2F−5

DB流通におけるデータ変換方式について*

星野 隆† 石黒 正典‡ 井戸 正幸§

NTT 情報通信網研究所¶

1 はじめに

従来, 企業では業務毎にシステム化を行ったため, 業務等の違いにより個別にデータベース (DB) を作成した.

近年, 業務連携が重要な課題となり, 個別作成されたDBのデータを共有し多目的に利用したいという要求が生じている [1].

この要求を実現するため, データの発生源となる原本DBのデータを, これを利用したいDBへ流通する (これを「DB流通」と呼ぶ.), DB流通システムが必要となる [2].

しかし, DB流通を実現するのは容易ではない. DB流通と同様, 複数のDB間でデータ / スキーマの共用化を行う Federated Database 等で, DB間の異種性が問題となっている [3], [4]. DB間の異種性とは, DB間でのデータ, スキーマ間の種々の相違のことである. たとえば, DBが個別に作成されたため, 各DBが使用しているDBMSが異なっている場合がある. また, 関連したデータ項目が一方では1つのレコード内に存在しても, もう一方は2つのレコードに分かれているように, 同一実体を示しているデータであったとしてもデータ構造が異なる等, DB間で同一の表現にはなっていない. さらに, 同じ意味を示しているデータ項目であっても, 型, 精度等が異なっている. そのうえ, データの意味が同一ではないが類似した意味を持つデータであったり, データを表現するコード体系が異なる等, データ値が異なっている場合がある.

このようなDB間の異種性を解決しDB流通を実現するために, 流通させたいデータをそれぞれのDBに適した形式に変換する, データ変換が重要である.

本稿では, 「DB流通基本システム [2]」の構成要素となるデータ変換処理を実現する, データ変換方式を提案する (図1).

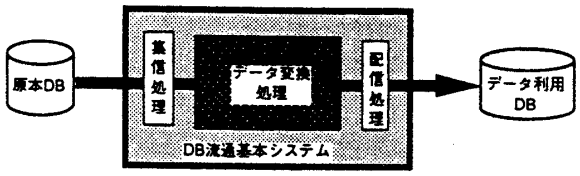

図 1: DB流通基本システム概略図

2 DB流通におけるデータ変換

業務連携の要求に応え, すでにDB流通を実現するDB流通システムは作成され始めている. 業務のシステム化と同様, DB流通システムも業務連携が必要なところから順次作成されるため, 流通の対象となるDBの組ごとに, 個別にDB流通システムが作成されてしまう. ソフトウェアの共用化を行わず, このような個別作成を行うと, 全社的に見たDB流通実現のためのソフトウェアの作成工数は膨大なものとなってしまう.

特にデータ変換処理は, DB流通システム作成工数の大部分を占めており, このデータ変換処理の共用化を行い, 作成工数を削減することは重要である. 実際, 弊社で実現したDB流通システムでは, データ変換処理の作成工数は全工数の約50%を占めていた.

このような, ソフトウェアの作成工数を削減するためには, 1) ソフトウェアの部品化と, 2) ソフトウェア部品を用いて簡易に記述可能な記述言語の策定が重要である.

筆者らは, 具体的方法論として, 問題領域 (Domain) を限定することにより, 1) 共用可能なソフトウェア部品を比較的少数に絞り込むことができ, 2) それらの組合せによりその問題領域で必要とするほとんどの機能を実現できる, Domain Analysis に基づく部品化手法 [5] を採用し, これに基づき伝送設備管理という Domain について, 実際にDB流通を行っているシステムの分析を行った. この結果,

1. データ変換のほとんどはデータ変換部品を用いて変換可能,
2. 複雑な変換もデータ変換部品の組合せで記述可能,

となり, 部品化を行うことにより, 少数の基本的な部品とその組合せでデータ変換処理が実現可能であるという見通しを得た.

本稿では, この見通しに基づき, データ変換部品と, その組み合わせを記述するデータ変換定義言語を用いたデータ変換方式を提案する (図2).

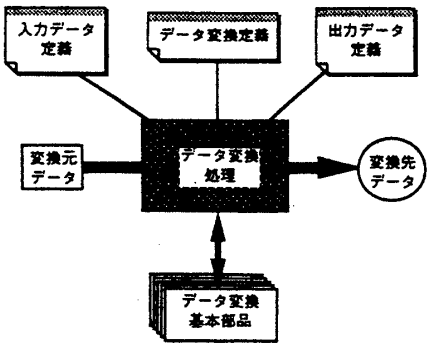

図 2: データ変換方式概略図

3 データ変換基本部品

一般にソフトウェアの部品化では, 1) 機能的に互いに独立した, 2) 共用性の高い部品を選択することが重要である. そこで, 本データ変換部品は以下の方法で選択する. まず, Domain を「伝送設備管理」として, そこで実現されているDB流通で用いているデータ変換ロジックを部品化し, 部品のリストアップを行う. 次に複合的な変換部品をより原子的な変換部品へと分解する. 最後に, 原子的な変換部品について整理・統合を行う.

このようにして求めた, データ変換を行う原子的な部品をデータ変換基本部品と呼ぶ. データ変換基本部品による変換の最小単位はデータ項目とする (データ項目レベル). さらにレコード間にまたがる操作や, レコード同士の関係を変換する必要から, レコードを単位とするデータ変換も可能とする (レコードレベル).

本データ変換部品の特徴は, データ項目の属性を変換する型・精度変換が可能な他に, データ値の変換を可能とするための文字列演算, 数値演算, ロジックで記述できない変換を行なうためのデータ変換表を用いた変換を可能とする. さらに, 複数のレコードに存在するデータ項目を参照するための変換部品により, データ構造の相違の

*A Data Conversion Method among Heterogeneous Databases.

†Takashi Hoshino
‡Masanori Ishiguro
§Masayuki Ido
¶Network Information Systems Labs., NTT.

変換も可能とする．また，レコード間の順序関係が必要なCODASYL型DBでのデータの追加，変更，削除を可能とするための レコードのsort機能等，DBに特有の制約を解決するためのデータ変換部品がある．加えて，DB流通システムの配信先[2]をデータ値により選択可能とするレコード振り分け機能が必要である．

このデータ変換基本部品一覧を表1，2に示す．Domainを限定し，データ変換部品の共用化をはかったことにより，データ変換基本部品は数十個のオーダという比較的少数のものとすることができた．

表 1: データ変換基本部品(レコードレベル)(抜粋)

機能	説明
レコード選択	データ変換に必要なレコードを選択
レコード振り分け	キー値を基にレコードを 異なる変換先へ振り分け
join	同一キー値を持つレコードを結合
sort	キー値の昇順 / 降順にレコードを並べ替え

表 2: データ変換基本部品(データ項目レベル)(抜粋)

	機能
型・精度変換	整数 ⟶文字列(altic())変換 文字列⟶ 文字列(altcc()) 変換 文字列⟶ 整数(altci()) 変換
数値演算	加，減，乗，除(+, −, ×, ÷) 剰余(mod())
文字列演算	連結(couple()) 部分列切出(substr()) 右詰め(rpack())
変換表変換	変換表変換(altcode())
無変換	代入(move())

表 3: 個別にデータ変換処理を作成した場合

流通システム	項目数	記述量	1項目あたり
A ⟶B	676	145.9 k step	215 step

表 4: データ変換定義言語で記述した場合

流通システム	項目数	記述量	1項目あたり
A ⟶C	563	10.6 k step	18.8 step
A ⟶D	1018	19.9 k step	19.5 step

A: 原本DB
B, C, D: データ利用DB
すべてのDBは，同一Domainに属する．

4 データ変換定義言語

データ変換処理の記述は，入出力データ定義と，変換を実行するためのデータ変換定義から構成される．

入出力データ定義は，データ変換処理の前後のDBのデータ構造の定義を行う．

データ変換定義は，データ変換が入出力関係を表現するものなので，入出力関係を明確化することだけで簡単に記述できることが望ましい．このため関数型言語とする．

言語スタイルとの整合を考慮して，データ項目レベルのデータ変換基本部品をそれぞれ1つの関数とし，データ変換基本部品を制御する種々のパラメータは，関数の引数とする．またレコードレベルのデータ変換基本部品は，関数で表すものと構文として表現するものがある．

データ変換定義では，変換元データと変換先データの対応づけ，データ変換基本部品を適用する条件と，その適用順序等を宣言的に定義する．データ変換を行う対象の大きさによってデータ変換基本部品のレベルが異なることから，データ変換定義もデータ項目レベル，レコードレベルに分けて定義を行う．

言語仕様の一覧(抜粋)を図3に示す．

- データ変換定義(レコードレベル)
  MAP < 変換先レコード名 > FROM < 変換元レコード名 >
  [, < 変換先レコード名 > ...] [WHERE < 選択条件 >]
- データ変換定義(データ項目レベル)
  < 変換先データ項目名 > := < 変換定義 >
  < 変換定義 > ::= < 関数 > | < 選択フォーム >
  < 選択フォーム > ::= { IF (< 選択条件 >) THEN < 変換定義 > }
  ... [ELSE < 関数 >]
  < 関数 > ::= < 関数名 > ([< 式 > [,< 式 > ...]])
  < 式 > ::= < 項 > | < 式 > < 加減演算子 > < 項 >
  < 項 > ::= < 因子 > | < 項 > < 乗除算演算子 > < 因子 >
  < 因子 > ::= < 定数 > | < データ項目名 > | < 関数 > | (< 式 >)

図 3: 言語仕様(抜粋)

5 評価

データ変換実現の作成工数の評価のために，データ変換処理実現に要するユーザのプログラム / 定義の記述量を比較した．同一Domainに属するDB間でのDB流通を対象とした評価結果を表3，4に示す．本稿で提案したデータ変換基本部品およびデータ変換定義言語を用いることで，1項目あたりの記述量を約1/10に減少させることができた．

また，同じDomainに属するDBであれば，データ変換定義言語で記述するDB流通が異なっていても，1項目あたりの記述量はほぼ同程度となっている(表4)．これから，このDomain内での部品化は十分行われていると考える．

6 おわりに

本稿では，DB流通を実現するためのDB流通基本システムの構成要素となる，比較的少数の共用化可能なデータ変換基本部品と，データ変換部品の組合せによりデータ変換処理を容易に記述可能とするデータ変換定義言語を用いて，ユーザが少ない工数で容易に実現可能なデータ変換方式を提案した．

本稿ではDB間の異種性に対して，データ変換を行う以前に行われるDB間のデータの対応づけや，データ変換ロジックを決定するという問題は扱っていない．効率のよい異種性の分析・対応関係の導出方法を早期に確立することを考えている[4]．

さらに，本稿で示したデータ変換基本部品について，今回分析を行ったDomainとは異なるDomainのDB間のDB流通について分析を行い，部品の種類を充実する予定である．

参考文献

[1] 堀内 一，データ中心システム設計，オーム社，1988.

[2] 池田哲夫，石垣昭一郎，村田達彦，“DB流通の基本方式について”，情報処理学会第46回全国大会，1993.

[3] A.P.Sheth, J.A.Larson, “Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases”, ACM Computing Surveys, Vol.22, No.3, 1990, pp.183-236.

[4] 石黒正典，坂田哲夫，大沼守一，“異データベース間におけるデータマッピング手法の提案(1) — 手法確立へのアプローチ —”，情報処理学会第45回全国大会，1992.

[5] G.Arango, “Domain Analysis Concepts and Research Directions”, Domain Analisys and Software Systems Modeling, IEEE Computer Society Press, 1991, pp.9-26.