6H－7

ＪＤＭＦのＳｍａｌｌｔａｌｋによる実現

野口宏*　尾関美明**　穂鷹良介**
*茨城大学　**筑波大学

１．はじめに

データベースの標準化のため、また少ないモデル化概念で高い表現能力を実現するためのデータモデル機能としてＪＤＭＦが提案されている。１９９１年にはその基準がＪＤＭＦ／ＭＯＤＥＬ－１９９１[2]として設定された。

しかし、この基準はデータモデルを定めたものであるので、この基準に対してデータベース管理システム(ＤＢＭＳ)が必要となってくる。ＤＢＭＳの基本操作は検索,挿入,修正,削除の４つであるが、[1]ではそのうちの検索,挿入の機能をＳｍａｌｌｔａｌｋにより実現した。

本稿では、その経験を踏まえてアーキテクチャの一部改良ならびに[1]で行なわれなかった汎化機能を考慮したデータの修正,削除機能の実現を行った事を報告する。

２．ＪＤＭＦ／ＭＯＤＥＬ－１９９１[2]

[2]は一見すると関係データモデルと非常によく似ているが、高い表現能力を持たせるために幾つかの拡張がなされている。その主なものとしては、

(1)「リスト実体型」の導入とその定義域としての使用

(2)「役割」，「定義域(実体型)」，「論理表現型」の明確な区別

が挙げられる。

(1)の「リスト実体型」とは、いくつかの実体型を列を指定し組み合わすことによって一つの概念を表現するものである。

(2)では、今までのデータモデル特に関係データモデルでは、列名,定義域の２つの区別程度しかなかった。しかし、[2]では列の持つ共通の意味を表現するものとして役割を考え、役割と定義域の明確な相違を述べている。このようにする事により、列名は表名に役割と定義域を付けたものとしている。また、ＳＱＬなどでは定義域とその表現形態である論理表現型の区別が曖昧になってしまいがちであるが、その違いも明確にしている。

３．ＪＤＭＦとＳｍａｌｌｔａｌｋとの比較

ＪＤＭＦにおける概念とオブジェクト指向言語であるＳｍａｌｌｔａｌｋにおける概念は、わずかな違いはあるものの非常によく似た概念が多い。

特に、ＪＤＭＦの実体実現値と実体型及びその関係は、オブジェクト指向で言うところのインスタンスとクラスの関係に非常によく似ている。例えば、ある会社の従業員Ａについて考える。Ａさん自身をデータベースの中で管理しようとすると、まずＡさんを実体実現値と考える。そして、その枠組みとして実体型「従業員」を考える。オブジェクト指向ではＡさんをオブジェクトと考え同時にとあるクラスのインスタンスであると考え、最後にクラスを考える。

[1]ではリスト実体型の実現値もオブジェクトとして考え列の値はインスタンス変数を用いる事により実現している。インスタンス変数のとり得る値は再びオブジェクトである性質から、ＪＤＭＦのネストしたリスト実体型も簡単に取り扱える。

ＪＤＭＦの表もリストの集合であると考えられているのでＳｍａｌｌｔａｌｋの集合のサブクラスとしてそのまま自然に表現することができる。

ＪＤＭＦでは管理実体型という概念をデータ共有の中心に据えているがＳｍａｌｌｔａｌｋではこのような考え方はない。なお両者の差については[3]を参照のこと。

４．アーキテクチャ

ＪＤＭＦを実現するにあたり、ＪＤＭＦのメタスキーマのアーキテクチャに対して若干の修正を行った。ＪＤＭＦのメタスキーマには６個の表があるが、ここでは簡単の為にその中の３個の表についてのみ例示を行った(図１参照)。

まず、ＪＤＭＦの表以外に「管理実体表」を新たに設けた。これは、ＪＤＭＦの６個の表をビューと考えるメタ実体表に対応するものである。この管理実体表の中にはＪＤＭＦのメタスキーマに現れる全ての実体が登録される事になる。逆に管理実体表の中に現れるものはＪＤＭＦの６個の表の中に必ず現れていなければならない。

次に、オブジェクト指向のｏｉｄ(Object ID)に相当するものを表の中の値として考えている事に特徴がある。これは、図の中ではサロゲート記号(¢)により表現してある。例えば、「¢表表」には表表そのものが値として入っていると考えられる。しかし表の中に値として表が入っているという事を直接表現することに困難があるので、このような表現法を用いた。図１を見ると表表の中の値として「¢表表」がある。この意味する事は、表表の中には値として表表があり、値としての表表の中に再び値として表表があるというように無限の繰り返しがあることである。図には、アプリケーションのための表の例としてＡ,Ｂを挙げているが、これも表表の中に登録される。つまり表表の中の値として、表Ａ,Ｂが含まれる。

Implementation of JDMF using Smalltalk
Hiroshi NOGUCHI*, Yoshiaki OZEKI**, Ryosuke HOTAKA**
*)Ibaraki University, **)University of Tsukuba

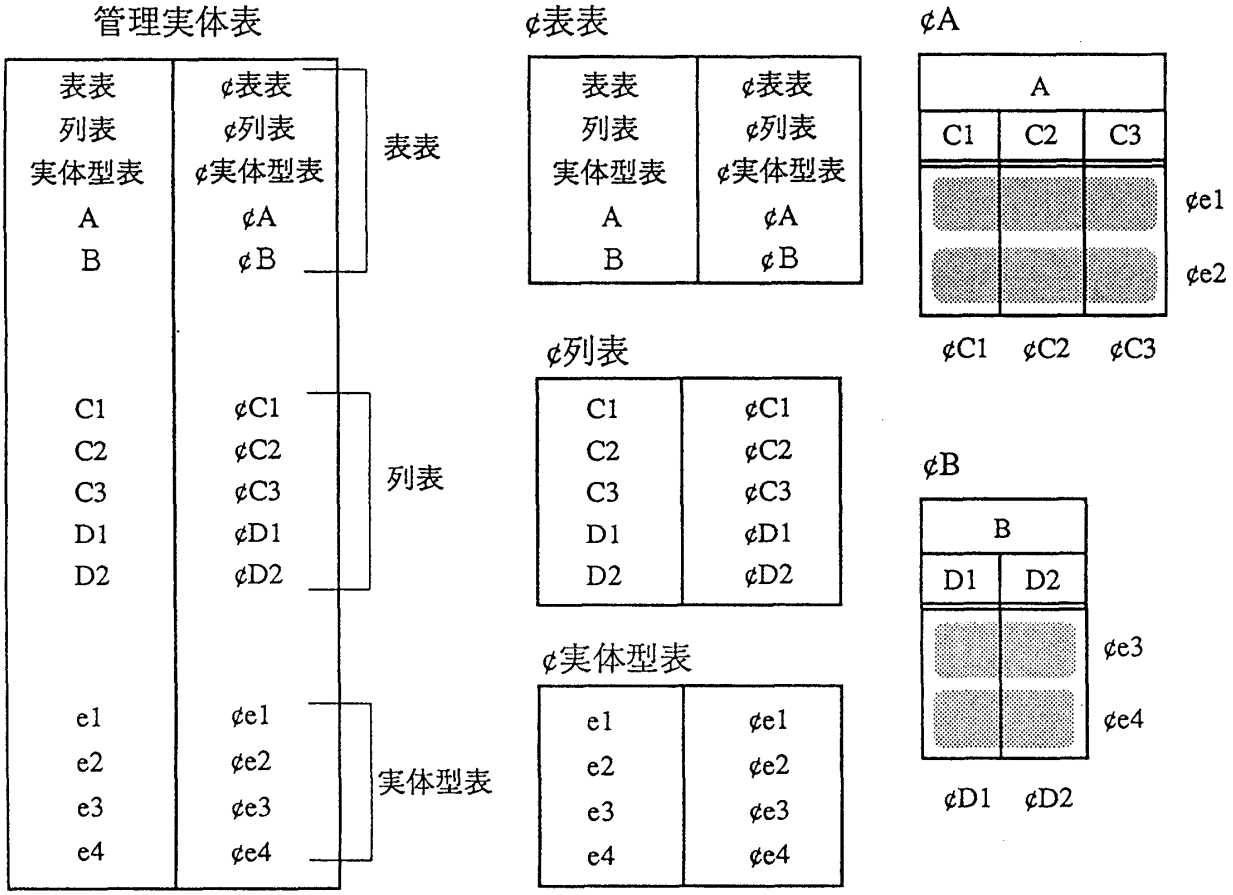
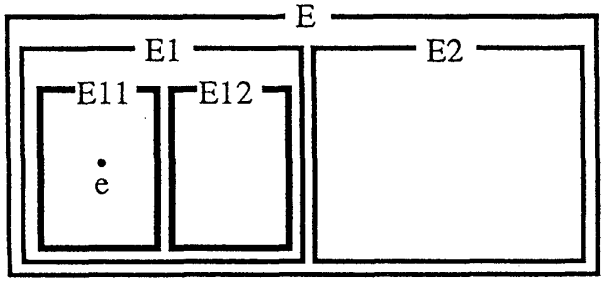


図１　JDMFのSmalltalk上のアーキテクチャ

５．汎化関係を考慮したデータ操作

汎化関係を考慮してＪＤＭＦのデータ操作を考えたとき、[2]の仕様だけではデータの操作を曖昧さなく定義することができないことが判明した。これを解決するために数学の分割概念を適用して、ある汎型の副型を構成する実体型の集合は分割となる制約を設けた。



{E1,E2}は、Eの分割
{E11,E12}は、E1の分割
点・の表現は、(E11,e)

図２　実体型汎化構造

副型の実現値を表現するときには実現値xを含む最小の実体型をeとするとき(e,x)の形で表現することとした。我々が仮定した実体型の汎化概念の構造を図２に概念的に示す。

６．終わりに

今回の実現により、基準[2]を満たしたデータベースに対して、簡単な修正及び削除が行えるようになった。これにより、基準[2]をモデルとするＤＢＭＳの完成へと一歩近づいたことになる。

今後は、複雑な修正及び削除への対応と、現在進められている[2]の改訂に対する対応などが課題として残されている。

参考文献

[1]尾関美明:オブジェクト指向アプローチによるＪＤＭＦデータの管理に関する研究,筑波大学大学院博士課程社会工学研究科経営工学特別演習
[2]日本規格協会:ＪＤＭＦ／ＭＯＤＥＬ－１９９１（データモデル機能ＪＤＭＦ）
[3]穂鷹良介：オブジェクト指向ｖｓ関係データモデル，情報処理学会第４４回全国大会（本予稿集）