Applications of Object-oriented Databases to Publishing Systems

N.Kuwano, T.Kanda, Y.Mohri
Fujitsu Limited
Y.Izumida
Fujitsu Laboratories Ltd.
R.Sato
Fujitsu Cyugoku System Engineering Ltd.
O.Yamada
Mazda Motor Corporation

Abstract

The application of object-oriented database has only started. We applied an object-oriented database "Jasmine* to an automobile service document publishing system "MASCOT" and confirmed its practicability.

The service material is a "hyper document" which includes logical structure and reference structure. The conventional database cannot deal with them easily. To overcome these difficulties, in MASCOT, each node of logical structure is directly associated with its corresponding class and language "Jasmine/c" is directly embedded in the text.

The access to a heterogeneous database or the data conversion can be encapsulated by using the procedural attributes and demon. This encapsulation hides the data location and the data unit from the application programmer.

We confirmed Jasmine has the ability to express the object, to manipulate, and to extend, which is required for the repository of hyper document. And we also confirmed that it achieves practical performance even on a workstation, by adopting static binding and caching method.

1 Introduction

The research and development of object-oriented databases has been increasing recently [1,2,3,4,5,6]. The investigation of applications, however, has barely started. It is unusual to find papers discussing issues of their practical application.

To develop new technology that is really useful, it is important to consider possible fields of application and to pay careful attention to feedback from users. The fields such as knowledge based system, CAD, CAP and CASE are potential fields. We applied an object-oriented database "Jasmine" [7,8] to a document processing system, and confirmed its practicability. The system, MASCOT (Mazda's Service materials COMposing system) [9], aims at increasing the efficiency in making and publishing automobile service materials.

Conventional document systems manage the documents in their own file formats. Therefore it is becoming increasingly difficult to satisfy various advanced needs, such as:

- To access information in the document from other applications
- To share and reuse the document information among documents
- To process various types of multi-media in an integrated way
- To exchange documents with other document systems

To satisfy these needs, standardization of the document model and the development of the document database technology are important. Document models are becoming standardized, and ODA [10] or SGML [11] are gradually being accepted in the publishing and the engineering fields. The conventional DBMS is not suitable for a document database because it cannot easily deal with various relationships, and manipulate documents that are a complex of multi-media. Therefore the application of a object-oriented database is necessary for document management [12,13,14].

In this paper, the application of Jasmine and its evaluation will be explained.

In chapter 2, the background of the development of MASCOT, the solution to the existing problems and essential conditions for the data management is stated. The requirements for multi-media data management and document structure management is also explained.

In chapter 3, the outline of Jasmine will be introduced to show that it satisfies the requirements mentioned in chapter 2.

In chapter 4, the system configuration of MASCOT and the application of Jasmine will be given. In "Application", the class structure design on the document structure and extended function for structural documents will be explained.
In chapter 5, the model representation ability, object manipulation ability and extendability of Jasmine will be described and confirmed. We will also show that it satisfies the performance required for the systems to operate on a work station.

In chapter 6, the future development of Jasmine will be described.

2 Background of the development of MASCOT and solution of problems

A wide, advanced knowledge is now necessary for after sale service as well as design and production, as a result of the increase in the variety of the cars and technological innovation in the car electronics field. Service materials such as the workshop manual and owner's manual play an important role in after sale service. The publication section must meet the demands of dealing with the increasing number of documents, shortening their production time and maintaining their quality.

2.1 Counter measures to present problems

In the MASCOT system, the following measures are taken to achieve the following three targets.

1. To deal with the increasing number of documents

Mazda publishes 300 kinds of service materials in 8 languages in 110 countries. They total some 100 thousand pages a year. The amount of documentation increases in proportion to the increase in the variety of the cars and the new technologies used in them. For example, the number of car types has increased by 40% for these two years, and the average number of pages in a document by 30%.

To deal with the increasing work, it is absolutely necessary to reuse existing documents and to computerize the editing. In particular, the reuse of the documents is useful because 70% of a new workshop manual can be made by cutting and pasting existing manuals.

Since an workshop manual consists of explanations part by part, the creation time will be greatly shortened if reusable data can be automatically fetched using a keyword in a car specification. Therefore as shown in Fig. 1, MASCOT is designed to search the existing data by using specification parameter (20 items such as engine type and transmission type). The retrieved data will be automatically arranged according to a specified order to make a rough document.

2. To shorten the document creation time

Creation of service materials needs to be speeded up as the development time of a given automobile is shortened. For example, a manual which was written in 5 months before must now be finished in 3 months.

To shorten this time, it is necessary to automate the block copy making, to reduce the need for brushing up, and to simplify whole process.

By letting the writer operate WYSIWYG type EP system himself, the block copy making can be skipped and the number of the layout corrections can be reduced. The making of an English manual will be efficiently done by utilizing a machine translation system and reusing the existing English manual.

3. To maintain the quality

Words or expressions sometimes differ according to writers, and even the repair specification data are occasionally different from those in the other manuals even though they may be numbers specified in the same procedure.

The AI brush-up support function and the repair specification database are provided to maintain the quality of the manuals.

The brush-up support detects errors and inappropriate expressions in the documents by using a database which contains typical misspellings and ambiguous expressions.

Numbers specified in service procedures are stored in the repair specification database, which is separated from manual database. To avoid errors in copying the numbers from a manual to another manual, numbers are always taken from repair specification database, and the updated, correct value will be always given to the manual.
2.2 Requirements of the DBMS

Workshop manuals, one of the target documents to be handled with MASCOT, have the following characteristics (cf. Figure 2):

- Large size (1,000 - 1,500 pages/volume).
- Regular contents and layouts.
- Large amount of data on specifications.
- Common document parts with other maintenance manuals.
- Likely to be translated into other languages.

The mechanisms and functions which a DBMS needs in order to easily store and retrieve this kind of document in database are described below.

1. Managing Multi-media Document Parts

Workshop manuals consist of illustrations (images), text and drawing, and thus a DBMS should be able to store and manage these multi-media document parts. To access a particular part stored in a large database, rapid keyword retrieval is necessary and since a particular document part is to be used in several manuals, a shared management system is also necessary.

Considering the above, a DBMS must have the following capabilities:

- Store and manipulate multimedia data.
- Rapidly retrieve with keywords.

2. Logical structure

When using data with an automatic collection function, an arbitrary level is chosen from a block (illustration and explanation in a pair) level to a chapter level in order to accurately utilize rough drafts. Then, retrieved existing drafts are compounded into one document. To efficiently retrieve and compound a document, managing those document parts in hierarchical structure is the most appropriate.

Therefore, the DBMS needs the following:

- Architecture modelled on a hierarchical structure.
- Controllable along hierarchical structure, or, if not, the possibility of install this kind of manipulation as a DBMS function.
- Possibility of rapid retrieval along hierarchical structure.

3. Reference structure

Various repair specification data and illustrations of tools necessary for automobile maintenance are included in a workshop manual. Since these data are frequently referred to for other materials and altered, they are stored separately in one database. To incorporate them into documents, associating the referred data with the document, and making alternations at actual incorporation (e.g. units for values, zoom of images, etc.) are necessary.

As a special association, a link for translation is available. When reusing an existing Japanese sentence, translation procedures are reduced by using the existing corresponding English translation. Thus, it is necessary to store the original Japanese sentences and their English translations in a form where they are associated.

To manage the interrelation of medias, the following capabilities are required:

- To associate each media, and to navigate them easily.
- To easily install data conversion function when referring to data.
- To Access the data in the unified way, even if the data exist in heterogeneous database.

3 Object-oriented Database -Jasmine-

As will shown below existing DBMS's cannot satisfy the requirements described in 2.2. We will then consider Jasmine, an object-oriented database system, for which Fujitsu Laboratories have completed a prototype.

Jasmine is a system in which AI technology and database technology are combined, and which makes it efficient and easy to develop, use, and maintain a multi-media knowledge base.

3.1 Design goal of Jasmine

The design goals for this system are described below.

1. Management of a large quantity of knowledge objects

A knowledge representation language shows a tendency to encapsulate the data, procedure and rule within the object-oriented paradigm, based on the frame. This languages have a strong capability to express the knowledge, but lack the ability to manage the mass of knowledge efficiently.

2. Data modeling and manipulation under the complex data

Conventional RDB's have the following problems.

- It enables only indirect value-oriented modeling.
- It cannot represent directly the semantic relationships between data.
- It cannot manipulate in a unified way composite media such as images, graphics, text, and documents which are consist of these media.
3. Definition and manipulation of multi-media

In order to manipulate many diverse forms of multimedia the ability to define the form of the new media and to extend the manipulation are important. Conventional DB's, however, lack these abilities.

3.2 Features of Jasmine

To solve above problems, Jasmine is based on an extended functional data model with the object-oriented paradigm. Its features are shown below.

1. A set of objects is called a class. Associative query is possible through a class.

2. A function defined on a class is called an attribute of the class. The attributes can be classified as static or procedural. A static attribute is used to connect one object with others statically, as well as to maintain static values. A procedural attribute is a procedure described with Jasmine/C, an object manipulation language which is an extension of the C language. It enables the determination of dynamic values and the expression of object behaviors.

3. A class hierarchy represent the generalized relationships, and superclass's attributes are inherited to subclass. This inheritance makes it possible to do differential programming. Looked at from the viewpoint of set theory, the superclass is the union of the set of subclasses, this makes comprehensive retrieval possible.

4. Demon can be defined in attributes. Demon means the procedure automatically invoked when values are updated, added, or deleted. The use of the demon function enables consistency control with flexibility.

Jasmine was adopted for document management in MASCOT, since the above characteristics fulfill the requirements explained in 2.2. that is,

1. The logical structure and reference structure can be represented directly in the relation between classes. Moreover, it is possible to retrieve the structure at high speed with static attributes. In RDB's, the frequent occurrence of join operations decreases the structure retrieval speed.

2. As the object-oriented database system provides a data abstraction facility, it is easy to extend the new multi-media such as pictures. In RDB's, the application program needs to manipulate the multi-media itself. Therefore, the application program must be revised to implement the new type of media.

3. The access to a heterogeneous database or the data conversion can be encapsulated by using the procedural attributes and demon. This encapsulation hides the data location and the data unit from the application programmers.

4 Implementation with Jasmine

This chapter describes the capacity of the database, the system configuration, and the design of Classes and Methods.
4.1 Capacity of Database and Flow of Data

A workshop manual is a fairly large document, averaging almost 1,500 pages.

Number of Logical Structure Objects approx. 20,400/volume
(Number of Instances in Logical Structure Class)
Number of Non-Leaf Node Objects approx. 2,700
Number of Leaf Node Objects approx. 17,700
Number of Content Objects
(Number of Instances in Content Class)
Number of Text Objects approx. 9,050 2MB
Number of Illust. Objects approx. 8,650 110MB

About 7.5G bytes of disks would be required at a rate of production of 100,000 pages a year. MASCOT uses a distributed system of host computer and workstation as a large amount of data is accumulated.

4.2 System Configuration and Flow of Data

The system of MASCOT is shown in Figure 3.

The host computer is used only as a master database for a large amount of service materials. When writing, the form in writing section and reusable data from previous documents are downloaded to the local database on the workstation.

On the workstation, word processor, DTP, and other application programs work together in cooperation with the core of document management provided by Jasmine. The functions above make reuse, writing, editing and composition possible. To display documents using DTP programs, start the layout method (the procedure for layout), and output layout information and actual data necessary for the DTP system. Then invoke the DTP display process to complete the procedure of document display.

4.3 Class Design

A class structure that expresses a document was designed based on the following concepts (see Fig. 4).

1. A document consists of:
   - a document class that contains the overall accompanying information,
   - a logical structure class that represents the logical structure of a document, and
   - a content class that contains the actual text data and the image data.

2. Each node of the logical structure is directly associated with its corresponding class so that its logical structure can be easily expressed.

3. Classes are connected with bi-directional links for a quick, vertical navigation between nodes of the hierarchical structure. In Jasmine/C, the navigation can be easily described using dot notation as follows:

   Example) To obtain all the medium-item headings contained in the Engine section:
   section.large-item.medium-item.heading
   where section.name='engine'

4. The superclass controls all the operation procedures such as display, printing, translation, and editing, providing a compact, shared program. The subclasses can use the inherited procedures.

A reference structure is designed as follows.

1. An object manipulation language is directly embedded in the text (see Fig. 5). The embedded object manipulation language is invoked by a reference demon when the desired text is referred. Thus, the latest repair specification data can be built in the text.
In order to obtain a specification for a tool to be used from an external resource (a special tools file), describe as below. A tool object is an object for accessing the special tools file.

```
tool.radius where tool.name="flange holder"
```

Figure 6: Access to external resource

2. If the required information is not contained in Jasmine, the external resource can be accessed through the object that manages it (see Fig. 6). This approach provides an integrated way for accessing the data not by hiding the data formats and access methods in the objects. This kind of approach is considered important because it not only maintains the value of the existing data but also reduces the burden on those programmers who operate multiple systems.

4.4 Document manipulation method

The object manipulation language has been enhanced to manipulate document by adding new functions to the document object, logical structure object, and content object as listed in Table 1. The document manipulation method consists of a basic function that handles logical structure and an application function linked to a desktop publishing system and a machine translation system.

In Jasmine/C, statements are written in a combination of "where" clause conditions and document manipulation methods. This syntax allows the user who develops application programs to write sophisticated functions in a simple way.

Example) In this example, all parts included in a large item that illustrates XX alternator are layout-output.

```
/* Defines multi-value variable x */
logicalStructure x multi;

/* Recursively retrieves the nodes in the logical structure class specified by the illustrated class up to a large item node. */
x = illust.logicalStructure.get-ancestor(large-item)
    where illust.keyword='XX alternator';

/* Sends layout messages to the retrieved large item. */
x.layout();
```

5 Evaluation

Jasmine was adopted as a document database to realize the following: representation of complex data structure, flexible structure retrieval, and high-speed access to data. The problems in the areas of representation of structure and structure retrieval are easily solved by direct modeling facility and extension facility by using the method as
described in Chapter 4. This chapter describes the simple performance values obtained through actual operation, and also explains the valuable characteristics of the Jasmine as a database programming language in developing the document manipulation methods.

5.1 Evaluation of performance

We used the following two patterns which are the ones most frequently used for the automatic collection:

Pattern 1: To retrieve all object identifiers of document elements in the Engine Section.

Pattern 2: To retrieve all text with the key word “drive belt.”

The models and setups for the simulation are close to real operation.

Testing environment

Machine: FMR70HX3 Micro computer (cpu 80386 25MHz)
OS: MS OS/2 V1.1
Memory size: 16M bytes (DBMS 1.8M bytes included)

Tested models

Class structure: Same as shown in Figure 4.
Number of logical structure instances in the section: 1830
Number of instances within the text: 725
Number of instances within the illustration: 680

<table>
<thead>
<tr>
<th>Method name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>select-child</td>
<td>select child nodes</td>
</tr>
<tr>
<td>select-descendent</td>
<td>select descendent nodes</td>
</tr>
<tr>
<td>select-parent</td>
<td>select parent nodes</td>
</tr>
<tr>
<td>select-ancestor</td>
<td>select ancestor nodes</td>
</tr>
<tr>
<td>delete-node</td>
<td>delete node, update relative links</td>
</tr>
<tr>
<td>delete-tree</td>
<td>delete the specified tree</td>
</tr>
<tr>
<td>update-link</td>
<td>update is-part-of link</td>
</tr>
<tr>
<td>copy-tree</td>
<td>copy the specified tree</td>
</tr>
<tr>
<td>move-tree</td>
<td>move the specified tree</td>
</tr>
<tr>
<td>create-child</td>
<td>create child node</td>
</tr>
<tr>
<td>layout</td>
<td>layout document elements using</td>
</tr>
<tr>
<td></td>
<td>layout rules, output it in OASYS-EP format</td>
</tr>
<tr>
<td>translate</td>
<td>translate text by using ATLAS</td>
</tr>
<tr>
<td></td>
<td>and link between Japanese text and English text</td>
</tr>
</tbody>
</table>

Measuring data

<table>
<thead>
<tr>
<th>Patterns</th>
<th>Total response time</th>
<th>Database access time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pattern 1</td>
<td>4.5 seconds</td>
<td>1.3 seconds</td>
</tr>
<tr>
<td>Pattern 2</td>
<td>2.8 seconds</td>
<td>0.2 seconds</td>
</tr>
</tbody>
</table>

Notes: The DB access time represents the processing time of the following sentences:

Pattern 1:
```
section.get-precedent()
where section.id='El';
```

Pattern 2:
```
text.body
where text.keyword='drive belt';
```

Consideration

To more accurately evaluate the performance of Jasmine, precise bench mark models must be tested and comparison with conventional RDB’s must be made. But from the ratio of database access time over the total response time, we have demonstrated that Jasmine achieves well in practical performance on OS/2. The following mechanisms enabled high-speed processing:

1. Static binding

Jasmine/C solves inheritance, optimizes, and determines the relative address for the attribute value reference in pre-compiling. This prevents the system from being slowed down, which often occurs for the object-oriented systems.

2. Active Object Space (AOT)

Jasmine retrieves the objects from a database and stores them in the AOT in the memory at object manipulation. Objects in
the AOT can be rapidly accessed by using the identifiers registered in a hash table.

5.2 Effectiveness on programming

In developing the document manipulation method, the following characteristics of Jasmine/C were useful for efficient programming.

1. Persistent object manipulation

Jasmine/C does not have a special way of manipulating a database. The database is treated basically the same way as the variables and structures in memory. Therefore, the programming load for manipulating databases was lightened.

2. Set-oriented query function

In the conventional list processing, the target data was evaluated by if statements via pointers. Jasmine/C only requires conditional statements. Because in Jasmine/C queries are not considered as methods, a query and method can be written simultaneously and this contributed to minimizing the programming load.

3. Object-oriented programming

The inheritance mechanism and polymorphism of object-oriented programming have minimized the necessary procedures. In particular, when the text or images are output, by sending an output message to the content class the output method applicable to the class to which the instance belongs is selected. This enabled abstract programming.

6 Future themes

The object-oriented approach has been implemented in various areas such as the database, computer languages, and user interfaces, and has passed the development stage to its application stage. We have demonstrated in this paper that Jasmine, an object-oriented database, can be used as a hyper-document database and can be put into the real operation considering its performance and function.

We would like to further our studies in the following areas:

1. Application to other fields

By applying Jasmine to the CAD database and CASE repository, we should be able to enhance our application technology for versions and design transaction. To specifically grasp the characteristics and advantages of object-oriented databases, performance and productivity must be measured in various fields and compared with other databases.

2. Application to the distributed network

Due to recent diffusion of workstations and microcomputers, the system must meet with the requirements of the distributed network environment. The system is will need to cope with the server-client model and distributed database, as well as the client systems with multiple OS.

3. Expansion to new media

CD-ROM publishing is a way of using service material accumulated in a database. A CD-ROM has the advantage over paper data of instant data retrieval. The MASCOT database can contribute to efficient CD-ROM publishing by using its logical structure and reference structure.

We would like to further our research for the maintenance expert system that has functions of failure diagnosis and CD-ROM.

Acknowledgements

We would like to express our special thanks to Professor Akifumi Makinouchi of Kyushu University for his warm guidance to us in developing this system and preparing this paper. We also would like to express our deep appreciation to the Service Materials Group of Mazda Motor Co. who granted us the opportunity to try our new system, and to the many Fujitsu people, Techno-project Ltd. and Fujitsu Techno System Ltd. who helped in its development.

References


