ソフトウェア工学 40－3
（1985. 2. 7）

E・Rモデル仕様データベースエディタ

車谷 博之、 大槻 繁、 野木 兼六
（日立製作所 システム開発研究所）

1．はじめに

ソフトウェアの開発及び保守過程の生産性・品質向上を目的として、要求定義や設計を対象とする支援システムが提案され、ソフトウェア仕様情報の計算機管理が為される。ソフトウェア開発保守過程では頻繁に仕様の変更が生じるため、仕様エディタが必要となる。[1)2)]

従来の手書きドキュメントによる記述方式では各ドキュメント間で冗長な情報があるため、ある仕様書の変更が他の仕様書の変更を引き起こし、しばしばこの変更を誤るため、仕様書間で矛盾が生じる。例えば手続き仕様書における手続き名の変更は、手続き一覧表・制御関連図等の変更を引き起こす。これは適切なデータモデルを用いてソフトウェア仕様情報を一元管理し、共通情報を一意に変更することにより解決できよう。

また従来、分かり易さの面から表形式や図形式の仕様書が多く用いられており、これらとの親和性が望まれる。そして、仕様書の形式は利用部門・仕様情報等によって多様であり、仕様エディタのユーザインタフェースの汎用化が望まれる。

そこで、以下のように、仕様エディタのユーザインタフェースと編集方式のモデルを考案した。

（1）E・Rモデルを仕様データベースのデータモデルとして採用[3)]

ソフトウェア仕様情報を比較的自然に表現できるようにE・Rモデルを拡張し、このモデルによる完全性を保証することによって矛盾を防止する。

（2）表形式画面インタフェースのモデル化

従来の日本語表形式仕様書を一般化した表形式モデルを導入する。

（3）表形式画面から仕様データベースを直接編集する方式のモデル化

表形式モデルのCRT画面インタフェースによる編集を可能とし、個々の表形式仕様書を仕様データベースのビューとして提供する。このため、表形式モデルとE・Rモデルとの対応方式をモデル化する。また、このような形式の使用は仕様情報の標準化という観点から好ましい。

（4）編集作業シーケンスをモデル化

編集作業では単一の仕様書の編集だけではなく、一連の関連した仕様書群を編集することが多い。このため、ユーザインタフェースとしては関連する仕様書間の遷移を容易に行える機構を導入し、各ビューを状態、ビュー更新を遷移とする状態遷移モデルを導入する。

本稿では、2章で本エディタが対象としているデータモデル E・Rモデルの概要を述べ、3章で従来仕様書で用いられてきた表の一般化モデルを述べ、4章でユーザインタフェースモデルを述べる。

2．E・Rモデル

本エディタが対象としているデータモデルは、P．ChenのE・Rモデルを以下のように拡張したものである。[4)]

（1）実体集合（Entity Set） 実体は存在するものであり、識別することが要求される。しかし、仕様情報において例えばif文やwhile文などを実体としても検索の必要がなく、得策といえない。ここでは、「名称を持つものであり、検索の要があるものとする。実体は実体集合としてグループ化される。例 設計仕様書における、「変数」、「型」、「手続き」

（2）関係集合（Relationship Set） 関係Rは、

$R = \{ [e_1, \ldots e_n] \mid e_j \in E_i \cup \cdots \cup E_m \}$ E_i 実体集合 である。

実体集合と同様関係集合としてグループ化される。

例 「手続き」が「手続き」を呼び出す関係

（３）弱関係集合（Weak Relationship Set）　弱関係は特別な２項関係であり、Ｅ・Ｒモデルの存在依存の概念を表現したものである。この２項は親と子に分け、子は親に存在依存する。この親子関係は木構造として捉えることができる。

例　「手続き」（親側）と局所的に宣言する「型」や「変数」（子側）

（４）属性（Attribute）　属性の型としては、整数、実数、文字列、スカラー、テキストがある。特別な属性として「局所名」がある。これは実体とする仕様情報単位の名前を表現する。また、「局所名」の別名として「同義名」がある。

例　「手続き」の「機能概要」属性の型としてテキスト、「手続き」の名称として「局所名」

（５）キー　キーは、実体、関係、弱関係を一意に識別するための値であり、ＤＢＭＳが与える。また、各実体を一意に識別する名称として「完全名」を導入した。これは、弱関係による存在依存の木構造において根の実体から対象とする実体までその「局所名」をデリミタで区切って並べたものである。「完全名」は実体キーに相当する。

例　コマンド処理_制御（デリミタは_）

Ｅ・Ｒモデルスキーマの例を図１に示す。

３．表形式モデル

表形式モデルは、仕様書に用いられる表を一般化したモデルである。[5)] 表は欄というＣＲＴ画面上入出力を行う矩形領域の集りである。欄集りの上に以下のように表・見出し・記載形式という集りの単位（構成要素を密着させ並べた矩形領域）と、並べ方をモデル化する。

（１）＜表＞は＜見出し＞と＜記載形式＞の並びである。ここで、＜見出し＞と＜記載形式＞の並べ方に縦型（上と下）と横型（左と右）がある。

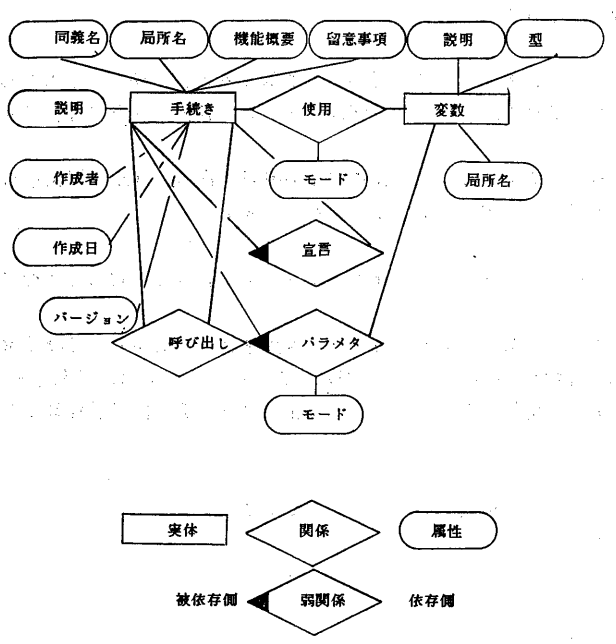


図１．Ｅ・Ｒモデルスキーマ例

手続き仕様書

名称	制御			環境名	コマンド処理
略称		説明			
使用手続き	手続き名		説明		
	コマンド処理_入力解析		入力コマンドを解析する。		
使用変数	変数名		説明		モード
機能概要	コマンド処理パッケージの各手続きの実行を制御する。				
留意事項					
引数	引数名	型名	説明		モード
作成者	幸谷博之	作成日	８５－０２－０５	バージョン	１０－０２

図２．表仕様書の例

（２）＜見出し＞は＜欄＞、または複数の＜欄＞の並び、である。
（３）＜記載形式＞は＜欄＞、複数の＜表＞の並び、または＜記載形式＞の配列並び、である。ここで、＜記載形式＞または＜表＞の並べ方に縦型（上から下）と横型（左から右）がある。この配列並びに包含される欄を配列欄という。
（４）＜欄＞はＣＲＴ画面上入出力を行う領域である。欄は見出し欄とデータ欄に分類する。
ここで＜見出し＞は「名称」や「機能概要」等の説明文であり、（１）～（４）の再帰的定義により階層構造を持つ表を表現できる。
表仕様書の例を図２に示す。

４．ユーザインタフェース

テキストエディタや構造エディタ等の会話型エディタでは、編集プロセスは次のよう対話を行うといわれる。（文献６　Ｐ．３２２より）
（１）操作対象となる部分をビューとして選択
（２）このビューの書式（format）を決定し、表示
（３）対象に対する操作の記述と実行
（４）次の適切なビューへの更新
ソフトウェア仕様を対象とした場合においても、上記のプロセスとなろう。
この場合の特徴は、各ビュー間にある種の関連があることである。例えば、「手続きＰ１」から「手続きＰ２」を呼び出す関係（Call）がある場合、「手続きＰ２」のパラメタの追加は「手続きＰ１」の仕様変更を必要とする。従って「手続き仕様書Ｐ２」から「手続き仕様書Ｐ１」へのビューの更新が必要となる。このような場合コマンドにより、関連あるビューへ即座に更新する機能は有用であろう。
以上の点を踏まえ、各ビューを状態、ビュー更新を遷移とする状態遷移モデルをユーザインタフェースモデルとして採用する。

４．１．画面形式

本エディタは、ＣＲＴ画面に複数の論理的な画面を表示し、互いに関連した情報を入出力しながら編集を行う。物理画面は４つの領域からなる。（図３）
（１）メッセージ領域　エラーメッセージなどの表示
（２）対話領域　表仕様書、メニューの表示
（３）コマンド投入領域　コマンドをテキスト形式で入力
（４）ソフトキー領域　　キーボード上の機能キーで入力可能なものをＣＲＴ画面上に表示するもの

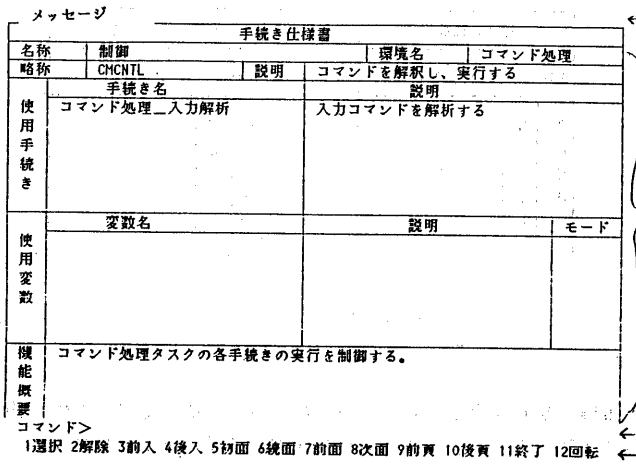


図３．画面形式

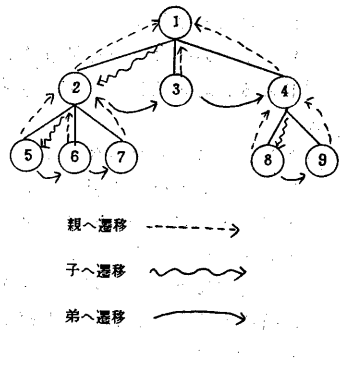


図４．ビュー間遷移

4．2．表示スコープ

表示画面領域が限定されている場合、個々のビュー全体を表示することは不可能である。また仕様書の標準化の観点から画面の形式を動的に変化させるのは好ましくない。このため、次の2種のスクロールによって各ビューのすべての情報を表示する。

（1）グローバルスクロール　あるビュー全体を上下にスクロールする。

（2）ローカルスクロール　配列欄単位に上下にスクロールする。

4．3．ビュー間遷移体系

ここではビューの更新をビュー間遷移と捉え、操作性を考え、ビュー遷移体系を図4のような木構造とする。ビューの遷移はあるビューで遷移先ビュー検索のキーとなる情報（手続きの名称など、これを遷移キーという）を選択することによって行う。複数個の遷移キーを選択することができる。木構造には、

（1）親子関係　遷移元を親、遷移先を子する関係

（2）兄弟関係　選択した複数の遷移先ビュー間の関係、がある。

ここで、編集作業を誘導し、操作性の高いユーザインタフェースを提供するため、遷移先仕様書の形式は予め設定する必要がある。

また、遷移ビュー記憶ユーザ負担を省くため、先祖及び弟群の遷移キーは本エディタが記憶する必要がある。

遷移機能には、

（1）親への遷移

（2）子への遷移

（3）弟への遷移、がある。（例　図5）

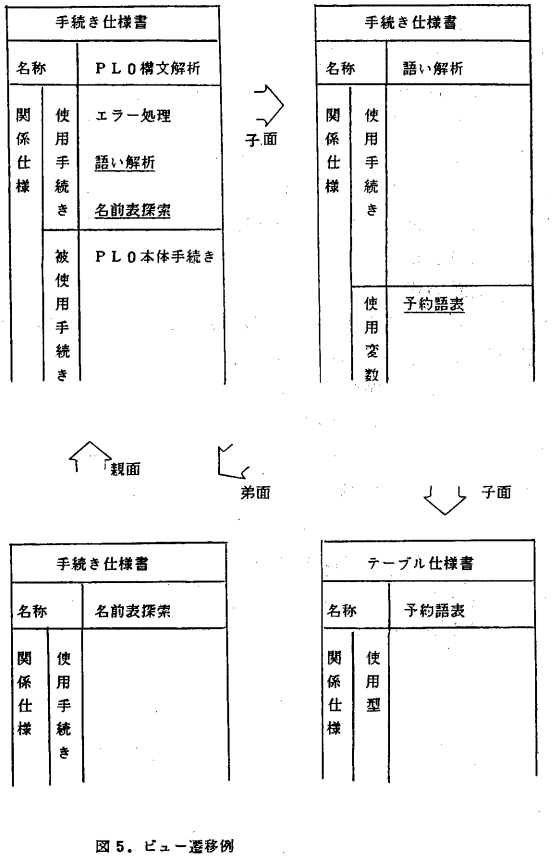


図5．ビュー遷移例

4．4．コマンド体系

本エディタは、（a）ガイダンス（ソフトキーによるコマンド誘導）、（b）完備性、（c）モードの最小化、（d）直交性、（e）コマンドの最小化、に留意したコマンド体系を持つ。

（1）位置付けコマンド

コマンドの対象（これを対象項目という）をカーソルによって指定するコマンド群である。

（2）ビュー遷移コマンド

親・子・弟へビューを遷移するコマンド群である。対象項目データを遷移キーとする。

（3）データベース更新コマンド

各ビューにおいて更新したデータを仕様データベースに反映させるコマンドである。

（4）スコープ指定コマンド

グローバルスクロール・ローカルスクロール機能を実行するコマンド群である。

（5）構造コマンド

カーソルが指示する配列欄データの挿入・削除・移動を実行するコマンド群である。従って複数の配列欄を含む記載形式についてはタブル挿入・タブル削除となり、表形式の整合性を保持することが可能になる。

（6）テキスト処理コマンド

テキスト項目において主に複数行に渡る機能（改行など）を実行するコマンドである。

（７）バリューコマンド

表形式構造には依存せず、データを複写・移動するコマンド群である。

また、コマンド誘導はソフトキー群をグループ化し、各グループをビューとすることによりビュー遷移と同様に行う。

４．５．Ｅ・Ｒモデルによる表形式モデルの意味付け

Ｅ・Ｒモデルは、実体・（弱）関係・属性をノード、実体や（弱）関係と属性の関連をアークとするネットワーク構造として把握可能である。一方、表形式モデルは３章で述べたように木構造として把握可能である。データベースの情報を表形式にて参照・更新するためには、Ｅ・Ｒモデルのデータ・これへの操作、と表形式モデルのデータ・これへの操作が対応する必要がある。このため、木構造を用いて表形式モデルの深層構造を表現する。この深層構造を表木という。

表形式モデルにおいて、欄データの更新は表木の葉ノード値の更新を意味し、欄データの挿入・削除は部分表木の生成・削除を意味する。

Ｅ・Ｒモデルの部分グラフを、あるノードを根とする木とみなし、サイクルをなす道はノードを複写し、切り離したものを表木と解釈することによって表形式モデルの操作を意味付けることができる。表木の解釈は次の規則に従って得られる。

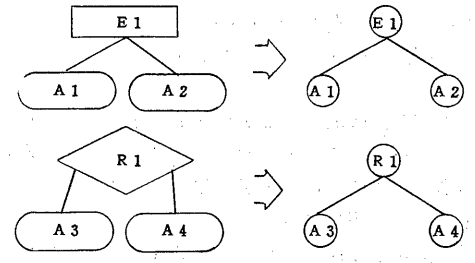
（１）実体・（弱）関係と属性の木表現

属性は、実体・（弱）関係の子あるいは子孫とする。（図６（ａ））

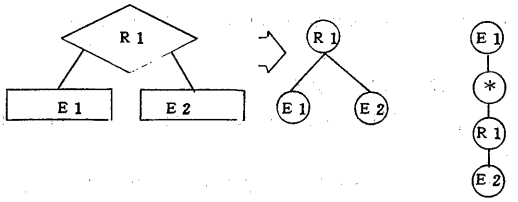
（２）実体と（弱）関係の木表現

（弱）関係で結ばれる実体について、これらの実体すべてを関係の子とする表現、または一個の実体を親とし（弱）関係をこの子孫　他の実体をこの関係の子とする表現、とする。（図６（ｂ））

図７にＥ・Ｒモデルで解釈した図１、図２に対応する表木の例を示す。図上＊のノードはこの子ノードのインスタンスが複数存在しうることを示す。このノードを列ノードという。



（ａ）実体・（弱）関係と属性の木表現



（ｂ）（弱）関係と実体の木表現

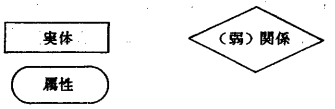


図６．Ｅ・Ｒモデルの木表現

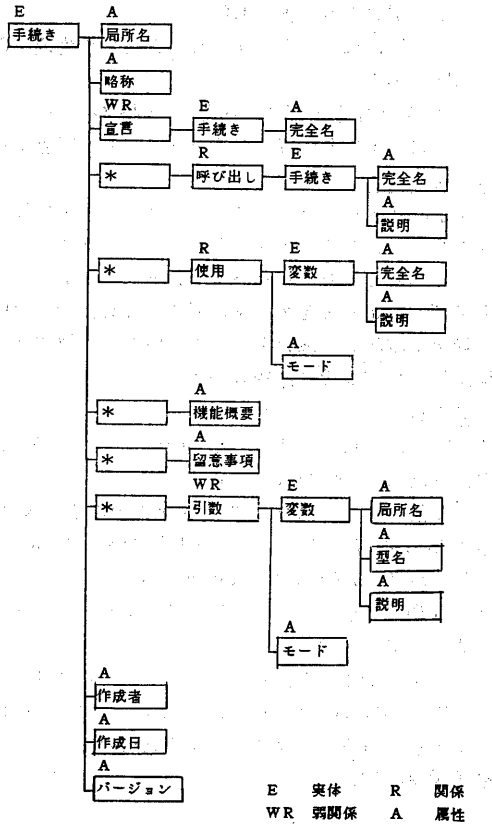


図７．Ｅ・Ｒモデルで解釈した表木の例

表木モデルではE・Rモデルに対応するノードを複写しているものがあるので、更新の意味を明確にする必要がある。このため、表木のノードに次の分類を設ける。

（1）実体定義側　表木上、

（i）先祖に実体・（弱）関係ノードがない実体ノード

（ii）（i）または（ii）を被存在依存側実体とする弱関係ノード及びその存在依存側実体ノード

（iii）（i）または（ii）の実体属性ノード

（2）実体参照側

（i）先祖に実体・（弱）関係ノードがない（弱）関係ノード

（ii）（i）、（ii）または（iii）をロール（実体）とする関係ノード、この関係による他のロールとなる実体ノード

（iii）（i）、（ii）または（iii）を存在依存側実体とする弱関係ノード、この被存在依存側実体ノード

（iv）（i）、（ii）または（iii）の実体属性ノード

（3）（弱）関係属性側

（i）（弱）関係属性ノード

これらの分類を用いて表木におけるノード値の更新と部分木の生成・削除をE・Rモデルで意味付ける。

（1）ノード値の更新

（i）実体定義側実体属性ノード　すべて実体属性値の更新を意味する。

（ii）実体参照側実体属性ノード　「局所名」、「完全名」については（弱）関係の更新を意味する。他は参照という観点から、無効である。

（iii）（弱）関係属性ノード　すべて（弱）関係属性値の更新を意味する。

（2）部分木の生成・削除

これは、列ノードの子となる部分木の生成・削除であり、E・Rモデルにおける実体・（弱）関係の生成・削除に相当する。

（i）実体定義側である実体・弱関係ノード　実体・弱関係の生成・削除を意味する。

（ii）実体参照側である実体・（弱）関係ノード　関係・弱関係の生成・削除を意味する。

5．あとがき

以上、E・Rモデルに基づくソフトウェア仕様データベースを表形式画面から編集するエディタのユーザインタフェースモデルを述べた。本エディタでは、E・Rモデルを仕様データベースのデータモデルとして利用するに加えて、表形式及びE・Rモデルによる表形式モデルの編集の意味付けをモデル化した。このため、E・Rモデルによる任意の仕様情報を任意の表形式画面から編集することが可能になる。

最後に本研究の機会を与えて頂いた日立製作所　大みか工場　久保岳明氏、ならびに有益な御討論を頂いた同工場　林利弘氏に感謝致します。

参考文献

1）Mori,T. et al:"PDAS:An Assistant for Detailed Design and Implementation of Programs",Proc. of 7th ICSE,pp.108-115(1984)

2）岸他1名:"対話型設計システムの作成支援ツール"，SE研資料32、1983

3）Chen,P.,P.,:"The Entity-Relationship Approach to Logical Data Base Design ",the Q.E.D.Monograph Series Data Base Manegement NO.6

4）大槻他3名："ソフトウェア仕様データベースインタフェースの一考察"，第29回情報処理全国大会予稿集p.569

5）山根、野木："表形式ドキュメントの汎用作成方式" 第28回情報処理全国大会予稿集p.537

6）Meyrowitz,N.,et al:"Interactive Editing Systems:PART1",ACM Computing Surveys,Vol14,No3,1982

7）車谷、野木："ソフトウェア仕様編集におけるユーザインタフェースの一方式" 第28回情報処理全国大会予稿集p.597