XML データベースを用いた オブジェクト指向データベースの試作

A Prototype Program of Object-Oriented Database using XML Database

谷 研究室 福澤 太
Tai Fukuzawa

概要

オブジェクト指向言語 Java を対象とする，XML データベースを用いた仮想的なオブジェクト指向データベースを試作する．

1 はじめに

今日のシステム開発においてデータベースといえば，関係データベースが主流だ．しかし，Java，C++をはじめとしたオブジェクト指向技術が台頭するにつれて，二次元の表による関係によってデータを表現する関係データベースが必ずしも最適とはいえない状況が発生している．

そこで関係データベースに代わるデータベースとして，オブジェクトをそのままにデータベースに格納できるオブジェクト指向データベースが注目されている．

本研究では，未だ発展段階のオブジェクト指向データベースに関して，XML データベースを用いた仮想的なオブジェクト指向データベースの提案をし，オブジェクト指向言語（オブジェクト指向なプログラミング言語）の一つ Java を対象として試作する．2 節ではオブジェクト指向について解説し，3 節では既存の技術であるオブジェクト指向言語と関係データベースの連携の問題点を紹介し，4 節ではオブジェクト指向データベースを解説し，5 節では XML を解説し，6 節では Java を解説し，7 節ではオブジェクト指向データベースの実現方法を解説する．

2 オブジェクト指向

本研究で用いるオブジェクト指向は，オブジェクト指向言語である Java がとるオブジェクト指向と同義である．

オブジェクト指向とは，以下に述べる抽象データ型，継承，オブジェクトの識別性の三つの概念を合わせたものである．オブジェクト指向により，対象をモノ (オブジェクト) としてとらえることができるようになり，より実世界の物事をそのままの形で表現できるようになる．

オブジェクト指向には本研究で取り上げるデータベースだけでなく，プログラミング言語，システム開発における要求定義から設計に至るまで多くの分野で応用が成されていると共に，オブジェクト指向という概念自体は Java がとるオブジェクト指向以外にも存在する．

2.1 抽象データ型

抽象データ型とは,「ユーザーがデータ型を定義可能」という意味である．また，定義されたデータ型のことをクラスまたはインターフェイスと呼ぶ．

2.1.1 データ型

データ型は，オブジェクトの集合がどのように表現されるかを記述する．データ型はオブジェクトを状態を表す属性（フィールド）と，そのオブジェクトに対する操作・振る舞い（メソッド）とを合わせたものである．

2.1.2 カプセル化

抽象データ型はオブジェクトの内部表現である属性を外部の世界から隠し，オブジェクトに対する外部からの干渉を，定義された振る舞いからのみ行うことができるようにし，オブジェクトを外部から保護する．これをカプセル化と言う．

2.1.3 インターフェイスとクラス

インターフェイスとは，オブジェクトが外部に公開する属性と振る舞いを定義したものである．

クラスとは，オブジェクトの外部に公開する属性と振る舞いだけでなく，公開しない属性と振る舞いも定義し，さらに各振る舞いの動作方法についても定義したものである．

2.1.4 インスタンス

抽象データ型に具体的な値を入れたものを実例 (インスタンス) と呼ぶ．インスタンスの取り扱いは，そのデータ型に付随する操作によって行われる．

2.2 継承

汎用性のある既存のクラスの属性と振る舞いを引き継いで，新たにクラス，インターフェイスを定義できるこ

とである．

2.3 オブジェクトの識別性

オブジェクトの識別性は，オブジェクトを他の全てのオブジェクトから区別する機能を持つことである．オブジェクトの識別性を用いることにより，オブジェクトは他のオブジェクトを格納したり，参照したりできる．また，全てのフィールドが同値のオブジェクトであったとしても，識別が可能である．

2.3.1 オブジェクトモデル

オブジェクトの識別性により，インスタンスをノードとする任意のグラフ構造を持つオブジェクトを，動的に構築できる．このグラフ構造をオブジェクトモデルと呼ぶ．

3 関係データベースとオブジェクト指向言語の連携

本研究で取り扱うデータベースはオブジェクト指向データベースである．しかし，現在データベースといえば，関係データベースを指していると言われるまでに，世にデータベースとして使われているのは関係データベースである．

そこで，以下にオブジェクト指向言語で作られたシステムと関係データベースを連携させていく上でおこる問題点を述べる．また，この問題の根本的な解決策として，オブジェクト指向データベースがある．

3.1 関係データベースの概要

関係データベースはリレーショナルデータモデルにもとづいて設計，開発されるデータベースである．

3.1.1 関係データモデル

データを表に似た構造で管理し，複数のデータ群が関係（リレーション）と呼ばれる構造で相互連結するしくみである．関係は組（表における行に相当し，一組のデータを表す），属性（表における列に相当する），定義域 (属性に入る値の範囲)，キーなどによって構成される．

3.2 インピーダンス・ミスマッチ

関係データベースとオブジェクト指向言語では，データの表現方法が異なる．この異なるデータ表現を連携する際の手間をインピーダンスミスマッチと呼ぶ．

オブジェクトモデルを関係データモデルの形に変える際の設計のための人的コストや変換の処理にかかるオーバーヘッドなどがインピーダンスミスマッチとしてあげられる．

関係データベースとオブジェクト指向言語の間にO/Rマッピングなどの技術を使うことで，以前よりインピーダンスミスマッチによって起こる手間は少なくなったが，インピーダスミスマッチは依然として関係データベースとオブジェクト指向言語の連携に対して大きな問題となっている．

4 オブジェクト指向データベース

4.1 オブジェクト指向データベースの概要

オブジェクト指向データベースはオブジェクト指向の概念をデータベースに持ち込んだモノである．

そこで，本研究で取り扱うオブジェクト指向データベースとは，

$$\text{オブジェクト指向データベース} = \text{オブジェクト指向} + \text{データベース機能} \quad (1)$$

とする．

4.2 オブジェクト指向データベースにおけるオブジェクト指向

4.2.1 抽象データ型と継承

オブジェクト指向データベースにおける抽象データ型と継承は，インスタンスを生成するオブジェクト指向言語のものをそのまま使用し， プログラム言語とデータベースの間のインピーダンスミスマッチを発生させない．

4.2.2 オブジェクトの識別性

オブジェクト指向データベースにおけるオブジェクトの識別性とは，オブジェクト指向言語によるプログラムと同様の識別性の機能を，データベース内であっても維持することである．

4.3 オブジェクト指向データベースにおけるデータベース機能

データベース機能とは，永続性，トランザクション，並行処理制御，障害回復，問い合せ，バージョン管理，整合性，セキュリティ，性能問題，以上などの機能に対応したものである．

4.3.1 永続性

オブジェクト指向データベースにおける永続性とは，オブジェクトが異なるプログラムの実行にわたって永続する機能である．オブジェクト指向データベース内に格納されたオブジェクトは，トランザクション，システム障害，メディア障害さえ越えて永続する．

4.3.2 トランザクション

トランザクションとはデータベースに対する仕事の単位である．トランザクションは ACID 特性を満たさなけ

ればならない.

4.3.3 並行処理制御

並行処理制御とは，トランザクションの直列可能な実行順序をつけることである．つまり，トランザクションの結果は同時に実行されたものであったとしても，(直列に) 次々と実行されたときの結果と同じにならなければならないことである.

4.3.4 障害回復

失敗したトランザクションの部分的結果や更新が，データベースに反映されないことを保証する機能である．回復すべき失敗とはトランザクションエラー，システムエラー，メディアエラーである.

また，障害が回復した後は失敗したトランザクションを正常に再開させる機能である.

4.3.5 問い合せ

問い合せは，オブジェクト指向言語などが，データベース内のオブジェクトの集合から部分集合であるサブオブジェクトに対するトランザクションなどを行うために，データベースと対話的にやりとりする機能である.

4.3.6 バージョン管理

オブジェクト指向データベースにおけるバージョン管理とは，オブジェクトの以前の状態にアクセスできるようにする機能である.

4.3.7 整合性

トランザクションにより，データベースのある一貫した (正しい) 状態を別の一貫した状態に変換する．データベースの一貫性を捉える述語は，整合性制約と呼ばれる．この整合性制約をデータベースに与えることが整合性の機能である.

オブジェクト指向データベースにおける整合性制約は，各クラス自身がカプセル化によりオブジェクトの整合性を管理をする.

4.3.8 セキュリティ

データベースの管理システムは，永続オブジェクトにアクセスしたり更新したりするために，セキュリティ機構を設けて，有害なアクセスから永続オブジェクトの保護をしなければならない．これがセキュリティ機能である.

4.3.9 性能問題

以上データベース機能を素早く動作するようにすることである.

4.4 ODMG 3.0

ODMG 3.0 は，ODMG(Object Data Management Group) によって定められた，オブジェクト指向データベースのための標準規格である.

4.4.1 ODL

ODL(Object Definition Language) は，ODMG が定めたオブジェクト指向データベース内に永続化されるオブジェクトの定義を記述するオブジェクト定義言語である.

4.4.2 OQL

OQL(Object Query Language) は，ODMG が定めたオブジェクト指向データベースに対する問い合わせ言語である．関係データベースための問い合わせ言語である SQL と非常によく似た構文規則となっている.

5 XML

XML(Extensible Markup Language) とは，インターネット上で交換可能な標準的なデータ記述方式を提供する仕様である．XML は SGML も元にして，W3C(World Wide Web Consortium) を中心に標準化作業が行われている.

5.1 SGML

SGML (Standard Generalized Markup Language) は，タグによるマーク付けによって，文書の論理構造，意味構造を記述する言語.

5.2 XML 1.0

1998 年 2 月 W3C によって標準化された XML の中心的な標準規格である.

XML はマークアップ言語であり，メタ言語である.

XML においては，SGML と違い妥当な XML 文章だけでなく，整形式 XML 文章も許される.

XML は XML 宣言，文書型宣言，XML インスタンスの三つの部分から成り立っている.

5.2.1 マークアップ言語

文書の一部をタグと呼ばれる特別な文字列で囲うことにより，文章の構造や，修飾情報を，文章中に記述していく記述言語.

XML 以外のマークアップ言語には，HTML，TEX などが存在する.

5.2.2 メタ言語

言語を定義するための言語．スキーマ言語とも呼ばれる.

XML においては，使用するタグ名，タグの階層構造，

タグで囲まれる内容データのデータ型を定義できる.

XML には DTD(Document Type Definition：文書型定義), XML Schema, RELAX NG などのメタ言語が存在する.

5.2.3 整形式 XML 文書

XML インスタンスが一つのルート要素から成り立っていること. 開始タグと終了タグで囲まれた要素と呼ばれるまとまりが, 入れ子構造になっていること. 要素名, 属性名などが名前の規則に従っていること. 以上の規則が守られた文書を整形式 XML 文書と呼び, 整形式 XML 文書でないものは XML でない.

5.2.4 妥当な XML 文書

整形式 XML 文書であり, かつ DTD が記述されており, XML インスタンスがその規則に従っている文書を, 妥当な XML 文書と呼ぶ.

5.3 DOM

DOM (Document Object Model) は, HTML と XML のための文書の論理構造と, プログラムから文書にアクセスし操作するため手段を定義し標準化した API. 様々なモジュールによって構成されており, 2007 年現在 Level 1 から Level 3 までの仕様が勧告された.

5.4 XPath

XPath (XML Path Language) は, XML インスタンスの特定の部分を指定する言語構文である.

1999 年 11 月 16 日に XPath 1.0 が, 2007 年 1 月 23 日に XPath 2.0 が W3C によって勧告された.

5.5 XQuery 1.0

XML に対する問い合わせ方法を定義した仕様.
2007 年 1 月 23 日に W3C によって勧告された.

5.5.1 FLWOR 表現式

XQuery の中の問い合わせ方法の一つ. For 句, Let 句, Where 句, Order by 句, Return 句によって構成される XML への問合せの為の関数型言語である.

- For 句：
  XPath 2.0 で指定したロケーションパスの展開結果である各ノードを繰り返し変数に格納し処理をする.

- Let 句：
  XPath 2.0 で指定したロケーションパスの単一ノードを変数に格納する.

- Where 句：
  For 句, Let 句で指定した変数に対して判定を行い TRUE となる場合のみ, 次の Return 句を実行する. 省略可能

- Order by 句 :
  Return 句を実行する順序を定義する. 省略可能.

- Return 句：
  結果として返す文書の生成規則を記述する.

5.5.2 ネイティブ XML データベース

XML を専門に扱うデータベースの事である. 関係データベースのデータ型として XML を取り扱うデータベースと区別するために, ネイティブ XML データベースと呼ばれている.

多数の製品が存在し多くの製品で XPath や XQuery による問い合わせが可能となっている. しかし, データベース内での XML の格納方法や, データベースへの XML の挿入や更新といった方法は標準化が行われていないため, 製品ごとに独自の方法となっている.

6 Java

Sun Microsystems 社が開発したオブジェクト指向なプログラミング言語.

Java で記述されたソースコードは, コンパイル時に Java バイトコードと呼ばれる中間コードに変換される. 実行時には Java 仮想マシン (JVM) と呼ばれるソフトウェアによって, 実行するプラットフォームに対応した形式 (ネイティブコード) に変換され, 実行される. プラットフォーム間の違いは JVM が吸収し, JVM 上で動作する Java プログラムは, プラットフォームの違いを意識しなくてもよい.

6.1 オブジェクト直列化

オブジェクト直列化は, オブジェクトとそこから参照されているオブジェクトを, バイトストリームにエンコーディングする. そして, そのストリームからオブジェクトグラフの補足的な再構築を行う.

6.2 JNI

Java Native Interface (JNI) は, Java ネイティブメソッドを書いたり, JVM をネイティブアプリケーションに組み込んだりするための標準プログラミングインタフェースです.

7 オブジェクト指向データベースの実現

7.1 システムの構成

クライアント側システム，サーバー側システム，XML データベースごとのドライバ，XML データベースの四つの部分によって，仮想的なオブジェクト指向データベースとなる．

- クライアントアプリケーション：
  オブジェクト指向データベースを使いたいアプリケーション部分．
- クライアント側システム：
  クライアントアプリケーションに対してオブジェクト指向データベースとしての窓口となる部分．
- サーバー側システム：
  クライアント側システムに対してネットワーク越しに処理を行うサーバー部分．
- XML データベースごとのドライバ :
  データベースごとに異なった仕様を吸収し，標準的な機能を提供するためのドライバ．サーバー側システムはこのドライバを通して，XML データベースとやりとりを行う．
- XML データベース：
  オブジェクトを XML に変換したものを格納する XML データベース．XQuery に対応した第二世代型 XML データベースであれば，XML データベースごとのドライバをプログラムすることにより対応できる．

7.2 XML への直列化と直列化復元

クライアント側システムにおいて，Java のオブジェクトを XML への直列化と，XML から Java のオブジェクトへの直列化復元を行う．

7.2.1 XML への直列化

Java のオブジェクトを，フィールドのデフォルト値がストリームに書き込まれない冗長性削減アルゴリズムを使用し，各インスタンスごとに分割した XML に変換し，インスタンスの参照関係を XML 上では id 属性と idref 属性による参照関係で構築する．

7.2.2 XML からの直列化復元

分割されている XML を直列に並べ直し，java.beans.XMLDecoder と同様の処理によりオブジェクトの復元を行う．

また，以前に直列化復元されていたインスタンスは，すでに復元されているインスタンスを用いることにより，二重の復元を避けた復元を行う．

7.3 外郭オブジェクト

オブジェクト指向データベースに格納されているオブジェクトを取り出す際には，一定の深さ以降の外郭オブジェクトとなれるインスタンスである XML が参照する XML は取り出さず，参照が全てそろっていないインスタンスを外郭オブジェクトとして，特別に扱う．

また，外郭オブジェクトは，全ての参照がそろった時点で正しいオブジェクトに復元される．

7.3.1 外郭オブジェクトとなれるインスタンス

外郭オブジェクトになるインスタンスは，コンストラクタによって復元されるインスタンスである．配列または，あるクラスのある静的メソッドによって復元されるインスタンスは，外郭オブジェクトにはならない．

7.3.2 外郭オブジェクトの実現

外郭オブジェクトの生成と一般オブジェクトへの復元は，JNI の機能を用いて実現する．

外郭オブジェクトの生成の際には，メモリーだけを確保したインスタンスを生成することによって実現する．

外郭オブジェクトから一般のオブジェクトへ復元する際には，JNI の機能であるコンストラクタを通常のメソッドのように呼び出すことによって実現する．

7.4 独自 OQL

ODMG が定めた OQL そのものはオブジェクト指向言語である C++を対象としたものであるため，Java に合わせて機能を限定した OQL のサブセットを使用して，本オブジェクト指向データベースはデータベースへの問い合わせ処理を行うことが出来る．

7.4.1 OQL から削除されている部分

struct が関わる文法は全て使えない．ただし，将来的には Object 型の二次元配列により実現予定である．

また，OQL 内からメソッドを呼び出すことはできない．ただし将来的には，プロパティーへの置き換えと，メソッドごとの特殊処理と，サーバー側における部分的なオブジェクトの復元によって実現予定である．

また，現在の所入れ子の OQL を実行することはできないが，将来的に対応予定である．

7.4.2 XQuery への変換

select 句，from 句，return 句に分けて処理を以下のように行うことによって OQL から XQuery に変換している．また，XQuery で用いる変数には OQL で用いた変

数に，先頭に$ 付けたそのままの変数名を用いている．

- select 句：
  XQuery の return 句にそのまま出力される．
- from 句：
  クラス宣言は for 句，別名は let 句，配列またはコレクションの展開は for 句と let 句を併用して XML の参照を辿るように XPath を生成する．
- return 句：
  比較対象が Object と Object の場合はその id 値を用いて，それ以外の場合は OQL の内容をそのまま出力する．

8 今後の課題

現在，オブジェクトを登録し，OQL を用いた単純な検索は可能となっているが，まだまだ実用レベルのデータベースとしては機能不足である．

ユーザー認証，並行処理，OQL 内からのメソッド呼び出し，インターフェイス・スーパークラスを用いての検索，Xpriori 以外の XMLDB への対応など，多数の機能を実現すべく，これからもこつこつと研究を進めていきたいと考えている．

参考文献

[1] 『The Object Data Standard: ODMG 3.0』 : http://www.odmg.org/

[2] 『オブジェクト指向データベース入門』 Won Kim 著 増永良文・鈴木幸市 監訳 (株) 共立出版

[3] 『オブジェクト指向データベース』 J.G. ヒューズ 著 植村俊亮 監訳 石丸知之 訳 サイエンス社

[4] 『オブジェクト指向データベースシステム』 永田元康 著 森北出版株式会社

[5] 『オブジェクト指向データベース』 Setrag Khoshafian 著 野口喜洋・小川束 訳 (株) 共立出版

[6] 『オブジェクト指向データベース』 石塚圭樹 著 (株) アスキー

[7] W3C(World Wide Web Consortium) : http://www.w3.org/

[8] Extensible Markup Language (XML) 1.1 : http://www.w3.org/TR/xml11/

[9] Document Object Model (DOM) Level 3 : http://www.w3.org/TR/DOM-Level-3-Core/

[10] XPath(XML Path Language) Version 2.0 : http://www.w3.org/TR/xpath20/

[11] XQuery(An XML Query Language) 1.0 : http://www.w3.org/TR/xquery/

[12] 『改訂版標準 XML 完全解説 (上)(下)』 (株) 日本ユニテック 中山幹敏 奥井康弘 技術評論社

[13] Xpriori : http://www.xmldb.jp/

[14] Java 2 Platform Standard Edition (J2SE) 5.0 : http://java.sun.com/j2se/1.5.0/

[15] 『Java RMI』 William Grosso 著 田和勝 訳 O'REILLY