A Reference Model of Cloud Operating and Open Source Software Implementation Mapping

Wenke Ji
Independent Consultant
Nanjing, R.R.C
e-mail: jiwenke@gmail.com

Jiangbo Ma
Alcatel-Lucent
Nanjing, P.R.C
e-mail: majb822@gmail.com

Xiaoyong Ji
Department of Electronics Science & Engineering
Nanjing University
Nanjing, P.R.C
e-mail: jxy@nju.edu

***Abstract*—in this article, a reference model is proposed. The model divides the cloud computing system with various components in a 3-layer hierarchy called infrastructure, platform and application. The details of the components are presented for its functionality assumed. Also the open source software implementation for the components in the model is addressed**

Keywords- Reference Model, Cloud Computing, Open Source Software, FCAPS

I. INTRODUCTION

Retrospect the past 20 years of those acceleration factors in the computing service evolution, the simplicity of the UNIX and the open source software such as LINUX plays an important role. As cloud application developers and utility users, some common points can be abstracted to a reference model. Following the analogy of the operation system, those common points can still be presented in the way that the traditional application developers are familiar with such as UNIX/LINUX architecture with a 3-layer hierarchy. With the open approach like LINUX, interoperability and communication can be made by the protocol defined as the component interfaces and those open source solutions can also be put into the stack like LINUX in the computing service ecology system.

II. DETAIL DESCRIPTION OF THE REFERENCE MODEL

In the model, the cloud system is clarified with a 3-layer hierarchy as follow:

- Cloud computing infrastructure layer provides a cluster of hardware resource such as CPU, memory, bandwidth and storage.
- The platform layer includes the components such as kernel, distributed file system; cloud IO, computing driver/engine, management and UI interface.
- The application layer host business domain specific application.

The model is showed in the following diagram:

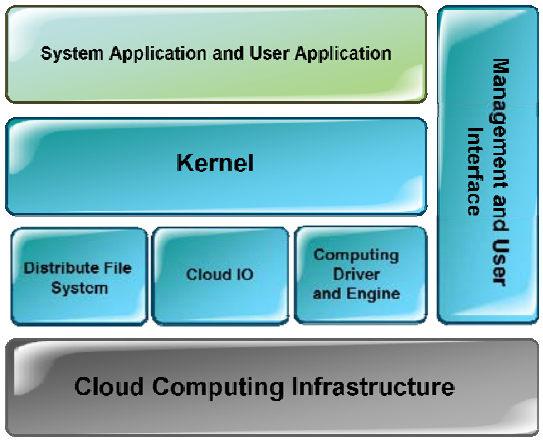

Figure 1. Reference model for the cloud computing utility users and application developers

Cloud Computing Infrastructure:

It provides basic environment delivering high scalability with network connection of the physical computing and data storage unit with virtualization service. Those basic physical cloud units can be a cluster of the commodity PC hardware or mainframe. In the basic physical cloud unit, the OS, network protocol implementation and virtualization software are needed for the functionality of this layer. The memory, storage device and network communication are managed by the operating system of the basic physical cloud units. Open source software such as LINUX/XEN can support the basic physical unit management and virtualization computing.

With the infrastructure layer, the platform layer can work independent with the hardware resource and provide cloud user with high scalability and manageability benefit. The infrastructure can be a public service such as Amazon EC2/S3 and others [1] or private owned solution.

Distributed File System:

A Distributed File System (DFS) is a network file system where data can be distributed among the physical nodes of cloud computing. File system objects are named in

1524-4547/09 $25.00 © 2009 IEEE
DOI 10.1109/WETICE.2009.28

a hierarchical name space among multiple name node servers. A DFS implementation takes the following functions [2]:

- Provide high data availability in the face of node failure, heavy load by allowing share in multiply different location;
- Provide a consistent view of the data seen by all clients in a DFS, and reliability in the case of failures, write operations are allowed to complete only after the data has been committed to stable storage
- To scale-up the commodity devices easily and economically on the large-scale cloud computing infrastructure. This includes the incremental scalable capability which is to add more devices to scale up the system in incremental fashion.
- Provide an effective secure manner.

Several implementations of distribute file system already existed. For example, Hadoop Distributed File System is an open source distributed parallel fault tolerant file system. It is designed to reliably store very large files across a large-scale cluster (HDFS) [3]. Google File System is a proprietary DFS for its own network level search engine. It is designed to provide efficient, reliable access to data using large clusters of commodity hardware (GFS) [4]. RedHat Global File System is an open-standard based system with great modularity and compatibility with interconnects, networking components and storage hardware. (RGFS) [5]. Amazon S3 also is one of the commercial services for it.

Kernel:

The kernel plays the role of global resource management on the basis of the infrastructure. It consists of four sub-components such as distributed tasks management, distributed memory management, system status monitor and communication utility.

Getting request from application, kernel take charge of tasks creation, assignment, schedule and execution exception handle based on the resource status. For the domain specific application, different task management implementation can be used according to application's characteristic, for example Goolge uses MapReduce [6] for its internet search service and All-InParis model is another one for the data intensive computing [7].

The distributed memory management delivers the service to the application which requires large volume memory to host consistent cache data. For data sharing and cache functionalities, both the distributed file system and memory system can be the solution candidates. Difference between the two approaches lies in whether the hardware IO operation is needed. In the bandwidth inversion cloud utility, the distributed memory approach is a more attractive one.

The communication utility serves the location independent data exchange which is requested from task and memory management component. For example in MapReduce model, it communicates data after Map complete and before Reduce begins [6].

To support high reliability distributed management, the resource monitor is a part of the kernel. It provides task and memory management with the health information of cloud computing system, such like status of bandwidth, CPU, memory and storage.

Open source software such as Apache Hadoop is a good candidate for the implementation and it uses MapReduce as task management model.

Cloud IO:

The Cloud IO encapsulates various kinds of data protocols and supports the data exchange for the tasks managed by the kernel. The cloud IO is independent with the hardware operation and the hardware operation is handled by the infrastructure layer. Cloud IO is a library waiting for service calling. The IO delivers its service directly to the kernel tasks. The IO also can provide data exchange service among different clouds with a defined protocol in the way such as web service.

In Apache open source software Hadoop, various IO encapsulated protocols are provided. Based on the framework and IO specification users can develop customized IO to support application specific need.

Computing Driver and Engine:

It provides the domain specific computing utility service to the application. In the scientific computing domain, the GNU octave [8] is such a kind of driver and Matlab is one of the commercial utility. Besides the scientific computing, compute engine in various domains can also be integrated as the driver component providing the specific computing capability to the domain specific application.

Management and User interface:

It provides the management console for the system admin and user as interface to the cloud. It supports system admin and users to monitor and manage cloud platform and applications. It includes fault management, configuration management, accounting management, performance management, and security management referred from TMN FCAPS model [9]. In this domain, many open source technology can be considered. The Web2.0 technology is a good candidate to play role in the building of user interface, which make cloud easily access through the Internet and WAN delivering desktop-like experience to the users.

System and User Application:

It is the application layer for the cloud; it can be system application which provides service to other application or the user application which serves the end-user. The application is domain specific, service oriented and its life cycle is managed by the system admin.

III. CONCLUSION

Mapping to the reference model proposed, some components like kernel, distributed file system already have many implementations with commercial or open source

software. For application developers, they don't need to care much detail of the service such as task management, memory, storage, communication any more. Contrarily, they only need to focus on application design including the application specific IO, distribute task design and domain use cases implementation. With the model proposed in this paper, the application developers can reuse the existed experience from traditional computing OS such as UNIX/LINUX and get a faster learning curve as the cloud utility users – which also make the communication more effective with the same context.

REFERENCES

[1] Rajkumar Buyya, Chee Shin Yeo, and Srikumar Venugopal "Market-Oriented Cloud Computing:Vision, Hype, and Reality for Delivering IT Services as Computing Utilities" The 10th IEEE International Conference on High Performance Computing and Communications , pp.10 Table 1

[2] Tran Doan Thanh; Subaji Mohan1;et al, "A Taxonomy and Survey on Distributed File Systems", School of Business IT, Kookmin University, Seoul, Korea, 2008

[3] Apache Hadoop Documentation http://hadoop.apache.org/core

[4] Ghemawat, S., Gobioff, H., Leung, S.T., "The Google file system", ACM SIGOPS Operating Systems Review, Volume 37 , Issue 5, pp. 29-43, December, 2003.

[5] Red Hat Global File System, White Paper, http://www.redhat.com/whitepapers/rha/gfs/GFS_INS0032US.pdf.

[6] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", Communications of the ACM, ACM, January 2008.

[7] Christopher Moretti, Jared Bulosan, Douglas Thain, and Patrick J. Flynn, "All-Pairs: An Abstraction for Data-Intensive Cloud Computing", Parallel and Distributed Processing, 2008. IEEE International Symposium

[8] GNU Octave Documentation http://www.gnu.org/software/octave/

[9] ISO/IEC 10040, 1998, "Information technology - Open Systems Interconnection - Systems management overview"