ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 6, June 2013

A way to cloud computing basic to multitenant environment

Twinkle Garg[1], Rajender Kumar[2], Jagtar Singh[3]

Research Scholar, Department of CSE, H.C.T.M Kaithal, India [1]

Assistant Professor, Department of CSE, H.C.T.M Campus, Kaithal, India [2]

Assistant Professor, Department of CSE, H.C.T.M Campus, Kaithal, India [3]

Abstract: Cloud Computing is evolving as a key computing platform for sharing resources that include infrastructures, software, applications, and business processes. Virtualization is widely used in the cloud computing, it is a core technology for enabling cloud resource sharing. Virtualization is used to virtualize a single instance to the multiple instances. However, most existing Cloud Computing platforms have not formally adopted the service-oriented architecture (SOA) that would make them more flexible, extensible, and reusable. This paper discusses the concept of cloud computing in single tenant and multitenant environment. Three layered architecture is very useful to understand the overall concept of cloud computing. Multitenant applications are very useful for the increased utilization of hardware resources and improved ease of maintenance. All these benefits of multitenancy shoud result in lower application coasts and give the major benefit to small and medium enterprises (SME). The main focus on this paper is to describe the overall concept of cloud computing to the multitenant application.

Keywords: Cloud Computing, Virtualization, Single Tenant Application, Multitenant Application, Layered Requirement Model.

I. INTRODUCTION

Cloud computing is a model of "on-demand" service delivery and access where dynamically scalable and virtualized resources are provided as service over the internet. Cloud computing is mainly used for the management of hardware and software resources to reduce the cost associated with it [3]. These services are broadly divided in to three categories:-

A. Infrastructure as a Service (IaaS)
B. Platform as a Service (PaaS)
C. Software as a Service (SaaS)

A. Infrastructure as a service

IaaS is also termed as Hardware as a Service. It provides infrastructure on rent such as server space, network equipment, storage space, data center space, Memory and CPU cycles as service. The benefit of IaaS is that instead of investing in building up and maintaining the hardware resources, just rent them. It takes the cost down as we just have to pay for the operational cost not capital expenditure and also based on the usage we can scale up and scale down the resources. This is called dynamic scaling and it helps in maintaining up the cost for running an organization. The way is the resources are consumed basically decides the billing.

B. Platform as a service

PaaS is an application delivery model in which the service provider distributes more than the infrastructure. PaaS delivers all the resources which are needed by the developer for building an application from scratch without downloading or installing the software locally on their system. It is considered as the advancement of the Web Hosting. As in the last few years, many web hosting companies are providing software for designing and building up the websites. The services offered by the PaaS are designing, developing, testing, deployment and hosting of the application. Simple Object Access Protocol (SOAP) and Representational State Transfer (REST) are also supported by the PaaS for the web development as they are useful in building web services. The problem with the PaaS applications are that different service providers lack portability and interoperability and also the service providers can just offer a particular development environment which makes little difficult for the developer to develop application. The main problem with PaaS is that if the service provider gets bank corrupt and run out of business, all the application and data of the users will also be lost. Due to this only, a PaaS service provider company called Zimki, which started in 2006 and by the mid of 2007 it was of out of business. Figure 1 show the platform as a service model where the service provider is providing its services.

Company	Platform
Mosso	PHP, .NET, Java, rails, Python
Google App Engine	Python
SalesForce	Proprietary
Morph	Ruby on Rails
Heroku	Ruby on Rails

Fig 1. Different companies name which are providing PaaS application

C. Software as a service

The third kind of service model provided by the cloud computing is of providing software and application as a service. SaaS is the basic service model which leads to the evolution of whole idea of cloud computing. In early times

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 6, June 2013

SaaS started as Application Service Providers (ASPs). In SaaS model, the application is access via the internet which is hosted at the location of the service provider. The basic idea behind is that you just need to use the software as an end user, there is no need to install the software on your local system, doing patching and upgrades and paying licensing fees for the software. You just need to pay for the per-use basis. Customer Resource Management (CRM) is the application which is mostly used in the Software as a Service model.

The cloud services are ubiquitous as a single point of access with four types of deployment models. These deployment models are:-

1) *Public Cloud*
2) *Private Cloud*
3) *Community Cloud*
4) *Hybrid Cloud*

1) Public cloud:

Public cloud services are characterized as being available to clients from a third party service provider via the Internet. The term "public" does not always mean free, even though it can be free or fairly inexpensive to use. A public cloud does not mean that a user's data is publically visible, public cloud vendors typically provide an access control mechanism for their users. Public clouds provide an elastic, cost effective means to deploy solutions.

2) Private cloud:

A private cloud offers many of the benefits of a public cloud computing environment, such as being elastic and service based. The difference between a private cloud and a public cloud is that in a private cloud-based service, data and processes are managed within the organization without the restrictions of network bandwidth, security exposures and legal requirements that using public cloud services might entail. In addition, private cloud services offer the provider and the user greater control of the cloud infrastructure, improving security and resiliency because user access and the networks used are restricted and designated.

3) Community Cloud:

A community cloud is controlled and used by a group of organizations that have shared interests, such as specific security requirement, policy or a common mission. The members of the community share access to the data and applications in the cloud. It may be managed by the organizations or a third party and may exist on premise or off premise.

4) Hybrid Cloud:

The Cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds). In this model users typically outsource non business critical information and processing to the public cloud, while keeping business-critical services and data in their control.

II. LAYERED ARCHITECTURAL REQUIREMENT

Many Cloud Computing systems are available from industry as well as academia. Despite significant advances, there are several open issues with cloud such as security, availability, scalability, interoperability, service level agreement, data migration, data governance, trusty pyramid, user centric privacy, transparency, political and legal issues, business service management etc. These issues can be addressed by proper design of Cloud Computing architecture. Only a standard and scalable architecture will be pivotal to the success of Cloud Computing, as it will serve as a means of coordination and communication between the service providers and end users. A three-layered classification of architectural requirements is classified [3] according to the requirements of cloud providers, the enterprises that use the cloud, and end-users. The architectural requirement of cloud systems is illustrated in Fig 2.

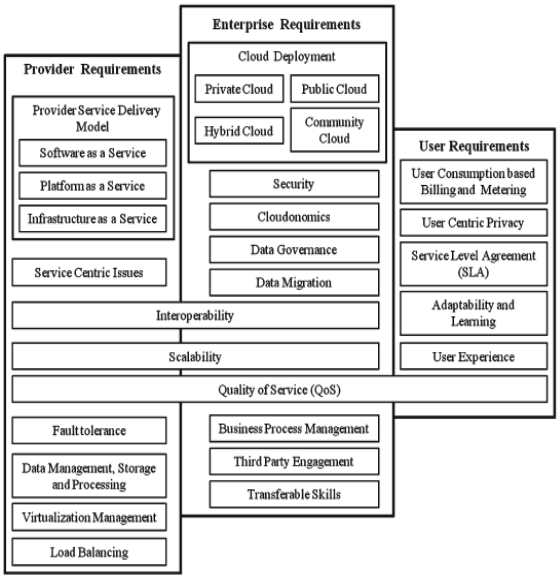

Fig 2. Three layered architecture requirement model

The layers of architectural requirements model are [3]:-

A. Provider Requirements
B. Enterprise Requirements
C. User Requirements

A. Provider Requirements

In the first phase of the requirement model firstly together the requirements and try to reduce the load, utilize price discrimination based on several factors like bandwidth, usage pattern and service level.

B. Enterprise Requirements

In this phase an enterprise should always make sure that they know what services they are paying for, and should pay careful attention to the issues like service

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 6, June 2013

levels, privacy matters, compliances, data ownership, and data mobility.

C. User Requirements

Users' requirements are the third key factor to the adoption of any cloud system within an enterprise. Cloud should be trustworthy enough to migrate critical user data. Users need assurance that their sensitive data and information are protected from compromise and loss that their data is available when required from anywhere of the world. For users, the trust issues are a major concern to the adoption of the cloud services. Stability and security can play a vital role to increase the trust between user and service providers. Cloud-based applications should be architected to be able to support personalization, localization and internationalization to make user-friendly environment. This phase has the main concern on user consumption-based billing and metering requirements, user centric privacy requirements, service level agreements, adaptability and learning requirements, and user experience requirements. SLA is the important part of the user requirement phase.

III. ESSENTIAL CHARACTERISTICS

The NIST definition describes five essential characteristics of cloud computing [1].

A. Rapid Elasticity

Elasticity is defined as the ability to scale resources both up and down as needed. To the consumer, the cloud appears to be infinite, and the consumer can purchase as much or as little computing power as they need. This is one of the essential characteristics of cloud computing in the NIST definition.

B. Measured Service

In a measured service, aspects of the cloud service are controlled and monitored by the cloud provider. This is crucial for billing, access control, resource optimization, capacity planning and other tasks.

C. On-Demand Self-Service

The on-demand and self-service aspects of cloud computing mean that a consumer can use cloud services as needed without any human interaction with the cloud provider.

D. Ubiquitous Network Access

Ubiquitous network access means that the cloud provider's capabilities are available over the network and can be accessed through standard mechanisms by both thick and thin clients.

E. Resource Pooling

Resource pooling allows a cloud provider to serve its consumers via a multi-tenant model. Physical and virtual resources are assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter).

IV. TERMS USED IN CLOUD COMPUTING CONCEPT

This section describes the terms which are used in cloud computing [19]:-

A. Interoperability

Interoperability is concerned with the ability of systems to communicate. It requires that the communicated information is understood by the receiving system. In the world of cloud computing, this means the ability to write code that works with more than one cloud provider simultaneously, regardless of the differences between the providers.

B. Portability

Portability is the ability to run components or systems written for one environment in another environment. In the world of cloud computing, this includes software and hardware environments (both physical and virtual).

C. Integration

Integration is the process of combining components or systems into an overall system. Integration among cloud-based components and systems can be complicated by issues such as multi-tenancy, federation and government regulations.

D. Service Level Agreement (SLA)

An SLA is contract between a provider and a consumer that specifies consumer requirements and the provider's commitment to them. Typically an SLA includes items such as uptime, privacy, security a backup procedures.

E. Federation

Federation is the act of combining data or identities across multiple systems. Federation can be done by a cloud provider or by a cloud broker.

F. Broker

A broker has no cloud resources of its own, but matches consumers and providers based on the SLA required by the consumer. The consumer has no knowledge that the broker does not control the resources.

G. Multi-Tenancy

Multi-tenancy is the property of multiple systems, applications or data from different enterprises hosted on the same physical hardware. Multitenancy is common to most cloud-based systems.

H. Cloud bursting

Cloud bursting is a technique used by hybrid clouds to provide additional resources to private clouds on an as-needed basis. If the private cloud has the processing power to handle its workloads, the hybrid cloud is not used. When workloads exceed the private cloud's capacity, the hybrid cloud automatically allocates additional resources to the private cloud.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 6, June 2013

I. Policy

A policy is a general term for an operating procedure. For example, a security policy might specify that all requests to a particular cloud service must be encrypted.

J. Governance

Governance refers to the controls and processes that make sure policies are enforced.

K. Virtual Machine (VM)

A file (typically called an image) that, when executed, looks to the user like an actual machine. Infrastructure as a Service is often provided as a VM image that can be started or stopped as needed. Change made to the VM while it is running can be stored to disk to make them persistent.

L. Application Programming Interface (API)

An application programming interface is a contract that tells a developer how to write code to interact with some kind of system. The API describes the syntax of the operations supported by the system. For each operation, the API specifies the information that should be sent to the system, the information that the system will send back, and any error conditions that might occur.

V. VIRTUALIZATION

With the increasing prevalence of large scale cloud computing environment, we have to draw more attention about how to provide software as a service through the internet. For this purpose a technique is used called virtualization. A virtual machine behaves exactly like a physical computer and contains its own virtual CPU, RAM hard disk and network interface card (NIC). It is the ability to run multiple operating systems on a single physical system and share the underlying hardware resources. Virtualization can also be defined as the technique of isolating the computer hardware resources into various execution environments. This can be achieved by various techniques such as partitioning the hardware and software, time sharing, partial or complete system simulation, emulation and various others [8].
Types of virtualization can be classified as:-

A. *Server Virtualization (ServV)*

In server virtualization, a single physical server is running multiple virtual machines on top of it. Every virtual machine is independent of the other virtual machine. Each virtual machine can run different operating system and application. This is due to the fact that each virtual machine is decoupled from the underlying host by a thin layer of software called Hypervisor. The virtual machine also known as Guest can be moved from one physical server host to another while running, it is called live migration.

B. Storage Virtualization (StoreV)

In storage virtualization, different physical storage devices are integrated which appears to be single storage device or pool. It can be of any type such as Network Attached Storage (NAS), Direct Attached Storage (DAS) and Storage Area Networks (SANs). These can be connected through various protocols such as Network File System (NFS), Internet Small Computer System Interface (iSCSI).

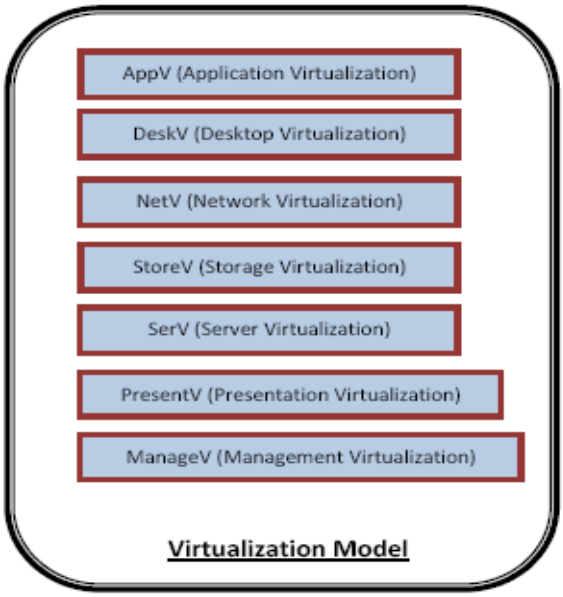

Fig 3. The seven aspects of virtualization

C. Network Virtualization (NetV)

The network virtualization allows us to take control of the available bandwidth by dividing it into various channels which further can be given to some specific resources for particular usage. The simplest example of network virtualization is virtual local area network (VLAN), which is a broadcast domain created by the switches.

D. Management Virtualization (ManageV)

In this technique, the physical and virtual datacenter are managed to give one single unified infrastructure for the provision of services. Resource Pools are the best example of Management virtualization.

E. Desktop Virtualization (DeskV)

Desktop virtualization is the technique in which the virtual machines are used for the provision desktop machines.

F. Presentation Virtualization (PresentV)

The presentation virtualization is known as Server based computing.

G. Application Virtualization

In the application virtualization, the application is not installed on the operating system. The principle is the same as that of software based Server Virtualization except the fact that it does not provide an engine to run an entire operating system.

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 6, June 2013

VI. SINGLE-TENANCY

Single-tenancy is an architecture in which a single instance of a software application and supporting infrastructure serves one customer. In the software-as-a-service (Saas) delivery model, a customer is called a tenant. In single-tenancy architecture, the tenant purchases their own copy of the software and the software can be customized to meet the specific and needs of that customer. Single-tenancy can be contrasted with multi-tenancy, an architecture in which a single instance of a software application serves multiple customers [5].

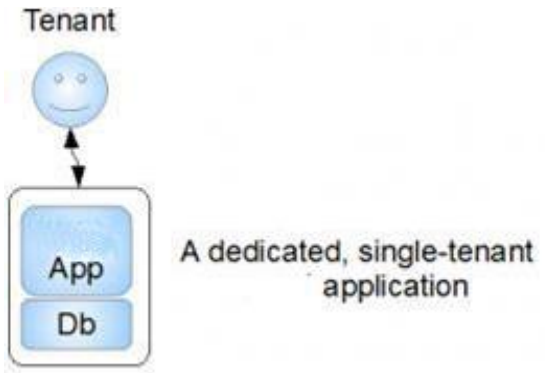

Fig 4. A Single-tenant Application

Single-tenancy is a situation where a single user or set of related users makes use of a dedicated resource.

A. Benefits of single-tenant applications

- You can customize the app to meet your specific needs, often without limitation.
- You can feel secure that no one else can access the data that the application maintains because the application executes on your dedicated machine(s) behind a firewall.
- You can control everything related to the application, including the maintenance schedule, backups, etc.

B. Drawbacks to the use of single-tenant applications:

- A traditional single-tenant application requires a dedicated set of resources to fulfill the needs of just one organization.
- You must learn how to install and configure the application for yourself. With PC software, this is usually not much of a concern. However, with enterprise software such as Oracle, it is expensive to hire and train a staff of administrators to manage operating systems, databases, application servers, and web servers.
- Once installed, you have to maintain the application yourself. Software maintenance includes tasks such as the installation of software updates and patches. Again, this can be a complex, time-consuming, and expensive proposition for enterprise software.
- You are responsible for backing up your data, and recovering it when problems arise. Again, think money for the enterprise.
- The unused CPU cycles, dedicated storage space, and power necessary to keep the application up-and-running go wasted, money down the drain that you'll never recover.

To decrease the cost of delivering the same application to many different sets of users a multitenant applications are used.

VII. MULTITENANCY

Multi-tenancy is an architectural pattern in which single instance of the software is run on the service provider's infrastructure, and multiple tenants access the same instance [14].

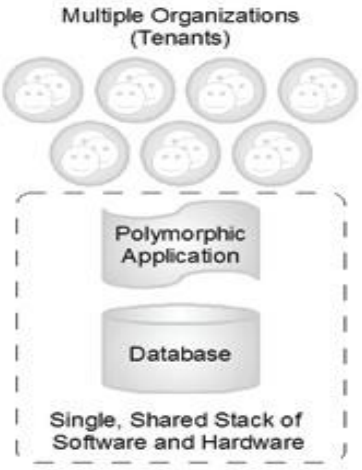

Fig 5. A multitenant application cost-efficiently shares a single stack of resources to satisfy the needs of multiple organizations.

A. Benefits of multitenant applications [2]

- Offer cheaper services as compared to other applications
- Higher utilization of hardware resources
- Ease to maintain.
- Give data aggregation opportunities.

B. Multitenancy can be achieved by three methods [2]

- Shared application, separate database.
- Shared application, separate database, separate table.
- Shared application, shared table

Achieving third type of multitenancy called pure multitenancy.

VIII. COMPONENTS OF MULTI-TENANCY ENABLEMENT LAYER

The five main service components of the multitenancy enablement layer are as following:

A. Security Isolation

In native multi-tenant system, besides those traditional security mechanisms (i.e., authentication, authorization,

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 6, June 2013

audit etc.), one also needs to consider additional potential security risk introduced by other tenants who share the same application instance and resources.

B. Performance Isolation

The tenants as well as the system administrators are naturally concerned about how to prevent one tenant's behaviors from affecting the performance of other tenants. In addition, the tenants have the rights to receive the service levels that they have paid for.

C. Availability Isolation

The tenants as well as the system administrators are naturally concerned about how to prevent one tenant's behaviors from affecting the performance of other tenants. In addition, the tenants have the rights to receive the service levels that they have paid for.

D. Administration Isolation

The administrative consoles should be isolated for individual tenants so that they can view and change the operational and business level data that are relevant to their own organizations.

E. On-the-Fly Customization

It is necessary to customize the application to fit the needs and particular situations of the tenant. For SMB tenants served under this framework, such customizations will be done in an intuitive and self-service mode. In addition, to minimize the system downtime, the customization is usually performed while the application instance is in operation.

IX. CONCLUSION

Multi-tenant software applications serve different organizations from a single instance and help to save development, maintenance, and administration costs. Existing attempts at a structured documentation of the underlying concepts are either technology-specific or restricted to certain details. In cloud-based architectures, multi-tenancy means that customers, organizations, and consumers are sharing infrastructure and databases in order to gain price and performance advantages. These services offer a pay-as-you-go lease style investment with little or no upfront costs versus buying all of the hardware and software outright. Other benefits include the ability to scale easily and tier more services and functionality on an "as needed" basis. The benefits, in fact, are so compelling that cloud computing is predicted by some to be the replacement for traditional means of obtaining these services and business capabilities by 2014. The CEO's of leading SaaS companies agree, it is not possible to grow a SaaS business without multitenancy.

REFERENCES

[1] "NIST Cloud Computing Definition", NIST SP 800- 145]

[2] Chang Jie Guo, Wei Sun, Ying Huang, Zhi Hu Wang, Bo Gao , "A Framework for Native Multi-Tenancy Application Development and Management''2007 9th IEEE International Conference on E-Commerce Technology and The 4th IEEE International Conference on Enterprise Computing, E-Commerce and E-Services.

[3] Bhaskar Prasad Rimal,Admela Jukan, Dimitrios Katsaros, Yves Goeleven, Architectural Requirements for Cloud Computing Systems, An Enterprise Cloud Approach Grid Computing Conference, Future generation computer systems, 2011.

[4] Craig D. Weissman and Steve Bobrowski, The design of the force.com multitenant internet application development platform, In Proc. of the 35th SIGMOD int. conf. on Management of data (SIGMOD), pages 889–896. ACM, 2009.

[5] Enrique Jimenez Domingo and Minguel Lagares Lemos, CLOUDIO: A Cloud Computing-oriented Multi-Tenant Architecture for Business Information Systems In Proc. of the 23rd International Conference on Cloud Computing pages 532-533.IEEE,2010.

[6] Luis M. Vaquero and Luis Rodero-Merino, A Break in the Clouds: Towards a Cloud Definition, Computer Communication Review, pages 50-55 ACM SIGCOMM Volume 39, Number 1, January 2009.

[7] Craig D. Weissman and Steve Bobrowski, The design of the force.com multitenant internet application development platform, In Proc. of the 35th SIGMOD int. conf. on Management of data (SIGMOD), pages 889–896. ACM, 2009.

[8] Kirill Kolyshkin, "Virtualization in Linux" September 1, 2006 kir@openvz.org

[9] Ralph Mietzner, Andreas Metzger, Frank Leymann, and Klaus Pohl, Variability modeling to support customization and deployment of multi-tenant-aware software as a service applications, In Proc. of the ICSE Workshop on Principles of Eng. Service Oriented Systems (PESOS), 18–25, 2009.

[10] Wikipedia, "Cloud Computing," In http://en. wikipedia.org/wiki/Cloud_computing.

[11] Liang Zhong, Tianyu Wo and Jianxin Li, Bo Li, A Virtualization-based SaaS Enabling Architecture for Cloud Computing in the process of Sixth International Conference on Autonomic and Autonomous Systems page 144-149 IEEE 2010.

[12] Matthias Schmidt, Niels Fallenbeck, Matthew Smith, Bernd Freisleben, Efficient Distribution of Virtual Machines for Cloud Computing in the process of 18th Euromicro Conference on Parallel, Distributed and Network-based Processing page 567-574, IEEE 2010.

[13] Tharam Dillon and Chen Wu and Elizabeth Chang, Cloud Computing: Issues and Challenges in the process of 24th IEEE.

[14] Waheed Iqbal, Matthew N. Dailey, David Carrera, Paul Janecek, Adaptive resource provisioning for read intensive multi-tier applications in the cloud, Future Generation Computer Systems, 2011.

[15] Franclin S. Foping, Ioannis M. Dokas, John Feehan, Syed Imran. ,"A New Hybrid Schema-Sharing Technique for Multitenant Applications", Cork Constraint Computation Centre, University College Cork, Ireland.

[16] Vassiliki Diamadopoulou, Christos Makris, Yannis Panagis, Techniques to support web service selection and consumption with qos characteristics, J. Netw. Comput. Appl., 31(2):108-130, 2008.

[17] Thomas Kwok and Ajay Mohindra, Resource calculations with constraints, and placement of tenants and instances for multi-tenant saas applications, In Proc. Int. Conf. on Service-Oriented Computing (ICSOC), volume 5364 of LNCS, pages 633–648. Springer, 2008.

[18] Zhi Hu Wang, Chang Jie Guo, Bo Gao, Wei Sun, Zhen Zhang, and Wen Hao An, A study and performance evaluation of the multi-tenant data tier design patterns for service oriented computing, In Proc. of the Int. Conf. on e-Business Engineering (ICEBE), 94–101, 2008.

[19] Dustin Amrhein, Patrick Anderson, Andrew de Andrade, Joe Armstrong, Ezhil Arasan B, Richard Bruklis, et n al. "Cloud Computing Use Cases" A white paper produced by the Cloud Computing Use Case Discussion Group, Version 2.0 30 October 2009.

[20] Lamia Youse, Maria Butrico, and Dilma Da Silva, Toward a unified ontology of cloud computing, In Grid Computing Environments Workshop (GCE08), Nov 2009.

[21] Xin Hui Li, Tiancheng Liu, Ying Li, and Ying Chen, SPIN: Service performance isolation infrastructure in multi-tenancy environment, In Proc. Int. Conf. on Service-Oriented Computing (ICSOC), volume 5364 of LNCS, 649–663, 2008.

[22] M.T. Hoogvliet, Saas interface design, Technical report, Rotterdam University 2008.

[23] Guoling Liu, Research on Independent SaaS Platform IEEE 2010.