データベース・システム 73－8
オペレーティング・システム 44－8
（１９８９．９．８）

オブジェクト指向データベース管理システムの物理構造仮想化方式

鶴岡 邦敏　　木村 裕

日本電気㈱ Ｃ＆Ｃシステム研究所

オブジェクト指向ＤＢＭＳに対して、応用プログラムが対象とする概念クラスと、物理構造に依存した内部クラス（格納クラス、ファイルクラス）とを分離し、内部クラスを階層化する方式を提案する。ファイルクラスは個々のファイル形式ごとに存在し、汎化階層を利用できると共に、部品化により複合的なファイル形式を定義できる。格納クラスはオブジェクトの内部形式と概念形式との差を吸収する。内部クラスの組合せにより、オブジェクトの種々の格納形式を容易に定義することができる。この方式により、関係ＤＢやメディア特有のファイルを初め、多様な形式のファイルをシステムに容易に組み込むことが可能となり、ＤＢＭＳの柔軟性が向上する。

ON PHYSICAL STRUCTURE VIRTUALIZATION FOR OBJECT-ORIENTED DATABASE MANAGEMENT SYSTEMS

Kunitoshi TSURUOKA and Yutaka KIMURA

C&C Systems Research Laboratories, NEC Corporation
4-1-1 Miyazaki, Miyamae-ku, Kawasaki, 213 Japan

A physical structure virtualizaition scheme is proposed for object-oriented DBMSs. The scheme separates conceptual classes which application programs handle, from internal classes (storage classes and file classes) which depend on physical structure and are organized hierarchically. The file class exists for each file organization. In addition to utilizing generalization hierarchy, complex file structure can be defined by components assembly. The storage class absorbs the structural difference between internal and conceptual objects. A wide range of object storage structures can easily be defined by combining internal classes. The scheme enables to incorporate a variety of file structures such as relational databases or distinct media files, which increases DBMS flexibility.

1．はじめに

オブジェクト指向データベース（ＯＯＤＢ）は、次世代データベースの本命として、ここ数年急速に研究、開発、製品化が進行している[AnHa87,BCGK87,CDRS86,KBBC88,etc.]。これは、データ型の拡張性と抽象化能力、手続きの組み込み、複雑な構造の操作等、ＯＯＤＢが持つ特長が、新しい応用分野（ＣＡＳＥ、ＣＡＤ、マルチメディア処理、エキスパートシステム等）からの要請に相当程度答え得るためと思われる。特に、多様で構造が複雑なデータを含む応用分野に対して、応用プログラムを容易かつ統一的に記述でき、システム開発を大幅に効率化できることが、ＯＯＤＢの大きな利点と言える。

一方、ＯＯＤＢが実用化される段階になると、以下のような機能が重要性を増してくる。

①関係ＤＢ、ＣＡＤファイル、イメージファイル等の既存のデータ資源を有効に活用できること。

②新しいファイル構造、アクセス手法、ハードウェア等を容易に利用できること。

③ＤＢＭＳ本体が軽くかつ柔軟で、利用者からの種々の要求（機能／性能）に応じて、容易にシステムの変更（機能追加、削除）ができること。

これまで提案されているオブジェクト指向ＤＢＭＳ（ＯＯＤＢＭＳ）は、まだ上記の要請を満たしているとは言えない。これは、従来の研究が概念モデル中心であり、上記で必要とされる物理構造のモデル化が不十分なためと思われる。これまでにも、ＧＥＮＥＳＩＳの "abstract file"[BBGS.88,Bato87]、Ｖｂａｓｅの "StorageManager" の階層 [DaLa88]、ＯＲＩＯＮの "Device Class"の階層[WoKi87]等が提案されている。しかしながら、オブジェクト指向モデルの利点を十分に生かした形で前記の機能を実現したものはない。

本資料では、応用プログラムが直接処理の対象とするオブジェクト（概念オブジェクト）以外に、ファイル等の物理構造もオブジェクト（内部オブジェクト）の階層として実現し、両者の間の写像関係を構築する方式（物理構造仮想化方式）を提案する。これによって、関係ＤＢやメディア特有のファイルを初め、種々の形式のファイルをシステムに容易に組み込むことが可能となる。

2．概念クラスと外部クラス

以下では、オブジェクト、クラス、インスタンス、メソッド、クラス変数、インスタンス変数等の用語を、通常のオブジェクト指向言語及びＯＯＤＢのそれと同様の意味で使用する（クラスもオブジェクトとして扱う）。但し、従来のクラスを特に「概念クラス」と呼ぶ。即ち概念クラスとは、利用者が対象とする領域をモデル化するために使用する抽象的な実体であり、格納構造等の実現手法には依存しない。次に、応用プログラム（ＡＰ）がオブジェクトをアクセスするために、ＡＰと概念オブジェクトとのインタフェース構造を記述したものを「外部クラス」と呼ぶ。外部クラスは、以下の機能を有する。

- 概念オブジェクトをＡＰの言語固有のデータ形式に変換する（外部クラス中のメソッドとして変換手続きを記述する）。特定の概念クラスに対して、一般にそれをアクセスする言語の数だけの外部クラスを定義する必要がある。
- マシンアーキテクチャや言語などの制約（データ長や個数等）により、概念オブジェクトからの出力を直接受け取れない場合に、出力情報の変換を行なうことによりその制約を吸収する。
- 概念オブジェクトとＤＢＭＳの実現方式を、ＡＰに意識させないような仮想化を行なう。例えば、ＡＰとＤＢＭＳのインタフェース（プロセスの分離等）、メッセージ送信の方法、概念オブジェクトのアドレス方法等を見えなくする。

外部クラスと概念クラスとの関係を、図１．に示す。

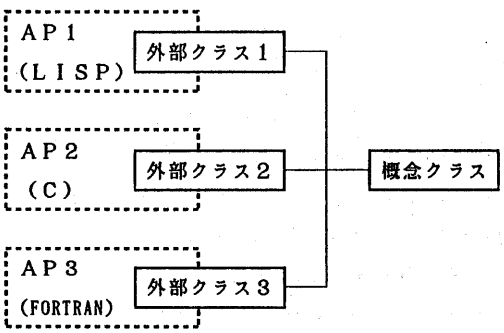

図１．外部クラスと概念クラスの関係

3．内部クラス

ファイル構造やアクセス手法等、概念オブジェクトの物理的な実現方式を規定するものを総称して「内部クラス」と呼ぶ。内部クラスには、格納クラスとファイルクラスとがある。

（1）格納クラス

概念オブジェクトの二次記憶への格納方法を規定するクラスを「格納クラス」と呼ぶ。格納クラスは、概念クラスとファイルクラス（後述）の間の写像関係を記述する機能を持つ。一つの概念クラスに対しては、格納クラスが唯一つ対応する（概念クラス：格納クラス＝m：1）。対応する格納クラスを持たない概念クラスは一時オブジェクトであり、二次記憶には格納されない。一つの格納クラスに対しては、ファイルクラスが唯一つ対応する（格納クラス：ファイルクラス＝n：1）。格納クラスは、概念オブジェクトの物理的な実現方式を応用プログラムから見えなくし、またオブジェクトの格納方法の変更に際して概念クラスの変更を不要にする。

格納クラスの機能を以下に示す。

- 二次記憶上のオブジェクト（内部オブジェクト）を、概念オブジェクトの形式に変換する。
- そのクラスのオブジェクトＩＤを管理し、オブジェクトＩＤを物理的なアドレス等（内部オブジェクトのキー情報）に変換する。
- 異種のデータベース（関係ＤＢ等）や別のマシン上のデータベースを用いる場合に、相手のＤＢＭＳとインタフェースを持ち、データ変換を行なう。

（2）ファイルクラス

特定のファイルに対して、そのアクセス手法を規定するクラスを「ファイルクラス」と呼ぶ。ファイルクラスは一つのファイル形式（例えば標準テキストファイル、イメージファイル等）に対して一つ存在し、そのインスタンスは実ファイルごとに存在する。更に、複数ファイルをまとめて扱うアクセス手法に対しても一つのファイルクラスが存在する。例えば、あるデータファイルのスペース管理用のビットマップを別ファイルとして持ち、全体として一つのファイル（アクセス手法）として扱いたい場合には、データファイルのクラス、ビットマップファイルのクラス、それらを統合するクラスという３つのファイルクラスが必要である（[Bato87]の手法に一部類似）。即ちファイルクラスは階層的に構成することが可能である（但し、後述するファイルクラスのクラス階層とは意味が異なる）。内部クラス（格納クラス、及びファイルクラス）と概念クラスの関係を、図２．に示す。

4．ファイルクラスの作成

（1）ファイルクラスのクラス階層

多種のファイル形式を統一的に管理するため、及び既存アクセス手法の利用により新しいファイルアクセス手法の記述を容易にするために、ファイルクラスに

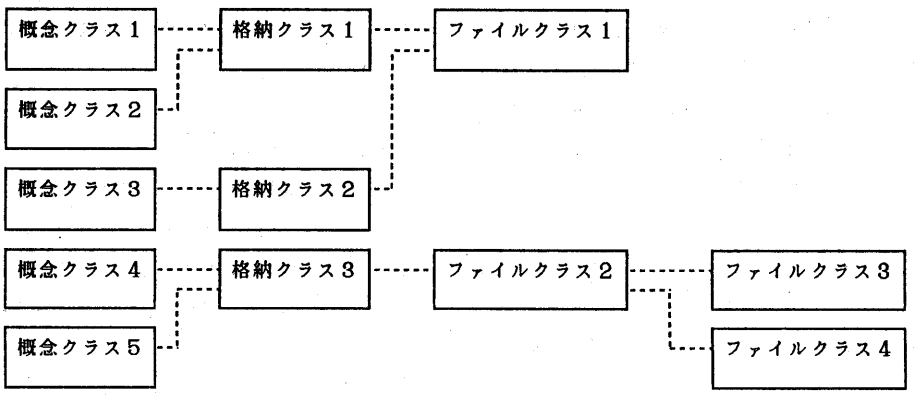

図２．　内部クラスと概念クラスの関係

もクラス階層（スーパークラス、サブクラス）を導入する。ファイルクラスの階層の例を、図３．に示す。

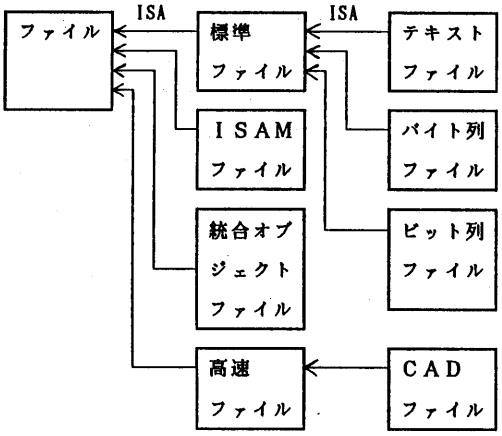

図３．ファイルクラスの階層

図で、例えばビット列ファイルは標準ファイルの特別なものであるから、ビット列ファイルのインスタンス（＝１つの実ファイル）に対して標準ファイルのメソッド（アクセスコマンド）を実行することが可能である。

（２）ファイルクラスの定義

利用者は、同種の物理構造を持つファイル（インスタンス）群をまとめて、一つのファイルクラスとして定義する。また同時に、そのファイル構造に依存した特定のアクセス手法をメソッドとして定義する。ある範囲のファイル形式に対して既に特定のアクセス手法が規定されている場合には、これをそのまま一つのファイルクラスとして定義することができる。例えば関係ＤＢの場合には、「ＲＤＢファイルクラス」というファイルクラスをひとつだけ定義し、ＳＱＬ文を発行するメソッドを記述すればよい。

５．格納クラスの作成

格納クラスは、内部オブジェクトを概念オブジェクトに変換する役割を持つ。このため、オブジェクト指向の立場に立つと、一般に一つのファイルクラスと一つの概念クラスの組に対して、唯一つの格納クラスが必要となる。ところが通常のＤＢＭＳでは、概念クラスの形式（スキーマの形式に相当）はシステムで固定にする場合が多い。この制限を設けると、格納クラスは、通常はファイルクラスごとに定義されるものとなる。例を図４．に示す。

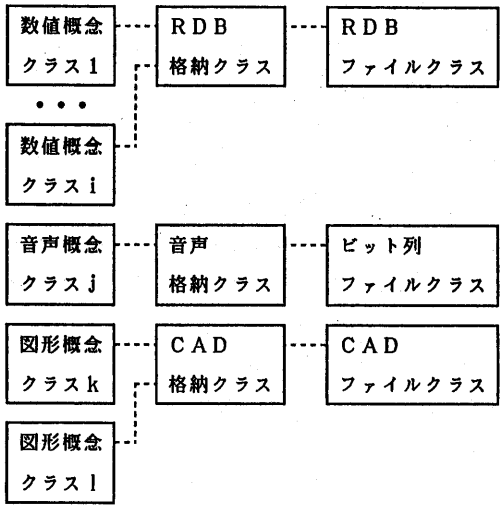

図４．格納クラスの作成

ここでデータ変換のための格納クラスのメソッドは、特定のメタデータ（スキーマ／システムカタログ）に依存した汎用のもの（図のＲＤＢ格納クラス）、及びファイルごとに手続き的に定義された専用のもの（図の音声格納クラス）のいずれでもよい。なお、図２．のように、一つのファイルクラスに複数の格納クラスを対応させてもよい。これは、一つのファイル内を区分して、異なる構造のデータ群を別々に格納する等の場合に利用できる。

６．内部クラス間の関連

以下では、格納クラスとファイルクラス、及びそれらのインスタンス群を用いることにより、種々の場合にオブジェクトの格納構造がどのように表現されるかを示す。それぞれ図の例を参照のこと（細実線の四角はインスタンスを示す）。

（１）１概念クラス→１ファイルインスタンス

図５．参照。例えば、１つの概念クラスの全インスタンスを一つのＩＳＡＭファイルに格納する場合など。

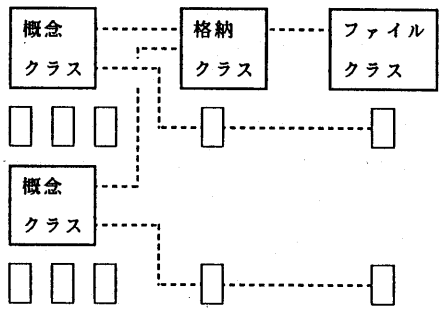

図５．　1概念ｸﾗｽ→1ﾌｧｲﾙｲﾝｽﾀﾝｽ

（２）ｍ概念クラス→１ファイルインスタンス

図６．参照。複数の概念クラスのインスタンス群を一つのファイルに混在させて格納する場合である。これにより、同時に参照されるデータ群のクラスタリングが実現でき、検索速度の向上が期待できる。

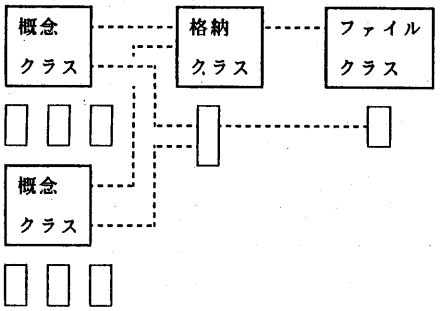

図６．　m概念ｸﾗｽ→1ﾌｧｲﾙｲﾝｽﾀﾝｽ

（３）１概念インスタンス→１ファイルインスタンス

図７．参照。例えば、イメージオブジェクト（インスタンス）を一個づつ別のファイルに格納する場合等がある。この場合には「ファイル組集合クラス」を利用し、そのインスタンスが、概念インスタンスとファイルインスタンスとの対応関係を記憶する。なお、ファイル組集合クラスは、集合クラスのサブクラスである組集合クラスのサブクラスである（図８．参照）。

（４）１概念クラス→ｎファイルインスタンス

図９．参照。性能上・管理上の目的で、概念クラスの内容を複数のファイルに分割して格納し、それらを一つのファイルとして連結して見たい場合、などに利用する。この場合には「ファイル組クラス」を利用し、そのインスタンスが、格納に使用するファイルインスタンスの組を記憶する。この時、概念インスタンスとそれが格納されるファイルインスタンスとの対応付けは、ファイル組クラス中の手続き（メソッド）によって記述される。なお、ファイル組クラスは、単純組クラスのサブクラスである（図８．参照）。

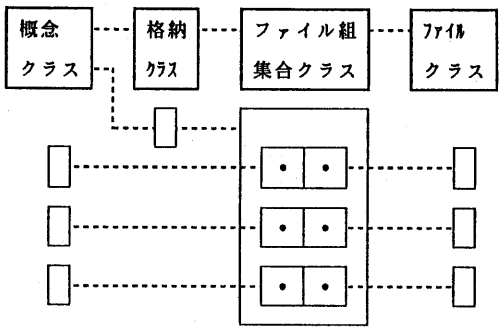

図７．　1概念ｲﾝｽﾀﾝｽ→1ﾌｧｲﾙｲﾝｽﾀﾝｽ

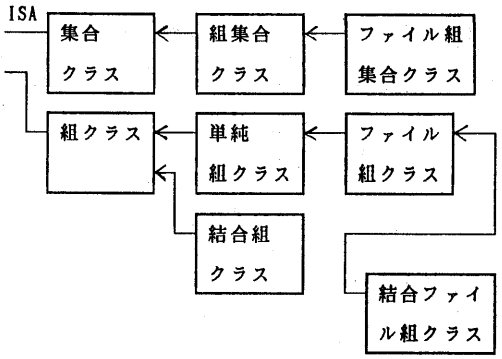

図８．集合クラスと組クラスの階層

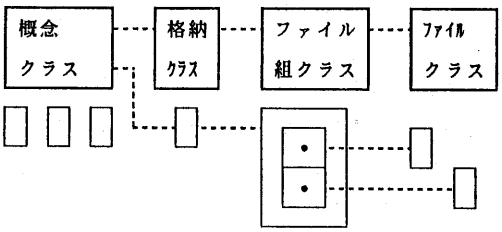

図９．　1概念ｸﾗｽ→nﾌｧｲﾙｲﾝｽﾀﾝｽ

（５）結合（ｊｏｉｎ）の表現

図１０．参照。オブジェクト間の結合では、ファイルレベルで結合を行う場合（図上）と、概念レベルで結合を行う場合（図下）とが考えられる。ファイルレベルの結合は「結合ファイル組クラス」によって表現され、結果は通常の概念オブジェクトとなる。結合ファイル組クラスは、ファイル組クラスに結合のためのメソッドを記述したものである（図８．参照）。一方概念レベルの結合では、結合組クラスのサブクラスが生成され、結合形式の概念オブジェクトができる。

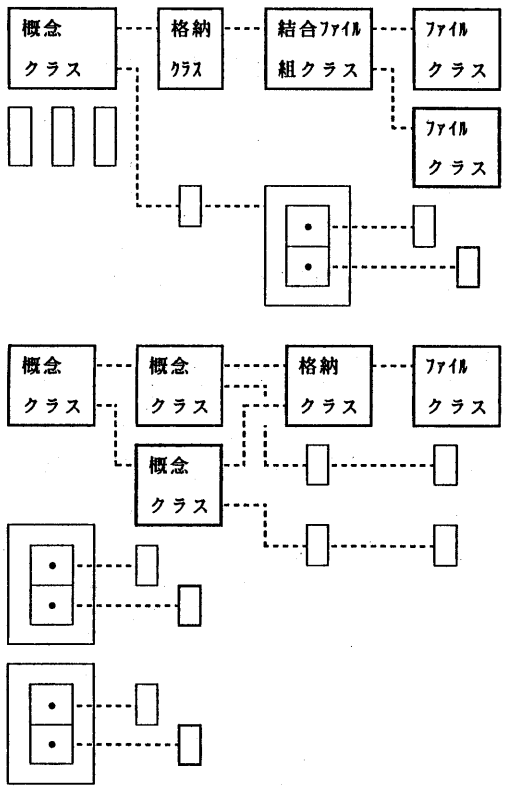

図１０．結合の２つの表現

７．おわりに

本資料で提案した物理構造仮想化方式は、以下の特長を持つ。

①データ資産の活用

- 関係ＤＢ、マルチメディア特有の形式、応用固有の形式等、種々の形式のファイルをシステムに容易に組み込むことができる。このため、既存のデータ資源を有効に活用することが可能となる。

②応用プログラム（ＡＰ）開発・保守コストの削減

- 既存のファイルクラスの利用により、通常のＡＰは格納構造を意識する必要がなく、ＡＰ開発が効率化する。
- データ格納手段の変更に対して、ＡＰを変更する必要がなく、データ独立性が向上する。

③ＤＢＭＳ開発・保守コストの削減

- 内部クラスの再利用により、新しいファイル構造やアクセス手法の定義が容易となる。また、それらをシステムに容易に組み込むことができる。
- 要求される性能、機能に応じて、ＡＰを変更することなしに、多種類のデータ格納手段を容易かつ柔軟に提供できる。
- ＤＢＭＳ自体をモジュール性の高いクラスの集合として記述することが可能となり、種々の応用分野に容易にカスタマイズできる拡張可能ＤＢＭＳが実現しやすくなる。

なお、筆者らのグループでは、次世代データベースアーキテクチャ研究の一環として、Ｏｄｉｎ（ｵｰﾃﾞｨﾝ；an Object-oriented database management system for data-intensive applications）と呼ぶオブジェクト指向ＤＢＭＳの研究・開発を進めている。Ｏｄｉｎの初版（Common Lisp+Flavorで記述）は既に稼働しており、本資料で提案した物理構造仮想化方式は、強化版に実装される予定である。

おわりに、オブジェクト指向ＤＢの応用に関して多くの御意見を頂いた日本電気㈱の平野文康氏、システム開発に協力して頂いた㈱日本電気技術情報システム開発の栗本光樹・大沢科子の両氏、及び研究方向に関して貴重な助言を頂いた日本電気㈱の永井義裕、阪田史郎の各氏に深謝致します。

参考文献

[AnHa87] Andrews, T. and Harris, C. "Combining Language and Database Advances in an Object-Oriented Development Environment," Proc. OOPSLA, 1987, pp.430-440.

[BBGS.88] Batory,D.S., Barnett,J.R., Garza,J.F.,

Smith,K.P., Tsukuda,K., Twichell,B.C. and Wise,T.E. "GENESIS: An Extensible Database Management System," IEEE Trans. Softw. Eng. Vol.14, No.11, 1988, pp.1711-1730.

[BCGK87] Banerjee,J., Chou,H.-T., Garza,J.F., Kim,W., Woelk,D., Ballou,N. and Kim,H.-J. "Data Model Issues for Object-Oriented Applications," ACM TOOIS, Vol.5, No.1, 1987, pp.3-26.

[Bato87] Batory,D.S. "Principles of Database Management System Extensibility," Database Engineering, 1987, pp.100-106.

[CDRS86] Carey,M.J., DeWitt,D.J., Richardson, J.E. and Shekita,E.J. "Object and File Management in the EXODUS Extensible Database System," Proc. VLDB, 1986, pp.91-100.

[DaLa88] Damon,C. and Landis,G. "Abstract Types and Storage Types in an OO-DBMS," Proc. COMPCON spring, 1988, pp.172-176.

[KBBC88] Kim,W., Ballou,N., Banerjee,J., Chou, H.-T., Garza,F. and Woelk,D. "Integrating an Object-Oriented Programming System with a Database System," Proc. OOPSLA, 1988, pp. 142-152.

[KiTs89] 木村、鶴岡「オブジェクト指向データベース管理システムの柔軟性について」、情報処理学会第38回（平成元年前期）全国大会、2R-5

[Tsur89] 鶴岡「オブジェクト指向データモデルによる非正規形レポート生成方式」、情報処理学会第38回（平成元年前期）全国大会、2R-3

[WoKi87] Woelk, D. and Kim, W "Multimedia Information Management in an Object-Oriented Database System," Proc. VLDB, 1987, pp.319-330.