ISSN 1512-1232

CLOUD COMPUTING : SAAS

S.Satyanarayana

HOD, Assoc. Professor CSE & Mathematics
Sri Venkateswara Institute of Science & Information Technology,
Tadepalligudem-534101,E-mail:s.satyans1@gmail.com

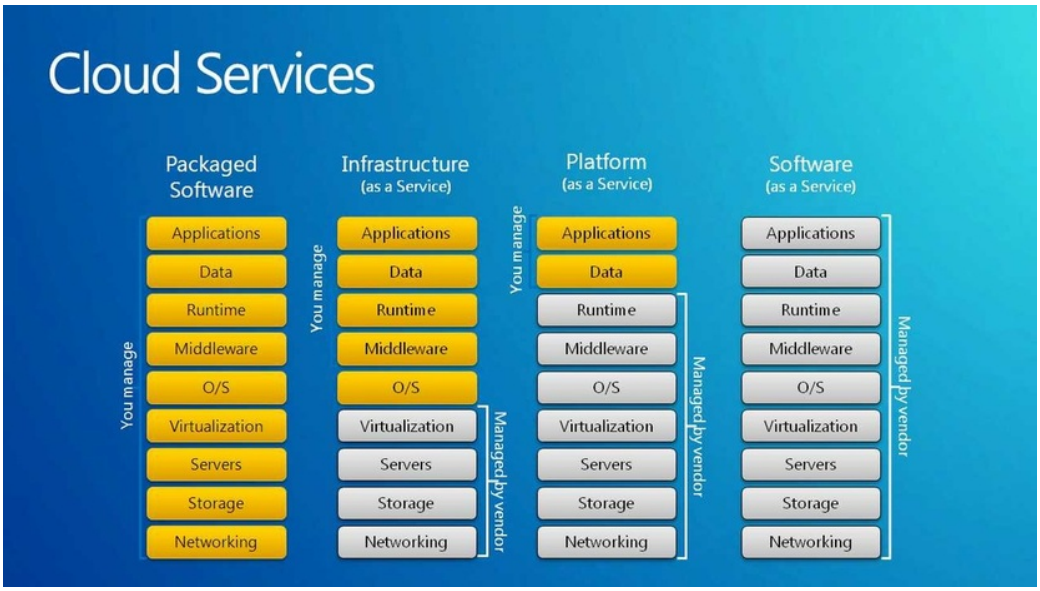
Abstract

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources[7] (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 is composed of five essential characteristics(On-demand self-service, Broad network access., Resource pooling, Rapid elasticity, Measured service), three service models(Software as a Service (SaaS). Platform as a Service (PaaS). Infrastructure as a Service (IaaS)) and four deployment models(Private cloud, Community cloud, Public cloud, Hybrid cloud) in This paper analysis current development of SaaS, Software as a Service (SaaS) is a newly emerging business model in the software industry. The growing speed of SaaS is fast. The paper gives a quick survey on SaaS., SaaS application architecture and SaaS Maturity Model.

Keywords*: Cloud Computing ,SaaS, SaaS Models, SaaS Layers, SaaS architecture maturity levels.*

I. INTRODUCTION

Cloud Computing Definition: 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 is composed of five essential characteristics(On-demand self-service, Broad network access., Resource pooling, Rapid elasticity, Measured service), three service models(Software as a Service (SaaS). Platform as a Service (PaaS). Infrastructure as a Service (IaaS)) and four deployment models(Private cloud, Community cloud, Public cloud, Hybrid cloud).



ISSN 1512-1232

2. SaaS[1,2]

Software-as-a-Service is a software distribution model in which applications are hosted by a vendor or service provider and made available to customers over a network, typically the Internet. SaaS is becoming an increasingly prevalent delivery model as underlying technologies that support web services and service-oriented architecture (SOA) mature and new developmental approaches become popular. SaaS is also often associated with a pay-as-you-go subscription licensing model. SaaS applications also must be able to interact with other data and other applications in an equally wide variety of environments and platforms. SaaS is closely related to other service delivery models we have described. SaaS is most often implemented to provide business software functionality to enterprise customers at a low cost while allowing those customers to obtain the same benefits of commercially licensed, internally operated software without the associated complexity of installation, management, support, licensing, and high initial cost.

2.1 SaaS models[3,4]

Accessed through the Web: End users access the application over the Internet using a standard web browser. The web-based approach is used instead of the traditional ,PC-based client accessing resources over a corporate WAN.

SaaS Vendor **Support:** Rather than being managed by the corporate IT department, the application is hosted and operated by the software developer themselves.

SaaS Subscription Pricing: Instead of paying upfront perpetual license fees, the customer pays a recurring monthly fee for use of the functionality.

SaaS Low Customization: Very little customization of the software is performed. Applications are highly standardized across customers, often hosted in a multitenant architecture model.

SaaS Managed Upgrades: Functionality enhancements are completely controlled by the vendor. Frequent upgrade cycles occur with new features being introduced multiple times per year.

The SaaS model represents a radical change to the software industry as it encompasses several major paradigm shifts. Pricing and upgrade models are an obvious area of change with SaaS. But the implications are much more far-reaching, impacting nearly every aspect of a software company's business from financial reporting to organizational structure. Perhaps the two biggest changes are:

SaaS Shift to service-based mentality—The SaaS model requires a paradigm shift away from a product-centric approach and toward a service model. In the new model SaaS vendors are accountable not just for developing the application, but the entire suite of services to support the software in production. Beyond just the application code, the vendor must provide the entire customer experience including implementation, testing, training, troubleshooting, maintenance, hosting, upgrades and security.

SaaS Success based revenue model—In the SaaS model a vendor's success is critically linked to the customer's success. The buyer makes no upfront investment in software, hardware or implementation resources. The vendor is only paid if the client is satisfied with the software and therefore continues their subscription. Unlike traditional software models, unsatisfied users of SaaS can easily unsubscribe and switch to a competing provider.

2.2 SaaS Layers[5,6]

SaaS pattern in the practical application development. The platform is designed according to SaaS pattern which conform high quality resources and provide users the component-lever services by multi-tiered abstraction. There are six logic levels in the platform.

SaaS User layer: Users may use two statuses to visit the technology platform: In school user and extracurricular user.

SaaS Protocol layer: Service transport The services provided by the technology platform
are transported to users by Internet. System is designed according to SOAP, adheres to XML and HTTPS protocol, describe service by WSDL, discovery and obtain service metadata by UDDI. When user and application procedure carries on the information exchange they need to follow the network security agreement (take cryptology as foundation exchange of information agreement) to guarantee the interactive security.

ISSN 1512-1232

SaaS Component layer: Service wrapped and schedule Besides releasing information, the technology platform provides the legal copy development environment and the components library for users as well as software courses self-training program for the user to choose Resources, here means practical training courses and self-training application, are encapsulated as services and you can configure them with parameters.

SaaS Function expanding layer: Service technology layer The medium operator may directly surmount this layer. For the self-training system, on the original several courses' foundation, may carry on the course system's expansion, development once more is realized through the API provided by the training system.

SaaS and PaaS platform: Applications layer Divided by layer structure, integration software engineering supporting environment and public component library on Trust IE belong to applications layer (PaaS platform) and they directly served to services wrap and schedule. While API and Web services provided by SaaS platform serve to service technology layer.

SaaS Data layer: Data and services management layer Technology platform conformity many independent system's resources, various part data are stored in independent databases. For the scheme of users use the sharing database and shared data structure, before the user data, increases the Tenant ID field to realize the data isolation.

2.3 SaaS Architectural Maturity Levels:

Many types of software components and applications frameworks may be employed in the development of SaaS applications. Using new technology found in these modern components and application frameworks can drastically reduce the time to market and cost of converting a traditional on-premises product into a SaaS solution. According to Microsoft, SaaS architectures can be classified into one of four maturity levels whose key attributes are ease of configuration, multitenant efficiency, and scalability. Each level is distinguished from the previous one by the addition of one of these three attributes. The levels described by Microsoft are as follows.

SaaS Architectural Maturity Level 1

Ad-Hoc/Custom. The first level of maturity is actually no maturity at all. Each customer has a unique, customized version of the hosted application. The application runs its own instance on the host's servers. Migrating a traditional non-networked or client-server application to this level of SaaS maturity typically requires the least development effort and reduces operating costs by consolidating server hardware and administration.

SaaS Architectural Maturity Level 2 Configurability The second level of SaaS maturity provides greater program flexibility through configuration metadata. At this level, many customers can use separate instances of the same application. This allows a vendor to meet the varying needs of each customer by using detailed configuration options. It also allows the vendor to ease the maintenance burden by being able to update a common code base.

SaaS Architectural Maturity Level 3 Multitenant Efficiency. The third maturity level adds multi tenancy to the second level. This results in a single program instance that has the capability to serve all of the vendor's customers. This approach enables more efficient use of server resources without any apparent difference to the end user, but ultimately this level is limited in its ability to scale massively.

SaaS Architectural Maturity Level 4—Scalable. At the fourth SaaS maturity level, scalability is added by using a multi tiered architecture. This architecture is capable of supporting a load-balanced farm of identical application instances running on a variable number of servers, sometimes in the hundreds or even thousands. System capacity can be dynamically increased or decreased to match load demand by adding or removing servers, with no need for further alteration of application software architecture.

2.4 SaaS offerings and tools:

Adobe Photoshop Express ,Online image processing, fluidOps, eCloud Manager, SAP Edition, SAP Landscape as a Service, Google Docs, Online office applications, Google Google Maps API, Service for the integration of maps and geo information, Google Open Social Generic programming interface for the integration of social networks into applications OpenID, Distributed

ISSN 1512-1232
cross-system user identity management system, Microsoft Windows Live Online office applications, Salesforce.com Extensible CRM system.

2.5 Successful Saas Architectures

Salesforce.com built their hosted solutions from ground up; they used completely web-based architectures as well as exploited internal cost advantages from 'multi-tenancy' and configurability. These architectural features enabled these solutions to offer sustained economic advantages over traditional on-premise software, as we examine below. Key elements of a successful, economically advantageous If one considers the costs of managing a software product in the traditional manner (i.e. installed within customer premises),a large part of the costs go into managing different versions of the product and supporting upgrades to multiple customers. As a consequence, often fewer and larger upgrades are made, introducing instability in the product together with the need for releasing and managing intermediate patches .On the customer side, there are corresponding costs as well receiving and applying upgrades, testing them, and also redoing any local customizations or integrations as required .A hosted SaaS model virtually removes these costs from the customer side, and on the provider, the 'multi-tenant' architecture brings down the costs of releasing upgrades by an order of magnitude: With multi-tenancy, the hosted SaaS application runs a single code base for all customers, while ensuring that the data seen by each customer is specific to them; i.e depending on who is logged in and the customer organization they belong to, an appropriate data partition is accessed

Conclusions

SaaS platform is not only a development platform but also a resources platform. In the scheme of SaaS, all data and software can be used as services. These services represent as API or applications provided for users.. The SaaS development and the traditional development difficulty are not very different. The key is the process of breaking the old development pattern and adapting SaaS thought. At present the SaaS development did not have the unification industrial standard There are a lot of works to do in studying SaaS architecture.

Acknowledgements:

We would like to express our thanks to referees for valuable comments that improved the paper. The Author wish to thank Principal & Management, Sri Venkateswara Institute of Science & Information Technology, Tadepalligudem for their encouragement and cooperation in the preparation of their research paper.

References

1. Greg Goth.Software-as-a-Service: The Spark That WillChange Software Engineering?. IEEE Distributed Systems, Online, 2008.
2. Manish Godse, Shrikant Mulik. An Approach for Selecting Software-as-a-Service (SaaS) , IEEE CS, 2009. 74, P.155-158.
3. Espadas Javier,Concha David,Molina Arturo. Application Development over Software-as-a-Service Platforms. IEEE, 2008. 48, P.97-104.
4. Hancheng LIAO. Design of SaaS-based Software Architecture. IEEE, 2009. 46, P277-P281
5. Yong Zhang, Shijun Liu, Xiangxu Meng. GridSaaS: A Gridenabled and SOA-based SaaS Application Platform. IEEE, 2009. 40, P.521-523.
6. Wu Chou. Web Services: Software-as-a-Service (SaaS),Communicaton and Beyond. IEEE Congress on Services. PartII, 2008.
7. Wu Jiyi, Ping Lingdi, Pan Xuezeng. Cloud Computing: Concept and Platform, Telecommunications Science, 12: 23-30, 2009.

Article received: 2011-12-28