I.J.Modern Education and Computer Science, **2012, 9, 39-44**
Published Online September 2012 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2012.09.05

An Approach Towards Customized Multi-Tenancy

Muhammad Fahad Khan
Department of Software Engineering
University of Engineering and Technology, Taxila
fahad.khan@uettaxila.edu.pk

Mirza Ahsan Ullah
Department of Software Engineering
University of Engineering and Technology, Taxila
mirza_ahsan48@yahoo.com

Aziz-ur-Rehman
Department of Software Engineering
University of Engineering and Technology, Taxila
comsian_17@yahoo.com

Abstract— Multi-tenant applications are usually cloud based software services which can serve different users at the same time. This is done using single instance of applications by sharing hardware, infrastructure, data storage and virtualization. To achieve multi tenancy different approaches are there at every layer (Application, Data, hardware).Multi tenancy helps to minimize the effort needed for deployment and maintenance with an adequate level of security and privacy. Now a day the work is going on custom multi tenancy by sharing everything at each layer. This paper will describe the techniques of multi tenancy and how security, flexibility and scalability relate to each other when it comes to multi tenancy for customization.

Index Terms **—** IaaS (Infrastructure as a Service), SaaS (Software as a Service), Horizontal customization, Vertical customization

I. Introduction

In cloud computing the architecture style using multi tenancy means that user and organizations are members of a common infrastructure and data store to gain the advantages like price and performance.

Cloud applications are internet applications having servers and every client can access that application by just having internet connectivity. Customer can store their data or retrieve any information from these servers as a tenant user of a particular application. Multi tenancy is a fundamental concept when it comes to cloud computing. While designing an application, make sure that infrastructure should support high scalability, performance and can handle multiple customers at the same time without any issues like synchronization. IaaS and SaaS are two main concepts of cloud computing. Multi tenancy in IaaS is achieved by sharing hardware, database and servers. In SaaS multi tenancy the tenants share same application means they have a common database or even can have a same table. We have to see the security levels in both cases and see at what level the security is compromised in favor of flexibility. [5]

The contribution of this paper will be to see the possibilities that how can we achieve custom multi tenancy. We will see what measures we have to take to implement the custom multi tenancy.

The paper is distributed into different sections. Section II includes different approaches of multitenancy. Section III includes Influencing factors while section IV explains different limitations on multitenancy. In section V a new architecture approach is proposed.

II. Multi tenancy approaches

A. Infrastructure Layer Tenancy

System Infrastructure can contribute to multi tenancy by using following approaches.

i. Virtualization

Virtualization is a concept where we run multiple operating systems on a single hardware by sharing all available resources. People say that virtualization is a strategy to cut down the cost of the system: its true, but technically it opens a way to implement cloud computing. We can implement the virtualization concept by having VMM virtual machine monitor. Virtualization provides the core attributes like scalability and flexibility. [2][4]

ii. Multiprocessing

System infrastructure can be used to initiate multiple processes for different operation. Each request is served by a different process to allow multiple users to access the system. The sharing of resources can be performed by setting a protocol which should be based on time. We can set the priority of a particular process to be served at urgent bases.

iii. Hybrid Approach

This technique uses the both above concepts. Virtualization approach gives us multiple virtual machines and on each machine we can implement the second multiple process technique. This approach is good for higher scalability and flexibility but more technicalities are involved in it.

B. Data Layer Tenancy

Generally for each distinct application we have a database. It's not ideal approach as it requires more cost and maintenance effort. To achieve multi-tenancy we need to share either the database schema or database logic. Multi-tenant database table is shared among all tenants. Each tenant has its own partition where its data is placed. When a user of a particular tenant is authenticated then that particular tenant has access to all data resides in the database. The data of different tenants is stored in different storage areas. This allows us the distribution of data with respect to tenants within a same table. This approach gives us the opportunity to take measures to increase performance and maintenance. In a multitenant database table there are some default values which are common for all tenants. [7]

When user is authenticated by a tenant user information is stored and then user can access the multitenant database. Now if an authenticated user of a particular tenant wants to see the data about user whose USR_ID is A3. Then the result of that statement will be the information of that requested user if that user exists in that particular accessed tenant. If other tenant has same ID then it will not be provided because the user is not authenticated for the other tenant. [8]

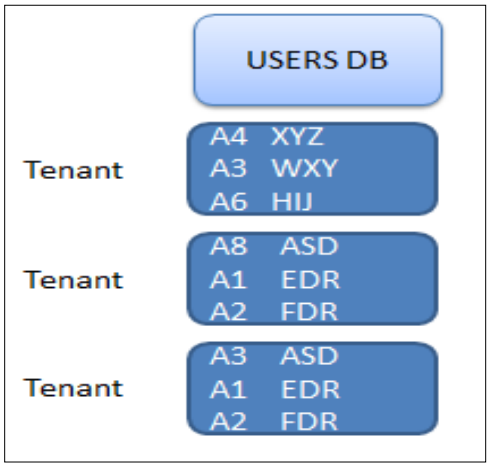

Figure 1 Tenant distribution in Multi-tenant database

The above Figure-1 shows tenant distribution in Multi-tenant database

There are also shared tables in a database which holds the data for all tenants. This data is accessible to all the tenants. This data is not tenant specific and is universal data for all tenants. In this case any authenticated user which does not belong to any specific tenant can also access the data of that table. Shared table will be part of the database but will not be a multitenant table. A particular tenant will have access of its own available data in tenant's table as well as data available in shared table. [1]

To further manage the database multitenant table the tenant ID should be introduced. This will be necessary to get the data about a particular tenant. In the Fig if we want to see the data about tenant 1 we can iterate the values of that particular tenant by using for each statement.

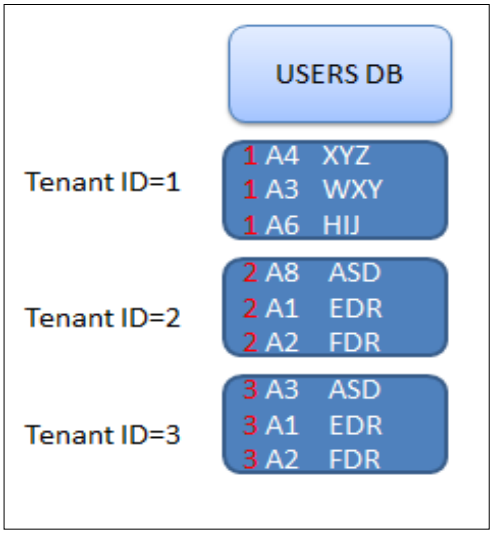

Figure 2 Tenant ID for each tenant

i. Master Tenant

There is a need of special type of tenant which can access the data of all other tenants. The purpose of this master tenant will be to maintain the other tenant's data as well as manage changes in other tenants data. This is very identical to the concept of admin user. There can be multiple master tenants of a database. For example if Main branch of any bank want to see the reports of its particular branch then it should be declared as a master tenant. [6]

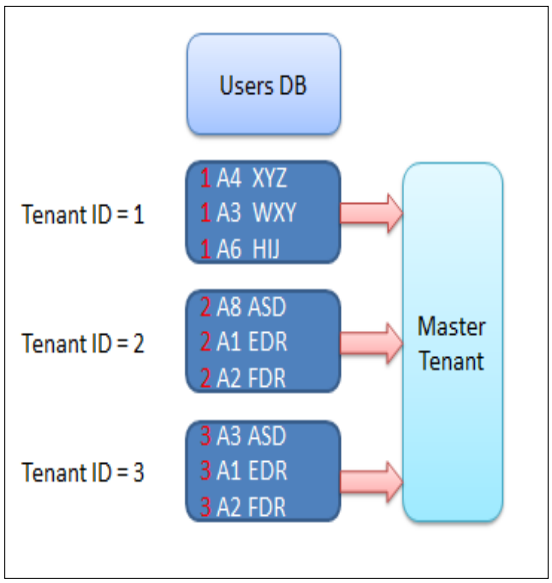

Figure 3 Master tenant access

Figure-3 showing Master tenant have access to all tenants

C. Application Layer Tenancy

Multi tenancy can be implemented at the application layer as well. For each tenant we can have a separate code module or a single module will be shared by all tenants. The later approach is better suited for multi tenancy. While making shared code module make sure that tenant related enhancement can be incorporated in future. Further strategies like code partitioning should also be implemented to enhance scalability and performance.

III. Factors Influencing Multi Tenancy

While implementing multi tenancy some factors must be analyzed to successfully implement the multitenant application.

i. System Cost
ii. System Complexity
iii. System Performance
iv. System Security
v. System Scalability
vi. System Flexibility

A. System Cost

It's the primary factor which is highly influential on multitenant applications in a positive way. Instead of dedicating separate infrastructure for each individual tenant we share the infrastructure by virtualization. This results in significant decrease in system cost. We can combine the extra capacity of each tenant and this combined unit of capacity then can be utilized by each tenant using multi tenancy.

B. System Complexity

Multitenant applications are implemented by sharing resources in different layers. For sharing resources we need a protocol for sharing the resources among different tenants. This increase the overall complexity of the system. The complexity can be minimized with the automation of management of multitenant application.

C. System Performance

Multi tenancy will affect the performance of the system. Sharing the resources can increase and decrease the overall performance of the system. It depends on the architecture of the system and different parameters like processing, bandwidth and power. Here one tenant can affect the performance of other if they are logically not isolated.

D. System Security

Security is the major concern when it comes to multi tenancy. Multiple clients are sharing the resources so there is a security concern. There are two approaches of security which are overall system security and individual security level of a tenant. Again it depends upon the architecture of the system. Customization also allows the user to set its own security and privacy levels. But the multitenant systems are less secure then dedicated systems.

E. System Scalability

The System architecture should be designed such that it should be scalable considering the future requirements. There is also a possibility of dynamic scalability with the help of modern virtualization techniques. Dynamic scalability will quickly increase the system capacity.

F. System Flexibility

Most of multitenant applications are cloud based and as the technology evolve; different cloud providers also implement these changes. So the architecture of SaaS application should be designed such that it can adopt the changes made by the cloud providers. Architecture can be designed to have loosely coupled components. This will make the application more agile and flexible.

IV. Limitations on Multi Tenancy Due to Tradeoffs

There are some limitations on multi-tenant applications due to the above stated factors. These limitations are due to tradeoffs between some factors. When designing the multi-tenant applications architecture these tradeoffs are strong hidden constraints.

A. Cost Vs. Complexity

Multi-tenant applications are shared application. Sharing the resources actually reduce the overall system cost. It looks a major material benefit because your infrastructure will require less money. The other side of the mirror is the technical disadvantage in the form of complexity of the system. This requires a technically more intelligent and automated system. More technical expertise is needed to build a system. Complexities like configuration data and maintenance of the system make the task more difficult.

B. *Security vs. Scalability and Flexibility*

Security is a key credential when it comes to a standard system. It specially becomes important for cloud based applications. But by sharing resources to achieve multi tenancy we have to compromise on security and it's a big risk. The positive aspect is that resource sharing makes application more scalable and flexible. This makes adaption and enhancement easy to implement. To ensure security some extra measures like encryption and authentication for each tenant are needs to be taken. These measures are different for each layer.

Figure 4 shows the constraints on multi tenancy.

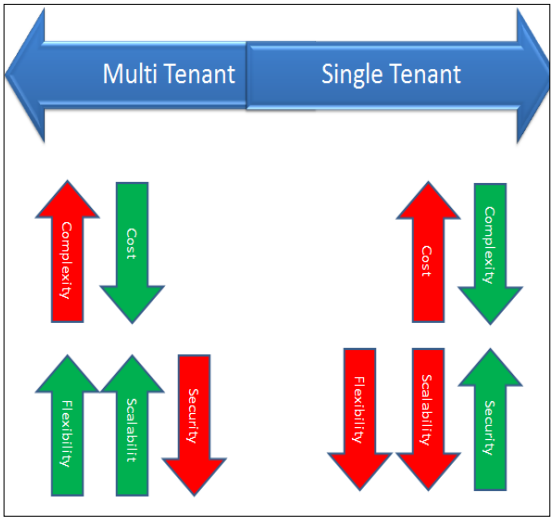

Figure-4 Constraints on multi tenancy

V. Customization is the Solution

In the previous section we have discussed hidden constraints on multi tenancy. The solution to this problem is customization at each layer and for each SaaS application.

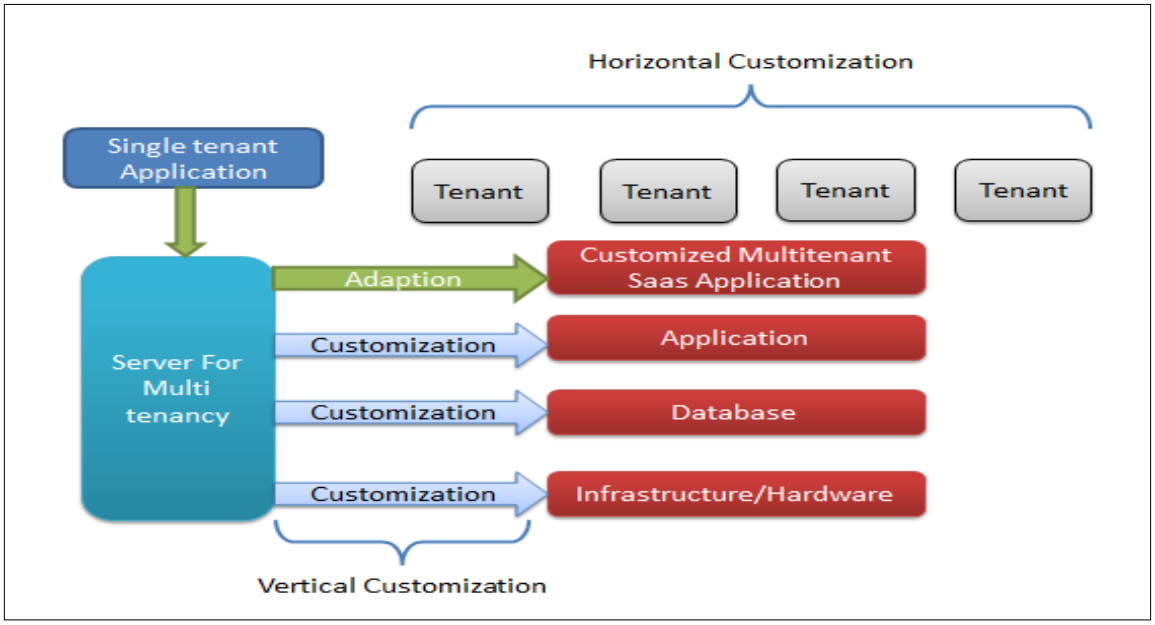

Figure 5 Customized Multi-tenant model

The above figure is showing model for customized Multi tenancy.

The above figure displays the model architecture which should be adopted in order to mitigate the effects of the hidden constraints introduced as a result of multi tenancy.

Multi tenancy server is responsible for adapting the single tenant application and converts it to multitenant application. For this purpose the server will read the existing structure of the data model and then for conversion make the according Meta model for multitenant application [9]. Then from that Meta model make the required database. The server will also generate the tenants for the converted application using tenant ID approach stated earlier. The server will also ensure the isolation between different tenants and different SaaS applications. Multitenant server also allows monitoring and controlling the SaaS users.

Horizontal customization is done by the server which is also called per tenant customization. By horizontal customization each tenant can be built by different approach for different user. This will solve issue of tenant isolation. Vertical customization is the approach in which we can have different SaaS application with different level of multi tenancy. Each SaaS application will have its own profile which will distinguish it from other apps. Vertical customization is per SaaS customization. This approach of using vertical and horizontal customization collectively will enhance the security of multitenant applications. The isolation between different SaaS apps and tenants will be increased using this approach. Enhanced customization is the order of the modern day

applications. Custom multi tenancy can be achieved by this. Scalability and flexibility is also increased in this model and security is also strengthened. The configuration will also be different for each application according to its profile by determining which layer multi tenancy is used in that particular application. This model architecture can be embedded in to modern architecture styles (MVC, Sposad).

In software engineering multi tenancy is the modern technology and work is going on in this regard. We have discussed the constraints and proposed the model approach to mitigate the effects of those and to achieve customized multi tenancy.

VI. Conclusion and Future Work

In recent days Multi-tenant applications are used in every business applications. The security and customization are two key features of an application. Lot of work is going on in this field. We have analyzed the deficiencies in multi tenancy and made a model to achieve customize and secure the multi tenancy. Cloud computing is the resent and the future of software engineering world and multi tenancy is the main drive. There is a strong need to work on security and complexity of multi-tenant applications. Future work should be on customization which is actually to give more importance to customer as they are the main drivers of a business. Security is also a major issue but for future work should be on customer collaboration as it will also improve the security of the system and acquire customer satisfaction.

References

[1] progress.com “Open edge Multi tenancy Overview” by Richard Banville , Rob Holzel 2012.

[2] juniper.net “Securing Multi Tenancy and Cloud Computing ” 2012.

[3] Towards an Architectural Style for Multi-tenant Software Applications by Heiko Koziolek 2010.

[4] Efficient Distribution of Virtual Machines for Cloud Computing by Matthias Schmidt, Niels Fallenbeck, Matthew Smith, Bernd Freisleben 2010 18th Euromicro Conference on Parallel, Distributed and Network-based Processing.

[5] An Open Multi-Tenant Architecture to Leverage SMEs by Irene S. Harris and Zubair Ahmed, European Journal of Scientific Research 2011.

[6] Z. H. Wang, C. J. Guo, B. Gao, W. Sun, Z. Zhang, and W. H. An, "A Study and Performance Evaluation of the Multi-Tenant Data Tier Design Patterns for Service Oriented Computing," E-Business Engineering, IEEE International Conference on, vol. 0, pp. 94-101, 2008.

[7] Progress.com “Saas Architecture” 2008-2009

[8] S. Aulbach, T. Grust, D. Jacobs, A. Kemper, and J. Rittinger,“Multi-tenant databases for software as a service: schema-mapping techniques,” inProc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD’08). New York, NY, USA:ACM, 2008, pp. 1195–1206.

[9] Convert your web application to a multi-tenant SaaS solution by Scott Chate ibm.com/developerworks/cloud/library/cl-multitenantsaas/

AUTHORS’ PROFILES

Engr. Muhammad Fahad Khan: is a PhD Scholar in the Department of Computer Engineering at University of Engineering and Technology Taxila, Pakistan. He has received his Master degree in Computer engineering from university of Engineering and Technology Taxila, Pakistan in February, 2010. He graduated from University of Engineering and Technology Taxila in Software Engineering in September 2007. His areas of interest are Video Summarization, Computer vision, Software Designing, Design Patterns, Software Requirements Analysis and Mobile application development.

Engr. Mirza Ahsan Ullah is doing MS in Software Engineering from Department of Software Engineering, University of Engineering and Technology Taxila, Pakistan. He has received his bachelor's degree in Computer engineering from university of Engineering and Technology Taxila, Pakistan in August, 2010.Also, he is a professional software developer specialized in ERP systems. His areas of interest are software development, Mobile application development, Data mining, Data bases, Digital image processing, Computer vision and Computer Networks.

Engr. Aziz Ur Rehman is a MS Scholar of Software Engineering from Department of Software Engineering, University of Engineering and Technology Taxila, Pakistan. He has received his bachelor's degree in Electrical engineering with specialize in Telecom from Comsat institute of Information Technology Lahore , Pakistan in July, 2010. His areas of interest are Mobile application development, Data bases, Digital image processing and Computer Networks.

