2R-1

データ構造から見た、データの高水準な意味付け

横井　伸司
日本アイ・ビー・エム(株)　東京基礎研究所

1．はじめに

近年提案されている、ユーザ・インターフェイス・マネジメント・システム（ＵＩＭＳ）を利用してアプリケーションを開発するという方法は、アプリケーションは、その機能を実現する部分とユーザとのインターフェイスを実現する部分（それぞれ、機能実現部分及びインターフェイス実現部分と呼ぶ）とに分離でき、それぞれを独立して開発できるという考えに基づいたものである。

我々は、この考え方に基づいて、幾つかのアプリケーション開発を行ってきた。そして、その経験を通じて以下ような現象とその問題点を発見した。

（１）機能実現部分の内部で使用されるデータ構造は、特に、知識ベース・システムの構築においては、アプリケーションの開発の過程で変更されることが多い。しかし、この変更に応じてインターフェイス実現部分の対応するコードを変更していく作業は時間がかかり、心理的にも単調な作業で負担になる。

（２）ユーザ・インターフェイスの仕様自体も、アプリケーション作成の過程や、完成後においても変更されることがありうる。このときに、機能実行部分にまで変更を加えることは先程と同様負担であり、バグを誘発する要因でもある。

我々の要求は、(A)機能実現部分でのデータ構造の変更を吸収し、かつ(B)ユーザ・インターフェイスの仕様変更に容易に対応できるようなインターフェイス実現部分の機構を開発することである。これらの要求を満たすために、我々は図１のような構成を考えた。画面上への表示は、アプリケーションからの表示すべきデータの収集、収集したデータの加工、そして最終的な表示の３つのルーチンで行われる。今回はこのうち前者について述べる。

要求Ａを満たすためには、アプリケーションと収集ルーチンとの両者の間で、データ構造の変更を吸収する汎用な情報交換手段を定義しておく必要がある。要求Ｂを満たすためには、収集するデータを指定するのにプログラムを書くのではなく、収集したい情報の仕様を与えれば、自動的にデータを収集するような汎用の機構、およびそのための記述言語を定義しておく必要がある。

我々は、汎用なデータ構造として仮想的なテーブルを想定し、射影関数によって機能実現部のデータ構造との対応を採ることにした。そしてテーブル上の収集データ記述言語として関係データ・ベースの問合せ言語を再帰的合併を許すように拡張した言語を用意した。

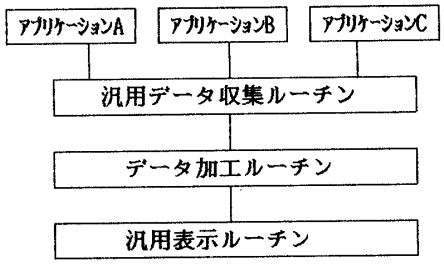

図１　インターフェイス実現部分の構成

2．問合せ言語の拡張

本節では、関係データ・ベースへの再帰的問合せ[1, 2]について述べる。この問合せは、合併(union)に対して、いわば再帰的なself_joinを許すように拡張したものであり、再帰的合併(recursive union)[2]と呼ばれる。シンタックスは次の通りである。ここで、(2)の<query expression>では、<correlation>を参照してjoin操作を指定することができる。

```
<recursive_union expression> ::=          ...(1)
    <query expression>                    ...(2)
  recursive_union                         ...(3)
    <query expression>                    ...(4)
  as <correlation>                        ...(5)
  [limit <limit expression>]              ...(6)
<correlation> ::=                         ...(7)
  <correlation name>                      ...(8)
    <column name>{,<column name>}...      ...(9)
```

セマンティクスは次の通りである。

（１）１番目の<query expression>を実行する。そして、その結果できた表をＴとする。
（２）表Ｔから一行抜き出し、表Ｒに追加する。
（３）いま抜き出した行を参照して、２番目の<query expression>を実行する。その結果できた表をＴに追加する。
（４）２、３の操作をＴが空になるまで行う。
（５）結果として、表Ｒを返す。

<limit expression>は、２、３の操作を行う回数を制限したり、サイクルの検出を指定する。

High level categorization of data structures for general purpose queries
Shinji YOKOI
IBM Research, Tokyo Research Laboratory

uid	color	wid	p_uid	n_uid
20	red	11	0	87
16	blu	35	87	24
24	yel	57	16	0
87	tur	24	20	16

図2 unitテーブル

uid	color	n_uid
20	red	87
87	tur	16
16	blu	24
24	yel	0

図3 問合せ結果

いま、図2でuidが20のユニットから、リンクで辿れる全てのユニットを取り出したいとする。この検索は以下の問合せ文で行うことができる。この問合せを実行した結果は図3の通りである。

```
select * from (
    select uid,color,n_uid
       from unit
       where unit.uid=20
  recursive_union
    select uid,color,n_uid
       from unit
       where unit.uid=px.n_uid)
 px(uid,color,n_uid)
```

3．考察：アプリケーションでの内部データ構造

各コンポーネントで汎用データ収集ルーチンに対して、収集すべきデータの指定方法について考えてみたい。そのために、アプリケーション内で表現されるデータについて考えてみる。

アプリケーション内のデータは、指標型、構造型の2種類に分類できる。指標型とは、各データがそれぞれ独立して指し示すものをもっている場合である。例えば、ユーザＩＤ、その日の日付、内部で使うモード等はこれに含まれる。このデータの特徴は、同じ意味タイプをもつ複数のデータが意味を持つのではなく、単独で意味をもつと言うことである。これに対して、構造型とは、同じ意味タイプのデータが複数個集まって意味をもつ場合である。これには、例えば、スケジューリング・システムに於けるスケジュール単位などが挙げられる。

指標型データについては、収集したいデータの個数が固定されており、かつそれが実行時に不変であれば、仕様として、あらかじめそれらを列挙しておけば、特に問題は生じない。また、変更の場合にも列挙されている名前を変更すれば済む。

問題なのは、収集したいデータが構造型の場合である。つまり、この場合はデータの個数が複数で、通常可変であるからそれらを列挙することは不可能である。従って、データの個数が変化してもそれに対応できるような仕様を与えておかなければならない。このような場合に再帰的合併が有効である。以下、その有効性について考察するために、図4に和田[3]によるデータ構造の分類を引用する。

配列については、射影関数によってこれを素直にテーブルと同一視できるので、一般の関係データベースの問合せ言語を用いて必要なデータを収集するための指定を行うことができる。プロローグのユニット・クローズやプロダクション・システムのファクトも配列と同様に考えられる。レコード型は、それ自身では指標型データに過ぎない。また要素がレコード型である配列もレコードを一つの行とみれば、テーブルであると見做せる。

レコード型が重要であるのは、これとポインタ型を使ってリンクによる構造を実現するときである。例えば、2節のユニット・テーブルは、このリンクによる構造とみることができる。この構造をとるデータの収集方法を指定するには、(ア)先頭のレコードと(イ)“隣”の関係を示すレコード内のフィールド名を指定しなければならない。2節で述べた再帰的合併による指定は、シンタックスの(2)、(4)の<query expression>がそれぞれ(ア)、(イ)に対応しており、容易に仕様を与えることができる。

以上の考察を纏めると、(少し乱暴であるが)次のように言える。即ち、内部表現が、リニアなものは、そのままの形でテーブル表現と同一視できるので、関係データ・ベースの問合せ言語で検索できる。内部表現がリンクのものに関しては、再帰的合併を許すような拡張によって扱える。

アクセスの仕方：
- ランダムアクセス
  - どの要素にも同一時間でアクセスできる
- シーケンシャルアクセス
  - “隣”の関係をたどって次々とアクセス

要素の総数：
- 固定　配列、レコード
- 可変　スタック、キュー、リスト、木構造

内部表現：
- リニア　“隣”の関係は記憶場所の関係も確定
- リンク　“隣”の関係はポインタにより可変

図4　データ構造の分類［文献3より引用］

4．まとめ

関係データ・ベースの問合せ言語の考え方を拡張して、ＵＩＭＳの設計に応用した。

参考文献

[1] Phil Shaw,CLOSURE expressions,
ANSI X3H2-87-330, 1987

[2] 芝野耕司、ＳＱＬ３における再帰的問合せ、
情報処理学会第３７回全国大会、1988

[3] 和田英一、手続き型言語におけるデータ構造、
情報処理、Vol.27 No.2 1986