データベース・システム　72－ 8
（1989. 7. 20）

オブジェクト指向型マルチメディア知識ベース　Jasmine　の実現

Implementation of the Object-Oriented Multi Media Knowledge Base System Jasmine

鈴木　文雄　＊　　石川　博　＊　　山根　康男　＊＊　　宮城島　実香　＊＊＊
青島　正明　＊＊　　小櫻　文彦　＊　　泉田　義男　＊　　牧之内　顕文　＊
＊　富士通　　＊＊　富士通研究所　　＊＊＊　富士通静岡エンジニアリング
Fumio Suzuki ＊　Hiroshi Ishikawa ＊　Yasuo Yamane ＊＊　Mika Miyagishima ＊＊＊
Masaaki Aoshima ＊＊　Fumihiko Kozakura ＊　Yoshio Izumida ＊　Akifumi Makinouchi ＊
＊ Fujitsu Ltd.　＊＊ Fujitsu Laboratories Ltd.　＊＊＊ Fujitsu Shizuoka Engineering Ltd.

1. はじめに

Jasmine 〔1〕－〔4〕はオブジェクト指向型マルチメディア知識ベース管理システムである。本稿ではこのシステムの設計思想と実現技術について報告する。

今日のような高度情報化社会では大量のデータを管理するデータベースは重要なものとなってきた。しかし，エンジニアリング等の高度応用では関係データベースは限界がある。新しい情報管理システムとして次のような要求がある。

(1) 人工知能分野の応用のための大量知識を効率良く格納し，かつ高速に格納・操作・管理できる。

(2) データベース分野の高度応用に必要となる複雑なデータも扱える。

(3) 何れの分野でも必須であるイメージ，グラフィックス等のマルチメディアを格納・操作・管理できる。

(4) 様々な応用（ＣＡＤ，ＯＩＳ，ＭＭＩ等）を簡単に作成できる（拡張性に富む）。

従来の情報管理システムとして，知識表現言語（ＡＩシェル）〔7〕－〔9〕と関係データベース管理システム〔10〕がある。双方の問題点について述べる。

従来の知識表現言語の問題点を列挙する。

(1) 処理速度が遅い。実現言語が LISP のため遅い物もあるが，オブジェクト指向のデータアクセスの基本である遺伝処理の高速化がなされていない。

(2) 大容量かつ大量の知識を効率良く管理する機能に欠ける。あるとしても外部データベース機能へのアクセス機能であり，知識表現の枠組みの中に統一的に組み込まれていない。

従来の関係データベース管理システムの問題点を列挙する。

(1) テーブルという単純なデータ構造のために，エンジニアリング等の分野で必要とされる複雑なデータ構造を表現できない。

(2) データの動的側面を表現できない。

(3) メタデータ操作言語がデータ操作言語と異なり複雑である。

上記の問題点を解決するシステムとして Jasmine を設計・開発した。 Jasmine の特徴を以下に列挙する。

(1) 大量の知識・データを拡張関係データベース（非正規系テーブル〔5〕）〔6〕に効率良くかつ高速に格納・管理する。データベースへのアクセス機能は知識表現の枠組みの中に統一的に組み込まれている。

(2) データと手続きを一体としたオブジェクトを扱うので，データの動的側面を表現でき，それらを組合せて利用できる。

(3) デモンにより一貫性の制御ができる。

(4) 未定義値に対するデフォルト値検索がある。さらに，空値も扱える。

(5) 高速である。それはＣ言語で記述されていることとプリプロセッサによる最適化により達成された。さらに，２次記憶の上に Active Object Space を設けることにより，主記憶上の処理も含め，処理を高速にした。

(6) 概念が単純であり，ユーザが理解し易く，また，システムも単純に実現できた。即ち，すべてがオブジェクトであり，クラスとインスタンスという２つの概念でシステムを構築している。さらに，クラスもインスタンスと同様に操作することができる。

(7) 拡張性（柔軟性）がある。オブジェクト指向モデルなので下位クラスを追加することにより，応用の作成が容易である。会話型言語 Jasmine/I のサポートにより知識ベース開発はさらに易しくなった。

(8) 属性情報が不完全な知識を管理できる。即ち，上位下位関係のリーフクラス以外のクラスにもインスタンスを作成できる。

従来のシステムの欠点を解決する試みは他のシステムにも見られる。以下に， Jasmine と他システムとの差異を述べる。

① 第２世代ＡＩ言語として KEE〔7〕，ART，LOOPS〔8〕，World Model 〔9〕等がある。 KEEには外部データアクセス機能が付加されたが，本システムは(1)，(5)，(6)の点で異なっている。 ART， LOOPS，MINERVAについても同様である。

② 関数型データベースとして，Iris〔11〕，GENESIS 〔12〕等がある。本システムは(3)，(4)，(6)の点で異なっている。さらに，本システムは非正規形テーブルにデータを格納するが，Irisは正規形テーブル，GENESIS はネットワークモデルである。

③ セマンティックデータベースとして，Taxis〔13〕，SIM〔14〕等がある。本システムは(6)の点で異なっている。また，SIMとは(2)，Taxisとは(8)の点で異なっている。さらに，Taxis とSIM では主記憶での高速化が考慮されていない。

④ 拡張関係データベースとして，EXODUS〔15〕，POSTGRES〔16〕がある。本システムは(6)と(8)の点で異なっている。また，EXODUSとPOSTGRESではユーザ定義関数とシステム定義コマンドを区別しているのに対し，Jasmine では同一のシンタクスで操作できる。

⑤ オブジェクト指向データベースとして，ORION〔17〕，GemStone等がある。本システムは(5)と(6)の点で異なっている。また，ORION と GemStone は集合とクラスを区別するのでシンタクスが複雑である〔1〕〔2〕。

⑥ オブジェクト指向言語として，Objective-C〔18〕，C++〔19〕，Smalltalk〔20〕，Hypertalk〔21〕 等 がある。本システムは，(1)，(3)，(4)，(5)，(6)，(7)の点で異なっている。

本稿では，第2章で設計思想，第3章で Jasmine におけるオブジェクトの概念と構成，第4章で Jasmine のシステム構成，第5章でオブジェクトの構造，第6章でオブジェクトへのアクセス方式，第7章でマルチメディアの実現技術について述べる。

2. 設計思想

Jasmine の設計に当たっては次の3点を考慮した。

(1) 単純性

Jasmine の基本操作単位はオブジェクトである。インタフェースはすべてのオブジェクトについて共通であり，単純である〔1〕。システムもオブジェクトとして参照，操作できる。また，クラス（メタ情報）も同様に操作できる。

また，Smalltalkや LOOPSと異なりメタクラスという概念はなく，オブジェクトはクラスとインスタンスの2階層のみである。よって，簡単にオブジェクト階層を理解でき，ユーザはクラスの定義，インスタンスの生成・操作を簡単に実行できる。

(2) 拡張性

Jasmine は既存オブジェクトの下位クラスを定義すれば応用を簡単に作成できる。同様に，システム拡張も簡単に行なえる。これはシステムもオブジェクトで形成しているからである。

第7章で説明するイメージ等のマルチメディアも同様にして拡張したものである。さらに，メディアの拡張も簡単に行なえる。

(3) 高速（効率）性

大容量かつ大量のデータを効率よく管理するために，非正規形テーブル〔5〕を利用した格納方式を採用した。これにより，格納効率とアクセス効率を向上できる。さらに，プリプロセッサによる最適化とコンパイル方式の採用により処理速度の高速化を図っている。また，2次記憶の上に Active Object Space を設けて，主記憶と2次記憶とのインタラクションを少なくし，処理速度の高速化を図っている。

3. Jasmine におけるオブジェクトの概念と構成

(1) オブジェクトの基本概念

オブジェクトにはクラスとインスタンスの2種類がある。クラスとはインスタンスの集合であり，インスタンスの共通の情報を持つ。例えば，人間の集合はクラス”人間”であり，鈴木さんや田中さんは人間に属するインタンスである。

オブジェクトとオブジェクトの間に関数を定義することができる。定義できる関数には2つの種類の関数がある。一つはインスタンスとインスタンスを対応させて結び付ける関数であり，静的（数え上げ）属性と呼ぶ。2つめはプログラムで関数を定義するもので，動的（手続き）属性と呼ぶ。

同じクラスに属するインスタンスはクラスに記述された属性を共通に持つ。また，クラス間に上位下位関係が定義できる。下位のクラスに属するインスタンスは上位クラスの属性を継承（遺伝）する。さらにクラスをインスタンスの集合とみる時に，上位クラスに下位クラスは包含される。

Jasmine のオブジェクトはすべて上記の性質を持ち，システムもオブジェクトで記述されている。そのことについて次に説明する。

(2) カーネル・オブジェクト

図1に Jasmine のカーネル・オブジェクトの階層を示す。

OBJECT はオブジェクト階層の最上位オブジェクトである。

オブジェクトは，自己参照型(IMMEDIATE) と参照型(REFRENCE)の2種類に分類できる。前者にはC言語で扱うデータタイプのほとんどが含まれる。さらに，可変長データやタプル(TUPLE) が追加されている。タプルとは非正規形テーブルの入れ子テーブルのタプルの構造や手続きを記述するクラスである。〔1〕

COMPOSITE は参照型のユーザクラスの最上位クラスである。

MEDIA はマルチメディア用のクラスであり，下位クラスとして第7章で説明するグラフィックス，イメージ，ウィンドゥがある。

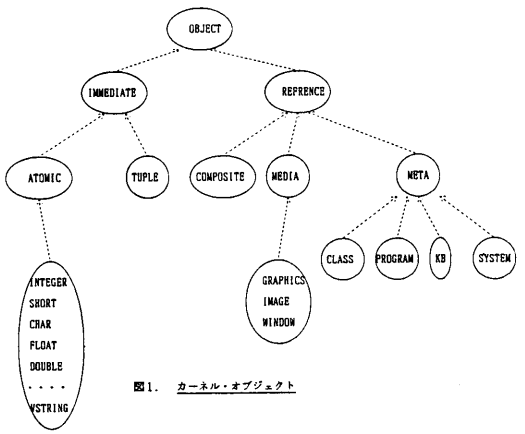

図1. カーネル・オブジェクト

METAはシステムクラスの上位クラスであり，システムクラスには，CLASS，PROGRAM，KB，SYSTEM がある。

CLASS はクラスを管理するクラスであり，すべてのクラスはこのクラスに属するので，インスタンスの操作と同じようにクラスを操作できる。PROGRAM は手続き属性やデモンやユーザプログラムを管理するクラスであり，コンパイル，リンク，関数アドレス検索等の手続きを持つ。KB は知識ベースファイルを管理するするクラスであり，生成，削除，オープン，クローズ等の手続きを持つ。SYSTEM はシステム全体を管理するクラスであり，セッションのオープン，クローズ，トランザクションの開始，終了等の手続きを持つ。

知識ベースのオープン時に，知識ベースに属するクラスを管理するクラスが検索され，CLASS の下位クラスとなる。このようにして，すべてのクラスは CLASS のインスタンスとなる。

4. システム構成

図2にシステム構成を示す。

最下層のデータ管理部は拡張関係データベースであり，非正規形のテーブルを管理できる。また，インデック検索やハッシュジョインという高速アクセスメソッドを持つ。さらに，タプル毎の処理を述語関数やマッピング関数というコンパイルコードで与え実行できる高速かつ汎用的なデータベースシステムエンジンである。〔5〕

オブジェクト管理部は第3章で示したようなオブジェクトを管理する。また，Active Object Space によりオブジェクト管理部とデータ管理部とのインタラクションを少なくし，データ管理部とオブジェクト管理部のデータ構造をほぼ同じにして検索コストを最小にしている。さらに，高機能かつ高速な遺伝処理を提供する。また，自動領域管理機構も提供している。

実行管理部は，プリプロセッサ Jasmine/C や会話型処理システム Jasmine/I の実行制御を行う。

構文解析部では，Jasmine/C 言語で記述されたプログラムの構文を解析し，中間情報を作成した後，最適化を施して，選択演算ブロック，結合演算ブロック，動的結合演算ブロックを中間情報として作成する。

コード生成部では，最適化されたCソースを生成する。

インタプリタでは，中間情報をもとにオブジェクト操作を実行する。中間情報をもとに，逆ポーランド命令列を生成し，スタックに基づくオブジェクト操作仮想マシンを逐次操作する。

Jasmine はオブジェクトベースプログラミング言語 Jasmine/C のプリプロセッサと会話型システム Jasmine/I を提供する。Jasmine/C と Jasmine/I は同じシンタクスで解析部を共有する。Jasmine/C プリプロセッサ は Jasmine/Cソースを解析し，最適な Cソースをコード生成する。会話型システム Jasmine/I はオブジェクト操作文を解析し，インタプリタで実行する。ただし，手続き属性はバイナリコードをダイナミックリンクして実行する。また，Jasmine/C より Jasmine/I を呼び出すこともできる。

図3に Jasmine/C を用いたプログラムの作成の流れを示す。まず，エディタで Jasmine/C のソースを作成し，Jasmine/C プリプロセッサにかけて，C ソースを作る。これをC コンパイルし，出力されたバイナリファイルと Jasmine実行ライブラリをリンクして実行可能ファイルを作る。これをデバッガで実行し，必要ならば修正し，上記の処理を繰り返す。

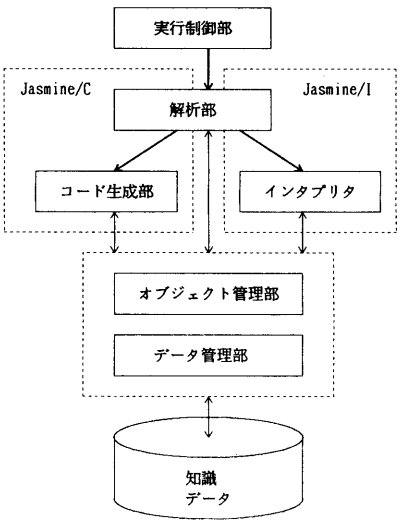

図2. Jasmine のシステム構成図

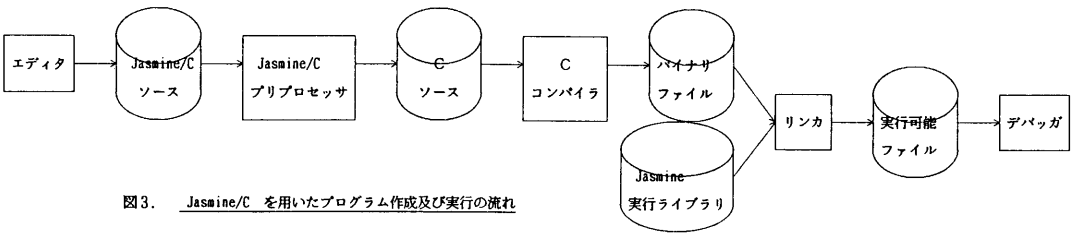

図3. Jasmine/C を用いたプログラム作成及び実行の流れ

5. オブジェクトの構造

5.1. オブジェクト格納用テーブルの種類

データ管理部で用意するテーブルは①シーケンシャル・テーブルと② B-tree テーブルの２種類である。 B-tree テーブルはインデックスに使われる。また，辞書等の格納テーブルとして使うこともできる。

5.2. オブジェクトの格納方式

データ管理部で提供するインナーテーブル(NF2) を利用して格納効率とアクセス効率の良い格納方式を採用した。

Jasmine では多値を扱える。従来の正規系テーブルで多値を扱おうとすると，タプル数が属性値の要素数の直積だけ必要となり，冗長であり，かつ，タプルが複数になるのでアクセス効率が悪い。インナーテーブルを使うことにより，格納効率とアクセス効率が最適になる。オブジェクトをテーブルに格納する時，①オブジェクト識別子，属性名，ファセット名，値の４つのフィールドからなるテーブルを用いる方法と②各属性をフィールドに対応させる方法を検討した。前者はテーブルの形が単純で，かつどのようなオブジェクトも格納できるが，同一属性に複数のファセットがある時，属性名が冗長になる欠点がある。また，条件検索する時，後者の選択演算が前者では結合演算になる。例えば，特定の属性値がある値であるようなオブジェクトを求める時，前者は後者に比べアクセスが遅い。よって， Jasmine では後者を採用した。後者の中で，①遺伝を反映させない方式と②遺伝を反映させる方式について検討した。前者は各クラスの固有の属性のみをフィールドに対応させるのに対し，後者は属性に遺伝された属性を含める。前者では，１つのオブジェクトが複数のタプルに分かれて存在するので，１つに結合する時にクラスタリングされていないのでアクセス効率が悪い。また，オブジェクト識別子が冗長である。よって， Jasmine では後者を採用した。

以上のように格納効率とアクセス効率を検討した結果，Jasmine では格納テーブルとしてインナーテーブルを採用し，各属性をテーブルのフィールドに対応させ，さらに属性には遺伝された属性を含め，１つのオブジェクトを１つのタプルとして格納する。図４にインスタンスの格納例を示す。

患者テーブル

oid	名前	体重	身長	年齢	趣味
3:36:1	田中	NIL	172.0	28	読書 旅行

図４ インスタンスの格納例

我々はクラスもインスタンスとして扱うので，同様な規則でテーブルに格納する。図５にクラスの属性関係，図６にクラスの格納テーブルの例を示す。

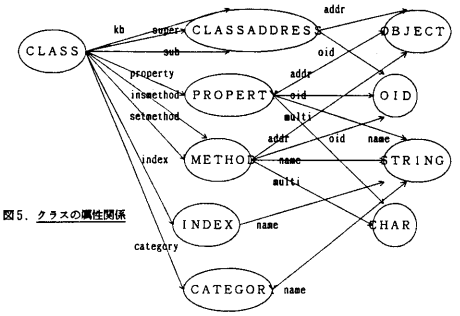

図５．クラスの属性関係

クラステーブル

クラス名	kbaddr	kboid	super		property			
			addr	oid	name	oid	addr	multi
人間	xxx	0:23:1	yyy	2:36:0	名前	2:25:0	zzz	'n'
					体重	2:24:0	aaa	'n'

図６．クラスの格納例

さらに，各テーブルにはオブジェクト識別子によるインデックスを設け，アクセスの高速化を図っている。また，クラスや知識ベースのテーブルには名前によるインデックスも設けている。

5.3. Active Object Space 上でのオブジェクト管理

オブジェクト管理部とデータ管理部とのインタラクションを少なくすることが重要である。そのために， Jasmine ではデータ管理部のページバッファの上に Active Object Space を設けた（図７）。オブジェクト操作時にはオブジェクトをデータベースから検索し，Active Object Space にデータ構造として蓄える。以後はそのアドレスをもとに直接アクセスすることにより，アクセス効率を良くした。さらに，Active Object Space 上のオブジェクトはオブジェクト識別子によるハッシュ表に登録され，オブジェクト識別子によるアクセスも高速になる。オブジェクト識別子によるアクセスでは先ずハッシュ表を探索し，なければ，各格納テーブルに張られている識別子によるインデックステーブルを利用して高速にタプルを求め，そのタプルをファイルから取り出し Active Object Space にコピーしてハッシュ表に登録する。さらに，タプル識別子（データベース中のタプルの位置を示す）を管理して，タプルに高速にアクセスできる。属性値を更新した時に属性に対応するビットを立てて属性更新情報を管理することにより，更新した属性のみを更新する。

また，Active Object Space でのオブジェクトのデータ構造とデータベースでのデータ構造をほとんど同じにすることにより，データ管理部のページバッファから Active Object Space への

一括コピーを可能にして，オブジェクトの検索や挿入の処理速度を高速にした。データ構造の違いは，データ管理部ではディスクアクセスの効率化のために連続領域にデータがあり可変長データについてはそのオフセットを持つようなデータ構造になるのにに対し，主記憶上では連続領域にある必要はなく，アクセス効率を上げるためにオフセットではなく，直接のポインタを持つ。データ管理部からオブジェクト管理部へのデータ検索では，データ管理部のページバッファ上のデータを一括コピーし，オフセットを絶対アドレスに直すだけである。この方式はデータ管理部のオブジェクトを Active Object Space にコピーする方式としては非常に高速なものと言える。

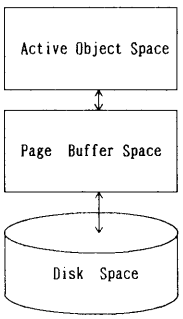

図7. Active Object Space

5.4. 自動領域解放

Jasmine では領域が不足した時に，オブジェクトの更新内容をデータベースに反映し，その領域を解放するという自動領域解放機構がある。

C++ や Objective-C には自動領域開放機構はない。また，Smalltalk ではインスタンス毎にリファレンスカウントを取り，World Model ではインスタンス毎に LRU(Least Recently Used) リストで管理する。インスタンス毎に管理すると，各インスタンス操作毎に自動領域開放に関連する処理を行うのでオーバーヘットとなる。それに対し，Jasmine ではクラス単位にＬＲＵリストを管理する。クラス単位で，クラスに属するインスタンスをデータベースへ反映するので，クラスタンリングされてデータベースアクセスを軽減できる。また，ＬＲＵリストの管理もデータベース検索という重い処理の中の行われるので，リストメインテナンスは十分無視できる。このように Jasmine の提供する自動領域開放機構は，個々のインスタンス操作時のオーバーヘッドはほとんどなく，かつ領域開放が高速である。

6. オブジェクトへのアクセス方式

6.1. 属性アクセスと遺伝

従来のオブジェクト指向システムでは，属性の遺伝機構の機能と速度に幾つかの問題点があった。Jasmine では以下のように解決している。

(1) ファセット遺伝

Jasmine はフレームの概念であるファセット（デフォルト値やデモン）を導入している。従来のオブジェクト指向システムでこのファセットを導入している場合において，ファセットに関する遺伝が不十分であった。Jasmine では，この問題を解決し，ファセットまで含めた遺伝機構を提供する。従来のシステム〔7〕〔9〕では，属性のある所で遺伝探索を止めていたが，Jasmine では最後までオーバーライドしながら遺伝探索を進めていく。即ち，遺伝探索においてすべてのファセットが満たされず，かつ上位クラスが存在する時は，遺伝探索を続ける。これにより，利用者は遺伝をファセットまで制御できる。この処理自体はコストが大きいが，以下に説明する高速化により，総合的には遺伝処理は高速になる。よって，Jasmine の遺伝処理は高速かつ高機能である。

(2) 静的結合による属性操作の高速化

従来のオブジェクト指向システム〔7〕〔8〕〔9〕〔20〕では遺伝のための検索処理を実行時に行う。例えば，ある患者の体重を求める時，そのインスタンスの体重属性を探し，あればその属性値をとる。その値が未定義値ならばデファルト値やデモン等を探索して実行する。これら一連の操作が実行時に行われる。手続き属性においても，その属性に対応する手続きを探索して実行する。

本システムではプリプロセス時に遺伝処理を解決して，実行時のオーバーヘッドを無くした。例えば，患者の体重を検索する時はプリプロセス時に求めた，属性値の格納場所の相対アドレスで参照するコードを生成し，デフォルト値やデモンもプログラム中に展開することにより，実行時には遺伝処理を行わない。また，手続きについては対応する関数の呼び出しをコード生成することにより，実行時には遺伝検索を行わない。

また，デモン制御文やデモン制御パラメータを予め解析するので，インタプリタのように実行時にデモン起動の有無等の判定をする必要はない。

さらに，ポリモルフィズム〔22〕については，インスタンスの属するクラスに応じて適用する処理を展開することにより解決した。現在は，if-then-else で展開しているが，オブジェクト識別子による switch 文の展開も検討している。

(3) ハッシュ表による遺伝情報への高速アクセス

Jasmine のオブジェクト管理部では，(1)で求めた遺伝情報をハッシュ表に登録して，遺伝探索は必ずハッシュ表を探索し，なければ，(1)の動作を行う。高機能で処理時間のかかる遺伝処理を(1)で行うが，１度探索を行えば後はハッシュ表の探索のみなので高速に探索できる。時間のかかる(1)の遺伝探索処理は１回のみなのでコストが小さい。また，(2)の静的結合により遺伝処理をさらに高速にしているので，Jasmine の遺伝処理は高機能かつ高速

であると言える。

6.2. 条件検索

従来のオブジェクト指向言語〔7〕—〔9〕〔18〕—〔20〕ではオブジェクトの集合を条件検索する機能は無かった。Jasmineではその機能を追加するとともに，その機能を高くし，さらに高速にした。

6.2.1. あいまい検索

オブジェクト指向ではユーザはオブジェクトやその属性とオブジェクト階層を理解して操作する。しかし，その階層が複雑になった時に，すべてのオブジェクトの状況を把握していなくてもオブジェクトを検索できたら便利である。そのために，Jasmineではオブジェクト操作文で指定された属性を持つクラスを下位関係を辿り探索し，属性を持つクラスについてのみ指定した操作を実行する。また，指定したクラスの下位クラスの中でその属性を保持するクラスがない場合はエラーとなる。例えば図8において，免許証番号がabc　の患者を求めることは，免許証番号を属性として持つクラスは大人のみなので，免許証番号がabc　の大人を求めることと等価となる。

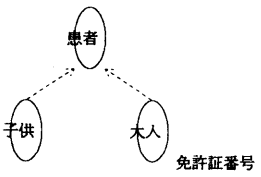

図8．あいまい検索の例

6.2.2. カテゴリ検索

カテゴリ属性を用いた探索テーブルの限定について説明する。

例えば，図9のように患者クラスが年齢により2つのクラスに分類でき，16才未満を小児，16才以上を大人とする時，35才以上の患者は，35才以上の大人の患者と等価であり，探索テーブルを大人の患者に限定できる。

このように，カテゴリ属性に関する分類条件と検索条件をマッチングすることにより，探索テーブルの範囲を限定できる。

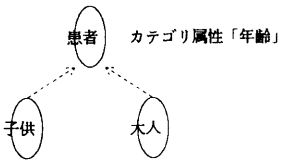

図9．カテゴリ検索の例

6.2.3. 動的ジョイン

従来の関係データベースでは，最適化処理をコマンド解析時にのみ行っていた。実行時でないと判らないような統計情報については予測を立てて，最適化していた。もし，この予測がはずれると，最適化処理とは言えなくなる。例えば，ある大きなテーブルを選択してできる中間結果テーブルと別のテーブルとのジョインを考える。アクセス前には大きなテーブルのページ数は分かるが選択後の中間結果テーブルのページ数は正確には分からない。従来のシステムではこのページ数を予測していたために，その予測がはずれると，最適なアクセスパスを選択しない場合があった。

Jasmine　では，このように実行時でないと判らないような統計情報を実行時に検索して，アクセスメソッドを選択するようにした。このことを動的ジョインを呼ぶ。動的ジョイン（条件が等号）では，インデックスの有無，ジョインテーブルのページ数に応じて①タプルサブステテューション，②ハッシュジョイン，③ネステッドループジョインの3つを選択する。選択コストは実際のデータベース処理の中では殆ど無視できる。

また，選択処理（条件が等号）において選択テーブルに条件で指定された属性によるインデックスが張られている時はインデックスを利用する。

6.2.4. 最適化

次の3種類の最適化を行う。

(1) 意味的最適化

モデルの意味に基づく最適化であり，6.2.2.で示したカテゴリ属性を利用した探索テーブルの限定があげられる。

(2) 経験的最適化

データベース処理の最適化で良く言われる一般原則であり，次の3つがあげられる。

① ジョインより選択を優先する。

② 射影を利用して不要なフィールドを処理の対象から外す。

③ 複雑なネットワークを辿るような条件検索において，ネットワークで重なるクラスにおいては，選択やジョイン処理の中間結果も重なる。この中間結果テーブルを有効に利用して，ジョインのタプル数を軽減できる。

(3) 物理的最適化

アクセスメソッドに関する最適化であり，前節の動的最適化があげられる。

7. マルチメディアの実現技術

Jasmine　ではイメージのような大規模かつ大量なデータをデータベースファイルに効率良く格納し，かつ，高速にアクセスできる。また，複合図形のような複雑なデータも非正規形テーブルを利用して，効率良く格納し，かつ，高速にアクセスできる。このように，マルチメディアデータを効率良くかつ高速に管理できる。さらに，それらのマルチメディアデータを複合して扱える。

また，表示されたオブジェクトはすべてオブジェクトであり，

直接操作できる。

7.1. 表示オブジェクト

グラフィックスやイメージやテキストといったマルチメディアのウィンドゥ上の配置を考える時，それぞれの並び変えや移動／縮小して様子を見る，といった試行錯誤が繰り返される。この時，操作したことをすぐに見て確認できるユーザインタフェース(wysiwyg: what you see is what you get) が重要である。

このような概念をさらに拡張して，Jasmine ではウィンドゥ上に表示された全てのオブジェクト（表示オブジェクト）にメッセージを送り，直接操作できる。

ウィンドゥ上に表示されたオブジェクトの位置情報を管理し，マウス・クリック時の座標と照合することにより，オブジェクトを認識する。位置情報の登録は表示順にプッシュされていき，重ね合わせがある時はユーザに見えるオブジェクトが認識される。また，マウス・クリックとメッセージの対応は各ウィンドゥに指定できる。

7.2. 機能概略

Jasmine で提供するメディアクラスのグラフィックス，イメージ，ウィンドゥについて機能の概略を説明する。

7.2.1. 共通の手続き属性

Jasmine はポリモルフィズムをサポートするので，グラフィックスとイメージにおける同一種類の操作は同一インタフェースで操作できる。

(1) 表示関連の手続き属性

ウィンドゥ上で，表示，消去，再表示，移動，拡大／縮小ができる。

(2) 入出力関連の手続き属性

ファイルからの入力とファイルへの出力ができる。

(3) イベント処理関連の手続き属性

マウスのボタンアップ処理とボタンダウン処理を感知し，指定された処理を実行することができる。

7.2.2. 共通のデータ属性

次のようなデータ属性を持つことにより，表示オブジェクトを自動配置したり，直接操作することができる。

(1) 表示領域属性

表示オブジェクトの二次元空間での最大座標と最小座標を管理することにより，そのオブジェクトを表示するため必要な領域を管理できる。

(2) センス領域属性

マウス・クリックに対応して，オブジェクトを認識するための領域である。ユーザが指定しなければ，表示領域と同じになる。この属性は適当な領域を複数個指定できるので，例えば，家の図形の部品図形としてベルの図形を用意し，ベルの領域のみを指定し，イベントを登録すれば，その領域をマウスクリックして，登録手続きを実行することにより，ベルを鳴らす等のような手続きを実行できる。

7.2.3. グラフィックス

基本図形として，直線，連続線，スプライン曲線，円，矩形，多角形，文字列等があり，これらの基本図形を組合せて，複合図形を作れ，また，マルチメディアを組合せた複合オブジェクトも作ることができる。線には色と線属性（破線等）を指定でき，面図形には色，線，面属性（格子形パターン等）を指定できる。

各基本図形は属性として，①座標点列，②長さ列，③角度列，④基本色，⑤図形オブジェクト，⑥入出力フォーマットを持つ。

ユーザの記述した図形情報を基本図形情報に変換することにより，格納効率，アクセス効率，認識効率を向上させた。

7.2.4. イメージ

モノクロ，カラーの２種類の画像を扱える。カラーでは RGBの三種類の情報を管理する。入力情報の解像度と階調はスキャナに依存し，表示情報の階調はディスプレイに依存する。現在使用しているスキャナの解像度は 16 dot/mmであり，階調は 8bit である。ディスプレイでは同時 256色表示可能であるが，システム定義の色があるので，各色 4階調の計 64 色とした。

スキャナから入力する時，解像度と入力位置と入力範囲を指定できる。また，別のスキャナを接続する場合は新しいクラスとして登録し必要な属性を用意すれば，使用できる。さらに，SIDBA フォーマットファイルとの変換ユーティリティを持つ。

7.2.5. ウィンドゥ

Jasmine は Sun上で開発しているので，SunView を十分に使用できるようにクラスを用意してある。

手続き属性として，ウィンドゥの生成，消去，オープン，クローズ，移動，拡大/ 縮小，メインループ等が用意されている。

7.1.節で説明したように，ウィンドゥ上に表示されているオブジェクトの位置を管理し，さらに，イベント処理の情報も管理する。座標系として，①ウィンドゥハード座標系，②ウィンドゥ実座標系，③ウィンドゥ仮想座標系，④オブジェクト仮想座標系の４つがある。メディア・オブジェクトを定義する時には個々のオブジェクト仮想座標系を意識するだけで良い。システムが自動的に座標変換する。

8. おわりに

本稿では，オブジェクト指向型マルチメディア知識ベースシステム Jasmine の設計思想とその実現技術について報告した。

Jasmine は単純で，かつ，拡張性に富む。また，大量かつ複雑なデータ，知識，マルチメディアデータを，高速に，かつ効率良く格納・管理できる。また，他のシステムより多くの面で優れたシステムである。ユーザは Jasmine を使用することにより，大規模な知識を効率良く処理する応用を簡単に作成できる。

現在，Jasmine のインプリメントを終了し，性能評価と改良を行っている。

今後は，システムの持つカーネルクラスとその手続きを充実させる。例えば，動画や音響等のメディアの追加，メディア変換等

がある。また，マルチメディア・オブジェクトに基づきマンマシン・インタフェースを強化する予定である。

謝辞

本研究の一部は通産省工業技術院大型プロジェクト「電子計算機相互運用データベースシステムの研究開発」の一貫として行われた。

参考文献

1) 牧之内 他: オブジェクトベースプログラムミング言語 Jasmine/C, データベース・システム研究会, データ工学研究会共催研究会 平成元年 7月

2) Ishikawa, H et al: Object-Oriented Multimedia Knowledge Base Manngement System: Design and Implementation, Proc. 2nd Intl. Sympodium on Interoperable Information System(INTAP) (Tokyo, JAPAN, 1988)

3) 石川 他: マルチメディアデータベースに対するオブジェクト指向アプローチについて. 情処全国大会(昭和61年前期)

4) 鈴木 他: オブジェクト指向型マルチメディア知識ベース Jasmine のプログラムインタフェースの実現について. 情処全国大会(昭和63年後期)

5) Makinouchi, A.: A consideration on normal form of not-necessarily-normalized relation in the relational data model. Proc. Intl. Conf. VLDB (October 1977, Tokyo)

6) Yamane, Y. et al: Design and evaluation of a high-speed extended relational database engine, XRDB. Intl. Symodium on Database Systems for Advanced Applications (Seoul, Korea, April 1989)

7) Bobrow, D. G and Stefik, M: The LOOPS Manual. Tech. Report KB-VLSI-81-13, Knowledge Systems Area, Xerox Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, Clif. 94304, (1981)

8) IntelliCorp: IntelliCorop KEE Software Development System Training Manual Version 2.0b-1.3(Zetalisp) 1985

9) Ishikawa, H. et al: KID Designing A Knowledge-Based Natural Language Interface. IEEE EXPERT (summer 1987) 57-70

10) Date, C. J: An Introduction to Database Systems, Addison-Wesley, Reading, Mass., (1975)

11) Fishman, D. H et al: Iris: An Object-Oriented Database Management System. ACM Trans. Office Inf. Synst. 5, 1(Jan 1987) 48-69

12) Batory, D. S, et al: Implementation Concepts for an Extensible Data Model and Data Language. ACM Trans. Data base Syst. 13, 3(Sept. 1988) 231-262

13) Nixon, B. et al: Implementation of a Compile for a Semantic Data Model: Experiences with Taxis. Proc. 1987 ACM SIGMOD(1987) 118-131

14) Jagannathan, D. et al: SIM: A Database System Based on the Semantic Model. Proc. 1988 ACM SIGMOD(1988) 46-55

15) Carey, M. J: A Data Model and Query Language for EXODUS. Proc. 1988 ACM SIGMOD (1988) 413-423

16) Stonebraker, M and Roe, L. A: THE DESIGN OF POSTGRES Proc. 1987 ACM SIGMOD (1987) 340-355

17) Banerjee, J et al: Data Model Issues for Object-Oriented Applications. ACM Trans. Office Inf. Syst. 5, 1 (Jan. 1987) 3-26

18) Brad J. Cox: Object Oriented Programming. Addison Wesley 1986

19) Bjarne Stroustrup: THE C++ PROGRAMMING LANGUAGE(邦訳) トッパン 1988年11月

20) 鈴木則久: Smalltalk

21) 西林瑞夫: Macintosh 増補版 Hyper Card 付 共立出版 1988年10月

22) Sefik, M and D. G. Bobrow: Object-Oriented Programing: Themes and Variations. AI MAGAZINE, vol. 6, no. 4, 1986