Development of Multitenant Application as SAAS

Vishal Vitthal Bandgar
Research scholar,
B.I.G. College of engineering,
Kegaon, Solapur

Gopika Avinash Fattepurkar
Asst.Professor
V.V.P.Polytechnic, Solapur.

C.M.Jadhav
H.O.D. Computer science and engineering,
B.I.G. college of engineering, Kegaon, Solapur

Sharad M. Kawale
H.O.D. Computer science and engineering,
S.V.E.R.I C.O.E. Pandharpur.

***Abstract*— Software as a Service (Saas) is one of the striking features of cloud computing. Saas has revolutionized the software engineering very significantly. Saas eliminates the requirement of customers (tenants) to purchase, install and maintenance of infrastructure and software. Customers just have to pay for services provided by Saas vendors. Multi-tenancy in Saas application is most important feature for the success of Saas application. However, there are many challenges in the development, deployment, and security of such application. This paper addresses the issue of how to effectively help multi-tenancy in Saas application and proposes Saas architecture to backing multi-tenancy in e- commerce application.**

***Keywords*— Cloud computing, Saas, Multi-tenancy, Saas Architecture, and e-commerce.**

I. INTRODUCTION

Cloud computing has various definitions based on the mode of implementation, however the most accepted definition is given by National Institute of Standards and Technology is as per the following:

A. NIST Definition of Cloud computing:

Cloud computing is a model for enabling omnipresent, convenient, on-demand network access to a shared pool of configurable figuring resources (e.g., networks, servers, stockpiling, applications, and administrations) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models [1]. The basic terminology and methodology to compare the cloud services that best fit for organizational needs are given by Miguel Reixa and et al. in their paper [2]. The authors give basic definitions and comparisons of various cloud computing services and models.

The basic ideology behind cloud computing is that, the user will be able to subscribe the service from cloud vendors and generally charging will be done by utilizing pay-per- use method of the subscribed services. This will eliminate the hardware or software purchasing. The service architecture of cloud computing can be visualized in layers. The description of those layers is:

The Main layer Infrastructure as a service (Iaas) is present above the real infrastructure includes various services like resources enterprise infrastructure, cloud facilitating, virtualization, network infrastructure, etc. The second layer Platform as a service (Paas) uses the Iaas as foundation and provides services like platform for cloud application development, virtual macine allocation, etc., and the uppermost Software as a Service (Saas) layer is used by the majority of the cloud users, this layer provides the web based applications, business process software (e.g. CRM), etc.

In last decade or somewhere in the vicinity web has become a platform for the deployment of many software items. This is the main reason behind innovation in web based application development of software. Software as a Service is one of the best answer for new software industry requirement. Saas is defined by many cloud providers in different words. Here we can determine software as a service, a software delivery model which is accessed by its customers with the help of meager client or web browser and deployed on cloud platform. There is no need to purchase hardware or licensed software to access Saas application. Users of Saas just need to subscribe the service and pay according to the agreement between provider and customer; generally pay-per-use basis for charging the customers.

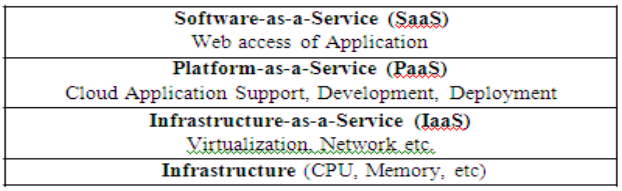

Fig. 1. Cloud Computing Service Architecture

Many companies are eager to upgrade their web based application to a Saas-based application [3]. Multi-tenancy is one of the most important characteristics of software as a service. The true multitenant Saas application is a single instance multiple tenants (customers) model of application delivery i.e. the Saas application – tenants demonstrates one-to- many relationship, where a single application (code and database) is accessed by multiple tenants at a particular time.

1.1 What Is a Tenant?

To define tenant in the context of cloud computing is not as simple as it appears. Take Amazon Web Services (AWS), for example. AWS is a cloud service provider who offers services like application facilitating, backup and storage, e-commerce, and media facilitating etc. Various organizations like Autodesk, Urban Spoon, and Second Life are tenants of AWS, i.e. they use AWS storage and compute resources to satisfy customers. Each firm withal has customers who store data like personal predilections, credit cards, and information as tenant users of these businesses. On account of Second Life, for example, if the tenants set up online businesses and accommodations of their own, they additionally will have tenants and so on. In the final analysis, a cloud service tenant is sharing a resource with a group. And similar to a building tenant, the tenant's space must be dissevered and isolated from other occupants to achieve a certain degree of security and privacy.

1.2 Defining Multi-Tenancy

The conception of multi-tenancy, or many tenants sharing resources, is fundamental to cloud computing. Service providers are able to manufacture network infrastructures and data architectures that are computationally very efficient, exceedingly scalable, and facilely incremented to accommodate the many customers that share them. Multi-tenancy spans the layers at which services are provided. In Iaas, tenants share infrastructure resources like hardware, computational servers, and data storage contrivances. With Saas, tenants are sourcing the same application (e.g., Salesforce.com), which denotes that data of multiple tenants is likely stored in the same database and may even share the same tables. When it comes to security, the perils with multi-tenancy must be addressed at all layers. The next few sections examine how this can be accomplished for shared hardware and application infrastructure.

The paper is divided in after manner: 2. Previous Work i.e. Literature review, 3. Problem analysis i.e. issues 4. Multitenant Saas Application Model, 5. Conclusion and Future Work, and 6. References.

II. LITERATURE REVIEW

A great deal of research has been done to discover ways of implementation of multi-tenancy in Saas applications. A business based on Saas application will be successful just on the off chance that it backs multi-tenancy. Multi-tenancy is an organization approach for Saas application. Key characteristics of multi- tenancy can be given as 1. Hardware and resource sharing, 2. High degree of reconfigurability, 3. Shared application and database instance [4]. Jinan Fiaidhi et al. [5] proposed general multitenant cloud architecture. Here author has written about managing data of multitenant Saas application. He specified various approaches as takes after 1. Putting away tenant data in separate databases, 2. Lodging multiple tenants in the same database, with tenant specific schema, and 3. Utilizing the same database and a same set of tables to have multiple tenants.

A document of MSDN Microsoft by Frederick Chong and et al. [6] shows architecture for multitenant data.

The multitenant can be divided into various categories depending on tenants data separation in an application database. This can be appeared,

- separate Databases; In this approach each tenant has its own particular separate database, a simple however not a true multitenant approach.

- shared Database, Separate Schemas, In this approach tenants do share the database, however a schema is different for each tenant. This approach is applicable for data separation and sharing database, this approach is also known as semi-multi- tenancy approach.

- shared Database, Shared Schema, This one is an exceedingly preferred approach and called as pure-multi-tenancy approach, in this approach each tenant data is stored in the same database and same schema, the separation of tenant is done by assigning tenant_id to columns in tables that the tenant possesses.

III. PROBLEM ANALYSIS

A multitenant Saas applications are a combination of shared application and database so we have to be very careful while designing these applications. The main issue we have to address in an implementation process of multitenant application is an architecture design. This architecture will comprise suggestions regarding the choice of the database and schema for an application [6], choose how the system identifies the tenant and how the rationale is handled by the system. This literature doesn't give a clear idea about an architecture required to implement multitenant application.

Cor-Paul Bezemer and et al. proposed a multitenant architecture conceptual blueprint [4] for implementation of multitenant Saas application. In the architecture tenant specific authentication is included to authenticate tenants, also a configuration component that can be implemented to allow customization for each tenant and an its database. The authentication process is used to authenticate the tenants and it separates the configuration component for each tenant. The configuration component includes work process, general configuration, layout style, etc. is to be used for customization by each tenant and the database includes few layers so it isolates the data in same environment for each tenant.

However, there are some disadvantages in the above architecture related to authentication and database. The authentication process uses a ticket server to recognize the tenant, which is not necessary in the event that we use a proper method to implement tenant system e.g. sub-domain can be used to identify tenant, which we will see in later section. The database uses various layers to isolate tenant-specific data which is not actually needed in the event that we use the any method given in [6] by Frederick Chong and et al. Rouven Krebs and et al. in [7] have proposed various approaches for resource sharing in cloud environment and also defines multi-tenancy in cloud environment. The papers differentiate in sharing of resources at layers of cloud computing like code base, sharing at data center, sharing utilizing virtualization at different. The white paper on Securing Multi-Tenancy and Cloud Computing[13] and Bo Tang_y and et al proposed a Multi-Tenant RBAC Model for

Collaborative Cloud Services[14] gives us idea about various security features for system.

The white paper by Jason Meiers and et al. of IBM [8] explains the requirement of multitenant Saas applications. It also specifies various requirements to be fulfilled for building multitenant network topology, multitenant databases etc. Bikram Sengupta and et al. [9] proposed a multi-tenancy re-engineering pattern, Here the possible sharing of database, reconfigurability of user interface required, work process etc. for such application is specified. We are extending our multitenant Saas application architecture from [4] and [6].craig D Weissman and et al. discussed platform for multitenant internet application [10].here the design of Multitenant application of force.com is described.

This clears our problem of multitenant Saas application for implementation. The next section proposes an answer for above mentioned problems in implementing multitenant Saas application.

IV. MULTI-TENANCY SAAS APPLICATION ARCHITECTURE APPROACH

Before going to architecture approach, you ought to first research the application that implemented utilizing this approach. Here implementation of an e-commerce application is to be discussed, which will have carts, users, items, and orders as components and metadata of each tenant. This application is to be implemented utilizing ruby on rails framework, Postgre SQL as database and deployed to Heroku [11] (Cloud Platform for rails) and then after deployment it will be accessible to users of various tenants through internet utilizing browsers.
The diagram shows architecture for the implementation of multitenant Saas application in the cloud environment. The architecture has four main parts. Each of these parts is designed to backing our multitenant Saas e-commerce application.

A. Sub-domain URL

In a general case URL is used to determine the address of simple web application. In such a cases components of a main application are identified by sub-domains. DNS server handles sub-domains for locating application server.
In this case the sub-domain is used as a parameter of the tenant which is used to discover the store of a particular tenant. This will help to identify the tenant_id of a tenant and execution of all operations in this store is totally based upon application rationale. This system is designed to help multiple tenants with multiple sub-domains for a single domain.

The application rationale will constantly run on the web server in the cloud environment. It will accept the HTTP requests from users of the system and replies with HTML file.

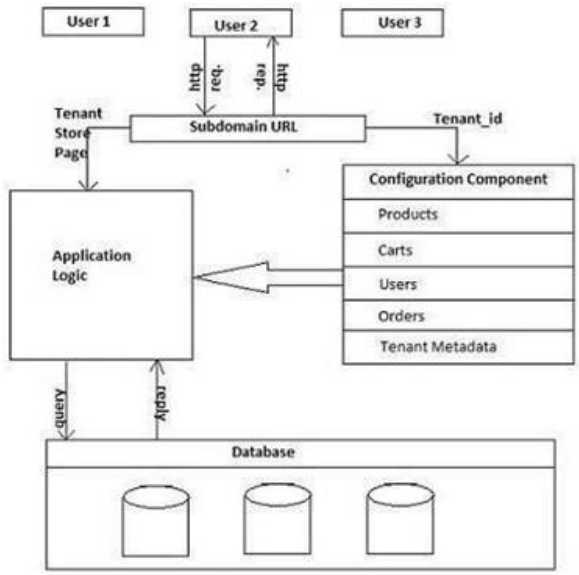

Fig. 2. Multitenant SaaS e-commerce Application Architecture

B. Configuration Component

In multitenant application configuration per tenant is the most important module. This e-commerce application will include various components like carts, users, items, orders, list of things to get etc. We are going to scope these components substratum of tenants and identification of such can be done by using tenant_id from components database. The main configurable components of our application are items and users. Each tenant user (seller) will have access to configuration components of the system. There will be isolation among components of all the tenants and each tenant will just able to see its own particular component. The tenant_id, name of the tenant and sub-domain will be present in tenant metadata.

C. Database

Database handling of each tenant has some issues cognate to storage and access to the data. However in particular to the multitenant database Jinan Fiaidhi and et al. in [5] proposed three approaches to implement multitenant database, they are shared database - shared schema (pure multi-tenancy), shared database - separate schema (semi- multi-tenancy), and finally separate databases. Also, in another paper by Stefan Aulbach and et al. multi-tenant Saas application database and schema mapping techniques is discussed [12]. In our dissertation, we will implement pure multi-tenancy approach in which tenant_id will share by all configurable components tables and they will access by identifying tenant_id from the sub-domain of the tenant. The pure multi-tenancy scheme isolates the data of each tenant hence it will surely provide privacy between each tenant.

V. EXPECTED RESULT

The expected results of multitenant application are given in the form of snapshots of store of various tenants and comparing them.

In single tenant e-commerce application has only one store to purchase or sell products. In this multitenant e-commerce application there will be 2 tenants selling their products from their respective store shown using different tenants.

The following figure shows a store of tenant1 which has sub-domain"tenant1" for domain "saas.com".

Fig. 3. View of Tenant 1 System with its sub-domain

The above diagram represents the book shop as one of the two tenants.

The following figure shows a store of tenant2 which has sub-domain "tenant2" for domain "saas.com".

Fig. 4. View of Tenant 2 System with its sub- domain

The above diagram represents the watch shop as one of the two tenants.

VI. CONCLUSION AND FUTURE WORK

The paper suggests a whole new approach for implementation of the multitenant SaaS applications; it also withal describes the procedure for the implementation of e-commerce multitenant SaaS application. In this approach identification of tenant is done with the help of sub-domain URL, providing reconfigurability in multi-tenancy and most important a data separation for each tenant in the cloud environment. This architecture can be used for implementation of any other multitenant SaaS application.

The architecture can be modified according to the desideratum of application to implemented. The dashboard can be developed, which will help tenants to choose the configurable components for his applications in multitenant application. This will make the multitenant application highly configurable.

REFERENCES

[1] Peter Mell, Timothy Grance, "The NIST Definition of Cloud Computing", NIST Special publication 800-145, September 2011.

[2] Miguel Reixa, Carlos Costa, Manuela Aparicio, "Cloud Services Evaluation Framework", OSDOC"12 June 11, 2012.

[3] Scott Chate, "Convert your web application to a multitenant SaaS solution", White paper, Developer- Works, IBM, 14 Dec 2010.

[4] Cor-Paul Bezemer, Andy Zaidman, "Multitenant SaaS Applications: Maintenance Dream or Nightmare?" The Delft University of Technology Software Engineering Research Group Technical Report Series, September 2010.

[5] Jinan Fiaidhi, Irena Bojanova, Jia Zhang and Liang-Jie Zhang, "Enforcing Multi-tenancy for Cloud Computing Environment", IT Pro January/February 2012.

[6] Frederick Chong, Gianpaolo Carraro, and Roger Wolter, **"Multitenant** Data Architecture, Microsoft **Corporation",** Available http://msdn.microsoft.com/enus/library/aa479086.aspx, as on April 2013

[7] Rouven Krebs, Christof Momm and Samuel Kounev, **"Architectural** Concerns in Multitenant SaaS **Applications."**

[8] Jason Meiers, **"Best practices for** cloud computing multi-**tenancy",** White paper, IBM, 06 Jul 2011

[9] Bikram **Sengupta, Abhik Roychoudhury, "Engineering** Multitenant Software-as-a-**Service Systems", ICSE"11,** May 21–28, 2011.

[10] Weissman, C. D. and Bobrowski, S. 2009), **"The design** of the force.com multitenant internet application **development platform", In** Proceedings of the 35th SIGMOD international conferenceon Management of data, **SIGMOD "2009**

[11] Gettig Started with Rails 3.x on Heroku Available https://devcenter.heroku.com/articles/rails3, as on April 2013.

[12] Stefan Aulbach, Torsten Grust, Dean Jacobs, Alfons Kemper, Jan **Rittinger, "Multitenant Databases for** Software as a Service: Schema-**Mapping Techniques", SIGMOD"08, June** 9–12, 2008.

[13] White Paper - Securing Multi-Tenancy and Cloud Computing 2012, Juniper Networks, Inc

[14] A Multi-Tenant RBAC Model for Collaborative Cloud Services Bo Tang_y, Qi Li_ and Ravi Sandhu. Institute for Cyber security Department of Computer Science University of Texas at San Antonio One UTSA Circle, San Antonio, Texas 78249 2013 Eleventh Annual Conference on Privacy, Security and Trust (PST).