Using an Interaction Model as a Requirement for Distributed Database Design to Enhance Data Access and Reliability in Higher Learning Institutions: A Study Conducted at Saint Augustine University of Tanzania

Anthony Mwombeki
Information Systems Department–College of Informatics and Virtual Education
University of Dodoma, UDOM
Dodoma, Tanzania
Email: mwombekianthony [AT] gmail.com

Abstract—**Data access and distribution is one of the key challenges facing Tanzania higher learning institutions especially those with distributed campuses in various regions and districts within the country. Saint Augustine University of Tanzania, having distributed campuses, experience data access, reliability and distribution challenges due to the presently used centralized database systems resulting to problems in data access and distribution as well as data reliability in case of system failure.**

This study explores the challenges resulting from the currently used centralized database systems and recommends appropriate solutions which can in turn be employed to utilize the current high bandwidth network within the country to design distributed database systems by making appropriate decisions on the placement of data and programs across different University campuses as sites of a computer network and probably designing the network itself.

The study proposes effective analysis of institutions key tasks with appropriate interaction models designed based on those tasks being fundamental to distributed database systems design. The top down design approach which involves designing of distributed database systems from scratch was employed with homogeneous distributed database environment being employed across various sites of a computer network.

Sybase replication server architecture was employed during replication design to enhance sharing of information across campuses of the University. The study finally proposes the use of appropriate interaction models for distributed database systems design and implementation to enhance data access, distribution and reliability in higher learning institutions specially institutions with distributed campuses.

Keywords—DDBMS, SAUTDDBS, Top down design, Interaction model, Horizontal fragmentation, Replication server.

I. INTRODUCTION

Distributed database system is the union of what appear to be two diametrically opposed approaches to data

processing which are database systems and computer networks [1]. Computer networks promote a mode of work that goes against centralization since the most important objective of database technology is data integration and not data centralization. Integration is possible without centralization, since integration of databases and networking does not mean centralization. Therefore, the main purpose of distributed database systems is to achieve data integration and data distribution transparency [1].

A distributed database system (DDBS) consists of two or more data files located at different sites on a computer network. Because the database is distributed, different users can access it without interfering with one another. A DDBS consists of distributed database management system (DDBMS) software. DDBMS is a collection of DBMS software whose function is to manage the DDBS and provides an access mechanism that makes data distribution transparent to the users. Thus, DDBMS software must periodically synchronize the distributed database at various sites to make sure that they all have constant data [1].

Higher learning institutions with distributed campuses particularly Saint Augustine University of Tanzania (SAUT) experience a challenge of how to improve data access in various campuses while ensuring data reliability in case of system failure. This is due to the fact that most higher learning institutions still use centralized database systems for various activities and processes involving both institutions members and their respective alumni. This challenge can be managed by designing and implementing distributed database systems through the use of appropriate models of

interactions making sure that all important parties, activities and processes are taken into account during the design process [1][2].

This research study proposes the use of an appropriate interaction model as an effective requirement for distributed database systems design where five campuses of SAUT have been taken as five sites of Saint Augustine University of Tanzania Distributed Database System (SAUTDDBS). The model can easily and effectively be used with the top down approach for distributed database design process. Thus, depending on data requirements at various sites, the database can then be fragmented into several fragments to ensure easy data access and reliability.

In Distribution design, the key problem is on how to make decisions about the placement of data and programs across the sites of a computer network as well as perhaps designing the network itself. However, this study proposes the use of appropriate requirement analysis tools in the early stages of distributed database design for simplification of distribution design process. The study proposes the use of an interaction model obtained as a requirement in the early stages of the top down DDBS design approach.

II. LITERATURE REVIEW

A. *Distributed and Centralized Database Systems Issues*

A distributed database consists of two or more data files located at different sites on a computer network. So, a distributed database system consists of two opposed approaches to data processing which are database systems and computer networks. Because the database is distributed, different users can access it without interfering with one another. However, the DDBMS must periodically synchronize the scattered database to make sure that they all have constant data [1][3].

Data is spread across multiple computer or servers which are not necessarily located in the same physical location. Data may however be accessible through an Internet based portal as a GUI for accessing data. The database requires software which is able to locate and index all of the data from different locations. Due to distributed data, different uses can access it without interfering with one another [1][3].

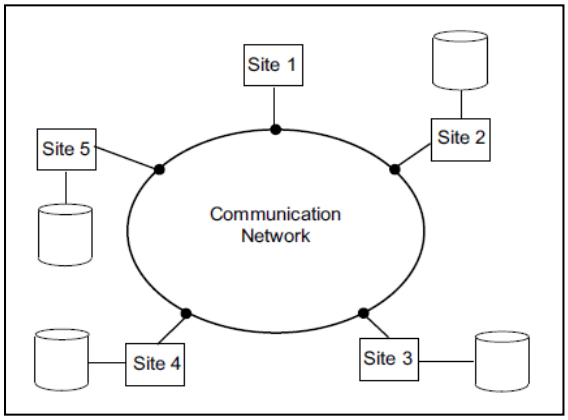


Figure 1. DDBS Environment [1]

DDBSs' plays a key role in increased reliability and availability, easy system expansion, reflection of organization structure while providing for protection of valued data which are not all held in one spot. With DDBSs', failure of one site does not bring the entire systems down due to high reliability as data is located on different sites of a computer network.

However, DDBSs' suffers from several complicating factors such as its complexity which results to high costs as increased complexity and more extensive infrastructure means extra labour costs. Security issue also affects DDBSs' as remote database fragments must be secured with the need for operating system to have the ability to support distributed environment [1][3].

A centralized database is the one located, maintained and managed in one location, unlike a distributed database. With centralized database, the database is located in one place so that it can be easily accessible and backed up. The databases are still accessible through Wide-Area Networks (WANs) and Virtual Private Network [1][4].

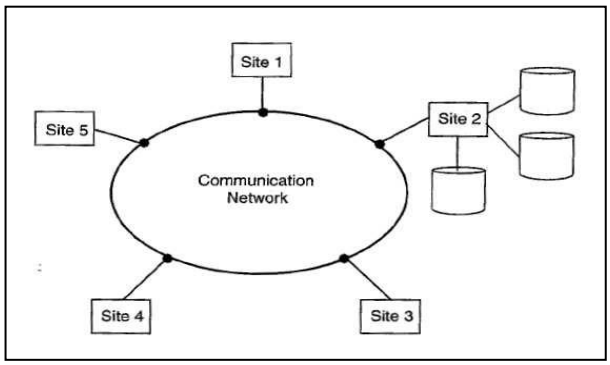


Figure 2. Central Database on a Network [1]

However, centralized databases are prone to bottlenecks as today's global enterprise may have many local area networks (LANs) in the form of international offices, as well as additional data servers and applications on the LANs. This is an issue because all staff may need access to the database [1][4].

B. *DDBSs' Design Approaches*

Distributed computing system design involves making decisions on the placement of data and programs in computer network nodes, and possibly designing the network itself. For distributed databases, and assuming that the network has been designed already and there is a copy of the DBMS software on each node in the network where data are stored, it remains to focus our attention on the distribution of data [3][5].

Distributed databases can be designed under two major approaches: top down design and bottom up design. The

strategies are very different approaches to the design process. But most applications are not so simple that it fits completely in one of these strategies, so it is important to know that these strategies should be used together as a complement to each other [6].

C. Top down design approach

This design approach is mostly used in designing systems from scratch. The process starts from a requirement analysis phase including analysis of the company situation where in this study; an interaction model illustrated in figure 3 assisted the author in analyzing SAUT situation. The process also involves defining objectives, and designing scope and boundaries. The next two activities are conceptual design and view design. Focusing on the data requirements, the conceptual design deals with entity relationship modeling and normalization [6][7].

It creates the abstract data structure to represent the real world items. The view design defines the user interfaces. The conceptual schema is a virtual view of all databases taken together in a distributed database environment. It should cover the entity and relationship requirement for all user views. The conceptual model should support existing applications as well as future applications. The definition of the global conceptual schema (GCS) comes from the conceptual design. The next step is distribution design [1][6].

The global conceptual schema and the access information collected from the view design activity are inputs of this step. By fragmenting and distributing entities over the system, this step designs the local conceptual schemas. Therefore, this step can be further divided into two steps: fragmentation and allocation [1][3][6].

Distribution design also includes the selection of DBMS software in each site. The mapping of the local conceptual schemas to the physical storage devices is accomplished through the physical design activity. Throughout the design and development of the distributed database system, the author constantly monitored, periodic adjusted and tuned the processes in order to achieve successful database implementation and suitable user interfaces [1][3].

D. Bottom Up Design

Top down approach is suitable when we are designing a DDBS starting from scratch. But it often happens that some databases already exist, and design activities must realize and integration. Bottom up approach is suitable for such environments. The starting point in designing bottom up is local conceptual schema [6].

E. Distributed Design Issues

The design of a distributed database introduces additional issues which complicate distributed database design. These issues include: how to partition the database into fragments, how many copies of a fragment should be replicated, how to allocate the fragments and replicas and how to test for correctness [5][7].

F. Fragmentation

Data fragmentation allows us to fragment relations to appropriate units of distribution since usually applications only deal with a subset of a relation. Each fragment can be stored at any site across the network. The decomposition of a relation enables the concurrent execution of several transactions. Three types of fragmentation strategies: horizontal, vertical, and mixed fragmentation. With horizontal fragmentation, a relation is fragmented into subsets of tuples (rows) based on the database information and application information. Each fragment consists of unique rows and is stored at a different site. The advantage of horizontal fragmentation is that it allows data locality by storing the fragments in the sites where they are most frequently accessed [8].

Vertical fragmentation can be achieved by fragmenting a relation along its attributes with the primary key attribute being available in each of the vertical fragments. However, in most cases simple horizontal or vertical fragmentation of a DB schema will not be sufficient to satisfy the requirements of the applications thus need for mixed fragmentation which consists of a horizontal fragment followed by a vertical fragmentation, or a vertical fragmentation followed by a horizontal fragmentation. The three fragmentation types should satisfy three correctness rules which are completeness, disjointness and reconstruction [3][8][9].

Completeness

- Decomposition of relation R into fragments R_1, R_2, . . . ,R_n is complete iff each data item in R can also be found in some R_i.

Reconstruction

- If relation R is decomposed into fragments R_1, R_2, . . . ,R_n, then there should exist some relational operator ∇ that reconstructs R from its fragments, i.e., R = R1∇. . .∇Rn

  ∗ Union to combine horizontal fragments
  ∗ Join to combine vertical fragments

Disjointness

- If relation R is decomposed into fragments R_1, R_2, . . . ,R_n and data item di appears in fragment R_j , then d_i should not appear in any other fragment R_k, k<>j

(Exception: primary key attribute for vertical fragmentation).

∗For horizontal fragmentation, data item is a tuple
∗For vertical fragmentation, data item is an attribute

III. METHODOLOGY

A. Study Design

This study firstly involved the analysis of key interactions, processes and activities taking place in various campuses of SAUT where face to face interviews, personal

observation and comparative analysis techniques for the strengths and weaknesses of centralized and distributed

database systems were employed. Secondly, an interaction model depicted in figure 3 illustrating key tasks at SAUT was designed based on the analysis process and used as a basic requirement for distributed database design. Finally, based on distributed database design approaches strengths, the top down approach was employed as a suitable approach for distributed database design at SAUT.

The study has employed both horizontal and vertical fragmentation techniques during fragmentation design process. However, this paper focuses mainly on horizontal fragmentation technique where both primary and derived horizontal fragmentations were employed. The study also employed Sybase Replication Server architecture for replication design.

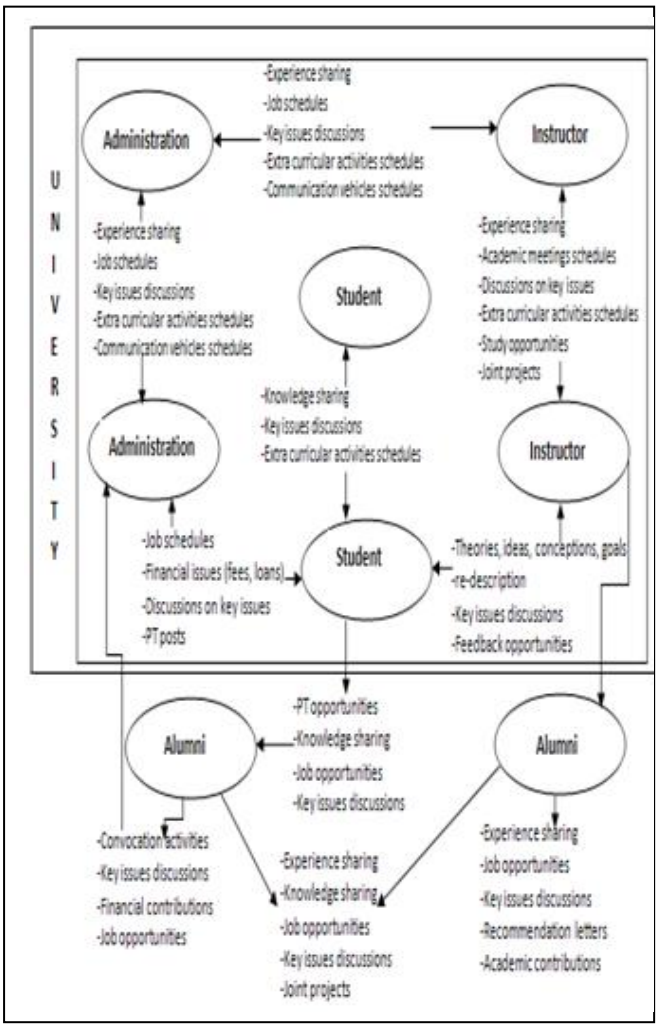
B. Proposed Interaction Model as the Base for DDBSs' Design in Higher Learning Institutions

Understanding key interactions among members of an organization plays an important role in effective DDBSs' design. This is because user requirements are usually required during the conceptual model design phase of the top down DDBSs design approach the requirements of which can be effectively obtained by clearly understanding important interactions and processes taking place among members of a respective organization [1][7][10].

This study proposes an interaction model depicted in figure 3 as the base for effectively obtaining user requirements. The model illustrates key interactions taking place among SAUT members and their respective alumni. This model is considered to effective since it does not exclude any important party among SAUT members and SAUT alumni and based on computer science and Internet security principles, the model goes further to prevent any external party to have access on it for anonymity problem control [10].

Figure 3. Interaction Model between SAUT Members and SAUT Alumni [10]

Top down design process is mostly used in designing distributed database systems from scratch. With top down design approach, it was easier to employ homogeneous distributed database technique since it is much easier to design and manage distributed databases designed using this technique. The technique also provides incremental growth



while allowing increased performance. Based on homogeneous distributed database environments requirements, same database management systems (DBMS) software have to be used across each site of the distributed database system.

Figure 4 illustrates the process of top down design. The process starts from a requirement analysis phase including analyzing of organization's situation, defining problems and constraints, defining objectives, and designing scope and boundaries the process of which has been key part of this research work with an interaction model in figure 3 illustrating.

The next two activities are conceptual design and view design. Focus on the data requirements, the conceptual design deals with entity relationship modeling and normalization [3][6]. It creates the abstract data structure to represent the real world items. The view design defines the user interfaces. The conceptual schema is a virtual view of all databases taken together in a distributed database environment. It should cover the entity and relationship requirement for all user views.

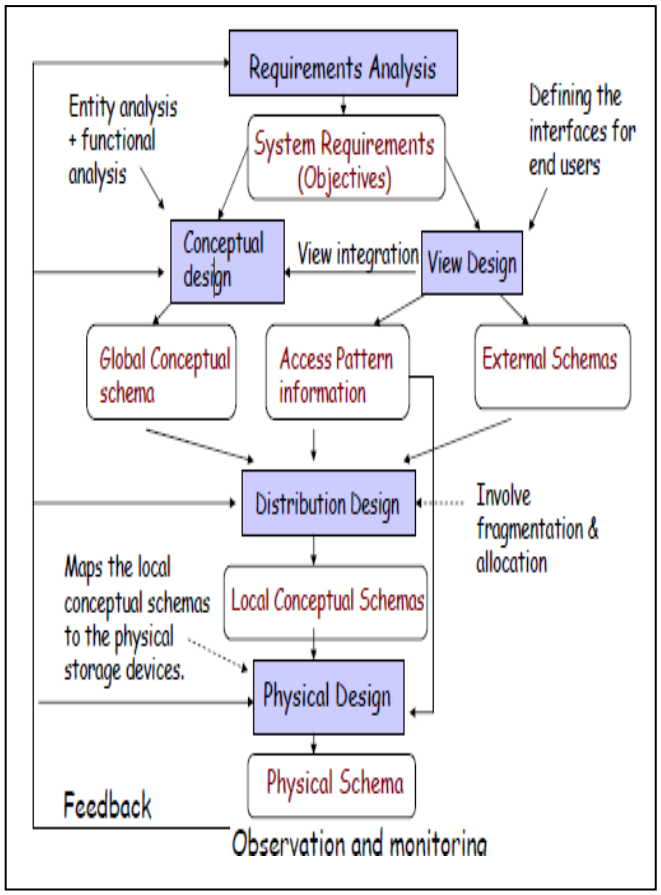


Figure 4. Top Down Design Approach [1]

Furthermore, the conceptual model should support existing applications as well as future applications. This study has provided the class diagram model based on the interaction model proposed as shown in figure 5.

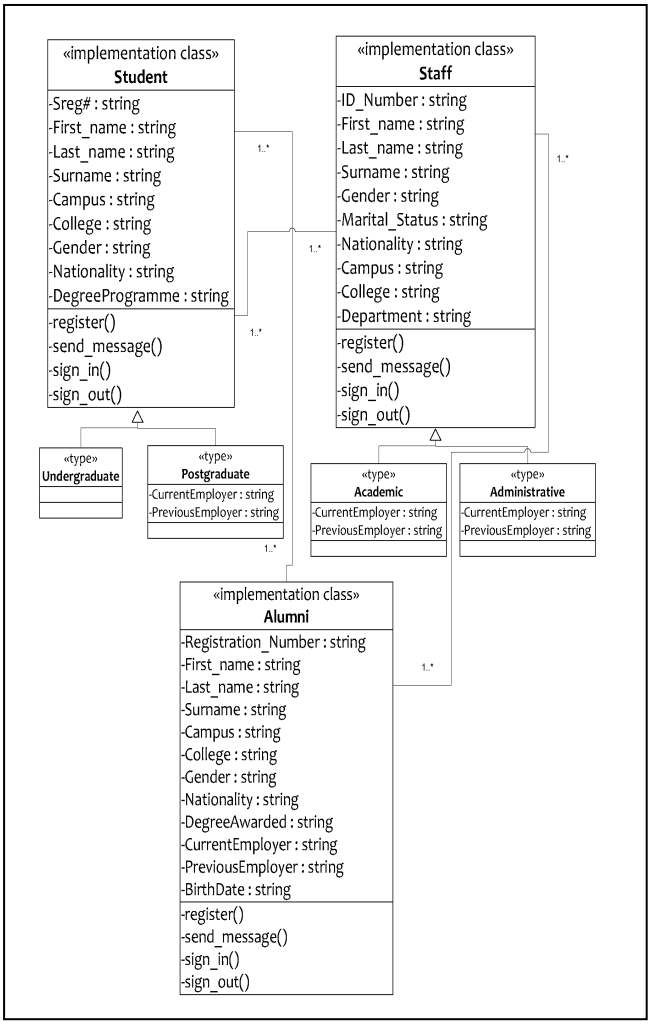


Figure 5. Class Diagram Used for SAUTDDBS Design

IV. SAUTDDBS DISTRIBUTION AND REPLICATION DESIGN PROCESSES

A. Sautddbs Campuses as Sites

The design process of SAUTDDBS involved designing the five campuses of SAUT as sites of the distributed database under discussion. Based on the class diagram designed from the interaction model in figure 3, the distribution design process was done.

During this distribution design process, fragmentation, replication and allocation requirements were accomplished based on data requirements at various sites.

B. Fragmentation

Fragmentation is a design technique to divide a single relation or class of a database into two or more partitions such that the combination of the partitions provides the original database without any loss of information [5][8].

A fragment i.e. horizontal or vertical of a database object in an object oriented database system contains subsets of its instance objects (or class extents) reflecting the way applications access the database objects [8].

Consider a relation with scheme *R*. The fragmentation of *R* consist of determining the number of fragments (sub scheme) *Ri* obtained by applying an algebraic relation on *R* (as operations on relations which show the logical properties of data). In this context, the fragmentation of data in this study was done under horizontal fragmentation technique.

The horizontal fragmentation of a relation R as applied in this study, was considered to be the subdivision of its tuples into subsets called fragments; the fragmentation was considered to be correct if each tuple of R was mapped into at least one tuple of the fragments (completeness condition).

An additional disjointness condition, required that each tuple of R be mapped into exactly one tuple of one of the fragments, was often introduced in SAUTDDBS design in order to control the existence of duplication explicitly at the fragment level (by having multiple copies of the same fragment). The resulted fragments R_i have the same scheme structure as well as collection R, but differ by the data they contain and are resulted by applying a selection on R [8].

Two versions of horizontal partitioning were applied during the design process which were primary and derived horizontal fragmentation. Primary horizontal fragmentation of a relation was achieved through the use of predicates defined on that relation which restricts the tuples of the relation with derived horizontal fragmentation being recognized by using predicates that were defined on other relations [9][11][12].

This process considered data fragmentation based on sub sets of relations as fragments to be appropriate units of distribution since usually applications only deal with a subset of a relation. Each fragment can in turn be stored at any site across the network. The decomposition of a relation enabled the concurrent execution of several transactions. Data fragmentation information is normally stored in the distributed data catalog for being accessed by the transaction processor to process user requests [8].

C. Primary Horizontal Fragmentation

A primary horizontal fragmentation is defined by a selection operation on the relations of a database schema. Given a relation R, its horizontal fragments are given by

$$R_i = \sigma F_i(R);\ 1 \leq i \leq w$$

where F_i is the selection formula used to obtain fragment R_i (also called the fragmentation predicate). Note that if F_i is in conjunctive normal form, it is a minterm predicate (mi). The algorithm applied in this study required that Fi be a minterm predicate [11][13].

Example, the decomposition of a relation Student into horizontal fragments by considering fragmentation based on student's Campus resulted into five horizontal fragments; Arusha, Bagamoyo, Iringa, Mbeya and Mwanza as Student_A, Student_B, Student_I, Student_M and Student_Z respectively is as follows:

Student_A = σCampus="Arusha" (Student)
Student_B =σCampus ="Bagamoyo" (Student)
Student_I = σCampus ="Iringa" (Student)
Student_M = σCampus ="Mbeya" (Student)

Student_Z = σCampus ="Mwanza" (Student)

Consider the Student Relation fragmented into five primary horizontal fragments as depicted in table I.

TABLE I. STUDENT RELATION

Stud_Reg#	**College**	**Campus**
SAUT/ARUSHA/2007_4657	EDUCATION	ARUSHA
SAUT/BAGAMOYO/2010_1123	NATURAL AND APPLIED SCIENCES	Bagamoyo
SAUT/BAGAMOYO/2012_2348	INFORMATION TECHNOLOGY	Bagamoyo
SAUT/IRINGA/2008_4507	ARTS	IRINGA
SAUT/MBEYA/2005_2997	SOCIAL SCIENCES	MBEYA
SAUT/MBEYA/20014_9997	ARTS	MBEYA
SAUT/MBEYA/2006_4567	SOCIAL SCIENCES	MBEYA
SAUT/MWANZA/2009_7890	MEDICAL SCIENCES	MWANZA
SAUT/MWANZA/2008_4588	MEDICAL SCIENCES	MWANZA

Note: Only some attributes and tuples of relation student are shown.

Below are the fragments of Student relation fragmented based on a campus where a student was enrolled.

Student_A = σCampus="Arusha" (Student)

TABLE II. STUDENT_A PRIMARY HORIZONTAL FRAGMENT

Stud_Reg#	**College**	**Campus**
SAUT/ARUSHA/2007_4657	EDUCATION	ARUSHA

Student_B = σCampus="Bagamoyo" (Student)

TABLE III. STUDENT_B PRIMARY HORIZONTAL FRAGMENT

Stud_Reg#	**College**	**Campus**
SAUT/BAGAMOYO/2010_1123	NATURAL AND APPLIED SCIENCES	Bagamoyo
SAUT/BAGAMOYO/2012_2348	INFORMATION TECHNOLOGY	Bagamoyo

Student_I = σCampus="Iringa" (Student)

TABLE IV. STUDENT_I PRIMARY HORIZONTAL FRAGMENT

Stud_Reg#	**College**	**Campus**
SAUT/IRINGA/2008_4507	ARTS	IRINGA

Student_M = σCampus="Mbeya" (Student)

TABLE V. STUDENT_M PRIMARY HORIZONTAL FRAGMENT

Stud_Reg#	**College**	**Campus**
SAUT/MBEYA/2005_2997	SOCIAL SCIENCES	MBEYA
SAUT/MBEYA/20014_9997	ARTS	MBEYA
SAUT/MBEYA/2006_4567	SOCIAL SCIENCES	MBEYA

Student_Z = σCampus="Mwanza" (Student)

TABLE VI. STUDENT_Z PRIMARY HORIZONTAL FRAGMENT

Stud_Reg#	**College**	**Campus**
SAUT/MWANZA/2009_7890	MEDICAL SCIENCES	MWANZA
SAUT/MWANZA/2008_4588	MEDICAL SCIENCES	MWANZA

D. Derived Horizontal Fragmentation

Derived horizontal fragmentation is used to splitting up a relation in dependence on another relation by applying semi-join operations [11][13]. Horizontal fragmentation of relation *S* based on the fragmentation of another relation R where R is already fragmented into R_1, R_2, R_3, …, R_n. Using the semi-join operator $S_i = S \infty R_i = S \infty \sigma p_i (R) = \pi S.*(S \infty \sigma p_i(R))$ fragmentation expression only refers to R.

Consider the derivation of derived horizontal fragmentation below; the relations have been distributed into other relations who depend on each primary horizontal fragment relation.

Distributing the relation R into to Student_A, Student_B, Student_I, Student_M and Student_Z for Staff_Student_Communication relation (i.e. Staff_Student_Communication relation was generated after breaking a many to many relationship existing between Student and Staff classes) we generate other five derived horizontal fragmentation based on the criteria for finding the number of times a particular student in a particular campus communicated with his/her staff for academic, research or consultation issues within a period of six study semesters.

Staff_Student_Communication (StaffID, Stud_Reg#, Consultation Times)

TABLE VII. STAFF_STUDENT_COMMUNICATION RELATION

StaffID	**Stud_Reg#**	**Consultation Times**
SAUT/ARUSHA/2012_1983	SAUT/MBEYA/2005_2997	4
SAUT/ARUSHA/2009_1007	SAUT/IRINGA/2008_4507	3
SAUT /ARUSHA/1999_1125	SAUT/BAGAMOYO/2010_1123	5
SAUT/BAGAMOYO/2013_2088	SAUT/MBEYA/20014_9997	2
SAUT /IRINGA/2001_0330	SAUT/MWANZA/2009_7890	1
SAUT/MWANZA/2014_0059	SAUT/ARUSHA/2007_4657	6
SAUT /MWANZA/2000_1983	SAUT/BAGAMOYO/2012_2348	4
SAUT /MBEYA/2012_1983	SAUT/MBEYA/2006_4567	5
SAUT /MBEYA/2010_2101	SAUT/MWANZA/2008_4588	3

Staff_Student_Communication1=Staff_Student_Communication ∞ Student_A

TABLE VIII. STAFF_STUDENT_COMMUNICATION1 DERIVED HORIZONTAL FRAGMENT

StaffID	Stud_Reg#	Consultation Times
SAUT/MWANZA/2014_0059	SAUT/ARUSHA/2007_4657	6

Staff_Student_Communication2=Staff_Student_Communication ∞ Student_B

TABLE IX. STAFF_STUDENT_COMMUNICATION2 DERIVED HORIZONTAL FRAGMENT

StaffID	Stud_Reg#	Consultation Times
SAUT /ARUSHA/1999_1125	SAUT/BAGAMOYO/2010_1123	5
SAUT /MWANZA/2000_1983	SAUT/BAGAMOYO/2012_2348	4

Staff_Student_Communication3=Staff_Student_Communication ∞ Student_I

TABLE X. STAFF_STUDENT_COMMUNICATION3 DERIVED HORIZONTAL FRAGMENT

StaffID	Stud_Reg#	Consultation Times
SAUT/ARUSHA/2009_1007	SAUT/IRINGA/2008_4507	3

Staff_Student_Communication4=Staff_Student_Communication ∞ Student_Z

TABLE XI. STAFF_STUDENT_COMMUNICATION4 DERIVED HORIZONTAL FRAGMENT

StaffID	Stud_Reg#	Consultation Times
SAUT /IRINGA/2001_0330	SAUT/MWANZA/2009_7890	1
SAUT /MBEYA/2010_2101	SAUT/MWANZA/2008_4588	3

Staff_Student_Communication5=Staff_Student_Communication ∞ Student_A

TABLE XII. STAFF_STUDENT_COMMUNICATION5 DERIVED HORIZONTAL FRAGMENT

StaffID	Stud_Reg#	Consultation Times
SAUT/ARUSHA/2012_1983	SAUT/MBEYA/2005_2997	4
SAUT/BAGAMOYO/2013_2088	SAUT/MBEYA/20014_9997	2
SAUT /MBEYA/2012_1983	SAUT/MBEYA/2006_4567	5

The derived horizontal fragmentation strategy above enabled the author to achieve the desired fragmentation with join characteristics. With derived fragmentation using join operations in distributed databases the author managed to retrieve the desired tuples or records according to the predicate or minterm efficiently.

E. Allocation

With data allocation, the determination of the location of the fragments based on the information of the database is done. The types of transactions to be applied to the database, the communication network, the storage capability of each site, and the design goal of cost, response time and data availability all need to be taken into account for effective data allocation [9][11].

Figure 6 illustrates five campuses of SAUT considered as sites of SAUTDDBS.

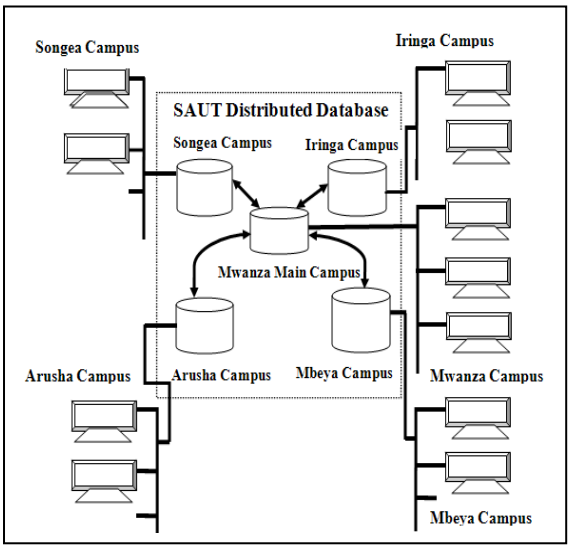


Figure 6. SAUTDDBS Sites Design

F. Replication

In a distributed database, a relation or a fragment can be replicated or copied. Copies of data may be stored redundantly in two or more sites to serve specific information requirements and enhance data availability. For example, SAUTDDBS has copies of a student relation fragments stored in the database of its campuses and also in the main campus database.

Data replication decisions need to consider the size of database and data usage frequency. If a database is fully replicated, it will then store multiple copies of each fragment at multiple sites. While in a partially replicated database, only some fragments are replicated. A database is fully redundant if each site contains a copy of the entire database [1][8].

Replication improves performance, increase the fault tolerance and introduce enhanced availability of the data into database systems while reducing the cost of accessing and transferring data. However, data replication increases the cost of updates since all replicas have to be updated to ensure they are all identical. Therefore, replication increases the complexity of the concurrency control [1][14].

In this study, replication design was done using Sybase replication server architecture. Replication Server enables sharing of information across various SAUT campuses by replicating it to and from different hardware platforms and data sources without losing the transactional integrity of the data [14].

Reliable replication system architecture must do much more than simply copy a piece of data. The system must be able to maintain the integrity of the data at the transaction level, deliver data quickly and efficiently across the network, allow distributed sites to modify data, be easy to monitor and manage (the most important, perhaps) and transfer data in any direction across heterogeneous data sources [14][15].

The use of replication server was essential since it supported replicating data to and from non Sybase data servers. Data can be replicated to non Sybase data servers such as Oracle, Informix, IBM DB2, and Microsoft SQL Server using Sybase DirectConnect gateways. Transactions can be captured and forwarded from non Sybase data servers using Sybase Replication Agents. Data can also be replicated from a non Sybase source, through Replication Server to a non Sybase destination [14][15].

The replication server helped to provide warm standby capability with a pair of databases where one was the active database and the other as the standby database, being supported by replication server's functionality. As clients update the active database, replication server copies transactions to the standby database, maintaining consistency between the two. Should the active database fail for any reason, you can switch to the standby database, making it the active database, and resume operations with little interruption. Figure 7 illustrates SAUTDDBS Replication Design.

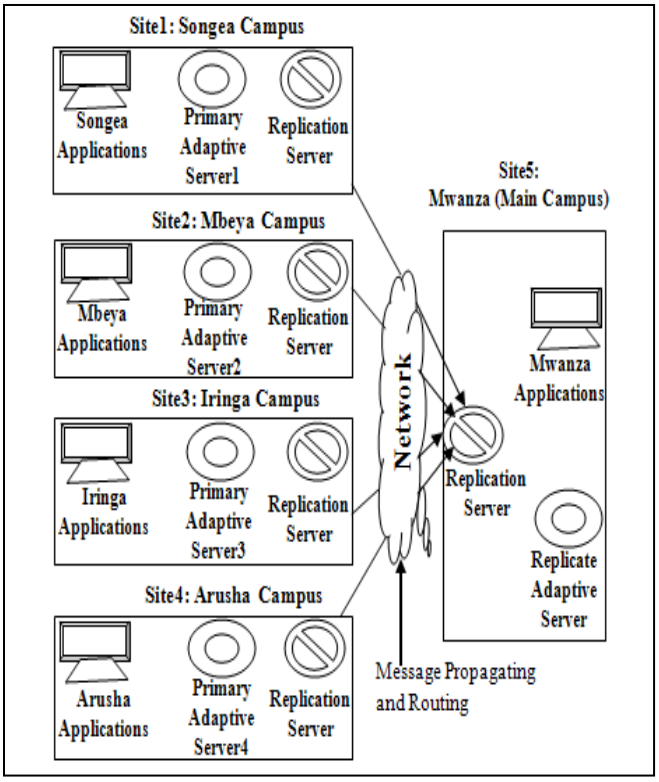


Figure 7. SAUTDDBS Replication Design

Adaptive Server: Manages databases that contain primary data, replicate data, or both.

Replication Server: Captures transactions from non Sybase data servers and sends them to a Replication Server.

TABLE XIII. SAUTDDBS AGAINST EXISTING CENTRALIZED DATABASE SYSTEM ANALYSIS

Process Enhanced or Affected	**Existing Centralized Database Systems capabilities/challenges**	**Designed Distributed Database System (SAUTDDBS) capabilities/challenges**
Distribution Design	No fragmentation needed and database design is much easier since data is centralized.	More complex since it required making proper decisions about data and programs placement across various sites of a computer network. Difficulties on how to employ homogeneous distributed database environment since some of the campuses had their databases implemented using different DBMS software. This was due to the fact that the interaction model proposed in this study could only be applied effectively with homogeneous distributed environment rather than heterogeneous distributed environment.
User involvement in the design process	University members and alumni were not effectively involved in the design of these systems.	The study has effectively involved University members (i.e. students, instructors and supporting staff). Designed by taking into account important alumni activities and processes which could enable the University to easily obtain alumni financial and academic contributions.
Reliability	Have very low data reliability since data is located, managed and maintained at one location.	Has high data reliability and good usability as data is stored at various sites (campuses), so, a disaster or system failure at one site will not cause loss of entire data.
Users' awareness, readiness and experience to use DDBSs'.	Have enough experience and readiness to use centralized database systems.	Users still very slow and not ready to use distributed database systems due to lack enough experience.
Replication Design	Not applicable, therefore problems in ensuring data availability as well as data reliability.	Enhanced data availability, parallelism as queries on a relation r can be processed by several nodes in parallel. Also reduced data transfer as a relation r is available locally at each site containing a replica of r. High costs for updates as each replica of relation r must be updated. Concurrency control more difficult since concurrent updates to distinct replicas may lead to inconsistent data.
Integrity control	Easy integrity control.	Integrity control was more difficult since communication and processing costs that are

Process Enhanced or Affected	**Existing Centralized Database Systems capabilities/challenges**	**Designed Distributed Database System (SAUTDDBS) capabilities/challenges**
		required to enforce integrity constraints are high as compared to centralized system.
Enabling system expansion	Difficulties in expanding database size.	Has been designed in such a way to easily accommodate increasing database sizes by only increasing processing and storage power to the network.
Data Access	A lot of remote data access requests due to a centralized database system existed.	Much improved data access since data can easily be accessed locally due to data replication at various SAUT campuses.
University Management Activities	Very slow and sometimes could take days due to dependence on centralized database.	Much improved since important information can be obtained easily and fast locally.

CONCLUSIONS

This study has employed an interaction model as a requirement for distributed database systems design in higher learning institutions. The model is effective due to its capability in supporting attaining both; data access and reliability capabilities when used as a requirement in distributed database systems design while providing the foundation for communication and interactive applications design between higher learning institutions and their respective alumni.

The study recommends the use of the proposed model for DDBSs' design especially with the top down design approach for distributed database systems design and implementation to augment data access, distribution and reliability in higher learning institutions with distributed campuses.

On the other hand, the author has provided the strategies used during data fragmentation where both primary and derived horizontal fragmentation techniques were applied. An analysis of the designed SAUTDDBS against existing centralized database system has as well been provided in Table XIII. The author has also provided a diagrammatic design of SAUTDDBS illustrating how various campuses of SAUT were made into sites with fragments depending on data access, distribution and reliability needs at those sites.

The study finally explains how Sybase replication server was employed during replication design as it enabled replicating data to and from non Sybase data servers for ensuring high data reliability.

REFERENCES

[1] Ozsu, M., & Valduriez, P. (2011), Principles of Distributed Database Systems, 3rd edition, Springer Science+Business Media, LLC, 233 Spring Street, New York, NY 10013, USA.

[2] Slonim, J., Schmidt, & Fisher, P. (1979), "Considerations for Determining the Degree of Centralization or Decentralization in the Computing Environment", North-Holland Publishing Company, Information & Management, 2 (1979), 15-29, USA.

[3] Ceri, S., & Pelagatti, G. (1984), Distributed Databases Principles and System, McGraw Hill, New York, USA.

[4] Elmasri, R. & Navathe. S. (2006), Fundamentals of Database Systems, 5th edition, Menlo Park, CA: Benjamin Cummings.

[5] Ma, H. (2007), "Distribution Design for Complex Value Databases", dissertation presented in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Information Systems at Massey University, New Zealand.

[6] Kung, H., Kung, L., & Gardiner, A. (2012), "Comparing Top down with Bottom up Approaches: Teaching Data Modeling", paper presented at the Proceedings of the Information Systems Educators Conference, New Orleans Louisiana, USA.

[7] Iacob, N. (2011), The use of distributed databases in e-learning systems, paper presented at the Proceedings of 3rd World Conference on Educational Sciences-WCES2011, Bahcesehir University, Istanbul, Turkey, 03-07 February 2011, 2673-2677.

[8] Huang, F. & Chen, J. (2001), "Fragment Allocation in Distributed Database Design", Journal of Information Science and Engineering, 17(1), 491-506.

[9] Daudpota, H. (1998) "Five Steps to Construct a Model of Data Allocation for Distributed Database Systems", Journal of Intelligent Information Systems, 11 (1), 153-168.

[10] Mwombeki, A., & Kimaro, H. (2013), "Adoption of Web Based Applications to Enhance Communication and Interaction Between Higher Learning Institutions and Alumni", International Journal of Computer and Information Technology, 2 (6).

[11] Marwa, F., Ali, I. & Hesham, A. (2008), "A heuristic approach for horizontal fragmentation and alllocation in DOODB," paper presented at the Proceedings of INFOS2008, 9-16.

[12] Cheng, C., Lee, W., & Wong, K. (2002), "A genetic algorithm-based clustering approach for database partitioning," IEEE Transactions on Systems, Man, and Cybernetics, 32 (3), 215–230.

[13] Baião, F., Mattoso, M., & Zaverucha, G. (2000), "Horizontal Fragmentation in Object DBMS: New Issues and Performance Evaluation", paper presented at the Proceedings of the "19th IEEE International Performance, Computing and Communications Conference" (IPCCC 2000), IEEE CS Press, Phoenix, Feb 2000, 108-114.

[14] Wiessman, M., & Pedone, F. (2000), Understanding replication in databases and distributed systems, paper presented at the Proceedings of the 20th IEEE International Conference on Distributed Computing Systems, Taipei, Taiwan, April 2000.

[15] Tanenbaum, S., & Steen, V. (2007), Distributed Systems, Pearson Education, Inc., Upper Saddle River, New Jersey, USA.