Computing, Information Systems & Development Informatics Journal
Vol. 8 No. 4, December, 2017 - www.cisdijournal.org

Transcript Request Processing System: A Multi-Tenant Framework

Amadin F.I. & Obienu A.C.
Department of Computer Science
University of Benin
Benin City. Nigeria.
[1]frankamadin@uniben.edu, [2]obienuac@gmail.com

Ejiofor, C. I
Department of Computer Science
University of Port Harcourt
Choba, Rivers State, Nigeria
christopher.ejiofor@uniport.edu.ng

ABSTRACT

The enormous time taken in processing academic transcript has impedes students' progress, whereby several applicants lose their admission, scholarship and job opportunities. These unpleasant circumstances have necessitated the search for better options. This search for better options resulted to the concept of multi-tenant database as a central repository for students' record. Since coming into the limelight around 2010, multi-tenancy has been gaining momentum in the computing world as a better means of database deployment with greater accessibility. Different operations and operators have benefitted from its ease of use, with many spheres of life been affected. Transcript processing is another challenging data processing area that needs the 'Macedonia call' for multi-tenant database to enhance real-time accessibility by user. This study therefore proposed a multitenant framework as a viable solution to realizing a Transcript Processing System that is timeless, efficient, and cost effective.

Keywords: Transcript processing system, Multi-tenancy, Database, Software architecture.

CISDI Journal Reference Format
Amadin F. I., Obienu A. C & Ejiofor, C. I (2017): Transcript Request Processing System: A Multi-Tenant Framework. Computing, Information Systems & Development Informatics Journal. Vol. 8 No. 4. Pp 1-6.
Available online at www.cisdijournal.net

1. INTRODUCTION

In recent times, the introduction and use of database management systems to manage information relating to students record has become popular. Transcript processing literatures show that most institutions in Nigeria still encourage their enterprise units (departments, faculties, and schools) to manage students' records independently and in their own ways. This has kept students' records disintegrated in different database systems located at various faculties. These days, there are a lot of databases containing students' record, scattered over many machines within the same institution. These records are needed for the preparation of academic transcript. Ebietomere and Ekuobase (2014) defines an Academic Transcript as a student's official record of courses taken, grades obtained and the honours or degrees conferred on the student during the period of studentship, vital document to the furtherance of a student's chosen career or academic pursuit. To Cristian et al, (2013), an academic transcript is a certified statement detailing a student's complete academic record at the University or higher institution of learning. In other word, it is an academic summary of a student's stay in an academic institution. Many school leavers and higher degree aspirers desire a faster and better approach to the procedures of applying and getting transcripts across to their employers or institution of learning.

However, the process of gathering, summarising and reporting student's academic transcript is becoming cumbersome. This is because the needed information to process student's transcript is scattered across multiple isolated information islands in a sea of computers and database systems, thereby, making it difficult for transcript officers to access applicant's record when needed. Moreover, they have to visit the applicants' department before they can have access to such record, since there is no common, standardised procedure for capturing, processing and storing this information. This makes the transcript processing susceptible to unnecessarily delayed in the process, as well as, difficulty in locating and accessing the needed information from multiple isolate databases containing students' records. On the other hand, this impedes students' progress, whereby several applicants lose their admission, scholarship and job opportunities.

As innovation and creativity continues, software developers continue to build applications that require new ways of accessing and manipulating data, which will aid to reduce cost of maintaining these databases, as well as minimize the time taken to process students' transcript. It is against this background that the researchers proposed the need to adopt an architectural framework that will provide for better management platform. This can be greatly achieved by consolidating the transcript system using the concept of multi-tenancy.

2. THE CONCEPT OF MULTI-TENANCY

Software as a Service (SaaS) constitutes a fast-growing business model for the sales of software that is based on the principle of outsourcing. With Software as a Service, a service provider hosts an application or software on its infrastructure and delivers it as a service to several tenants. According to Olumuyiwa et al. (2015), A Multi-Tenant Database (MTD) is a way of deploying a Database as a Service (DaaS). It refers to a principle where a single instance of a Database Management System (DBMS) runs on a server, serving multiple clients (tenants) as depicted in figure 1. In other words, it is a database architecture that allows multiple databases (pluggable databases) to be consolidated into a container database without changing their applications. Multi-tenancy provides the benefit of managing these multiple databases as a single database, and yet retains the isolation and resource prioritization of individual databases and DBMS service can also be out sourced for some reasons that are obvious to the tenants. The concept of Container Databases (CDBs) and Pluggable Databases (PDBs) is a new concept, and is fundamental to providing the 'missing link' in database consolidation (Morle, 2013; Amadin and Obienu, 2015). A CDB can contain one or more PDB, and a PDB is the actual 'database' from the viewpoint of the application. PDBs can be plugged into and unplugged from CDBs using simple commands, and they can be cloned and moved to other CDBs. All the PDBs that are plugged into a CDB share a single instance and can be resource-managed by a single set of controls within the CDB.

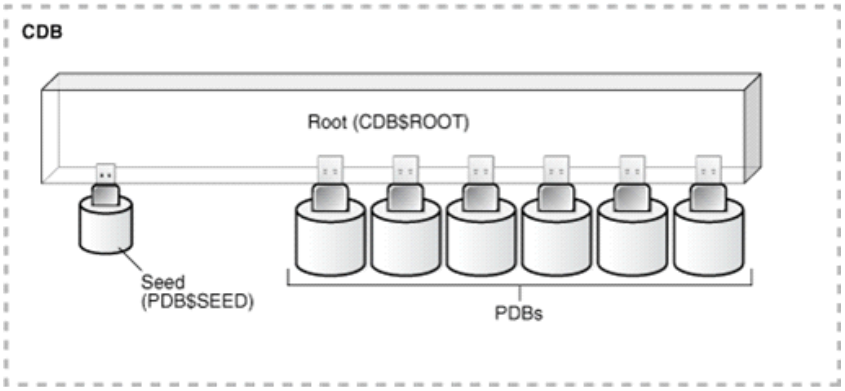

Figure 1: Multitenant Database

Since coming into the limelight around 2010, multi-tenancy has been gaining momentum in the computing world as a better means of database deployment with greater accessibility. Different operations and operators have benefitted from its ease of use, with many spheres of life been affected. Transcript processing is one challenging data processing area that needs the 'Macedonia call' for multi-tenant database to enhance real-time accessibility by user. Although multi-tenant database require more complex implementation, it is significantly economical to maintain over time, especially as the number of users it serve grows (Chappell 2012).

3. THE NEED FOR MULTI-TENANT SYSTEM

Generally, information about students as well as examination results for each semester are expected to be properly calculated and preserved by educational institutions (Abel and Abu, 2013). Despite the existence of several student results' computation systems, many institutions still compute and store student's results manually and this can be traceable to high cost of acquisition (Orobor, 2015). The amount of time spent during preparation of students' transcript is high, such that there is an urgent need to greatly reduce it. This is also attributed to the immense volume of students records generate and stored in recent time. Studies have shown that multi-tenant database is essential in the educational sector for storage and retrieval of students' records (Amadin and Obienu, 2015). Currently, Educational institutions in Nigeria manage student information independently and in their own ways. There is no common, standardised procedure for capturing, processing and storing this information. These days, it is fairly common to find hundreds, or thousands of databases scattered over many machines within an institution with reasonably large size holding students' record. This has kept the needed information to process students' transcript disintegrated in different database systems.

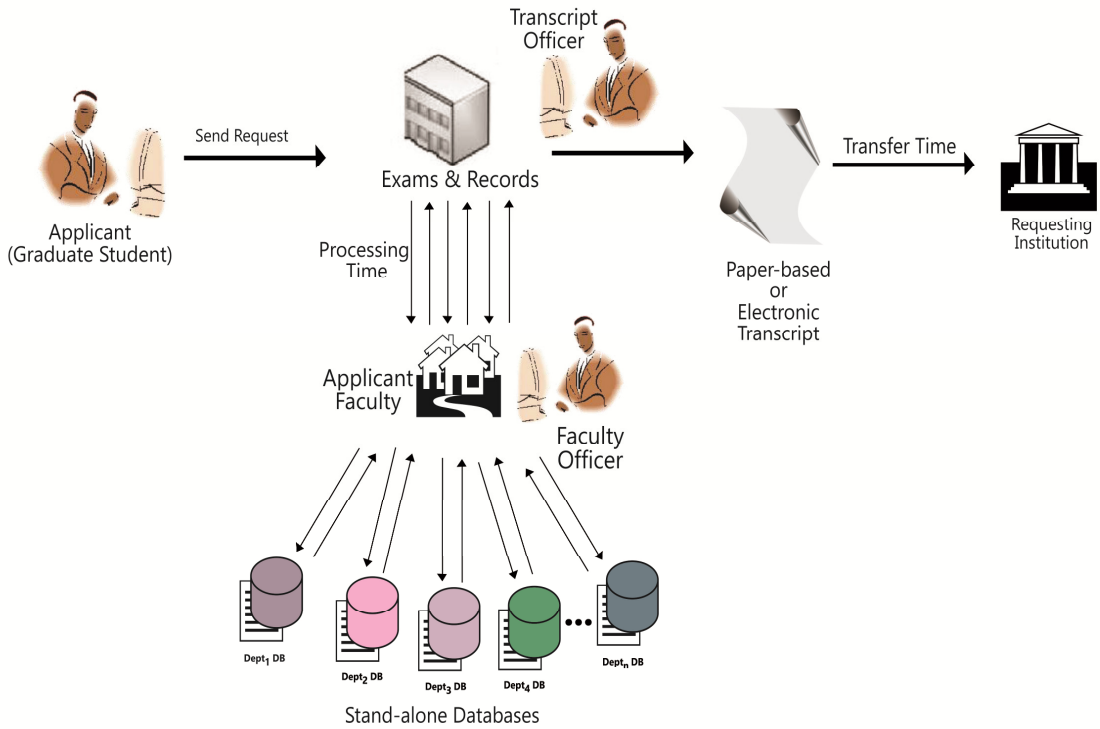

Figure 2: Current Scenario for Transcript Requesting Processing

This current scenario is described in Figure 2. The student, who wishes to obtain a transcript from a particular college/university, has to first send a request to the exams and record department responsible for issuing the transcript, either by mail, fax, or by person. Then the transcript officer visits the applicants' department for his/her record. Afterward, the office issues a paper-based transcript after some processing time. Then this transcript is mailed to the particular institution the student requested. This in turn involves some transfer time. If the requested institution is international, then the transfer time is even longer. Thus the student has to make sure that sufficient time is given for the whole process to unfold so that the transcript reaches the destination on time.

On the other hand, the institution issuing the transcript too has to utilize considerable time and efforts during the issuance of the paper-based transcript. For each request, it has to repeat the whole process of retrieving the student information, issuing transcripts, and transferring them to each and every institution the student requested. From the admission office point of view, this is susceptible to:

I. Error while collecting documents and information from several computers and databases, and assemble them into a single application package for a particular applicant.
II. Unnecessarily delayed in preparing the transcript, since the transcript officer has no direct access to the database systems containing the student's record.
III. It also makes it easy for one to alter the scores of the student. This makes Nigeria's education system susceptible to fraud.
IV. High operational cost associated with current model of database deployment.
V. Low utilization of these dedicated database servers.

This makes efficient processing of transcripts impossible and thus contributes to the overall delay of the admission process. Moreover, this impedes students' progress, whereby several applicants lose their admission, scholarship and job opportunities. Unless an accurate and effective method is adopted, results computed are apt to convey misleading information to decision makers (Eyo and Ofoegbu, 2012). Taking University of Benin as a case study for this work, and considering the fact that students' record are disintegrated in different database systems, it becomes necessary to analyze and design a multitenant framework as a viable solution to realizing a transcript processing system that is timeless, efficient, and cost effective.

4. THE PROPOSED SYSTEM FRAMEWORK

With the aim to address the above drawbacks associated with the current transcript processing system, Multitenant Transcript Architecture (MTA) is proposed. This new architecture is adopted because it will provide flexibility to facilitate the student application process (Amadin and Obienu, 2015).
The Multitenant architecture of the transcript request system will be implemented through the concept of Pluggable Databases (PDBs). A PDB is a portable collection of schemas, schema objects, and non-schema objects that appears as a non-CDB. A PDB is essentially the same as what we have always used in the past, with all the generic data dictionary information removed. These PDBs cannot execute by themselves, they need to be plugged into a Container Database (CDB), which contains all the data dictionary information that is absent from the PDB. A CDB can host multiple PDBs, and so allows a single set of data dictionary information, background processes and memory allocation to be shared across multiple PDBs.

Figure 3 provides an overview of transcript request processing using the concept of multi-tenant architecture. In this scenario, the applicant can interact with University web interface (Provided by the institution the applicant graduated from), which in turn interact with the web service (provided by the requesting institution). The Application Server receives the request and confirms details provided by the applicant (E.g, Requesting Institution address or email).

Then, it implements a distributed search of metadata regarding the system repositories. The output of search queries provides metadata information about applicant information and a link of their location. The transcript officer browse metadata information then selects a link of interest. Then access of data content that resides in multitenant databases is enabled. In accordance with this interface, the retrieval engine supports a restricted SQL query language to query the data.

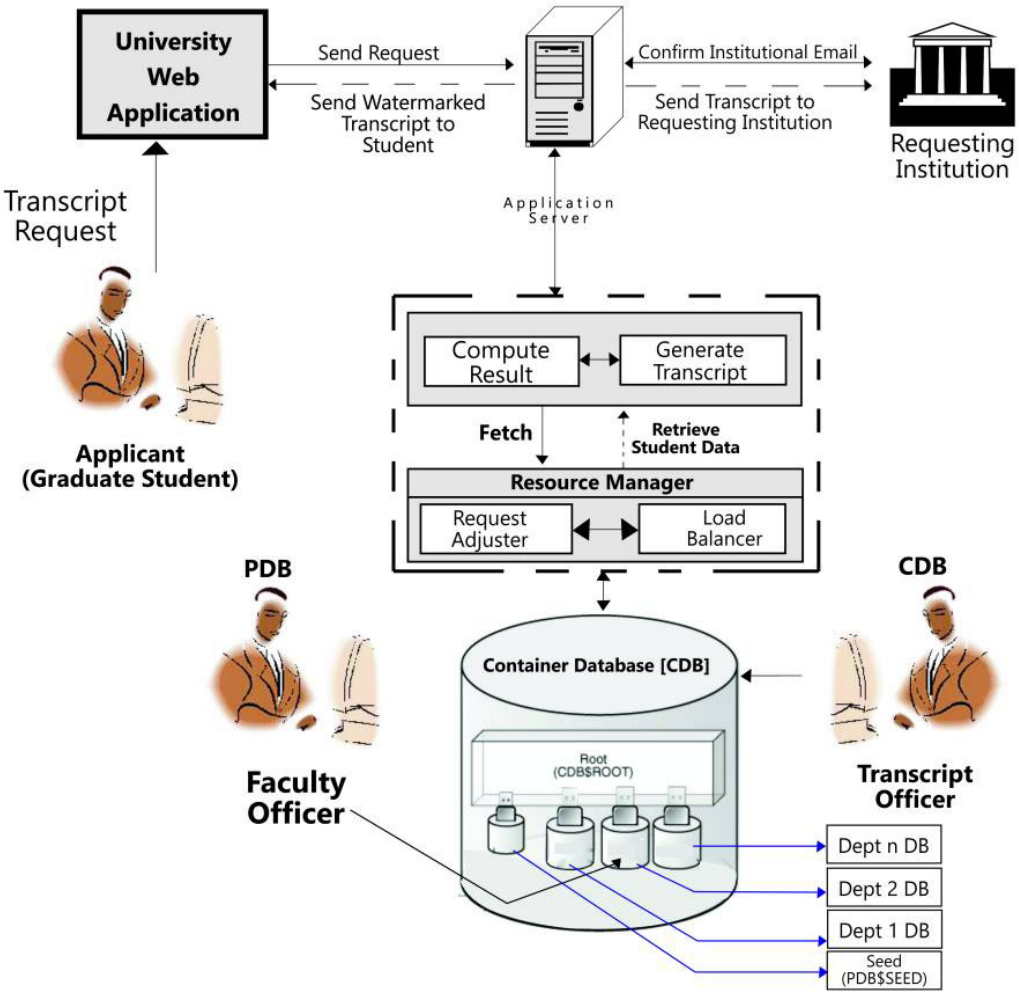

Figure 3: Proposed Framework with Multi-tenant Database

To support a distributed access to data, the query has to be transformed in a format supported by the system. A software module, called query adjuster, takes charge of the dynamic translation of the original query into a supported format when queried. This engine maps the user global query into local queries, each for a different query execution engine of some remote publishing site, and a composition query for producing the final result. The query execution engine has a runtime system to integrate the results of local queries. The container database has five containers: the root, seed and three Pluggable databases which represent each department database. The root is a collection of schemas, schema objects and non-schema objects to which all pluggable databases belong. Every container database has only one root which is required to manage all pluggable databases. Pluggable database is a user created set of schemas and object. Each pluggable database has its own dedicated applications.

The Seed is a template for creating new PDBs. So when queried, the system searches the entire PDBs to locate the required information. This system will provide flexibility to facilitate the transcript request process.

5. CONCLUSION

The paper briefly discusses multi-tenancy and their potential for re-shaping the software development industry that can help both the students and the academic institutions in transcript processing. It showcases the technologies behind multi-tenant databases and therefore, proposed a Multitenant Architectural Framework as a viable solution to realizing a Transcript Processing System that is timeless, efficient, and cost effective.

REFERENCES

1. Amadin, F. I. and Obienu, A.C. (2015), Consolidation of School E-Libraries Database Using Multitenancy Approach. Afr J. of Comp & ICTs. Vol 8, No. 4. Pp 11-18.
2. Chappell D., *Building Saas Applications on Windows Azure: Things to Think About Before You Start*, Chappell & Associate, 2012. [Online]. Available at: http://www.davidchappell.com/writing/white_papers/Building_SaaS_Apps_on_Windows_Azure-Chappell_v1_0.pdf
3. Cristian C., Lev T., and Eugenia T. (2013)."Secure and convenient computerized transcript system:6.033 Design Project",Available online @ http://citeseerx.ist.psu.edu.
4. Ebietomere, E. P. and Ekuobase, G. O. (2014), "A Framework for Automating Transcript Processing", J. of NAMP, Vol. 27, No. 1
5. Eyo O. U and Ofoegbu F. A, "A Software Application for University Students Results Processing", *Journal of Theoretical and Applied Information Technology*, 35 (1), 2012.
6. Morle, J. (2013). "Database Consolidation with Oracle 12c Multitenancy". [Accessed 23 August 2015]. Available at http://www.implementerslab.com/atifacts
7. Olumuyiwa ,M. O, Mary, G.,& Kevan, A. B. (2015). "Predicting the Impact of the Factors That Influence the Adoption of Multi-Tenant Databases". Proceedings of the International Conference on Computer and Information Science and Technology Ottawa, Ontario, Canada, May 11– 12, 2015 Paper No. 105
8. Orobor, A. I. (2015). "A Novel Framework for Student Result Computation as a Cloud Computing Service." *American Journal of Systems and Software*, vol. 3, no. 1: 13-19. doi: 10.12691/ajss-3-1-2.