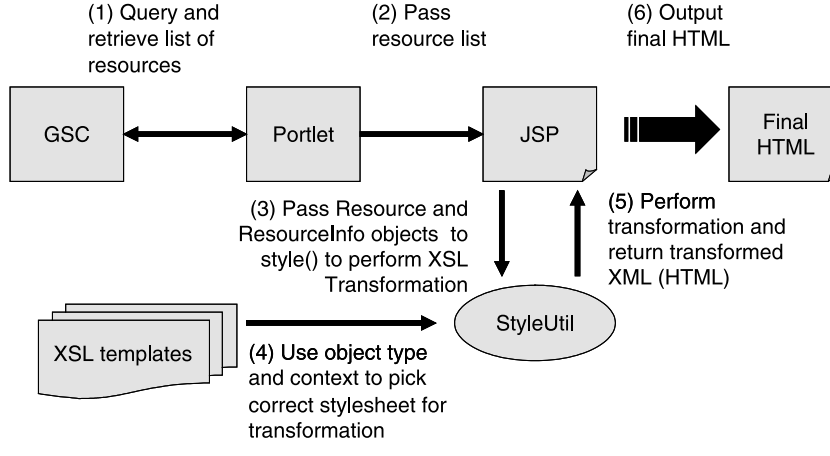

Figure 7. Using XSLT to format objects for presentation.

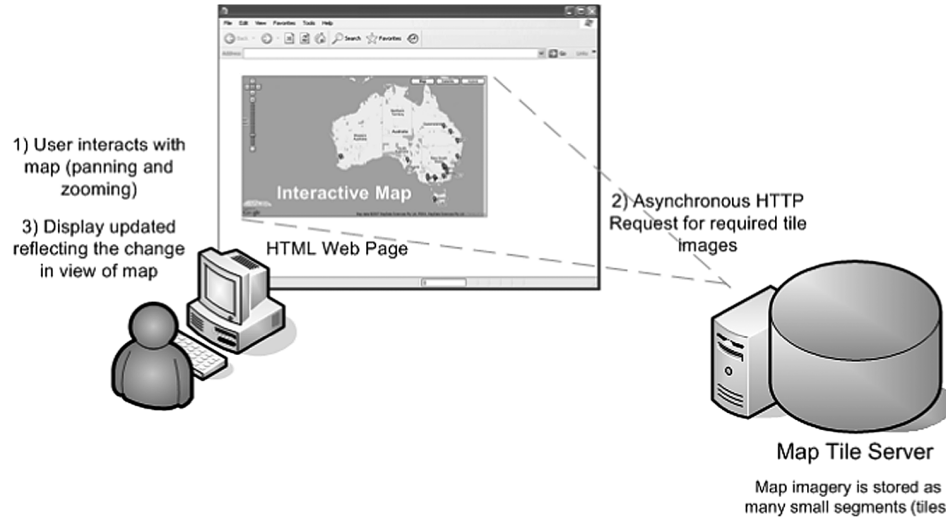

Figure 8. Interacting with the Google Maps user interface.

The remainder of this section will provide some technical detail on the integration of the portal with Google Maps, which enables the display of the interactive map and resource information. For the sake of explanation we assume that the administrator has already customised the appearance of the map and defined a set of resources to view, details of which appear in the following section, Section 5. The configuration and resource information for display is first read from the database and passed onto the JSP for presentation. In the JSP we define the variables 'appcfg' and 'resources' and assign them with the map configuration and resource information respectively, to be accessed later. To display the map, the Google Maps API requires a HTML DIV element to be created as a place-holder for the map. The height and width attributes specified for the DIV become the dimensions for the map.

```
<div id="map" style="width:<%= appcfg.getDimW()%>;
              height:<%= appcfg.getDimH()%>"></div>
```

The DIV element is given an 'id' which is required to identify it as the place-holder and the custom 'height' and 'width' settings are taken from the configuration object 'appcfg'. The next step is to initialise the map and set its initial state based on the configuration settings of the admin. Here the id of the DIV is passed during the creation of the map object.

```
        map = new GMap(document.getElementById("map"));
```

The initial location at which the map is centred at, and the level of zoom is also set, along with the map type and which controls are available to the user.

```
    map.centerAndZoom(new GPoint(<%= appcfg.getLat()%>,
                                 <%= appcfg.getLng()%>),
                                 <%= appcfg.getZoomLevel() %>);
    map.setMapType(<%= appcfg.getMapType() %>);
    map.addControl(new GMapTypeControl());
```

Depending on whether the admin wants users to be allowed to have pan or zoom interactivity the appropriate options are set.

```
<% if (appcfg.isAllowZoom()) { %>
        map.addControl(new GLargeMapControl());
<% } else { %>
        map.disableDragging();
<% } %>
```

Once the map has been configured, the markers representing each resource then need to be displayed. A loop is used in the JSP to generate appropriate JavaScript code for each marker. The information required to create the marker includes the resources location and the specific information about the resource. At the end of the loop, the resource information is formatted using XSLT as described in Section 4.3.1.

```
<% for (Iterator it=resources.values().iterator();it.hasNext();) { %>
<%     r = (Resource)it.next(); %>
<%     l = r.getLocation(); %>
<%     info = r.getInfo(); %>

var point = new GPoint(<%= l.getLatitude() %>,<%= l.getLongitude() %>);
var marker = new GMarker(point,icon);

GEvent.addListener(marker, 'click', function() {
       showInfo(<%= r.getID() %>);
});

map.addOverlay(marker);

markers[<%= r.getID() %>] = marker;
markersInfo[<%= r.getID() %>] =
              '<%= StyleUtil.style(info,r.getType(),
              StyleUtil.ELEMENT_INFO,StyleUtil.FORMAT_POPUP) %>';
<% } %>
```

5. Gridscape in practice

5.1 Deployment

Gridscape II is easily deployed and installed. A simple script is provided to deploy the portlets into a portlet container running in a web server and another script is used to set up the SQL database. No additional software needs to be installed on the resources that are to be monitored. Each information service is queried at a specified poll interval, so, communication overhead is controlled by varying the length of the interval. In the remainder of this section we look at managing and browsing the GSP.

5.2 Customising the portal

The Gridscape II edit portlet allows administration of Grid resources and general presentation details for your portal. Here, resources can be added and removed and existing resources' details can be modified. The advanced options allow for customization of the interactive map including its size, the part of the world to focus on and whether users should be allowed to zoom and pan.

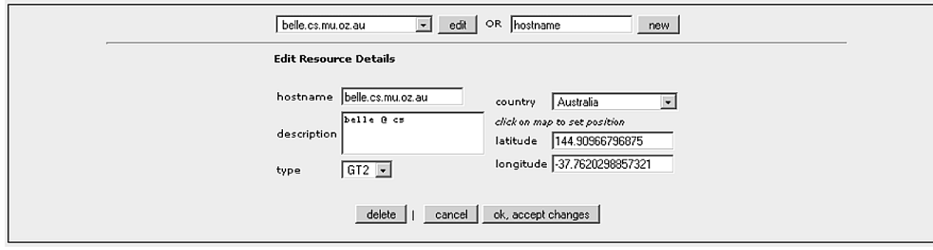

Figure 9. Editing resource details in Gridscape II.

5.2.1 Adding a new resource

Adding a new resource requires simply entering the hostname and then submitting the form. This will add the new resource to the list of existing resources and select it for editing so that its details can be entered.

5.2.2 Editing resource details

To change the properties of a resource, select the desired resource. This will present the properties of that resource in a form allowing them to be modified, as shown in Figure 9. The resource will also be shown as a green marker on the map. The hostname and type must be set correctly for the resources information to be successfully queried by the resource monitor. The location of the resource can be specified by clicking directly onto the map in the desired position. The map will update by displaying the new position.

5.2.3 Deleting a resource

Resources that are no longer part of your Grid can be deleted from the system while selected for editing. This will permanently remove the resource, its marker and all other information that was gathered.

5.2.4 Advanced customisation

The advanced customisation options shown in Figure 10, allows the administrator to enter the Google Maps API key for the domain under which Gridscape II will run, which is a necessity when applying Google Maps to any Web site. The appearance and functionality of how the portal will be displayed to users can also be controlled here.

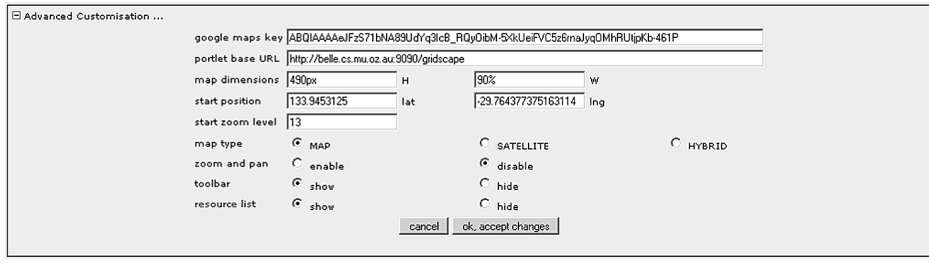

Figure 10. Advanced customisation options in Gridscape II.

5.3 Browsing the portal

Once customisation is complete, the portal can be made available to users via the display portlet shown in Figure 11. The GSRM should also be allowed to run so that the resource information can be gathered and updated for display in the portal.

5.3.1 Interacting with the map interface

The map area shows the positions of all resources that have been entered into the GSP. The user can interact with this by panning and zooming using the mouse and graphical interface controls. When a resource marker is selected, the information for that resource will be shown in an information window that will pop up above the marker. This information will be specific to the type of information source the user has selected.

5.3.2 Search toolbar

A search toolbar is available allowing users to search for specific resources using keywords. The search will look for any information or property that matches the keyword. Only those resources that are found to match will be displayed in the portal.

5.3.3 Resource list

The resource list is displayed under the map and lists the properties and a summary of the resource specific information for each resource. This gives the user an overall view of the state of all the resources at a glance.

Figure 11. Gridscape II display portlet for viewing resource information.

6. Future work and applications

In this section, we discuss a number of ideas we have identified for enhancing Gridscape II and outline some potential extension applications for the system.

Integration with Globus: As identified earlier, the Globus middleware provides a sophisticated information aggregation and publishing system, but the tools for presenting that information to users is lacking. A possibility that we have investigated is the integration of Gridscape II with Globus Index Services. In this scenario, the GSRM could be replaced by a Globus Index Service. This would provide an enhanced user interface to groups of Globus resources and offers an alternative to WebMDS.

Monitoring grid applications: Another extension for Gridscape II is to introduce the ability to include additional layers of information and presenting that information using the multiple overlays supported by Google Maps. Using this feature we intend on presenting information related to Grid application execution by integrating with the Gridbus Broker [5]. The Gridbus Broker, which is used for managing the execution of Grid applications, can provide information such as where jobs are executing, job statuses, data communication paths, which resources involved in a particular workflow and workflow progress. The aim is to assist in managing and steering Grid applications. A similar extension we are interested in pursuing is to visualise the groups of resources involved in Virtual Execution Environments so as to observe the distribution of these networks of resources and how they expand and contract in relation to workload and demand.

Support for additional resource types: We would like to provide support for a larger number of existing resource types, such as Alchemi [27], Unicore [1], XGrid [20] and other arbitrary nodes via SSH. Support for other types of Grid resources such as Virtual Organisation systems [15] and data nodes may also be provided in a future release.

Automating resource positioning: Currently, the geographic location of resources needs to be specified manually, which is a time consuming process. We intend on improving this by geolocating resources based on their IP or hostname or by using there supplied global positioning coordinates. It is possible for resources to transmit their global positioning coordinates to Gridscape II while they are queried, and this is something currently defined in the GLUE schema [32]. Using this, the global positioning of resources could be more accurately plotted without the need for manual intervention.

Compatibility with other Mapping APIs: There are a number of other Map APIs from companies such as Microsoft (Windows Live Local) and Yahoo! (Yahoo! Maps) which are very similar to Google's, providing similar APIs. We would like to adapt the Gridscape portal component so that it is compatible with other Mapping APIs to provide additional flexibility. Another possibility is to integrate with 3D mapping frameworks such as Google Earth.

7. Conclusions

With the overwhelming amount of Grid resource information being made available from various sources, tools such as Gridscape II, which provide a simple mechanism to navigate and view relevant resource information, are required in order to improve our efficiency in working within such environments. There are currently a number of tools designed to monitor very specific details about a single type of resource or project, however, it is difficult for diverse information to be integrated into a single interface or for them to be customised and applied in other scenarios or Grids. Gridscape II provides a highly flexible solution that allows integration with current and future information sources, as well as providing an easy mechanism for customising which information is to be displayed and

how it should be presented. Essentially Gridscape II can be applied in many situations and need not be restricted to any specific middleware or Grid project or even to Grids in general.

We are currently extending Gridscape II to support new capabilities introduced in Section 6. The current version of Gridscape II is available for download at: http://www.gridbus.org/gridscape

Acknowledgements

We would like to acknowledge Google for inspiring the latest version of Gridscape with the release of its Google Maps API. We appreciate Srikumar Venugopal for sharing his thoughts on the implementation of Gridscape II. This work is partially supported by research grants from the Australian Research Council (ARC) and the Department of Education, Science and Training (DEST).

References

[1] C. Baru, R. Moore, A. Rajasekar, and M. Wan, *The SDSC storage resource broker*, in *Proceedings of the 1998 Conference of the Centre for Advanced Studies on Collaborative Research (CASCON'98),* 30 November – 3 December, 1998, Toronto, Canada.

[2] F. Bonnassieux, R. Harakaly, and P. Primet, *MapCenter: An open GRID status visualization tool*, in *Proceedings of ISCA 15th International Conference on Parallel And Distributed Computing Systems,* 19–21 September, Louisville, Kentucky, USA, 2002.

[3] K. Czajkowski, S. Fitzgerald, I. Foster, and C. Kesselman, *Grid information services for distributed resource sharing*, in *Proceedings of the 10th IEEE International Symposium on High Performance Distributed Computing (HPDC-10, 7–9 August 2001, San Francisco, USA)*, IEEE Computer Society Press, Los Alamitos, CA, USA, 2001.

[4] T. Erickson, *From PIM to GIM: Personal information management in group contexts*, Commun. ACM ACM Press, New York, 49(1) (January 2006), pp. 74–75.

[5] I. Foster and C. Kesselman, *Globus: A metacomputing infrastructure toolkit*, Int. J. Supercomputer Appl. 11(2) (1997), pp. 115–128.

[6] I. Foster and C. Kesselman (eds.), *The Grid: Blueprint for a Future Computing Infrastructure*, Morgan Kaufmann Publishers, San Francisco, CA, USA, 1999.

[7] J.J. Garrett, *Ajax: A new approach to web applications*, http://www.adaptivepath.com/publications/essays/archives/000385.php, 18 February 2005.

[8] W. Gentzsch, *Sun grid engine: Towards creating a compute power grid*, CCGrid in *First International Symposium on Cluster Computing and the Grid*, 2001, p. 35.

[9] H. Gibbins and R. Buyya, *Gridscape: A tool for the creation of interactive and dynamic grid testbed web portals*, in *Proceedings of the 5th International Workshop on Distributed Computing (IWDC 2003, 27–30 December 2003, India)*. ISBN: 3-540-20745-7, Springer Verlag Publications (LNCS Series), Berlin, Germany, 2003, pp. 131–143.

[10] Globus 4 WebMDS homepage, http://www.globus.org/toolkit/docs/4.0/info/webmds/, accessed March 2007.

[11] GLUE Schema, http://glueschema.forge.cnaf.infn.it/, accessed March 2007.

[12] Google Earth homepage, http://earth.google.com/, accessed March 2007.

[13] Google Maps Mania, http://googlemapsmania.blogspot.com/, accessed March 2007.

[14] GridCat, http://www.ivdgl.org/gridcat/home/, accessed in April 2006.

[15] R.L. Henderson, *Job scheduling under the portable batch system*, in *Proceedings of the Workshop on Job Scheduling Strategies for Parallel Processing*, 949, Springer-Verlag, London, 1995, pp. 279–294.

[16] T. Hey and A.E. Trefethen, *The UK e-science core programme and the grid*, Future Generation Comput. Syst. 18(8) (2002), pp. 1017–1031.

[17] Java Server Pages Technology (JSP), http://java.sun.com/products/jsp/, accessed April 2006.

[18] JSR 168, http://jcp.org/en/jsr/detail?id = 168, accessed April 2006.

[19] D.R. Karger and W. Jones, *Data unification in personal information management*, Commun. ACM 49(1) (January 2006), pp. 77–82, ACM Press, New York.

[20] D.A. Kramer and M. MacInnis, *Utilization of a local grid of Mac OS X-based computers using Xgrid.HPDC-13*, in *Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing*, IEEE Computer Society 2004, pp. 264–265.

[21] I.C. Legrand et al., *MonALISA: An agent based, dynamic service system to monitor, control and optimize grid based applications*, CHEP 2004, Interlaken, Switzerland (September 2004).

[22] A. Luther et al., *Alchemi: A.NET-Based Enterprise Grid Computing System*, in *Proceedings of the 6th International Conference on Internet Computing (ICOMP'05),* 27–30 June, Las Vegas, USA, 2005.
[23] M.L. Massie et al., *The ganglia distributed monitoring system: Design, implementation and experience*, Parallel Comput. 30(7) (July 2004).
[24] J. Novotny, M. Russell, and O. Wehrens, *GridSphere: A portal framework for building collaborations*, J Concurrency Comput.: Practice and Experience 16(5) (2004), pp. 503–513.
[25] J. Rolia, J. Pruyne, X. Zhu, and M. Arlitt, *Grids for enterprise applications*, in *Proceedings of the 9th Workshop on Job Scheduling Strategies for Parallel Programs (JSSPP, Seattle, USA, June 2003)*, LNCS Volume 2862, Springer Verlag, Berlin, Germany, Oct 2003.
[26] M. Romberg, *The UNICORE grid infrastructure*, Sci. Program. 10(2) (2002), IOS Press pp. 149–157.
[27] G. Seshadri, *Understanding JavaServer Pages Model 2 architecture*, December 1999, http://www.javaworld.com/javaworld/jw-12-1999/jw-12-ssj-jspmvc.html, accessed March 2007.
[28] O. Smirnova et al., *The NorduGrid architecture and middleware for scientific applications*, in *International Conference on Computational Science (ICCS, Springer)*, 2003.
[29] The Google Maps API homepage, http://www.google.com/apis/maps/, accessed April 2006.
[30] The Yahoo! Maps Developer APIs homepage, http://developer.yahoo.com/maps/, accessed March 2007.
[31] M. Thomas and J. Boisseau, *Building grid computing portals: The NPACI grid portal toolkit*, in *Grid Computing: Making the Global Infrastructure a Reality*, F. Berman, G. Fox and T. Hey eds., Ch 28, John Wiley and Sons, Chichester, 2003.
[32] S. Venugopal, R. Buyya, and L. Winton, *A grid service broker for scheduling e-science applications on global data grids*, J. Concurrency Comput.: Practice and Experience 18(6) (2006), pp. 685–699, New York, USA, May 2006.
[33] Virtual Earth Interactive SDK homepage, http://dev.live.com/virtualearth/sdk/, accessed March 2007.
[34] XML Path Language (XPath) W3C Recommendation, http://www.w3.org/TR/xpath, accessed March 2007.
[35] XSL Transforms (XSLT) W3C Recommendation, http://www.w3.org/TR/xslt, accessed March 2007.