2013-03

Three-Dimensional Space to Assess Cloud Interoperability

Fazai, Sallouha

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/32818

NAVAL
POSTGRADUATE
SCHOOL

MONTEREY, CALIFORNIA

THESIS

THREE-DIMENSIONAL SPACE TO ASSESS CLOUD INTEROPERABILITY

by

Sallouha Fazai

March 2013

Thesis Advisor:	Man-Tak Shing
Second Reader:	Albert Barreto III

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** March 2013	**3. REPORT TYPE AND DATES COVERED** Master's Thesis
4. TITLE AND SUBTITLE Three-Dimensional Space to Assess Cloud Interoperability		**5. FUNDING NUMBERS**
6. AUTHOR(S) Sallouha Fazai		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number N/A.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

Cloud computing is an emerging technology that promises the reduction of IT costs (personnel, software, and hardware) for enterprises, as well as individual users. Despite this appealing offer, this technology has still not been widely adopted in the enterprise IT. Users are still worried about vendor lock-in; they will not be able to move their data and applications from one cloud provider to another easily or return to in-house IT. Currently, users do not have the means to specify and assess the interoperability level of the cloud provider that they desire to entrust their IT operations. In this thesis work, we provide a three-dimensional space to assess and visualize the interoperability level of any cloud provider so that cloud users can select the provider's services that better fit their interoperability needs.

14. SUBJECT TERMS Cloud computing, Interoperability, cloud provider, and lock-in			**15. NUMBER OF PAGES** 65
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

THREE-DIMENSIONAL SPACE TO ASSESS CLOUD INTEROPERABILITY

Sallouha Fazai
Lieutenant Junior Grade, Tunisian Navy
Computer Engineering, Tunisian Navy, 2007

Submitted in partial fulfillment of the
requirements for the degree of

**MASTER OF SCIENCE IN
INFORMATION TECHNOLOGY MANAGEMENT**

from the

**NAVAL POSTGRADUATE SCHOOL
March 2013**

Author:	Sallouha Fazai
Approved by:	Man-Tak Shing Thesis Advisor
	Albert Barreto III Second Reader
	Dr. Dan Boger Chair, Department of Information Sciences

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Cloud computing is an emerging technology that promises the reduction of IT costs (personnel, software, and hardware) for enterprises, as well as individual users. Despite this appealing offer, this technology has still not been widely adopted in the enterprise IT. Users are still worried about vendor lock-in; they will not be able to move their data and applications from one cloud provider to another easily or return to in-house IT. Currently, users do not have the means to specify and assess the interoperability level of the cloud provider that they desire to entrust their IT operations. In this thesis work, we provide a three-dimensional space to assess and visualize the interoperability level of any cloud provider so that cloud users can select the provider’s services that better fit their interoperability needs.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION 1
- **A. THESIS STRUCTURE 2**

II. OVERVIEW OF CLOUD COMPUTING 3
- **A. CONCEPT 3**
- **B. SERVICE MODELS 4**
  - **1. SaaS 6**
  - **2. PaaS 6**
  - **3. IaaS 6**
- **C. DEPLOYMENT MODELS 6**
- **D. CLOUD STANDARDS 7**
  - **1. Open Cloud Computing Interface (OCCI) 7**
  - **2. Open Virtualization Format (OVF) 9**
  - **3. Cloud Data Management Interface (CDMI) 9**
  - **4. Standards under Development 10**

III. CLOUD INTEROPERABILITY 11
- **A. CONCEPT 11**
  - **1. Vertical Dimension 11**
  - **2. Horizontal Dimension 12**
- **B. USE CASES 12**
- **C. CHALLENGES 12**
  - **1. Portability and Mobility 13**
  - **2. Security, Privacy, and Trust 13**
  - **3. Management 13**
- **D. REQUIREMENTS 13**
  - **1. Technology 14**
    - ***a. Virtualization 15***
    - ***b. Security Mechanism 15***
    - ***c. Service Architecture 15***
    - ***d. Data Format 16***
  - **2. Management 16**
    - ***a. Management Interface 16***
    - ***b. Provisioning/Scheduling 16***
    - ***c. Security 17***
    - ***d. Monitoring/Reporting 17***
    - ***e. Metering/Usage/Billing 17***
    - ***f. SLA and QoS 17***
    - ***g. Auditing 18***
  - **3. Policy 18**
    - ***a. Internal Policy 19***
    - ***b. Geographical Policy 19***

E. THREE-DIMENSIONAL SPACE FOR CLOUD INTEROPERABILITY 19
1. Description of the Three-dimensional Space 20
a. Technology Dimension 20
b. Management Dimension 20
c. Policy Dimension 21
2. Comparing Cloud Services Using the Three-dimensional Space ..22

IV. CASE STUDY: USING THE THREE-DIMENSIONAL SPACE TO COMPARE THE INTEROPERABILITY LEVELS OF TWO CLOUD PROVIDERS 23
A. OPENSTACK CLOUD 23
1. Overview 23
1. OpenStack Interoperability 26
B. OPENNEBULA CLOUD 28
1. Overview 28
3. OpenNebula Interoperability 30
C. COMPARISON OF OPENSTACK AND OPENNEBULA 32
1. Visualization of Interoperability in the Three-dimensional Space 34

V. CONCLUSIONS AND RECOMMENDATIONS 37
A. THESIS SUMMARY 37
B. LIMITATIONS AND RECOMMENDATIONS 37
C. FUTURE WORK 38
1. Cloud Auditing 38
2. SLA and QoS 38
3. Policy 39

LIST OF REFERENCES 41

INITIAL DISTRIBUTION LIST 47

LIST OF FIGURES

Figure 1. Overview of the Different Service Models of Cloud Computing (From [9]) 5
Figure 2. Control and Management Responsibilities of the Cloud Provider and the Cloud User (From [10]) 5
Figure 3. Place of the OCCI in the Provider Architecture (From [11]) 8
Figure 4. Overview of Openstack Architecture (From [40])....................24
Figure 5. Overview of OpenNebula Architecture (From [60]) 29
Figure 6. OpenStack (OS) and OpenNebula (ON) Interoperability....................36

THIS PAGE LEFT INTENTIONALLY BLANK

LIST OF TABLES

Table 1. Comparison of OpenStack and OpenNebula 33

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ACL	Access Control List
AD	Active Directory
ANSI	American National Standard Institute
API	Application Programming Interface
AWS	Amazon Web Service
CDMI	Cloud Data Management Interface
CLI	Command Line Interface
CPU	Central Processing Unit
CSA	Cloud Security Alliance
DMTF	Distributed Management Task Force
EBS	Elastic Block Storage
GUI	Graphical User Interface
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
ICITS	InterNational Committee for Information Technology Standards
iSCSI/LVM	Internet Small Computer System Interface/Logical Volume Manager
ISO	International Organization for Standardization
IT	Information Technology
KVM	Virtual-based Virtual Machine
LDAP	Lightweight Directory Access Protocol
LXC	Linux Containers
NIST	International Institute of Standards and Technology
qcow	QEMU Copy on Write
QEMU	Quick EMUlator
QoS	Quality of Service
oAuth	Open Authentication Protocol
OCA	OpenNebula Cloud API
OCC	Open Cloud Consortium

OCCI	Open Cloud Computing Interface
OGF	Open Grid Forum
REST	Representational State Transfer
SLA	Service Level Agreement
RPC	Remote Procedure Call
RSA	Ron Rivest, Adi Shami, and Leonard Adleman (algorithm for public key encryption)
SAML	Security Assertion Markup Language
SNIA	Storage Networking Industry Association
SQL	Structured Query Language
SOA	Services Oriented Architecture
SSH	Secure Shell
SSL	Secure Socket Layer
S3	Amazon Simple Storage Service
swauth	Authentication middleware for Swift
UML	Unified Modeling Language
VM	virtual Machine
VMDK	Virtual Machine Disk
VMFS	Virtual Machine File System
VNC	Virtual Network Computing
XML	Extensible Markup Language

ACKNOWLEDGMENTS

The true sign of intelligence is not knowledge but imagination.

—Albert Einstein

I would like to thank my thesis advisor, Professor Man-Tak Shing, for his guidance, constructive criticism, patience, and the unlimited time that he devoted to this work, regardless of his hectic schedule.

I also would like to thank my second reader, Albert Barreto, an Information Sciences lecturer. His recommendations helped me shift focus from network security to emerging technology, and specifically, to cloud computing, which is gaining greater attention from industry and academia. The change was the start of a new journey that would be considered as the first takeoff into the cloud technology. I also would like to thank him for the time that he devoted to read this work and provide constructive feedback.

I am thankful to Mr. Richard Cook and Donna Cuadrez for the time they devoted to read and edit this thesis.

I am grateful to my parents, brothers, sisters, and friends for their unlimited support.

Thanks to everyone who contributed to making my stay in Monterey enjoyable.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Cloud computing is an evolving technology that promises the reduction of IT expense by reducing maintenance, licensing and hardware expenditures. It succeeded in attracting the attention of industry, and the competition in this domain led to the emergence of different cloud providers, such as Google, Microsoft, and Apple. The number of cloud providers is likely going to increase. The lack of standardization, due to the infancy of this technology and the reluctance of cloud providers to adhere to standards, led to different cloud implementations. Building an interoperable cloud ecosystem becomes a mandatory requirement to free cloud users from provider lock-in, thereby stimulating cloud adoption. Indeed, users will be unwilling to move to the cloud unless they know that they can move seamlessly their data and applications from one cloud to another.

Research efforts devoted to cloud interoperability have provided various solutions, such as gateways (middleware) and plugins or drivers to allow interaction among different cloud platforms. Organizations such as DMTF, OVF, and CSA focused on developing cloud standards. However, cloud interoperability is still challenging since the provided solutions have limitations. The use of gateways degrades performance as the number of systems increases, and it requires the development of a translator and adaptor for each protocol [1]. The plugin and driver approach needs an extensive coding effort to respond to the speedy increase of the number of cloud providers [2] . In addition, standardization efforts were hindered primarily because of the reluctance of cloud providers to follow these standards [3]. Moreover, these standards (e.g., OCCI, OVF, and CDMI) could not provide solutions to all the interoperability issues [4], and the specific implementation options that are included in these standards led to different cloud implementations, while using the same standard [1].

Considering the limitation of the aforementioned solutions to resolving cloud interoperability issues, this thesis focuses on providing a framework to assess the interoperability level of any cloud provider. This framework will help users select the cloud services that better fit their interoperability needs and help cloud providers to improve the interoperability level of their services.

A. THESIS STRUCTURE

In Chapter II, we provide an overview of cloud technology: concept, service models, service deployments, and cloud standards.

In Chapter III, we focus on cloud interoperability. We present main use cases and highlight challenging issues. We then identify interoperability requirements. We end this chapter with the presentation of a three-dimensional space for assessing cloud interoperability.

Chapter IV is devoted to a case study: comparison of the interoperability levels of two major cloud providers, OpenStack and OpeNebula, to demonstrate the usage of the three-dimensional space and its benefits. We start this chapter with a study and analysis of the cloud platforms provided by both cloud providers, and we end it with a discussion and visualization of the interoperability levels of the two cloud platforms in the three-dimensional space.

In Chapter V, we provide a summary of this thesis work. We present its limitations and provide recommendations. Finally, we highlight several areas of interest that still need more attention from the academic, research, and industrial sectors, in order to achieve cloud interoperation.

II. OVERVIEW OF CLOUD COMPUTING

A. CONCEPT

Several computing paradigms, such as cluster computing, grid computing, virtualization, and Web2.0, enabled the birth of cloud computing [5]. Since its birth in late 2007, this technology has gotten many definitions; there is still no common standard to define it. In effect, it is defined as a collection of distributed computers that provides resources and services over a network depending on customer demand [6]. Another definition considers cloud computing as a collection of network-enabled services that guarantees to provide a scalable, easy accessible, reliable, and personalized computing infrastructure, based on demand with low-cost [7]. In 2011, NIST released an informal definition of this emerging technology, and claimed that this definition will change over time depending on the evolution of cloud computing:

> Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential characteristics**,** three service models, and four deployment models [8].

The cloud computing paradigm aims to free cloud users from hardware and software dependency. It requires possession of a web browser to access these resources over a network (generally, the Internet). According to the latest NIST draft [8], cloud computing is characterized by the following characteristics that differentiate it from other computing paradigms, such as grid computing (GC) and virtualization.

- **On-demand self-services**: cloud computing provides services that can be accessed on demand easily by the customer. The customer is billed based on time of usage of these services. This billing type is referred as “pay as you go.”
- **Broad network access**: access to cloud services is guaranteed to diverse users using a standard mechanism.

- **Resource pooling:** cloud services can be accessed by different customers simultaneously based on their demands, and typically, without caring about the location of these resources.
- **Rapid elasticity:** cloud services should be quickly and easily provisioned without limitations.
- **Measured service**: the use of cloud-computing services should be controlled, monitored, and recorded to avoid any problem that may occur between customers and cloud providers.

B. SERVICE MODELS

Many different concepts are used in research to describe cloud models, such as SaaS (Software as a Service), PaaS (Platform as a service), IaaS (Infrastructure as a Service), SaaS (Storage as a Service), and HaaS (Hardware as a Service). In this thesis, we will adopt the NIST models which are SaaS (Software as a Service), PaaS, and IaaS [8]. These models are defined according to services offered and customer ability to control and manage the compounds of the cloud infrastructure (e.g., applications, network, servers, operating systems, storage). Figure 1 provides an overview of the different service models, and Figure 2 shows the scope of control and management responsibilities of the cloud resources for the cloud provider and user (cloud service consumer).

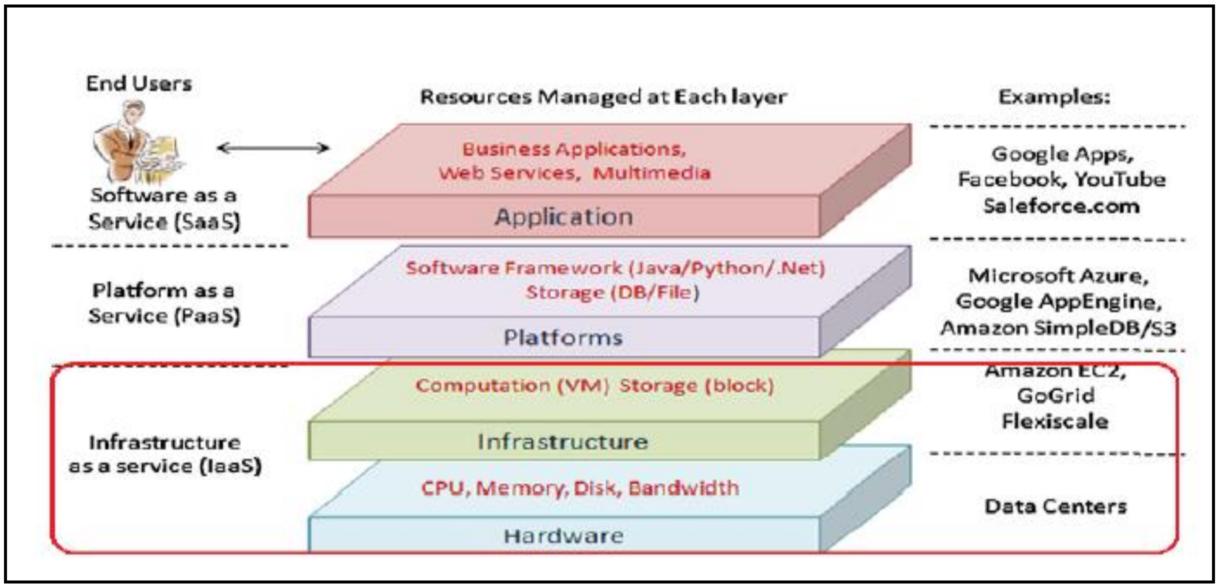
End Users
Resources Managed at Each layer
Examples:
Software as a Service (SaaS)
Business Applications, Web Services, Multimedia
Application
Google Apps, Facebook, YouTube Saleforce.com
Platform as a Service (PaaS)
Software Framework (Java/Python/.Net) Storage (DB/File)
Platforms
Microsoft Azure, Google AppEngine, Amazon SimpleDB/S3
Infrastructure as a service (IaaS)
Computation (VM) Storage (block)
Infrastructure
Amazon EC2, GoGrid Flexiscale
CPU, Memory, Disk, Bandwidth
Hardware
Data Centers

Figure 1. Overview of the Different Service Models of Cloud Computing (From [9])

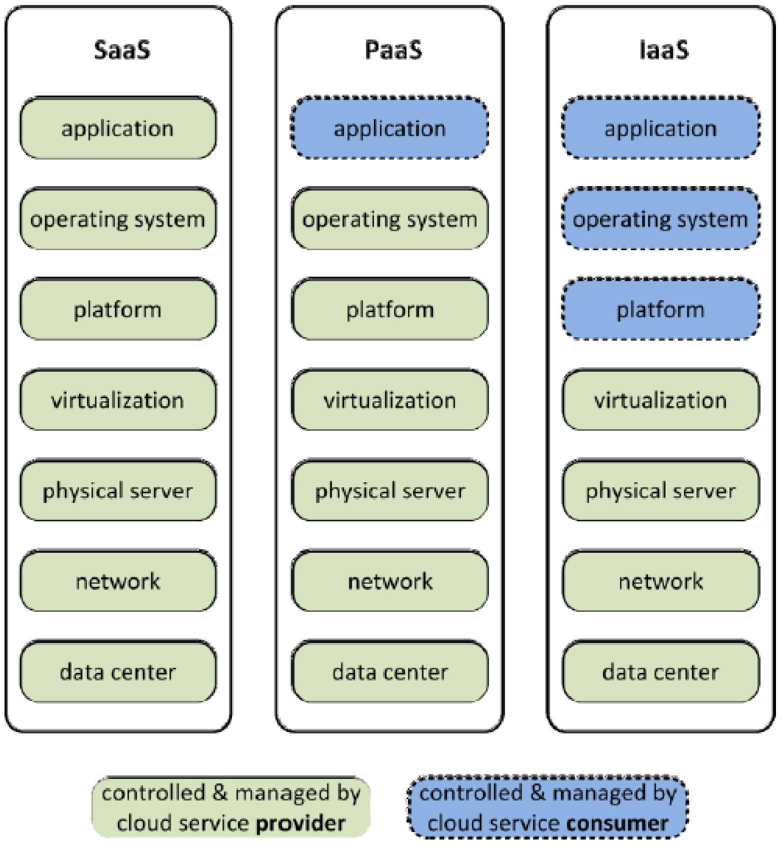

Figure 2. Control and Management Responsibilities of the Cloud Provider and the Cloud User (From [10])

1. SaaS

The customer does not possess any control or management privileges over the cloud infrastructure (e.g., network, storage, computing resources, operating system) or applications, except very restricted application configuration settings. The customer just accesses the cloud applications through a web browser, or any other interface, and uses these applications. This model, frees the customer from installing software and paying a licensing cost. Google Docs and Microsoft Office Web Apps are examples of SaaS implementations.

2. PaaS

The customer does not possess any management or control privileges over the cloud infrastructure. The customer can only deploy and manage his/her own applications and specify the settings of the application hosting environment. Google App Engine and Amazon Web Services (AWS) are examples of PaaS implementations.

3. IaaS

This model offers a virtual cloud infrastructure (e.g., computing resources, networks, storage) where the customer can select and configure the cloud platform that will host his/her applications, data, and software (e.g., operating systems). Users do not have any control over the physical or virtual cloud infrastructure. They may have limited control over specific network devices (e.g., host firewall). GoGrid's Cloud Servers and Amazon EC2 are examples of IaaS implementations.

C. DEPLOYMENT MODELS

NIST distinguishes four deployment models for cloud computing: public, private, community, and hybrid. When the cloud infrastructure is reserved only to one organization regardless of its location (on or off premises), the cloud is referred to as private cloud. The organization can take charge of the management responsibilities of its cloud, or delegate them to a third party. In contrast, the cloud infrastructure that is accessible by the public or a large number of organizations and is normally owned by a cloud service seller is referred to as public cloud. Organizations that have common interests (e.g., mission, security requirements) may agree to build a shared cloud. This cloud is referred to as community cloud. Regardless of whether it is on or off premises, it can be managed by these organizations or another party. The combination of two or more types of clouds, with the ability to move applications and data within these clouds, is referred to as hybrid cloud [8].

D. CLOUD STANDARDS

The competition among cloud providers led to different proprietary cloud implementations that resulted in provider lock-in. This lock-in hindered the adoption of cloud computing. Research efforts conducted by non-profit working groups, such as OGF, DMTF, and SNIA, focused on standardization to stimulate the evolution and adoption of cloud computing. Many standards were developed and others are still under development. These standards aim to build an interoperable cloud ecosystem, where cloud users can move their data and applications from one cloud provider to another without any difficulty. In this section, we provide an overview of these standards.

1. Open Cloud Computing Interface (OCCI)

OCCI is an open standard developed by OGF that aims to offer provider neutral tools to manage cloud resources. It was originally developed to support IaaS interoperability and later extended to include SaaS and PaaS. It focuses mainly on integration**,** portability, and interoperability. OGF is collaborating with other working groups, such as DMTF and SNIA, to improve cloud interoperability. As shown in Figure 3, OCCI acts as intermediate or interface between the cloud provider and users (end users or another system)**.** The OCCI has a modular core that supports only specific cases, yet it can be expanded using renderings[1] and extensions. It has a default RESTful HTTP rendering. In order to adhere to the OCCI standard, OGF specifies mandatory requirements to guarantee an acceptable level of compatibility between different OCCI implementations [11].

[1] Renderings: define how to interact with the core model**.**

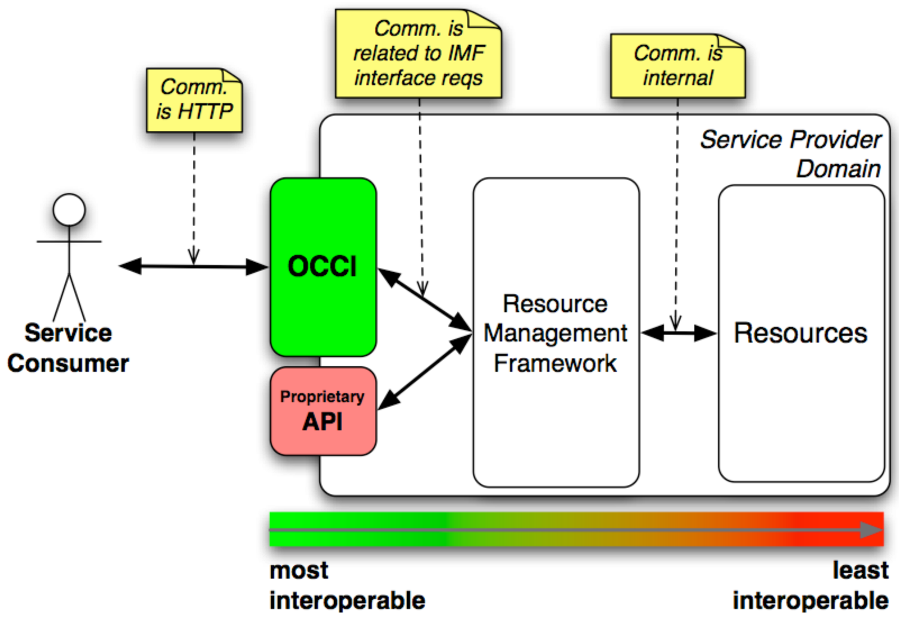

Figure 3. Place of the OCCI in the Provider Architecture (From [11])

OCCI is not just specification; it has many real implementations. We cite mainly the following projects (for other projects and more details, see this link "http://occi-wg.org/community/implementations/"):

- Eucalyptus: a broadly used, open-source software for the deployment of private and hybrid cloud. It aims to provide an interoperable IaaS without imposing infrastructure restrictions. It exposes the existing infrastructure as a web service [12].
- RESERVOIR (Resources and Services Virtualization without Barriers): a European Union project for the deployment and management of complex IT systems. OCCI is used to integrate RESERVOIR project and the SLA@SOI project (SLA@SOI is also a European project that focuses on SLA) [13].
- OpenStack: an open operating system for building public and private cloud computing developed by NASA and RackSpace. Many other organizations joined the project later, such as HP, DELL, CISCO, and RedHat[14].
- The Morfeo Claudia Platform: a cloud platform for the dynamic control of service provisioning and scalability for IaaS cloud. It can be extended through Tcloud to include PaaS and SaaS [15].
- OpenNebula: an open source solution for the management of cloud data centers. It aims to provide a solution to the management of these data centers without imposing infrastructure restrictions [16].

- LibVirt: a toolkit for the management of an operating system instance running of virtual machine [17].

2. Open Virtualization Format (OVF)

OVF is an open standard developed by DMTF to provide *an open, secure, portable, efficient and extensible format for the packaging and distribution of software to be run in virtual machines*, thereby guaranteeing portability among different virtualization platforms. OVF is a platform and vendor neutral format that can be used for single VM or multiple VMs. A virtual appliance[2] autonomously configures and modifies its configuration; therefore, there is a separation between the virtualization platform and appliance. OVF supports all the existing virtual hard-disk formats, and it can be extended to include new formats. OVF format can be extended either by adding new sections or expanding existing sections. OVF has a certification and integration mechanism that enables the platform to check the provenance and integrity of the virtual appliance. This gives more transparency to users [18].

3. Cloud Data Management Interface (CDMI)

CDMI is a SNIA interface for the management of the data stored in the cloud. The mechanisms that are used to manage data are referred to as *control path* and the mechanisms of data storage and retrieval are referred to as *data path*. CDMI features allow clients to manage their data and the containers containing this data. This interface offers management functionalities that include container and domain management, security access, monitoring and billing information. These capabilities are exposed, so they can be discovered by CDMI clients. It is compatible with standardized and proprietary data path interfaces and legacy systems, and it can be deployed above them or at the same level. CDMI is based on the notion of objects. It has five types of objects—data, container, domain, queue, and capability—and it has different metadata models that are used to manage the stored data. We mainly cite HTTP metadata, data system metadata, user metadata, and storage system metadata. SNIA interface enables the migration of data and metadata seamlessly from one cloud provider to another. It is on its

[2] Virtual appliance is software installed on one or many virtual machines, and delivered as services.

way to becoming an ANSI and ISO standard. An open source implementation of this interface is available. There is an existing implementation of the integration of OCCI and CDMI (R2AD) and integration with OVF is under development [19].

4. Standards under Development

OCC is focusing on large data clouds. It is working to provide a unified cloud interface that unifies different cloud APIs to enable cloud interoperability [20]. IEEE is also working to provide a portability standard referred to as IEEE P2301, Draft Guide for Cloud Portability and Interoperability Profiles (CPIP), and a standard to allow two different cloud implementations to interact with each other referred to as IEEE P2302, Draft Standard for Intercloud Interoperability and Federation (SIIF) [21].

III. CLOUD INTEROPERABILITY

Although a big concern, security will not impede the adoption of cloud computing. Interoperability remains the main hindrance. With the emerging proprietary cloud implementations (e.g., EC2, Google Apps, and Microsoft Azure), cloud customers, and especially enterprises, are reluctant to move to the cloud because of vendor lock-in. They are afraid that if they are no longer satisfied with the provided services for financial, QoS, or any other reasons, they will not be able to seamlessly move to another cloud or return to in-house services. Moreover, cloud customers are unable to combine different cloud services to get “the best of the bread.” In this chapter, we focus on cloud interoperability: concept, use cases, and challenges. We then identify the interoperability requirements, and we end this chapter by presenting a three-dimensional space that can be used to assess the cloud provider’s interoperability level.

A. CONCEPT

Generally, interoperability means the ability of different systems to communicate and interact with each other. Since cloud computing is an evolving technology, defining cloud interoperability is challenging [22]. The concept evolves as the technology evolves. In an interoperable cloud system, different cloud platforms (on or off premises) should be able to collaborate [23]. Cloud interoperability includes data, applications, physical or virtual machines, and other features, such as management, provisioning, policy, SLA, and QoS [3]. Cloud interoperability has the following two dimensions of focus [24].

1. Vertical Dimension

Vertical dimension is concerned only with the interoperability of a single cloud provider to the end user’s devices and applications. This means that users are able to access, retrieve, store, and process their data and they can run their own applications using any device connected to the internet (e.g., laptop, desktop, and iPhone). Briefly, the vertical dimension frees users from device and location dependency.

2. Horizontal Dimension

Interoperability is addressed among different cloud providers. This dimension can be referred to as *the cloud to cloud interoperability* as described in [25]. The horizontal dimension aims to unlock the provider lock-in so users will be able to move seamlessly from one provider to another or combine services from different clouds. The horizontal dimension includes technical issues as well as other incompatibility issues, such as privacy, contracts, QoS, and security policy.

B. USE CASES

Cloud interoperability involves different scenarios. Generally, the most common scenarios are:

- Using multiple cloud providers: either because of the limitation of one cloud provider or for other reasons, such as finance, QoS, and SLA, cloud users are willing to combine different cloud services to get "the best of the bread".
- Combining cloud technology and in-house IT: organizations may want to combine their own IT infrastructure and applications with the cloud. For instance, they may want to use in-house IT to manage their critical assets and move unclassified information and remaining applications to the cloud.
- Changing the cloud provider: for whatever reasons, cloud users may need to move to another provider.
- Cloud federation: due to an unexpected increase in usage of the cloud resources, or for other reasons that could affect the operation of any cloud provider—such as security breaches, bankruptcy, and natural disasters—a cloud provider may become unable to provide services to its customers, so the provider requests services from one or more other cloud providers.

C. CHALLENGES

Standardization is a key enabler for cloud-to-cloud interoperability, yet due to the infancy of this technology, mature standards are lacking to overcome the challenges that are hindering cloud interoperation. Actually, major cloud providers are unwilling to follow standards. Every provider implements its proprietary solutions to protect its assets [26, 27]. We can classify these challenges into the following three categories.

1. Portability and Mobility

Moving VMs, data, and applications across the cloud, as well as integrating in-house IT with the cloud, or combining cloud resources or services from different cloud providers, is still impracticable. It may require a lot of coding efforts or the use of middleware, which causes performance degradation as the number of cloud providers increases, especially for the combination or integration scenario. Cloud providers frequently offer IaaS, SaaS, and PaaS as a one stack. This makes portability and mobility difficult, because these services are highly correlated [28].

2. Security, Privacy, and Trust

Despite the improvements in authentication, identification, and integrity mechanisms in the cloud, we still lack a strong authentication mechanism for a single provider, as well as across different cloud providers. Data needs protection, whether it is in transit, or stored. Since data moves from one cloud provider to another, holding one entity liable—in case of privacy violation, for example—is impossible. In addition, different legislation systems around the world remain big challenges to cloud interoperability [27].

3. Management

Cloud interoperation requires the automation of all the management tasks across cloud provider boundaries, yet automation is still not achieved, even for a single cloud provider [29].

D. REQUIREMENTS

Interoperability requirements are driven from the user's needs and expectations, not the cloud provider's. Providers normally look for user lock-in to protect their assets. In this section, we classify interoperability requirements into three categories: technology (the technology used by the provider to build its cloud), management, and policy. For each category, we identify the attributes to assess the interoperability level from a user perspective. SaaS and PaaS cannot be separated from the IaaS; PaaS is implemented over IaaS, and SaaS implemented over PaaS [29]. As stated earlier in this chapter, cloud providers usually offer these services as one stack. Therefore, we look at the three service models as one stack with a main focus on PaaS/IaaS since it is the basic foundation of SaaS.

1. Technology

In order to build an interoperable cloud ecosystem, the technology used by each cloud provider (virtualization, security mechanism, data representation) to build its own cloud, should be able to communicate with any other cloud provider’s technology. This communication should guarantee cloud portability and mobility. Portability and mobility refer to the ability to move data, applications, images or VMs among different cloud platforms or providers. Urquhart[3] referred to portability as *the ability to move an image in a "down" state, and boot it at its destination,* and mobility as *the ability to move a live compute workload without losing client connections or in-flight state* [32]. As cloud technology evolves, the user’s expectation from the cloud evolves too. Therefore, in this research, we will consider a combination of these two concepts. Cloud users may wish to move their applications, data, and VMS in a down state, as well as in-flight state. Cloud portability and mobility targets three main cloud resources: VMs, data, and applications.

- VMs: The ability to import locally created VMs into the cloud and move VMs across different cloud platforms or providers or hypervisors4 with minimum effort. For instance, reconfiguration of network settings is not needed [22]. VMs can be in down state or running. OVF is a step toward VM portability and mobility.
- Application: The ability to seamlessly move applications (including legacy applications) and their related data from one cloud to a different one and run these applications at the destination successfully. Application portability includes the migration of running applications with the required monitoring and management features [30]. The use of standardized cloud APIs could facilitate application portability [31].
- Data: Data in the cloud are either stored in structured (e.g., database) or unstructured (file) forms. Data portability in the cloud is not restricted to just moving data from one location to another. Data portability depends on the applications that use this data. While moving from one cloud provider to another, cloud users need to be able to use their data in the new cloud’s

[3] Market strategist for Cloud computing.

[4] Called also VM manager; controls different guest operating systems at the host machine.

equivalent applications. Data is not just users' data. It may include data related to management and policy [32]. Achieving data portability requires platform-neutral data format, standardization of data import and export, and compatible or platform independent tools to access and store data in cloud [22], [33].

We identify the following attributes to assess the portability and mobility level of cloud services

a. Virtualization

The virtualization technology should offer a complete abstraction of the cloud provider infrastructure (hardware and software). Users just need a device connected to the Internet (e.g., iPad, Laptop, iPhone) to access the cloud. In addition, they can access and manage their data and deploy their applications in the cloud without imposing the use of specific software (e.g., OS, web browser, programming language). Virtualization should also comply with the following requirements:

- Enable the migration of workloads (applications and VMs) in down state or on the fly among different clouds.
- Hypervisor should be OS neutral and support any existing VM format.
- "*An open, secure, portable, efficient, and flexible format for the packaging and distribution of one or more virtual machines*" [28].

b. Security Mechanism

The deployed security mechanism should not inhibit communication with other clouds having a similar security level. It should also offer an "acceptable" level for the protection of the transmission of data, VMs, and applications. We focus mainly on authentication and integrity.

c. Service Architecture

Cloud services should be able to communicate with each other independent of their providers. How these services are designed is critical to achieving this goal. They should be designed following standards and design principles that support cloud interoperability. Following SOA principles (composability, discovery, autonomy, loose coupling, abstraction, and standardized service contract) as proposed in [25, 34] leads to building interoperable cloud services.

d. Data Format

Independent of how the data is stored in data centers, cloud providers should provide tools that allow the conversion of data from one format to another and export of data, in order to allow data portability around the cloud.

2. Management

Moving data, applications, and VMs around the cloud, is not enough to build an interoperable cloud ecosystem. A fundamental interoperability requirement is providing efficient management tools that allow users to monitor, provision, and control different cloud resources [35]. The ability to move from one cloud provider to another, requires checking security policy, SLA, QoS, reliability, and so on. To achieve these objectives, we need unification, automation, and openness of management activities of each cloud provider. We identify the following attributes to assess the interoperability level of provider management capabilities.

a. Management Interface

Cloud interoperation requires a unified and user-friendly interface for the management of services pooled from different clouds, with the least effort and interaction with the cloud provider, as stated in NIST definition of cloud computing [8]. Cloud providers are required either to provide this interface, or allow their services to be controlled by the management tools of a third party. Users can delegate the management task to the provider.

b. Provisioning/Scheduling

The cloud provider should provide an automatic engine that allows the allocation of its resources, as well as other cloud resources. The usage of cloud resources varies, based on user demands. If demands exceed the provider’s capabilities, the provision engine allows for the allocation of resources from other providers [36]. Resources should be published in a public directory so they can be discovered.

c. Security

An implemented security mechanism that guarantees the protection of user data and applications as desired. Users need an automatic engine to assess the security level of the provider's services before using them. For the scope of our work, we focus mainly on authentication and integrity.

d. Monitoring/Reporting

The cloud platform should provide secure and reliable tools that allow users to monitor the services they are using. These tools should alert users when there is an anomaly, such as performance degradation, increase in usage, or unavailability of service. The monitoring information should be registered for a fixed period stated clearly in the SLA or the provider's policy to avoid problem that may arise between the users and the provider.

e. Metering/Usage/Billing

Cloud providers should provide a reliable pricing model, such as pay as you go, so users cannot be locked out because of the payment strategy. Providers should also offer monitoring tools for users to monitor usage and pricing, so they can make informed decisions. For example, in the case of a price increase, they can either stop or change providers.

f. SLA and QoS

The automatic negotiation of SLAs is a critical requirement for cloud interoperation. Cloud platforms should provide an engine that supports this functionality. The negotiation should be based on QoS metrics (e.g., performance, availability, cost, reliability, security) as stated in [27]. Since cloud user demands change over time, the SLA implementation should be flexible enough to accommodate this change.

g. Auditing

Checking the compliance of the cloud providers with regulations, SLA, security policy and standards, and so on increases trust between users and cloud providers. Providing the necessary tools and information to allow automatic evaluation of the cloud services is a paramount step toward an interoperable cloud ecosystem. For reliability and objectivity, we suggest that this functionality be carried out by a third trusted party which uses standardized auditing methods and publishes the audited reports openly. As discussed in [37], we are looking for a near real-time auditing capability to evaluate the offered services.

3. Policy

To build an interoperable cloud ecosystem, cloud users want to move their data and applications from one cloud provider to another without changes that could affect the level of the provided services. The provider's technology is not the only obstacle to cloud interoperation. Policy can be a major impediment. For instance, providers who follow standards and best practices can build interoperable cloud implementations, yet their policies can hinder this interoperation. In fact, it is very likely that providers will have different policies: how they handle contracts, how long they keep user's records, or how often they do backups.

In addition to the diversity of provider policies, legislation systems can impose barriers to cloud interoperation. Each country has its own rules concerning privacy, data protection, reliability, liability, and other issues that relate to the cloud. For instance, Europe is more conservative about privacy information than the USA [38]. Inside the same country, rules may differ at the municipal as well as the state level.

Cloud interoperability mandates two fundamental requirements. First, automated and standardized engine to allow cloud providers implement their cloud policies. Second, since provider's policy cannot go beyond legislative requirements, a standard legislative system that handles all cloud issues is needed. Thereby, the provider's policy cannot be restricted because of geographical location. We propose the following attributes to assess the interoperability level of the provider's policy.

a. Internal Policy

The cloud provider should provide a full, clear, and detailed description of its policy (security plan, contract management). Provider should state any constraint, requirement or obligation that should be considered in order to access its services.

b. Geographical Policy

Cloud providers should provide information about the physical location of their services and the legislative rules (privacy, liability, or any other rule related to cloud computing) that must be respected in order to comply with the regulation requirements of this location.

E. THREE-DIMENSIONAL SPACE FOR CLOUD INTEROPERABILITY

Cloud users care about maintaining the same level of services, while moving around the cloud or combining different clouds. They may also wish to keep the same SLA, security level, and QoS. Before adopting any cloud provider, users need answers to many questions to assess the interoperability level of the provider services. These questions include:

- Independent of the software and hardware that they have, can users access to the provider's services?
- Does the provider technology allow users to move easily their data, applications, and VMs to another cloud either on the fly or in a down state?
- Can users pool services from other providers and still control and manage all the pooled services in a unified way?
- Does the provider policy allow users to move across the cloud while maintaining the same level of services? Will Geographical regulation pose a problem?

We build a three-dimensional space to provide and visualize a unified answer to all these questions. In this section, first we start by presenting the three-dimensional space. Then, we discuss how interoperability is evaluated at each dimension.

1. Description of the Three-dimensional Space

a. Technology Dimension

This dimension evaluates the interoperability level of the technology used by the provider to build its own cloud, but does not consider the management capability or the tools used to implement policy. It evaluates the portability and mobility level of the provider's services; that is, whether cloud users are able to move their applications, data, and VMS without facing technical problems. We will use the following attributes to evaluate the interoperability of the provider's technology

- Virtualization technology
- Service architecture
- Security mechanism
- Data format

The interoperability level is considered low in cases where the virtualization technology, the security mechanism, the service architecture, or the data format prevents the interaction with other cloud services. Significant efforts (time and money) are needed to achieve interoperability. The interoperability level is considered medium in cases where users are able to move their data, applications and VMs in a down state, yet they cannot move them in an in-flight state. The provider services do not support cloud mobility. The interoperability level is considered high in cases where the provider's services support cloud portability and mobility. There may be minor issues, but they can be resolved easily without waste of time and money.

b. Management Dimension

This dimension evaluates only the provider management capabilities. It helps users to know whether the provider's management tools support cloud interoperability. As explained earlier in this thesis, the interoperability level of the management capabilities will be evaluated based on the following attributes:

- Management interface
- Provisioning/Scheduling capabilities
- Metering/Usage/Billing capabilities

- Monitoring/Reporting capabilities
- SLA and QoS
- Security (Does the provider have a mechanism for user authentication and data protection, and can users automatically evaluate the security level of this mechanism?)
- Auditing capability

Automation plays a critical role in the evaluation of management capabilities. We associate a low interoperability level when the provider management capability does not meet the basic requirements that support cloud interoperability, such as: the cloud provider does not provide a management interface that allows users to manage different cloud services; its services cannot be managed by third party tools; or the pricing model does not support cloud interoperability. A lot of manual interaction with the cloud provider is required to resolve management issues. A medium interoperability level is achieved when the cloud provider's management capabilities allow users to manage (provision and monitor) different cloud services in a unified way, or the provider services can be managed by third party tools. The provider still lacks automated tools that enable automatic SLA negotiation, security assessment, and auditing. A high level is achieved when the provider management capabilities is fully automated.

c. Policy Dimension

This dimension evaluates the interoperability level of the provider's policy. It helps users to assess whether this policy is able to interact with any other provider's policy without being locked because of geographical regulations or constraints imposed by the provider. Transparency, automation, and standardization play a critical role in the evaluation process.

A high level of interoperability is associated with policy that does not stand as barrier to cloud collaborations. The provider policy is expressed in a standardized way and can automatically interact with any other policy without being hindered by geographical regulations (e.g., universal agreement on legislation, best practices and standardization for the security policy). Attaining this level is still infeasible because it goes beyond the provider level to include political issues that should be tackled at the regional, national, and international level. The interoperability level is considered medium when the provider policy is able to interact automatically with any other policy within the same geographical boundaries. Legislation requirements still prevent wider cloud interoperation. The interoperability level is considered low when the provider's policy is not automated or still lacks information about either the provider internal policy or the geographical regulations.

2. Comparing Cloud Services Using the Three-dimensional Space

The three aforementioned dimensions help cloud users to assess the interoperability level of the cloud provider services based on the technology used to build the cloud, management capabilities, and policy. Users can consider using the three dimensions together, or they may use two or only one of the dimensions in case they want to make tradeoffs. For example, some cloud users may not care about the management capabilities, and they only want to avoid technical and policy issues while moving from one cloud to another. Therefore, the interoperability level of the provider's services can be defined by the following equation.

$$Interoperabily = \alpha * Ltech + \beta * Lmang + \gamma * Lpol \text{ where } \alpha + \beta + \gamma = 1$$

α, β, and γ are weights defined by the cloud user and Ltech, Lmang, and Lpol respectively represent the interoperability levels at each dimension, technology, management, and policy which can be low, medium, and high. The levels—low, medium, and high—can be replaced respectively by 1, 2, and 3 to quantify the interoperability levels so they can be compared easily.

IV. CASE STUDY: USING THE THREE-DIMENSIONAL SPACE TO COMPARE THE INTEROPERABILITY LEVELS OF TWO CLOUD PROVIDERS

Open source cloud-computing projects are gaining great attention. Unlike proprietary solutions, these projects are speeding the evolution and adoption of cloud computing. They help to save licensing costs and build an interoperable cloud environment, where users do not need to worry about vendor lock-in. Since the source code is open, either new or existing cloud providers can adopt successful cloud solutions, and reuse them in building their own clouds. This leverages cloud interoperability. Examples of these projects include: OpenStack, Eucalyptus, OpenNebula, and Nimbus. In this chapter, we will use the three-dimensional space presented in Chapter III to assess the interoperability level of two major IaaS cloud providers: OpenStack and OpenNebula. We start by providing an overview of the two cloud platforms provided by these providers, and then we discuss the interoperability level of each platform using one dimension at a time. We end this chapter with a comparison of the interoperability levels of the two cloud platforms.

A. OPENSTACK CLOUD

1. Overview

OpenStack is a cloud operating system developed by NASA and Rackspace for building private and public clouds licensed under Apache license version 2.0, and written in Python. More than 190 companies support this project (e.g., Dell, HP, IBM, Cisco, RedHat). The OpenStack project was originally composed of three separate components: a compute service known as Nova; an object storage service known as Swift; and an image service known as Glance. Other components were added to later releases. For now, Folsom release includes these additional components: Dashboard, referred to as Horizon; an identity service, known as Keystone; network service, known as Quantum; and block storage service, known as Cinder. As shown in Figure 4, these services interact with each other through RESTful APIs. Users can interact directly with these services through their APIs, or by using Dashboard. Keystone is the default authentication mechanism for all OpenStack services [39-41] .

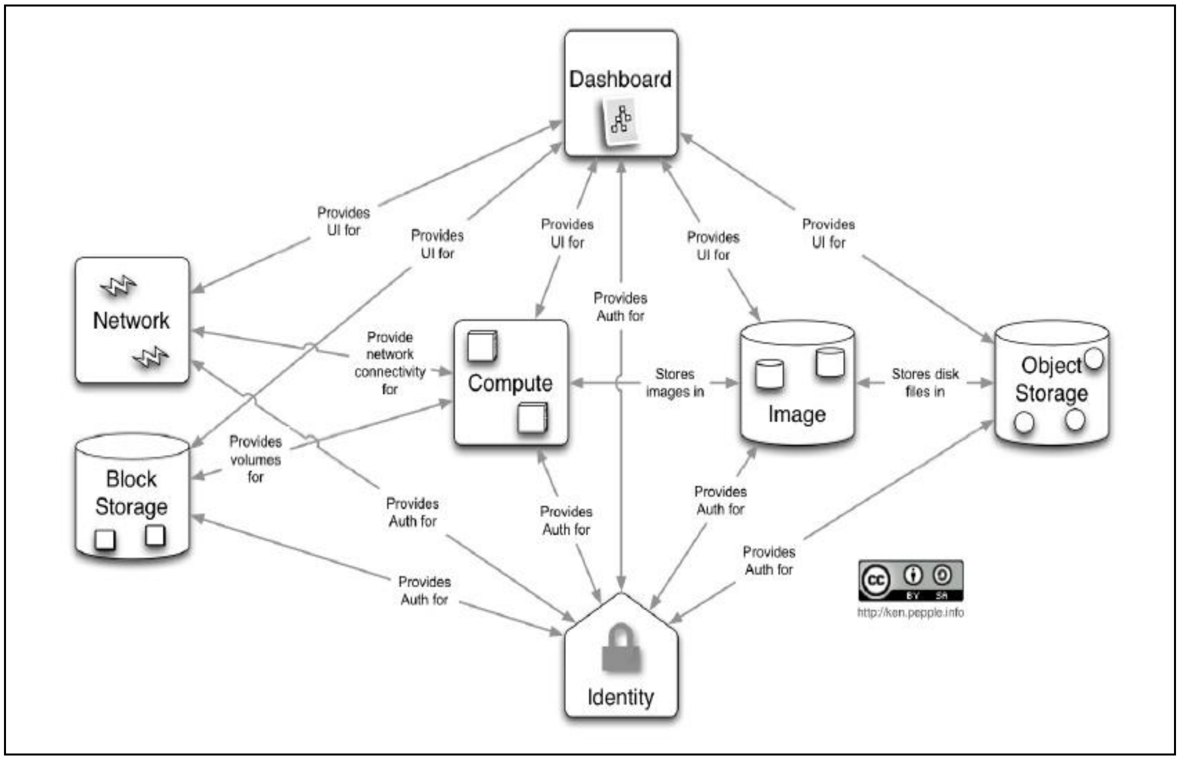

Figure 4. Overview of Openstack Architecture (From [40])

Compute: the compute service offers computing resources (e.g., network, server, CPU, memory) to the end users as virtual resources. It is implemented as RESTful API, which offers web service for the orchestration of cloud services. This API is hardware and hypervisor neutral. Nova components (API, compute, network, schedule, and queue) collaborate in order to respond to user requests. Access to the computing service requires authentication which can be done by default through the integration of Keystone services [42]. The compute services support the following virtualization solutions: Qemu, Xen, UML (User Mode Linux), VMware ESX/ESXi, LXC, and KVM [40].

Object Storage: offers web service implemented as a RESTful API [43] that has a distributed and decentralized architecture with multiple access points to create storage space for huge amount of data characterized mainly by its scalability and data replication to guarantee availability. Unlike traditional file system storage, object storage is for storing *static data, such as virtual machine images, photo storage, email storage, backups and archives* [44]. Users need connection credentials and authentication tokens to access OpenStack object storage. Authentication can be done through the OpenStack identity service, Keystone, or using middleware, such as swauth [45].

Image Service: a RESTful API that provides storage services, such as discovering, retrieving, creating, and storing virtual images. VM images retrieved or created using Glance support different storage locations starting from simple file systems to object storage location, like Swift project. While creating a new disk image, Glance service allows users to specify the format of the virtual disk image and its containers. The disk format can be raw (unstructured), vhd (common disk format supported by many virtual machine monitors, such as VMware, Xen, Microsoft, VirtualBox), VMDK (*common disk format supported by many virtual machine monitors*, such as VMware, VirtualBox, and QEMU), vdi (*disk format supported by VirtualBox and QEMU*), iso (*An archive format for the data contents of an optical disc*), qcow2 (*disk format supported by QEMU*), aki (Amazon kernel image)**,** ari (Amazon ramdisk image), and ami (Amazon machine image) and the container format can be bare *(no container or metadata envelope for the image*), OVF, aki, ami, and ari. Glance generates notifications whenever a virtual image is sent, uploaded, deleted or updated [46, 47].

Identity Service: implemented as a RESTful web service used by default as the authentication (authN) and high-level authorization (authZ) mechanism to access to OpenStack services. This service offers token service to authenticate OpenStack cloud users, a catalog service to provide the list of available OpenStack services and the locations of their associated endpoint APIs, and a policy service that provides a rule-based authorization engine and a management interface to manage the rules (verifying that the users have the privilege to perform actions). Different forms of authentication can be used with Keystone; users can authenticate using password and x.509 [48] credentials or tokens. It accepts SSL over HTTPS authentication requests. Keystone supports AWS's identity management system and can be updated to support *proxying external services and AuthN/AuthZ mechanisms such as oAuth, SAML and openID* [49-51].

Dashboard: a Django, web-based interface for the management of OpenStack services (compute, storage, and networking resources), it allows users to automatically provision their cloud services based on administration settings. It has an extensible design that easily allows the integration of third party tools, such as monitoring or billing tools. OpenStack services can be managed also through other management tools, such as EC2 compatibility API. Like any OpenStack service, dashboard users authenticate using the Keystone service [52].

Network Service: offers a pluggable and extensible architecture API to provide network connectivity as a web service. This service is managed by OpenStack compute services, and authentication is handled by default by Keystone service. The network service allows users to create their virtual networks and then assign interfaces to them. It supports many network vendor technologies [40, 53].

Block Storage: offers web services implemented as RESTful API. They interact with compute service, Nova, in order to create, attach, and detach a volume to a compute instance. The block storage functionalities were offered originally by the compute service component, nova-volume. In the Folsom release, block storage is also included in nova-volume. Block Storage includes a management tool, Snapshot, that allows data backup. It uses simple Linux server storage and supports many storage technologies (e.g., Ceph, NetApp, Nexenta and SolidFire). Block storage provides different types of storage: database storage, expandable file system storage, or raw block level storage. It supports S3 API and tries to achieve compatibility with other cloud APIs, such as EC2. It provides a mechanism for data protection and backup [44].

1. OpenStack Interoperability

Technology dimension: We attribute a medium interoperability level to the OpenStack technology. It is based on a collection of open source software solutions that are implemented as independent services. These services are implemented as RESTful APIs (except EC2 compatibility API [54]) that can be extended to include new features or integrate specific vendor solutions. These APIs do not mandate the use of any specific software or hardware. The adoption of the OVF standard enabled OpenStack to become hypervisor neutral; it supports all the existing virtualization technologies (e.g., QEMU, Xen, VMware ESX/ESXi, LXC, and KVM). OpenStack cloud supports various virtual-disk image formats, as stated earlier in the overview section about OpenStack in the paragraph about image service. Therefore, its virtualization technology does not hinder cloud portability. Live migration is possible only inside OpenStack clouds [40] (there is no documentation for live migration outside OpenStack cloud). The OpenStack authN/authZ mechanism supports external authN/authZ mechanisms. The stored data is replicated in different nodes. In the case of the failure of one node, data can be restored from the other nodes. OpenStack does not provide any tool to convert data from one format to another. However, it supports structured as well as unstructured data storage (e.g., database storage, raw block storage, expandable file system storage, emails, photos, backups). Consequently, with OpenStack technology, users need to care only about the live migration of their workloads to another cloud provider. The service architecture and openness of OpenStack will foster cloud interoperability.

Management: We attribute a medium interoperability level to the management capabilities of OpenStack. OpenStack offers a unified and user-friendly interface, Horizon, which allows users to manage their cloud services; they can access, control, and self-provision their cloud resources. The extensibility of this interface allows the integration of third-party tools, such as monitoring and billing tools. Other management tools for OpenStack cloud can be built using the native OpenStack API or EC2 API compatibility. However, OpenStack still lacks monitoring, billing, and reporting capabilities [55]. The upcoming release, Grizzly, aims to provide monitoring and metering capabilities. A project named Ceilometer is under development to achieve these objectives. The first version of this project was released in October 2012, yet it is still not included in the OpenStack project due to its infancy [56, 57]. OpenStack users use the Keystone service as the default authN/authZ mechanism, and the stored data is replicated to guarantee its availability and protection. Like other cloud providers, OpenStack is still unable to provide a reliable and robust security mechanism for the authN/authZ and the protection of user data. For instance, it does not check password complexity or provide data encryption tools [58]. Users need to rely on themselves to encrypt and manage their encryption keys. OpenStack cloud-management capabilities are still not fully automated to provide automatic engines that allow users to assess the security level of cloud services, generate auditing reports, or support SLA negotiations based on QoS.

Policy: OpenStack does not provide any engine that allows the automatic implementation of policy so that automatic interaction among different provider policies will be possible. Therefore, we consider its interoperability level low.

B. OPENNEBULA CLOUD

1. Overview

OpenNebula started as a research project in 2005, and evolved into an open source project directed by C12G labs. It is a software solution for building private, public, and hybrid cloud, licensed under Apache license version 2.0 and written in Java and Ruby. It has an active research community that includes individuals as well as organizations. The use of this software is not limited to research communities. It is used by hosting and cloud services providers, telecommunication companies, aerospace companies, IT vendors, etc. [16].

As shown in Figure 5, OpenNebula [59-62] internal architecture can be divided into three layers: drivers, core, and tools.

Core: the core is responsible for the control and monitoring of VMs, virtual networks, storage and physical hosts. It is written in highly optimized C++ code. The core does its job by invoking the appropriate driver. It is composed mainly of the following components:

- **Request Manager**: handles client requests.
- **Virtual Machine Manager**: manages and monitors VMs.
- **Transfer Manager:** manages VM images. It guarantees the transfer of all the files required to deploy the VM.
- **Virtual Network Manager**: manages virtual networks. It keeps track of the IP and MAC addresses assigned to networks and their associated VMs, and the physical bridges where the VMs are residing.
- **Host Manager**: manages and monitors physical hosts. This task is performed by invoking the appropriate driver.
- **Database**: OpenNebula internal data structure is based on a scalable and reliable backend SQLite database.

Drivers: pluggable modules that are designed to interact with other cloud technologies (e.g., virtualization, network, monitoring, authentication).

Tools: Management tools distributed with OpenNebula, or provided by a third-party to interact with OpenNebula, such as CLI, scheduler, OCCI and EC2 interfaces. The scheduler is implemented as an independent, generic component responsible for the placement of the VMs based on information provided by the running VMs and the physical resources. It can be substituted by third-party tools.

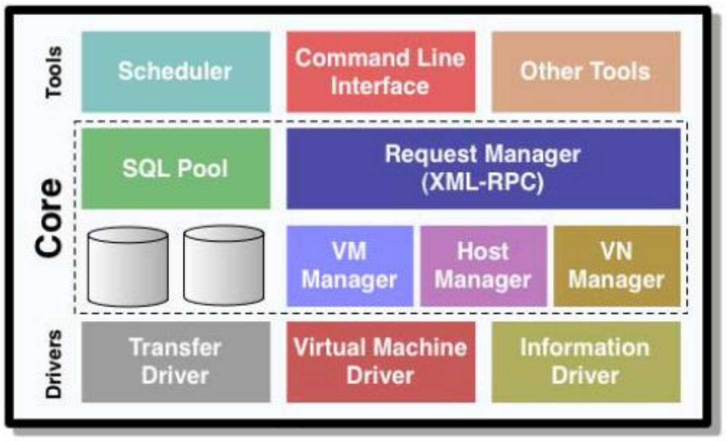

Figure 5. Overview of OpenNebula Architecture (From [60])

OpenNebula offers different interfaces [63], [64] that allow end users to interact with the OpenNebula, cloud or any other cloud infrastructure:

Interfaces for cloud consumers: OpenNebula is distributed with three cloud interfaces to allow cloud consumers to manage their cloud resources: OCCI, EC2 and EBS interfaces, and a self-service portal. The EC2 interface aims to achieve compatibility with EC2. It allows users to request EC2 resources and manages these resources in an EC2-like way (e.g., upload images, register them, run, monitor, terminate). The OCCI interface is a RESTful service, implemented using the OCCI specifications, to allow users to create, control, and monitor their cloud resources. It includes extensions

requested by the OpenNebula community. The self-service portal is a user-friendly GUI designated specifically to non-IT users to allow them to access and manage their cloud resources easily.

Administrator and advanced-user interfaces: Cloud administrators and advanced users can interact with OpenNebula cloud using either CLI or GUI. The GUI referred to as Sunstone can also be used by regular users. SunStone facilitates the management of the cloud resources in hybrid and private clouds (e.g., visualize the usage of cloud resources, provides statistical functionalities like cloudwatch, VNC support, different system views for different roles, catalog access). It can be customized or extended using plugins.

System interfaces: These interfaces aim to customize OpenNebula in order to fit any cloud infrastructure. OpenNebula is distributed with RPC XML and OCA interfaces. The RPC XML interface is the main interface to interact with OpenNebula; it allows the management of any OpenNebula resource. The OCA interface provides the same functionalities as the RPC XML interface; it aims to facilitate interaction with the OpenNebula core.

OpenNebula marketplace: This is a catalog containing third-party appliances that can be used within the OpenNebula cloud. It can be accessed using SunStone or CLI. This catalog contains only the appliance metadata [65]**.**

3. OpenNebula Interoperability

Technology dimension: We attribute a medium interoperability level to OpenNebula clouds. It has a flexible and modular architecture based on pluggable drivers, which allows it to interact with any storage, network, and virtualization technology. New pluggable drivers can be developed using any language to fit OpenNebula to any cloud infrastructure. Private clouds can be combined with public cloud either for cloud federation or bursting (responding to an increase of demands by allocating resources from other clouds). Since version 3.2, OpenNebula supports live migration of workload inside OpenNebula clouds [66], yet it is still unable to allow the live migration to a different cloud. OpenNebula comes with the standard login credentials (username and password), and ACLs are used as an authorization mechanism; users can manipulate cloud resources according to predefined ACLs. This authN/authZ mechanism supports SSH RSA keypairs, X509, LDAP or AD, as well as other external authN/authZ mechanisms [67]. OpenNebula has a modular storage system that can be extended to support any storage technology. Unlike, OpenStack, the OpenNebula storage system is suitable for storing virtual instances, instead of static data [68]. As of this writing, this system is distributed with five different datastore types (referred to earlier as image repository types) [69]:

- System: temporarily stores images of running VMs. An image can be a complete copy of the original image, qcow deltas, or simple file system links.
- File system datastore: stores a VM image in file format. It supports any file format depending on the targeted hypervisor.
- iSCSI/LVM: stores VM in block devices instead of a file format
- VMware: is a datastore for the VMware hypervisor for the VMDK format
- VMFS: is a datastore that can be used with the VMware hypervisor to handle VMFS format

Management dimension: We assign a medium interoperability level to OpenNebula Management capabilities. OpenNebula provides different GUI and CLI interfaces with which cloud users can interact with OpenNebula and manage their cloud resources: the OCCI and EC2 compatible interfaces, a self-service portal designated for non-IT users, Unix-like CLI, Suntone GUI, XML RPC interface, and OCA interface. These interfaces satisfy different levels of cloud users, from non-IT users to developers. Sunstone and self-portal are convenient GUIs that allow any cloud users to access and manage cloud resources in a convenient and unified way. Since OpenNebula has a pluggable architecture, external cloud resources are managed in the same way as other local resources [70]. Users are able to dynamically provision their cloud resources, which can be leased from public clouds to respond to increases in demand. The scheduler keeps track of the leased cloud resources. Haizea, *open source virtual machine-based lease management architecture* [71], can be used as a scheduler to offer advanced scheduling features, such as best-effort lease and advanced reservation capacity [72]. The OpenNebula accounting system visualizes and generates reports about the usage of cloud resources. This system is designed to be integrated with external chargeback and billing systems [73]. OpenNebula has an internal monitoring system that controls the cloud resources. It can be integrated with other data centers' monitoring tools, such as Ganglia, to improve performance, especially when OpenNebula is deployed in large enterprises [74].

However, OpenNebula monitoring capabilities are limited to gathering information about cloud resources (usage and state). It still lacks advanced functionalities, such as sending alerts when there is degradation in QoS or a dramatic increase in usage, or security breaches. While OpenNebula offers a mechanism to control user authentication and authorization which supports external authZ/authN mechanisms, it does not provide a mechanism to guarantee the integrity of a user's data (e.g, encryption tools). This is important, especially when private and public clouds are combined. OpenNebula management capabilities are still not fully automated to support SLA negotiation based on QoS and to provide engines that allow users to generate auditing reports and assess the security level of cloud services. The pluggable architecture of OpenNebula can compensate for all these deficiencies by integrating third-party tools, or users can develop their own applications, though we mainly restrict this to professional cloud users (e.g., developers).

Policy dimension: OpenNebula does not provide any automatic engine that allows hosting cloud providers to implement policy so that automatic interaction among cloud providers' policies will be possible. Therefore, we consider its interoperability level low.

C. COMPARISON OF OPENSTACK AND OPENNEBULA

Considering the predefined attributes in the evaluation of the interoperability level of Openstack and OpenNebula, we attribute the same interoperability levels at each dimension to both platforms (Technology: medium; Management: medium; policy: low). The two platforms do not support live migration in hybrid clouds and lack management capabilities and automatic implementation of the provider's policy.

However, this does not mean that OpenNebula and OpenStack have equivalent interoperability levels. Table 1 shows the main differences and similarities between the two platforms. Now, we need to visualize the interoperability level of each platform in the three-dimensional space so that cloud users can select the more suitable platform for cloud interoperation based on their needs.

Table 1. Comparison of OpenStack and OpenNebula

	OpenStack	**OpenNebula**
Deployment model	Public and private cloud	Public, private, and hybrid
Service model	IaaS	IaaS
License	Apache version 2.0	Apache version 2.0
Architecture	Modular and flexible architecture: RESTful APIs implemented as independent web services which can operate as standalone services or collaborate with each other.	Modular and flexible architecture: Driver-based architecture; drivers to communicate with any cloud infrastructure.

Security mechanism (authentication and authorization)	Password credentials, tokens, LDAP, x.509 credentials, and AWS style logins;
	Role-based Access Control (RBAC) for user authorization;
	Supports external authN/authZ mechanisms.
		Password credentials, SSH rsa keypairs, x.509 credentials, LDAP, and AD;
		ACLs for user authorization;
		Supports external authN/authZ mechanisms.

Hypervisor	Broad hypervisor support, such as Qemu, Xen, VMware ESX/ESXi, LXC, and KVM. Can be extended to integrate any hypervisor type.	Xen, KVM, and VMware. Can be extended to integrate any hypervisor type.
Migration	Cold migration (on down state); Live migration within OpenStack clouds.	Cold migration; Live migration within OpenNebula clouds.
Storage system	Block and object storage; structured as well as unstructured data storage (e.g., database storage, raw block storage, expandable filesystem storage, emails, photos, backups).	Suitable for storing virtual instances instead of static data ; Distributed with five different datastores: System, filesytem, iSCSI/LVM, Vmware, and

	OpenStack	**OpenNebula**
		VMFS.
User interface	Extensible web interface referred to as Dashboard that allows users to manage and self-provision their cloud resources.	EC2, OCCI, XML RPC, OCA, Self-service portal (SunStone), and CLI to interact with any cloud infrastructure, access, self-provision and manage hybrid cloud resources.
Compatibility with public clouds	EC2 and S3	EC2
Compatibility with interoperability standards	OVF, and OCCI(still a work in progress)	OCCI

1. Visualization of Interoperability in the Three-dimensional Space

Technology dimension: Both platforms have a modular and flexible architecture that can be extended or integrated with third-party tools. However, unlike OpenStack, Opennebula has a driver-based architecture. This means that new drivers are required to communicate with any new cloud technology (compute, network, authN/authZ, and storage). As stated in [2], developing new drivers is not an easy task, especially from non-IT, cloud-user perspective. Cloud developers also may not be able to keep up with technological development, considering the increasing number of cloud providers. OpenStack exposes all its functionalities as web services. These web services can work as standalone services or collaborate together. These services can be published in a public registry, so they can be accessed and reused by any cloud user. At the same time, OpenStack users can use any publicly available web service offered by any other cloud provider without the need of developing a driver. Therefore, OpenStack technology has higher interoperability level than OpenNebula.

Management dimension: As of this writing, the management capabilities of both platforms are not fully automated, and still lack advanced monitoring features. They both offer user-friendly interfaces that allow cloud users to access and manage their cloud resources, yet these management capabilities are limited to controlling virtual resources and monitoring their usages and states. OpenNebula’s capabilities better fit the management requirements of cloud interoperation. Within an OpenNebula cloud, any external resource is managed as a local resource. The integration of Haizea allows OpenNebula to offer the best provision capabilities [72]. Unlike Openstack, which is still working on a billing system called Ceilometer, OpenNebula offers an accounting system that can be combined with any external or chargeback system, and can be integrated with the Ganglia monitoring system to improve monitoring performance, in case OpenNebula is deployed in large enterprises.

Policy dimension: OpenNebula and Openstack lack automated engines to implement policy. Both platforms are still not mature enough to support the automatic implementation of policy, so cloud interoperation cannot be hindered because of the provider’s policy or legislative requirements.

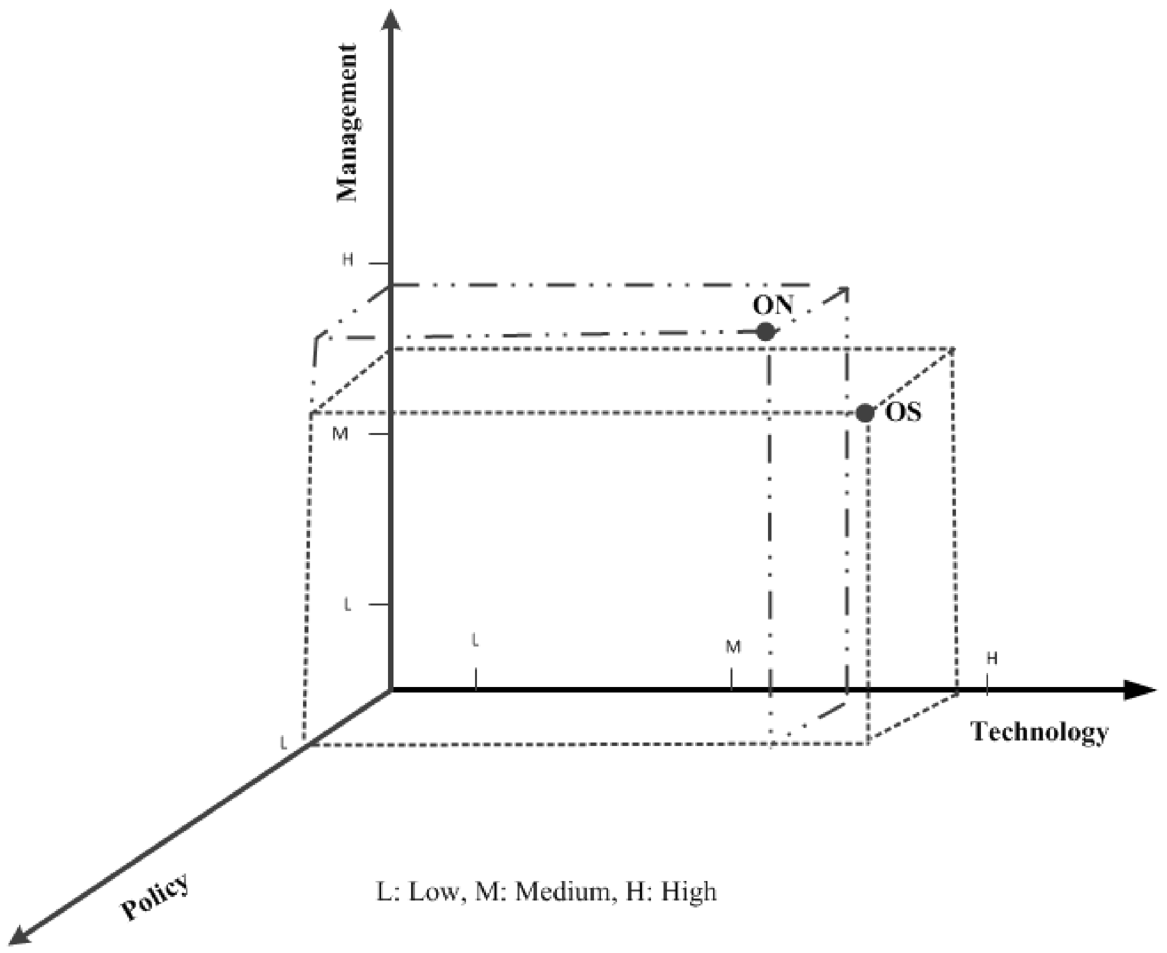

Figure 6. OpenStack (OS) and OpenNebula (ON) Interoperability

The visualization of the interoperability, Figure 6, shows that OpenNebula management capabilities are more suitable for the management of hybrid clouds and that OpenStack offers more flexible and modular architecture for building an interoperable IaaS architecture. Neither OpenStack nor OpenNebula is yet mature enough to offer a solution for the automatic implementation of cloud policy. The visualization of the interoperability in the three-dimensional space enables cloud users to select the platform that better fits their needs and, at the same time be aware of the limitation of their selection. For instance, OpenNebula is the best choice for cloud users who care about cloud management and OpenStack is the best choice for users who do not want to be concerned with the development of drivers or the management of hybrid clouds.

V. CONCLUSIONS AND RECOMMENDATIONS

A. THESIS SUMMARY

The adoption of cloud computing is a paramount step toward the evolution and maturation of this embryonic technology. Users will always be reluctant to move to the cloud until cloud interoperability is achieved; users want to be able to move their data and applications from one cloud to another and manage and monitor their cloud resources without difficulty.

Therefore, we focused in this thesis on building a framework to assess the interoperability level of cloud provider in order to help users select the provider's services that fit better their interoperability needs. We classified interoperability requirements into three categories: technology, management, and policy. Then we identified the attributes of each category. These categories were used later as the three dimensions of a trade-off space that we defined to visualize the provider interoperability levels. We ended this thesis work by providing a case study showing how the three-dimensional space can be used to compare interoperability levels of cloud providers and the value of the graphical visualization of the results in that space.

B. LIMITATIONS AND RECOMMENDATIONS

Cloud computing is an evolving technology without a standard definition or widely adopted de-jure or de-facto standards. The evolution of this technology may change the interoperability requirements and mandate new ones. Therefore, we recommend a revision of this work in the future when cloud computing becomes more mature.

There are factors that cloud users should consider: the scope of the support provided by the cloud provider to customers, and the improvement of the provider's services over time. First, a cloud provider may take charge of resolving all or specific interoperability issues that customers encounter. In this case, although the provider's interoperability level is low, users needn't worry about these issues. Second, cloud providers who are not updating their services cannot maintain their interoperability level.

For example, after two development cycles (6 months for both OpenStack and OpenNebula), OpenStack's management capabilities may surpass those of OpenNebula.

We also recommend that cloud users consider open source cloud solutions. Unlike proprietary solutions, open source solutions can increase cloud interoperability since they can be used in any cloud implementations without incurring restrictions (freedom of modification and extension).

C. FUTURE WORK

Cloud computing is still not mature enough to overcome challenging issues that hinder building an interoperable cloud ecosystem. This thesis work helped us to reveal these issues (e.g., no SLA negotiation based on QoS, monitoring capabilities limited to monitoring the state and usage of the cloud resources, un-automated provider's policy). In this section, we highlight some areas of interest that need more attention from academia, research, and industry to provide solutions to these issues and, therefore, build interoperable clouds.

1. Cloud Auditing

Cloud users would rather be locked into a cloud provider that they trust, rather than one they do not. Cloud users (especially normal users) generally do not have enough expertise and information to evaluate provider services. Looking to protect their assets, cloud providers hide their weaknesses (e.g., security breaches, and vulnerabilities in their systems). Therefore, having a trusted third-party responsible for auditing cloud providers is a key enabler to cloud interoperation. Despite research efforts done in cloud auditing, there is still no trusted party responsible for cloud auditing. More work is still required in this field to establish an auditing authority that provides near real-time auditing reports, including especially information users need to know about the cloud provider's services (e.g., security, QoS, reliability).

2. SLA and QoS

Cloud interoperation requires that a cloud provider be able to provide services that satisfy different customers' demands. Cloud providers still use a static SLA that all users must obey in order to use their services. There is still no automatic engine that allows automatic SLA negotiation based on QoS, even within the same provider. Much research has been focused on this issue, yet there is still no practical implementation. This provides interesting work that should be considered in the future, in order to stimulate cloud interoperability.

3. Policy

Overcoming all the technical issues that hinder cloud interoperation and having cloud providers agree on a standardized and automatic engine that allows the automatic interaction among different providers' policies would not enable worldwide cloud interoperation. We strongly believe that legislation requirements, especially at the international level are the main hindrance to cloud interoperability. Standardized and international legislation that governs world issues related to cloud computing (e.g., privacy, and liability) is required. Although this seems impossible to achieve, we strongly believe that starting discussion on this topic will lead to fruitful results in the future.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] T. Rings, J. Grabowski and S. Schulz, “A testing framework for assessing grid and cloud infrastructure interoperability,” *International Journal on Advances in Systems and Measurements*, vol. 4, pp. 95–108, 2011.

[2] J. Ejarque, J. Alvarez, R. Sirvent, and R. M. Badia, “A rule-based approach for infrastructure providers’’interoperability,” in *2011 IEEE 3rd Int. Conference,* INT. 2011, pp. 272–279.

[3] N. Loutas, E. Kamateri, F. Bosi, and K. Tarabanis, “Cloud computing interoperability: The state of play,” in *2011 IEEE 3rd Int. Conference on Cloud Computing Technology and Science*, INT. 2011, pp. 752–757.

[4] R. Teckelmann, C. Reich, and A. Sulistio, “Mapping of cloud standards to the taxonomy of interoperability in IaaS,” in *2011 IEEE 3rd Int. Conference on Cloud Computing Technology and Science*, 2011, pp. 522–526.

[5] L. Youseff, M. Butrico, and D. Da Silva, “Toward a unified ontology of cloud computing,” in *Grid Computing Environments Workshop, 2008. GCE'08*, 2008, pp. 1–10.

[6] R. L. Grossman, “The case for cloud computing,” *IT Professional*, vol. 11, pp. 23–27, 2009.

[7] L. Wang, G. Von Laszewski, A. Younge, X. He, M. Kunze, J. Tao, and C. Fu, “Cloud computing: a perspective study,” *New Generation Computing*, vol. 28, pp. 137–146, 2010.

[8] P. Mell and T. Grance, “The NIST definition of cloud computing,” *NIST Special Publication* 800–145, 2011.

[9] C. T. Yang, S. F. Wang, K. L. Huang, and J. C. Liu, “On construction of cloud IaaS for VM live migration using KVM and OpenNebula,” *Algorithms and Architectures for Parallel Processing*, pp. 225–234, 2012.

[10] F. Carsten, “Infrastructure suitability assessment modelling for cloud computing solutions,” M.S. thesis, Naval Postgraduate School, Monterey, CA, September 2011.

[11] R. Nyrén, A. Edmonds, A. Papaspyrou, and T. Metsch, “Open cloud computing interface–core,” OCCI-WG, Rep. GFD-P-R.183, 2011. Available: http://www.ogf.org/documents/GFD.183.pdf.

[12] Eucalyptus Systems. The eucalyptus cloud [Online]. Available: http://www.eucalyptus.com/eucalyptus-cloud/iaas.

[13] Open Cloud Computing Interface. OCCI in RESERVOIR [Online]. Available, http://occi-wg.org/2011/03/22/occi-reservoir/.

[14] OpenStack LLC. OpenStack: The Open Source Operating System [Online]. Available: http://www.openstack.org/software/.

[15] Anonymous. Welcome to the Claudia platform [Online]. Available: http://claudia.morfeo-project.org/.

[16] OpenNebula Project. About the OpenNebula.org project [Online]. Available: http://opennebula.org/about:about.

[17] Anonymous. The virtualization API [Online]. Available: http://libvirt.org/.

[18] Distributed Management Task Force, Inc. (January 12, 2012). Open virtualization format specification [Online]. Available: http://dmtf.org/sites/default/files/standards/documents/DSP0243_1.1.0.pdf.

[19] Storage Networking Industry Association. (June 4, 2012). Cloud data management interface [Online]. Available: http://snia.org/sites/default/files/CDMI%20v1.0.2.pdf.

[20] Cloud-standards.org. Welcome to the cloud standards wiki [Online]. Available: http://cloud/standards.org/wiki/index.php?title=Main_Page.

[21] M. Hogan, F. Liu, A. Sokol and J. Tong. NIST cloud computing standards roadmap [Online]. Available: http://www.nist.gov/customcf/get_pdf.cfm?pub_id=909024.

[22] D. Petcu, "Portability and interoperability between clouds: challenges and case study," *4th European Conf., ServiceWave 2011, Poznan, PL, 2011*, pp. 62–74.

[23] R. Cohen.(February 27, 2009). Examining cloud compatibility, portability and interoperability [Online]. Available: http://www.elasticvapor.com/2009/02/examining-cloud-compatibility.html.

[24] M. Becker, "Interoperability Case Study Cloud Computing," Berkman Center, Cambridge, MA. Rep.2012-11, 2012.

[25] S. Dowell, A. Barreto, J. B. Michael and M. T. Shing, "Cloud to cloud interoperability," in *Proc. of the 2011 6th Int. Conference* on *System of Systems Engineering*. Albuquerque, New Mexico, 2011, pp. 258–263.

[26] T. Dillon, C. Wu and E. Chang, "Cloud computing: Issues and challenges," in *2010 24th IEEE Int. Conference on Advanced Information Networking and Applications*, 2010, pp. 27–33.

[27] R. Moreno-Vozmediano, R. Montero and I. Llorente, “Key Challenges in Cloud Computing to Enable the Future Internet of Services,” *Internet Computing, IEEE,* vol.PP, no.99, 2012.

[28] D. Bernstein, E. Ludvigson, K. Sankar, S. Diamond and M. Morrow, “Blueprint for the intercloud—protocols and formats for cloud computing interoperability,” in *2009 4th Int. Conference on Internet and Web Applications and Services,* 2009, pp. 328–336.

[29] M. Papazoglou, “Cloud blueprints for integrating and managing cloud federations,” *Software Service and Application Engineering,* pp. 102–119, 2012.

[30] K. Oberle and M. Fisher, “ETSI CLOUD–Initial Standardization Requirements for Cloud Services,” *Economics of Grids, Clouds, Systems, and Services*, pp. 105–115, 2010.

[31] B. Claybrook. Application portability: What enterprises need to know [Online]. Available: http://searchcloudapplications.techtarget.com/feature/Application-portability-What-enterprises-need-to-know.

[32] J. Urquhart. (May 7, 2009). Exploring cloud interoperability, part 2 [Online]. Available: http://news.cnet.com/8301-19413_3-10235492-240.html.

[33] J. Bozman and G. Chen, “Cloud Computing: The Need for Portability and Interoperability,” *IDC Executive Insight,* 2010.

[34] F. Paraiso, N. Haderer, P. Merle, R. Rouvoy and L. Seinturier, “A federated multi-cloud PaaS infrastructure,” in *2012 IEEE 5th Int. Conf. on Cloud Computing,* 2012, pp. 392–399.

[35] A. Govindarajan, “Overview of Cloud Standards,” *Cloud Computing,* pp. 77–89, 2010.

[36] R. Mietzner and F. Leymann, “Towards provisioning the cloud: On the usage of multi-granularity flows and services to realize a unified provisioning infrastructure for SaaS applications,” in *2008 IEEE Congress on Services 2008 - Part I,* 2008, pp. 3–10.

[37] J. S. Park, E. Spetka, H. Rasheed, P. Ratazzi and K. J. Han, “Near-real-time cloud auditing for rapid response,” in *2012 26th Int. Conference on Advanced Information Networking and Applications Workshops,* 2012, pp. 1252–1257.

[38] S. Harris, “Information security and risk management,” in *CISSP Certification all-in-One Exam Guide.* New York: McGraw-Hill, 2007, pp. 45–152.

[39] Racksapce. OpenStack$^{®}$: The open alternative to cloud lock-in invented by rackspace and NASA [Online]. Available: http://www.rackspace.com/cloud/openstack/.

[40] OpenStack LLC. (November 9, 2012). OpenStack compute administration manual [Online]. Available: http://docs.openstack.org/folsom/openstack-compute/admin/bk-compute-adminguide-folsom.pdf.

[41] O. Sefraoui, M. Aissaoui and M. Eleuldj, "OpenStack: toward an open-source solution for cloud computing," *Int. Journal of Computer Applications*, vol. 55, pp. 38–42, 2012.

[42] OpenStack LLC . (May 30, 2012). OpenStack compute developper guide [Online]. Available: http://docs.openstack.org/api/openstack-compute/2/os-compute-devguide-2.pdf .

[43] OpenStack LLC. (September 4, 2012). OpenStack object storage administration guide [Online]. Available: http://docs.openstack.org/api/openstack-object-storage/1.0/os-objectstorage-devguide-1.0.pdf.

[44] OpenStack LLC. OpenStack storage [Online]. Available: http://www.openstack.org/software/openstack-storage/.

[45] OpenStack LLC. (November 9, 2012). OpenStack object storage administration manual [Online]. Available: http://docs.openstack.org/folsom/openstack-object-storage/admin/content/index.html.

[46] OpenStack LLC. Disk and container formats [Online]. Available: http://docs.openstack.org/developer/glance/formats.html.

[47] OpenStack LLC. Welcome to Glance's documentation! [Online]. Available: http://docs.openstack.org/developer/glance/.

[48] G. von Laszewski, J. Diaz, F. Wang and G. C. Fox, "Comparison of multiple cloud frameworks," in *2012 IEEE Fifth Int. Conference on Cloud Computing, 2012*, pp. 734–741.

[49] OpenStack LLC. (August 29, 2011). OpenStack Identity Developper Guide (API v2.0) [Online]. Available: http://docs.openstack.org/api/openstack-identity-service/2.0/content/identity-dev-guide-2.0.pdf.

[50] OpenStack LLC. OpenStack shared services [Online]. Available: http://www.openstack.org/software/openstack-shared-services/.

[51] OpenStack LLC. OpenStack identity ("keystone") [Online]. Available: http://wiki.openstack.org/Keystone.

[52] OpenStack LLC. Horizon: The OpenStack dashboard project [Online]. Available: http://docs.openstack.org/developer/horizon/.

[53] OpenStack LLC.(August 17, 2012). Quantum API guide (v2.0) [Online]. Available: http://docs.openstack.org/api/openstack-network/2.0/content/.

[54] X. Wen, G. Gu, Q. Li, Y. Gao and X. Zhang, "Comparison of open-source cloud management platforms: OpenStack and OpenNebula," in *2012 9th Int. Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2012)*, 2012, pp. 2457–2461.

[55] L. L. Ted Chamberlin. Magic quadrant for managed hosting [Online]. Available: http://www.gartner.com/technology/reprints.do?id=1-19K9FPH&ct=120305&st=sb.

[56] OpenStack LLC. Ceilometer [Online]. Available: https://launchpad.net/ceilometer.

[57] OpenStack LLC. Welcome to the ceilometer developer documentation! [Online]. Available: http://docs.openstack.org/developer/ceilometer/.

[58] P. Cigoj and T. Klobučar, "Cloud security and OpenStack," in *The 1st Int. Conf. on CLoud Assisted ServiceS*, 2012, pp. 20.

[59] G. Toraldo, "OpenNebula and why it matters?" In *Opennebula 3 Cloud Computing*, Birmingham, UK*:* Packt Publishing, May 2012.

[60] P. OpenNebula. OpenNebula 2.0 architecture [Online]. Available: http://opennebula.org/documentation:archives:rel2.0:architecture.

[61] Y. Zhao, Y. Zhang, W. Tian, R. Xue and C. Lin, "Designing and deploying a scientific computing cloud platform," in *2012 ACM/IEEE 13th Int. Conf. on Grid Computing*, 2012, pp. 104–113.

[62] T. Cordeiro, D. Damalio, N. Pereira, P. Endo, A. Palhares, G. Gonçalves, D. Sadok, J. Kelner, B. Melander and V. Souza, "Open source cloud computing platforms," in *2010 Ninth Int. Conference on Grid and Cloud Computing*, 2010, pp. 366–371.

[63] OpenNebula Project. An overview of OpenNebula 4.0 [Online]. Available: http://opennebula.org/documentation:rel4.0:intro.

[64] OpenNebula Project. Scalable architecture and APIs 3.8 [Online]. Available: http://opennebula.org/documentation:rel3.8:introapis.

[65] OpenNebula Project. Interacting with the OpenNebula marketplace [Online]. Available: http://opennebula.org/documentation:rel3.8:marketplace.

[66] OpenNebula Project. OpenNebula 3.2 (codename red spider) [Online]. Available, http://opennebula.org/software:rnotes:rn-rel3.2.

[67] OpenNebula Project. External auth overview 4.0 [Online]. Available: http://opennebula.org/documentation:rel4.0:external_auth.

[68] X. Gao, P. Shah, A. Yoga, A. Kodgire and X. Ni. Cloud storage survey [Online]. Available: http://salsahpc.indiana.edu/b649/collection/2010/prj%23F/akodgire_prj%23F.pdf.

[69] OpenNebula Project. Storage overview 3.8 [Online] Available: http://opennebula.org/documentation:rel3.8:sm.

[70] OpenNebula Project. An overview of OpenNebula 3.8 [Online]. Available: http://opennebula.org/documentation:rel3.8:intro.

[71] OpenNebula Project. Haizea [Online]. Available: http://opennebula.org/software:ecosystem:haizea.

[72] A. Innocent, "Cloud infrastructure service management-a review," *International Journal of Computer Science Issues*, vol. 9, pp.287–292, 2012.

[73] OpenNebula Project. Accounting client 4.0 [Online]. Available: http://opennebula.org/documentation:rel4.0:accounting.

[74] OpenNebula Project. Ganglia monitoring 4.0 [Online]. Available: http://opennebula.org/documentation:rel4.0:ganglia.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Man-Tak Shing
   Naval Postgraduate School
   Monterey, California

4. Albert Barreto III
   Naval Postgraduate School
   Monterey, California

5. Dan Boger
   Naval Postgrade School
   Monterey, California

6. Dr. Bret Michael
   Naval Postgraduate School
   Monterey, California

7. Mr. Tao Rocha
   SPAWAR Atlantic
   Charleston, South Carolina

8. CDR. Kurt Rothenhaus
   PEO C4I, PMW/A 170
   San Diego, California

9. Charles Suggs
   SPAWAR PEOC4I
   San Diego, California

10. Nancy J. Kelley
    SPAWAR Sytems Center - Pacific
    San Diego, California