A SOA-based Solution for Resource Monitoring within a Grid System

Sabin C. Buraga and Alina Sirbu

Abstract—**The paper presents the architectural details and the practical deployment aspects concerning a service-oriented application focused on the resource monitoring within a Globus-like Grid system. The implementation employs the Java and .NET technologies and the user interaction is facilitated by a usable Web interface.**

Index Terms—**Distributed Computing, Grid, Resource Management, Web Service-based Interaction**

I. Introduction

GRID *computing* [8] – considered as a new paradigm for next-generation computing – enables the sharing, selection, and aggregation of world-wide distributed heterogeneous (hardware, software, logical) resources for solving large-scale problems in different areas of interest or for proving access to massive repositories of data, information, or knowledge.

Resource management and scheduling in existing environments is a complex task. The geographic distribution of the resources owned by diverse organization with different usage policies, cost models, and varying load and availability patterns is problematic. The producers – the owners of resources – and consumers – the users of resources – have different goals, objectives, strategies, and requirements [8, 10].

The paper will present a practical solution for easy remote administration of a given Grid system by using a modular Web application which facilitates a proper interaction between the Grid platform and users (administrators). This solution is denoted by *GriW* – a Service-Oriented Architecture (SOA) for resource monitoring within a Grid based on the Globus Toolkit.

The article has the following structure: next section shortly presents the most important aspects of the Grid computing. Section III describes the most important functional requirements, and section IV gives a general perspective of the GriW service-based architecture. Several examples of practical deployment are also presented. The paper concludes with final remarks and further directions of research.

II. Overview of Grid Computing

The actual Internet technologies' opportunities have led to the undreamt opportunity of using distributed computers as a single, unified computer resource, conducting to what is known as *Grid computing* [8, 10, 12]. Grids enable the sharing, selection, and aggregation of a wide variety of heterogeneous resources, such as supercomputers, storage systems, data sources, specialized devices (e.g., wireless terminals) and others, that are geographically distributed and owned by diverse organizations for solving large-scale computational and data intensive problems in science, engineering and commerce. Our previous research directions concerning different topics are available in [4-7].

According to IBM, "Grid computing allows you to unite pools of servers, storage systems, and networks into a single large system so you can deliver the power of multiple-systems resources to a single user point for a specific purpose. To a user, data file, or an application, the system appears to be a single enormous virtual computing system." [13]

Another most used definition is the following [15]: "Grid computing enables virtual organizations to share geographically distributed resources as they pursue common goals, assuming the absence of central location, central control, omniscience, and an existing trust relationship".

Virtual organizations can span from small corporate departments that are in the same physical location to large groups of people from different organizations that are spread out across the globe. Virtual organizations can be large or small, static or dynamic. An example is one concerning medical communities composed by several types of users: hospitals personnel, clinical employers, patients themselves.

A *resource* is a shared entity available in the Grid. It can be computational, such as a personal digital assistant (PDA), laptop, desktop, workstation, server, cluster, and supercomputer or a storage resource such as a hard drive in a desktop, RAID (Redundant Array of Independent Disks), and terabyte storage device. Other types of resources are the I/O ones: sensors, networks (e.g., bandwidth), printers, etc. Within a Grid, logical resources are also available: users, time counters and others.

This work was supported in part by the Romanian Ministry of Education and Research under the "TELEMON – Complex System for Real-Time Telemonitoring of Pacients" – Partnership Program (PNCDI PN II) Research Grant (2007-2009).

Sabin C. Buraga is with the Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iasi, Romania (phone: +40 232 201090; fax: +40 232 201490; e-mail: busaco@info.uaic.ro).

Alina Sirbu is a Bachelor of Science, Faculty of Computer Science, "Alexandru Ioan Cuza" University of Iasi, Romania.

The absence of a central location and central control implies that Grid resources do not involve a particular central location for their management.

The ultimate key point is that in a Grid environment the resources do not have prior information about each other nor do they have pre-defined security relationships [1].

Related technologies to Grid computing are peer-to-peer (P2P) network architectures, cluster computing and, of course, Internet and Web computing.

Grid applications are distinguished from traditional Internet applications – mostly based on client/server model – by their simultaneous use of large number of (hardware and software) resources. That implies dynamic resource requirements, multiple administrative domains, complex and reliable communication structures, stringent performance requirements, etc. [8].

According to [1], some of the important issues regarding resource sharing across boundaries of organizations are the following:

- Identity and Authentication;
- Authorization and Policy;
- Resource Discovery;
- Resource Characterization;
- Resource Allocation;
- Resource Management;
- Accounting/Billing/Service Level Agreement (SLA);
- Security.

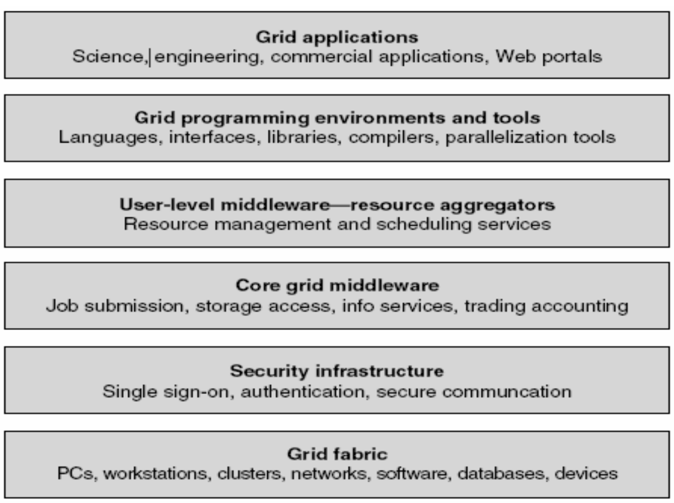

Fig. 1. Layered Grid architecture [8].

Because the Grid systems are multifaceted, they present a layered architecture – see Figure 1. One of the most important initiatives in this area is the *Open Grid Services Architecture* (OGSA) that employs the use of Web services technologies in the context of Grid computing [3]. Grid services are in fact Web services [9] executed to give access to resources by using actual Web technologies and languages (e.g., WSDL – Web Service Description Language, SOAP protocol, XML – Extensible Markup Language).

A standardized model of infrastructure is available: *Open Grid Services Infrastructure* (OGSI) [10] – see also Figure 2.

In order to include different Web service extensions – for example, WS-Security and WS-Trust – and to define stateful Web services, an important proposal is the *Web Services Resource Framework* (WSRF).

A reference implementation of the Grid architecture and Grid protocols is *Globus* providing software tools in order to build grids and Grid-based applications. These open source tools are collectively called the *Globus Toolkit* [15] – the current version is Globus 4.

Other related projects are Apple XGrid, Condor, Legion, and Sun Grid Engine.

The effort of standardization of Grid protocols, architectural models, and software tools is carried by the *Global Grid Forum* [14] and other related organizations.

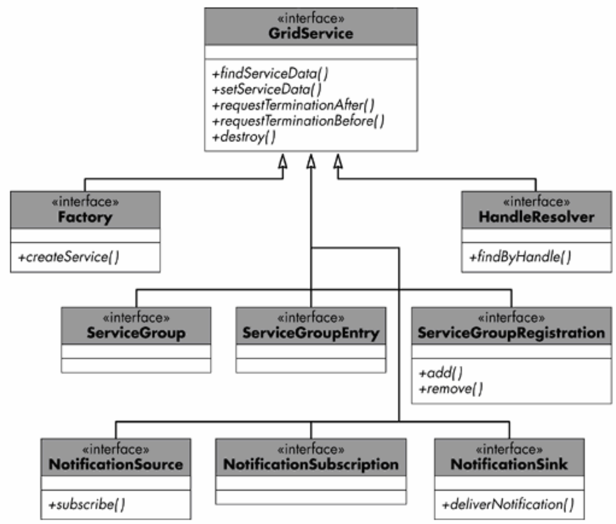

Fig. 2. OGSI interfaces [10].

III. Requirements regarding Resource Monitoring within a Grid System

From the perspective of functional requirements, the proposed application must provide an ergonomic user-interface which facilitates the monitoring of Grid resources: existing nodes (computers), average load for each node, data transfers between nodes, etc. Every status change – e.g., a file transfer is starting – must be notified. Periodically, the application must query certain nodes about the available resources.

In our case, there are two levels of resource monitoring.

By default, a given user can view only the Grid nodes and he/she is graphically informed regarding the status change. Selecting a node, the user must be able to see detailed information. Certain important properties – for example the node name and the loading percentage – are available as tooltips. Additionally, for large Grid systems, users can filter desired information to be presented – e.g., seeing only nodes which respect a given criteria: a range of IP addresses, a form of a symbolic name, etc.

On the other hand, GriW must provide means for administration of the Grid, including user management – e.g., adding, modifying, or deleting users – and access or/and adjust the system configuration.

IV. SOA-based Architecture of the Application

A. *Brief Presentation of the SOA Paradigm*

SOA (Service Oriented Architecture) refers to the design of a distributed system. SOA is an approach that leads to take concrete decisions when you design concrete software architecture. Therefore, SOA represents a design methodology, aimed at maximizing the reuse of multiple services – possibly implemented on different platforms and using multiple programming languages.

In a SOA platform, the services generally have some important characteristics [9]:

- Services are *autonomous*;
- Services must be *loosely coupled*;
- Services can be composed to provide other services;
- Services can participate in a workflow;
- Services can be easily (automatic) discovered;

In addition, services must expose information – and additional metadata (data about data) – such as their capabilities, interfaces, policies and supported protocols. Other details such as the programming language or the information about the platform are not useful for consumers and – usually – are not revealed.

Other details are provided by [11].

B. *A SOA Approach for GriW Application*

In order to provide its functionalities in a flexible manner, GriW is represented by certain independent services. To monitor the Grid resources, the WSRF (Web Service Resource Framework) approach must be considered. WSRF is a specification aligned to the SOA paradigm.

Each Grid resource is viewed as a WS-Resource – a mapping between a Web service and a stateful resource. All information within a Grid platform is managed (accessed, altered, erased, etc.) by invoking the proper Web services.

Data obtained from called Web services will be available for other applications via a GriW Web service, in an open and platform-independent manner.

Also, the monitoring activity will be provided by another Web service – named *GriWService* –, which is stateless, because do not need to store the state of each of its clients. This Java-based service effectively observes the Grid resources and publishes important information about them (including metadata).

As subservices, the following can be presented:

1. *GridClient* is responsible with the interaction to the existing Grid platform by using the Globus Toolkit API; in this context, Monitoring and Discovery Services (MDS) are invoked to obtain information about the system and its nodes. In order to have access to information regarding the available resources and their properties, the WS GRAM (Globus Resource Allocation Manager) is used. The current file transfers can be acquired via RFT (Reliable File Transfer) service. The loading percentage for each node is obtained by using SNMP (Simple Network Management Protocol) because this kind of information is not directly available at the Grid level. All information will be assembled and published as SVG (Scalable Vector Graphics) documents to be easily presented to the final users. The above mentioned services will be periodically invoked. The status changes will be routed to the specific services (which subscribed to these sorts of notifications).
2. *SvgGenerator* is a module able to serialize information provided by *GridClient* as a SVG document.
3. *GriWConfiguration* provides the operations regarding the application configuration. All configuration parameters are stored as XML files, available at a local or remote level for the users with the administrator credentials.

The user interaction is accomplished by the *WebGriW* component which provides the Web interface. This component is implemented in ASP.NET. Information obtained from GriWService is presented as a SVG document directly in the Web browser.

If the user has the administrator credentials, WebGriW offers the possibility of changing the system's configuration. The authorization mechanism uses the native methods provided by the .NET Framework.

Several reports can be generated on request or can also be sent by email.

The overall architecture of the implemented application is illustrated by the Figure 3.

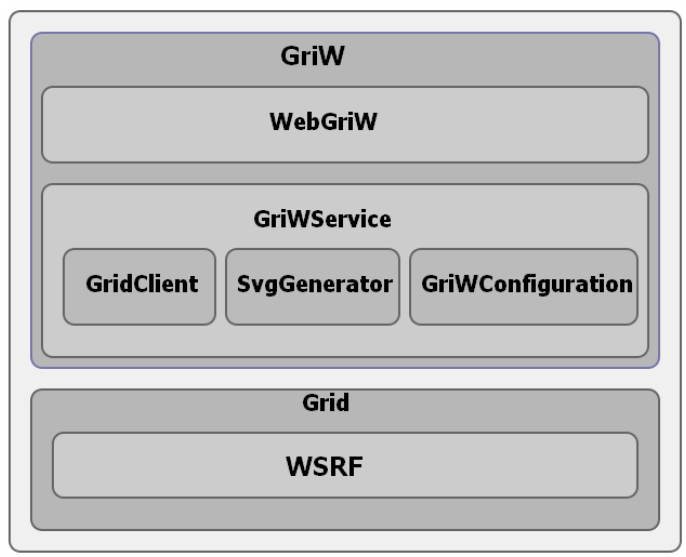

Fig. 3. GriW service-based architecture.

C. *GriW Implementation*

From the software engineering perspective, GriW uses several design patterns, such as MVC (Model-View-Controller), Observer, Proxy, and Singleton.

At the implementation level, GriWService is denoted by a collection of Java packages. All Grid services – exploited via Globus Toolkit 4 – are running into the Apache Axis container. The structure of most important classes defined by the GridClient is depicted in the Figure 4. To have access to the SNMP functionalities, the SNMP4J open-source library was adopted. In order to efficiently generate the SVG representations, Apache Batik SVG Toolkit was used.

For WebGriW, in addition to ASP.NET 2.0, the WSE (Web Services Enhancements) was installed, to effortlessly invoke remote Web services. The Web interface is partly generated by using asynchronous calls in JavaScript via AJAX (Asynchronous JavaScript And XML).

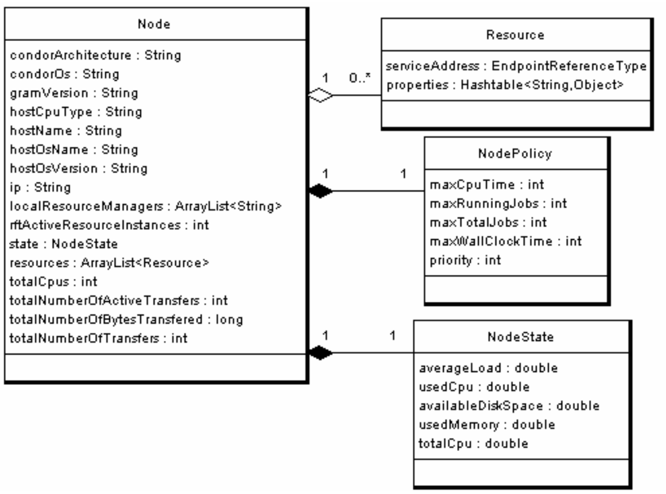

Fig. 4. Classes used for Grid-related information encapsulation.

V. CASE STUDIES

A. *Monitoring the Grid of an Academic Institution*

This case study regards the access to the information concerning the nodes of an academic Grid platform.

After authentication, the user will be able to consult the general information regarding each node of the given Grid. By default, only important date is available: hardware characteristics (number and type of processors, processing architecture, total memory, hard-disks capacity etc.) and software attributes (average load, operating system, number of services, transferred bytes) – see Figure 5.

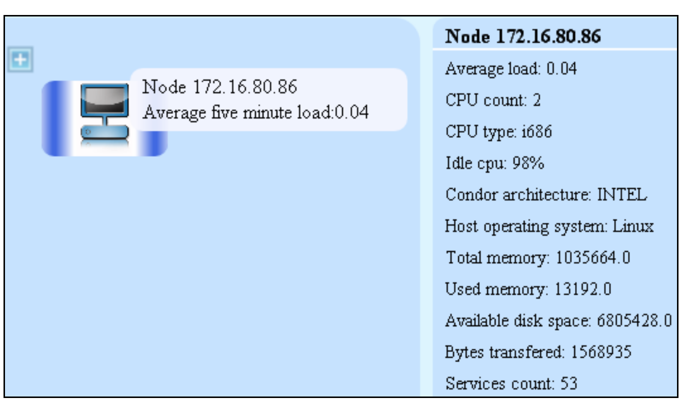

Fig. 5. Essential information regarding a given node of a Grid.

B. *Resource Monitoring in the Context of a Medical Grid*

The second case study is focused on resource monitoring issues on a Grid platform used in the context of e-health. To accomplish this goal, the Telemon system was extended to a test Grid. Telemon [2, 16] is an e-health system, intended to allow real time patient monitoring by using Web technologies. Because the Telemon architecture is aligned to the SOA paradigm, the integration of its services into a Grid platform was straightforward.

Using GriW functionalities, it is easy to notice the general status of average load at the level of the Grid nodes and the problems that could arise on a geographical area of interest (e.g., Telemon regional subsidiaries).

In the Figure 6, detailed information regarding a given node is provided, including the data transfers – in this case, patients' personal records, hospitals' availability, replicated data, and many others.

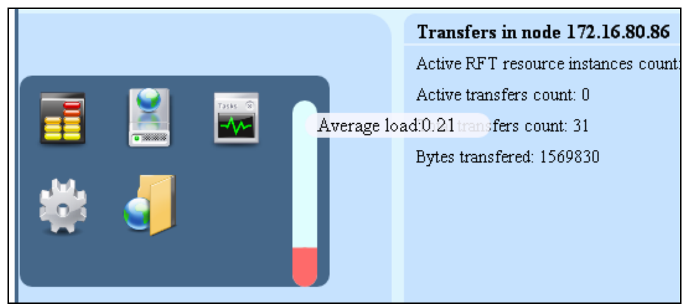

Fig. 6. Detailed information about a node of the e-health system.

To easily recognize every kind of information presented to the final user, a collection of assorted icons was designed. Empirical interaction tests we conducted prove that solution increased the ergonomics of the Web interface.

VI. CONCLUSION AND FURTHER WORK

The paper drew the main aspects regarding resource monitoring within Grid systems. After presenting a general view of Grid computing, in the section II most important functional requirements was listed.

Section III detailed the SOA-based architecture of the proposed application, called GriW, including the implemented services at the Grid level and the Web interface available for the authenticated users (administrators). Information collected about the Grid nodes and subsequent resources can be periodically updated via a notification mechanism.

Our solution is denoted by Grid services based on the Java technologies and an ASP.NET C# application in charge with the user interaction. The graphical interactive elements are denoted as SVG constructs, not as bitmap pictures.

Two case studies were also described in the section V.

Our further direction of research is to provide more detailed information for the each Grid node, including the semantic ones – following the ideas from [4] –, and to study the multiple deployment problems that can occur, including the possible technological solutions.

REFERENCES

[1] A. Abbas (Editor), Grid Computing: A Practical Guide to Technology and Applications, Charles River Media, 2004

[2] L. Alboaie, S. Buraga, V. Felea, "TELEMON – a SOA-based e-Health System. Designing the Main Architectural Components", Proceedings of the 9th International Conference on Development and Application Systems – DAS 2008, Suceava, 2008

[3] F. Berman, G. Fox, T. Hey (Editors), Grid Computing. Making the Global Infrastructure a Reality, Wiley, 2003

[4] S. Buraga, "Using the Grid and Semantic Web Technologies for Resource Management", Proceedings of the 9th International Conference on Development and Application Systems –DAS 2008, Suceava, 2008

[5] S. Buraga, L. Alboaie, "Grid Computing in the Context of Semantic Web-based Resource Management", Proceedings of the 8th International Conference on Informatics in Economy, Bucuresti, 2007

[6] S. Buraga, L. Alboaie, "A Metadata Level for the tuBiG Grid-aware Infrastructure", Proceedings of the 6th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing – SYNASC 2004, Mirton Publishing House, Timisoara, 2004

[7] S. Buraga, M. Cioca, L. Cioca, "Grid-based Decision Support System Used in Disaster Management", Studies in Informatics and Control, Number 3, 2007

[8] R. Buyya, "Economic-based Distributed Resource Management and Scheduling for Grid Computing", PhD Thesis, Monash University, Melbourne, Australia, 2002

[9] T. Erl. Service-Oriented Architecture: Concepts, Technology, and Design, Prentice Hall PTR, 2005

[10] J. Joseph, C. Fellenstein, Grid Computing, Prentice Hall Ptr., 2003

[11] N. Josuttis, SOA in Practice. The Art of Distributed System Design, O'Reilly, 2007

[12] G. Von Laszewski, P. Wagstrom, "Gestalt of the Grid", Tools and Environments for Parallel and Distributed Computing, Wiley, 2004

[13] IBM Corporation's Grid Computing Online Resources, Available: http://www.ibm.com/developerworks/grid

[14] Global Grid Forum, Available: http://www.globalgridforum.org/

[15] Globus, Available: http://www.globus.org/

[16] Telemon Project, Available: http://profs.info.uaic.ro/˜telemonfcs/