

2L-7

複合オブジェクトに関する一検討
——リレーショナル・モデルとの比較

宇田川 佳久

(三菱電機㈱・情報電子研究所)

1. はじめに

オブジェクト指向データベース(OODB)は、複雑化するデータベース応用に対処するための技術と位置付けることができるものである。例えば、エンジニアリングやOAといった応用では、オブジェクト(処理の対象)がより単純なオブジェクトの組み合わせによって表現されることが多く、これまで、OODBの研究対象とされてきた。“より単純なオブジェクトの組み合わせによって表現される”オブジェクトは、複合オブジェクトと呼ばれており、OODB宣言でも最初に提示されている重要な概念である⁽⁴⁾。それにもかかわらず、体系的な定義あるいは他の手法との比較が十分に行われていないのが現状である。

本文では、リレーショナル・モデルによって回路を記述する方法を示し、オブジェクト指向の観点から考察を加えている。その結果、回路を記述する場合でも、複合オブジェクトや汎化を扱う枠組みが重要であることが判明した。

2. リレーショナル・モデルによる回路の記述

これまで、リレーショナル・モデルによって回路を記述する方法が提案されてきた^(2,3)。これらの提案では次のような共通点を見出すことができる。

- (1) 回路は上位のモジュールの構成要素になり得る。
- (2) 回路はより単純な回路の組み合わせである。
- (3) 回路には端点がある。
- (4) 端点は結線によって結ばれる。

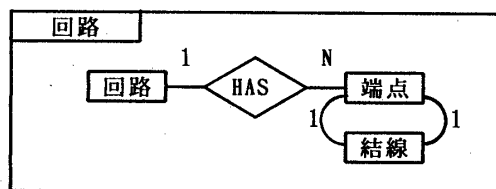


図1. E-R図による回路の記述

(1)と(2)は回路が階層的に定義されていることを言っており、本質的な差異はないと考えられる。一般に、一つの回路には複数の端点に対応している(1対N対応がある)。また、結線は端点どうしを一つ一つ結ぶものと見なすことができる。図1は(1)から(4)の条件をE-R図を用いて表現したものである。属性に関しては、次のものが考えられる。

回路: 機能名、表示座標、遅延時間、消費電力

端点: 入出力区分、ファンアウト数、表示座標

結線: 結線長

回路、端点、結線は、直観的に独立したものとして認識できるものであるから、回路、端点、結線にリレーションを割り当てることが自然である。また、回路ID、端点ID、結線IDが決まれば、上記の属性値が決まるので、次に示すスキーマを得ることができる。

回路(回路ID、機能名、表示座標、遅延時間、消費電力)

端点(端点ID、回路ID、入出力区分、ファンアウト数、表示座標)

結線(結線ID、入力端点ID、出力端点ID、結線長)

回路、端点、結線を表すリレーションには、以下のような参照制約があり、全体として複合オブジェクトを形成している。

 $\{\text{回路の回路ID}\} \supseteq \{\text{端点の回路ID}\}$ $\{\text{端点の端点ID}\} \supseteq \{\text{結線の入力端点ID}\} \cup \{\text{結線の出力端点ID}\}$

このスキーマは、次の観点からも正当付けられるものである。すなわち、回路と端点には1:Nの関係が

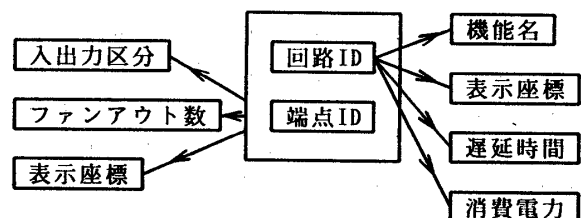


図2. 回路と端点に関する関数従属

A case study on a composite object — a comparison with the relational approach

Yoshihisa UDAGAWA

MITSUBISHI Electric Corp.

あるため、回路と端点を一つのリレーションで表現しようとする、図2に示した関数従属を持つリレーションとなる。このリレーションは、回路に関する属性値が繰り返し現われ、記憶容量、一貫性管理の観点からも問題を生じる⁽¹⁾。

3. オブジェクト指向の観点からの記述

以上、リレーショナル・データベースによって表現することを前提にした議論を展開してきたが、回路と端点は本当に別々に扱うべきものであるのか、疑問が残る。なぜなら、エディタで操作するシンボルにしても、チップ上に焼き付けられたパターンにしても、回路と端点は一体として扱われているからである。理論的には別々のものかも知れないが、少なくとも操作の対象としては、回路と端点を一体として（複合オブジェクトとして）扱う方が自然であるように思える。回路と端点を別々のリレーションで扱わなければならなかったのは、むしろリレーショナル・モデルの“お家事情”によるものであることが、図2に示した関数従属性からも推察することができる。

回路と端点を合わせて一つのオブジェクトとして扱うためには、一つの回路と対応する端点の集合を一つの記憶単位とするメカニズムが必要になることが、リレーショナル・モデルによる回路の記述例から理解することができる。図3は、回路と端点を合わせて1つの複合オブジェクトとしたときの属性間の構造を示したものである。2章で述べたスキーマと図3のスキーマとの違いは、図3では入力端点と出力端点を独立した属性として扱われていることである。また、端点は入力端点と出力端点を汎化したものであることを露に表現していることである。この汎化をリレーショナル・データモデルで表現するには、次の2つの方法が考えられる。第1の方法は、汎化を直接表現するものである。この場合、入力端点と出力端点を別々のリレーションで表現する。この方法は一般的なもので、仮に入力端点と出力端点とに異なる属性があったとしても、属性を追加するだけで対応することができる。第2の方法は2章で述べたように、端点を一つのリレーションで表現し、入／出力端点を区別するための属性（例えば“入／出力”）を追加するものである。いずれの方法でも、回路と端点を一体として扱うためには、結合(join)演算が必要となることになる。

これに対し、図3に示したスキーマでは、属性として（すなわち独立した概念として）入力端点と出力端点を区別しており、かつ、それらは端点の一種（サブクラス）であることを陽に表現している。図4は、図3をレコードを作る構成子(record)と集合を作る構成

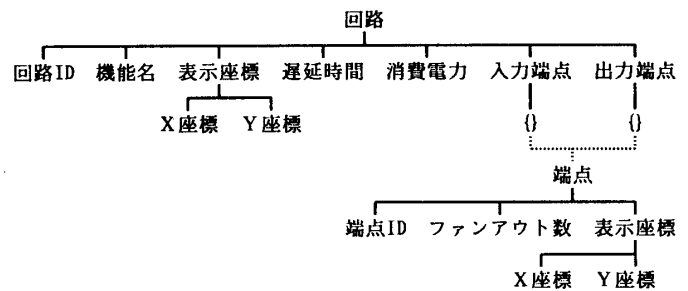


図3. 回路と端点を一体とした複合オブジェクト

```
回路 = record 回路ID;
           機能名;
           表示座標 = record X座標;
                           Y座標;
                       end
           遅延時間;
           消費電力;
           入力端点 = set 端点;
           出力端点 = set 端点;
       end

端点 = record 端点ID;
           ファンアウト数;
           表示座標 = record X座標;
                           Y座標;
                       end
       end
```

図4. 図3のデータ構造の宣言例

子(set)を用いて記述したものである。

4. おわりに

本文では、回路を対象としてリレーショナルとオブジェクト指向のアプローチの違いを比較した。その結果、リレーションというデータ構造の制約のために、利用する側から見て1つのオブジェクトであるものが、複数のリレーションによって表現されることがあることが分かった。本文では、これまでに多くの研究がなされてきた回路を対象としたが、今後、建築物のレイアウト図やドキュメント・データなどを検討し、複合オブジェクトに対する理解を深めたい。

参考文献

- (1) Date, C. J.: An introduction to DB system, Vol .1, 4th ed. Addison-Wesley, 1986.
- (2) Haskin, K. et al : On extending the function ..., Proc. ACM SIGMOD 1982, pp. 207-212.
- (3) Ketabchi, M. A. et al : Mathematical model of ..., IEEE Trans. of Soft. Eng. 1988, pp. 71-84.
- (4) Atkinson, K. et al : The object-oriented DB system manifesto, Proc. DOOD, Kyoto, 1989.