1S-3

ソフトウェア開発プロセスを支援するデータベースの一方式

*柳 一美、 大槻 繁、**中野 利彦、高橋 勇喜
*(株)日立製作所 システム開発研究所 **同左 大みか工場

1．はじめに

ソフトウェアの開発プロセスを総合的に計算機で支援するためには、これ等の活動に関わる情報を蓄積・管理するデータベースが必要である。この分野は近年、「ソフトウェア・エンジニアリング・データベース（以下ＳＥＤＢと略す）」と称されて注目されてきている[1]。ＳＥＤＢの基盤技術の開発では、従来の階層型、ネットワーク型、関係形式型等のデータベース技術に加え、ソフトウェア・エンジニアリングの局面からの研究アプローチが不可欠である。

本論文では、実体・関連モデルを基礎とし、ソフトウェアの各種成果物を開発プロセスの中で適切に扱えるようにしたデータベースのデータモデルについて論ずる。

2．ＳＥＤＢへの要求

（1）成果物の管理

ソフトウェアの開発プロセスでは、要求仕様、設計仕様、プログラム(ソースコード)、テストデータ等の様々な形態の成果物が生産される。

個々の成果物は、その内部で固有の構造を持っている。例えば、設計仕様では、関数、テーブル、データ、型等の要素と、これ等の間の呼出し、参照、包含等の関係に基づく構造が存在する。

成果物間の関係に基づく、マクロな構造も存在する。例えば、設計仕様に従ったプログラムのテスト仕様は、要求仕様を充足しているかを確認するためのテストデータを規定している。

ＳＥＤＢでは、上記の成果物内部、成果物間の構造を含めて、開発プロセス上で生成されるすべての情報を蓄積・管理することが要請される。さらに、各成果物を生成する作業工程の、いかなる時点においても利用可能になっていなくてはならない。すなわち、ＳＥＤＢのデータモデルには、工程内の各種成果物の不完全状態を扱えるしかけがなくてはならない。

（2）構成管理

開発途中で仕様が変更されることは少なくない。仕様の変更により、影響を受ける仕様があれば、その仕様も変更しなければならない。他の開発者が担当している仕様であれば、それを知らせなければならない。仕様間の整合性を保つためには、変更による影響の範囲や程度を解析する必要がある。

レビュー、テスト等において発見された不良は、取り除かなければならない。そのために、それ以前の開発段階へ遡って原因を分析し修正するバックトラック作業の支援が必要である。

設計仕様とプログラムとの不一致や仕様の変更によるプログラムの修正もれを防ぎ、プログラミング作業を低減するためには、プログラムを自動的に生成できることが必要である。

ワークステーションやネットワークの普及により、開発環境が分散化してきている。これに伴い、仕様は開発の必要性に応じて物理的に分割されたり、複製が作られたりする。こういった状況下でも仕様の論理的一貫性を保つためには、分散化に対応した仕様の構成管理が必要である。

各工程の取り纏め担当者などの管理者は、誰が何を担当しているか、どこまで開発が進んでいるか、予定と比べ遅れている部分はないかなど開発状況を管理するプロジェクト支援機能が必要である。

3．データモデル

3．1 概要

データモデルは実体･関連モデルを基礎としている[2]。その主な拡張上の特徴は、次の通りである。

（1）関連集合は、一般的にｎ項でもよい。これにより、プロジェクトやメンバの込み入った契約関係をも表現できる。

（2）関連集合は、実体化することができ、その時にのみ属性をもつことができる。これにより、操作をすべて実体集合の操作として定義することができ、体系が単純化できる。

（3）関連に存在依存性を定義することができる。これにより、ある仕様が他の仕様に存在依存していたり、包含されている場合を表現できる。

（4）ある実体集合に対し、その要素を表示する表現式を導入し、その表現式の示す実体が存在しなくてもよい過渡的な状態を記述できる。これを半関連というのに対し、実体が存在する関連を正関連という。

（5）属性は、省略可能あるいは必須、多値あるいは単一値である。この規定により、実体集合と値集合との存在関係を明確に扱える。

（6）属性の値域は、直積、直和、列に基づく構造（複合型）を持っていてもよい。これにより、構造に文法をもったものが扱える。

"A Method of Database for Software Development Process Environments",
Kazumi Yanagi , Shigeru Otsuki, Tosihiko Nakano, Yuki Takahasi
Systems Development Laboratory and Omika Works ,Hitachi,Ltd.

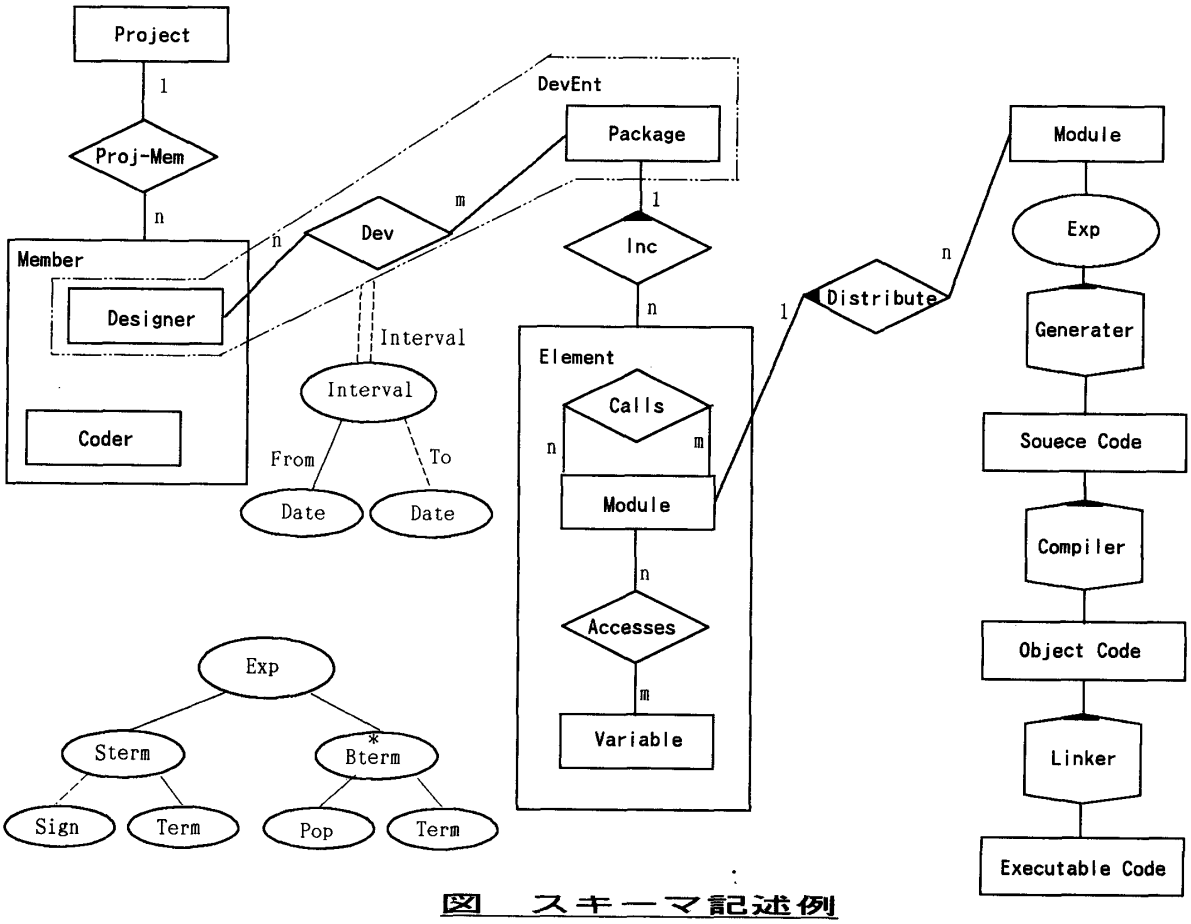

図　スキーマ記述例

（7）ある仕様はその仕様の前提となっている仕様を、ある規則に基づいて構造変換したものであることを記述できる。（導出関連）

（8）原本とその複製、あるいは、原本とその一部など分散した仕様の関係を記述できる。（分散関連）

3．2　例題

図にスキーマの記述例を示す。これは、設計・プログラミング工程でのプロジェクト概念と構成管理を付加したスキーマである。ここで表現されている事柄は、以下のような事柄である。

（1）ProjectのMemberとして、DesignerとCoderがいる。DesignerはProjectの開発(Dev)を任されていて、開発期間(Interval)が管理されている。

（2）ModuleがModuleを呼び出す(Calls)という関係は、必ずしも、呼び出すModuleの存在を仮定しなくてもよい。このModuleの仕様を作成している段階では別のModuleはまだできていないが、Moduleが別のModuleを呼び出す関係は、半関連CALLSで定義できる。別のModuleの仕様が作成されると、この半関連は正関連CALLに変換される。

（3）ロジック設計は、モジュール設計で定義されたModuleの外部仕様を複製(Distribute)してきて別の場所で行う。設計が終わるとこの関係をたどって仕様を統合できる。

（4）Moduleの内部仕様は、複合的な構造をもつ属性Expによって記述される。Source Codeはある規則に基づく構造変換によって作成(Generate)される。Moduleのインタフェースを変更すると、複製・分配(Distribute)によって作られた Moduleの仕様およびこれから派生している Source Code から Object Code、Executable Codeまで変更される。

4．おわりに

ソフトウェア開発環境を指向したデータベースでは、ライフサイクル全般にわたり生産されるすべてのソフトウェア・オブジェクトを、開発プロセスの各段階に応じて適切に管理しなくてはならない。ここでは、基本的な枠組みとしてデータモデルを中心に提案した。本提案は、多人数による分散開発方式を含めた開発プロセス・モデリングの定式化の道具としても有効である。

<参考文献>

[1] L.A.ROWE："Report on the 1989 SOFTWARE CAD DATABASES WORKSHOP",IFIP1989 pp719~725

[2] 大槻他１名：「ソフトウェア仕様を対象としたプロセス代数」，１９８９年３月，ＣＡＳＥ環境シンポジウム論文集 pp15~22