

データ主導情報モデリング設計支援システムIMDSSについて

穂鷹良介

筑波大学

hotaka@shako.sk.tsukuba.ac.jp

概要

現代における情報処理の方式は益々多様化の傾向を深めており、情報システム構築の方法論も分野別に別々のものが使用されている。個々の応用分野ごとに特殊事情があって別々の方法論を使用したくなるのは自然ではあるが、対象の種類の違いに応じて処理方式も別々のものを使用していたのでは処理方式の種類数は増加する一方で、やがて複雑な処理に追従できなくなる事態に至ることが懸念される。複雑な情報処理を行うためにこそ、統一された単純化が必要とされる。

あらゆる情報処理はデータを扱うものと考えられる。したがってあらゆるデータを表現できるデータモデルが存在すればすべての情報処理をデータの側面から同一にみなすことができ単純化を実現することができると思われる。本論文はこのようなデータ主導の情報システム設計方法を支援する具体的な手段の構想を述べる。

キーワード：DOA, データ中心設計, データモデル, 情報システム設計, メタデータベース

Data Oriented Information Modelling Design Support System IMDSS

Ryosuke HOTAKA

University of Tsukuba

hotaka@shako.sk.tsukuba.ac.jp

Current information processing methods tend to become versatile and information system development methodologies adopted in various application fields differ from each other. Though it is natural that each application field adopts its own methodology, this will increase the number of processing methods leading to unmanageable complexities. If complex information processes are to be implemented, a simple unified approach is required.

Every information process can be considered to manipulate data. Thus, if there is a data model that can be used to represent every kind of data, we could simplify the data aspect of information processing. This paper proposes a concrete data oriented design support system for information modelling.

Keywords: DOA, data oriented design, data model, information system design, meta database

1. はじめに

現代における情報処理の方式は益々多様化の傾向を深めており、情報システム構築の方法論も分野別に別々のものが使用されている。個々の応用分野ごとに特殊事情があって別々の方法論を使用したくなるのは自然ではあるが、対象の種類の違いに応じて処理方式も別々のものを使用していたのでは処理方式の種類数は増加する一方でやがて複雑な処理に追従できなくなる事態に至ることが懸念される。複雑な情報処理を行うためにこそ、単純化が必要とされる。

本稿ではこの単純化を達成するための一つの情報システム設計方法論を提案し、更にそれを支援するシステム IMDSS (Information Modelling Design Support System) について開発の基本的な考え方を述べる。

まず第一にあらゆる情報処理で用いるデータモデルを一つのものに限定してシステム開発をすることを考える。このためには十分表現能力の高いデータモデルを採用しなければならず非常に困難であるが2および[野口93]に示すようにその可能性がある。

データベースの特長は、ある対象を取り扱う応用分野が複数存在するときでも同一対象の表現は一度だけ行い、それを利用する各分野がその対象に対して独自の見方を有するときには分野ごとにビューを設けるだけで同一対象に対して矛盾のある表現を避けることにある。同一対象に対する複数の矛盾した表現を未然に防ぐことによって情報システムの複雑化を避けている。ここで提案する方法論も当然この特長を活かすものとなる。

同一対象の表現を一つの情報システム内で一意にすることはその情報システムの管理者の責任で可能である。いま一つの対象が管理者の異なる複数の情報システムA、Bで扱われる場合を想

定して見よう。この場合同じ対象の表現がA、Bの両方で一致しない限りAまたはBのいずれかで変換が必要となり複雑さが増す。つまり真の単純化は単独の情報システムの問題ではなく世の中の情報システム全体の問題である。これはデータモデルの問題で2で考察される。

市販のCASEツール（たとえば [SYNON89]）では、設計作業を容易にするために決められた枠内から事前列挙された概念を選択する会話型の定型処理方式が採用されている。IMDSSも同様なアプローチを試みるが、事前列挙の範囲をメタデータの定義とその処理にまで拡大することによって定型処理方式でありながら、柔軟な情報システム設計を可能にする方式を3で提案する。

メソッドの定義は会話型のインタフェースを重視してビジュアル言語PROGRAPHを用いる。ビジュアル言語の使用については[Björn93]で詳しく述べられるが、概要について4以下に述べる。

2. データモデル

2.1 データモデル階層

あらゆる情報システムに現われるデータを一つのデータモデルで扱うことを前提にした場合、そのデータモデルは非常に広範囲な対象を取り扱うことができなくてはならない。それと同時に特定の応用分野でデータモデルを用いたとき不必要な概念の習得を必要としない単純なものでなくてはならない。この一見相反する要求を満たすために採用するデータモデルは単純なモデルから応用分野の複雑度に合わせた特殊化が調和した形での併存を許すものとなっている。

このモデルで最も基本的なものは単にインスタンスとクラスの区別だけを有しているもの（[野口93]の最小データモデル）で従来の意味でのデータモデルよりはるかに一般的な対象を表現する目的に用いることができる。いま人

工知能分野の知識表現にデータモデルを標準化しようという動きが顕在化しているが

([Fullton91], [Genesereth92], [Sowa84]), これらを扱うためには従来のデータモデル固有の概念はむしろ妨げになる。

通常データモデル, たとえば関係データモデルあるいはJDMF ([INSTAC93])などは最小データモデルにいくつかのモデル化概念を追加することによって一般モデルの特殊形として得られる ([野口93])。したがって一般モデルによって記述できる部分はそのまま一般モデルの表現を使用することとし, 一般モデルで記述できない特殊対象のみを応用分野別のあるいは特別な機能を提供するデータモデルのモデル化概念によって表現するアプローチを採用する。いいかえるならば採用するデータモデルは, 一貫した思想の下でのマルチデータモデル体系というべき性格のものである。

2. 2 特徴

採用するデータモデルの体系は[野口93]に詳述されているがその特徴を概括すれば以下の通りである。

(1) インスタンス変数とメソッドの両者を汎化して新たに特徴 (feature) という概念として捉える ([Meyer88])。メソッドはオブジェクトクラスのインスタンスに対してだけ定義される。

(2) 特徴に通常の属性におけると同様に定義域を考える。このことによってメソッドの共有をオブジェクトクラスの汎化階層 (サブクラス, スーパークラスの関係) と独立に記述できる。

(3) 共有されるオブジェクトに関して客観的なオブジェクト設計ガイドラインが存在する ([穂鷹91], [INSTAC93])。

(4) オブジェクトクラスの定義にひきずられてメソッドの定義場所が客観的に定まる。

(5) データモデル機能と応用データモデルの境界が消滅する。データモデルにデータを投入するということはデータモデルであるからには当然の操作であるが, 対象とするデータモデ

ルがメタデータモデルであるときにはデータの投入は新たなデータモデルの定義を意味する。しかし両者は等しくデータモデルのデータと考えられるという意味で判然とした区別はない。世の中で単にデータモデルとってデータモデル機能と, 応用データモデルの両者を区別せず指すことがあることに対応する。

(6) オブジェクトのインスタンスの全体は Objectといわれる一つのクラスであると考え, 同時にそれ (Object) はクラスの全体を一つのクラスとして考えたObject*のインスタンスであると考え。しかしObject*そのものもまた何らかのクラスのインスタンスと考えると Object*をインスタンスとするクラス Object**を再び考える必要がある。このような推論を繰り返す行くと次のような無限系列が必要となる:

Object, Object*, Object**, . . .

あらかじめこの系列を無限個用意しておくことによって新しいモデル化概念を生成することができる ([野口93])。

3. データ主導アプローチ

IMDSSではすべての情報処理は対象に対して操作を施すものという認識に立ち, 対象の把握を先行しそのあとにおおのの操作を設計するデータ主導の考え方を採用する。この考え方は

(1) データの共有

(2) データの共有を基礎としての処理 (メソッド) の共有を達成しようとするときに不可避である。

データの設計を先行させ, すべての情報処理をデータを中心に考えるDOAの考え方は, 不必要なプログラム開発を抑制し, システムの品質低下を回避するものとして期待されている ([堀内92])。

本論文はDOAの考え方をメタデータに対しても等しく適用することによって上記(1),

(2)の目的を情報システム設計作業にも達成しようとするところにある。

たとえば受注などの通常の事務処理では利用者が扱うオブジェクトは「受注」, 「商品」など事前列挙が可能であり, それらに対して行う操作もやはり事前列挙が可能である。事前列挙がされているオブジェクトに対してはたとえば画面にオブジェクトの種類を列挙し, さらに各オブジェクトに適用可能な操作をメニューから選択させるという方式で情報処理が可能である。

類似の方法はたとえばマッキントッシュなどで採用しているGUIの多くのケースで見ることができる。CASEツール ([SYNON89]), ビジュアル言語 ([TGS Systems92])などでも事前列挙されたオブジェクトに対する事前列挙されたメソッドの選択という処理方式が好んで採用されている。

このようなオブジェクトとその操作(メソッド)の事前列挙は, それらのオブジェクト自身, メソッド自身の定義時にも適用可能である。たとえば「受注」, 「商品」を定義するときには業務に現われるオブジェクトクラス1個1個をインスタンスとするメタオブジェクトクラスを事前列挙しておき同時にメタオブジェクトクラスの

インスタンスに対して適用可能なメソッドを事前列挙しておけばよい。

本支援システム開発の目的は, データに対するメタデータを無制限に何階層も許すことにより, ルーチンワーク的な業務の実行だけではなくそれらの設計作業のすべてを事前列挙したものからの選択という方式で実現できることを立証することにある。

この考えを実現するときには肝要と思われる事が二つある。一つは不定回数のメタデータを許すデータモデルであり, これについては2で述べた。実際には日本規格協会が開発中のJDMF-92 ([INSTAC93])に若干手を加えたものを利用することとした。現在のJDMF-92は不定回のメタデータ構造を, 使用する範囲内で制約した形になっているがこれを不定回のメタ構造を有しているデータモデルに拡張して用いる。もう一つはメソッドの定義方法でこれについては4で述べる。[Björn93]ではこれについて技術的な詳細が述べられている。

以下IMDSSで考えているDOAに従う具体的な設計支援方法を簡単化されたモデルで例示する。

EMPオブジェクトクラスに新しいインスタンスを投入するための条件

- (1) EMPオブジェクトがあらかじめ定義されている。
- (2) EMPオブジェクトの属性(インスタンス変数)があらかじめ定義されている。
- (3) EMPオブジェクトクラスに新しいインスタンスを生成するメソッドが定義されている。

EMP		
ENO	ENAME	ADDRESS
120	Ohta	Tokyo
226	Tanaka	Osaka

図1 応用データモデルへのデータ入力

メタデータベースにデータを投入する（応用データモデルを定義する）ための条件

- (1) メタデータベースを構成するクラス (ObjectとAttribute) が定義されている。
- (2) Object と Attribute の属性が定義されている。
- (3) それぞれのクラスのインスタンスを生成するメソッドが定義されている。

(注) 右の図において\$\$xxxはxxxという名前をもつオブジェクトそのものを表わしていると仮定している。

Object	&name	&class
CITY		\$\$Object
EMP		\$\$Object

CITY
CITYNO CITYNAME

EMP
ENO ENAME ADDRESS

Attribute	&name	&primer	&domain	&class
CITYNO	\$\$CITY	\$\$integer	\$\$integer	\$\$Attribute
CITYNAME	\$\$CITY	\$\$char	\$\$char	\$\$Attribute
ENO	\$\$EMP	\$\$integer	\$\$integer	\$\$Attribute
ENAME	\$\$EMP	\$\$char	\$\$char	\$\$Attribute
ADDRESS	\$\$EMP	\$\$char	\$\$char	\$\$Attribute

図2 応用データモデルの定義

図1と図2とを比較して見ると明らかのように一方は通常のデータベースに対するデータの入力であり、一方はそのデータベースのスキーマ自身を定義するアエタレベルが一段上の概念操作であるが、形式的には両者はまったく同じである。これはデータモデル機能も応用データモデルも区別なくオブジェクト階層の一部に同じ考え方で位置付けるといって単純化によって得られるメリットである。

上記ではクラス Object に対してインスタンスの生成を行ったがクラス Object よりも一段メタレベルの高い ObjectClass([野口93]では Object* と表現した) に対してインスタンス生成のメソッドが定義されていれば、データモデル機能あるいはより高次な機能を形成するオブジェクトを作り出すことができる。

このときどのようなデータモデル機能が定義されるかその意味を決定するものは、生成のとき指定できるメソッドである。IMDSSでは当分の間、メソッド自身の定義は4で説明するように人間が定める。

4. メソッドの定義

現在のところこれは何らかの形でプログラミングに依存せざるを得ない。本支援システムではこのためにビジュアル言語 PROGRAPH ([TGS Systems Ltd. 92])を用いることとした。PROGRAPHは言語要素があらかじめ事前列挙されており、プログラムの作成はそれらの中から適当なものを選択するという形になっていて我々が目指そうとしているシステムの思想に近い。

PROGRAPHはコンパイル可能な言語なので、メソッドを定義するときに扱うオブジェクトクラス、そこに現われる属性等が完全に決まっていることが必要である。したがってまったく新しいオブジェクトクラスを動的に生成して行くような用途にはPROGRAPHは向いていないのではないかという不安がある。

まだ実験を通じて完全に実証できたわけではないが、通常の情報システム設計の業務で動的に新しいオブジェクトクラスを生成するメソッドそれ自身を生成しなくてはならないということはほとんどないと予想している。キチンとした設計方法論であれば必ずどのような意味でオブジェクトのインスタンスを生成するかということが事前に判明していて、そのようなメソッドをあらかじめ作成するフェーズを一段前に置くことができるからである。

5. まとめ、残された問題

あらゆる情報処理に現われるデータを例外を置かず統一的に表現できるデータモデルを基礎に、すべての情報処理をオブジェクトに対してメソッドを適用するという見方だけで取り扱う方法論を提案した。

この見方は単に通常データベースに対してデータを投入するときだけではなく、適切なメタデータの存在を仮定すれば情報システムの殆ど全域に現われる設計問題に対して同様に適用できるのではないかという予想を述べた。

この方法論は作業の前にオブジェクトと関連ある特徴（インスタンス変数とメソッドの汎化概念）の事前列挙を前提とするが、この制約は実務的には問題がないことを主張した。

事前列挙は、本方法論で必要とされる前提であるが、同時にそれはデータあるいはメソッドの共有を目指すときには避けられないことと思われる。

本支援システムで作成される設計の成果物はすべて本システムが使用するデータモデルのデータとして蓄積される。それは一般の応用データモデルの運用のときにデータベースにデータを蓄積するのと本質的に同種の作業で、ここでも情報処理の考え方の単純化ができることを主張した。すべてデータの扱いはデータのインスタンスの投入もしくは変更とするアプローチを採用しているため、設計作業と通常のデータ処理とで考え方を変更する必要はないだけでなく、別プロセサの開発の必要もないことを示した。

オブジェクト指向でマルチインヘリタンスによって既存のメソッドを共有するアイデアが出されているが本システムではメソッドに定義域概念を持ち込みデータベースにおけるデータ共有メカニズムと類似の方法を提唱した。

データとメタデータの扱いを同様にしたことによってどのメタレベルにデータを位置付けるべきかという問題が自然解消する。たとえばIRDSの実際の適用に関してCASEツールのリポジトリをIRDS ([ISO93]) で定義した応用レベル対、情報資源辞書レベル対、情報資源辞書定義レベル対のいずれに位置付けようかという問題が生じるが、IMDSSの立場ではどこに位置付けても差し支えないし、どこに位置付けられてもその処理に本質的な差はないので単なる実装上の問題とみなすことができる。

IMDSSを実際に応用するとき、更に将来以下のような問題を解決しなくてはならないであろう。

- (1) あるメタレベルに位置付けられたデータモデルを別のメタレベルに移し変える処理
- (2) 既存のオブジェクトのクラス間の移動
- (3) 一定の時間間隔ごとにあらかじめ定義されている条件が成立しているかどうかをチェックするメソッドを考えることによるアクティブデータベースの実現

IMDSS プロジェクトを以上の考え方のもとに今後数年間の時間をかけて具体化して行く所存である。

参考文献

[Björn93] Björn, M.: Virtual Class-Generation in Visual Language, 第9 2回データベースシステム研究会

[Fulton91] Fulton, J.A.: The Semantic Unification Meta -Model: Technical Approach, ISO TC184/SC4/WG3 N81.

[Genesereth92] Genesereth, M.R. and Fikes, R.E.: Knowledge Interchange Format Version 3.0 Reference Manual, Computer Science Department Stanford University, Stanford, California 94305

[ISO93] ISO/IEC 10728: Information Resource Dictionary System (IRDS) Services Interface

[穂鷹91] 穂鷹良介: 標準データモデル機能のモデル化概念, アドバンストデータベースシンポジウム, 情報処理学会, Tokyo, December 5-6, 1991, pp.71-76.

[堀内92] 堀内 一: 企業情報システムにおけるデータ中心手法導入の要件, 情報処理学会論文誌, 第33巻第4号, pp.521-531(1992).

[INSTAC93] 情報資源スキーマ調査研究委員会: データモデル機能JDMF-92 v1.3, 1993.2.16

[Meyer88] Meyer, B.: Object-oriented Software Construction, Prentice-Hall, 1988.

[野口93] 野口 宏, 穂鷹良介: JDMF-92におけるクラスメソッドおよびクラス変数の扱いについて, 第9 2回データベースシステム研究会

[Sowa84] Sowa, J.: Conceptual Structures: Information Processing in Mind and Machine, Addison-Wesley 1984

[SYNON89]SYNON LTD.:An Introduction to Synon/2E, CASE for the IBM AS/400, (V2,0/V9,0), 1989.

[TGS Systems Limited 92] The Gunakara Sun Systems, Ltd.: prograph 2.5 Tutorial, 1992.