UNIVERSITI PUTRA MALAYSIA

A DESIGN AND IMPLEMENTATION OF VERSION MODEL IN OBJECT-ORIENTED DATABASES

ZULKIFLI BIN YAZID

FSAS 1997 23

A DESIGN AND IMPLEMENTATION OF VERSION MODEL IN OBJECT-ORIENTED DATABASES

By

ZULKIFLI BIN YAZID

the Degree of Master of Science in the Faculty of
Science and Environmental Studies
Universiti Putra Malaysia

July, 1997

Say Hay Amor Bliss Upon Day Dream

ACKNOWLEDGEMENTS

The author wishes to express his deepest gratitude to Dr. Ali Mamat, Chairman of the Supervisory Committee, for his guidance, assistance and encouragement during the entire course of study.

The author also wishes to express his thanks to members of the Supervisory Committee Prof. Madya Dr. Abu Talib Othman and Dr. Abdul Azim A. Ghani, Lecturers, Department of Computer Science, for their kind assistance and scholarly guidance.

The author further likes to express special thanks to En. Roslan Ismail and Miss Shafida Hermy Halamy for their valuable assistance, motivation and comments during the preparation of the thesis.

The author also acknowledges the cooperation, moral support and encouragement from his loved ones and friends.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI	:	Artificial Intelligence
CAD	:	Computer Aided Design
CAM	:	Computer Aided Manufacturing
CASE	:	Computer Aided Software Engineering
DBMS	:	Database Management System
GSM	:	Generic Semantic Model
ICE	:	Integrated Computer Environment
IS	:	Information System
ODE	:	Object Database and Environment
OID	:	Object Identifier
OODBMS	:	Object-oriented Database Management System
SEE	:	Software Engineering Environment
SPPD	:	Sistem Pengurusan Pangkalan Data

Abstract of the thesis submitted to the Senate of Universiti Putra Malaysia in fulfilment of the requirements for the degree of Master of Science.

A DESIGN AND IMPLEMENTATION OF A VERSION MODEL IN OBJECT-ORIENTED DATABASES

By

ZULKIFLI BIN YAZID

July, 1997

Chairman : Dr. Ali Mamat

Faculty : Science and Environmental Studies

Due to the advances in computer technology, new applications such as office automation, software engineering and computer aided design (CAD) have emerged. These new applications not only demand fast retrieval and modification as the earlier applications but also new requirements, for instance, the capability to represent complex object. Many approaches have been proposed in order to meet the new requirements. It is claimed that Object-oriented Database Management Systems (OODBMSs) offer a good solution. One of OODBMSs' facilities is the version control management. With a version control management, the management and

control of object versions can be done in a systematic way.

However, object versions have been plagued by the lack of adequate knowledge for managing and controlling version activities. A structured approach is needed within which object versions can effectively be applied specifically in software engineering. Consequently, the study addresses the design and implementation of a version model in object-oriented databases.

The study has proposed a generalised version model. The model can overcome some drawbacks of some previous version models. The generalised version model is then incorporated into a data model to provide a global view in developing a structured schema and allows the semantic of versions to be represented at the data model level. Elements of the data model are then programmed into version control mechanisms and embedded into the prototype of a versions management. The principal task of the prototype is to determine whether object versions are manageable and controllable. Evidently, the prototype draws a set of guideline for managing and controlling object versions.

Indirectly the guideline approves the efficiency of the proposed generalised version model to overcome some drawbacks of some previous version models. Moreover, the guideline provides a direction in developing a structured schema and allows the semantic of versions to be represented at the data model level.

The evidence which has been summarised from the study reveals that the notion of object versions in IS activities should be given attention. This, in turn, can make the interaction with database systems a more pleasant experience for end-users and at the same time increases their productivity.

Abstrak tesis yang dikemukakan kepada Senat Universiti Putra Malaysia bagi memenuhi syarat untuk Ijazah Master Sains.

REKABENTUK DAN PERLAKSANAAN MODEL VERSI DI DALAM PANGKALAN DATA BERORIENTASIKAN OBJEK

Oleh

ZULKIFLI BIN YAZID

Julai, 1997

Pengerusi : Dr. Ali Mamat

Fakulti : Sains dan Pengajian Alam Sekitar

Berikutan daripada kemajuan dalam teknologi komputer, banyak aplikasi baru seperti automasi pejabat, kejuruteraan perisian dan bantuan komputer untuk rekabentuk (CAD) telah muncul. Aplikasi-aplikasi baru ini bukan hanya memerlukan capaian dan modifikasi yang pantas seperti aplikasi sebelumnya tetapi memerlukan juga keperluan tambahan seperti keupayaan mewakil objek kompleks. Pelbagai pendekatan telah dicadangkan untuk memenuhi keperluan baru ini. Salah satu penyelesaian yang baik ialah Sistem Pengurusan Pangkalan Data (SPPD) Berorientasikan Objek. Salah satu kemudahannya ialah untuk menguruskan pengawalan versi.

Dengan ini, pengurusan dan pengawalan versi objek boleh dilaksanakan dengan lebih sistematik.

Walaubagaimanapun, perlaksanaan versi objek menghadapi masalah dari segi pengetahuan mengurus dan mengawal aktiviti versi. Suatu pendekatan berstruktur diperlukan supaya versi objek boleh dilaksanakan secara berkesan khususnya di bidang kejuruteraan perisian. Oleh itu kajian ini cuba mengenalpasti satu rekabentuk dan perlaksaan model versi di dalam pangkalan data berorientasikan objek.

Kajian ini mencadangkan suatu model versi yang mempunyai sifat am. Model-model ini boleh memberi penyelesaian kepada masalah-masalah yang dihadapi oleh model-model sebelumnya. Model versi ini digabungkan dengan model data untuk memberi pandangan global pembangunan skima berstruktur dan membenarkan semantik sesuatu versi diwakili di peringkat model data. Unsur-unsur model data akan diprogram ke dalam mekanisma pengawalan versi dan dicerna ke dalam Prototaip Pengurusan Versi. Tugas utama prototaip ini ialah untuk menentukan samada versi objek tersebut boleh diurus dan dikawal. Terbukti, hasil dari prototaip dapat memberi

satu garispanduan untuk mengurus dan mengawal versi objek. Secara tidak langsung garispanduan ini dapat menunjukkan keberkesanan model versi yang dicadangkan untuk menyelesaikan masalah model-model sebelumnya. Begitu juga garispanduan ini memberi arah di dalam pembangunan skima berstruktur dan membenarkan semantik sesuatu versi diwakili di peringkat model data.

Bukti yang diberi oleh kajian ini menunjukkan bahawa versi objek di dalam aktiviti Sistem Maklumat (IS) harus diberi perhatian. Ini boleh memudahkan interaksi di antara pengguna dengan sistem pangkalan data, sekaligus dapat meningkatkan produktiviti.

CHAPTER I

INTRODUCTION

Conventional Database Management Systems(DBMSs) especially Relational DBMS, have long been recognised as powerful and efficient tools for managing organisation information. These systems are particularly suited for application areas such as banking, payroll, inventory and the like. Due to the advances in computer technology, new applications such as office automation, software engineering and computer aided design (CAD) have emerged. These new applications not only demand fast retrieval and modification as the earlier applications but also new requirements, for instance, the capability to represent complex object.

Many approaches have been proposed in order to meet the new requirements. It is claimed that Object-oriented Database Management Systems (OODBMSs) offer a good solution (Bertino and Martino, 1991). In OODBMSs every entity is considered as an object. An object can be as simple as a single unit data or as complex as we

want. In addition, OODBMSs provide some facilities which are not found in other types of database systems. One of such facilities is the version control mechanism and therefore OODBMSs are very suitable for engineering design applications.

In applications such as computer aided design (CAD), computer aided manufacturing (CAM) or computer aided software engineering (CASE), the same object undergoes multiple changes, or state transitions. It is desirable to access or investigate previous states, or versions, of the object. With a version control mechanism, the organisation of versions can be done in a systematic way. While few commercial DBMSs appear in the market offering version facilities, some issues pertaining to the object versions remained to be solved.

This chapter introduces the basic concepts and terminologies. This chapter also highlights the inherent weakness that impedes the implementation of object versions and argues for a need of research to overcome drawbacks of object versions. This chapter also presents objectives of the study. Finally, the chapter presents the organisation of the thesis.

Basic Concepts and Terminologies

This section introduces some aspects of object-orientation as the background of the study. The introduction starts with the object-oriented features and follows by a brief explanation on the emergence of object-oriented database as resourceful tools for new applications.

This section also highlights the database support for engineering applications which has caught the attention of most software developers.

The Object-oriented Paradigm

The object-oriented is an approach primarily introduced in the design of advanced programming languages and environments. The approach was first introduced by the language called Simula (Birtwistle et al., 1973), a language for programming computer simulation. Most recently the language of C++ and Smalltalk (Goldberg and Robson, 1983) have become the most widely known object-oriented languages.

Several fundamental ideas have been proposed in underlying the object-oriented features. These features

have been discussed extensively by authors such as Bertino and Martino (1991), Coplien (1992), Joseph et al. (1991), Martin (1993), Mattos et al. (1993), Nierstrasz (1989), Ozkarahan (1990) and, Unland and Schalageter (1990). The object-oriented features are as the following.

Objects and Classes

Each real-world entity is modelled by an object. Each object is associated with a unique identifier (OID). An object is an instance of an object type. An object type is a category of object. An object is concerned with both data and the methods with which the data is manipulated. Each object has a set of instance attributes and methods. The value of an attribute can be an object or a set of objects. This characteristic permits arbitrarily complex objects to be defined as an aggregation of other objects. The set of attributes of an object and the set of methods represent the object structure and behaviour, respectively. The term class refers to the implementation of an object type. A class can be defined as a specialisation of one or more classes. A class defined as a specialisation is called subclass, and inherits attributes and methods from its superclass. A class specified its instances by defining:

(a) a structure, that is, a set of instance attributes

(b) a set of messages that defines the external interface

(c) a set of methods that are invoked by messages.

Method

Operations are used to manipulate the data structure of an object type. An operation will be sent as a message to manipulate the data structure of an object of an object type. Once the operations are encoded in software, they are usually referred to as method.

Message

To perform operations on an object, we need to send a message. The message causes an operation to be invoked but it does not indicate how the operation should be performed.

Inheritance

An object type can has subtypes. There is a hierarchy of object types, subtypes, and so on. A class implements the object type. A subclass inherits

properties of its parent class; sub-subclass will inherit the property of the subclass and so on.

Encapsulation

The object hides its data from other objects and allows the data to be accessed via its operations. Encapsulation is the result of hiding the implementation details of an object from its user.

Some other features of object-oriented such as polymorphism, persistence, object identity, etc., were also highlighted by Andrew (1990), Bancilhon et al. (1988), Martin (1993), Ozkarahan (1990), Unland and Schalageter (1990). However, the five elements which have been described are the core of object-orientation.

Object-oriented Databases

One of today's most burning database issues is how to adequately support new classes of applications that are not well served by conventional database system (Unland and Schlageter, 1990). For example CAD, CASE, data intensive Artificial Intelligence (AI) application or image and voice processing place demands on database systems that exceed the capabilities of conventional database systems by far. However, recent DBMSs such as

Avance (Bjonerstedt and Hulten, 1989), Gemstone (Breitl et al., 1989), Iris (Wilkinson et al., 1990), O2 (Bancilhon et al., 1988), Orion (Kim et al., 1989), Postgres (Rowe and Stonebraker, 1987) and Vbase (Andrew et al., 1990) have taken up the challenge by designing an architecture which exceeds the capabilities of conventional database systems and supported the object-oriented features.

Ozkarahan (1990) gives two reasons that contributed to the influence of object-oriented features in the database field. First, the database technology has evolved rapidly and widespread over time. Second, many areas in the computer science have become more integrated. The overall objective, therefore, is to extend the technology for capturing more meaningful information. This implies to the semantic and behavioural aspects of data into modelling formalism.

The first publicised semantic model appears in 1974 (Abrial, 1974). The area matured rapidly in recent years, and database researchers have turned their attention in incorporating behavioural aspect of data into modelling formalism (King, 1988; Lockemann et al., 1990). Thus, the modelling formalism is integrated with the object-oriented paradigm and paved the way for

research directions on object-oriented data modelling. Several attempts (Atkinson et al., 1989; Stein et al.,1989; University of California, 1990) have been made to forge a common agreement on the modelling concept but they have so far been unsuccessful (Ling and Teo, 1993). However, the core concept of object-oriented data model (Kim et al., 1989; Ozkarahan, 1990) lies with the usual features of object-oriented languages such as the notion of classes, objects, encapsulation, inheritance, persistence, data abstraction, etc.

Database Support for Engineering Design Applications

An engineering application process involves numerous types of information and enormous amount of data. These data items can be maintained in various formats and media. However, Liu and Horowitz (1991) have suggested the use of database support to maintain the data and improves the productivity of software development. In addition to this, the database is also useful in software maintenance and reverse engineering (Ketabchi et al., 1989). A great deal of work has been taking place in developing new database system based on this approach. Although there has been an increase awareness among software developers about the strength