National Conference on Computer Security, Image Processing, Graphics, Mobility and Analytics (NCCSIGMA)
*International Journal of Advanced Engineering Research and Science (IJAERS)* *Special Issue (NCCSIGMA-16)*
https://dx.doi.org/10.22161/ijaers/si.2 *ISSN: 2349-6495(P) | 2456-1908(O)*

# Simple API for Grid Applications and IDEAS Design for Several Cloud Applications

Md.Rafeeq[1], Dr.C. Sunil Kumar[2], Dr.N. Subhash Chandra[3]

[1]Associate Professor in CSE, CMR Technical Campus-Hyd,TS,India.
[2]Professor in IT -SNIST-Ghatkesar,Hyd,TS,India.
[3]Principal of HITS-Ghatkesar,Hyd,TS,India.

***Abstract*— *Distributed systems and their specific incarnations have evolved significantly over the years. Most often, these evolutionary steps have been a consequence of external technology trends, such as the significant increase in network/bandwidth capabilities that have occurred. It can be argued that the single most important driver for cloud computing environments is the advance in visualization technology that has taken place. We view clouds not as a monolithic isolated platform but as part of a large distributed ecosystem. Prima facie, cloud concepts are derived from other systems, such as the implicit model of clusters as static bounded sets of resources, which leads to batch-queue extensions to virtualization [10]. IDEAS prevalent in grids to address dynamic application requirements and resource capabilities, such as pilot jobs, that are being redesigned and modified for clouds. In either case, clouds are an outgrowth of the systems and ideas that have come before them, and we want to consciously consider our underlying assumptions, to make sure we are not blindly carrying over assumptions about previous types of parallel and distributed computing.*

***Keywords*—*Resource Management, Scale-Out And Scale-Up, HPC And HTC, Map Reduce.***

## I. INTRODUCTION

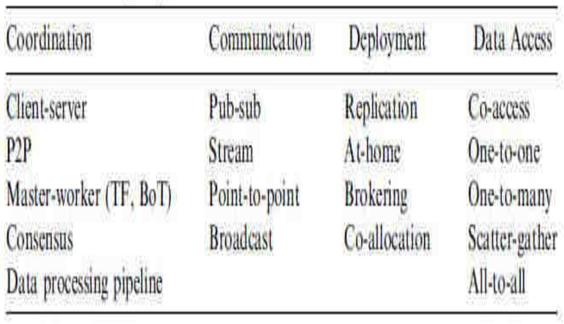
**The IDEAS design objectives— Interoperability, Distributed scale-out, Extensibility, Adaptivity , and Simplicity—** Clouds will have a broad impact on legacy scientific applications, because we anticipate that many existing legacy applications will adapt to and take advantage of new capabilities.

However, it is unclear if clouds as currently presented are likely to change (many of) the fundamental reformulation of the development of scientific applications. We believe that there is novelty in the resource management and capacity planning capabilities for clouds. Thanks to their ability to provide an illusion of unlimited and/or immediately available resources, as currently provisioned, clouds in conjunction with traditional HPC and HTC grids provide a balanced infrastructure supporting scale-out and scale-up,as well as capability (HPC) and quick turn-around (HTC) computing for a range of application (model) sizes and requirements. he novelty in resource management and capacity planning capabilities is likely to influence changes in the usage mode, as well deployment and execution management/planning. The ability to exploit these attributes could lead to applications with new and interesting usage modes and dynamic execution on clouds and therefore new

## II. SCIENTIFIC CLOUD APPLICATIONS AS DISTRIBUTED APPLICATIONS

We determined that understanding the execution units, communication requirements, coordination mechanisms, and execution environment of a distributed application was a necessary (minimally complete) set of requirements. We will argue that both the
vectors and the abstractions (patterns) for cloud-based applications are essentially the same as those for grid-based applications, further lending credibility to the claim that cloud-based applications are of the broader distributed applications class.

*Table.1: Some commonly occurring patterns In distributed computing.*



| Coordination | Communication | Deployment | Data Access |
|---|---|---|---|
| Client-server | Pub-sub | Replication | Co-access |
| P2P | Stream | At-home | One-to-one |
| Master-worker (TF, BoT) | Point-to-point | Brokering | One-to-many |
| Consensus | Broadcast | Co-allocation | Scatter-gather |
| Data processing pipeline | | | All-to-all |

Cloud capabilities will enable applications to exploit new scenarios, for example, the dynamic adjustment of application parameters (such as the accuracy) or the dynamic addition of new resources to an application. SAGA, which is an API for distributed applications as a viable programming system for clouds. We establish this with three distinct applications that have been developed for clouds using SAGA, further bolstering the connection between cloud applications and distributed applications.

**CLASSIFICATION OF SCIENTIFIC APPLICATIONS AND SERVICES IN THE CLOUD:** The services of each layer can be composed from the services of the layer underneath, and each layer may include one or more services that share the same or equivalent levels of abstraction. The proposed layers consist of the following: the Software as a Service (SaaS) layer, the platform as a service (PaaS) layer, and the Infrastructure as a Service (IaaS) layer. The IaaS layer can be further divided into the computational resources, storage, and communications sub layers, the software kernel layer, and the hardware/firmware layer that consists of the actual physical system components. clouds can also be classified according to their deployment model into public and private clouds. A public cloud is generally available on pay-per-use basis. Several infrastructures have emerged that enable the creation of so-called private clouds—that is, clouds that are only accessible from within an organization.

Clouds provide services at different levels (IaaS, PaaS, SaaS). The amount of control available to users and developers decreases with the level of abstraction. According to their deployment model, clouds can be categorized into public and private clouds.

Although our taxonomy is targeted toward specific cloud environments, we strongly believe that a scientific application should and must remain interoperable regardless of the execution backend or the initial development infrastructure.

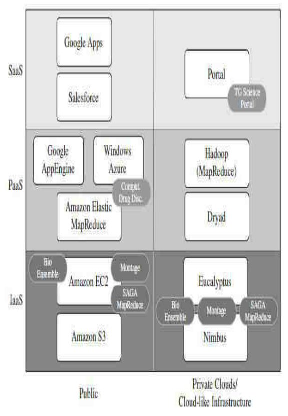


*Fig.1: Cloud Taxonomy and Application examples:*

The identification of how cloud application services fit into the layers may allow software developers to better comprehend the nature of parameters introduced in each layer. Such an assumption could lead into easier and more efficient implementation of cloud-operable scientific applications. Research work from the traditional cluster/grid era systems has already determined important features like scalability, extensibility, and high availability that should play an integral role in a distributed application's core functionality.

Several scientific applications from different domains (e.g., life sciences, high energy physics, astrophysics, computational chemistry) have been ported to cloud environments The majority of these applications rely on IaaS cloud services and solely utilize static execution modes: A scientist leases some virtual resources in order to deploy their testing services. One may select different number of instances to run their tests on. An instance of aVM is perceived as a node or a processing unit. There can be a multiple number of instances under the same VM, depending on the SLA one has agreed on. Once the service is deployed, a scientist can begin testing on the virtual nodes; this is similar to how one would use a traditional set of local clusters.

**III. SAGA-BASEDSCIENTIFIC CLOUDS:** SAGA is a Simple API for Grid Applications (SAGA) [9] provides a means to implement first-principle distributed applications. Both the SAGA standard [3] and the various SAGA implementations [1,2] ultimately strive to provide higher-level programming abstractions to developers, while at the same time shielding them from the heterogeneity and dynamics of the underlying infrastructure. The low-level decomposition of distributed applications can thus be expressed via the relatively high-level SAGA API.

SAGA has been used to develop scientific applications that can utilize an ever-increasing set of infrastructure, ranging from vanilla clouds such as EC2, to "open source" clouds based upon Eucalyptus, to regular HPC and HTC grids, as well to a proposed set of emerging "special-purpose" clouds. SAGA has also been used in conjunction with multiple VM management systems such as OpenNebula (work in progress) and Condor (established). The resulting applications are fit to utilize cloud environments. In other words, if clouds can be defined as elastic distributed systems that support specific usage modes, then it seems viable to expect explicit application level support for those usage modes, in order to allow applications to express that usage mode in the first place. The SAGA layer allows to abstract the specific way in which that usage mode is provided—either implicitly by adding additional structure to the distributed environment, or explicitly by exploiting support for that usage mode, for example, the elasticity in a specific cloud. This section will discuss several SAGA-based scientific cloud applications, but we assert that the discussion holds just as well for applications that express their decomposition in other ways programmatically. We do not claim that SAGA is the ultimate approach to develop cloud applications, but given our experience so far, it at least seems to be a viable approach that allows applications to directly benefit from the features that clouds, as specific distributed environments, provide: (a) support for specific usage modes and (b) elasticity of resources. Below we will present a number of examples that illustrate and verify that approach.

## IV. MAPREDUCE FRAMEWORK

MapReduce (MR) is a prominent example for a PaaS: The MR framework allows users to (a) define their own specific map and reduce algorithms and (b) utilize the respective PaaS infrastructure with its MR supporting usage modes (elasticity, communication, etc.). The SAGA apReduce [3] provides a MR development and runtime environment that is implemented using the SAGA. The main advantage of a SAGA-based approach is that it is infrastructure-independent while still providing a maximum of control over the deployment, distribution, and runtime decomposition. In particular, the ability to control the distribution and placement of the computation units (workers) is critical in order to implement the ability to move computational work to the data.

A master-worker paradigm is used to implement the MapReduce pattern. The diagram shows several different infrastructure options that can be utilized by the application.

Figure 2. show the architecture of the SAGA MR framework. Several SAGA adaptors have been developed to utilize SAGA MapReduce seamlessly on different grid and cloud infrastructures [8] For this purpose, adaptors for the SAGA job and file package are provided. The SAGA job API is used to orchestrate mapping and reduction tasks, while the file API is utilized to access data. In addition to the local adaptors for testing, we use the Globus adaptors for grids and the AWS adaptors for cloud environments. Furthermore, we provide various adaptors for cloud-like infrastructure, such as different open source distributed file systems (e.g., HDFS [5] and CloudStore [4]), and key/value stores (e.g., HBase [3]).The SAGA-based MapReduce implementation has shown to be easily applicable to sequence search applications, which in turn can make excellent use of the MapReduce algorithm and of a variety of middleware backends.

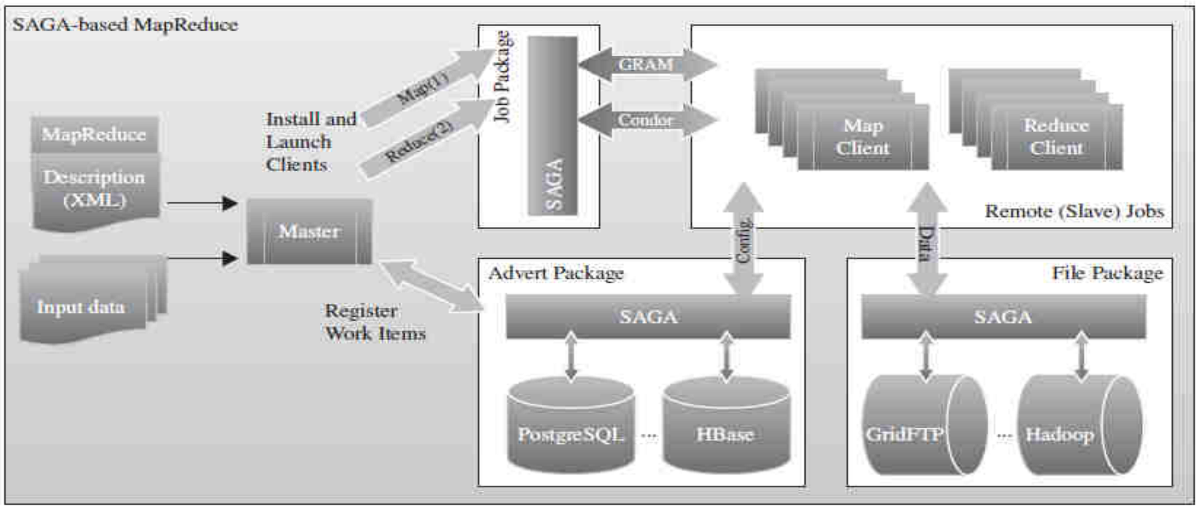


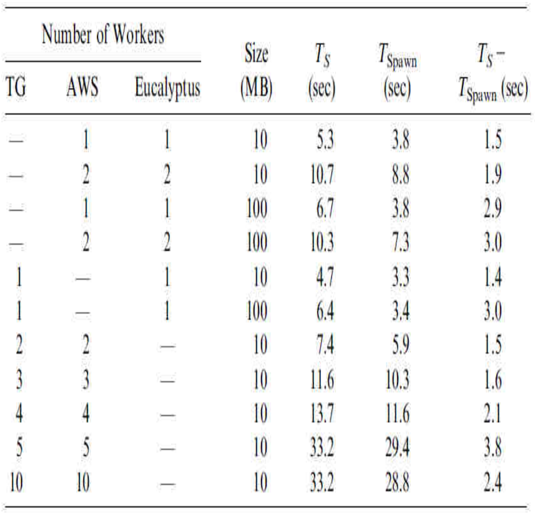
*Fig.2: SAGA MapReduce framework: The diagram shows several different infrastructure options that can be utilized by the application.*

**SAGA montage:**

Montage [6] an astronomical image mosaicking application that is also one of the most commonly studied workflow applications, has also been studied [8] with SAGA. Montage is designed to take multiple astronomical images (from telescopes or other instruments) and stitch them together into a mosaic that appears to be from a single instrument. Montage initially focused on being scientifically accurate and useful to astronomers, without being concerned about computational efficiency, and it is being used by many production science instruments and astronomy projects [9]. Montage was envisioned to be customizable, so that different astronomers could choose to use all, much, or some of the functionality, and so that they could add their own code if so desired.

A Montage run is a set of tasks, each having input and output data, and many of the tasks are the same executable run on different data, referred to as a stage. The generality of Montage as a workflow application has led it to become an exemplar for those in the computer science workflow community, such as those working on: Pegasus, ASKALON [10], quality-of-service (QoS)-enabled GridFTP [45], SWIFT [46], SCALEA-G [47], VGrADS [48], and so on.

*Table.2: Performance Data for Different Configurations of Worker Placements onTG, Eucalyptus_Cloud, and EC2.*



| Number of Workers | | | Size | $T_S$ | $T_{Spawn}$ | $T_S -$ |
|---|---|---|---|---|---|---|
| TG | AWS | Eucalyptus | (MB) | (sec) | (sec) | $T_{Spawn}$ (sec) |
| – | 1 | 1 | 10 | 5.3 | 3.8 | 1.5 |
| – | 2 | 2 | 10 | 10.7 | 8.8 | 1.9 |
| – | 1 | 1 | 100 | 6.7 | 3.8 | 2.9 |
| – | 2 | 2 | 100 | 10.3 | 7.3 | 3.0 |
| 1 | – | 1 | 10 | 4.7 | 3.3 | 1.4 |
| 1 | – | 1 | 100 | 6.4 | 3.4 | 3.0 |
| 2 | 2 | – | 10 | 7.4 | 5.9 | 1.5 |
| 3 | 3 | – | 10 | 11.6 | 10.3 | 1.6 |
| 4 | 4 | – | 10 | 13.7 | 11.6 | 2.1 |
| 5 | 5 | – | 10 | 33.2 | 29.4 | 3.8 |
| 10 | 10 | – | 10 | 33.2 | 28.8 | 2.4 |

## V. IDEAS REVISITED

SAGA and SAGA-based abstractions help to advance the IDEAS design objectives: Interoperability, Distributed scale-out, Extensibilty, Adaptivity and Simplicity:

**Interoperability of Scientific Applications across Clouds and HPC/Grids**

The SAGA programming system provides a standard interface and can support powerful programming models. SAGA allows application developers to implement common and basic distributed functionality, such as application decomposition, distributed job submission, and distributed file movement/ management, independently of the underlying infrastructure. The SAGA cloud adaptors provide the foundation for accessing cloud storage and compute resource via the SAGA API. The ability to design and develop applications in an infrastructure-independent way leads to new kinds of application, such as dynamic applications. Such applications have dynamic runtime requirements and are able to adapt to changing runtime environments and resource availabilities.

SAGA provides developers with new capability while introducing a new set of challenges and trade-offs. There has been a lot of testing on simple applications performing map and reduce computations on VMs as well as on traditional local clusters in order to first verify the scalability of performance that the proposed model successfully offers and then, most importantly, guarantee interoperability between VMs and local clusters for a given application. As shown, SAGA MapReduce is able to run across different cloud and cloud-like back-end infrastructures.

**Application Performance Considerations**

There are proposals on including HPC tools and scientific libraries in EC2 AMIs and have them ready to run on request. This might lead to re-implementing some HPC tools and deploying public images on Amazon or other vendors specifically for scientific purposes (e.g., the SGI Cyclone Cloud [5]). Still, in order to include ready-touse MPI clusters on EC2, there are several challenges to be met: The machine images must be manually prepared, which involves setting up the operating system, the
application's software environment and the security credentials.

There are two types of overhead: (i) added computational overhead of a VM and (ii) communication overhead when communicating between VMs. The first type of overhead results from the use of VMs and the fact that the underlying hardware is shared. While clouds nowadays deploy highly efficient virtualization solutions that impose very low overheads on applications ,unanticipated load increases on the cloud providers infrastructure can affect the runtime of scientific applications. The communication overhead mainly results from the fact that most clouds do not use networking hardware that is as low-overhead as that of dedicated HPC systems. There are at least two routes to parallelism in VMs. The first is a single VM across multiple cores; the second is parallelism across VMs. The latter type is especially affected from these communication overheads; that is, tightly coupled workloads (e.g., MPI jobs) are likely to see a degraded performance if they run across multiple VMs.

## VI. CONCLUSION

The usability and effectiveness of a programming model is dependent upon the desired degree of control in the application development, deployment, and execution. To efficiently support coordinated execution across heterogeneous grid and cloud infrastructures, programming tools and systems are required. It is important to ensure that such programming systems and tools provide open interfaces and support the IDEAS design objectives. Furthermore, these tools must address the cloud's inherent elasticity and support applications with dynamic resource requirements and execution modes. Programming systems such as SAGA provide developers with ability to express application decompositions and coordinations via a simple, high-level API.

## ACKNOWLEDGEMENT

I would like to express my sincere thanks to **Dr. C. Sunil Kumar Supervisor** and **Dr. N. Subhash Chandra Co-Supervisor** for constant help and guidance in PhD work and also I thank to my Organization CMRTC for encouragement**.**

## AUTHOR’S PROFILE

**1. Md. Rafeeq,** Working as a Associate Professor in Department of Computer Science and Engineering at CMR Technical Campus, Hyderabad. He completed B.Tech, M.Tech in Computer science and engineering and Pursuing PhD (CSE) in Cloud Computing at JNTHU Hyd.

**2. Dr C. Sunil Kumar** B.E ,M. Tech, PhD,Professor in IT, SNIST Ghatkesar, Hyderabad,

He has 46 research publications at International/National Journals and Conferences. His research interests are Distributed Databases, Data warehousing and Data Mining.

**3. Dr. N. Subhash Chandra**, M.Tech., Ph.D. , Principal and Professor in HITS,Bogaram,Keesara Hyderabad ,23 years of Teaching 1 Year industrial experience. His research area is Image Processing and published 15 National 32 International papers in various conference and journals.

## REFERENCES

[1] M. Haghighat, S. Zonouz, & M. Abdel-Mottaleb (2015). CloudID: Trustworthy Cloud-based and Cross-Enterprise Biometric Identification. Expert Systems with Applications, 42(21), 7905–7916.

[2] "ElasticHosts Blog". Elastichosts. 2014-04-Retrieved 2016-06-02.

[3] Jump up ^ Amies, Alex; Sluiman, Harm; Tong, Qiang Guo; Liu, Guo Ning (July 2012). "Infrastructure as a Service Cloud Concepts". Developing and Hosting Applications on the Cloud. IBM Press. ISBN 978-0-13-306684-5.

[4] M. Armbrust et al., Above the clouds: A Berkeley View of Cloud Computing, Technical Report UCB/EECS-2009 28, EECS Department, University of California, Berkeley, February 2009.

[5] N. Wilkins-Diehr, D. Gannon, G. Klimeck, S. Oster, and S. Pamidighantam,TeraGrid science gateways andtheirimpacton science.Computer, 41(11):32 41, 2008.

[6] 6.GoogleAppEngine,http://code.google.com/app engine/.windowsAzure, http://www.microsoft.com/windowsazure/.

[7] P. Watson, D. Leahy, H. Hiden, S. Woodman, and J. Berry, An Azure Science Cloud for Drug Discovery, Microsoft External Research Symposium, 2009.

[8]J. Dean and S. Ghemawat, MapReduce: Simplified data processing on large clusters, in Proceedings of the 6th Conference on Symposium on Operating Systems Design & Implementation, Berkeley, CA, USENIX Association, 2004, pp. 137 150.

[9] L. Youseff, M. Butrico, and D. Da Silva, Toward a unified ontology of cloud computing, in Proceedings of the Grid Computing Environments Workshop, GCE ’08, November 2008, 1_10.

[10] [10]Rajkumar Buyya, Chee Shin Yeo, and Srikumar Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In

[11] High Perfor- mance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on , pages 5{13, 2008.