International Journal of Innovative Research in Advanced Engineering (IJIRAE) ISSN: 2349-2163
Volume 1 Issue 6 (July 2014) http://ijirae.com

Using Platform-As-A-Service (Paas) for Better Resource Utilization and Better Quality Applications

Shrijit Pandey*
Department of CE & IT, VJTI, Mumbai, India

Varshapriya J. N.
Department of CE & IT, VJTI, Mumbai, India

Abstract — Popularity of cloud computing has increased many times in the last few years. One major driving force behind this rapid increase in adoption of cloud is the economic benefits that the cloud provides. The benefits imply the economies of scale that go with the pool of configurable computing resources which together constitute the cloud. Cloud frees the user from the job of setting up and maintaining the computational infrastructure and helps him to focus on developing and perfecting his application. Also the cloud provides the benefit of scaling (manual/real-time) so that the application continues to work even under heavy load. However moving onto cloud is not an easy process and requires planning. In this paper we review some techniques that have been used or proposed by research scholars and cloud experts to create customized cloud platforms. These techniques can be used to design our own cloud infrastructure to enable us to reap the benefits that cloud computing has to provide.
Keywords— Cloud, computing, platform, PaaS, mobile, SaaS

I. INTRODUCTION

National Institute of Standards and Technology (NIST) defines cloud computing in the following way – "Cloud computing is a model for enabling 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." A cloud typically refers to a large pool of computing resources. From this large pool, resources are commissioned for use by a user as per his demand. Thus a typical cloud can support hundreds or thousands of users at the same time. The users access the resources from the network. The network may be a local area network (LAN), a wide area network (WAN) or it may be the internet. From the large pool of resources, the user may demand for his share as per his/her requirements. The pool may have large number of processors; user may want to run some high intensity computation job and may ask for a dozen of processors. This is the basic concept of the cloud. The concept of cloud has been around since 1950s. Over the years it gained popularity but large scale adoption started from the year 2000. Now-a-days, cloud is mainly used for delivering web 2.0 applications to users and for mission/time critical applications such as banking or transactional applications.

There are three fundamental models of cloud computing – Infrastructure-as-a-service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-service (SaaS).

IaaS is the most basic cloud-service model. Vendors of IaaS offer computers – physical or virtual machines and other resources. A hypervisor runs the virtual machines as guests and pools of hypervisors within the cloud operational support-system can support large numbers of virtual machines and the ability to scale services up and down according to customers' varying requirements. IaaS clouds often offer additional resources such as a virtual-machine disk image library, raw (block) and file-based storage, firewalls, load balancers, IP addresses, virtual local area networks (VLANs), and software bundles. IaaS-cloud providers supply these resources on-demand from their large pools installed in data centres.

In the PaaS model, cloud providers deliver a computing platform, which typically includes operating system, programming language execution environment, database, and web server. Application developers can develop and run their software solutions on a cloud platform without the cost and complexity of buying and managing the underlying hardware and software layers. With some PaaS offers like Windows Azure, the underlying computer and storage resources scale automatically to match application demand so that the cloud user does not have to allocate resources manually. The latter has also been proposed by an architecture aiming to facilitate real-time scaling in cloud environments. In the SaaS model, cloud providers install and operate application software in the cloud and cloud users access the software from cloud clients. Cloud users do not manage the cloud infrastructure and platform where the application runs. This eliminates the need to install and run the application on the cloud user's own computers, which simplifies maintenance and support. Cloud applications are different from other applications in their scalability — which can be achieved by cloning tasks onto multiple virtual machines at run-time to meet changing work demand. Load balancers distribute the work over the set of virtual machines. This process is transparent to the cloud user, who sees only a single access point.

Of these three models, the PaaS model is thought to be the most important one. Cloud experts predict that the coming years, cloud vendors would shift from IaaS to PaaS. PaaS is especially important for users of private cloud. Many times an organization has got a lot of infrastructure available on its premises but in isolated form. These isolated parts of computing hardware require separate maintenance teams and generally their utilization is as low as 15% - 20%. Such isolated resources can be combined together to form a cloud infrastructure.

Once setup, the cloud can be maintained by a single team and will provide all the benefits associated with a cloud. However setting up the cloud isn't an easy task and needs thorough thinking about all the aspects which drive the IT facet of the organization. We need to think about how the applications currently running on dedicated hosting must be modified to run on cloud. Still greater planning is needed on how the applications would be deployed to the cloud and how the resources would be provisioned. In order to reach a conclusion on these issues, we take a look at a few of the techniques that are being used by researchers and experts for creating a cloud platform.

II. PAAS IN VIRTUAL CLOUD COMPUTING LAB

In their paper titled "A Framework for Implementing and Managing Platform as a Service in a Virtual Cloud Computing Lab", Wenhong Tian[1], Sheng Su[1] and Guoming Lu[1] propose a framework for design and implementation of PaaS in the cloud. Their main focus is on resource management.

A. Architecture

The whole system is divided into three major modules. They are user management module, resource management module and connection management module. Each of these modules can then be divided into sub-modules. These modules work together to provide cloud services to the user.

The User Management module is sub-divided into basic information management and user access management sub-modules. The Basic information management sub-module handles users' information records in the database. User login and authentication is managed by User login management sub-module. It also handles the user interface as well.

Resource allocation module is the heart of the management system. It includes resource usage, resource status and resource renewal sub-modules. Resource usage sub-module manages the immediate users and books resources for future users. Resource status management sub-module keeps track of status of all available resources. Resource renewal management sub-module is concerned with handling user's request for renewing the use of resources if possible.

Connection management module job is to handle users' access to the resources. It is also responsible for remote access management and remote connection management. These can be done in the remote servers together with management node. Control of PaaS resources is under the purview of one management node or many nodes in the Cloud.

The user interacts with the system using a web portal. User makes requests to the system via this web interface. Web server collects the information that the user wants to access. Then it forwards this information to the management system. The management system processes the request and returns the information to the web server. The web server displays the returned information to the user. User is given a list of choices for software-OS combinations. He can choose any one combination. Also user must specify whether he wants to use the resource immediately or after sometime along with the time period for which he wants to use the requested resources. All the information pertaining to the system and the user is stored in a database server. The database server stores four kinds of data – user information, platform information, platform state status and user connection information. The system makes extensive use of virtualization technology. The system offers operating system level virtualization so that resources can be shared between different operating systems simultaneously. It allows multiple operating systems to run on the same physical node simultaneously.

B. Implementation

The entire system is developed using open source softwares like Apache web server, MySQL database server, OpenSSH remote access tools. In order to create virtual environment VMWare workstation 5.5 is used. The user can select appropriate operating platforms with application software. There are two kinds of choices: immediate (now) application and reservation for future use. The user should choose amount of time for his application. Theoretically, this system can be used to provide and manage hundreds of real and virtual platforms. The authors plan to include imaging of software and hardware platforms, load balancing and complete automatic provisioning of resources in the future so that the system can be applied in large-scale and distributed environment.

III. PLATFORM FOR INTEGRATED MOBILE APPLICATIONS

Jagan Sankaranarayanan[2] et al., in their paper titled "COSMOS: A Platform for Seamless Mobile Services in the Cloud" propose a novel platform which allows different mobile applications to share data amongst themselves thereby offering a much better experience to the user. By allowing data to be shared among applications, developers can create a suite of applications which are aware of users needs and can provide services which serve the current context. The authors have named their platform as "Clouddb fOr Seamless MObile Services (COSMOS)". Figure below shows the architecture for the COSMOS platform.

The COSMOS system consists of a Database Coordinator for managing the computing infrastructure, query processing engine for dispatching and scheduling queries, SMILE middleware for enabling sharing between tenants, and a COSMOS data store for storing data.The Sharing MIddLEware for Mobile computing (SMILE) is a component of COSMOS that provides support for tenants to set up shared datasets. Consumers can negotiate SLA guarantees on the shared space. Figure below shows high level SMILE architecture. SMILE consists of multiple sub modules. Offline sharing negotiation module sets up sharing between an owner and consumer. Capacity planner determines how the data has to be organized (and constantly reorganized) in order to ensure that SLA guarantees will be met. Query scheduling module helps in the processing of queries from tenants and consumers, while ensuring that the throughput is maximized and all tenant policies with respect to consumer queries are satisfied. Query translation module ensures that queries made

by consumers are correctly translated to access the correct databases, relations and attributes. Access control can be implemented on the queries so that consumers can only query the data they have access.

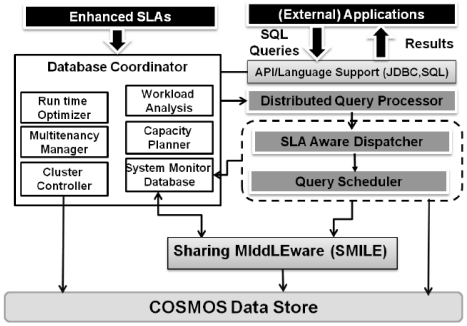

Figure 1: COSMOS Architecture

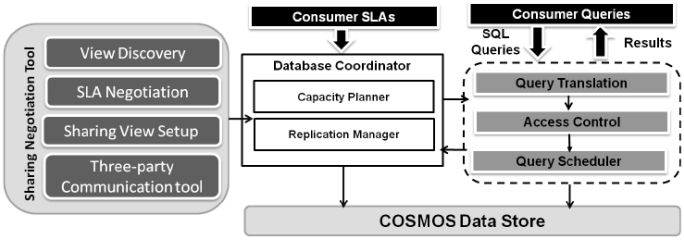

Figure 2: SMILE Architecture

The authors describe a scenario which shows how integrated mobile applications can make the user more productive. Suppose a user receives an email confirmation of acceptance of a paper to a conference. The email app pulls out the event information and populates the calendar app with the relevant information. Later when the mobile user invokes an airline booking app to purchase tickets, the app has access to the conference details, using it to determine if the user will be travelling, which airport the user is flying to/from etc. The mobile user's interaction with the airline ticket booking app is the first instance of a seamless mobile experience, which happened because the calendar app shared information with the airline booking app to make the airline reservation process intuitive for the user. Next if the user invokes a hotel reservation app, it has access to the conference details, dates of travel, air ticket, and other information from the calendar and airline reservation apps, and tries to present the user with hotel options that are proximate to the conference venue as well as being on the same dates as the conference. Subsequently, when the user invokes an app to reserve a taxi ride to the airport, the app has all the required information such as the place to pick up and drop as well as relevant parts of the user's itinerary. When the user is at the airport, and looks for restaurants to dine, the app shows options from the same terminal where the user is currently located, but not showing those that the user did not prefer in the past.

IV. OPEN-SOURCE PAAS WITH AUTOMATED RESOURCE MANAGEMENT

Calin Sandru[3] et al., in their paper titled "Building an Open-source Platform-as-a-Service with Intelligent Management of Multiple Cloud Resources" describe the architecture and implementation of a PaaS with intelligent resource management using application descriptors. In this architecture the application developer makes use of a tool to generate his application's descriptor file before deploying it on the platform. The platform uses this descriptor to provision and manage the resources set up for the application. The figure below shows the different components of the platform.

The developer of an application can currently describe the application in Java, Python, Erlang or Node.js so that the application is not depending on a certain implementation of a Cloud service. An application can profit from the elasticity at the level of components (special feature in mOSAIC, instead at a larger granularity level, at virtual machine level, as usual), if the component is able to scale (independent from other components, using message passing instead synchronous communications etc). An event-driven programming style has been adopted to reduce the network traffic. The connection with the Cloud resources is done to the level of the API through the so-called Cloudlets and Connectors, first ones expressing the reaction of the application to the events related to Cloud resource consumption, second ones being generic in terms of operations allowed for a certain type of Cloud resource (e.g. key-value store, distributed file system, http gateway, or message passing system).

The Application Tools are assisting the developers in the deployment process by offering tools for the application descriptor (stating e.g. the basic requirements in terms of the relationships between the application components), control interfaces for the deployed applications (e.g. web interface allowing to manually stop, start, replace components without stopping the entire application). The Portable Testbed Cluster allows developing and debugging of application on a desktop or on local premises. The Software Platform includes the core modules of the PaaS that are ensuring the

packaging of the application and deployment in virtual machines acquired in Private or Public Clouds, the control of the deployed components, the registration and discovery of new components, or the credential management

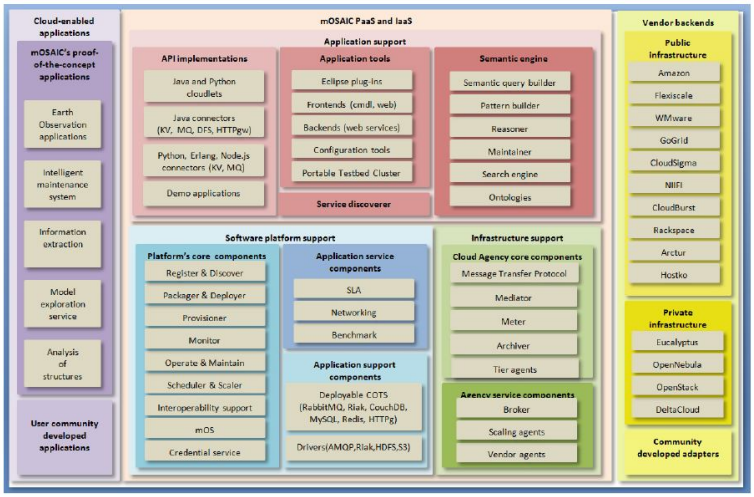

Figure 3: mOSAIC Platform Architecture

. The interoperability service is acting as a proxy between a vendor agnostic and language-dependent Connector used by a certain application and a Driver of a certain type of Cloud resource (e.g. for message passing protocol AMQP, for Riak key value store, for Hadoop distributed file system, or for Amazon S3). mOS is the key module that is ensuring the platform functionality on virtual machines from different Cloud providers (with Xen, KVM or LXC support, tested until now on CentOS, Ubuntu, Suse, Redhat). Deployable open-source technologies (like RabbitMQ as message queuing system, Riak as key-value store, Jetty as web server) are used also as Cloud resources available on the provisioned virtual machines running the mOSAIC platform. The application developer can use any of these cloud resources by including it in its deployment descriptor. Suppose an application needs the CA support, thus a CA Virtual Machine has to be available; the mOSAIC Platform support, thus the Platform Control VM and Platform Execution VMs should be available; a storage for a mOSAIC Platform Driver implementing a key-value storage; the Web Server functionality presented as a distinct tier with all the resources at this tier being load balanced. All these resources can be requested via the deployment descriptor. Once deployed, the platform will automatically provision the required resources as requested by the application via the deployment descriptor.

V. CONCLUSIONS

The techniques reviewed here describe how we can implement a PaaS for various different scenarios. These techniques can be used as a starting base for developing our own PaaS. Though quite thorough, the techniques have got some weak points which need to be taken into consideration. The first technique (virtual cloud computing lab) is more suitable for research purposes and may not be suitable for use by a business organization; especially if the business wants interoperability with commercial cloud service providers. That said this technique very well serves its purpose for an educational institution. The second technique is more focused towards data sharing. It appears to be more of a platform for sharing data rather than efficient resource utilization. However the technique is quite good in the sense that it allows seamless data exchange between applications resulting in richer and enhanced user experience. The third technique is most suitable for business needs. It allows automatic resource provisioning and management. However, it seems quite complex to implement. This complexity stems from the fact that the platform tries to decouple the deployed application from the underlying architecture. This is necessary to make the application development independent from the architecture of the cloud. Still, once implemented, the platform would be able to serve its purpose of automated resource management.

REFERENCES

[1] Wenhong Tian, Sheng Su, Guoming Lu, "*A Framework for Implementing and Managing Platform as a Service in a Virtual Cloud Computing Lab*", 2010 Second International Workshop on Education Technology and Computer Science.

[2] Jagan Sankaranarayanan, Hakan Hacigumus, Junichi Tatemura, "*COSMOS: A Platform for Seamless Mobile Services in the Cloud*", 2011 12th IEEE International Conference on Mobile Data Management.

[3] Calin Sandru, Dana Petcu, Victor Ion Munteanu, "*Building an Open-source Platform-as-a-Service with Intelligent Management of Multiple Cloud Resources*", 2012 IEEE/ACM Fifth International Conference on Utility and Cloud Computing.

