Nursing Informatics
U. Gerdin et al. (Eds.)
IOS Press 1997

Database Issues in Object-Oriented Clinical Information Systems Design

S.C. Chu[a] and J.B. Thom[b]
[a]*PhD candidate, Centre of Medical Informatics, Faculty of Medicine, Monash University, Australia,* [b]*Research Fellow, Department of Nursing, Victoria University of Technology, Australia*

A clinical information system (CIS) prototype was created from an Object-Oriented (OO) design. We experienced considerable difficulties when implementing the OO data model in a relational database management system (RDBMS), including lack of semantic power and support for complex objects, inability to encapsulate object methods, and performance degradation due to extensive join operations. This paper reflects on the experiences of a CIS research project and explores issues related to the use of RDBMS and Object-Oriented Database Management Systems (OODBMS) in CIS design and development.

Introduction

There are a number of characteristic features that clearly differentiate Clinical Information Systems from many other automated applications. They include:

Structure of clinical data are highly complex, heterogenic[1], and time-oriented. Relationships among the data are also complex;

Information about clinical care is highly dynamic, becomes available incrementally, is subject to change and requires flexibility in the database schema;

It is necessary to support numerous different user (clinician) views, organisation schemes, and user interfaces[2], hence the system is polymorphic; and

The costs in developing complex application for a vertical market such as healthcare has traditionally been high. Thus creating considerable pressure to create re-useable codes that meet the needs of different hospitals.

As OO techniques can more effectively and accurately model the complexity of the real world[3], this technique is well suited for designing and developing clinical information systems applications. Many clinical applications or electronic medical records research and development efforts using object-oriented technology have been reported[1,3-8].

This paper discusses the database implementation difficulties we have encountered and explores some of the database design issues for Clinical Information Systems.

The object-oriented approach to systems development has been evolving from the concepts of data abstraction, information hiding, inheritance and polymorphism. This technique creates a model of direct representation of the real world's distinct object classes[9]. With OO approach, a BP_Object can be defined which captures all these attributes and the required functions as illustrated in *Figure 1.*

Object-Oriented Design

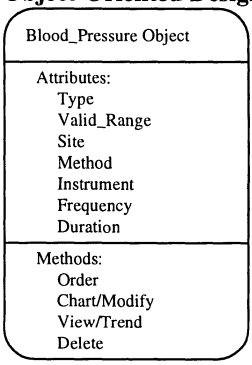

Figure 1: The BP_Object

Inheritance is the ability to construct the object subclass hierarchy[10-11] (or more strictly, a lattice) by defining subclasses of objects which inherit attributes and functions from their parent object classes in the system. This feature allows program codes created to provide functionality of one object class to be re-used by all its subclasses. In our project we have defined a number of super object classes including: Observation Objects, Treatment Objects, Order Objects, Charting Objects, Timing Objects, and Alarm Objects, each containing a set of well defined attributes and functions. For example, we had constructed a Swan Gantz Observation Object (*Figure 2*) which contained groups of complex data or objects including: pulmonary arterial pressure, Cardiac Output (CO), Mixed Venous Oxygen Saturation (SVO2), Pulmonary Vascular Resistance (PVR), and Systemic Vascular Resistance (SVR) objects.

Relational model and relational database management systems

The relational model is based on solid mathematical foundation --- set theory and predicate logic. All data are stored in base tables which are treated as mathematical relations. The mathematical basis provides powerful language/operators for dealing with relations and manipulation of stored data. Normalisation theory and procedures applied to the relations aims at avoiding redundancy of data storage, hence preventing problems such as update anomalies and data inconsistency in the database. These advantages enabled RDBMS to become the dominant force and de facto standard since the first relational products began to appear in the late 1900s and early 1980s [12-13]. Recent experience has shown that RDBMS are inefficient and not suited for applications requiring to manipulate complex data structures (e.g. clinical/patient data) and complex relationships among these structures[8]. The normalisation processes tend to spread an entity across several relations resulting in database fragmentation. When applied to OO data design, normalisation often requires that information concerning one object be distributed over multiple relations. Related data are later brought together as needed by join operations. The relational joins required to build complex data structures from many base relations impose an unacceptable performance overhead[14]. When implementing our OO design, we often have to represent information about a single object in several base tables. For instance, data store for the Medication Object has to be split into the Orders, Drug_Adminstration, and Drug_Effects tables. If the drug is to be administered through an intravenous infusion, additional tables are required to store the IV order and administration information (*Figure 3*).

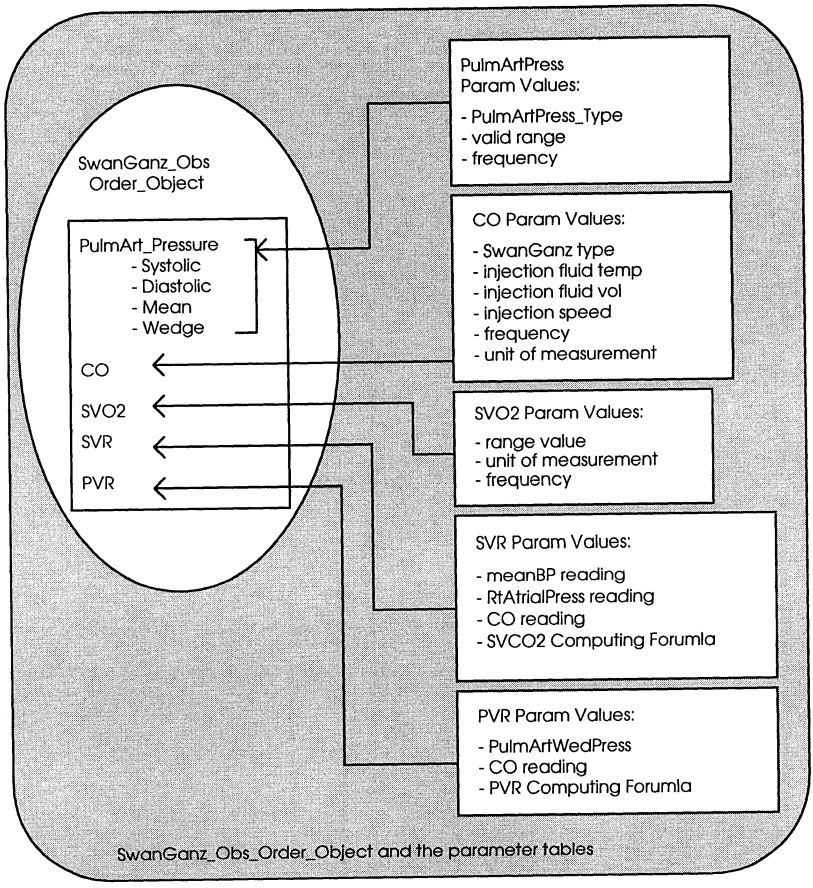

Figure 2: SwanGanz Observation Object

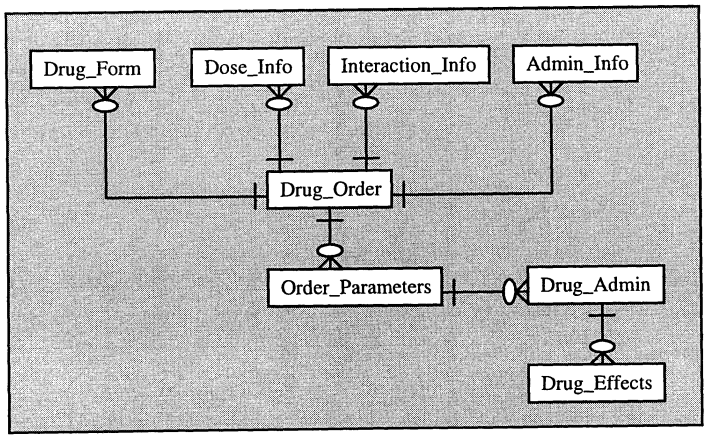

Figure 3: Drug Order and Administration Tables

The rigid structure of relational model has limited semantic power. Due to simplicity and "flatness" of its structure, it loses the valuable information contained in the relationships or "links" among the data/objects[15]. Knowledge about the use of data and their relationships is mapped into and hidden within the application programs. The semantic information is therefore separated from the data to which it relates.

Object-oriented database management systems

An object-oriented database management system (OODBMS) is a persistent data store and management system for collections of objects and object classes. It possesses a number of unique features[16]:

There have been three approaches to building an OODBMS[9]: When we started to implement our CIS design three and a half years ago, we had intensively investigated possible database backends for our system. We studied, in detail, features of OODBMS available and identified a number of concerns that steered us away from OODBMS on market at that time. One of the major concerns was lack of standards in OO data models and their implementation. None of these models or their implementation had emerged as the pre-eminent standard[14]. Choosing any product would lock our development onto a proprietary platform and hence defeated our open system design objective.

OODBMS were initially developed for single-user CAD/CAM applications and to support management of complex objects in complicated engineering projects. A survey of such systems published by Zand et al.[9] in 1995 revealed that only 35% of the OODBMS in their study supported distribution of objects.

While message expressions prove object-oriented systems superior to the relational model, it is impossible to pre-empt all possible queries and provide methods for them. Issues such as how methods should be used to participate queries and how to optimise queries in the presence of arbitrary methods still remained to be resolved[14]. Navigational techniques for traversing lattice/type hierarchy in OODBMS suffer from performance problem especially when the database is large. It is desirable and necessary to have a standard object-oriented query language to retrieve a set of object instances satisfying some search predicates[17]. Unlike RDBMS which has adopted SQL as the de facto standard, no standardised declarative query language was available for OODBMS when we began our development efforts more than three and a half years ago.

Resolving the issues

A number of modifications/extensions have been proposed and implemented in RDBMS to overcome the semantic weakness and fragmentation problem. Kung[11] uses the "join view" method in which some of the objects are defined as views (virtual tables) which can then be operated on by join queries. The view mechanism provides users with an integral view of objects of interest. Wiederhold[18] proposes the "view objects" technique which are specific instantiations of object templates (*Figure 4*). These templates provide a mechanism to create complex objects in a flexible way that joins cannot provide. Barsalou[19] implements a "structural model" and a "view-object" layer to add object-oriented features to RDBMS. The structural model creates a semantic data model by explicitly capturing knowledge about constraints and dependencies among relations in basic RDBM. These techniques added some form of "object-extension" to the relational model. Others[20-21] proposed semantically rich frame structures independent of or mapped onto simple relational tuples.

Instead of reducing the number of base relations in our database, we elected to create all relations necessary for each object. We have also moved towards a post-relational database management system which supports multi-valued data fields. For example, in the Respiration_Observation Object, we are now able to collapse the Resp_Rate, Resp_Pattern, and Resp_Volume into one base relation. Operations on the objects are implemented as stored procedures or subroutines with published interfaces which can be invoked by messaging other program components. For example, a "BP_SELECT" store procedure/subroutine is created which can be activated by the BP_Object to select and return a set of blood pressure reading data according to parameters specified by the users.

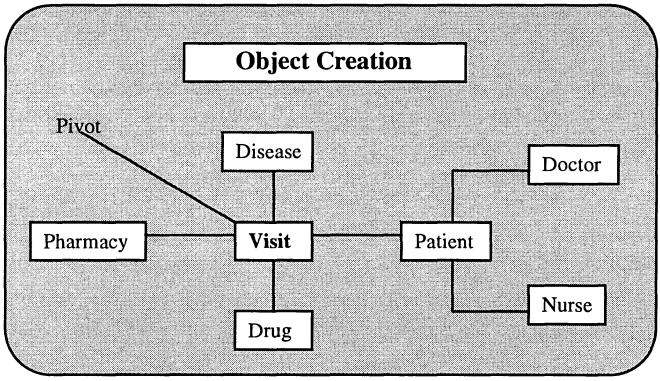

Figure 4: View Object Creation: "Visit" selected as Pivot, activating the creation of links to "Drug_Received" object.

Much improvement has been made to OODBMS since our initial review on these products. Distributed support and Object Algebra/Object SQL[22], or Set-Oriented Query Language[10] are also implemented in some commercial products. Issues of object standards and distributed object management are now being addressed by COBRA[22-23] We are convinced that OODBMS will satisfy our need to effectively manage large amount of complex clinical data in a distributed environment. We are beginning to move towards a COBRA based design and are investigating the use of O2 (an OODBMS developed at GIP Altair in Le Chasnay, France) as our CIS database backend.

Conclusion

The need to be able to deal with large amounts of complex clinical data/objects and easily transfer patient databases between departmental and backup servers pushes our design towards an distributed objects environment instead of the traditional two tier client-server approach. Experiences from our project confirmed the superiority of OO design, especially in the user interface construction and the development of a functionally rich and reliable CIS. However, the implementation of an OO design over a RDBMS during our prototyping stage revealed considerable problems including: mapping of complex objects to base relations, performance degradation due to extensive "join" operations, poor semantic power and inability of RDBMS to encapsulate object functions/methods. Post-Relational Databases seemed to have improved capability of complex object implementation but still lacks the ability to encapsulate object methods and semantic information. We are watching research and development in OODBMS with great interest. Standardisation of object query language

and distributed object management, which are critical for maturation of OODB technology, have recently been published by the ODMG[23] (Object Database Management Group). There is a large body of theory that has been developed for the relational model. Valuable techniques such as query optimisation and normalisation have materialised from the rich theoretical frameworks. Many strands of database research has contributed to the development of OODB, and a comparable body of theory has now been developed for the OODBMS. However, in sharp contract to RDBMS, there is little real world experience in the use and implementation of OODB even in the commercial world, let alone the health informatics community. It took the RDBMS community over 20 years of intensive research, implementation and refinement to accomplish the successes we see today. More active, large scale implementation and improvement of OODB are required for it to be successful and help resolve problems identified in earlier section. We will continue to investigate the use of OODBMS in clinical information systems development.

References

1. Minarelli Della Valle R, Ferri F, Pisanelli DM, Ricci FL, Tittarelli F. CADMIO: Computer Aided Design for Medical Information Objects. in R.A. Greenes, H.E. Peterson, D.J. Protti (eds.) *Proceedings, The Eighth World Congress on Medical Informatics, MEDINFO '95*. Vancouver, 23-27 July. 1995: 231-235.
2. Green RA. Approaches to Sharing and Collaboration through Modular System Design: A Focus on Knowledge Management. In: Timmers T, Blums BI eds. Software Engineering in Medical Informatics. Elsevier Science Publishers. Netherlands. 1991: 409-428.
3. Regan B. Using Object-Oriented Design for Medical Records. In: Lun KC,. Degoulet P, . Piemme PT, Rienhoff O eds. *Proceedings, MEDINFO 92: The Seventh World Congress on Medical Informatics*. Geneva, 6-10 September. Elsevier Science Publishers. 1992: 697-702.
4. Donkers H, Hasman A. Using Object-Oriented Turbo Pascal 5.5. In:. Timmers T,. Blums BI eds. Software Engineering in Medical Informatics. Elsevier Science Publishers. Netherlands. 1991: 323-346.
5. Griesser G, Hofmann G. Developing and Implementing Hospital Information and Communication Systems by Object-Oriented Programming. In: Timmers T, Blums BI eds. Software Engineering in Medical Informatics. Elsevier Science Publishers. Netherlands. 1991: 241-253.
6. Thom J, Chu S, McCrann L, Chandler G, Rogers S, Edgecumbe J. A Nursing Interface. In: Grobe SJ ed. *Proceedings, The Fifth International Conference on Nursing use of Computers and Information Science*. 17-22 June. Texas, USA. Amsterdam:Elsevier. 1994: 250-254.
7. Timpka T, Hedblom P, Holmgren H. Action Design: Using an Object-Oriented Environment for Group Process Development of Medical Software. In: Timmers T, Blums BI. eds. Software Engineering in Medical Informatics. Amsterdam: Elsevier Science Publishers. Netherlands. 1991 151-165.
8. Wiederhold G. Objects and Domains for Managing Medical Data and Knowledge. Methods of Information in Medicine, 1995: 34: 40-46.
9. Zand M, Collins V, Caviness D. A Survey of Current Object-Oriented Databases. *Data Base Advances*. 1995; 26(1):14-29.
10. Ishikawa H, Yamane Y, Izumida Y. An Object-Oriented database system Jasmine: Implementation, application and extension. *IEEE Transactions on Knowledge and Data Engineering*. 1996; 8(2): 285-303.
11. Kung C. Object subclass hierarchy in SQL: A simple approach. *Communications of The ACM* 1990; 33(7):117-125.
12. Date CJ. An Introduction to Database Systems: Volume I. Addison-Wesley Publishing Company. 5th ed. 1990.
13. Halpin T. Conceptual Schema & Relational Database Design. Prentice Hall. 2nd ed. 1995.
14. Ling TW, Teo PK. Towards resolving inadequacies in object-oriented data models. *Information and Software Technology*. 1993; 35(5):267-276.
15. Navathe SB. Evolution of Data Modelling for Databases. *Communications of the ACM*. 1992; 35(9):112-123.
16. McLeod D. Perspective on Object Databases. *Information and Software Technology*. 1991; 33(1): 13-21.
17. Alhajj R, Arkun, ME. An object algebra for object-oriented database systems. *Database*. 1993; August: 13-22.
18. Wiederhold G. Views, Objects and Databases. *Computer*. 1986; 19(December): 37-44.
19. Barsalou T, Wiederhold G. Applying a semantic model to an immunology database. in Stead W. (ed.) *Proceedings, 11th Symposium on Computer Applications in Medical Care*. IEEE Computer Society Press. 1987.

20. Bishop C. A new format for the medical record. *MD Computing*. 1991; 8: 208-215.
21. Canfield K, Bray B, Huff S, Warner H. Database capture of natural language echocardiographic reports: A Unified Medical Language approach. in Kingsland LC. (ed.) *Proceeding, 13th Symposium on Computer Application in Medical Care*. 1989. 559-563.
22. Lee BS. Object-Oriented databases: Systems and standards. *Computers*. 1995; October: 64-65.
23. Cattell RGG (ed.). *The Object Database Standard: ODMG-94*. Morgan Kaufman Publishers. 1994.

Acknowledgement
This paper draws on work undertaken by the Clinical Information Project at Victoria University of Technology (Melbourne, Australia), a joint research project of: Victoria University of Technology, Spotless Services Limited and Global Networks Pty Ltd.