“Service-Oriented Architecture For Enterprise Application Based Cloud Computing”

Mr. Sachin Darekar  
Computer Department  
MGM’s college of engineering and Technology, Kamothe  
Mumbai University  
sachindarekar1@gmail.com

Prof. D.R.Ingle  
Computer Department  
Bharati Vidyapeeth’s College of Engineering, CBD Belapur  
Mumbai University  
dringleus@yahoo.com

Abstract— Clouds have emerged as a computing infrastructure that enables rapid delivery of computing resources as a utility in a dynamically scalable, virtualized manner. Cloud computing offers its service to the end users on a rental basis which reduces the computing cost of an enterprise or business. Cloud computing is new paradigm for provision of a computing infrastructure and services the over the network using a pool of abstracted, virtualized, and scalable, computing resources. One of the challenges is the lack of standard in configuration, management, and programming. Our project aims at integrating service oriented architecture with cloud environment so that we can provide best optimal SOA services with the cloud. The project proposes architecture for a Service Oriented Cloud Computing System built mostly from the standard components available from Microsoft Windows Cluster and Visual Studio programming environment. Thus, we propose that a service oriented cloud can be built and program using Microsoft Windows Server and program using Microsoft CCR/DSS.

Keywords—Cloud computing, Service oriented Architecture, Service Oriented Cloud Computing System, and Cloud Service.

1.0 Introduction

Today, many organizations strive to cope with rapid market changes, such as evolving customer requirements and new business processes. One of the latest challenges is how to work with service-oriented computing (SOC) in a cloud computing environment. In the recent years, as technology advances and more and more people have their own personal computers, cloud computing has become more popular than ever. Products like Windows 8 using cloud computing and many companies like Amazon and Google making use of cloud computing. The concept of a uniform architecture for all cloud providers has risen up. Concepts such as service oriented cloud computing have sprung up and have set goals to achieve a uniform architecture that all cloud providers can use so that all people can interact with all cloud providers in a uniform manner. There are many proposed architectures that put forward the ideas on how to make a cloud architecture that will do all this. This paper will take a look at service oriented cloud computing, the proposed system aspires to improve upon current cloud architectures and make a unified architecture that all clouds should use. This architectures use service oriented approach. Service-oriented computing and cloud computing have a reciprocal relationship one provides the computing of services, and the other provides the services of computing. Although service oriented computing in cloud computing environments presents a new set of research challenges the cloud computing is an emerging new paradigm that can provide almost unlimited computing power and storage to user through the network. The goal is to substantially reduce the costs associated with the management of hardware and software resources throughout the organization. There are many cloud computing initiatives from it giants such as google, amazon, microsoft, ibm .example: gmail. There are three categories of cloud computing. Applications delivered as service on top saas layer consumed directly by the user. saas provides fully functional software over the internet to user instead of letting him install software on his computer. The second model of cloud computing is platform as a service (paas). Paas is when a provider provides a software platform that the users can use to make their own applications. Examples of paas are online development tools, such as windows azure, google app engine, and force.com. All of these products are tools that are used to develop programs online and using their infrastructure and are intended to make developing applications easier. The third main cloud computing model is infrastructure as a service (IaaS). This is when a provider provides computing power and storage for their users. It is used for letting customers install their own operating systems on their portion of the cloud that they have purchased. An example of Paas is Amazons Elastic
Cloud computing cloud (EC2). Amazon rents users disk space, CPU power and memory to run operating systems and applications on the space users have rented.

Applications lack SOA principles such as reusability, multi-tenancy, flexibility to customize, which can have significant impact on the efficiency of a cloud. We propose a system which will have some principles of SOA implemented in a cloud which will have significant advantage over traditional application architecture

2.0. Literature Review

Most of the current clouds are built on top of modern data centers. It incorporates Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), and provides these services like utilities, so the end users are billed by how much they used. View of Cloud Computing[1]

Data Centers: This is the foundation of cloud computing which provides the hardware the clouds run on. Data centers are usually built in less populated areas with cheaper energy rate and lower probability of natural disasters. Modern data centers usually consist of thousands of inter-connected servers.

Infrastructure as a Service: Built on top of data centers layer, IaaS layer virtualizes computing power, storage and network connectivity of the data centers, and offers it as provisioned services to consumers. Users can scale up and down these computing resources on demand dynamically. Typically, multiple tenants coexist on the same infrastructure resources [2]. Examples of this layer include Amazon EC2, Microsoft Azure Platform.

Platform as a Service: PaaS, often referred as cloudware, provides a development platform with a set of services to assist application design, development, testing, deployment, monitoring, hosting on the cloud. It usually requires no software download or installation, and supports geographically distributed teams to work on projects collaboratively. Google App Engine, Microsoft Azure, Amazon Map Reduce/Simple Storage Service are among examples of this layer.

Software as a Service: In SaaS[3], Software is presented to the end users as services on demand, usually in a browser. It saves the users from the troubles of software deployment and maintenance. The software is often shared by multiple tenants, automatically updated from the clouds, and no additional license needs to be purchased. Features can be requested on demand, and are rolled out more frequently. Because of its service characteristics, SaaS can often be easily integrated with other mashup applications. An example of SaaS is Google Maps, and its mashups across from the internet. Other examples include Salesforce.com and Zoho[4] productivity and collaboration suite. The dividing lines for the four layers are not distinctive. Components and features of one layer can also be considered to be in another layer. For example, data storage service can be considered to be either in as IaaS or PaaS. Hierarchical relationship among the different layers; however, it does not mean the upper layer has to be built on top its immediate lower layer. For example, a SaaS application can be built directly over IaaS, instead of PaaS. In the cloud computing environment, everything can be implemented and treated as a service. Both academia and industry have been active on cloud computing research, and several cloud computing architectures have been proposed. In [5], IBM considers current single providers cloud as limited resource, and the lack of interoperability among cloud providers prevents deployment across different clouds. A cloud computing architecture named Reservoir was proposed to create a federation from multiple cloud providers which acts as a global fabric of resources that can guarantee the required SLA. In Reservoir architecture [6], the computational resource within a site is partitioned by a virtualization layer into virtual execution environments (VEEs). A service application is decomposed into a set of software components/services running on VEEs on the same or different VEEs within a site or across from different sites. However, Reservoir architecture does not allow a component/service to run on its duplicates on different VEEs; Moreover, computing resources are abstracted as hosting service which might not be necessarily true for all clouds. In [7], a software platform for .NET based cloud computing named Aneka was introduced. Aneka[8] is a customizable and extensible service oriented runtime environment that enables developers to build .NET applications with the supports of APIs and multiple programming models. Aneka is a service-oriented, pure PaaS cloud solution. In [9], Rajkumar and his colleagues explained a market-oriented cloud architecture in detail used by Aneka, which regulates the supply and demand of cloud resources to achieve market equilibrium, adds economic incentives for both cloud consumers and providers, and promotes QoS-based resource allocation mechanisms that differentiates service request based on their utility. The key component of this architecture is SLA (Service Level Agreement) Resource Allocator which is consisted of Service Request Examiner and Access Control, VM (Virtual Machines) monitor, Service Request Monitor, and Request Dispatcher. Based on the feedback from VM and Service Request monitors, the dispatcher routes the requests from users/brokers to the cloud resources that can fulfill their QoS requirements. In [10], Huang and her colleagues from IBM described a service oriented cloud computing platform that enables web-delivery of application-based services with a set of common business and operational services. The platform supports multi-tenancy feature by utilizing single application instance model. The isolation among tenants is taken care by the underline design. Other services include subscription
management, federated ID management, application firewall, etc.

2.1 Issues with Current Clouds

Current cloud computing has following characteristics:

**A. Users are often tied with one cloud provider:**

Even though up-front cost for a cloud computing deployment is reduced and long term lease is eliminated, much effort and money is spent on developing the application for a specific cloud platform which makes it difficult to migrate the same application onto a different cloud. Often, migration simply may mean redevelopment. For example, applications deployed on Amazon EC2 cannot be migrated easily due its particular storage framework [11].

**B. Computing components are tightly coupled:**

This can be clearly explained using an analogy. Suppose one wants a new computer, this person has the choices of either buying a ready-to-use computer from a manufacturer (buying) or purchasing the components separately and building the computer in a DIY style (building). The advantages of building over buying include wider selection of components, flexibility to customize, and cheaper cost. However, the computing resources over the internet, current cloud implementations do not allow this kind of flexibility. If a customer opts to use Amazon S3 storage service, he is then stuck with other cloud computing services Amazon provides, such as EC2, Elastic Map Reduce.

**C. Lack of SLA supports:**

Currently, SLA is an obstacle that prevents wide adoption for cloud computing. Cloud computing infrastructure services such as EC2 are not yet able to sign the SLA needed by companies that want to use cloud computing for serious business deployment [12]. Moreover, business is dynamic. Static SLA is not able to adapt to the changes in business needs as cloud computing promises to.

**D. Lack of Multi-tenancy supports:**

Multi-tenancy can support multiple client tenants simultaneously to achieve the goal of cost effectiveness. Currently, one has three types of multi-tenancy enablement approaches: virtualization, mediation and sharing. To achieve the full potential of multi-tenancy, three issues remain to be solved [13]:

1. **Resource sharing:** To reduce the hardware, software and management cost of each tenant.
2. **Security isolation:** To prevent the potential invalid access, conflict and interference among tenants.
3. **Customization:** To support tenant-specific UI, access control, process, data, etc.

**E. Lack of Flexibility for User Interface:**

UI is an important part of the application, and user experience can be a major evaluation factor for a business application. However, cloud/SaaS users are limited with UI choices because UI composition frameworks, such as the one proposed in [14], have not been integrated with cloud computing.

The Cloud computing is now being employed to build a massive platform for many company such as Google, Microsoft, Face book, Yahoo, EBay, and many more. Underneath the cloud is the use of large scale clustering technology to link together a massive number of computing resources such as computing nodes and storages with high speed gigabit network. Currently, the multicore technology even helps deliver a very high performance computing system at a very low cost for the cloud computing system. Nevertheless, one of the main obstacles for a broad adoption of cloud computing technology is the lack of standard in system the configuration, management, and programming. Most of the well-known cloud implementation is still rely on a proprietary technology. For example, a cloud can be viewed by programmers through various API such as Amazon EC2 API, Go Grid API, Sun Cloud API, ElasticHosts API. Although many open standard efforts are now underway such as OCCI by OGF [15], the work is still in a very early stage.

Cloud computing system must find a way to solve the problem of service description and conversion, that is, to convert the user’s service demand into infrastructure needs. Cloud service users will be a huge user group, and different consumers have different levels of requirements towards QoS. Therefore, when the cloud system set up, it should consider the multiple QoS needs of different users.

Static algorithm which is wildly used currently is suitable for a work process of resources allocation without immediate change. Most existing scheduling algorithms are only suitable for simple processes or the QoS constraints that only have one single object. Facing the multiple QoS constraints that cloud computing required, how to ensure multi-level QoS, how to meet the multi-workflow, we need to find out new solutions. The most important advantage of Cloud computing are low-cost equipment and highly universal function. So the hardware can be easily extended, for the cluster data, it needs to divide the task as much as possible. The way of programming have to make a program could automatically run on different scalable processing nodes. The unusual fault conditions of the low-cost machines are far more than the proprietary hardware platform. Firstly, the design of cloud system must take full account of the machine anomalies. Secondly, it also needs to consider about the problem that the speed does not match the heterogeneous environment, and this has a great influence on the execution of parallel task. Suppose that lets take example for mutitenancy issue. Mutitenancy is another property that a service oriented architecture must have. This is what a lot of current cloud computing architectures do not have. The reason mutitenancy is important is for efficiency. Terms that need to be known are single tenancy and mutitenancy. Single tenancy is when a provider has an application running and only one user is using it at
a time. An example of a single tenancy program would be a text editor. A user has an application on their computer and only they can run that application. Multitenancy [16] is when a provider has one instance of a program running on a server and many people connect and use the application instance at the same time. An example of a multitenancy program would be Gmail or hotmail.

There are positives and negatives to both of these options. Most of the advantages to single tenancy programs are when the program is being used on a person’s personal machine or not when running on the cloud. The major downside to single tenancy programs on the cloud is that they are less efficient because for each person that uses that application there has to be a new instance and this uses a large amount of resources. The positives of multitenancy are handled for the user by the provider. This includes security of user data, backup of user data, and updates to software and hardware. The downsides are that you have to trust that provider is doing their job. There is never any guarantee that the company providing the service will not go out of business or make a mistake managing the user’s data.

Competitiveness requires that companies continually modify their IT systems by adding new features or deleting old ones in a relatively short period of time. Traditional software lifecycle models haven’t explicitly addressed this requirement for continuous integration of new capabilities. SOC aims to use services as basic blocks to construct rapid, low-cost — yet secure and reliable — applications. It reduces the need to develop new software components each time a new business process arises. A service is different from a traditional software artifact in that it’s autonomous, self described, reusable, and highly portable. Services range from doing simple arithmetic calculations to executing complicated programs in distributed environments. By using standard description languages, such as the Web Service Description Language (WSDL), a service can expose its interface to the outside world for service discovery and, either by SOAP [17] or Representational State Transfer (REST) protocols, be invoked separately or as a composition of multiple services. The advantages of this new computing paradigm are visible: companies and organizations can develop massively distributed software systems by assembling basic services dynamically. These services may come from different service providers and use markup language techniques, such as XML, to exchange program information and data. Five decades ago, in 1961, computing pioneer John McCarthy predicted that “computational may someday be organized as a public utility.”[18] Cloud computing is that realization, as the paradigm facilitates the delivery of computing-on-demand much like other public utilities, such as electricity and gas. However, cloud computing isn’t a new concept. Other computing paradigms — utility computing, grid computing, and on-demand computing — precede cloud computing by addressing the problems of organizing computational power as a publicly available and easily accessible resource. Effectively, cloud computing wraps traditional distributed computing or grid computing paradigms with a dynamically scaled business model that mitigates the risk of service over or under provisioning for service providers by offering shared computational power that can be accessed as needed. In contrast to grid computing, where traditionally a user needed to trust share some resources before he or she could be granted access to a larger pool of shared resources, a cloud computing user need only pay for the computing services. With cloud computing, new Internet services can be developed and deployed without capital acquisitions of hardware or large human integration expenses. Amazon launched its cloud offering back in 2006, known as the Elastic Computing Cloud or EC2. Other companies, such as Google’s App Engine and Microsoft’s Azure Platform, released their cloud platforms later in 2008.

2.2 Compounded Challenges
SOC and cloud computing have many open issues. We wonder if the reciprocal de:nitions of these two paradigms also suggest that the challenges of one might serve as an opportunity for the other. Figure 1 illustrates the challenges unique to SOC and cloud computing in addition to challenges and opportunities gained when you combine the paradigms.

A. Maintaining High Service Availability
Often, SOC-based systems require that their underlying services maintain high availability level. However, this requirement becomes more difficult to fulfill in cloud computing environments because services now reside on one cloud computing provider’s infrastructure. Consequently, the availability and responsiveness of these services has a dependency not only on the service provider, but also on the cloud provider. Failures because of outages, network problems, and human error are compounded. An intuitive solution is to deploy services across multiple cloud computing providers to increase availability and provide redundancy. Unfortunately, current cloud computing interfaces tend to be proprietary and aren’t explicitly designed for cross-platform interoperability. Consequently, consumers can’t easily migrate their deployed services and programs from one cloud computing provider to another as a countermeasure against failures. In the near term, this might challenge widespread adoption of cloud computing.

B. Providing End-to-End Secure Solutions
Security concerns exist for both SOC and cloud computing. In SOC, services underlying a composite service may originate from different providers across multiple organizations. It’s difficult to ensure an end-to-end security solution, as it would require all service providers to guarantee the same level of security
This is compounded when services reside in cloud computing environments. The underlying infrastructure of the service providers reside with other third-party providers, and as such, there are significant negotiations required between end users, cloud service consumers, and cloud providers to define a certain level of security. A significant aspect of SOC is message passing. In realizing a workflow of services, messages are passed between services or between services and the service container. An issue related to security is the threat to the privacy of proprietary information. Not only is there a risk that messages could be intercepted, but there's also the threat that competitors might be able to infer business operations from message traffic. Cloud environments suffer the same problems with privacy. In addition to message transfer, cloud consumers also have the concern that their stored information could be compromised or used inappropriately.

C. Managing Longer-Standing Service Workflows

SOC requires the management of loosely coupled services to maintain its working condition. A single service might be integrated into many complex applications and, as such, failure of one service can cause negative effects to numerous interdependent applications. Furthermore, each service within a workflow could reside with unique service providers. This is a challenge to service discovery because current service repositories are decentralized and not well advertised. Consequently, most operational service workflows either reside within one enterprise, or service providers are required to constantly monitor the health of their underlying service so that applications built from these services can maintain a certain level of performance. In a cloud computing environment [19], the challenge of service management and monitoring is extended. Current cloud computing providers don’t offer user-customized management and monitoring mechanisms built into their infrastructure. Hence, it’s still the service developer’s responsibility to provide programs and utilities to manage and monitor services.

2.3 Transformative Opportunities through Reciprocation

Now that we've looked at some of the challenges, what are some of the ways we could combine each paradigm’s strengths to help neutralize the other’s weaknesses? We can see at least three opportunities.

A. Service Discovery through Federated Clouds

In practice, the notion of automated service discovery and composition has been a major challenge. Techniques and technologies for openly discovering services on the Internet are limited. The UDDI[20] standard hasn’t been widely adopted or deployed. However, cloud computing might offer a significant opportunity. With next generation services being deployed in cloud environments, these services effectively become more centralized. The database management community uses the term federate to connect multiple distributed databases. Federated databases[21] facilitate the query of information across multiple distributed data stores. Similarly, federated clouds may facilitate the ability to openly discover the services residing within them. To realize this opportunity, the cloud community must incorporate service-based information similar to the type of information captured in UDDI directory services. Then researchers must investigate new approaches to cross-cloud connectivity. These technologies require specific protocols so that service information can be shared among cloud providers anonymously.

B. Rapid Service Deployment

SOC promotes the tenet that Web services can be accessed from anywhere and combined with other services to create higher-level works. However, merging and new organizations might need to relocate existing services to new IT environments. Portability, in these regards, is a significant challenge for SOC; cloud computing environments tend to mitigate these issues. In the future, standard cloud APIs will let service providers deploy their services seamlessly to multiple cloud computing providers. We believe that the demand for cross-cloud service deployment will increase as cloud environments are more widely adopted. Furthermore, cloud computing providers should add features to their cloud infrastructures to enable management and monitoring for deployed services. These management and monitoring functionalities should not only consider the status of deployed services but also take into account the status of underlying cloud infrastructures. Service-level agreements in future integrated service and cloud systems will operate with more accuracy and confidence.

2.4 Analysis of existing methods

It becomes very cumbersome and monotonous to follow the approach that was mentioned. As a result, changes can cause confusion as the project team proceeds. The real systems rarely follow the models as proposed. We would appreciate that the proposed system should be more flexible as compared to the former systems.

3.0 Proposed Methodology:

In this paper, we propose that a Service Oriented Cloud Computing System (SOCCS) can be constructed by combining CCR/DSS Software to form scalable services to a client application. A Service Oriented Cloud Computing System consists of many layers of hardware and software. The lowest layer is the cluster hardware that composed of computing resources connected together with high speed interconnection network. Each computing node has its own Windows operating system installed as a local operating system. Microsoft CCR/DSS must be installed on every node in the system. One node will act as a special control node that running a controller component called Cloud Service Management (CSM). CSM acts as a resources management system that keeps track of the availability of services on the cloud.
CSM can be developed as a CCR service. Thus, there is no need for any specialized software integration at all. To develop a cloud application, programmer must decompose the application is to a set of CCR/DSS services and cloud application that integrate and coordinate these services together. The decomposition of service is usually very straightforward in CCR system. An easy to use services development environment that comes with the system can dramatically help shorten the development time. When user application needs resources for the execution, cloud application will send the request to CSM. Then, CSM will discover and allocate a number of services (resources) to users in an on-demand basis. After CSM finally allocates services, it will send services information back to Cloud application. Finally, Cloud application can directly communicate with a group of services and coordinate the execution of services to solve the analysis problem. User can view the results using UI component for user interaction.

### 4.0. System Architecture

![Service Oriented Cloud Computing System](image)

**Fig. 1 Service Oriented Cloud Computing System**

This is the system architecture of our Proposed System which uses service oriented applications in cloud computing which is a method for providing more flexible and scalable service. In CCR, application is constructed using a set of services. Each service is implemented as a thread pool controlled by Dispatcher. When launched, the main part of the application will break the task to be executed into many small sub-tasks. Then, Arbiter will queue these computing requests with data to the work queue of a Dispatcher. The role of Dispatcher is to dispatch the work unit to the threads. After the result has been generated, output is sent to Arbiter again as an event. Then, Arbiter can collect the result and save them to storage for later processing. CCR helps manage both port and threads with optimized dispatcher that efficiently iterate over multiple threads. According to this programming model, a Master/Worker paradigm can be directly apply to structure the parallel and distributed application. Since CCR can manage remote invocation of thread across machine automatically, a single program can easily be scaled to run on multiple machines with a very minor configuration. As the software scale across one machine, the Decentralized Software Services (DSS), a layer of software on top of CCR, can be used to link multiple services component together across multiple machines. DSS provides a lightweight and representational state transfer (REST) oriented application model with a system level approach for building high performance, scalable applications. DSS particularly suited for creating coarse grain applications, and. DSS uses decentralized software services protocol (DSSP) and HTTP for interaction with services, and DSS provides a hosting environment, publish/subscribe, security, monitoring, logging, debugging, they are set of infrastructure services can use for create service.

Fig.1 illustrates the concept behind the software development paradigm used. To developed a cloud application, programmer must decompose the application is to a set of CCR/DSS services and cloud application that integrate and coordinate these services together. The decomposition of service is usually very straightforward in CCR system. An easy to use services development environment (as shown in Fig. 1) that comes with the system can dramatically help shorten the development time. When user application needs resources for the execution, cloud application will send the request to CSM. Then, CSM will discover and allocate a number of services (resources) to users in an on-demand basis. After CSM finally allocates services, it will send services information back to Cloud application. Finally, Cloud application can directly communicate with a group of services and coordinate the execution of services to solve the analysis problem. User can view the results using UI component for user interaction.

### 5.0 Conclusion and Future Scope

Service-oriented and cloud computing combined will indeed begin to challenge the way in which we think about enterprise computing. However, the potential for sharing could not only remove historical barriers
but also encourage organizations to think more collaboratively.

In this paper, concluded proposed a service-oriented cloud computing architecture systems (SOCCS) that allows an application to run on different clouds and interoperate with each other. The SOCCA is a 4-layer architecture that supports both SOA and cloud computing. SOCCA supports easy application migration from one cloud to another and service redeployment to different clouds by separating the roles of service logic provider and service hosting/cloud providers. It promotes an open platform on which open standards, ontology are embraced. More scalable and better Cloud Service management (CSM) is planned in future for better interoperability.

6.0 References
[3] Yi Wei and M. Brian Blake • University of Notre Dame “service oriented cloud computing”