ンドウシステムはこの WA に含まれると考えることができる.

• Dialogue Manager (DM)

WA が提供する低レベルの入出力プリミティブを対話技法に組み上げる. これは AP や WM に対するメディア独立なインタフェースの提供と見ることができる.

• Workstation Manager (WM)

ユーザが同時に複数の対話を行う際の文脈の切り替えなどの「メタ対話」の管理及び複数の対話の間のデバイス共有の管理を行う.

• Application (AP)

アプリケーションの意味に関する処理を行う. つまりアプリケーションの本体部分である.

ここでメディア依存/独立という言葉が出てくるが, これは, 従来よく言われるデバイス依存/独立と違い, キャラクタやイメージといったデバイスの型 (メディア) に依存するか否かということを指している.

このような参照モデルを実装するにあたっては, 複数のデバイスの同時利用や, ネットワークやマルチプロセッサの利点を最大限に活かすということから, 対話型ソフトウェアのマルチプロセスを用いた構成が必要である. 特に同じアドレス空間を共有する複数のプロセスがチームを構成して共有メモリを通して交信し, チーム同士がメッセージを用いて交信するという階層構成が望まれる. 参照モデルにおける各要素がこのプロセスチームを構成することになる. 例えば, DM のチームにおいては独立した複数のプロセスがそれぞれひとつの対話に使われ, それらがユーザのプロファイルや対話の仕様を維持するための共有データベースをアクセスする. このような参照モデル及びその実装法を採ることにより, マルチウィンドウの環境でユーザが同時に複数の対話に従事するようなユーザインタフェースシステムを構築することができる.

[評] これまでひとつのアプリケーション内での対話の記述方法が主であった UIMS の研究にたいして, 複数の対話への対応やウィンドウシステムなど周りの環境に対する考慮を与えたという点で興味深い文献である. ここで提案された参照モデルは Stanford 大で開発された分散 OS である V-System のインタフェース部分の基礎となっており今後の分散環境でのユーザインタフェースの構築方法に大きな影響を与えると思われる.

この論文及びほかの三つのワーキンググループの報告は現在及び今後のユーザインタフェースの研究の動向を知るうえで一読の価値がある.

(三菱電機(株)・情報電子研究所 辻 順一郎)

88-34 設計環境支援用データベースにおける変更の管理

R. H. Katz and E. Chang: Managing Changes in a Computer-Aided Design Database

[*Proc. of 13 th VLDB Conference*, pp. 455-462 (1987)]

Key: Object-oriented data models, computer-aided design databases, inheritance, change propagation, constraint propagation.

計算機支援設計 (CAD) システムは, 機械系, 電気系などさまざまな分野で実用化が進んでいるが, 今後の発展の方向性として, 統合的なシステムの構築が考えられている. 統合的環境においては, そこで利用される各種情報を蓄えるデータベースが必要不可欠である. このような背景から, CAD データベースが今日広く研究の対象とされている.

本論文の著者は, VLSI 設計に焦点を絞り, それに適したデータモデルの構築, 並びにそのモデルに基づいたシステムの試作などを行ってきている. ここでは, カリフォルニア大学バークレー校において著者らが開発している Version Server と呼ばれる試作システムにおいて, 設計活動の進展にともなうデータの変更をどのように管理しているかを述べている.

ここで使われているデータモデルはオブジェクト指向の概念を利用し, VLSI におけるモジュール間の階層構造をクラス間の階層構造になぞらえている. 設計の最小単位は「設計対象 (design object)」と呼ばれる. この設計対象には, 設計情報を記述したファイルなどの形で実体が存在するもの (representational object) と, それ自身の実体はなく階層構造を組み上げるために使われるもの (structural object) の二種類がある. また, 設計対象同士は三種類の関係によって互いに関係付けられている. 三種類の関係とは, 「バージョン履歴」, 「配置」, それに「同値」である. バージョン履歴の関係は, 新たなバージョンが作られたときに, それがどのバージョンを基にして作られたかを示すものである. 配置の関係は, ある設計対象がほかの設計対象によって構成されていたり, または, ある設計対象の一部となっていたりする様子を表現する. また, 同値の関係は, 同一対象に関する種類の異

なる表現（マスクパターンとゲート結線図など）を関係付けるものである．

このようにして表現された設計対象間の関係は，設計対象をノードとするループを持たない有向グラフを構成する．ここで起こり得る最も簡単な変更は新しいバージョンの追加である．この新バージョンはその親のバージョンと同じ値を初期値として持っているのが望ましい．これはオブジェクト指向環境の持つ継承機構を利用することによって実現できる．

一般に，階層構造の一部に変更が生じるとそれは構造全体にさまざまな形で伝播し，階層構造を上にたどっていくにつれて，可能な伝播経路の数は急速に増大する．そこで，伝播の範囲を限定し可能な経路の数を最小限に抑える必要が生じてくる．本論文ではこのような問題を解決する方法がいくつか提示されているが，ここではその中からグループチェックイン/チェックアウトの手法について紹介する．A，B，C三つの部品があり，AはBとCから構成されていると仮定する．ここでB, Cがそれぞれ変更されてB′, C′となったとする．この時，Aにはどのような形で変更が伝播されるだろうか．すべての可能性を網羅しようとすれば (B′, C), (B, C′), (B′, C′) の三つの可能性がある．しかし，設計者の意図が「BもCも作り直して新しいAを作ろう」というものだったとすると，最初の二つは必要ないことになる．このような時は，BとCをまとめてグループとしてデータベースから取り出し，変更を加えた後 B′ と C′ のグループとして再びデータベースに登録するというのが，グループチェックイン/チェックアウトの考え方であり，可能性の数を大幅に減らすことができる．

［評］ 従来の方法では，変更時にその自動的な伝播が行われていなかったのに対して，本論文ではそれを積極的に取り入れたところが最大の特徴といえよう．ここで使われているモデルは，簡潔でありながらも強力な表現能力を持っている．今後の発展が期待されるところであるが，実用的なシステムを構築するためには，既存の CAD ツールとの結合などいくつかの問題点も残されている．

（東大・理 白井靖人）

88-35 Warren 抽象マシンに関する経験的考察

Touati, H. and Despain, A.: An Empirical Study of the Warren Abstract Machine

[*Proc. of 1987 SLP*, pp. 114-124 (Sep. 1987)]

Key: Prolog, abstract machine, program analysis, compiler optimization.

D. H. D. Warren によって提案された Warren 抽象マシン（以降 WAM）は，Prolog 処理系の効率的な実装を行うための仮想マシン・アーキテクチャを規定したものであり，その発表以来，Prolog マシンを初めとする多くの Prolog 処理系が WAM をベースとして開発されてきている．また，そういった状況を反映して，ここ 1〜2 年のロジック・プログラミング関連の国際学会では WAM と銘打ったセッションが設けられ活況を呈している．WAM に基づく Prolog マシンの代表的なものに，カリフォルニア大学バークレイ分校で研究・開発され，昨年 ZENOLOGIC 社から X-1 という名前で製品化された PLM (Programmed Logic Machine) が挙げられる．この PLM はマイクロプログラミング方式によって WAM の命令セットを実現したものであり，第5世代プロジェクトの PSI と共に Prolog マシンの草分け的存在である．

本論文では，PLM の開発経験をもとに，WAM に基づく Prolog 処理系を構築する際に考慮すべきいくつかの項目についてベンチマーク・プログラムの解析結果に基づいて論じられており，Prolog 処理系の構築に興味を持つ者にとって非常に有益な内容となっている．ベンチマークとして用いられているのは，ハノイの塔や8クイーンといった数十行から数百行程度の19の Prolog プログラムであり，それらのプログラムを PLM のシミュレータを改造した WAM エミュレータを用いて実行することによって解析に必要な各種のデータを収集している．

評価が行われているのは，①ユニフィケーション実行時に参照ポインタをたどってゆくデリファレンス処理の効率化，②変数タグを参照タグから独立させた場合の得失，③ユニフィケーション・ルーチンの効率化，④バックトラック処理に用いられるチョイス・ポイント・フレームの生成の最適化，⑤複数の節呼び出しにまたがる変数を格納するためのエンバイロメント・フレームの再利用，⑥変数への値の代入にともなうトレイル判定の効率化，⑦リストの CDR 表現の有効性といった7つの項目である．

これらの評価項目に関するデータがそれぞれ表としてまとめられ，それらのデータをもとに各項目に対するコメントが述べられている．それらの内のいくつか