

## Modeling of Mixed Fragmentation in Distributed Database Using UML 2.0

S. Jagannatha<sup>1</sup>, M.Mrunalini<sup>1</sup>, T V Suresh Kumar<sup>1</sup>, K Rajani Kanth<sup>1</sup>

<sup>1</sup>M S Ramaiah Institute of Technology, Bangalore 560 054, India

jagannatha@msrit.edu, mrunalini@msrit.edu, tvsureshkumar@msrit.edu, rajanikanth@msrit.edu

**Abstract:** In order to reduce the irrelevant data accessed by the applications and data exchange among sites, we are modeling the distributed database fragmentation by using UML 2.0. Object Oriented Database Management System (OODBMS) design level fragmentation helps to improve the performance of applications. We propose a model for mixed fragmentation using UML2.0. The UML model we presented uses the advantages of being guided by design process. We discuss the model with a case study.

**Keywords:** Distributed database, Fragmentation, Unified modeling language, Performance Engineering, Model Driven Architecture (MDA).

### 1. Introduction

A distributed database system (DDS) typically consists of a number of distinct yet interrelated databases (fragments) located at different geographic sites which can communicate through a network. Typically, such a system is managed by a distributed database management system (DDBMS). A site also participates in the execution of global transactions involving databases at two or more remote sites [4]. Designing DDS are complex task because several interacting design decisions are involved. The design involves making decisions on divide the fragment and the placement of data across the sites of a computer network [5]. In a top down approach, the distributed design has two phases: fragmentation and allocation. The fragmentation phase is the process of clustering in fragments information based on the application requirement and these fragments are accessed simultaneously by applications, and the allocation phase is the process of distributing the generated fragments over the database system sites in order to reduce the communication cost. The combination of these fragments yields the original database without any loss of information.

To fragment a class, it is possible to use two basic techniques: vertical fragmentation and horizontal fragmentation. In an object oriented (OO) environment, horizontal fragmentation distributes class instances across the fragments, which will have exactly the same structure but different contents. Thus, a horizontal fragment of a class contains a subset of the whole class extension. Vertical fragmentation breaks the logical structure of the class and distributes them across the fragments, which will logically contain the same objects, but with different properties. It is also possible to perform mixed fragmentation on a class, combining these two techniques. The benefits of mixed fragmentation and hybrid fragmentation to increase the performance of applications [10].

The object oriented design methods usually used in the industry such as OMT [3], OOA/OOD [4], OOD [5] do not integrate the distribution of OODB. To make distributed design task easier it is important to support with a common modeling language such as UML. The remaining part of this paper is organized as follows.

---

Corresponding author: S. Jagannatha. Tel.: 91-80- 23600822; fax: 91-80-23603124  
E-mail address: jagannatha@msrit.edu

In Section 2 we proposed to study the related work. The modelling of mixed fragmentation using UML 2.0 is proposed in the section 3. We discuss a case study in section 4. Finally a conclusion and future work is given in Section 5.

## 2. Related work

Various researchers made significant contribution to distributed object oriented databases. The benefits of mixed fragmentation and hybrid fragmentation to increase the performance of applications have been discussed [6]. Appropriate fragment allocation in Distributed Database Systems is an important issue in design of Distributed Database Management System [7]. A framework to handle the class fragmentation problem during the design of distributed object databases are discussed in [14]. On a Wide Area Network (WAN), fragment allocation is a major issue in distributed database design, two heuristic algorithms are developed to find a near-optimal allocation such that the total communication cost is minimized as much as possible addressed in [8].

In [9] author describes one of the first attempts at successfully combining fragmentation, allocation, and replication into a single step of distribution and applying the combination to a practical problem with positive results. Shamalakar B. Navathe et al proposed to define mixed fragmentation as a process of simultaneously applying the horizontal and vertical fragmentation on a relation. [14]. The Fernanda BaiSio et al illustrates [13] Horizontal fragmentation may improve the performance of database systems. Software Performance Engineering (SPE) has evolved over the years and has been demonstrated to be effective during the development of many large systems [12], [13]. The extensions to SPE process and its associated models for assessing distributed object-systems are discussed in [13]. Predictive performance modeling environment that enables performance measures for distributed object-oriented systems is described in [15].

## 3. The Modeling of Mixed Fragmentation using UML 2.0

### 3.1 Mixed Fragmentation Methodology

This procedure involves all activities prior to allocation. The steps of the mixed partitioning methodology are given below.

1. Specification of inputs: In this step, the following inputs are specified. System requirements: System and network information, Transaction information: name, frequency, attribute usage, predicate usage etc. The attribute usage matrix is a matrix containing transaction as rows and attributes as columns. Distribution constraints: any predetermined partitions or fixed allocation of data. System information: number of sites, transaction costs etc. This information is used particularly to solve the allocation problem.
2. Vertical partitioning of grid: In this step all candidate vertical fragments are determined
3. Horizontal partitioning of grid: In this step all candidate horizontal fragments are determined
4. Populating the system catalogue with the representation of grid cells: A scheme for representing grid cells and mixed fragments is developed and stored in the system catalog.
5. Mixed fragment generation: The number of disk accesses required execute a transaction will be used to compute the optimal set of mixed fragments so as to minimize the total number of disk accesses required to process the transaction

The Distributed Database Designer (DDBD) gather system information such as network information number of sites, transaction costs etc, schema information such as relation, attributes, cardinalities, attribute sizes, predicates used by the database operations, etc, and he gather the transaction information such as name, frequency, attribute usage, predicate usage etc, and he gathers the Distribution constraints: any predetermined partitions or fixed allocation of data. Based on the above inputs DDBD performs vertical fragments and horizontal fragments and finally based on the number of disk accesses required execute a transaction will be used to compute the optimal set of mixed.

Deciding the most adequate fragmentation technique for a given use case and also to decide on the most adequate fragmentation technique for each use case in the database, we need to identify the fragmentation

technique by using the probability distribution function. The Execution frequency (Ef) of a transaction is estimated using the formula given below.

$$Ef = (a + 4m + b) / 6 \tag{1}$$

Where a = is Optimistic time, b = Pessimistic time, m= Most likely time

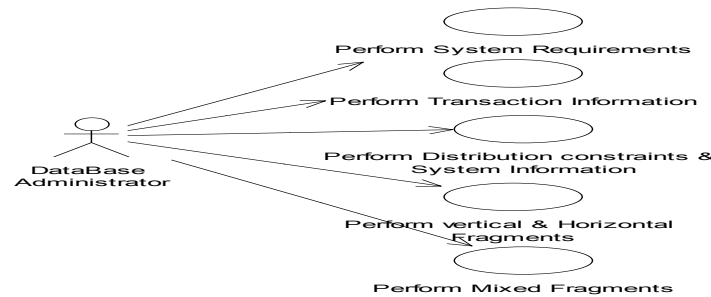


Fig 1. Use Case Diagram for the Mixed Fragmentation in Distributed Database

Thus priority is most adequate fragmentation technique for each use case given to the most frequent transactions, and the transactions are involved in use cases on those transactions are indicated for horizontal and/or vertical fragmentation according to the defined heuristics. Those use cases on the intersection of the horizontal and vertical sets might proceed to mixed fragmentation.

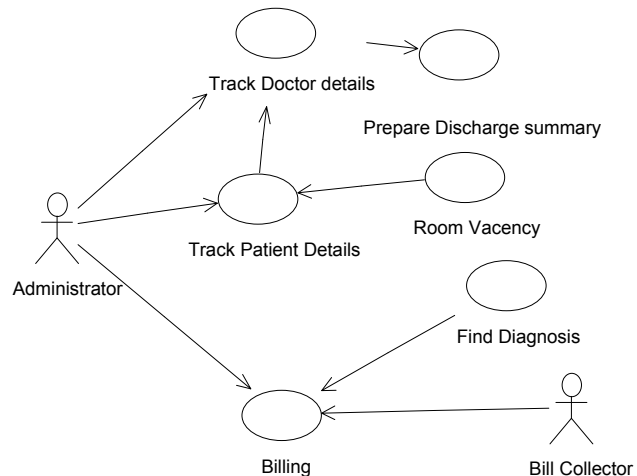
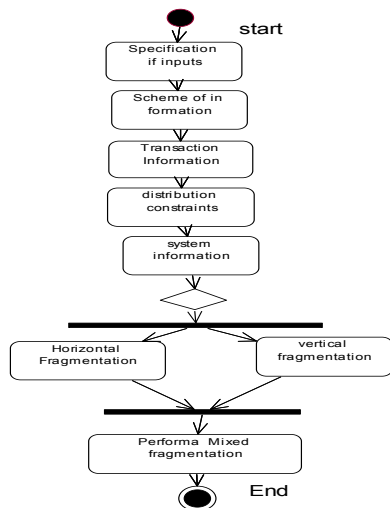


Fig 2: Activity diagram for the Mixed fragmentation. Fig 3 Use Case diagram for the Hospital Management System

In figure 2 indicate the activity diagram for the mixed fragmentation. The set activates are System requirements: System and network information Transaction information: Distribution constraints: any predetermined partitions or fixed allocation of data etc specifies the input for the fragmentation. The DDBD has to decide the vertical and horizontal fragments and finally performs the mixed fragments.

#### 4. Validity of our Algorithm Using Case Study

We have considered a case study on Patient Monitoring System (PMS) given [17]. The Use case diagram of the given PMS is shown in figure 3. In this diagram represents different functionality such as billing, patient details, doctor details prepare discharge summary, availability of rooms, and find the diagnosis details etc. The administrator is involved in the keep track of the details of patient, Doctor schedule, Staff. The billing department prepares the bills based on information such as diagnosis, pharmacy, Bed charges, Doctor and Nurse Charges etc. The accounts section collets all the bill amount form the patient. Patient details: Patient no, name, address ,phone no, sex, date of joining, history, In/out patient,get patientdetails(), putPatient(), Billing: Bill no, Bill date, Patient details, Diagnosis details, Room details and doctor Details,get details(), generate report(), Doctor Details: Doctor no, Name, Address, Phone no, Specialization, Get details() , Find doctor(),Room Details: Room no, Level no, Type, No of rooms get details(), Check

availability() ,Discharge summary: get patient detail(),get diagnosis details(), get pharmacy detail().get room details()

**Vertical Fragmentation of Schema:** All the schemas are fragmented vertically based on the mechanism specified in [11] [18]. The following are the attributes of PATIENT entity and the methods which access PATIENT entity. A1: PId A2: Pname A3: Med# A4: DOJ A5: DOD A6: History M1: Diagnose(), M2: Dispense() M3: Calculate Bill(), M4: Display() The table 1. Shows the attribute usage matrix (AUM) of Patient entity.

Method/ Att.	A1	A2	A3	A4	A5	A6	Access Freq.
M1	1	0	1	0	0	1	40
M2	0	1	1	0	0	0	60
M3	1	1	0	1	1	0	40
M4	1	1	0	0	0	1	30

Table 1: Attribute usage matrix

Att./Att.	A1	A2	A3	A4	A5	A6
A1	110	70	40	40	40	70
A2	70	130	60	40	40	30
A3	40	60	100	0	0	40
A4	40	40	0	40	40	0
A5	40	40	0	40	40	0
A6	70	30	40	0	0	70

Table 2: Attribute affinity matrix

The value 1 and 0 respectively indicates if an attribute is referenced or not by a method. For example the method M1, this permits DOCTOR to diagnose the PATIENT, accesses A1, A3 and A6. Access frequency is the number of times the method is called. From every AUM, an attribute affinity matrix (AAM) is build, which shows the coupling between two attributes of a class or entity depending on how they are accessed by the methods. This shows how the attributes are linked, whether they can be grouped together to fragment or not. The table 2. is the AAM of the PATIENT entity. From the AAM clustered affinity matrix (CAM) can be generated, which shows a semi-block diagonal form.

Att./Att.	A6	A3	A2	A1	A4	A5
A6	70	40	30	70	0	0
A3	40	100	60	40	0	0
A2	30	60	130	70	40	40
A1	70	40	70	110	40	40
A4	0	0	40	40	40	40
A5	0	0	40	40	40	40

Table 3: Clustered affinity matrix

Att./Att.	A5	A2	A1	A3	A4
A5	100	50	100	0	0
A2	50	130	130	0	0
A1	100	130	220	40	40
A3	0	0	40	40	40
A4	0	0	40	40	40

Table 4: Clustered affinity matrix of patient class

The attributes can be fragmented into two clusters (Table 3) as C1 with A1, A2, A4, A5 and C2 with A1, A2, A3, and A6 .The table 4 is the CAM of Patient class .The attributes can be vertically fragmented into two clusters as C1 with (A1, A2, A5) and C2 with (A1, A3, A4).

**Horizontal Fragmentation of Schema:** For the horizontal fragmentation we propose the predicate usage matrix. We obtain a set of non overlapping sub blocks containing a set of selection predicates. We outline the horizontal partitioning methodology by using a hospital management system. The inputs are the set of transaction and a corresponding set of predicates as follows

T1: Prepare discharge summary: PID= 10 (P1), DID=20 (P2) T2:Registration :WARDNO = 200 (P3),

T3: Billing : P1,P3 T4:Diagnosis: EQUIPID=27(P4), ROOMNO=15(P5) T5:Dispense: Dispense No=98,P2,P1 P3 T6:Display:P1, BillNo=32(P6) 12/12/2008 <BillDate<1/12/2008 (P7)

Transaction	Predicates							Acc Freq
	P1	P2	P3	P4	P5	P6	P7	
T1	1	1	0	0	0	0	0	40
T2	0	0	1	0	0	0	0	50
T3	1	0	1	0	0	0	0	60
T4	0	0	0	1	1	0	0	70
T5	1	1	1	0	0	0	0	80
T6	1	0	0	0	0	1	1	30

Table 5: Predicate Usage Matrix

Predicates	Predicates						
	P1	P2	P3	P4	P5	P6	P7
P1		120	140	0	0	30	0
P2	120		80	0	0	0	0
P3	140	80		0	0	0	0
P4	0	0	0		70	0	0
P5	0	0	0	70		0	0
P6	30	0	0	0	0		30
P7	30	0	0	0	0	30	

Table 6: Predicate affinity matrix

The table 5 shows the predicate usage matrix which represents the use of predicates in important transaction. Columns correspond to known predicate used by transaction. Each row refers to one transaction the "1" entry in a column indicates that the transaction uses the corresponding predicates. The Predicate affinity is generated in a manner similar to attribute affinity. The table 6 shows the predicate affinity matrix. The numerical value of the (i, j) element in this matrix gives the combined frequency of all transaction accessing both predicates i and j from the Predicate affinity matrix a set of cluster can be generated, which

shows the set predicates can grouped, using these groups user can perform horizontal fragmentation. The predicates can be horizontally fragmented into four clusters as H1 with P1, P2, P3, and H2 with P6, P7, and H3 with P4, P5. In the hospital management system the grids which are intersect are H1 and C1 can perform mixed fragments. For example preparing discharge summary these two clusters can be mixed fragmented.

## 5. Conclusion and Future Work

We proposed an UML model, which helps in representing fragmentation of Distributed Databases. The more accuracy about the frequency of transaction gives more clarity for mixed fragmentation. In this paper we attempted to present an UML2.0 based on modeling for Distributed Databases Design fragmentation. We illustrated the validity of the model using a case study by using the concept of attribute usage matrix, predicate usage matrix. Based on these it is easy to determine the set of use cases that can be mixed fragmented. We are proposing to simulate mixed fragmentation in the distributed database design and assess the performance.

## 6. References

- [1] Connie U. Smith, Performance Engineering of Software Systems, Reading, MA, Addison-Wesley, 1990.
- [2] C. I Ezeife, Ken Berker "Distributed object oriented Design" Vertical class fragmentation in a DOBS Mar 30 1998
- [3] C. I Ezeife, Ken Berker "Distributed object oriented Design" Vertical class fragmentation in a DOBS Mar 30 1998
- [4] S. Ceri and G. Pelagatti, Distributed Databases: Principles and Systems. New York: McGraw-Hill, 1984
- [5] F. Ravat, G. Zurfluh, "Issues in the fragmentation of object oriented databases" Basque International Workshop on Information Technology, San Sebastian, Spain, July 1995
- [6] F. Baião and M. Mattoso, "A Mixed Fragmentation Algorithm for Distributed Object Oriented Databases". In Special Issue of the Journal of Computing and Information (JCI), vol. 3(1), ICCI 98, March 2000, ISSN 1201-8511, pp. 141- 148.
- [7] Reza Basseda "Fragment Allocation in Distributed Database Systems" "Database Research Group". R.Bassed@ece.ut.ac.ir
- [8] j yin-fu huang and jyh-her chen "Fragment Allocation in Distributed Database Design" Institute of Electronics and Information Engineering, National Yunlin University of Science and Technology
- [9] Ajit M. Tamhankar and Sudha Ram, Member, IEEE " Database Fragmentation and Allocation: An Integrated methodology and Case Study" 288 IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, VOL. 28, NO. 3, MAY 1998
- [10] Fernando Baião, Marta Mattoso "A Mixed Fragmentation Algorithm for Distributed Object Oriented Databases" Proceedings of the Ninth International Conference on Computing, Information, Winnipeg, Canada, June 1998
- [11] Shamkant B. Navathe et al "A Mixed Fragmentation Methodology for Initial Distributed Database Design" University of Florida, Gainesville FL32611
- [12] Connie U. Smith and Lloyd G. Williams, Performance Solutions, 2000
- [13] Connie U. Smith and Lloyd G. Williams, "Performance Engineering Models of CORBA-based distributed-object systems", Performance Engineering Services and Software Engineering Research, 1998
- [14] Fernanda BaiSio, Marta Mattoso, Gerson Zaverucha "Horizontal Fragmentation in Object DBMS: New Issues and Performance Evaluation" 0-7803-5979-810\$01 0.00 0 2000 IEEE
- [15] Susan D. Urban and Suzanne W. Dietrich Department of Computer Science and Engineering Arizona State University Tempe, AZ 85287-5406 " Using UML Class Diagrams for a Comparative Analysis of Relational, Object-Oriented, and Object-Relational Database Mappings SIGCSE'03, February 19-23, 2003, Reno, Nevada, USA. Copyright 2003, ACM 1-58113-648 -X/03/00
- [16] Jau-Ji Shena and Tzung-Liang Hungb." Minimizing Transmission Cost for Distributed Database Design" Proceedings of the 17th International Conference on Advanced Information Networking and Applications (AINA'03) 0-7695-1906-7/03 \$17.00 © 2003 IEEE
- [17] M Mrunalini T V Suresh Kumar K Rajani Kanth "Consistency Checking of Fragmentation before and after Database Integration" International Conference on Recent Applications of Soft Computing in Engineering & Technology", in Technical Sponsorship of IEEE Delhi Section, 22-23 December 2007.