データベース・システム 65－3
（１９８８．５．１９）

距離空間データモデルＭｅＳＯＤに基づいた オブジェクト指向データベース管理システムの設計

田渕　仁浩　村岡　洋一
早稲田大学　理工学部

本稿では、我々が先に提案した距離空間データモデルＭｅＳＯＤに基づくオブジェクト指向データベース管理システム（ＤＢＭＳ）の設計概念とその実現方法について述べた。距離空間データモデルＭｅＳＯＤは、マルチメディアデータベース技術の高度応用として進めている電子化図書システムHyper Bookの核となるマルチメディアデータモデルである。このＭｅＳＯＤでは、オブジェクトの実現値とオブジェクトの存在とは独立している。また、ＭｅＳＯＤではデータ定義域と各データ間の関連度を計量する距離関数の集合を一つの抽象データ型（空間型という）にまとめている。これら二つの点でＭｅＳＯＤとオブジェクト指向は接している。ＭｅＳＯＤ-ＤＢＭＳの設計にオブジェクト指向を取り入れるということは、ＭｅＳＯＤでいうオブジェクト（OBJと書く）とオブジェクト指向言語で言うオブジェクトとを融合するということである。

Design of object oriented data base management system which is based on Metric Spatial Object Data model,MeSOD.

Masahiro TABUCHI　Yoichi MURAOKA
Department of Electronics and Communication Engineering,
School of Science and Engineering,Waseda University
3-4-1 , Ohkubo, Shinjuku-ku, Tokyo , 160 , Japan

In this paper, we discuss the design and the implementaion method of the data base management system that is based on Metric Spatial Object Data model (MeSOD), which we have proposed . MeSOD is a data model for multi-media application, and we are developing a multi-media electronic book: Hyper Book, based on MeSOD. In MeSOD existence of an object is represented by an object identifier; therefore, existence of an object is independent of an object's occurence. Moreover, object's domain and a set of distance function are component of Spatial Object Type which is abstract data type.Thus MeSOD has something in common with Object-Oriented Languages.The main role of MeSOD-DBMS is to provide the unification of MeSOD's object (OBJ) and Object-Oriented Languages' object.

１．はじめに

我々が研究開発中のマルチメディア電子化図書システムHyper Bookは、マルチメディアデータベースを核としたアプリケーションである。このHyper Bookは、従来の紙メディアが担ってきた情報や知識の蓄積、伝達などの役割をより高度なものにすることを目的としている〔TabMur87b〕。このようなマルチメディアなどの高度応用向けのデータモデルの要件として、

(1)従来のキーワード検索を前提としたモデルではなく、柔軟な検索方法に適応できるモデルであること〔TabMur87b〕〔SbSbTM88〕。

(2)複雑な構造を扱えるメカニズムをもっていること。

などが挙げられる。(1)の要件は、マルチメディアデータのように情報量の多いデータの検索は、数値/文字列中心の応用向けのモデルでは不十分なので、類似検索やブラウジングなどが容易にできなくてはならないということを意味している。(2)の要件は、高度な応用においては、より複雑な構造を持つ情報を扱うためにデータや操作を抽象化して扱える必要があるということを意味している。

これらの要件に対して距離空間データモデルＭｅＳＯＤ〔TabMur87a〕では、

(a)オブジェクト間の意味的な相関関係を距離位相で表現し、内容の検索を距離に基づく近傍検索で実現すること。

(b)積空間OBJ(IS-PART-OF関連)と商空間OBJ(IS-A関連)という、より抽象度の高い複合空間OBJを作り出すメカニズム。

を提案している。(a)の実例として、〔SbSbTM88〕においてキーワードによらないマルチメディアデータの内容検索の可能性を示している。

本稿では、このＭｅＳＯＤに基づくオブジェクト指向データベース管理システム(ＤＢＭＳ)の設計概念について述べる。

ＭｅＳＯＤデータベース管理システム(ＭｅＳＯＤ-ＤＢＭＳ)はHyper Bookシステムの核である。Hyper Bookでは、情報や知識の表現にマルチメディアデータを積極的に取り入れると同時に、これを直接的に扱えること、状況の変化や目的に応じて表現方法や構造も変化できることをその要件としている。具体的には、メディアデータによるメディアデータの内容検索や新しいメディアの追加、動的なスキーマの変更を可能とするようにＭｅＳＯＤ-ＤＢＭＳを設計する必要がある。そこで採用したのが、オブジェクト指向概念に基づくアプローチである。しかしながら、オブジェクト指向そのものは一種の哲学であり、これをどのようにＤＢＭＳの設計に取り込んでいくかが重要な課題である。ＭｅＳＯＤ-ＤＢＭＳでは、システム構成の概念としてのオブジェクト指向とオブジェクト指向言語へのインターフェースとしてのオブジェクト指向を取り扱う。

図１はＭｅＳＯＤ-ＤＢＭＳの概念的構成図であるが、図のようにＭｅＳＯＤ-ＤＢＭＳはObject Serverと呼ばれる核を持ち、アプリケーションはこの核に対して要求を発効するという、いわゆるサーバー・クライアント方式を採用する。この方法は、既存のウィンドウシステムの成功例が示すように２章で述べているようなＭｅＳＯＤ-ＤＢＭＳに対するいくつかの要件を解決する。２章における要件を受けてオブジェクト指向アプローチを採るとして、それではＭｅＳＯＤにおける概念とオブジェクト指向概念とがどのように結びつき、どのように役立つのかということについては３章で述べる。４章では、図１に対応するＭｅＳＯＤ-ＤＢＭＳのソフトウェアアーキテクチャについて述べ、５章ではオブジェクト概念を実現するＭｅＳＯＤ核の基本機能とその実現方法について議論する。最後に６章ではまとめと今後の課題について触れる。

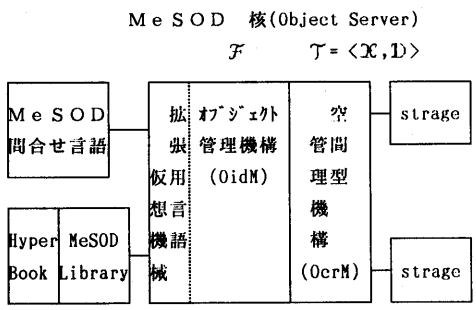


図１　ＭｅＳＯＤ-ＤＢＭＳの構成

２．ＭｅＳＯＤ-ＤＢＭＳに対する要件

Hyper Bookのような各メディアが有機的に結合した応用では、検索要求に対する自由度や情報表現の多様性、システムの拡張性などが重要である。したがってＭｅＳＯＤ-ＤＢＭＳに対する要件は、以下のようなものになる。

（１）距離の概念の取り扱いに関する要件

ＭｅＳＯＤでは高度な検索を実現するために距離の概念を導入している。しかし、距離関数の定義をユーザーに直接行わせるのは望ましくない。例えば数値データなどの場合は、手続きとして定義できるが、より複雑な構造を持つオブジェクトや構造そのものの距離を作り出すのは困難である。したがって、ディスプレイ上のアイコンの位置を変えると距離も変わるというような間接的な定義機能が必要である。また、システムに登録する距離関数は、距離表の形で実現することもある。この場合も、ユーザーによる直接的な扱いは矛盾を生じる原因となる。したがって距離の実現方式には依存しないインターフェースも必要になる。

（２）マルチメディアストレージのサポート

マルチメディアデータには画像や音声のように大容量のストレージを必要とするものから、従来の文字列のように比較的小容量のストレージでも良いものまでさまざまである。同時にこれらは扱い方も異なるため、各メディアに適したストレージを扱える必要がある。

（３）ストレージ独立性

(2)の要件を支援する場合に、各応用にとっては、ストレージは高度に抽象化されている必要がある。すなわち、ストレージが光ディスクだろうが、磁気ディスクだろうが、あるいは専用の機能ディスクであろうが応用プログラムを書き直したりすることがあってはならない。

（４）デバイス独立性

Hyper Bookでは音声メディアなら音声で、図形メディアなら図形で内容検索を可能とする。しかし、そのためのメディアデータの入力装置と応用とは、独立していなくてはならない。また検索結果の表示などの情報の出力についても、ディスプレイからでも高解像度の画像出力装置でも区別無く扱える必要がある。

（５）ネットワーク透過性

Hyper Bookのプラットホームは、ワークステーションであり、このワークステーションの重要な機能の一つがネットワーキング機能である。ＭｅＳＯＤ-ＤＢＭＳでもこのネットワーク機能を十二分に利用できたほうが良い。例えば、別のワークステーションの画像と音声用のストレージをユーザーは自分の所にあるようにアクセスできる必要がある。

（６）ユーザーインターフェースのカスタマイズ

オブジェクト管理のためのユーザーインターフェースは、ユーザーや応用毎に異なってくる。例えば、システム設計者とエンドユーザでは、オブジェクトの管理や利用の仕方が変わってくる。したがってオブジェクトマネージャは、いわゆるView機能を持っている必要がある。

（７）マネージャ機能の拡張可能性

システムが、現在備えている機能を使って新しいタイプのストレージを追加したり、より機能の高いシステムへの拡張ができなくてはならない。

（８）並行制御

初期設計においては、この重要な問題を避けて通るが、実行効率や(5)の機能をより有効なものとするためには必要不可欠な要件である。

３．ＭｅＳＯＤにおけるオブジェクト概念

２.で挙げた距離概念や独立性、開放性の要件には、抽象化の技法が有効である。一方、オブジェクト指向概念はデータと手続きの、より進んだ抽象化技法の一つであり、このオブジェクト指向に基づいた設計を考える。そこで、この章ではＭｅＳＯＤの基本概念である空間オブジェクト(以下ではＭｅＳＯＤのオブジェクトをOBJで表す。)とオブジェクト指向言語(以下ではOOPLと書く。)におけるオブジェクトとの融合を図る。

３.１ ＭｅＳＯＤの概要〔TabMur87b〕

ＭｅＳＯＤの枠組みでは、実体の存在と実体の実現値とは独立して扱う。すなわち〈Ｆ，Ｘ，Ｄ〉の３つ組で空間OBJを表現する。ここでＦは空間を表す一つのオブジェクト識別子(例えば空間名)でＸは定義域(データ構造)、Ｄは定義域Ｘ内のデータを計量することのできる距離関数の集合(手続き)である。ＭｅＳＯＤでは、さらに空間型の概念を導入し、空間型Ｔ＝〈Ｘ，Ｄ〉としている。これを用いて空間OBJは〈Ｆ，Ｔ〉と書くことができる。ＭｅＳＯＤにおける空間型は、データ構造とそれに対するアクセス手段の組であるから一種の抽象データ型である。このように、ＭｅＳＯＤではOBJの存在をその実現値とは独立に扱えることと、OBJは抽象化されたデータ型を持っているという２つの点でオブジェクト指向言語(OOPL)のオブジェクトと接点を持っている。OOPLにおけるオブジェクトとの違いは、一つの空間名(オブジェクト識別子)に対応する空間OBJが複数存在できるということである。つまり、オブジェクトと空間OBJは１対多の関係にある(図２)。

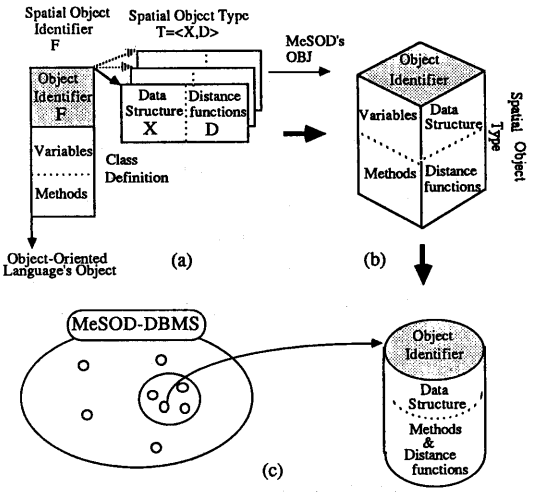


図２　ＭｅＳＯＤのOBJとオブジェクト

また、ＭｅＳＯＤには積空間OBJと商空間OBJという二つの複合空間OBJを作り出すメカニズムがあるがOOPLにはない。以下に、これらの複合空間OBJの概要を示す。

（１）積空間OBJ

積空間OBJは、プロパティと呼ばれる構成要素空間OBJの集約により作られる複合空間OBJである。例えば、動物空間OBJを名前、種別、生息地の各空間OBJからなる積空間OBJと定義すると、個々の動物OBJはこの積空間OBJ内の点OBJとして表現できる。なお積空間OBJの距離は、プロパティOBJの距離から合成される。

（２）商空間OBJ

商空間OBJは、ある空間OBJ、Ｆを同値関係～で分割し

た部分集合を点OBJとして持つ空間OBJで、F/～と書く。例えば、名前空間OBJの定義域が漢字名、平仮名の名前、片仮名の名前の３つに分けられるすると、商空間OBJ、(名前/文字種)空間OBJは、漢字名空間OBJ、平仮名の名前空間OBJ、片仮名の名前空間OBJを点OBJに持つ。

このように商空間OBJは、空間OBJを点OBJとして持つので、全く定義域の異なるいくつかの空間OBJ、F_1を点OBJとする商空間OBJを定義すれば、定義域の異なる空間OBJを一般化した空間OBJも定義できる。この商空間オブジェクト内では、空間OBJ間の距離を定義していることになる。

３.２ ＭｅＳＯＤとオブジェクト指向

３.２.１ オブジェクト指向の有利な点

オブジェクト指向については、研究者により様々でありOOPLと言われるものにも大なり小なり相違点が存在するが、ここではSmalltalk[Goldberg]における概念と対比しながら我々のＭｅＳＯＤにおけるオブジェクト概念について述べる。

図２(a)のようにＭｅＳＯＤのOBJとOOPLのオブジェクトとは、ちょうどオブジェクト識別子を原点にして直交したような関係にある。つまり、OBJにとってはOOPLのオブジェクトは、オブジェクト識別子としてしか見えないし、OOPLにとってのOBJも同様である。クライアント(ユーザーやアプリケーション)がこのOOPLを介してＭｅＳＯＤのOBJをアクセスすれば、OBJの構造を意識することなく利用できる。またオブジェクト指向言語のメッセージ実行はＤＢＭＳ構造のモジュール化に有利である。

ＭｅＳＯＤ-ＤＢＭＳの中心的役割は、図２(b),(c)のようにOBJとOOPLのオブジェクトとの融合である。以下では、融合の際に問題となるOBJとオブジェクトの間のいくつかの相違点について述べる。

３.２.２ OBJとオブジェクト

（１）メソッドと距離関数

ＭｅＳＯＤでは、概念モデルの枠組みの中に唯一の手続きとして距離関数の存在を仮定している。この距離関数と点OBJの実現値を知るための関数や空間指定演算子をオブジェクトのメソッドとして呼び出すことによりインターフェースの共通化が図れる。

（２）インスタンス変数と積空間OBJのプロパティ

Smalltalkのインスタンス変数は、オブジェクトの状態や性質を保存するものである。ＭｅＳＯＤにはOBJのインスタンス変数という概念はないが、積空間OBJのプロパティOBJが類似の概念である。しかし、インスタンス変数がオブジェクトの私有のデータ構造として情報の隠蔽がなされているのに対してプロパティOBJには、隠蔽という概念がない。したがってプロパティOBJをどのようにインスタンス変数で実現するかが問題となる。

（３）プリミティブなオブジェクトとシステム規定の空間OBJ

SmalltalkではクラスNumberのインスタンスのように、値そのものがオブジェクトであるプリミティブなオブジェクトがある。我々は、これをシステム規定のOBJ(原始空間OBJという)のインスタンスとして扱う。

（４）クラスと空間OBJ

Smalltalkのクラスのインスタンス変数は、ひとつのオブジェクトとして見たときのインスタンス変数としての性格より、クラスの全てのインスタンスのための共有変数として位置づけられている。しかし、ＭｅＳＯＤでは空間OBJは一つのOBJとして存在できる。(商空間OBJは、特化空間OBJを点OBJとして持つ。)

（５）メタクラスと空間型

空間型がメタクラスの概念に対応する。Smalltalkのメタクラスはその唯一のインスタンスであるクラス無しには存在できないが、空間型は空間OBJがなくても存在できる。また、ＭｅＳＯＤでは異なる空間OBJが同じ空間型を共有することが許される。さらに、空間OBJはオブジェクト識別子(空間名)と空間型との組でありデータベース上のオカレンスであるため、同じ空間名で別の空間型に型付けされた空間OBJを動的に生成することができる。この機能を利用すると、オブジェクトは動的に空間OBJとしての扱いを受けたり点OBJとしての扱いを受けることが可能である。

（６）インスタンスと点OBJ

空間OBJに対する点OBJの概念がSmalltalkにおけるインスタンスに対応するが、このインスタンスは相対的な概念ではなく、クラスのインスタンスはインスタンスを持つことができない。商空間OBJの点OBJは特化空間OBJであるから、点OBJは空間OBJにもなりえる。

（７）インヘリタンスと商写像

インヘリタンスは、スーパークラスからサブクラスへそのインスタンス変数やメソッドを継承する機能であるが、これはあらかじめ定められたクラス階層に従う。ＭｅＳＯＤでは、クラス階層という概念はなく、商写像に基づく商空間OBJにより管理されるため動的なis-a階層が実現できる。詳細については、３.３節で述べる。

３.３ 商空間OBJとクラス階層

ＭｅＳＯＤの商空間OBJは、ある空間OBJを何らかの同値関係で分割した結果として得られる同値類を点OBJとする。このとき、商空間OBJの点OBJを分割元の空間OBJの特化空間OBJといい、分割元を汎化空間OBJという。つまり商空間OBJは、空間OBJの汎化－特化関係を保持している(図３)。Smalltalkにおけるクラス階層はサブクラス側でスーパークラスへの明示的なリンクを持つことにより実現されている。したがってクラスを新しく作り出すときには、常にあるクラスのサブクラスという形を採るので容易に継承階層が作り出せる。しかしその様にし

て作り出した継承階層は、統一的な役割や根拠に基づいた階層ではない。ＭｅＳＯＤの商空間OBJでは同値関係あるいは商写像が汎化－特化関係の根拠を示している。またＭｅＳＯＤでは継承という概念はないが特化空間OBJは汎化空間OBJの部分集合なのでその性質やデータ構造を自然な形で受け継ぐことができる。

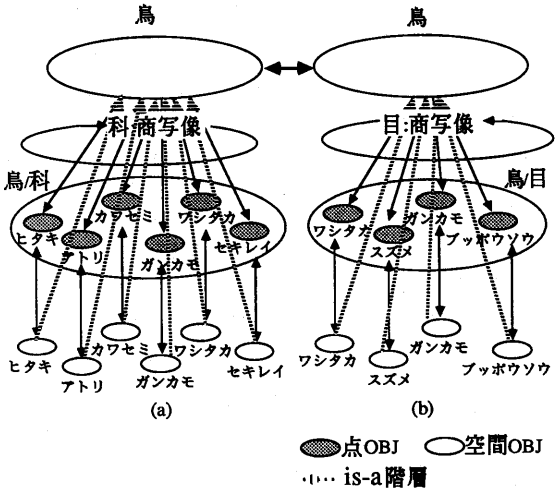


図３　商空間OBJとis-a階層

４．実現のためのソフトウェアアーキテクチャ

４.１ ソフトウェアアーキテクチャ

図１に示したようにＭｅＳＯＤの核は、大きく分けてオブジェクトマネージャとオカレンスマネージャからなる。空間OBJや点OBJの定義は、二つのマネージャとの間のやり取りで行われる。以下に、その大間かな役割について述べる。

（１）オブジェクトマネージャ(OidM)

メディア表現などオブジェクトの内容に依存しない部分であり、ＭｅＳＯＤにおけるオブジェクト概念を実現する。このオブジェクトマネージャがＭｅＳＯＤ-ＤＢＭＳの動作の中心で、ここから他のマネージャの手続きを呼び出すことで機能する。例えば、新たに空間型を作ったり、距離関数を変更したりする場合には、オカレンスマネージャにそのためのメッセージを送る。

（２）オカレンスマネージャ(OcrM)

ＭｅＳＯＤにおける定義域Ｘの意味は、各メディアデータ用のストレージに対応する。つまり、オカレンスマネージャはストレージマネージャを内蔵している。このストレージマネージャの機能を使って距離関数を定義し、空間型を作り出す。これを空間型表に登録することによりストレージ独立性が実現できる。

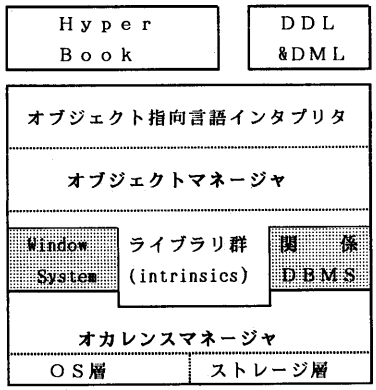


図４　Hyper BookとＭｅＳＯＤ核

４.１.１ 各層の主な役割

図４はＭｅＳＯＤ核の構成とHyper Bookの位置づけを示した図であるが、オブジェクトマネージャとオカレンスマネージャは、さらにいくつかのモジュールに分れる以下に、主な層の役割について説明する。

（１）関係ＤＢＭＳ

この層では、図５のようなＭｅＳＯＤのメタデータを管理する。メタデータ管理の詳細については４.２節で述べる。

（２）ライブラリ群(intrinsics)

これは、関係ＤＢＭＳが管理するメタデータをＭｅＳＯＤの概念単位(空間OBJや点OBJなど)で操作するための関数群でオブジェクト指向的に構成する。

（３）オブジェクト指向インタプリタ

この層は(2)のライブラリ群を用いて構成され、データ定義言語(DDL)やデータ操作言語(DML)からの要求やアプリケーションからの要求を翻訳し実行する。また、ＭｅＳＯＤ核の全ての機能を利用できるとともに、システム拡張用の機能も備えている。詳細は５.４節で述べる。

図１、４のようにＭｅＳＯＤ核をオブジェクトマネージャとオカレンスマネージャとに分離することにより、新しいメディアや非構造データの抽象的扱いが可能になる。すなわち、オブジェクトマネージャはクラスに相当する空間OBJとインスタンスに相当する点OBJしか管理しない。一方メタクラスに相当する空間型とオブジェクトの実現値はオカレンスマネージャ内で管理される。オブジェクトマネージャは、起動時にオカレンスマネージャのメタクラス群をバインドする。このとき、オブジェクトマネージャから空間型を生成することも、オカレンスマネージャ自身が新たに空間型を生成することもできるしたがって、オカレンスマネージャに新しいメディア用のストレージと距離関数を登録すれば、その後はこれを

空間型として利用することができる。

4.2 メタデータ管理

ＭｅＳＯＤのメタデータの管理として既存の関係ＤＢＭＳを利用する。これは、ＭｅＳＯＤのメタデータを関係ＤＢＭＳ上のデータ操作の対象として扱えるのでスキーマの動的変更が容易であることと、表に分割したメタデータの独立的な管理とそれに伴う拡張性を考えてのことである。具体的には、ＭｅＳＯＤにおけるメタデータである各種概念や概念間の関連を関係表の実現値とし、ＭｅＳＯＤの概念スキーマ辞書を関係データベースで構成・管理する。ただし、この際、関係ＤＢＭＳ上ではデータ操作などに拡張、修正が必要となる。

ＭｅＳＯＤ上のＤＤＬで定義されたＭｅＳＯＤデータベース・スキーマを図４のオブジェクト指向言語インタプリタが関係表の実現値に変換し、ユーザーは、これをＭｅＳＯＤ上のＤＭＬを介して利用する。これにより、ＭｅＳＯＤデータベース・スキーマは、ＭｅＳＯＤのＤＭＬにより変更可能になる。また、オブジェクト指向言語インタプリタを直接操作することも可能で、高級言語に組み込んで応用を作ることもできる。

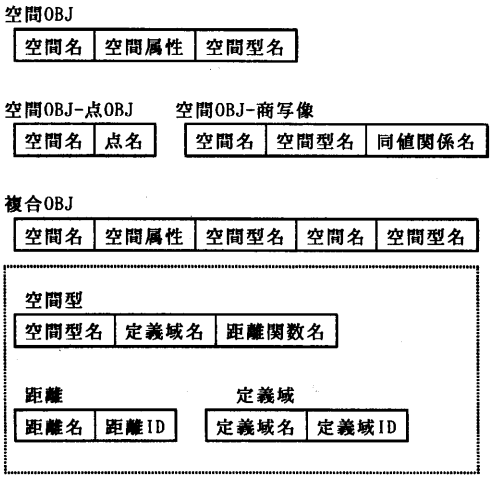


図５　ＭｅＳＯＤのメタデータの関係表現

4.2.1 ＭｅＳＯＤの基本的概念と関連

ＭｅＳＯＤ上の基本的概念は空間OBJ、点OBJ、空間型、定義域、距離である。これらの間の関連としては、空間OBJと空間型との間にinstance-of関連、点OBJと空間OBJの間にelement-of関連(MeSOD-DBMSではこれを特殊なinstance-of関連として扱う)、(定義域、距離)と空間型の間にpart-ofの関連、定義域と距離との間にdomain-of関連がそれぞれ存在する。これを図５のような関係表で表現する。点線枠で囲まれた関係表は、オカレンスマネージャがオブジェクトマネージャとの間のインターフェースをとるために持つ関係表である。

５．ＭｅＳＯＤ核の基本機能

ＭｅＳＯＤ-ＤＢＭＳでは、オブジェクト識別子とオブジェクトの定義域である空間型とを独立して扱い、この両者を結び付けることで空間OBJや点OBJを生成する。以下では、このような機能を提供する各マネージャの基本機能について述べる。

５.１ ＭｅＳＯＤ オブジェクトマネージャ

オブジェクトマネージャは主にオブジェクトの存在を管理する。また、その役割はスキーマ定義/変更要求やデータ操作や質問処理を解析してオカレンスマネージャの機能を呼び出すことである。ここでの操作の中心は、空間OBJや点OBJのオブジェクト識別子をオブジェクトとして提供することにあり、空間名に対する空間型の変更や点OBJと空間OBJのスイッチなどの機能を支援する。例えば、オブジェクトのインスタンス変数には、空間型名を保持するものを持っている。以下では、オブジェクトマネージャの主なオブジェクト操作について述べる。

５.１.１ オブジェクト

オブジェクトは、アプリケーションの定義域である実世界の実体を意味する。オブジェクトはデータベース内ではユニークに存在し、その存在は属性値にはよらないしかし、ＭｅＳＯＤでは同じ空間名で複数の空間OBJを提供できるが、OOPLでは空間名＝オブジェクトとなる。そこで、ＭｅＳＯＤの空間型をオブジェクト内のインスタンス変数に保持する。このインスタンス変数に空間型名が入っている状態が一つの空間OBJである。空間型名をインスタンス変数とすれば、点OBJの実現値を知る関数や空間指定演算子をインスタンスメソッドとして実現できる。これらのメソッドはオカレンスマネージャへのメッセージという形で実現される。このようにＭｅＳＯＤの概念をインスタンス変数としカプセル化することによってOOPLとのインターフェースを提供する。

５.１.２ 主なオペレーション

ここではオブジェクトに対応する空間OBJを提供する方法とオブジェクトの生成や削除などの操作以外のＭｅＳＯＤに固有の操作について述べる。

（１）空間OBJを提供するまでの機能

空間名に対するオブジェクトを空間型オブジェクトにメッセージ(例えばnew)を送ることにより生成する。このとき、このオブジェクトの空間型はインスタンス変数(例えば、spaceType)に保持されている。一つの空間OBJがオブジェクトとして生成されると空間OBJ表にその組が追加される。つまり、空間OBJ表の一つの組がオブジェクトに対応している。以後、空間名＝オブジェクトとしての扱いが可能になる。

（2）オブジェクトのメソッド

(a)点と空間の相互変換 … SpaceToPoint,PointToSpace

点OBJを空間OBJに、空間OBJを点OBJにする機能である。点OBJを空間OBJにする場合、商写像として恒等写像を用いると空間OBJを商空間OBJにできることにより実現する。また空間OBJを点OBJにするには、その空間OBJが何らかの空間OBJの特化空間OBJでなければならない。これは、オブジェクトが持っているクラス変数の状態を変えることで実現する。この場合、点であるか空間であるかを保持する変数を持つ。

(b)空間属性の変更 … changeSpaceAtr

空間OBJの空間属性を変更する機能を持つ。空間型を変更するときに使う。

(c)空間型の変更 … changeSpaceType

現在のオブジェクトの空間型を変更する。これは、オブジェクトと空間OBJの関係を変更することに等しい。変更できる空間型名は空間OBJ表に登録されている同じ空間名に対応するものに限られる。

(d)積空間OBJの構成 … createCartisianOBJ

システム規定の空間OBJか既存の空間OBJを用いて積空間OBJを生成する。この場合、空間型が存在していない場合には、メタ空間型了オブジェクトにdefineTypeメッセージを送る。

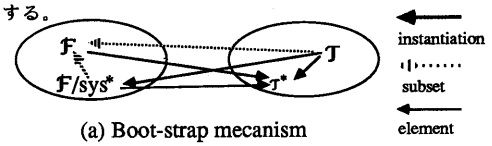
(e)商空間OBJの生成 … createQuotientOBJ

商空間OBJを作り出す機能である。空間OBJを商写像表にある同値関係により分割し、そのブロック(特化空間OBJ)をインスタンスとする空間OBJを作る。この場合、汎化空間OBJにおける実現値は関係データベースのビュー機能により継承できるが、新しいプロパティOBJを追加するような場合には、その様な空間型を作る必要がある。

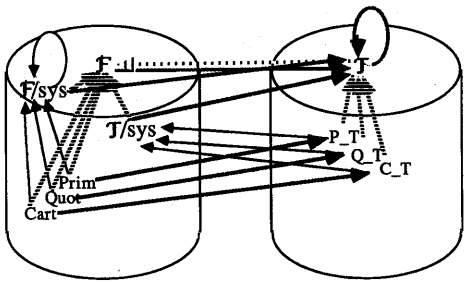
5.2 MeSOD オカレンスマネージャ(OcrM)

MeSOD核のオブジェクト概念は、[COINTE87]に類似している。つまり、オカレンスマネージャは、メタクラスに対応するメタ空間型を最初のオブジェクトとし、このメタ空間型が空間型を生成する。生成された空間型が空間OBJをオブジェクトとして生成する。しかし、オブジェクト概念の枠組みの中で新しい空間型を作るためには、空間型もオブジェクトである必要がある。そこで、[COINTE87]で採用している空間型のブートストラップによりこれを実現する。すなわち、メタ空間型了の骨組みをあらかじめ用意したうえで、オブジェクトマネージャのルートℱを作成し、そのサブクラスとして新しいメタ空間型了を生成する。

図6はその様子を示したものである。ℱ、了はそれぞれオブジェクトマネージャ、オカレンスマネージャを意味する。了*は了を作り出すまでの骨組みでありℱとℱ/sys*を起動した後、了をサブクラスとして作り出す。(b)の了の再帰ループはbootstrapを意味する。商空間OBJ、ℱ/sys*は{ℱ/sys, 了/sys, 了}となっているので、了が了/sysを商空間OBJとして生成し、システム規定の型と積空間型、商空間型を作り出す。また、ℱ/sysは空間OBJの種類を保持する商空間OBJとして了により新しく生成される(ℱ/sys={Prim,Quot,Cart})。ユーザー定義のオブジェクトは、ℱ/sysの点OBJのサブクラスとして定義される。ℱ/sysの再帰ループは、ℱ/sys, 了/sys, 了を点OBJに持つ商空間OBJはℱ/sysの前身であることを意味する。



(a) Boot-strap mecanism



(b)MeSOD核の初期状態

図6 MeSOD核におけるオブジェクトの生成関係

5.2.1 オブジェクト

オカレンスマネージャにおいては空間型を作り出すメタ空間型了(=オカレンスマネージャ)がルートオブジェクトである。しかし、上述したブートストラップ機能によりオブジェクトとしての性質は、オカレンスマネージャのルートオブジェクトℱから継承する。空間型オブジェクトは、空間型名と距離関数名をインスタンス変数に持っている。なお、距離概念の扱いについては5.3節で述べる。

5.2.2 主なオペレーション

（1）メタ空間型了のメソッド

(a)空間型の定義 … defineType

空間型を新たに作り出す。定義域と距離関数集合の組を空間型表に追加することにより実現される。実際には以下のメソッドが距離関数や定義域を作成する。

(b)定義域の登録 … createDomain

新しく定義域をオカレンスマネージャに登録する。この機能によりストレージなどの拡張が行える。

(c)距離関数の定義 … defineMetric

定義域に対する距離関数を定義する機能である。距離値表で実現する場合と、距離を求める手続きになる場合のどちらも同じである。（5.3節参照）

(d)距離の合成 … comp_distance

プロパティOBJの距離関数から直積距離空間の距離を作る。

（2）空間型Tのメソッド

(a)空間指定機能 … specifySpace

空間型オブジェクト内の距離関数名を保持する変数に指定された距離関数名代入し、アクセスの対象を一つの距離空間とする。

(b)距離関数の起動 … callMetric

空間指定メッセージspecifySpaceにより定められた距離関数の機能を呼び出す。

(c)オブジェクトの近傍を得る … neighbor

現在、空間型に保持されている距離空間における任意のオブジェクトの近傍を得る手続きである。

5.3 距離概念の実現

ＭｅＳＯＤでは、距離関数の違いで観点を表現し、距離値で関連度を表現する。ＭｅＳＯＤ-ＤＢＭＳでは、この距離関数をオブジェクト間のメッセージにより起動するので応用とのインターフェース実現方式に依存しない。つまり、距離表の形であろうと距離関数手続きであろうともメッセージ呼び出しという共通のインターフェースを提供できる。その一方、距離関数の実現方式と実行方式が問題となる。

（1）実現方式

(a)オブジェクトの付値から距離関数を作る。

システム規定のオブジェクトについては、あらかじめ付値を求める手続きを用意しておく。これを元に、複合空間OBJの距離を作り出す。

(b)オブジェクト間の距離値表を持つ。

オカレンスマネージャ内に空間OBJに対応する距離値表を持つことで距離関数を実現する。距離値表は、距離関数の数だけ必要になる。しかし、アイコン操作で間接的に距離を作り出したり、学習したりするようなインターフェースには向いている。

（2）実行方式

(a)距離関数デーモン

このデーモンは、空間型表に管理されている距離関数をクライアント(空間OBJ)からの要求に答えるサーバーとして起動するものである。

(b)メッセージ駆動

空間OBJからのメッセージによって距離関数を起動する。空間指定メッセージにより距離関数を起動する方法が考えられる。

5.4 機能拡張用のオブジェクト指向親言語

上述したようなＭｅＳＯＤ核が提供するオブジェクト概念は、核を構成するプリミティブな操作の集まりにより実現される。これらは、オブジェクト識別子の生成、削除、メタデータに対する操作などを実現したものである。このプリミティブな操作は４．２節のライブラリ群に対応しているが、システムの拡張性や開放性を実現する方法として、このような操作群を言語として提供するアプローチ[NeWS][Rich87]を採用する。この言語の上に上述の各マネージャの機能を実現することにより３章であげたＤＤＬ/ＤＭＬのカスタマイズ、マネージャの機能拡張や応用の独立性が実現可能になる。

また、このような言語をインタプリタとしてＤＢＭＳ内部に持つことにより他の高級言語とのインターフェースを取ったりアプリケーションを作成することもできるＭｅＳＯＤのＤＤＬやＤＭＬは、これを用いた一つのアプリケーションとして構成される。

６． おわりに

本稿では、マルチメディア電子化図書システムHyper Bookの核となるＭｅＳＯＤ-ＤＢＭＳのオブジェクト指向概念に基づく設計について述べた。３章で述べたようにＭｅＳＯＤでは、情報実体の存在を実現値とは独立して扱うことデータへのアクセスは距離関数という手続きを介して行うという２点がオブジェクト指向言語(OOPL)との接点であるが、OOPLのオブジェクトとＭｅＳＯＤのOBJは一対多の関係にある。このOBJをオブジェクト指向言語の枠組みの中で利用できるようにするのがＭｅＳＯＤ-ＤＢＭＳの主な役割である。そのために必要な基本機能については４，５章で述べた。現在、SUNワークステーション上のObjective-C,関係DBMSにUNIFY4.0を用いてＭｅＳＯＤ核のインプリメントをしている。なお、メディア依存部として図形の入出力にPC9801VM2を音声の入出力にMacIIを接続している。

参考文献

[TabMur87a]田渕,村岡:"電子化図書構想とデータモデリング",情処35回全大6Bb-5,pp.363-364,1987.

[TabMur87b]田渕,村岡:"距離空間に基づいたデータモデルの提案について～ＭｅＳＯＤ～",情処研報Vol.87,No.66,87-DB-61-1,1987.

[ShShTM88] 芝崎,柴山,田渕,村岡:"距離空間データモデルＭｅＳＯＤに基づくメディア検索について",情処36回全大2E-3,pp.383-384,1988

[COINTE87]COINTE,P.:"Metaclasses are First Class:the ObjVlisp Model",OOPSLA'87,pp.156-165,Oct.,1987.

[NeWS]Sun Microsystems,Inc.:"NeWS Technical Overview",Mar.1987.

[Rich87]Ricahrdson,J.,Carey,M.:"Programming Constr-ucts for Database System Implementation in EXODUS",Proc. ACM-SIGMOD'87,1987.

[Goldberg]Goldberg,A.:"The Interactive Programming Environment",SMALLTALK-80 -対話型プログラミング環境-",相磯秀夫(監訳),オーム社,1986