

OODBのOSIディレクトリシステムへの適用

5C-10

～スキーマ管理～

山上 俊之 山田 広佳 岩崎 孝夫

(株) 東芝 情報処理・機器技術研究所

1. はじめに

筆者らはOSIディレクトリシステム(DS)をオブジェクト指向データベース管理システム(OODBMS)を用いて実現した。DSではディレクトリスキーマと呼ばれるアプリケーションスキーマが規定されている。筆者らのシステムでは、このディレクトリスキーマを後から操作できる、システム管理者向けのツールを提供することを意識して実装を行なった[1]。

DSをOODBMSで実現する場合、ディレクトリスキーマとOODBMSのスキーマには一部重なるものがあり、ディレクトリスキーマの変更をOODBMSのスキーマに反映する処理が必要であった。

本稿では、ディレクトリスキーマ情報の格納形態とOODBMSのスキーマに反映する仕組みを中心に報告する。

2. ディレクトリスキーマの概要

ディレクトリシステムにおいて規定されているディレクトリスキーマには、

- 1) DIT内のエン트리間の関係を規定するDIT構造定義
- 2) 各オブジェクトクラスが持つべき属性を規定したオブジェクトクラス定義
- 3) 属性型の属性構文、マルチバリュウの可否などを定めた属性型定義
- 4) 各属性の照合規則などを定義した属性構文定義

がある。

これらのディレクトリスキーマのあるものは、国際的、国内的に標準化されることになる。加えて、私的な

Implementation of OSI Directory System using OODB
Toshiyuki Yamagami, Hiroyoshi Yamada, Takao Iwasaki
Information Systems Engineering Laboratory,
TOSHIBA Corporation

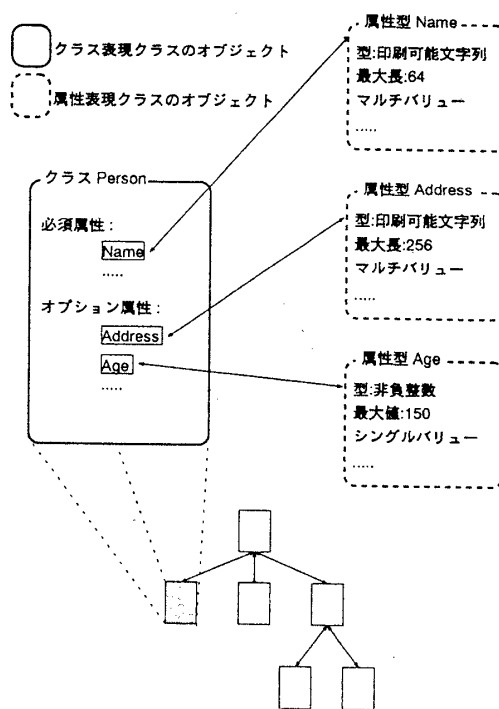


図1: ディレクトリスキーマのオブジェクトによる表現

組織内でも決められることを想定しており、このためにシステム管理者向けのツールの提供を考慮している。

筆者らのシステムでは、DSにおけるオブジェクトクラスをOODBMSのクラスに、属性型をそのインスタンス変数に対応づけて実現している[2]。したがって、ディレクトリスキーマのうちオブジェクトクラス定義の変更、たとえば使用するオブジェクトクラスの追加や持つべき属性の変更に際しては、OODBMSのスキーマに反映させる必要が生じることになる。

3. スキーマ情報の格納形態

ディレクトリスキーマ情報はオブジェクトにより管理されている(図1)。各クラス毎に作成されるオブジェクト群(クラス表現クラスのオブジェクト)、属

性型毎に作成されるオブジェクト群（属性表現クラスのオブジェクト）を基本構成要素とし、それらの関係でスキーマ情報を表現している。属性表現クラスのオブジェクトには、前述のディレクトリスキーマの内3), 4)の属性型定義、属性構文定義の情報がそのインスタンス変数の値として保持されており、クラス表現クラスのオブジェクトと属性表現クラスのオブジェクトとのリンクにより、オブジェクトクラス定義を表現する。図の例では、クラス Person は属性として Name, Address, Age を持つというオブジェクトクラス定義をオブジェクトで表現している。

さらにクラス表現オブジェクト間には、2種類のリンクが張られており、1つでDIT構造定義をもう1つでクラス間の継承関係を表現している。

スキーマの操作においては、オブジェクトクラスにエントリがある場合や、サブクラスがある場合などには、該当オブジェクトクラスの定義は変更できないようにしている。このための確認機構は、管理ツール側のコードではなく、カプセル化の技法を用いてクラス表現クラスおよび属性表現クラスのオブジェクト側に組み込んで実現している。

4. スキーマの更新処理

前述したように、ディレクトリスキーマのうちオブジェクトクラス定義が変更されると、これをOODBMSのスキーマに反映する必要がある。

使用した市販オブジェクト指向データベース管理システムでは、ある程度のスキーマ進化機能をサポートしている。しかしながら、筆者らのシステムでは、ディレクトリデータとスキーマデータをアクセスするプログラムを分けているため、この機能を2つのプログラムで共用するには問題があると判断した。

このため、データベースへのスキーマ登録に必要なヘッダファイルと、クラス実装ファイル、およびディレクトリデータにアクセスするプログラムから使用されるアクセスライブラリを必要に応じて自動生成し、コンパイルを行なうものとした。この部分の処理をスキーマ更新処理と呼ぶことにする。

このようにスキーマ更新処理は、ソースの自動生成、再コンパイルという形態で行なわれるので、時間コストがかかる。このため、管理ツールによって変更されたディレクトリスキーマ情報は、ツールの終了時にまとめてスキーマ更新処理を行なうものとした。

クラス表現クラスのオブジェクトには終了のメッセージを受けると自分のクラスに関するソースをはきだし、

継承関係を表すリンク先に同じ終了のメッセージを送るメソッドを実装している。ディレクトリシステムにおけるオブジェクトクラスは、必ず特別なクラスであるトップクラスから継承されているので、トップクラスに対応するクラス表現クラスのオブジェクトに終了のメッセージを送ると、終了のメッセージがオブジェクトで表現しているオブジェクトクラス構造をリカールに伝わり、自動的に必要なファイルが生成される。

OODBMSのスキーマ登録は、ユーティリティを介してC++のヘッダファイルを各クラス毎に通すことで行なわれる。この時には、登録するクラスに関して、そのスーパークラスは既に登録が完了してはならないが、上述の仕組みによりはきだされたヘッダファイルを作成された順に登録すれば、この条件は容易に満たすことができる。

5. おわりに

今回はOODBMSスキーマ操作を、ソースの自動生成とコンパイルという方法で実現している。ディレクトリシステムでは、データ操作が比較的単純なこともありこの方法でも実現することができた。しかしながら、複雑な操作を必要とするシステムでは、スキーマ操作の実現がかなり開発の負担になるものと思われる。

このため、OODBMSのスキーマ進化機能充実が今後望まれる。

参考文献

- [1] 岩崎他, “OODBのOSIディレクトリシステムへの適用～概要～”, 情報処理学会第47回全国大会, 1993
- [2] 山田他, “OODBのOSIディレクトリシステムへの適用～データ管理～”, 情報処理学会第47回全国大会, 1993