データ流通プラットフォームシステム：DB-STREAM

池田哲夫[†1] 伊土誠一[†2]
石垣昭一郎[†3] 村田達彦[†4]

多くの企業において，同一データが複数のデータベースに重複して投入・格納され，投入稼働の重複，投入漏れ/投入誤りの発生などの問題が生じている．この問題に対し，重複したデータのうち1つのデータの更新に基づき，他の重複データを自動更新するシステム（データ流通システム）を，既存のデータベース処理とは別のものとして付加することによって解決する現実的な方法がある．このようなデータ流通システムを個別にプログラム開発すると，開発期間・開発稼働の増大や流通先の追加などの拡張性に乏しいという問題が生じる．筆者らは，様々な形態のデータ流通システムを簡易に構築できるようにする汎用的なデータ流通プラットフォームを確立することを目的に，(1) データ流通処理モデル，ならびに (2) これに沿ったデータ流通プラットフォームを提案する．データ流通処理モデルでは，データベースのデータの整合性を保証するために流通ユニットと呼ぶ流通データの単位を導入し，様々なデータ異種性を解決するデータ変換機能を明確化した．さらに，流通に必要な機能をソフトウェア部品として備え，流通の動作記述を行う高水準仕様記述言語を有するデータ流通プラットフォームを提案した．試作したデータ流通プラットフォーム DB-STREAM を実際のデータ流通システムに適用し，大幅な開発稼働の削減効果を確認し，上記のデータ流通処理モデルおよびこれに沿ったデータ流通プラットフォームの有効性を確認した．

Data Delivery Platform System: DB-STREAM

Tetsuo Ikeda,[†1] Seiichi Ido,[†2] Shoichiro Ishigaki[†3] and Tatsuhiko Murata[†4]

Recently, many companies have been plagued by redundant data entry and storage in different databases (called the redundant data problem). To address this problem, a practical solution was devised which adds to existing database systems, a data delivery system that automatically updates data from a source database to other databases. However, if this kind of data delivery system is developed individually, problems will arise such as increases in development time and overhead, and increased difficulty in system expansion such as adding a target database system. Aiming to create a common delivery platform that would facilitate construction of various data delivery systems, we propose a data delivery processing model and its corresponding data delivery platform. In the proposed data delivery processing model, a new concept is introduced called the delivery unit which ensures data integrity, and a precise definition of a family of data transformation functions is given. Furthermore, the proposed data delivery platform has a powerful set of software modules and a high-level system specification language for specifying data delivery system behavior to facilitate user development of these systems. An experimental data delivery platform, DB-STREAM, was applied to an actual data delivery system and a decrease in the overhead for large-scale development along with the practicality of the above data delivery processing model and its corresponding data delivery platform were confirmed.

†1 NTT 情報通信研究所
NTT Information Systems and Communications Laboratories
†2 NTT ソフトウェア研究所
NTT Software Laboratories
†3 NTT OCN 事業部
NTT OCN Service Division
†4 NTT 法人営業本部
NTT Business Communications Headquarters

1. はじめに

リレーショナルデータベースの普及にともない，データベース（これ以降，DB と略す）はより身近な存在になり，DB の非専門家でも DB を手軽に作成するようになった．しかしその結果，多くのデータが複数の DB にそれぞれ固有の形式で重複して格納されるという状況が生じている．データの重複格納には，バンキ

ングシステムの勘定系DBと情報系DBとの例のように，意図的にデータを重複して格納することにより，DBシステム全体の性能向上を図った戦略的な重複格納と，DBを構築する部門間の連携が悪かったなどの何らかの事情で無秩序にDBを構築してきたために結果的に重複してしまった無駄な重複格納とがある．後者については入力稼働やシステムコストに悪影響を及ぼすのみならず，データの信頼性の低下を引き起こす要因になっている（以下，後者のケースを重複格納問題と呼ぶこととする）．

重複格納問題は，データ中心設計[1)]の考え方に沿ってあらかじめ企業として必要なデータをすべて抽出・整理し，全社的な統一スキーマを作成してからDB構築に着手すれば起こることはない．しかしながら，多くの企業では多数のDBシステム（大企業では数千に及ぶ）で膨大なデータ項目（同数十万に及ぶ）を保有しており，さらに企業活動で必要なデータはつねに変化しており，全社的な統一スキーマの作成は現実には不可能といわざるをえず[2)]，システム更改時などに部分的に統一する努力がなされているのが現状である．

一方，必ずしも全社的な統一スキーマを必要としない連邦的統合DB[3)]を利用した解決法が考えられる．すなわち，個々のDBの自律性を尊重しつつ仮想的に統合する方法である．しかし，個々のDB管理システム間での異種性（質問言語，トランザクション管理機構などの異種性）が大きい場合には，現状では連邦的統合の実現機構は存在せず，また新たな機構の実現も困難である．

これに対し，既存DBシステムはそのまま活用しつつ，重複したデータを複数DB間で流通させることにより問題を解決する現実的方法がある．すなわち，1つのデータの更新に基づき他の重複データを自動更新するシステム（データ流通システム）を，既存のDB処理とは別のものとして付加する．本方式は，流通するデータに関してのみ共通的なスキーマを作成すればよいので，全社的な統一スキーマの作成が不要であり，また既存DBシステムへ影響をほとんど与えることなく重複格納問題を解決可能である．この考え方に基づいたデータ流通システムが構築されているが，それらは汎用言語で個別にプログラム開発されており，開発期間・開発稼働が大であり，また流通先の追加などの拡張性に乏しいという問題がある．

筆者らは，様々な形態のデータ流通システムを簡易に構築できるようにする汎用的な基盤（以降，データ流通プラットフォームと呼ぶ）を確立することを目的に，(1) データ流通処理モデル，ならびに (2) これに沿ったデータ流通プラットフォームを提案した．さらに，試作したデータ流通プラットフォームを実際のデータ流通に適用し，上記 (1)，(2) の効果を検証した．

データ流通システムを実現するためには，流通先のDBのデータの整合性の保証，データ異種性の解消などの課題があり，さらにデータ流通システムを短期間で構築できるようにする必要があるが，これらに関する研究は見当たらない．筆者らが提案するデータ流通処理モデルでは，整合性を保証するために流通ユニットと呼ぶ流通データの単位を導入し，様々なデータ異種性を解決するデータ変換機能の明確化を行った．さらに，流通に必要な機能をソフトウェア部品として備え，流通の動作記述を行う高水準な仕様記述言語を備えるデータ流通プラットフォームを提案している．

以下，2章では，様々な環境におけるデータ流通システムに対する要求条件について，3章ではデータ流通システムを実現するための基本となるデータ流通処理モデルについて，4章ではデータ流通プラットフォームおよび試作したDB-STREAMについて，5章ではDB-STREAMの実際の適用例を通じて，データ流通処理モデルとデータ流通プラットフォームの効果について述べる．

2. データ流通システムへの要求条件

本論文では，**図1**に示すように，データ流通システムを，あるDBシステム（流通元DBシステムと呼ぶ）でデータの変更（挿入/更新/削除）が発生した場合に，それらのデータを利用する他のDBシステム（流通先DBシステムと呼ぶ）へデータを反映するシステムと定義する．なお，以降では「流通元DBシステム」および「流通先DBシステム」をそれぞれ単に「流通元」および「流通先」と略して使用する．

データ流通システムへの要求条件は，データ流通システムの処理機能に対する要求条件と，データ流通システムの開発に対する要求条件に大別される．

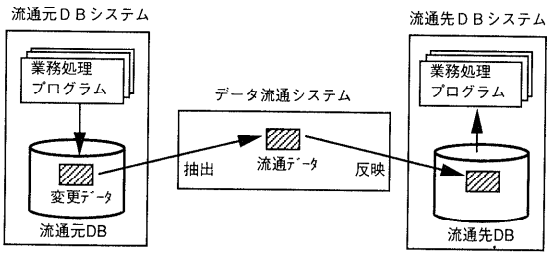


図1 データ流通システム
Fig. 1 Data delivery system.

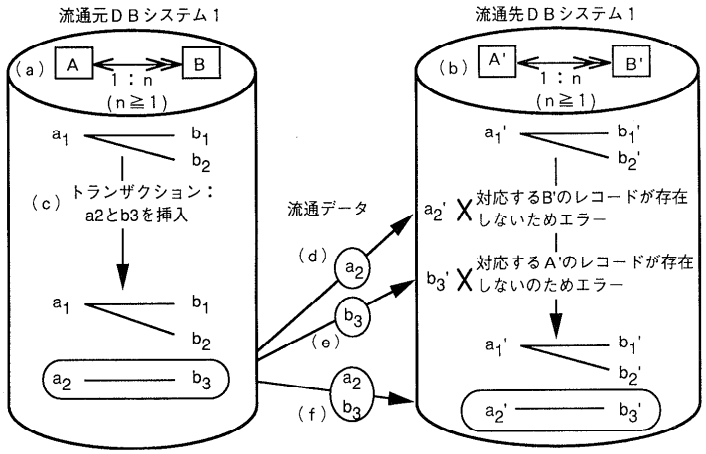


図2 複数の差分レコードをまとめて送る例
Fig. 2 Example of delivering multiple updated records in one unit.

2.1 データ流通システムの処理機能に対する要求条件

2.1.1 流通データの集合化

DBは，現実世界をマッピングしたものであり，現実世界の実体間に存在する意味制約もDB中のレコード間の意味制約にマッピングされてある．データ流通において重要なことは，流通されるレコードが流通先DBでの意味制約を満たすように確実に反映されることである．そのためには，ある意味制約を満たす一群のレコードを，一まとまりのデータとして流通させることが必要であり，これを流通データの集合化と呼ぶ．

たとえば，**図2**(a)のように流通元1にレコードA，Bが存在し，A，B間には参照関係の基数が一対多という意味制約が存在するとする．一方，流通先1でも図2(b)のように流通元1と同様に，レコードA'，B'が存在するとする．ここで，図2(c)のように，流通元1の1トランザクションにおいて，意味制約を満たす2つのレコードオカレンス（a_2，およびa_2の参照キー値を含むb_3）を挿入することを考える．トランザクション完了後に，図2(d)あるいは(e)のように，変更データであるa_2とb_3を別々に流通すると，流通先1では相手レコードが存在しないためエラーとなり，DBの矛盾発生は回避できるが，DBへの反映を行うことはできない．この場合には，図2(f)のように，a_2とb_3をまとめて流通させることにより，流通先1において意味制約を満たすことの確認および反映が可能になる．

このように意味制約を満たすようにデータ流通を行うためには，流通元のトランザクション内で変更されたレコード（差分レコード）をまとめて送り，流通先がまとめて受け取れるようにする必要がある．

2.1.2 データ異種性の解消

本論文では，実世界の同一の実体に対する，各DBでのデータの表現形式の違いをデータ異種性と呼ぶ．既存DBシステムは業務の効率化を目指しているため，DB設計は他のDBシステムとは独立に行われることが多く，そのためDB間で種々のデータ異種性が生じることがある．

データ異種性の分類については，マルチDBシステム[3]において，スキーマ定義言語/質問言語に要求される記述能力を明らかにすることなどを狙いとして，関係型モデルにおけるデータ異種性を分類したKimの分類[4]（**表1**参照）が知られている．なお，2.1.1項で述べた意味制約に関連する異種性は表1から除外してある．本論文では議論の範囲を関係型モデルに限定して進める．

データ流通システムでは，表1にあげたデータ異種性を解決する機能が要求される．

ここで，表1のうち，項番8「誤入力データ」は，本来DBシステム側で解決すべき問題であり，また項番9「陳腐化データ」は，データ流通によって自動的に解決される問題であるため，データ流通システムで解決すべき異種性から除外できる．

なお，データ異種性解消にあたっては，標準データ形式（データ項目が標準準拠のデータ形式）を介した2段階の変換を行うことが望まれる．

2.1.3 流通処理量の削減

流通処理時間の短縮のため，および流通処理の負荷削減のためには，全体的な流通処理量を削減することがデータ流通システムに要求される．

表1 データ異種性の分類
Table 1 Classification of data heterogeneity.

項番	異種性の分類 大分類	中分類	小分類	説明	Kimの分類との対応
1	スキーマ的異種性 注1)	1レコード対1レコード	名称の差異	同名異義あるいは異名同義.	I.A.1.a Table name conflicts
2			構成の差異	属性の多寡.	I.A.1.b Table structure conflicts
3		多レコード対多レコード		DBシステム間でレコードの対応が1対多あるいは多対多.	I.A.2 Many-to-many table conflicts
4		1データ項目対1データ項目	名称の差異	同名異義あるいは異名同義.	I.B.1.a. Attribute name conflicts
5			データ型の差異	例:CHAR対INTEGER	I.B.1.c.1 Data Type conflicts
6		複数データ項目対複数データ項目		DBシステム間でデータ項目の対応が1対多あるいは多対多.	I.B.2 Many-to-many attribute conflicts
7		レコード対データ項目		片方のDBシステムでのレコードが他方のDBシステムのデータ項目に対応.	I.C Table-versus-attribute conflicts
8	データ的異種性 注2)	誤データ 注3)	誤入力データ	(片方或いは両方のDBシステムのデータに)正しくない値が入力されたこと.	II.A.1 Incorrect - entry data
9			陳腐化データ	異なるDBシステムのデータ間において更新の同期がとれていないこと.	II.A.2 Obsolete data
10		値の表現方法の差異	異なる表現	例:会社名での正式名称対略称	II.B.1 Different expressions
11			異なる単位	例:長さでのm対cm	II.B.2 Different units
12			異なる精度	値域(domain)の基数が異なること. 例:重さでの整数表示対{重, 中, 軽}の3段階表示	II.B.3 Different precisions

注1)スキーマ定義文(ANSI SQLのスキーマ定義文)の記述の差異として表現できる異種性.
注2)スキーマ定義文では差異を記述できない異種性.
注3)項番8と項番9はデータ流通システムで解決すべき課題から除外.

2.1.4 流通元/流通先DBシステムへの擾乱解消

(1) 既存センタへの影響軽減

一般にDBシステムは性能的に最適設計されており，DBシステムの搭載センタによっては性能的な余裕が少ない場合がある．この場合，データ流通処理のうち既存DBシステムの搭載センタ上で走行させる処理を最小限にする必要がある．

(2) 高負荷時間帯回避

同様に，DBシステムによっては高負荷時間帯に処理を実行させることが許されない場合がある．この場合，既存DBシステムの高負荷時間帯を回避して流通処理を実行させる機能が必要である．

2.2 データ流通システムの開発に対する要求条件

今後多くの場面でデータ流通システムの構築が必要になると予想されることから短期間で開発できる簡易性が望まれる．そこで，手続き型プログラミング言語を用いて一から構築するよりも簡易にデータ流通システムを構築可能とすることが望まれる．

3. データ流通処理モデル

本章では，2章の要求条件を満たすために，データ流通プラットフォームが具備すべき処理機能を抽出・整理することを狙いとして，データ流通処理モデルを提案する．

3.1 流通データの単位

流通データの集合化を可能とする(2.1.1項)ためには，関連するレコード群を1つのまとまりとして管理できるようにする必要がある．そこで，流通ユニットと呼ぶ流通データの単位を新たに設ける．流通ユニットUの定義を以下に示す．なお，これ以降の形式的な記述方法は，植村[5]，数学辞典[6]に従う．

$$\begin{aligned}\mathrm{U}(R_1,\ldots,R_n,\mathrm{PredU}) \equiv \{(r_1,\ldots,r_n)| \\ r_1 \subseteq D_{11}\times\ldots\times D_{1m_1}\wedge\ldots\wedge r_n \\ \subseteq D_{n1}\times\ldots\times D_{nm_n}\wedge \\ \mathrm{PredU}(r_1,\ldots,r_n)\}\end{aligned} \quad (1)$$

ここで，$R_1,\ldots,R_n$ は関係を表し，$D_{i1},\ldots,D_{im_i}$ は関係 R_i がその直積の部分集合になるような定義域を表し，PredUは $P(D_{11}\times\ldots\times D_{1m_1})\times\ldots\times P(D_{n1}\times\ldots\times D_{nm_n})$ 上の計算可能な述語を表す．なお，$P(S)$ は集合 S のべき集合，$\times$ は集合の直積を表す．

流通ユニットの個々の要素を流通ユニットオカレン

スと呼ぶ．曖昧性がなければ，流通ユニットオカレンスを単に流通ユニットと呼ぶこともある．

流通元 DB で送出できる流通ユニットを流通元ユニットと呼び，流通先 DB で受け取りたい流通ユニットを流通先ユニットと呼ぶ．

流通先ユニットの場合は DB 中に存在しないデータであるため，流通ユニット U の定義には R_i を用いずに $D_{i1},\ldots,D_{im_i}$ を用いている．式 (1) の PredU は，流通ユニット U が満たさなければならない制約を表す述語である．

流通ユニットの定義例を示す．今回のトランザクションで更新された関係 A の 1 レコードについて，そのレコードの更新前イメージ（これ以降，BI と略す）と更新後イメージ（これ以降，AI と略す）と，そのレコードを参照する関係 B （A と B の参照関係の基数を 1 対 $n1$ とする）のレコードとを一まとめにする，流通元ユニット U(A, B, PredU) は以下のように定義できる．

まず，トランザクションによる関係の変化を表現するために以下の概念を導入する．

$R(t) \equiv$ トランザクション t のコミット時点における関係 R

ここで，t はトランザクションの直列順序を表し，t の値は 1 から始まる自然数値とする．トランザクション t のコミット時点で未コミットのトランザクションによる変更は $R(t)$ には含まれないものとし，あるレコードが $R(t)$ に含まれるか否かは，更新ログなどの手段を用いることにより，流通元において判定可能とする． $R(t)$ を用いることにより PredU を以下のように定義できる．

$\mathrm{PredU}(a,b) \equiv$
$\mathrm{Card}(a)=2 \wedge \mathrm{Card}(b)=n1 \wedge$
/* a のオカレンスは 2 つ．
b のオカンレスは $n1$ */
$\exists a_i \in a(a_i \in \mathrm{A(current)} \wedge \neg a_i \in \mathrm{A(current-1)}) \wedge$
$\exists a_i \in a(\neg a_i \in \mathrm{A(current)} \wedge a_i \in \mathrm{A(current-1)}) \wedge$
/* a は BI，AI を 1 つずつ含む */
$\forall a_i \in a \forall a_j \in a(a_i.\text{キー値} = a_j.\text{キー値}) \wedge$
/* BI，AI のキー値は等しい */
$\forall b_j \in b(b_j \in \mathrm{B(current)}) \wedge$
/* b のオカレンスはすべて B(current) の要素 */
$\forall a_i \in a \forall b_j \in b(b_j.\text{参照キー値} = a_i.\text{キー値})$
/* b_j の参照キー値は a_i のキー値と等しい */

ここで，$\mathrm{Card}(S)$ は集合 S の濃度を表し，current は今回のトランザクションを表す．

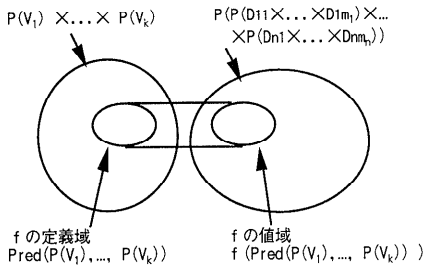


図 3 流通可能性の説明
Fig. 3 Explanation of deliverability.

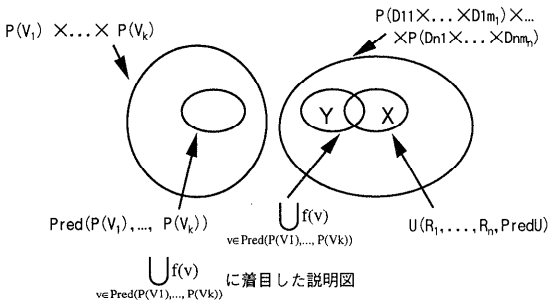


図 4 流通可能性の説明（続き）
Fig. 4 Explanation of deliverability (cont.).

3.2 データ流通可能性

ここでは，データ流通システムで必要となる機能を数学的に整理する．まず，流通可能性の形式的な定義を行う．

ある流通元ユニット $V_1(S_{11},\ldots,S_{1p_1},\mathrm{Pred}V_1),\ldots,V_k(S_{k1},\ldots,S_{kp_k},\mathrm{Pred}V_k)$（ただし k は有限）から流通先ユニット $\mathrm{U}(R_1,\ldots,R_n,\mathrm{PredU})$ へ流通可能とは，以下の計算可能な述語 Pred と，計算可能な関数 f が存在することを言う（**図 3**，**図 4**）．述語 Pred，関数 f の存在はしかるべき期間成り立つものとする．

$$
\begin{aligned}
& f : \mathrm{Pred}(P(V_1),\ldots,P(V_k)) \\
& \quad \to P(P(D_{11}\times\ldots\times D_{1m_1})\times\ldots \\
& \quad \times P(D_{n1}\times\ldots\times D_{nm_n})), \text{かつ} \\
& \quad \bigcup_{v\in\mathrm{Pred}(P(V_1),\ldots,P(V_k))} f(v) \cap U(R_1,\ldots,R_n,\mathrm{PredU}) \\
& \quad \neq \emptyset \qquad (2)
\end{aligned}
$$

ここで，Pred は $P(V_1)\times\ldots\times P(V_k)$ 上の述語を表し，$D_{i1},\ldots,D_{im_i}$ は関係 R_i がその直積の部分集合になるような定義域を表す．なお，$\mathrm{Pred}(P(V_1),\ldots,P(V_k))$ は，述語 Pred を満たす $P(V_1)\times\ldots\times P(V_k)$ の部分集合を表す略記である．定義を以下に示す．

$$
\begin{aligned}
& \mathrm{Pred}(P(V_1),\ldots,P(V_k)) \\
& \quad \equiv \{\langle v_1,\ldots,v_k\rangle | v_1 \in P(V_1) \wedge\ldots\wedge v_k \\
& \qquad \in P(V_k) \wedge \mathrm{Pred}(v_1,\ldots,v_k)\}
\end{aligned}
$$

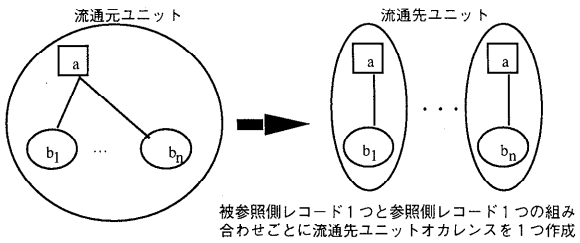


図5 複数の流通先ユニットオカレンスの作成例
Fig. 5 Example of transforming a source delivery unit occurence into target delivery unit occurrences.

関数 f の定義域を，$V_1 \times \ldots \times V_k$ ではなく，各 V_j のべき集合を用いて定義した理由は，同一流通元ユニットの複数オカレンスから，1つの流通先ユニットオカレンスを作る必要がありうるからである．

関数 f の定義域を，述語 Pred を用いて定義した理由は，$P(V_1) \times \ldots \times P(V_k)$ の中で流通して意味のある組合せを選び，そうすることによって計算可能な f が存在できなくなるのを回避するためである．たとえば，実世界における1つの実体の属性群が，流通先では1つの関係で管理されているが，流通元では，異なる流通元の複数の関係で管理されている場合がある．この場合，v_1.キー値 $= \ldots =$ v_k.キー値（ただし，$\{v_i\} \in P(V_i)$）なる述語を満たす $\langle\{v_1\}, \ldots, \{v_k\}\rangle$ の組合せ以外を流通する必要はない．

関数 f の値域を，単なる直積 $P(D_{11} \times \ldots \times D_{1m_1}) \times \ldots \times P(D_{n1} \times \ldots \times D_{nm_n})$ ではなく，そのべき集合に含まれるとした理由は，**図5** の例のように1つの流通元ユニットオカレンスから複数の流通先ユニットオカレンスを作る必要がありうるからである．

式 (2) の「かつ」以降の条件式を，$\cup f(v) = U(R_1, \ldots, R_n, \mathrm{PredU})$ としなかった理由は以下のとおりである．$U - \cup f(v) = \emptyset$ でない理由（図4の領域 X が存在する理由）は，たとえば，流通先 DB が全国 DB で流通元 DB が支店 DB の場合のように，流通先で必要な全オカレンスを流通元から導けるとは限らないためであり，$\cup f(v) - U = \emptyset$ でない理由（図4の領域 Y が存在する理由）は，実世界の意味制約のうち，流通先の方がより多くの制約を具現化している，すなわち，流通先での制約が流通元よりも厳しい場合がありうるからである．

データ異種性の解消（2.1.2 項）の観点からは，ここで提案した流通可能性の定義は十分汎用的である．汎用性を，(1) 関数 f の定義域の記述方法（記述可能な範囲）の汎用性，すなわち流通先ユニットを作成可能な流通元ユニットの組合せが，関数 f の定義域として記述可能なことと，(2) 関数 f の汎用性，すなわちデータ異種性解消機能が関数 f の集合に含まれることとの2つの観点から論じる．

(1) 関数 f の定義域の記述方法の汎用性：

流通ユニットが表す実世界の（1つ以上の）実体に着目する．式 (2) において流通元ユニットの数 k の制約は有限であることだけであり，述語 Pred の制約は計算可能であることだけである．したがって，1つの流通先ユニットが表す実体と流通元ユニットとの対応が最も複雑な場合，すなわち，全流通元の全流通元ユニットの複数オカレンスが対応し，かつそれら複数オカレンスの，特定の述語を満たす組合せのみが対応する場合でも，その流通元ユニットの組合せとその特定の述語とを関数 f の定義域の定義に記述可能である．

上記より，関数 f の定義域の記述方法（記述可能な範囲）は十分汎用であると考える．

(2) 関数 f の汎用性：

関数 f の制約は計算可能であることだけである．したがって，データ異種性解消機能は，計算機上で実現可能な機能である限りは，必ず関数 f の集合に含まれる．関数 f の集合は十分汎用であると考える．

さらに，実世界の同一の実体ではないが流通元と流通先とで計算によって導ける関係（たとえば平均値などの統計的な関係）にある実体間での変換も関数 f の集合に含まれる．したがって，重複格納問題の解決以外でのデータ流通（データウェアハウスへの/からのデータ流通など）にも流通可能性の概念は適用可能である．

本論文の目的は，式 (2) に示す流通可能性が存在することが明らかになった後に，これを実現するデータ流通システムの構築を簡易化するプラットフォームを確立することである．

3.3 操作機能

式 (2) で定義されるデータ流通を計算機上で実現するには，以下の3つの要素機能が必要である．

- 式 (1)，式 (2) の PredU あるいは Pred を評価する述語評価 P
- 関数 f の計算を行う変換 F
- 複数の流通ユニットの部分集合の直積を作成する統合 C（なお，これ以降，統合の説明において，曖昧性がなければ，「流通ユニットの部分集合」を単に「流通ユニット」と略す．）

式 (2) は上記3つの要素機能を CPFP と並べた1つの処理機能で実現できる．しかし，この1つの塊の処理機能だけでは，流通元ユニットと流通先ユニットの組合せごとに流通システムを構築する必要が生じる．以下ではデータ流通の全体的な流通処理量を削減する

(2.1.3項) ために，プラットフォームが基本的に持つべき処理機能について論じる．

(1) 流通ユニットの共用化・流通処理の共通化

異なるデータ流通において設計されたそれぞれの流通元ユニットが同じであり，かつ流通処理の先頭から途中までが同じ処理である場合，流通処理の同一処理部分を共通化し途中からの分岐を可能とすることにより，流通処理量を大幅に削減できる．このためには，3つの要素機能をそれぞれ独立にし，互いに組合せ可能にしておく必要がある．

さらに，異なる変換関数の入力データが同一の場合，あるいは異なる述語の入力データが同一の場合，入力データの読み込み処理を共通化することにより，処理量の削減が図れる場合がある．すなわち，変換Fにおいては複数種類の変換関数を，また述語評価Pにおいては複数種類の論理式の評価を行えるようにしておくことも処理量削減に効果がある．

ここで，流通元ユニット，流通先ユニット以外の中間データを「中間流通ユニット」，各要素機能の入力，出力となる中間流通ユニットをそれぞれ「入力流通ユニット」,「出力流通ユニット」と呼ぶことにする．中間流通ユニットの数学的な定義は，式 (1) におけるPredUを恒真な述語に固定したものである．

(2) 要素機能の実行順序を自由とする

式 (2) の述語Predの積標準形 (conjunctive normal form) が，以下の形になることがある．

$$\mathrm{Pred}(v_1,\ldots,v_k) \equiv \mathrm{Pred}_{n_1}(v_{n_1}) \wedge \ldots \wedge \mathrm{Pred}_{n_m}(v_{n_m}) \wedge \mathrm{Pred}'(v_1,\ldots,v_k)$$

ただし，$v_i \in P(V_i)$，v_{n_i} は $v_1,\ldots,v_k$ のいずれかとする．

この場合，各流通元ユニットの直積をとってからPredを評価することと，各々の流通元ユニットに対して Pred_{n_i} を評価した後に直積をとり Pred' を評価することが等価になる．後者により，直積を作成するCの処理量が軽減される．また，標準形式を介したデータ変換を行うためには，変換Fを2段階で行う必要がある．さらに，データ流通の中には，変換が恒等変換でFが不要な場合など，3つの要素機能をすべて必要としない場合もある．そこで，3つの要素機能は自由な順序で繋げるようにしておく必要がある．

(3) 統合Cの機能について

式 (2) のPredの積標準形が，関係演算の結合演算相当の部分とそれ以外との積の形になることがある．単純化のため，各 v_i ($v_i \in P(V_i)$) が要素が1つの集合である場合を考える．

$$\mathrm{Pred}(v_1,\ldots,v_k) \equiv J_1 \wedge \ldots \wedge J_m \wedge \mathrm{Pred}''(v_1,\ldots,v_k)$$

$$J_i \equiv A_{i_1}(v_{i_1})\theta_i A_{i_2}(v_{i_2})$$

ただし，$\{v_{i_j}\}$ は $v_1,\ldots,v_k$ のいずれか，A_{i_j} は v_{i_j} の定義域（属性）のいずれか，θ_i は比較演算子とする．

この場合，統合Cの中で結合演算相当部分（$J_1 \wedge \ldots \wedge J_m$．これ以降統合条件と呼ぶ）の評価も合わせて行うことにすれば，結合演算の高速化技法（マージ結合法など[7]）と同様な技法を用いることにより，処理量を大幅に削減できる．統合Cでは，述語評価も合わせて行えるように複合機能化しておく必要がある．また，式 (2) のPredUを互いに満たすような複数の流通元ユニットが同時に到着する保証はないので，入力流通ユニットの一時的な蓄積機能が必要である．

以上述べた3つの機能のほかに，流通元DBから流通元ユニットを作成する抽出T，流通先ユニットを流通先DBに反映する反映H，および異なるセンタ間で流通ユニットの送受信を行う集信Rと配信Sの各機能が必要である．合わせて7つの処理機能をデータ流通処理モデルにおける「操作機能」と呼ぶ．操作機能の一覧を**表2**に示す．表2には，データフロー図にならった各操作機能の入出力関係を合わせて示す．

また，7種類の操作機能のつなぎ方の規則（操作機能のトポロジ）は以下を除いて自由とする．

- 先頭は必ず抽出，最後は必ず反映．
- 配信の直後は集信．集信の直前は配信．
- 有向サイクルになるつなぎ方は不可．

3.4 操作機能のセンタ配置と実行契機

3.4.1 操作機能の物理的配置

既存DBシステムの搭載センタへの流通処理の影響を減らす（2.1.4項 (1)）ためには，上記の各操作機能の物理的配置は，以下を除いて自由に選択できるようにしておく必要がある．

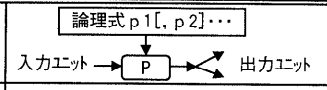
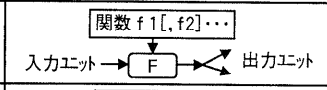
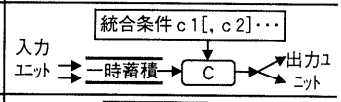
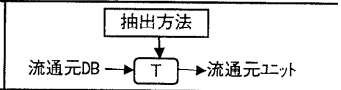
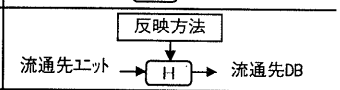
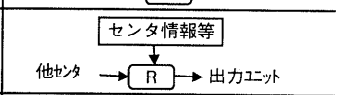
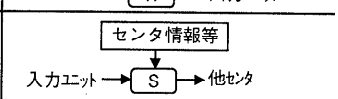
- 抽出は流通元，反映は流通先のセンタに配置．
- 集信と配信は，センタ間での繋がり位置に配置．

特に，流通元，流通先センタとは別の独立したセンタに操作機能を配置すれば，既存のセンタである流通元および流通先センタの負荷を極小化できる．

3.4.2 操作機能の実行契機

高負荷時間帯回避（2.1.4項 (2)）のためには，流通元あるいは流通先センタの閑散時間帯に流通処理を開始/再開始できるようにする機能が必要である．このためには，各操作機能が，その入力データの到着時にただちに処理を開始する内部イベント起動のほかに，あらかじめ定義された契機（定時起動など）や運用者による指示などの外部イベント起動によって，各操作

表2 操作機能一覧
Table 2 Fundamental functions of data delivery.

操作機能名	操作機能の説明	入力流通ユニットの種類数	出力流通ユニットの種類数	操作機能の繋がり規則	データフロー図
述語評価 P	入力流通ユニットから論理式を満たすものを選択する機能	1	複数	制約なし	
変換 F	入力流通ユニットに対して関数fを計算し、出力流通ユニットを作成する機能	1	複数	制約なし	
統合 C	複数の入力流通ユニットの直積から統合条件を満たすものを作成する機能	複数	複数	制約なし	
抽出 T	流通対象レコードを流通元DBから抽出し，それらのレコードをまとめ，流通ユニットを作成する機能	—（流通元DB）	1	繋がりの先頭	
反映 H	流通ユニットを構成レコードに分解し，流通先DBへ反映する機能・	1	—（流通先DB）	繋がりの末尾	
集信 R	流通元と流通先のセンタが異なる場合に，相手センタから流通データを受信する機能	1	1	直前は配信	
配信 S	流通元と流通先のセンタが異なる場合に，相手センタへ流通データを送信する機能	1	1	直後は集信	

機能を開始/再開始する機能などが必要である．なお，統合Cでは，統合処理の開始契機の指定のほかに，一時的に蓄積したデータの削除契機の指定が必要である．

4. データ流通プラットフォームシステム

4.1 データ流通プラットフォームの条件

前章では，様々なデータ流通の形態に対応できるようにするためにデータ流通プラットフォームが持つべき処理機能を，データ流通処理モデルとして示した．本章では，データ流通システムを簡易に構築できるようにするためにプラットフォームが持つべき機能を論じる．

開発の簡易化の要求（2.2節）に対しては，生成方式[8]，すなわち(1)手続き型プログラミング言語よりも高水準な言語（シナリオ言語と呼ぶ）で仕様定義を行い，(2)ソフトウェア部品をその中に取り込んだツールを用いて目的プログラムを自動生成する方式を備えることが望まれる．それぞれについて，以下に述べる．

4.1.1 高水準な仕様定義言語

(1) 流通ユニットの定義

流通ユニットの定義では，3章の式(1)に示した，複数の関係のレコードの集合を表現できること，および流通ユニットが満たすべき制約を表す述語の高水準な定義ができることが必要である．

(2) 操作機能の定義

具体的な流通処理で実際に使用する操作機能と，その繋がり方，および起動契機の指定が定義できること．各操作機能ごとに必要な定義情報を**表3**に示す．特に，統合と述語評価で指定する論理式と，変換で指定する変換方法の定義とは，高水準な言語仕様で記述可能とすることが要求される．

4.1.2 ソフトウェア部品化

3章で述べた各操作機能をそれぞれ，あるいは複合化した形で部品化していること．

またこれらソフトウェア部品を，指定された順序で指定された契機に実行する実行制御機能を備えていることが必要である．なお，統合における開始契機としては，

- 入力流通ユニットがすべてそろった時点などの内部イベント起動
- 定時起動や運用者による起動などの外部イベント起動

などが必要であり，また，一時蓄積削除契機としては，

- 計算可能な指定条件（たとえば統合条件の成立回数など）に基づいて削除する内部イベント契機
- 定時起動や運用者による起動などの外部イベント起動

などが必要である．

表3 定義情報の一覧
Table 3 Information necessary to define behavior of the fundamental functions.

定義情報の分類＼操作機能	変換	統合	述語評価	抽出	反映	集信	配信
操作機能の繋がり方の指定	先行操作機能	先行操作機能	先行操作機能	—	先行操作機能	—	先行操作機能
入出力ユニットの指定	入力ユニット名	入力ユニット名(複数)	入力ユニット名	—	入力ユニット名	—	入力ユニット名
	出力ユニット名(複数)	出力ユニット名(複数)	出力ユニット名(複数)	出力ユニット名	—	出力ユニット名	—
起動契機の指定	開始契機	開始契機 一時蓄積削除契機	開始契機	開始契機	開始契機	開始契機	開始契機
処理の内容	変換方法(出力ユニット対応) 入力項目と変換方法(出力項目対応)	統合条件式(出力ユニット対応)	論理式(出力ユニット対応)	DBからのデータの抽出方法	DBへの反映方法	集信元センタ情報等	配信先センタ情報等

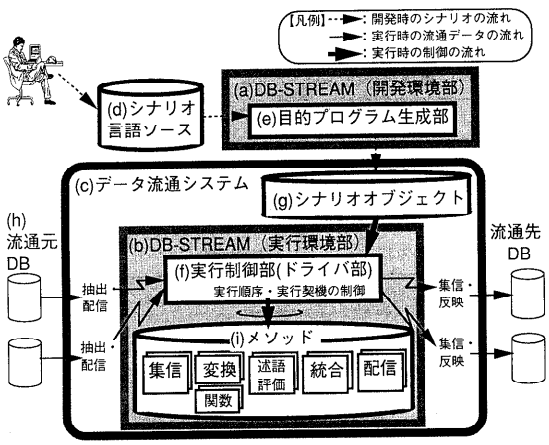


図6 DB-STREAM およびデータ流通システムのアーキテクチャ
Fig. 6 DB-STREAM and data delivery system architecture.

4.2 データ流通プラットフォームシステム DB-STREAM の試作

筆者らは，上記の考えに従ったデータ流通プラットフォームシステム（DB-STREAM と呼ぶ）を試作した．

4.2.1 アーキテクチャ

DB-STREAM および DB-STREAM を利用したデータ流通システムのアーキテクチャを**図6**に示す．

DB-STREAM では，既存 DB システムのセンタへの負荷を極小化するため，抽出・反映（およびそれらに繋がる配信・集信）以外の操作機能をすべて独立センタで実行する物理的配置を採用した．また，実行制御機能は，実行時の独立機能（実行制御部）として実現した．

DB-STREAM は大きく開発環境部（図6(a)）と実行環境部（図6(b)）とからなる．

データ流通システム（図6(c)）の開発時には，開発者が作成したシナリオ言語ソース（図6(d)）を基に，目的プログラム生成部（図6(e)）が，実行制御部（図6(f)）で解釈実行可能な目的プログラム（シナリオオブジェクトと呼ぶ）（図6(g)）を生成する．

データ流通の実行時には，流通元（図6(h)）からのデータの集信を契機に，実行制御部が該当するシナリオオブジェクトを選択し，そのシナリオオブジェクトにしたがって必要なメソッド（図6(i)）を順次コールすることにより，流通処理全体が実行される．

4.2.2 機　　能

DB-STREAM の機能的特徴を示す．

(1) メソッド

DB-STREAM では，部品化したソフトウェア機能をメソッドと呼ぶ．メソッドは，操作機能に対応するメソッドと，変換関数として使われるデータ項目レベルの変換機能に対応するメソッドの2階層のメソッドとして実現した．

(a) 操作機能メソッド

(i) メソッドの種類：データ流通処理モデルの操作機能のうち，抽出と反映は既存 DB システム固有の機能に依存するため部品化の対象外とし，変換，統合，述語評価，集信，配信の5つの操作機能をメソッドとして実現した．なお，変換メソッド内でも述語評価を行えるように複合機能化した．これは，操作機能の繋がりにおいて，変換と述語評価とが隣接する操作機能である場合に，操作機能の定義数，中間流通ユニットの定義数を削減するためである．

(ii) 論理式の記述能力：統合，変換，述語評価で指定する論理式の数学的な記述能力を決定するため，既存の8つの DB システムの意味制約ロジックをサンプル調査した．その結果，各システムに固有の機能に深く結び付いた複雑なロジックが多く，記述能力を限定するのは困難なことが判明した．一方，比較述語を

表4 項目レベル変換機能メソッド（抜粋）注1)
Table 4 Methods corresponding to data item conversion functions (extract).

項番	異種性の分類			項目レベル変換機能メソッドの分類 注2)			関数名
	大分類	中分類	小分類	大分類	中分類	小分類	
1	ｽｷｰﾏ的異種性	1ﾚｺｰﾄﾞ対1ﾚｺｰﾄﾞ	名称の差異	ﾚｺｰﾄﾞ名とそのﾃﾞｰﾀ項目名を，ｼﾅﾘｵの流通ﾕﾆｯﾄ定義構文のﾕﾆｯﾄ名とﾕﾆｯﾄ内のﾃﾞｰﾀ項目名に対応づけることにより解決.			－
2			構成の差異	規定値設定によるﾃﾞｰﾀ項目の作成	単純代入	単純代入	move
3		多ﾚｺｰﾄﾞ対多ﾚｺｰﾄﾞ		同一ﾕﾆｯﾄ内のﾚｺｰﾄﾞの対応の場合は項番1に同じ. 異なるﾕﾆｯﾄの場合は、"統合"ﾒｿｯﾄﾞにより解決.			－
4		1ﾃﾞｰﾀ項目対1ﾃﾞｰﾀ項目	名称の差異	ｼﾅﾘｵの変換定義構文における<変換指定>の<項目変換指定>で、入力ﾃﾞｰﾀ項目と出力ﾃﾞｰﾀ項目を対応づけることにより解決			－
5			ﾃﾞｰﾀ型の差異	ﾃﾞｰﾀ型の変換	ﾃﾞｰﾀ型変換	INT->CHAR	altic
						CHAR->INT	altci
						CHAR->SHORTINT	altcs
						その他30種	省略
6		複数ﾃﾞｰﾀ項目対複数ﾃﾞｰﾀ項目		ﾃﾞｰﾀ項目の統合・分解	文字列操作	文字列結合	couple
						部分文字列	substr
7		ﾚｺｰﾄﾞ対ﾃﾞｰﾀ項目				左ｼﾌﾄ/右ｼﾌﾄ	lshift,rshift
						その他11種（文字探索/置換等）	省略
					算術演算	四則演算	+,-,*,/
						剰余	mod
						最大,最小	max,min
8	ﾃﾞｰﾀ的異種性	値の表現方法の差異	異なる表現	ﾃﾞｰﾀ型の値の体系の変換	対応表変換	対応表変換	altcode
9			異なる単位		算術演算	四則演算	+,-,*,/
10			異なる精度		対応表変換	対応表変換	altcode
11	異種性なし			無変換代入	単純代入	単純代入	move

注1)表1の項番8,9は除いた．また,異種性が存在しない場合を追加した．
注2)ｽｷｰﾏ的異種性については,ｽｷｰﾏ的異種性の解決に伴って必要なﾃﾞｰﾀの値の変換ﾒｿｯﾄﾞを対応させた．

論理演算子で結んだ形の論理式の記述はほぼ共通して必要であることも判明した．そこで，比較述語を論理演算子で結んだ形の論理式を記述する機能（4.2.2項(2)(b)で後述するWHERE句およびON句）を実現し，合わせて不足機能を補うために外部プログラミングインタフェースを設けた．

(iii) メソッドの起動契機：配信と統合のみに外部起動契機を設けた．配信に設けた理由は，独立センタ構成としたので，既存センタの高負荷時間帯の回避のためには配信に外部起動契機を設ければ十分だからである．

統合の開始契機は，入力流通ユニットのすべてがそろった時点と，指定時刻による起動の機能を，また一時蓄積削除契機は，指定された統合条件の成立回数によって削除する機能を設けた．

(b) 項目レベル変換機能メソッド

既存の8つのDBシステムをサンプル調査した結果，58種類のメソッドを単独あるいは組み合わせて使用することによりDBシステム間でのデータ変換機能のほとんどを実現できる見通しを得，それらすべてを作成した．主なメソッドを表1の分類に対応させて**表4**に示す．

(2) シナリオ言語

(a) 流通ユニット定義

以下の構成要素を記述することにより，様々な分野のデータ流通に適用できるようにした（**図7**）．

- データ項目とデータ項目の繰返し（図7(1)）
- 構造体と構造体の繰返し，ネスト（図7(2)）

なお，流通ユニットの制約を表す述語は，統合，変換，述語評価の中で記述できるため，流通ユニット定義の中に述語の定義用の構文は設けなかった．

(b) 操作機能の詳細定義

表3にまとめた定義情報のうち，流通処理の中核である変換，統合，述語評価の定義文を説明する．

変換については，出力ユニット対応に，WHERE句（**図8**(1)）で述語評価用の論理式を指定し，構文要素〈変換指定〉（図8(2)）で変換方法を指定する．また，構文要素〈項目変換指定〉（図8(3)）で項目レベルの変換方法を指定する．〈項目変換指定〉の中で記述する関数は，項目レベル変換機能メソッドに対応する．

統合については，ON句（**図9**(1)）で統合条件を指定し，WAKE句（図9(2)）で起動契機（定時または入力がすべて揃った時点）を指定し，WAIT句（図9(3)）で一時蓄積削除契機（統合条件の成立回数が指定回数に達した時点）を指定する．

```
<流通ユニット定義> ::=
UNIT <識別子> "{" <ユニット要素>... "}"

<ユニット要素> ::= <データ項目> | <名前付き構造体>
<データ項目> ::= <識別子> [<配列指示>] <データ型>                    ・・・(1)
<名前付き構造体> ::= <識別子> [<配列指示>] "{" <ユニット要素>... "}"  ・・・(2)
<配列指示> ::= "[" <繰り返し指定> "]"
```

図7 流通ユニット定義構文（抜粋）
Fig. 7 Syntax of data delivery unit definition (extract).

```
<変換定義> ::=
CONV <処理名>
PREDECESSOR <先行処理名>
FROM <流通元ユニット参照>
  {MAP <流通先ユニット参照>
    WHERE <論理式>                                    ・・・(1)
    <変換指定>  }...                                  ・・・(2)

<変換指定> ::= "{" <項目変換指定>... "}"

<項目変換指定> ::= <データ項目>:=<式>
<式>::=<定数> | <データ項目> | <関数> | (<式>)        ・・・(3)
<関数>::=<関数記号>([<式>[,<式>]...])
```

図8 変換定義構文（抜粋）
Fig. 8 Syntax of data conversion process definition (extract).

```
<統合定義> ::=
INTEGRATE <処理名>
PREDECESSOR <先行処理名>{, <先行処理名>}...
FROM <流通元ユニット参照>{, ITG <流通元ユニット参照>}...
  {MAP <流通先ユニット参照>
  ON <論理式>                                         ・・・(1)
  WAKE {<起動時刻指定>| WHEN_ALL_REACH }              ・・・(2)
  WAIT <流通元ユニット参照> UNTIL <一時蓄積削除条件>  ・・・(3)
    {, <流通元ユニット参照> UNTIL  <一時蓄積削除条件>}...}...
```

図9 統合定義構文（抜粋）
Fig. 9 Syntax of integration process definition (extract).

```
<述語評価定義> ::=
DISTRIBUTE <処理名>
PREDECESSOR <先行処理名>
FROM <流通ユニット参照>
  {MAP <流通ユニット参照>
  WHERE <論理式>}...                                  ・・・(1)
```

図10 述語評価定義構文（抜粋）
Fig. 10 Syntax of predicate evaluation process definition (extract).

述語評価については，変換と同様の WHERE 句（**図10**(1)）で出力流通ユニット対応に論理式を指定する．

各操作機能の繋ぎ方は，先行機能を指定することにより指定する（図8～10の PREDECESSOR 句）．

5. 適用事例と評価

5.1 DB-STREAM の適用事例

DB-STREAM を，通信ネットワークを構成する設備（回線，パス，関連する装置など）の設計業務を行う3つの DB システムから，運用中の設備の監視業務を行う2つの DB システムへ，データを流通させるデータ流通システム（設備データ流通システムと呼ぶ）に適用した（**図11**）[9]．以降，図11の各 DB システムをそれぞれ DB-A～DB-E と呼ぶ．これらの DB システムは，回線，パス，装置などの設備の種別に応じて段階的に個別に構築されてきた既存のシステムである．網型 DB の DB-E を除いて，他はすべて関係型 DB である．

DB-STREAM 適用前の，設計系の DB システム（DB-A，DB-B，DB-C）から監視系の DB システム（DB-D，DB-E）へのデータ流通は，設計系 DB の更新データを参考に，人手で監視系 DB のデータ形式に変換し，入力していた．そのため，無駄な入力稼動とともに，たびたび変換誤りや入力誤りによる DB 品質の劣化を招いていた．

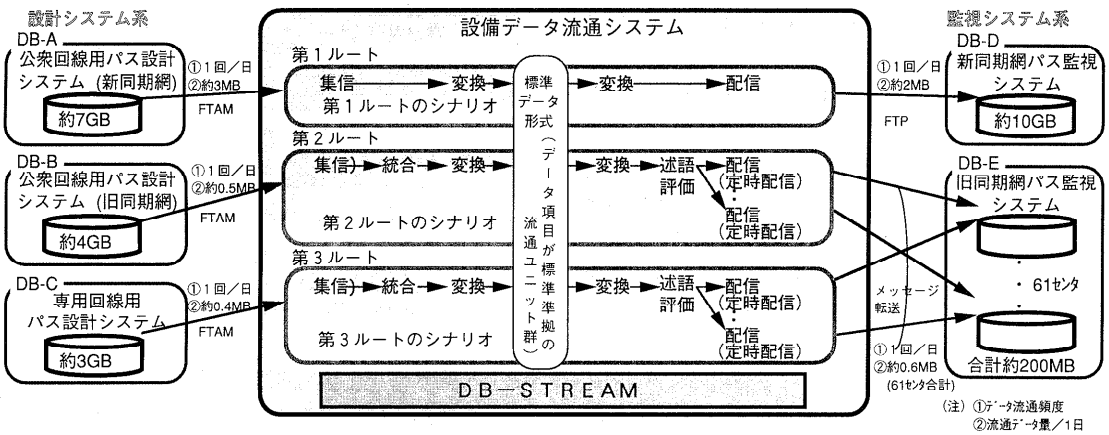


図 11 設備データ流通システム
Fig. 11 Equipment data delivery system.

設備データ流通システムは図 11 に示すように以下の 3 つのルートのデータ流通をサポートしている．

(1) 第 1 ルート：DB-A のデータを DB-D へ流通．

(2) 第 2 ルート：DB-B のデータを DB-E へ流通．第 2 ルートでは，複数種類の流通元ユニット（後述するように，それぞれが DB-B での異なる設計トランザクションの変更レコード群に対応）を基に，1 つの流通先ユニット（DB-E での 1 つの設備情報入力トランザクションの入力レコード群に対応）を作成しなければならないケースがあり，図 11 に示すようにシナリオの中で統合操作機能を使用している．また，流通先の DB-E が 1 システム複数センタ構成をとる分散型システムであるため，シナリオ内で述語評価操作機能を用いて分解を行っている．流通は閑散時間帯である夕刻に定時配信を行う．

(3) 第 3 ルート：DB-C のデータを DB-E へ流通．第 3 ルートも第 2 ルートと同様の理由で，図 11 に示すようにシナリオの中に，統合，述語評価，定時配信を使用している．

各ルートにおける流通ユニットは以下のように設計した．

各流通先 DB システムでは，設備情報の入力トランザクション毎に，そのトランザクションに必要な入力レコード群を流通先ユニットとした．一方，各流通元 DB システムでは，設備の設計トランザクションごとに，そのトランザクションで変更されるレコード群を流通元ユニットとした．

5.2 評　　価

前節で示した DB-STREAM の適用事例を通じて，筆者らが提案する，(1) 様々なデータ流通に適用可能なデータ流通処理モデルの妥当性，および (2) 開発を簡易化するデータ流通プラットフォームの効果，について評価する．さらに，(3) 適用事例におけるデータ流通の効果も評価する．

5.2.1 データ流通処理モデルの評価

適用事例で述べたように，3 つのルートのデータ流通において操作機能を組み合わせることにより，プログラムをまったく開発することなく DB-STREAM のメソッドのみで流通システムを構築でき，データ流通処理モデルの妥当性が検証できた．その詳細を以下に示す．

(1) 流通ユニット

適用事例での各流通先 DB システムでは運用中の設備の監視業務を行っており，その基礎となる設備のデータに矛盾があることは許されず，DB の意味制約の保証が厳しく要求される．流通ユニットという概念によるデータ流通を行うことにより，流通先 DB システムの入力トランザクションに必要なデータ群をひとまとめにして渡すことができ，流通先での意味制約の保証が容易になった．

(2) データ異種性の解消

各 DB システムとも個別に開発されてきた古いシステムであるためデータの異種性は大きかった．3 つのルートにおける，流通対象のデータ項目総数約 2100 のうちの約 2/3 が何らかの変換を必要とした．分析の結果，DB-STREAM の提供する 58 種の変換関数のうち約 70%（40 種）を使用してすべての変換を実現できた．項目レベルの変換メソッドは今後も必要に応じて追加していく予定であるが，表 4 に示したように，今回作成した 58 種の変換関数はデータ異種性の解消機能としてほぼ充足しており，かつ流通分野に依存しない機能であるため，他のデータ流通システムに適用

する際においても大きな追加は不要と想定している．

また，図11に示すように標準データ形式を介する流通シナリオを採用することにより，流通元DBシステム，流通先DBシステムとも，それぞれのシステムが非同期に更改，追加，削除することを可能にしている．この特徴は，大企業などで多くのDBシステムを有し，関連するDBシステムを同期して開発，更改することのできないときに有効であり，現在，この標準データ形式を別の新たに追加した監視系のDBシステムへ流通することも行っている．

(3) 既存センタへの影響

図11における第2ルートおよび第3ルートの流通元/流通先センタはいずれも新たな処理を追加できる性能的な余裕が少なく，流通処理の負荷をできるだけ軽減することが望まれた．DB-STREAMが独立センタ構成であることより，以下のように既存センタへの影響を最小限にすることができた．第2ルートでの処理時間は，流通ユニットオカレンス数が約40のときのサンプルで，流通元センタでの抽出処理時間を1とすると，流通元センタでの配信処理時間が約0.05，データ流通システムセンタでの統合，変換，述語評価，複数センタへの配信の処理時間が約4.5であり，独立センタ化により，反映以外の処理をすべて流通元センタで行う場合に比べ，流通元センタの負荷を約19%（$\cong (1+0.05)/(1+4.5)$）に軽減できた．データ流通システムの独立センタ構成を可能とし，操作機能の自由な配置を可能とする処理モデルの有効性が確認できた．

(4) データ流通処理モデルのデータモデルの適用範囲

3章のデータ流通処理モデルは関係型モデルに限定して議論を進めたが，関係型と網型，階層型との間の変換は知られており[10)~12)]，データ流通処理モデルおよびデータ流通プラットフォームは網型，階層型の場合でも適用可能である．

5.2.2 データ流通システム開発簡易化

試験稼働も含めたデータ流通システム構築稼働は，**図12**に示すように全体で約60%を削減できた．また，データ流通システム構築期間は約1/3に短縮できた．このうちシナリオ開発稼働については約10倍の生産性向上を達成し，大きな効果があった．

分計はしていないが，データ流通システム構築稼働の中には，流通ユニットの構成データ項目の追加/変更や，変換方法の修正等の稼働も含まれており，これらの流通定義仕様の修正が，シナリオの部分修正のみで可能だったことの効果が大きい．プラットフォームとして高水準な仕様記述言語を提供し，シナリオ定義のみで構築可能とすることの重要性が確認できた．

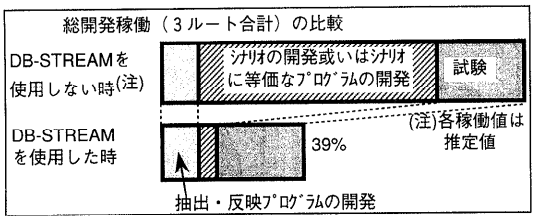


図12 データ流通システム開発稼働の比較
Fig. 12 Comparison of man-months required to develop a data delivery system.

5.2.3 データ流通システムの適用効果

流通先DBシステムに流通されるデータ量は図11に示すように1日あたり約2.6 MBである．自動データ流通の実現により，多重投入稼動を0（従来は平均250人時/日）にでき，流通先DBでの投入誤り/投入漏れを0（従来は全流通データの5~10%）にでき，流通先DBでの投入遅延をたかだか1日（従来は1日から1月）にすることができた．

6. まとめ

大企業で顕在化しているデータの重複格納問題に対して，1つのデータの更新に基づき他の重複データを自動更新することによる解法（データ流通システム）がある．

本論文では，様々な形態のデータ流通システムを簡易に構築できるようにする汎用的なデータ流通プラットフォームを確立することを目的に，(1) データ流通処理モデル、ならびに (2) これに沿ったデータ流通プラットフォームを提案した。

データ流通処理モデルでは、データベースのデータの整合性を保証するために流通ユニットと呼ぶ流通データの単位を導入し，様々なデータ異種性を解決するデータ変換機能を明確化した。

さらに、流通に必要な機能をソフトウェア部品として備え，流通の動作記述を行う高水準な仕様記述言語を備えるデータ流通プラットフォームを提案した。試作したデータ流通プラットフォームDB-STREAMを実際のデータ流通システムに適用し，その効果を示した．本論文では重複格納問題に絞って議論をしたが，ここで提案したデータ流通処理モデル，データ流通プラットフォームは戦略的にデータを重複させた場合のデータ流通，すなわちデータ移行ツールなどにも適用可能である．

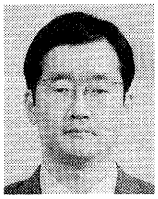
本研究では，複数DB中の流通すべきデータの特定

や，DBシステム間でのデータの変換方法の設計などは事前に人手でなされていることを暗黙に仮定している．今後さらにデータ流通の適用性を高めるには，複数DB中の流通すべきデータを自動的に特定する研究，データの変換シナリオの自動生成に関する研究などが有効であり，この分野の成果が期待される．

参考文献

1) 堀内 一：データ中心システム設計，オーム社，東京 (1988).
2) Brodie, M.L.: The Promise of distributed computing and the challenges of legacy information systems, *Proc. IFIP TC2/WG2.6 Conference on Semantics of Interoperable Database Systems*, pp.1–31 (1993).
3) Sheth, A.P., et al.: Federated Databases for Managing Distributed, Heterogeneous, and Autonomous Databases, *Comput. Surv.*, Vol.22, No.3, pp.183–236 (1990).
4) Kim, W. and Seo, J.: Classifying Schematic and Data Heterogeneity in Multidatabase Systems, *Computer*, Vol.24, No.12, pp.12–18 (1993).
5) 植村俊亮：データベースシステムの基礎，オーム社，東京 (1979).
6) 日本数学会（編）：数学辞典（第3版），岩波書店，東京 (1985).
7) 増永良文：リレーショナルデータベース入門，サイエンス社，東京 (1991).
8) 磯田定宏：ソフトウェア再利用，情報処理ハンドブック，6編8章，pp.784–793，オーム社，東京 (1995).
9) 横山雅明，合田信久，堀田英一：ネットワークデータベースの基盤整備，NTT技術ジャーナル，Vol.6, No.9, pp.49–53 (1994).
10) Lien, Y.: Hierarchical Schema for relational Databases, *ACM Trans. Database Syst.*, Vol.6, No.1, pp.48–69 (1981).
11) Tsichritzis, J. and Lochovsky, F.: Data Models, Prentice-Hall, Englewood Cliffs, NJ (1982).
12) Zaniolo, C.: Design of relational views over network schemas, *Proc. ACM SIGMOD Conference*, pp.179–190 (1979).

(平成8年10月11日受付)
(平成9年 9月10日採録)



池田 哲夫（正会員）

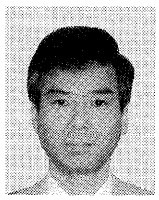
1956年生．1979年東京大学理学部情報科学科卒業．1981年同大学大学院理学系研究科情報科学専攻修士課程修了．同年，日本電信電話公社（現NTT）電気通信研究所入所．現在，NTT情報通信研究所データベース研究部主任研究員．この間，プログラム言語の意味論の研究，データベース管理システムの研究開発などに従事．ACM，IEEE CS各会員．

伊土 誠一（正会員）

1947年生．1969年北海道大学工学部電子工学科卒業．1971年同大学大学院工学研究科電子工学専攻修士課程修了．同年，日本電信電話公社（現NTT）電気通信研究所入所．現在，NTTソフトウェア研究所所長．この間，DIPS用OSの研究開発，大規模情報処理システムの開発，ソフトウェア工学，およびデータベースの研究開発に従事．現在，エレクトロニックコマース（EC），情報流通，インターネットの研究開発に興味を持つ．

石垣昭一郎（正会員）

1948年生．1971年京都大学工学部数理工学科卒業．1973年同大学大学院工学研究科数理工学専攻修士課程修了．同年，日本電信電話公社（現NTT）電気通信研究所入所．現在，NTT OCN事業部販売推進担当部門長．研究所在籍中，データベース管理システムの研究開発などに従事．経営情報学会会員．



村田 達彦（正会員）

1952年生．1965年早稲田大学理工学部電子通信学科卒業．1967年同大学大学院理工学研究科情報工学専攻修士課程修了．同年，日本電信電話公社（現NTT）電気通信研究所入所．現在，NTT法人営業本部第二営業部（都市開発部）担当部長．研究所在籍中，データベース管理システム，データベース利用技術の研究開発などに従事．