

リアルタイム用リレーショナルデータベース
管理システム ADF/RS (2)

2J-2

—データ・モデルと操作—

広田 敦彦* 大脇 隆志* 山口 康隆** 林 利弘*

* (株)日立製作所 大みか工場

** 同 システム開発研究所

1. はじめに

リアルタイム用リレーショナルデータベース管理システム ADF/RS を開発した。

本システムは、リレーショナルデータベースの使い勝手に着目し、これを、定型業務を主体とする実時間システム用に、データ・モデル、操作機能、性能の面で拡張、改良を加えている。

ここでは、ADFで採用したデータ・モデルとその操作の特徴について報告する。

* ADF/RS: Advanced Data management Facilities for Real-time application System

2. 要求仕様と実現目標

リレーショナルDBは、従来の構造型DBと比べ、操作の容易さといった観点からは、非常に優れたものである。特に、

- (1) 操作の容易さ、柔軟性が要求される
- (2) 特に厳しい応答性は要求されない
- (3) データ構造が動的に変化する

といった特質を持つ応用分野においては、その威力を十分に発揮する。

一方、リアルタイム制御分野においては、

- (1) 操作の柔軟性よりも、操作結果の整合性を重視する
- (2) 極めて高い応答性が要求される
- (3) データ構造がある程度、固定的である

といった異なった特質を持っている。

このため、リレーショナルDBをそのまま適用すると、以下の問題が生じる。

- (1) 高い柔軟性を持った操作と、データ構造を実行時に確定するメカニズムとにより、逆にデータ操作が複雑となり、また処理の応答性も低下する。
- (2) データ間の関連をDBMSが認識していないため、データの整合性維持をユーザ責任で行わねばならず、ユーザの負担が増加し、また、システムとしての信頼性も低下する。

(3) データ整合性維持のためには、複数のリレーションを用いて、同時に操作を行わなくてはならない。このため排他制御をユーザが意識して行う必要があり、ユーザの負担が増加し、またシステムとしての信頼性・応答性劣化の要因となりやすい。

このため、ADFでは以下を開発目標とした。

- (1) 内部スキーマレベルでは、データ間の関連を認識し、効率良いデータ操作を行うが、ユーザ・インタフェースは、リレーショナルDBと同様、簡単な形式のものとする。
- (2) 排他制御を含めたデータベース整合性維持をユーザに一切意識させないようにする。

3. 具体的なアプローチ

3.1 データ・モデル

(1) 論理ファイル

定型業務を主体とする処理分野においては、一般に、前もってデータ構造やアクセス仕様を決められる場合が多い。このためADFでは、処理に必要なデータ項目を、処理で扱い易い形で抽出した表(論理ファイル)を予め規定できるようにし、リレーショナルDBにおける「射影(PROJECTION)」や「結合(JOIN)」のような、実行時にデータ構造を規定する操作を行わずに良いようにした。(図1)

このため、ユーザは、処理に必要なデータ項目を意識し、データ選択操作だけを行えば良いため、プログラムが簡単になる。また、事前処理が可能であり、操作の応答性も向上する。

(2) 階層表

1:N対応のデータ構造を表現できる「階層表」を実現した。(図2)

このため、データをより自然な形で表現できるようになる。また、リレーショナルDBのような不要なデータの重複が生じないため、データ操作時の応答性も向上する。

ADF/RS (2) Data Model and Data Manipulation

Atsuhiko HIROTA¹, Takashi OWAKI¹, Yasutaka YAMAGUCHI², Toshihiro HAYASHI¹

¹ HITACHI, LTD. ONIKA WORKS.; ² HITACHI, LTD. SYSTEM DEVELOPMENT LAB.

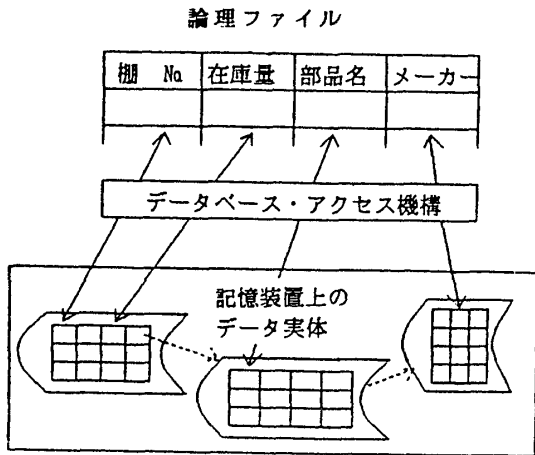


図1 論理ファイル

棚No	格納部品		
	部品No	製造元	
		メーカー名	所在地
1	P001	XYZ工業	東京
		ABC製作所	大阪
		IJK部品	福岡
1	P002	LMNプレス	横浜
		ABC製作所	大阪
2	P006	STU電機	札幌
		BCDエンジニアリング	大阪
		DEF発条	名古屋

図2 階層表

3.2 データ操作

(1) 構造化データベース言語

データ操作言語は、データに対する選択条件を指定するだけでデータの検索が行えるようにした。また、論理ファイルに対する一連の処理を処理のブロック構造で記述できるような形式とした。(図3)

このことにより、データ操作をHow(手段)レベルではなく、What(目的)レベルで行えるようになる。また、構造化プログラミング言語との親和性が高まるとともに、データ構造とプログラムの処理構造との間の親和性も高まるため、プログラムの記述容易性、可視性が大幅に向上する。

(2) システム制御

データ操作言語をブロック構造化し、ユーザ・トランザクションの開始・終了時点を認識できるようにすることにより、排他制御や異常時の

バックアウト処理を、ADFが自動的に実施するようにした。

このことにより、ユーザは、複雑なシステム制御に拘わる事項を一切意識することなくデータ操作が行えるようになり、プログラム作成が簡単化するとともに、システムの信頼性も向上する。

4. まとめ

以上説明したように、ADF/RSのデータ・モデルとデータ操作機能により、定型業務を主体とするリアルタイム処理システムのソフトウェア開発における生産性・信頼性・保守性の向上を図ることができた。

また、リレーショナルDB、構造型DBそれぞれの特長を活かした形での統合インターフェースの実現に対して寄与できたものと考えている。

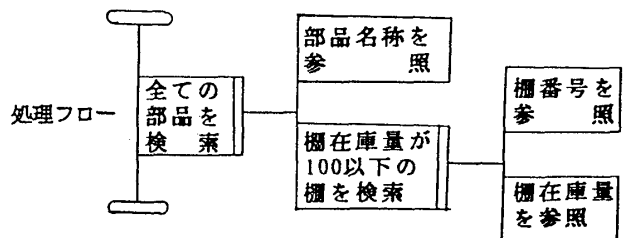
参考文献

- 1) C.J.Date: An Introduction to Database Systems, Addison Wesley, 1981.
- 2) Y.Kanbayashi: Semantics of Data Structures, 情報処理 VOL.27 No.2 1986.

データ構造

部品	格納棚
----	-----

部品名称	棚番号	棚在庫量
ボルト	3	150
	7	80
	10	40
	40	100
ナット	4	105
	6	40
	15	50
	20	150



プログラム例

```

while lcats (ALL) of (部品)
  repeat
    eobtan (部品名称) to (wk.name);
    while lcatss (棚在庫量(=100) of (在庫状況)
      repeat
        eobtan (棚番号) to (wk.tanano);
        eobtan (棚在庫量) to (wk.zaiko);
        CRT^ (wk) ヲ ショウヨクスル;
      end;
    ends of (在庫状況);
  end;
ends of (部品);
    
```

図3 データ構造とデータ操作プログラム