オブジェクト指向データベースシステムにおけるオブジェクト移動の実現法*

2R−6　Mohamed Elsharkawi　彭智勇　上林弥彦†

京都高度技術所　京都大学工学部‡

1 まえがき

近年, CAD、CIM、ネットワーク管理、マルチメディア、地理情報システム(GIS)、文書管理などの分野で、データベース利用の高度化に伴い、複雑なデータに対する、より効率的な管理の必要性が認識されている。オブジェクト指向データベース管理システムは関係データベースと比較して、強力なデータモデリング能力などの機能を提供できる特色がある。しかし、ONTOSなどの商用化されているオブジェクト指向データベースシステムでは オブジェクトが異なるクラスに移るという問題を直接扱うことはできない。このようなオブジェクト移動はスキーマ変更よりもよく起こるため、その解決へ向けた研究が急務となっている。

この問題を解決するためには、オブジェクト指向データベースにおけるオブジェクト間の関連を モデル化し、オブジェクト移動に伴う起こる影響の処理機構を 備えるべきである。本稿では オブジェクト移動についての問題を分析し、モデルを定義し、このモデルに基づいたオブジェクト移動の実現法について検討する。

2 オブジェクト移動についての問題

オブジェクト指向データベースにおけるオブジェクトには、各種の複雑な制約が存在する。オブジェクトが一つだけのクラスに属するという制約は 基本的な制約の一つである。そのほかに オブジェクト間の参照関連も非常に重要な制約であると思われている。参照関連によってデータ値の更新波及があり、オブジェクト移動が波及する可能性もある。文献(1)では これらの問題を分析した上で、解決方法を提案した。実際のシステムに応用するために、この方法は以下の三つの方向に拡張する必要があると考えている。

(1) オブジェクト移動範囲の拡大

オブジェクトの属性の値域はそれが属しているクラスによって定義されている。オブジェクト中のある属性が変化すると 元のクラスでは この属性値が定義できなくなる場合がある。このオブジェクトを再定義するためにクラス階層で適当なクラスを探さなければならない。適当なクラスがあるなら、このオブジェクトは元のクラスから新しいクラスまで移動して、対応する属性が定義できることになる。文献(1)に述べた方法には新しいオブジェクトの属性は完全に定義できないクラスがシステムに存在しないとき、オブジェクト移動を拒否する。したがって、オブジェクト移動範囲があまり広くない。本稿ではクラス間の多重継承関係を充分に利用して、なるべく小さい探索空間で適当なクラスをできるだけ探すというアルゴリズムを設計した。このクラスは 新しいオブジェクトの属性をできるだけ多く定義するようなものである。完全に定義できない場合にはユーザを介して、この見つけたクラスのサブクラスとして新しいクラスを作る。このようにシステムに現存するクラスの属性を充分に利用することで、オブジェクト移動範囲が拡大できる。

(2) オブジェクト移動に対する更新波及の回避

オブジェクトが識別子によって他のオブジェクトに参照されていることがある。たとえば、ある子供は学生として親に参照されている。この子供が卒業してから、会社員になったとき、親のオブジェクト内の対応する属性値を学生から会社員に更新する必要がある。この問題に対して、新しいオブジェクトと旧いオブジェクトの識別子を交換させるあるいは新しいオブジェクトは旧いオブジェクトのバージョンとして保存させることによってこのような更新波及が避けられると考えられる。

(3) オブジェクト属性値間のルールによる移動波及

通常、オブジェクトには複数の属性がある。これらの属性は 属性値として 複数のオブジェクトを参照している。属性値の間に 各種のルールが存在している可能性がある。このルールによって ある属性値が変化すると、関連がある属性値も変化しなければならない。したがって、オブジェクトが移動すれば、ルールによって別のオブジェクトに対して移動波及の可能性がある。たとえば、 ある人は国籍と子供という属性があり、この二つの属性の間に 「父と子供の国籍が同じである」、「父が日本人でなければ子供が留学生である」というルールがあると仮定する。このルールによって、この人の国籍が日本からアメリカになれば、彼の子供が同時に日本人学生から留学生になるという移動波及が存在する。

3 データモデル

オブジェクト指向データベース中のデータをすべてオブジェクトとして扱う。オブジェクトは静的な性質である属性と動的な性質であるメソッドを持っている。一つのクラスには属性集合が同じであるオブジェクトが属している。クラスの間には多重継承機構による関連がある。

3.1 オブジェクト

オブジェクトでは 以下のような記号等が定義されている。

(1)OID: オブジェクトの識別子を表す。
(2)OCLASS():OCLASS(OID) は オブジェクト OID が属しているクラスを表す関数である。
(3)OATTRIi():OATTRIi(OID) は オブジェクト OID の i 番目属性の値を表す関数である。
(4)VERSION():VERSION(OID) はオブジェクト OID の新しいバージョンの識別子を表す関数である。

3.2 クラス

クラスでは 以下のような記号等が定義されている。

(1)CNAME: クラスの名前を表す。
(2)CSUPER():CSUPER(CNAME) はクラス CNAME のあらゆる直接スーパークラス集合を表す関数である。
(3)CSUB():CSUB(CNAME) はクラス CNAME のあらゆる直接サブクラス集合を表す関数である。
(4)CATTRIi():CATTRIi(CNAME) はクラス CNAME の i 番目属性の名前を表す関数である。

---

*The Realization Method on the Object Migration in the Object-Oriented Database

†Zhiyong PENG, Yahiko KAMBAYASHI
‡ASTEM, Faculty of Engineering Kyoto University

(5)CPREi():CPREi(CNAME) はクラス CNAME の i 番目属性の制約述語を表す関数である。
(6)CATTRI():CATTRI(CNAME) はクラス CNAME のあらゆる属性の名前集合を表す関数である。
(7)CMDS():CMDS(CNAME) はクラス CNAME のあらゆるメソッド集合を表す関数である。
(8)CTOC():CTOC(CNAME) はクラス CNAME に含まれるオブジェクトが移動できる目標クラス集合を表す関数である。
(9)CCATTi():CCATTi(CNAME) は クラス CNAME の i 番目属性を定義しているクラスを表す関数である。

3.3 継承

クラスは 自分の属性とメソッドを持つ以外に すべてのスーパークラスの属性とメソッドを継承することができる。クラスの属性とメソッドは インスタンスで使用されるだけでなく、すべてのサブクラスに継承される。継承によって発生する名前の衝突は自己のクラスのものを優先する。スーパークラスの宣言の順に継承の優先度を規定することで、多重継承での衝突を回避できる。また、スーパークラスとして宣言するクラスはすでに定義されているクラスでなければならない。

4 オブジェクト移動の実現方式

通常、オブジェクトが変化した場合、どのようなクラスまで移動させるか分からない。したがって、適したクラスを探す操作はデータベースのクラス階層を利用してなるべくデータベース中の現存のクラスを利用すべきである。同時に探索空間をできるだけ小さくするように行なうべきである。これは クラス間の多重継承関係を利用して、以下のアルゴリズムによって実現できる。見つけたクラスでは移動したいオブジェクトの属性値は完全には定義できない場合がある。この場合にはユーザを介して新しいクラスを作る必要がある。新しいクラスは見つけたクラスのサブクラスとして 定義され、現存のクラスの属性を充分に利用できる。

移動目標クラスが見つかると、オブジェクトができるだけほかのオブジェクトに影響しないように移動すべきである。オブジェクト間の関連はオブジェクトの識別子によって構築されている。そして、オブジェクトの識別子が変化しないと オブジェクト移動に対する更新波及を回避することができる。そのために、オブジェクト移動は まず目標クラスに対するオブジェクトをつくり、元のオブジェクト中の必要な属性値をこの作られたオブジェクトにコピーし、あるいは規則によって変換し、新しい属性値はユーザあるいはシステムによって設定し、最後に この二つのオブジェクトの識別子を交換するということによって実現する。最後に 元のオブジェクトはごみとして処理される。もう一つの方法では新しいオブジェクトを新しい バージョンとして元のオブジェクトも保存する。

もちろん、オブジェクトが移動してから、このオブジェクトを参照しているか参照されているオブジェクトおよびルールによって関連があるオブジェクトに影響を及ぼすかもしれない。そうすれば、これらのオブジェクトを連鎖的に移動させる。

オブジェクト属性についての変化は 属性増加と属性削除と属性変更に分類することができる。属性変更は属性削除と属性増加の合成とみなすことができる。だから、基本的な変化は属性増加と属性削除だけとなる。

一般の例として、オブジェクト O(A1,A2,...,An) は属性 $Ai,...,Aj \in \{A1,...,An\}$ を削除されて、属性 $Ak,...,Ah \notin \{A1,...,An\}$ を加えられてオブジェクト O'($\{A1,...,An\}$ -{ Ai,...,Aj}) ∪$\{Ak,...Ah\}$ になる。オブジェクト O は次のように移動させる。

(1) { Ai,...,Aj } が空集合であれば、CSET2= { OCLASS(O) } と設定して、(3) に行く。なければ、クラス集合 CSET1= { OCLASS(O), CCATTi(OCLASS(O)),...,CCATTj(OCLASS(O))} ∪{ C |$C \in CSUB(CCATTi(OCLASS(O)))$ ... あるいは $C \in CSUB(CCATTj(OCLASS(O)))$ しかも、OCLASS(O) は C の 直接あるいは間接サブクラスである } を生成する。

(2) クラス集合 CSET2={ C |$C \notin CSET1$, ある $C' \in CSET1$ が 存在し、C' に対して、$C \in CSUPER(C')$ } を生成する。CSET2 はクラス集合 CSET1 中のクラスの直接スーパークラス集合である。

(3) クラス集合 CSET3={ C |$C \notin CSET1$, どんな $C' \in CSET2$ でも、C' は C の直接あるいは間接スーパークラスあるいは C 自身である。しかも、$C \in CTOC(OCLASS(O))$ } を生成する。CSET3 はオブジェクト O の可能な移動目標クラスあるいはそのスーパークラス集合である。

(4) CSET3 が非空集合である場合には $C \in CSET3$ を選択する。どんな $C' \in CSET3$ でも、num(ATTRI(C)∪(({A1,...,An}−{Ai,...,Aj})∪{Ak,...,Ah}) − ATTRI (C)) ≤ $num(ATTRI(C')\cup((\{A1,...,An\}-\{Ai,...,Aj\})\cup\{Ak,...,Ah\}) - ATTRI(C'))$ という条件を成り立つ。もし $((\{A1,...,An\}-\{Ai,...,Aj\})\cup\{Ak,...,Ah\}) \subseteq ATTRI(C)$ であれば、C"=C と設定する。なければクラス C のサブクラスとして新しいクラス C"を作る、C"の中で属性 $ATTRI(C) \cup ((\{A1,...,An\}- \{ Ai,...,Aj\}) \cup\{Ak,...,Ah\}) - ATTRI$ (C)) を定義する。

CSET3 が空集合である場合には、新しいクラス C"を作る。どんなクラス $C' \in CSET2$ でも、$C" \in CSUB(C')$ である。C"の中で属性 $((\{A1,...,An\}-\{Ai,...,Aj\}) \cup\{Ak,...,Ah\}) - \{ATTRI(C')|C' \in CSET2\}$ を定義する。

(5) クラス C"のインスタンスとして、オブジェクト O' を作る。属性 { A1,...,An} -{ Ai,...,Aj} の値はオブジェクト O からオブジェクト O' にコピーして あるいは規則によって変換し、O' のほかの属性の値はシステムあるいはユーザが設定する。

(6) オブジェクト O と O' の識別子を交換して、オブジェクト O' はごみとして処理する。あるいは、オブジェクト O' を VERSION(O)=O' として保存する。

(7) オブジェクト O を移動してから、関連があるあらゆるオブジェクトを検査する。影響を受けたオブジェクトは 以上の方法により 連鎖的に 移動させる。

5 あとがき

本稿では オブジェクト指向データベースシステムにおけるオブジェクト移動の実現法について過去の成果をさらに一般化した成果について述べた。今後はこの実現法を詳細化するとともに、有効性について評価したい。

参考文献

(1) Mohamed E.El-Sharkawi, Yahiko Kambayashi: Object Migration Mechanisms to Support Updates in Object-Oriented Databases. Databases: Theory, Design, and Application, IEEE Computer Society Press, pp.73-91, 1991

(2) ONTOS Developer's Guide.