Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 11, November 2014, pg.605 – 611

RESEARCH ARTICLE

OO-Database to UML Conversion with Measuring Response Time

Mr. Malhar K.Kurhadkar[1], Mr. G.R.Gosavi[2]

[1]Department of Information Technology, Prof. Ram Meghe College of Engineering, Amravati
[2]Department of Computer Science & Engineering, Dr.Panjabrao Deshmukh Polytechnic, Amravati
[1] malhar.kurhadkar@gmail.com, [2] dinesh.godsavi@rediffmail.com

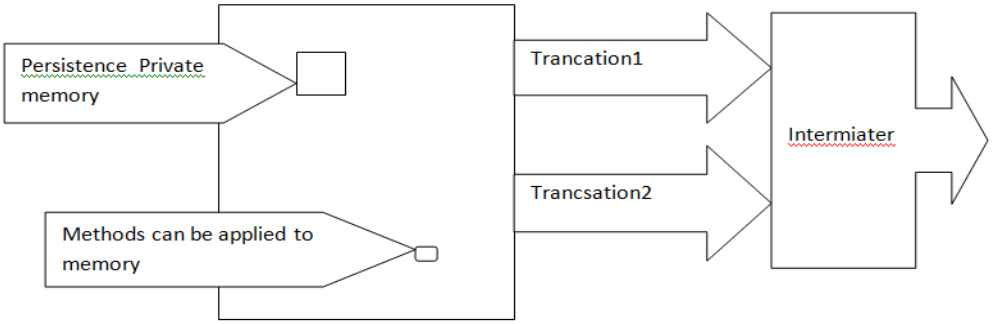
Abstract—Object oriented database models are largely used object in today's word as you know this object are used for required data are stored in this object because this is integrated programming in this program their rare mostly used the various programming language these are purpose used object database .This research we use modeling language this the graphical representation of the using different representation given for different sceneries such as class diagrams and other things.

In this research paper we developed the system for the conversion of object oriented data models in the class diagrams with the help of case study it very beneficial to the developed because the object database mode consist object and also the unified modeling represent object in the different formats.

Keywords: Dynamic Binding, Inheritance, oo database model concepts, unified modeling with use case, interactive view

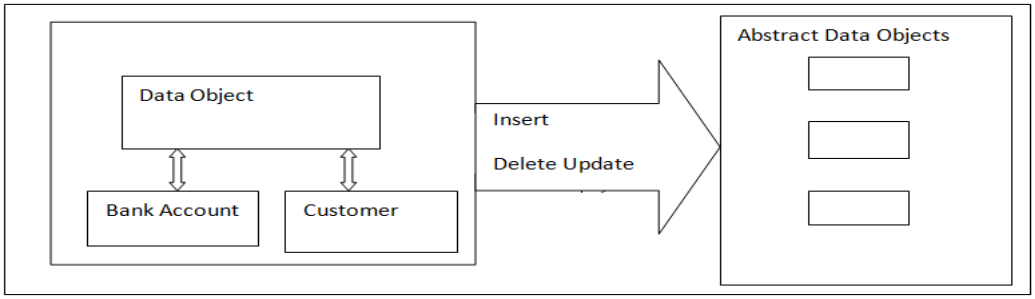
INTRODUCTION

Object-oriented systems are .very powerful in today world .Its making the impact on the applications which developed in software. The idea was taken from procedure driven programming now it examples are Simula, Smalltalk and C++. It has since been adopted and extended, particularly to cover the broader range of software engineering activities including modeling, specifications and design phases of software construction. Even the field of artificial intelligence, especially knowledge engineering, has (somewhat independently and in parallel with development in programming) found the object-oriented approach to be particularly effective. Following is the basic diagrams for these object oriented modeling which object stored the database for retrieving or processing



Following are the concept of object oriented with the help of Bank management system in which bank management system use the database for the different whole bank in which the data objects are created for the each and every entities each and every entity having different attributes .Persistent memory having the abstract database objects the different operation store different truncations and intermediate provide operations it is interface user interface and different concept one by one

Classes: The persistent data of tables in the relational model is replaced by a collection of persistent ADOs, and the generalized data manipulation procedures are replaced by the public interfaces of objects.



Dynamic Binding & user interface

We have seen that data classes define the meta-structure of an object-oriented database as well as the behavior of ADOs that instantiate them. Different applications will typically call for different data classes to be defined .A standard user interface of generalized operations that is application independent. Instead, the user interface for an application must be defined in the object-oriented database schema itself.

Dynamic Binding

The name of a procedure statically defines the code to be executed, ie. it is possible to determine at compile time the code to be activated for any procedure call. Suppose for example, the following procedure was defined: Object-Oriented Data Model

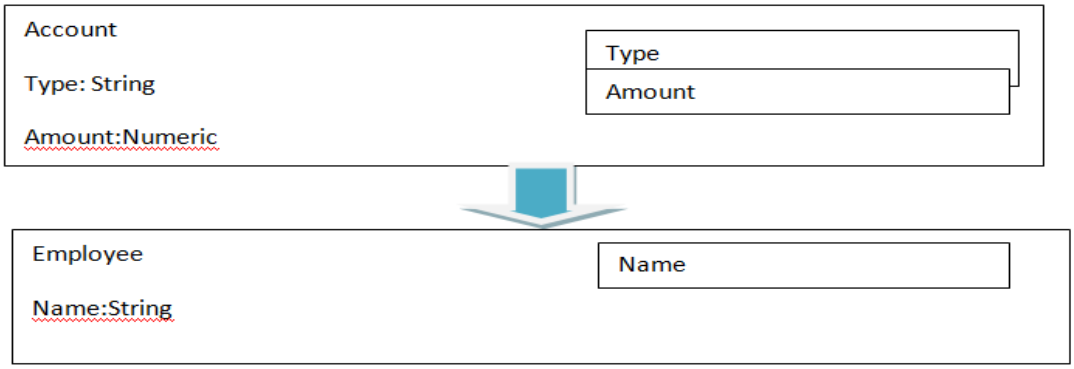
Print_balance(accountno: String, amount: Numeric) {… code–for–Print_Person …}

The procedure name 'Print_balance' will be statically bound to {… code–for–Print_Person …}.

Dynamic & Static Inheritance

A user-defined class will typically use system-provided classes (and other defined classes) to structure its private memory and define its methods. Memory is structured as a set of variable-domain bindings,

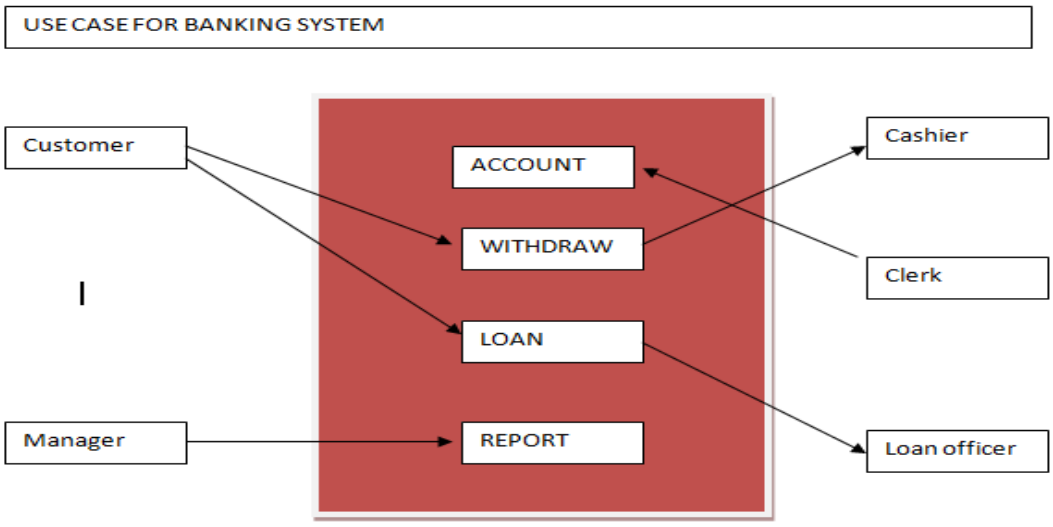
allowing essentially client-server associations between Data objects. In object-oriented systems, messages play the role of procedure calls and method selectors are analagous to procedure names. In contrast, however, it is not possible to determine statically the code to be executed for a given message. This is because the instance to which a message is directed can only be determined dynamically (at run-time) and the code associated with the specified selector is likely to be different for objects from different classes



UML(Unified Modeling Language)

IT professional use unifying language as for developing computer application. It is standard programming in dependant The similar Phenomena spawn Linux,J2EE,It works on diagrams ,there are different graphical representation for different entities The most useful, standard UML diagrams are: use case diagram, class diagram, sequence diagram, state chart diagram, activity diagram, component diagram, and deployment diagram. Let explain all These with the help of different examples

Use Case: The relationships among different use cases. Use-case diagrams generally show groups of use cases -- either all. main purpose of the use-case diagram is to help development teams visualize the functional requirements of a system, Following figure illustrated the Use Case



The diagrams shows use case of customer who are looking for withdrawal of money also manager shows the annual report the truncation will be done cashier and also those customer having loan that

case will section by loan officer these complete diagrams shows the overall structure of application which also helps to developed the database object using different data models

Class diagrams

The class diagram shows how the different entities (people, things, and data) relate to each other; in other words, it shows the static structures of the system. A class diagram can be used to display logical classes, which are typically the kinds of things the business people in an organization talk about – Loan,account,customer,banks place,loans, and interest rates. Class diagrams can also be used to show Implementation classes, which are the things that programmers typically deal with. An implementation class diagram will probably show some of the same classes as the logical classes diagram. The implementation class diagram won't be drawn with the same attributes, however, because it will most likely have references to things like Vectors and Hash Maps.

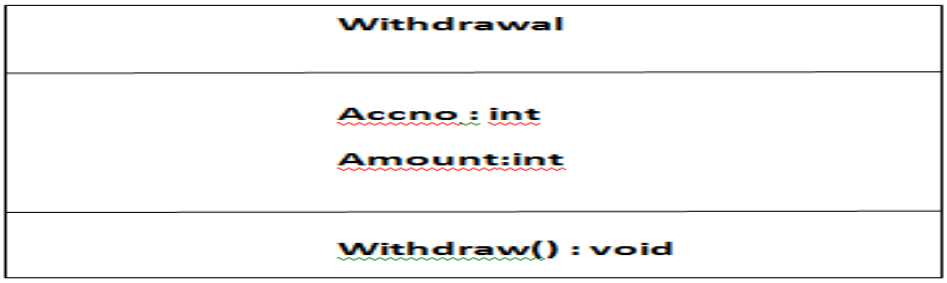


Figure: Sample diagram for the class diagram for Withdrawal

Every developer know the class diagram how to create with the help of use caser also it provide the inheritance concept e.g. when any one wants to open account there are two type have account 1.saving 2.current both having some similar attribute which comes from account i.e. account no,& amount etc. There number of types for diagrammatic model such as

Sequence diagram:

A sequence diagram has two dimensions: The vertical dimension shows the sequence of messages/calls in the time order that they occur; the horizontal dimension shows the object instances to which the messages are sent. State chart Diagram it shows the flow of truncation how the flow of the operations happened state to state It can be argued that every class has a state, but that every class shouldn't have a state chart diagram. Only classes with "interesting" states -- that is, classes with three or more potential states during system activity

Activity diagram

It is the procedural flow of diagram it actually shows the high level business diagrams to show the operation in the business process

Deployment diagram

It is the actually deployment of the representation or its called physical implementation. We are developing the bank model now we knows the OO database models consist Different objects

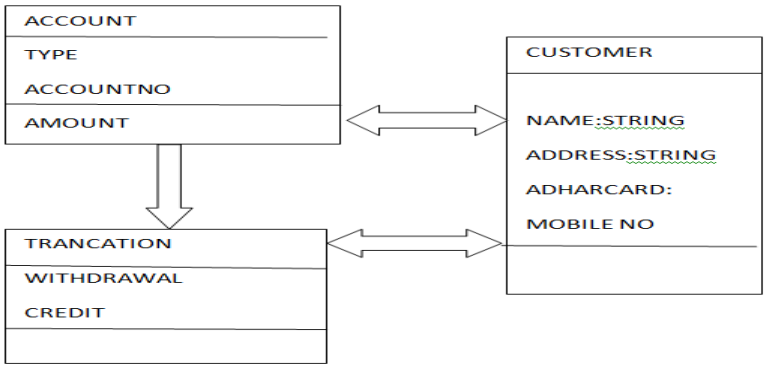
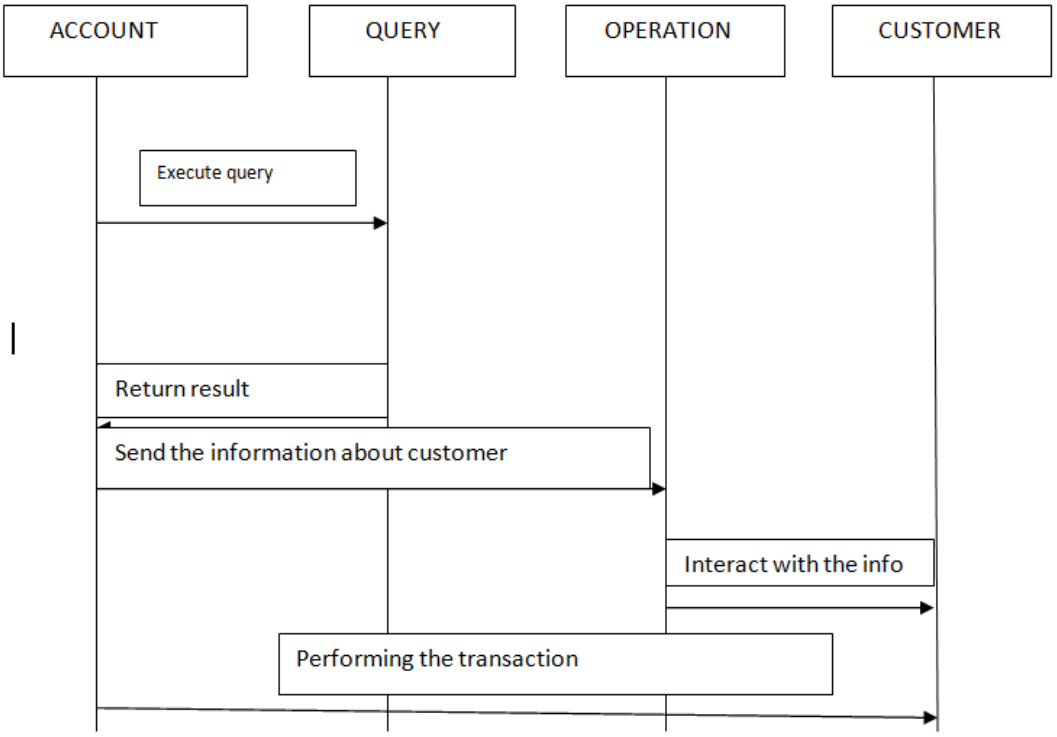


Figure For the class diagrams for entities

A sample of object oriented database this the Sequence diagram for banking system



Analytical Study of OO-Database object to the UML Model are

Query Response Time calculation

In object database query process the query processor is a procedure that selects the suitable strategy which executes and generates the result. Query processor select the appropriate query for produce the result at minimum time. The query response time is a time in which the query processor executes and

generates a result in the response of requested data-base by the user. The query response time (QRT) is the part of the query processing strategy (QPS)

For calculating the response time the algorithm given below as

```
DECLARE@EndTime DATETime
DECLARE@StartTime DATETIME
DECLARE@I INT
SET@I= 0
SET@StartTime = GETDATE()
SET@EndTime = GETDATE()
While(@I<10)
Begin
PRINT 'StartTime = ' + CONVERT(VARCHAR(30), @StartTime,121)
Select Cust_Name,Date_of_Commt,Prem_Due_Date from tblCustomers
SET @I= @I+1
PRINT' EndTime = ' + CONVERT (VARCHAR(30), @EndTime,121)
PRINT 'RESPONSETIME = '+ CONVERT (VAR CHAR (30),@ENDTIME-@STARTTIME,114)+"
End
```

y the use of above code, a query response time is compute on the basis of five runs. It is computed in millisecond and on the basis of five runs, average response time is evaluated. In query, lines of codes are increase from 10 to 10^5. As depicted in the graph, it is increasing as the lines of code increasing,

Conclusion

In this paper we show UML is the powerful Programming language and the language helps to converting the object oriented data model to UML model it help to developed application and database also it calculate the execution speed with the help of total response time. query response time is computed after designing of the UML class and the steps of executions in the form of sequence diagram. This conversion of object oriented database model is very powerful tool for the converting into UML model which helps to develop the readymade programs It very useful for developing the large application.

REFERENCES

1. *M. Blaha and J. Rumbaugh, "Object Oriented Modeling," 2nd Edition, Prentice Hall, Upper Saddle River, 2005.*
2. *G. Booch, J. Rumbaugh and I. Jacobson, "The Unified Modeling Language User Guide," 12th Indian Reprint, Pearson Education, Upper Saddle River, 2004.*
3. *G. Booch, J. Rumbaugh and I. Jacobson, "The Unified Modeling Language User Guide," Addison Wesley, Reading, 1999.*
4. *G. Booch, "Object-Oriented Analysis and Design with Applications," 2nd Edition, Addison Wesley, Reading, 1994.*
5. *S. Cranefiel and M. Purvis ,"UML as an Ontology Modeling Language," Proceedings of the Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence, Barcelona, 16-22 July 1999.*
6. *U. Dayal, "Of Nests and Trees: A Unified Approach to Processing Queries That Contain the Nested Sub Queries, Aggregates and Quantifiers," Proceedings of 13th VLDB Conference, Brighton, 1-4 September 1987, pp. 197-208.*

7. *A. Deshpande, S. Nath, B. P. Gibbons and S. Seshan, "Cache-and-Query for Wide Area Sensor Databases," SIGMOD'03, San Diego, 8 June 2003, pp. 503-514.*
8. *R. Ganski and H. K. T. Wong, "Optimization of Nested SQL Queries Revisited," Proceedings of SIGMOD Conference, Vol. 16, No. 3, 1987, pp. 23-33. doi:10.1145/38714.38723*
9. *H. Gomaa, "Designing Concurrent, Distributed, and RealTime Applications with UML," Proceedings of the 23rd International Conference on Software Engineering, Toronto, 12-19 May 2001, p. 829.*
10. *E. Holz, "Application of UML within the Scope of New Telecommunication Architectures," GROOM Work-shop on UML, Physicaverlag, Mannheim, 1997.*
11. *S. Huaiming, W. Yang, A. Mingyuan, W. Weiping and S. Ninghui, "Query Prediction in Large Scale Data Intensive Event Stream Analysis Systems," 7th International Conference on Grid and Cooperative Computing, Shenzhen, 24-26 October 2008.*
12. *OMG, "UML Profile for Schedulability, Performance and Time," OMG Document Ptc/03-02-03, Needham, 2002.*
13. *OMG, "Unified Modeling Language (UML)—Version 1.5," OMG Document Formal/2003-03-01, Needham, 2003.*
14. *OMG, "Unified Modeling Language Specification," 2011. http://www.omg.org*
15. *S. Pllana and T. Fahringer, "UML Based Modeling of Performance Oriented Parallel and Distributed Applications," Winter Simulation Conference USA, Vol. 1, 2002, pp. 497-505.*
16. *V. Saxena, D. Arora and S. Ahmad, "Object Oriented Distributed Architecture System through UML," IEEE International Conference on Advanced in Computer Vision and Information Technology, Aurangabad, 28-30 November 2007, pp. 305-310.*
17. *W. Liang, B. Chen and J. X. Yu, "Response Time Constrained Top-k Query Evaluation in Sensor Networks," 14th IEEE International Conference on Parallel and Distributed Systems, Melbourne, 8-10 December 2008, pp. 575-582. doi:10.1109/ICPADS.2008.65*
18. *S. D. Muruganathan, A. B. Sesay and W. A. Krzymien, "Analytical Query Response Time Evaluation for a Two-Level Clustering Hierarchy Based Wireless Sensor Network Routing Protocol," Communications Letters, IEEE, Vol. 14, No. 5, 2010, pp. 486-488.doi:10.1109/LCOMM.2010.05.091473*
19. *G. C. Wei and L. T. Ming, "A P2P Object-Oriented Database System That Supports Multi-Attribute and Range Queries with Improved Query Response Time," Information*
20. *C. A. Yfoulis, A. Gounaris and D. Tzolas, "Minimization of the Response Time in Parallel Database Queries: An Adaptive Cost-Aware MPC-Based Solution," 19th Mediterranean Conference on Control & Automation (MED), Corfu, 20-23 June 2011, pp. 813-818.*
21. *N. V. Murray and E. Rosenthal, "Linear Response Time for Implicate and Implicant Queries," Knowledge and Information Systems, Vol. 22, No. 3, 2010, pp. 287-317.*
22. *Object Oriented Query Response Time for UML Models Journal of Software engineering and Applications Vol. 5 No. 7 (2012) , Article ID: 19736 , 6 esDOI:10.4236/jsea.2012.57059*

