International Journal of Research in Advent Technology, Vol.2, No.2, February 2014
E-ISSN: 2321-9637

The Role of Grid Computing Technologies In Cloud Computing

Prof. R.T Nakhate, Prof. G. D Korde
Department of Information Technology, Department of Electronics & Telecommunication,
Nagpur University,DMIETR salod, Wardha, Nagpur University,BDCOE, SevagramWardha.
rajesh.nakhate@gmail.com, gdkorde@yahoo.com@yahoo.com

Abstract: The fields of Grid, Utility and Cloud Computing have a set of common objectives in harnessing shared resources to optimally meet a great variety of demands cost-effectively and in a timely manner Since Grid Computing started its technological journey about a decade earlier than Cloud Computing, the Cloud can benefit from the technologies and experience of the Grid in building an infrastructure for distributed computing. Our comparison of Grid and Cloud starts with their basic characteristics and interaction models with clients, resource consumers and providers. Then the similarities and differences in architectural layers and key usage patterns are examined. This is followed by an in depth look at the technologies and best practices that have applicability from Grid to Cloud computing, including scheduling, service orientation, security, data management, monitoring, interoperability, simulation and autonomic support. Finally, we offer insights on how the techniques will help solve the current challenges faced by Cloud computing.

I. INTRODUCTION

Cloud computing exploits the advances in computing hardware and programmingmodels that can be brought together to provide utility solutions to large-scale computing problems. At the hardware level, the last half century has seen prolific progress in computing power. This results from many improvements at the processorlevel, and in recent years the availability of low cost multi-core circuits. Additional progress in high speed, low latency interconnects, has allowed building large-scale local clusters for distributed computing, and the extension to wide-area collaborating clusters in the Grid. Now, the recent availability of hardware support for platform virtualization on commodity machines provides a key enabler for Cloud based computing. Softwaremodelsmove in lockstep to match advances in hardware. There is a considerable practical experience implementing distributed computing solutions and in supporting parallel programming models on clusters. These models now work to leverage the concurrency provided by multi-core and multi-systems. Additionally, there are two other areas of software evolution that are moving quickly to support the Cloud paradigm: one is the improving maturity and capability of software to manage virtual machines, and the other is the migration from a monolithic approach in constructing software solutions to a service approach in which complex processes are composed of loosely coupled components.

II. BASICS OF GRID AND CLOUD COMPUTING

1. Basics of Grid Computing

Grid computing harnesses distributed resources from various institutions (resourceproviders), to meet the demands of clients consuming them. Resources from different providers are likely to be diverse and heterogeneous in their functions (computing, Storage, software, etc.), hardware architectures (Intel x86, IBM PowerPC, etc.),and usage policies set by owning institutions. Developed under the umbrella of Grid Computing, information services, name services, and resource brokering services are important technologies responsible for the aggregation of resource information and availability, selection of resources to meet the clients' specific requirements and the quality of services criteria while adhering to the resource usage policies. Figure showsan exemplary relationship of resource providers and consumers for a collaborative Grid computing scenario. Clients or users submit their Requests for application execution along with resource requirements from their home domains. A Resource broker selects a domain with appropriate resources to acquire from and to execute the application or route the application to domain for execution with results and status returning to the home domain.

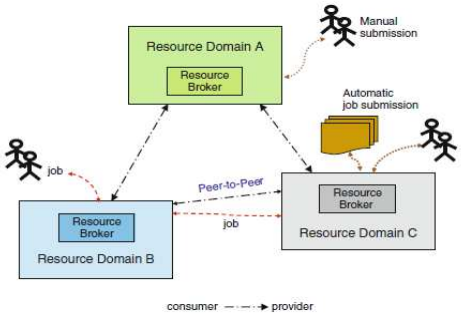

Fig.Grid collaborating domains

2. Basics of Cloud Computing

Internet Data Centers IDC[6]defined two specific aspects of Clouds: Cloud Services and Cloud Computing.Cloud Services are "consumer and business products, services and solutions that are delivered and consumed in real-time over the Internet" while Cloud Computing is
"an emerging IT development, deployment and delivery model, enabling real-timedelivery of products, services and solutions over the Internet (i.e., enabling Cloud services)". In this chapter, we will focus the computing infrastructure and platform aspects of the Cloud. Amazon's Elastic Compute Cloud2 popularized the Cloud computing model by providing an on-demand provisioning of virtualized computational resources as metered services to clients or users. While not restricted, most of the clients are individual users that acquire necessary resources for their own usage through EC2's APIs without cross organization agreements or contracts. Figure illustratespossible usage models from clients C1 and C2 for resources/services of Cloud providers. As Cloud models evolve, many are developing the hybrid Cloud model in which enterprise resource brokers may acquire additional needed resources from external Cloud providers to meet the demands of submitted enterprise workloads (E1) and client work requests (E2). Moreover, the enterprise resource domain and Cloud providers may all belong to one corporation and thus form a private Cloud model.

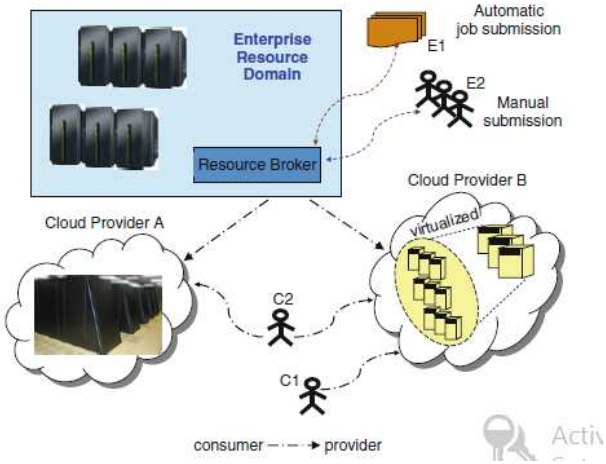

Fig Cloud usage models

3. Interaction Models of Grid and Cloud Computing

One of the most scalable interaction models of Grid domains is peer-to-peer,where most of the Grid participating organizations are both consumers and providers. In practice, there are usually agreements of resource sharing among thepeers. Furthermore, clients of consumer organizations in Grids use heterogeneous resources from more than one resource provider belonging to the same Virtual Organization (VO) to execute their applications. It is important for participating resource providers and consumers to have common information models, interaction protocols, application execution states, etc. The organization of Open Grid Forum (OGF)3has the goal of establishing relevant and necessary standards for Grid computing. Some proposed standards include Job Submission Description Language (JSDL), Basic Execution Service (BES) and others. Currently, most of the Cloud providers offer their own proprietary service protocols and information formats. As Cloud computing becomes mature and widely adopted, clients and consumer organizations would likely interact with more than one provider for various reasons, including finding the most cost effective solutions or acquiring a variety of services from different providers (e.g., compute providers or data providers). Cloud consumers will likely demand common protocols and standardized information formats for ease of federated usage and interoperability.

4. Distributed Computing in the Grid and Cloud

The Grid encompasses two areas of distributed system activity. One is operational with an objective

of administrating and managing an interoperable collection of distributed compute resource clusters on which to execute client jobs, typically scientific/HPC applications. The procedures and protocols required to support clients from complex services built on distributed components that handle job submission, security, machine provisioning, and data staging. The Cloud has similar operational requirements for supporting complex services to provide clients with services on different levels of support such application, platform and infrastructure. The Grid also represents as a coherent entity a collection of compute resources that may be under different administrative domains, such as universities, but inter-operate transparently to form virtual organizations. Although interoperability is not a near term priority, there is a precedent for commercial Clouds to move in this direction similarly to how utilities such as power or communication contract with their competitors to provide overflow capacity.

III. LAYERED MODELS AND USAGE PATTERNS IN GRID AND CLOUD

There are many similarities in Grid and Cloud computing systems. We compare the approaches by differentiating three layers of abstraction in Grid: Infrastructure, Platform and Application. Then we map these three layers to the Cloud services of IaaS, PaaS, and SaaS. An example of the relations among layers can be seen in Fig.

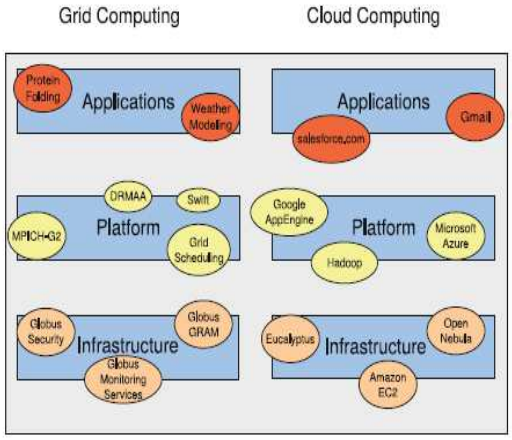

Fig.Grid and cloud layers

1. Infrastructure

This is the layer in which Clouds share most characteristics with the original purposeof Grid middleware. Some examples are Eucalyptus (Nurmi et al., 2009),OpenNebula,or Amazon EC2. In these systems users can provision executionenvironments in the form of virtual machines through interfaces such as APIs or command line tools. The act of defining an execution environment and sending a request to the final resource has many similarities with scheduling a job in the Grid. The main steps, shared by all of the cited Cloud environments are discussed below. We use Globus as the reference Grid technology.

2. Platform

This layer is built on top of the physical infrastructure and offers a higher level of abstraction to users. The interface provided by a PaaS solution allows developers to build additional services without being exposed to the underlying physical or virtual resources. These facts enable additional features to be implemented as part of the model, such as presenting seemingly infinite resources to the user or allowing Elastic behavior on demand. Examples of Cloud solutions that present these features are Google App Engine,8Salesforce's force.com9 or Microsoft Azure.

1. Abstraction from Physical Resources

The Infrastructure layer provides users with direct access to the underlying infrastructure. While this is required for the lower levels of resource interaction, in the Platform level a user should be isolated from them. This allows developers to create new software that is not susceptible to the number of provisioned machines or their network configuration, for example.

2. Programming API to Support New Services

The Platform layer allows developers to build new software that takes advantage of the available resources. The choice of API directly influences the programs that can be built on the Cloud, therefore each PaaS solution is usually designed with a type of application in mind. With these characteristics Grid systems allow developers to produce new software that take advantage of the shared resources in order to compare them with PaaS solutions.

3. Applications

There is no clear distinctions between applications developed on Grids and those that use Clouds to perform execution and storage. The choice of platform should not influence the final result, since the computations delegated to the underlying systems can take different shapes to accommodate to the available APIs and resources. On the other hand, it is undeniable that the vast majority of Grid applications

fall in the realm of scientific software, while software running in Clouds has leaned towards commercial workloads. Here we try to identify some possible causes for the different levels of adoption of these technologies for the development of applications:

• **Lack of business opportunities in Grids.**

Usually Grid middleware is installed only in hardware intended for scientific usage. This phenomenon has not successfully produced business opportunities that could be exploited by industry. Conversely, Clouds are usually backed up by industry which have had better ways to monetize their investments.

• **Complexity of Grid tools.**
Perhaps due to the goal of providing a standardized, one-size-fits-all solution, Grid middleware is perceived by many as complex and difficult to install and manage. On the other hand, Cloud infrastructures have Usuallybeen developed by providers to fit their organization's needs and with a concrete purpose in mind, making them easier to use and solution oriented.
• **Affinity with target software**.

Most Grid software is developed with scientific Applications in mind, which is not true for the majority of Cloud systems.Scientific programs need to get the most performance from execution resources and many of them cannot be run on Clouds efficiently, for example because of virtualization overhead. Clouds are more targeted to web applications. These different affinities to distinct paradigms make both solutions specially effective for their target applications.

IV. TECHNIQUES

Here we discuss the impact of techniques used in Grid computing that can be applied in Clouds. From the time the concept of Grid was introduced, a variety of problems had to be solved in order to enable its wide adoption. Some examples of these areuser interfacing, data transfer, resource monitoring or security . These basic techniques for the enablement of Grids were designed to fulfill its main goals, namely, to allow the sharing of heterogeneous resources among individuals belonging to remote administrative domains. These goals determine the areas of application of the described techniques in Clouds, therefore we will find the most valuable set of improvements to be in the field of Cloud interoperability.

1. Service Orientation and Web Services

The Cloud is both a provider of services (e.g. IaaS,PaaS, and SaaS) and a place to host services on behalf of clients. To implement the former operational aspects while maintaining flexibility, Cloud administrative functions should be constructed from software components. The Grid faced similar challenges in building a distributed infrastructure to support and evolve its administrative functions such as security, job submission, and creation of Virtual Organizations. The architectural principle adopted by the Grid is Service Orientation (SO) with software components connected by Web Services (WS). This section summarizes contributions of the Open Grid Forum (OGF) to SO in distributed computing and and how they apply to the Cloud. SO as an architecture, andWeb Services as a mechanism of inter-component communication are explored here in the context of similarities between Grid and Cloud requirements.

2. Data Management

In Grid computing, data-intensive applications such as the scientific software in domains like high energy physics, bio-informatics, astronomy or earth sciences involve large amounts of data, sometimes in the scale of PetaBytes (PB) and beyond(Moore, Prince, &Ellisman, 1998).Data management techniques to discover and access information are essential for this kind of applications. Network bandwidth, transfer latency and storage resources are as important as computational resources to determine the tasks' latency and performance. For example, a data-intensive application will preferably be run at a site that has an ample and fast network channel to its dataset so that the network overhead can be reduced, and if it generates a large amount of data, we would also prefer a site that has enough storage space close to it.

3. Monitoring

Although some Cloud monitoring tools have already been developed, they provide high level information and, in most cases, the monitoring functionality is embedded in the VMmanagement system following specific mechanisms and models. The current challenge for Cloud monitoring tools is providing information from the Clouds and application/service requests with sufficient level of detail in nearly real time in order to take effective decisions rather than providing a simple and graphical representation of the Cloud status. To do this, different Grid monitoring technologies can be applied to Clouds, especially those of them that are capable to provide monitoring data in aggregate form due to the large scale and dynamic behavior of Clouds.

Several data centers that provide resources to Cloud systems have adoptedGanglia as a monitoring tool. However, virtualized environments have more specific needs that have motivated Cloud computing technology providers to develop their own monitoring system. Some of them are summarized below:

Amazon Cloud Watch[1]is a web service that provides monitoring for Amazon Web Services Cloud resources such as Amazon EC2. It collects raw data from AmazonWeb Services and then processes the information into readable metrics that are recorded for a period of two weeks. It provides the users with visibility into resource utilization, operational performance, and overall demand patterns - including metrics such as CPU utilization, disk reads and writes, and network traffic.

Windows Azure Diagnostic Monitor[2]collects data in local storage for every
diagnostic type that is enabled and can transfer the data it gathers to an Azure Storage account for permanent storage. It can be scheduled to push the collected data to storage at regular intervals or it can be requested an on-demand transfer whenever this information is required.

The **OpenNebula Information Manager** (IM) is in charge of monitoring the different nodes in a Cloud. It comes with various sensors, each one responsible
for different aspects of the compute resource to be monitored (CPU, memory, hostname). Also, there are sensors prepared to gather information from different hypervisors.

The monitoring functionality of **Aneka** (Vecchiola, Chu, &Buyya, 2009) is implemented by the core middleware, which provides a wide set of services
including also negotiation of the quality of service, admission control, execution management, accounting and billing. To help administrators to
tune the overall performance of the Cloud, the Management Studio provides aggregated dynamic statistics.

Nimsoft Monitoring Solution[3](NMS), built on the Nimsoft Unified Monitoring Architecture, delivers monitoring functionality to any combination of virtualized data center, on hosted or managed infrastructure, in the Cloud on IaaS or PaaS or delivered as SaaS services. Specifically, it provides
unified monitoring for data centers, private Clouds and public Clouds such as Amazon WS, including service level and response time monitoring, visualization and reporting.

HypericCloudStatus[4]provides open source monitoring and management software for all types of web applications, whether hosted in the Cloud or on premise, including Amazon Web Services and Google App Engine. Cloud Status gives users real-time reports and weekly trends on infrastructure
metrics.

4. Autonomic Computing

Inspired by the autonomic nervous system, autonomic computing aims at designing and building self-managing systems and has emerged as a promising approach for addressing the challenges due to softwarecomplexity .An autonomic system is able to make decisions to respond to changes in operating condition at runtime using high-level policies that are typically provided by an expert. Such a system constantly monitors and optimizes its operation and automatically adapts itself to changing conditions so that it continues to achieve its objectives. There are several important and valuable milestones to reach fully autonomic computing: first, automated functions will merely collect and aggregate information to support decisions by human users. Later, they will serve as advisors, suggesting possible courses of action for humans to consider. Self-management is the essence of autonomic computing and has been defined in terms of the following four aspects of self-management (Jeffrey &Kephart, 2001).

- **Self configuration:** Autonomic systems will configure themselves automatically in accordance with high-level policies representing business-level objectives that, for example, specify what is desired and not how it is to be accomplished. When a component is introduced, it will incorporate itself seamlessly, and the rest of the system will adapt to its presence.
- **Self optimization:** Autonomic systems will continually seek ways to improve their operation, identifying and seizing opportunities to make themselves more efficient in performance and/or cost. Autonomic systems will monitor, experiment with, and tune their own parameters and will learn to make appropriate choices about keeping functions or outsourcing them.
- **Self-healing:** Autonomic computing systems will detect, diagnose, and repair localized problems resulting from bugs or failures in software and hardware.
- **Self-protection:** Autonomic systems will be self-protecting in two senses. They will defend the system as a whole against large-scale, correlated problems arising from malicious attacks or cascading failures that remain uncorrected by self-healing measures. They also will anticipate problems based on early reports from sensors and take steps to avoid or mitigate them.

5. Scheduling, Meta-scheduling, and Resource Provisioning

In the last few decades a lot of effort has been devoted to the research of job scheduling, especially in centers with High Performance Computing (HPC) facilities. The general scheduling problem consists of, given a set of jobs and requirements, a setof resources, and the system status, deciding which jobs to start executing and in which resources.

In Grid computing, scheduling techniques have evolved to incorporate otherfactors, such as the heterogeneity of resources or geographical distribution. The software component responsible for scheduling tasks in Grids is usually called meta-scheduler or Grid resource broker. The main actions that are performed by a Grid resource broker are: resource discovery and monitoring, resource selection, job execution, handling and monitoring. However, it may be also responsible for other additional tasks such as security mechanisms, accounting, quality of service (QoS) ensuring, advance reservations, negotiation with other scheduling entities, policy enforcement, migration, etc. A taxonomy and survey of Grid brokering systems can be found in (Krauter, Buyya, &Maheswaran, 2002).

Some of their mostcommon characteristics are discussed as follows:

• They can involve different scheduling layers through several software components between the Grid resource broker and the resources where the application will run. Thus, the information and control available at the resource broker level is far less than that available at a cluster scheduling level.

• A Grid resource broker usually does not have ownership or control over the resources. Moreover, the cluster scheduling systems may have their own local policies that can conflict with the Grid scheduling strategy.

• There are conflicting performance goals between the users and the resource owners. While the users focus on optimizing the performance of a single application for a specified cost goal, the resource owners aim to obtain the best system throughput or minimize the response time.

6. Interoperability in Grids and Clouds

One goal of Grid computing is to provide uniform and consistent access to resources distributed in different data centers and institutions. This is because the majority of Grids are formed based on regional as opposed to local initiatives so interoperation is a key objective. interoperability using a uniform access interface to application users. Software layers beneath the user interface then abstract the complexity of the underlying heterogeneous supercomputing infrastructures. One tool that takes this approach for Grid interoperation is meta-brokering (Kertesz&Kacsuk, 2008), illustrated in Figure Meta-brokering supports the Grid interoperability from the viewpoint of the resource management and scheduling. Many projects explore this approach with varied emphases. Examples grouped loosely by primary technical foci, are reviewed below.

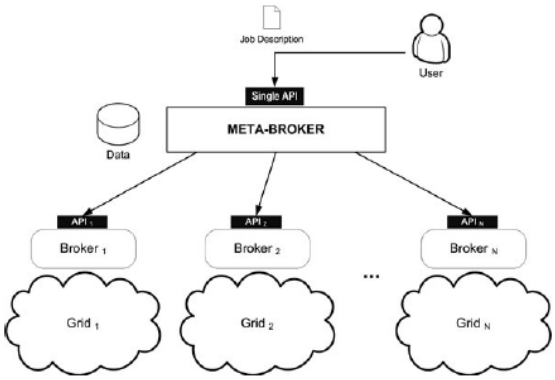

Fig.Meta-brokering architecture

7. Security and User Management

Clouds currently lack many of the mechanisms required for fluid inter site operation of which security is a key enabling factor. Interoperability mandates common security mechanisms that can be translated to the models chosen by local administrators. Additionally, users need to be able to submit requests regardless of the institutions involved in the process of performing the requested task or providing the necessary data. This requires introduction to the Cloud of a mechanism of privilege delegation enabling single sign-on. Finally, collaboration between multiple institutions sharing resources requires development of new methods to manage user privileges. These challenges have already been addressed in Grid computing where a primary goal is to allow sharing of resources among Virtual Organizations. A VO defines a group of people and resources that can spawn across multiple administrative domains, and allows the definition of fine grained security policies on those resources. The Grid solutions are described in the context of Globus middleware and show how the concepts can be applied to Clouds.

8. Modeling and Simulation of Clouds and Grids

It is difficult or even not feasible to evaluate different usages on real Grid testbeds, different simulators have been developed in order to study complex scenarios. Simulations allow us to research policies for large and complex configurations with numerous jobs and high demand of resources and to easily include modifications and refinements in the policies. There are many rich simulation models developed by the Grid community

V. CONCLUDING REMARKS

Grids and Clouds have many similarities in their architectures, technologies and techniques. Nowadays, it seems Cloud computing is taking more significance as a means to offer an elastic platform to access remote processing resources: this is backed up by the blooming market interest on new platforms, the number of new businesses that use and provide Cloud services and the interest of academia in this new paradigm. However, there are still multiple facets of Cloud computing that need to be addressed, such as vendor lock-in, security concerns, better monitoring systems, etc. We believe that the technologies developed in Grid computing can be leverage to accelerate the maturity of the Cloud, and the new opportunities presented by the latter will in term address some of the shortcomings of the Grid.

VI. REFERENCES

1. http://aws.amazon.com/cloudwatch/
2. http://www.microsoft.com/windowsazure
3. http://www.nimsoft.com/solutions/
4. http://www.hyperic.com/
5. http://blogs.idc.com/ie/?p=190
6. Jens, F. (September 2008). Defining cloud services and cloud computing.
7. Agarwal, A., Ahmed, M., Berman, A., Caron, B. L., Charbonneau, A., Deatrich, D., et al. (2007).
8. GridX1: A Canadian computational grid. Future Generation Computer Systems, 23, 680–687.
9. Alessandrini, V., &Niederberger, R. (2004). The deisa project: Motivations, strategies, technologies.
10. 19th International Supercomputer Conference, Heidelberg, Germany.
11. Allcock, B., Bester, J., Bresnahan, J., Chervenak, A. L., Foster, I., Kesselman, C., et al. (2001).
12. Secure, efficient data transport and replica management for high-performance data-intensivecomputing. In Processings of IEEE Mass Storage Conference, IEEE Press, San Diego, CA,USA.
13. Allcock, W., (2003). Gridftp protocol specification. Global grid forum recommendation GFD.20(Tech. Rep., Open Grid Forum (OGF)).
14. Anjomshoaa, A., Drescher, M., et al. (2005). Job submission description language (JSDL)specification version 1.0, GFD-R.056 (Tech. Rep., Open Grid Forum (OGF)).
15. Antonioletti, M., Krause, A., & Paton, N. W. (2005). An outline of the global grid forum dataaccess and integration service specifications. Data Management in Grids LNCS, 3836, 71–84.
16. Appleby, K., Fakhouri, S., Fong, L., Goldszmidt, G., Kalandar, M., Krishnakumar, S., et al.
17. (2001). Oceano-sla based management of a computing utility. Proceeding of the 7th IFIP/IEEEInternational Symposium on Integrated Network Management (IM 2001), Seattle, WA.
18. Armbrust, M., Fox, A., & Griffith, R., et al. (2009). Above the clouds: A berkeley view ofcloud computing (CoRR UCB/EECS-2009-28, EECS Department, University of California,Berkeley)