7M-3　オブジェクト指向ＤＢＭＳ－Ｏｄｉｎ　オブジェクト操作言語

木村　裕　　鶴岡　邦敏

（日本電気㈱Ｃ＆Ｃシステム研究所）

1．はじめに

筆者らはオブジェクト指向データベースの研究の一環として、そのＤＢＭＳ－Odinの開発を行っている。既に、概念オブジェクトと物理ファイルを独立させ、任意のファイル／ＤＢを参照可能とする物理構造仮想化方式を提案した[KiTs89,Tsur89]。今回は、概念オブジェクトを定義・操作するためのオブジェクト操作言語の特徴について報告する。

従来のプログラミング言語に比べ、オブジェクト指向データベースにおける言語の特徴は、クラス定義、メッセージ送信、集合検索／更新操作、IS-A、PART-OF等の機能を備えている点である[Bakk88,KBBC88]。Odinでは、これらの機能以外に射影、別名、結合、グループ化の機能を持つ。これらの機能は一種のビューの役割を果たす。ただし、Odinのビューは、関係データベースでのビューと異なり、元のオブジェクトを直接参照する方式のため、ビュー更新に伴う問題を容易に解決できる。射影、別名は、検索後のオブジェクトのメソッドおよびインスタンス変数に対するアクセスを制限する（3節）。また、結合／グループ化操作により、複数のクラス／インスタンスを１つのクラス／インスタンスとして扱うことが可能となる（4節）。

以下では、2節でOdinの言語の概要を述べ、3,4節でオブジェクトの定義と操作の仕様を記述する。

2．概要

Odinの言語は、CLOS(Common Lisp Object System)とFlavorに似た構文を持つ。メソッド送信は、

(sendb <selector> <object> {<parameter>})

の構文を持つ。<selector>はメソッド名のシンボルである。一方、ＤＢ操作用のクラス・メソッドの送信は、sendbを使用しない構文で記述する。

以下の説明では、次のようなＤＢを仮定する。

```
class                Person
instance variable    name   (string)
   (domain)          city   (string)

class                City
instance variable    city   (string)
                     (domain)        mayor (Person)
```

3．オブジェクト定義

クラスは、define-classマクロで定義する。構文は下記の通り：

```
(define-class <class> ({<superclass>})
    <class-option> ({<instance-variable>}))
```

ここで、<class-option>は説明文や生存期間指定（一時的か永久的か）、従属指定（このクラスのインスタンスが他のクラスのインスタンスに従属しているか否か）、種別指定（[KiTs89]で述べたファイル・クラス等のクラス種別）等を記述する。<instance-variable>は、インスタンス変数を定義する。変数の定義では、その変数のドメイン、割当方法（個別か共有か）、初期値、等を指定できる。たとえば、クラスCityの定義は、

```
(define-class City ()
     :document "city class definition"
     ((city  :type string)
      (mayor :type Person) ))
```

となる。クラス定義の結果、クラス・オブジェクトと集合オブジェクトが生成される。図1にクラスの基本構造を示す。集合オブジェクトは、クラス

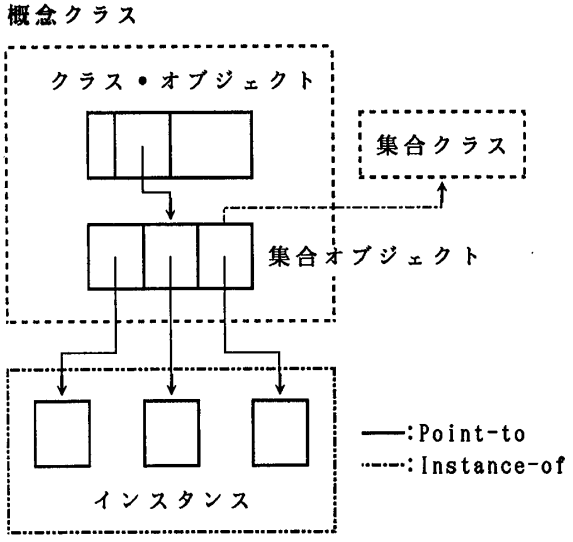

図１　概念クラスの構成

Object Manipulation Language for an
Object-Oriented Database Management System - Odin
Yutaka KIMURA, Kunitoshi TSURUOKA
NEC Corporation

から生成されたインスタンスを要素として持つオブジェクトであり、集合クラスのインスタンスである。

インスタンスの定義（生成）は、

(define-instance <class> {<init>})

<init> ::= '<variable> <object> | empty

のように、クラスへdefine-instanceメッセージを送ることでインスタンスが作られる。

4. オブジェクト操作

オブジェクト操作用メソッドとして、select,join,group-byを提供する。

Select（選択）の構文は次の通り：

(select <set-object> '(<object-variable>)
'(<condition>) '(<visible-method>))
<visible-method> ::= (<method-name <alias>)
{(<method-name <alias>)}

selectメッセージが集合オブジェクト<set-object>へ送信され、selectメソッドが起動される。このメソッドは、<set-object>の各要素オブジェクトに対して、<condition>に指定された式を適用し、この条件を満足する要素を含む集合オブジェクトを返す。<visible-method>により、結果オブジェクトのメソッドの参照を制限したり、または別名による参照に制限したりできる。

Join（結合）の構文は次の通り：

(join '(<join-terms>) '(<condition>))
<join-terms> ::= <join-term> <join-term>
{<join-term>}
<join-term> ::= (<set-object>
{<object-variable>}

結合メソッドは、<join-terms>の<set-object>で指された要素オブジェクトの中で<condition>を満足するオブジェクトを１つのクラスとして結合する。Joinメソッドは、複数のオブジェクトを結合するために、新しいインスタンス（結合インスタンス）を生成する（図2）。

Group-by（グループ化）の構文は次の通り：

(group-by <set-object> '(<group-unit>))
<group-unit> ::= {<instance-methods>}

<set-object>の要素を<group-unit>のメソッドが返す値ごとにグループに分割する。Group-byメソッドは、複数のインスタンスを分類するために、集合オブジェクトを生成する（図3）。

5. おわりに

本稿では、オブジェクト指向ＤＢＭＳ－Odinの言語の特徴的な部分について記述した。この言語の操作select,join,group-byは、一種の一時的なビューを作成しているとみられる。今後、ＤＢＭＳの機能として不可欠なビューを考えていく上で、この操作の性質をより明確にしていく予定である。

参考文献

[BaKK88] J.Banerjee,et.al."Queries in an Object-Oriented Databases," DE'88.

[KBBC88] W.Kim,et.al."Integrating an Object-Oriented Programming System with a Database System," OOPSLA'88.

[KiTs89] 木村,鶴岡 "ｵﾌﾞｼﾞｪｸﾄ指向ﾃﾞｰﾀﾍﾞｰｽ管理ｼｽﾃﾑの柔軟性について," 第38回全国大会 1989.

[Tsur89] 鶴岡 "ｵﾌﾞｼﾞｪｸﾄ指向ﾃﾞｰﾀﾍﾞｰｽ管理ｼｽﾃﾑの物理構造仮想化方式" ﾃﾞｰﾀﾍﾞｰｽｼｽﾃﾑ研究会，1989/9月.

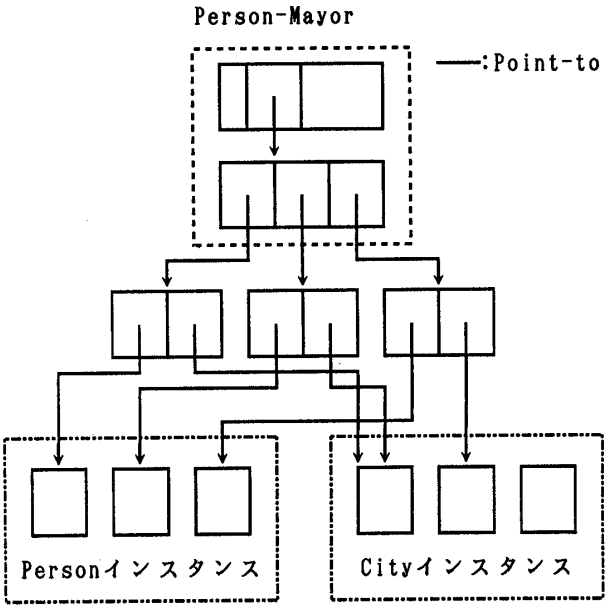

図２　結合後のインスタンスの関連

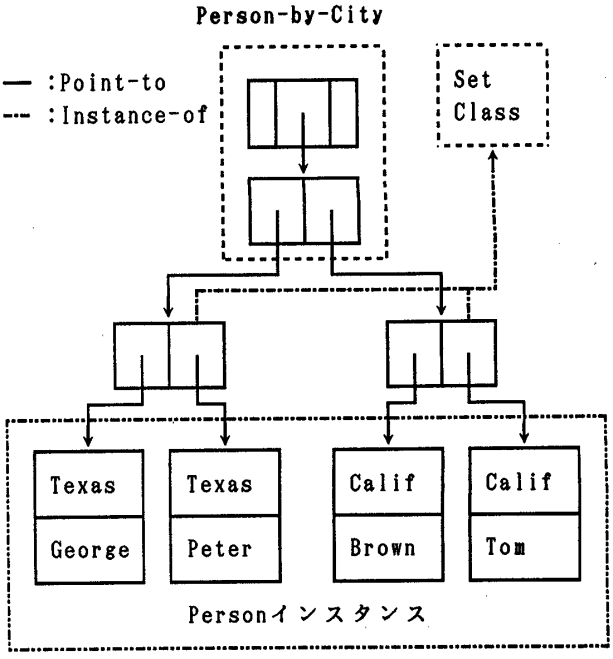

図３　グループ化後のインスタンスの関連