解　説

4. オブジェクト指向データベースの応用

4.2 オブジェクト指向データベースのCADへの応用†

宇田川　佳久††

1. はじめに

CAD (Computer Aided Design) データベースの必要性が広く認められだしたのは，1980年代に入ってからである．CADツールの充実と，エンジニアリング・ワークステーション (EWS) の普及によりCADが一般に普及し始めた時期である．データベースの分野では，リレーショナル・データベースが実用期を迎えている．リレーショナル・データベースは，事務処理を前提に研究開発されてきたものであるが，この時期になるとCADデータベース管理への適用の研究も盛んになっている．これらの研究によって，CADデータを扱うためには，リレーショナル・データモデルの枠組を越えたデータモデルが必要であるとの指摘がなされた[16),17),29)]．

図面を中心とするCADデータは，階層構造があり，しかも，各階層のデータに特有の手続きが付随する．このため，データの階層構造を積極的に扱い，データと手続きとを一体として扱うオブジェクト指向データベースがCADデータを管理するうえで適していると指摘する研究者が多い[7),8),18),22),30),31),34),36)]．CADデータの管理にオブジェクト指向データベースを使うことによる効果は，データ操作を容易にすることや，処理効率の向上にあるが，定量的な効果を確認するためには，今後の研究開発を待たなければならないのが現状である．

本文では，CADシステムにおけるデータの一元管理の必要性と分類，CADデータベースの開発過程，それに代表的な研究事例を紹介する．

2. CADデータの一元管理とデータベース

2.1 CADデータベースの出現背景

1980年初めまで，データベースといえば，事務処理に限られてきたわけであるが，計算機，とりわけEWSの普及によりCADの環境が大幅に変化し，CADデータベースの研究開発が活発になってきた．EWSの高度な対話機能と良好なレスポンスにより，設計の上流段階から計算機を利用して設計が進められている．EWSの普及以前は“清書”が中心であったことを考えると，確かに一世代変わったのだということを実感することができる．

さて，“清書”以上の仕事をするために，ソフトウェアも大幅に充実してきた．図面入力ツール，シミュレータ，設計ルール・チェッカ，加工データ生成ツール，レポート生成ツールなどがその例である．このように，CADシステムが多数のツール群によって構成されるようになると，(1)ツールに共通した使い勝手の良いユーザ・インタフェースと(2)ツールに提供するデータを一元管理するデータベースが，システムの使い勝手と性能向上に大きな影響を与えるようになる．すなわち，図-1に示したように，各種のツールをユーザ・インタフェースとデータベースではさみ，利用者には同一のユーザ・インタフェースを提供する．一方，ツールの開発者にはツール間のデータの整合性を容易に取れるようデータベースが提供されている．設計作業には変更が付きものなので，ツール間のデータの一元管理は特に重要である．CADデータベースは，このような背景で研究開発されているものである．

2.2 CADにおけるデータ管理の分類

CADにおけるデータ管理は，大きく3種類に分類することができよう．

† Application of Object-Oriented Databases to CAD by Yosihisa UDAGAWA (Information Systems and Electronics Development Lab., MITSUBISHI ELECTRIC Corp.).

†† 三菱電機(株)情報電子研究所

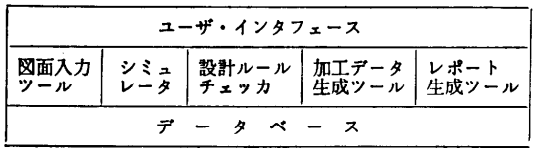


図-1 CAD システムの構成例

(1) 一つの設計対象を複数のエンジニアが協調して設計することを支援する.

(2) 異なる種類の CAD ツールにまたがったデータの一元管理を支援する.

(3) 異なる CAD システム間でのデータ交換を支援する.

CAD システムが強力なものになるにつれ, CAD で設計する対象物も複雑になってきた. 設計対象物を, いくつかのモジュールに分解し, それぞれのモジュールを多くのエンジニアによって共同開発することも珍しくない. 今や, CAD は個人レベルの設計作業を支援する段階を越え, 多くのエンジニアによる共同作業を支援する段階になっている. (1)のレベルの CAD データ管理は, たとえば図面編集ツールといった, 同じ種類の複数の CAD ツール間で, データの共同利用を可能にすることにより, CAD システムの効率的な利用を可能にしようとするものである.

(2)のレベルの CAD データ管理は, 異なる種類の CAD ツールを統合することにより, 計算機による製図—解析—シミュレーション—製造データ生成—レポート生成などの一連の設計作業を合理化しようとするものである. 目指すところは, CIM (Computer Integrated Manufacturing) の実現である. CAD ツールの独立性を保証しつつ, ツールを協調して動作させることが技術的な課題である.

(3)は, データ交換の標準化に関するものである. CAD システムの導入は, 長期にわたることもめずらしくない. CAD の導入過程で, はからずも (あるいは政策的に) マルチ・ベンダ構成となることがある. 業務を効率的に処理するためには, 異なるベンダから提供されている CAD システムの間で, データ交換をしなければならないことがある. 機械系 CAD のデータ交換標準として IGES (Initial Graphics Exchange Specification)[1], STEP (STandard for the Exchange of Product model data), また, 電子系 CAD のデータ交換フォーマットとして EDIF (Electronic Design Interchange Format)[12] などが使われている. これらの標準は, データの交換フォーマットを規定することを主な目的としている.

"CAD データベース" を, CAD データの標準フォーマットとデータの一元管理を提供するものと定義するならば, CAD データベースは上記の(1)と(2)の範囲のデータ管理を行うものと位置付けることができる.

2.3 CAD データベースの開発経緯

2.3.1 1970 年代の研究開発

CAD データベースの最初の研究がいつ行われたのか定かではないが, 1970 年代中期以前であることは確かである[24]. この章では, CAD データベースが一つの研究分野として認められ, 研究論文が継続的に発表されるようになってからの経緯を述べる.

1970 年後半までは, CAD データベースの必要性と要件についての議論が盛んであった[27),29)]. また, 限定された範囲での CAD データベースの開発が行われた. たとえば, Ciampi[9] らは, LSI CAD 用のデータベース構造を論じ, LSI データを表現するためには, デバイス (部品), ピン (端点), 接続情報と信号が必要であることを述べている. このほか, Wong[35], Kenneth[21], Zintl[37], Keller[19] らも LSI CAD 用のデータベースについて報告している. 建築 CAD におけるデータベースについては, Eastman[11] の研究がある.

これらのデータベースは, 独自のファイル・システムに基づいたものと, CODASYL 型に基づいたものに大別できる. 独自のファイル・システムを使ったものは, 高いパフォーマンスと特定業務に対する豊富な機能が提供されているが, 体系的なデータ操作言語, 並行処理機能, バックアップ・リカバリ機能などが不十分であるとの指摘がなされている. 一方, CODASYL 型データベースを使ったものの問題点としては, データ独立性の低さが指摘されている. CODASYL 型データベースでは, データ項目どうしをポインタで結合してデータを記憶している. ポインタは物理的な記憶アドレスと密接に関係しているから, データ構造に変更が起きると, データベース全体のダンプと再ロードが必要になる. さらに, アプリケーション・プログラムである CAD ツールの書き換え

も余儀なくされる．このころからCADツールの開発が活発に行われたため，データベースの構造とCADツールとのデータ独立性が重視され始めた．

2.3.2 1980年代前半の研究開発

1980年代に入ると，リレーショナル・データベースが普及し始める．リレーショナル・データベースは，高いデータ独立を特徴としているものである．リレーションという簡潔なデータ構造，動的なデータ項目の追加機能，集合演算によるデータ操作，ビュー機能により一つのデータに対していろいろな見方を定義することができる，といったことが高いデータ独立の達成を可能にした．当然，リレーショナル・データベースのCADへの適用が盛んに検討された．特に，リレーショナル・データベースには明快な数学的な基盤があったために，学会での議論も活発であった．これらの研究の結果，次のようなリレーショナル・データベースの限界が明らかになった[16),17),29)]．

（1） データの階層を扱う機能が不十分である．

（2） 図形・画像データの扱いと同一物に対する異なるデータ表現の等価性を扱うことが難しい．

（3） バージョン（改訂版）の管理機能が不十分である．

（4） 設計作業に合ったトランザクション管理機能がない．

一方では，リレーショナル・データベースに基づいたCADデータベースの提案が盛んに行われた[15)]．System Rの改良[14)]およびINGRES/POSTGRES[31),32)]といった体系的な開発も行われるようになった．これらの一連の研究により，密接に関連した異種のデータの集まりがCADにおける操作単位となることが指摘され，データの階層構造の必要性も広く認められるようになった．

2.3.3 1980年中期以降の研究開発

1980年中期以降は，CADデータのモデル化が議論されるようになった[6),8),18),30),34),36)]．リレーショナル・データベースの限界として指摘された上記の項目が議論の中心である．以下，研究の状況を簡単に述べる．

（1） データの階層を扱う機能

設計対象物は，いくつかの部品を組み合わせて作られることが多い．こうして作られた設計物は，さらに大きな設計物の部品として使われる．このように，設計対象物は部品の引用関係によって階層構造を作っている．一方，部品を使うとき部品の内部構造を知る必要はない．部品の外から見える性質（LSIなら端点に出入りする信号の仕様）が分かれば部品を使うことができる．このような考えから，CADデータを内部構造を表現する部分と，外から見えるインタフェースの部分とに分けて管理する手法が提案されている[6),18),34)]．

（2） データ表現の等価性を扱う機能

設計対象物は複数の表現をもつことがある．たとえば，LSI回路の場合，論理図，トランジスタ接続図，マスクレイアウト図などがある．ソフトウェアならば，仕様記述，ソース・プログラム，オブジェクト・プログラムといった表現をもつ．一般に，一つの表現からほかの表現への変換は，複雑な処理を経て行われる．したがって，データを見ただけでは同じ対象物を表現しているかどうか判断することができないことが多い．このため，等価性を扱うメカニズムが必要であると指摘され，これを実現する手法が研究されている[7),18)]．

（3） バージョン管理機能

設計は改良を加えながら行われるものであるから，バージョンの管理が重要になる．設計が，より良いものへと直線的に改善されていくのであれば話は簡単であるが，一般には，“3日前の設計を使いたい”などと，過去の設計に後戻りすることも起こり得る．また，一つの設計対象物に対する改善案もいくつか存在することが普通で，もとの設計対象物から複数のバージョンが作り出される．

Changと Katz[7)]は，図-2に示したCADデー

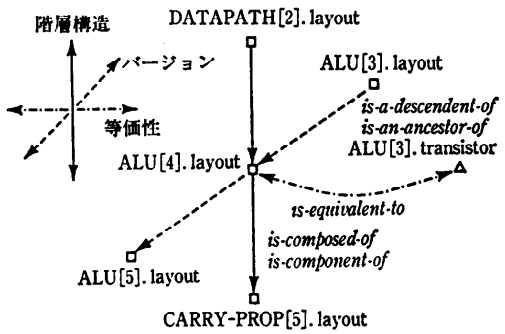


図-2 CADデータモデリングのフレームワーク

タ・モデリングのフレームワークを提案している．一つの CAD データは三つの独立した次元の中に存在する．まず，データの階層構造の次元がある．図-2 の ALU [4]. layout というレイアウト図は DATAPATH [2]. layout というレイアウト図から引用されており，CARRY-PROP [5]. layout というレイアウト図を部品として引用している．第2の次元はデータ表現の等価性である．ALU [4]. layout なるレイアウト図は，ALU [3]. transistor という等価なトランジスタ図に対応している．第3の次元がバージョンである．ALU [4]. layout は ALU [3]. layout の改良版であり，さらに，ALU [5]. layout という改良版のもとになっている．一つの設計対象に対し複数のバージョンが存在するために，ある時点でオーソライズされているバージョンを定め，これをカレント・バージョン (Current Version) と呼んでいる．

（4） 設計トランザクション管理

データベース管理システムは，論理的に完結した一連のデータ処理を実行するのならそれらをすべて行い，なんらかの理由で完結できなかったら元の状態に戻すというデータ実行制御を行っている．この制御を，トランザクション管理と呼んでいる．

事務処理におけるトランザクションは，トランザクションの開始—処理の集まり—トランザクションの終り，という単純な構造をしている．一方，CAD データの処理では，トランザクションの中に小さなトランザクションが含まれる構造になることが知られている．すなわち，トランザクションが入れ子構造をしている．このような構造になったのは，CAD ツールで1回に処理されるデータが独立性の高いモジュールを構成しており，しかも，このモジュールに含まれるデータ量が多い（数百バイト～数メガバイト）ことによる．また，CAD システムが，EWS をネットワークで結合した分散システムの形態をしていることも，CAD データベース特有のトランザクション管理を必要とする要因である．

図-3 は，典型的な設計トランザクションの管理方式を示している[18]．設計作業はワークステーションで行われ，CAD データはファイル・サーバが管理している．ワークステーションから

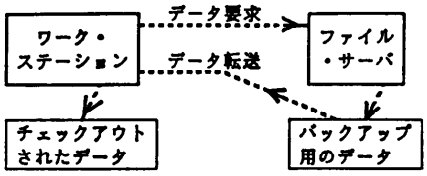


図-3 設計トランザクション管理の例

CAD データの転送要求があったとき，サーバは要求されたデータが貸し出し (Check-Out) 可能かどうか調べ，可能ならバックアップのために転送要求があったデータのコピーをサーバ内に作る．続いて，データをワークステーションに転送する．なお，転送の対象になる CAD データは，カレント・バージョンのデータである．

3. CAD データ管理とオブジェクト指向データベース

3.1 CAD データの特徴と CAD データベースの要件

CAD データを扱う機能が明らかになるのにともない，オブジェクト指向データベースの研究が活発になった．CAD データを管理する上で，オブジェクト指向データベースが従来のデータベースよりも適しているとされるのは，次のような理由によるものと考えられる[4),18),22)]．

第1の理由は，利用者からみた CAD データに階層構造が存在し，各階層の特有のデータ操作／一貫性管理のための手続きが付随していることである．このため，従来のデータ構造を中心としたモデリングでは限界があり，手続きをも含めたモデリングの枠組が必要になってきた．第2の理由は，CAD データが物理的に分散していることである．サイト間でデータの表現形式が異なっていたり，サイト間でのデータの一貫性を維持するために，特定の手続きを実行する必要があることが多い．これらの手続きは，データベース・スキーマと密接に関係しており，オブジェクト指向のアプローチが有効とされている．第3に，CAD では改訂が頻繁に行われるので，バージョンの管理が重要になる．バージョン管理は，利用者からみたオブジェクトを単位とすることが自然である．第4に，CAD データは多種の表現形態と構造をもつために，固定的なデータ型（たとえば，リレーショナル・データベースではタプルの集合）を前

提としたデータベースでは扱いがかえって難しくなる．むしろ，基本となるデータ型から，目的に合致したデータ型を組み立てることができるデータベースが有効であると考えられている．

3.2 CAD データのモデリング例

3.2.1 幾何形状の表現

多くのオブジェクト指向データベースは，プログラミング言語 Smalltalk または C＋＋と親和性のある言語インタフェースを提供している．ここでは，Smalltalk の流れを汲むデータベース Gem Stone[25] によって，幾何形状を表現した例を示す．

幾何形状の表現方法は，大きく二つに分類することができる[26]．その一つは，CSG (Constructive Solid Geometry) と呼ばれる手法で，基本幾何形状を組み合わせて目的とする形状を定義するものである．もう一つは，境界表現 (Boundary Representation) と呼ばれるもので，立体は面から，面は稜線から，稜線は端点から定義されるものとして形状を表現する．ここでは，紙面の都合で境界表現による例だけを示すが，同様の方法で CSG による形状表現も扱うことができる[20]．

図-4 は，最も基本的な境界表現のデータ構造を示している．端点 (Vertices) は，X 軸，Y 軸，Z 軸方向の座標によって表現される．稜線 (Edges) は端点の集合によって表現され，面 (Faces) は色を表す属性 color と稜線の集合によって表現されている．機械部品 (Mech-Parts) は，部品の名称 (name) と面の集合によって表現されている．図-4 の下側は，具体的なデータの値の例である．

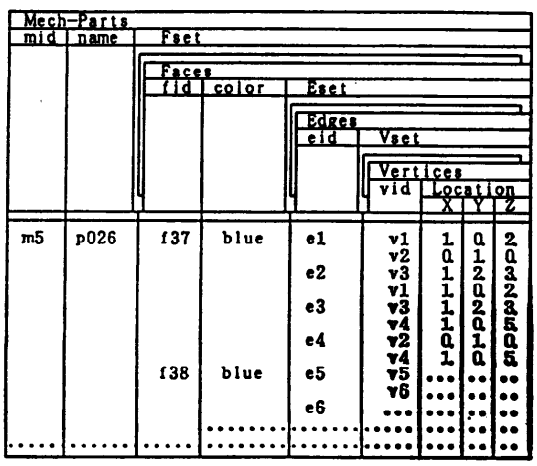


図-4 形状を表現するデータ構造の例

```
Object subclass : 'Location'
   instVarNames: #[ 'X','Y','Z' ]
   constraints: #[ #[ #X, Float ],
                   #[ #Y, Float ],
                   #[ #Z, Float ]].
Object subclass : 'Vertices'
   instVarNames: #[ 'vid','loc' ]
   constraints: #[ #[#vid,String],
                   #[#loc,Location]].
Set subclass : 'Vset'
   constraints: Vertices.
Object subclass : 'Edges'
   instVarNames: #[ 'eid','verset' ]
   constraints: #[ #[#eid,String],
                   #[#verset,Vset]].
Set subclass : 'Eset'
   constraints: Edges.
Object subclass : 'Faces'
   instVarNames: #[ 'fid','color',
                    'edgeset' ]
   constraints: #[ #[#fid,String],
                   #[#color,String],
                   #[#edgeset,Eset]].
Set subclass : 'Fset'
   constraints: Faces.
Object subclass : 'Mech-Parts'
   instVarNames: #[ 'mid','name',
                    'faceset' ]
   constraints: #[ #[#mid,String],
                   #[#name,String],
                   #[#faceset,Fset]].
```

図-5 図-4 のデータ構造の宣言例

3.2.2 データ構造の定義

図-5 は，図-4 に示したデータ構造を Gem Stone のデータ操作言語 OPAL によって定義したものである．オブジェクトの宣言は，三つの部分から構成されている．

（1） subclass：＜クラス名＞によってクラス名を宣言する．

（2） instVarNames：＜変数名＞によってインスタンス変数名を宣言する．

（3） constraints：＜データ型＞によってインスタンス変数のデータ型を宣言する．

図-5 の最初は Location（座標）の宣言をしている．#[...] は配列を意味している．また，変数を示すためには，変数名の先頭に # 印を付けた # ＜変数名＞を使う．システムが提供する基本データ型は，String, Integer, Float などである．2 番目は，Vertices を宣言している．インスタンス変数 #loc のデータ型は，先に宣言したクラス Location である．3 番目は，Vertices の集合 Vset の宣言で，Vset は 4 番目の宣言でインスタンス変数 #verset のデータ型として使われている．以下同様にして，図-4 のデータ構造が宣言されている．

3.2.3 メソッドの定義と実行

次は，オブジェクトにメソッドを定義する．メソッド定義の構文は，

```
method : <クラス名>
         <メッセージ・フォーマット>
         <メソッド本体>
%
```

である．メッセージ・フォーマットは，関数名（サブルーチン名）と引数を合わせたものに対応する．次に示したメソッドは，クラス Location にデータを登録するものである．

```
method : Location
  Xarg : xloc
  Yarg : yloc
  Zarg : zloc
    X :=xloc
    Y :=yloc
    Z :=zloc
%
```

クラスからインスタンスを生成するには

<クラス名> new

とする．new はシステムが提供しているインスタンスを作り出すためのメソッドである．この式によって生成されたインスタンスは，識別子が不明なので，インスタンスに対して操作を加えることができない．インスタンスの識別子を変数に保持するには，次のようにする．

<変数名> := <クラス名> new

こうして定義された変数を使えば，インスタンスに対して操作を加えることができる．たとえば，クラス Location のインスタンスを生成し，そこに座標 (1.0, 0.0, 2.0) を代入するには，次のようにする．

```
anLoc := Location new
anLoc Xarg : 1.0 Yarg : 0.0 Zarg : 2.0
```

データの更新，削除もデータの登録と同様にメソッドを定義することによって行うことができる．

4. おわりに

CAD データベースの必要性，研究開発の経緯と代表的な研究事例などについて述べた．CAD データベースは，今日の CAD システムの普及状況を考えると，ますます重要な技術になってゆくであろう．しかし，実現のための技術課題も多い．CAD データには，図形・画像・ドキュメントといった多数のデータが含まれる．さらに，データはワークステーションやファイル・サーバなどに分散して記憶されている．データの一貫性制約も複雑である．これらの性質は，事務処理にはみられなかったものもあり，リレーショナル・データベースの改良やオブジェクト指向データベースの開発をうながしてきた．しかし，現状では，オブジェクト指向データベースの普及はこれからであり，実際の CAD データ管理への適用は今後の研究開発を待たなければならない．

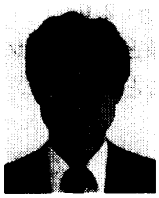
本文では，機械系 CAD データを例にして論じたが，オブジェクト指向データベースは，CASE (Computer Aided Software Engineering) やエンジニアリング・ドキュメントなど，幅広い分野に適用されようとしている．本特集には，これらのテーマについての解説も掲載されている．

参考文献

1) A Brief History of IGES—A Project of the Air Force ICAM Program. IGES Newsletter, National Bureau of Standards, pp. 1 (May 1980).
2) Andrews, T. et al.: Combining Language and Database Advances in an Object-Oriented Development Environment, Proc. of OOPSLA '87, pp. 430-440 (1987).
3) Atkinson, M. et al.: The Object-Oriented Database Manifesto, Proc. of Deductive and Object-Oriented Databases, pp. 40-57 (1989).
4) Atwood, T. M.: An Object-Oriented DBMS for Design Support Applications, Proc. of Compint-Computer Aided Technologies, pp. 299-307 (1985).
5) Barsalou, T. et al.: Complex Objects for Relational Databases, Computer-Aided Design, Vol. 22, No. 8, pp. 458-468 (1990).
6) Batory, D. S. et al.: Modeling Concepts for VLSI CAD Objects, ACM Trans. on Database Systems, Vol. 10, No. 3, pp. 322-346 (1985).
7) Chang, E. E. et al.: Inheritance in Computer-Aided Design Databases, Computer-Aided Design, Vol. 22, No. 8, pp. 489-499 (1990).
8) Chou, H. T. et al.: A Unifying Framework for Version Control in a CAD Environment, Proc. of 12th Very Large Data Bases, pp. 336-344 (1986).
9) Ciampi, P. L. et al.: Concepts in CAD Database Structures, Proc. of 12th ACM/IEEE Design Automation Conf., pp. 290-294 (1975).
10) Dadam, P. et al.: A DBMS Prototype to Support Extended NF2-Relations, Proc. of ACM SIGMOD Conf., pp. 356-367 (1986).
11) Eastman, C. M.: System Facilites for CAD Databases, Proc. of 17th ACM/IEEE Design Automation Conf., pp. 50-56 (1980).
12) EDIF: Electronic Design Interchange Format,

EDIF Steering Committee, Electronic Industries Association (1987).
13) Fishman, D. H.: Overview of the Iris DBMS, in Object-Oriented Concepts, Databases and Applications, Kim, W. et al. ed. Addison-Wesley, pp. 219-250 (1989).
14) Haskin, R. L. et al.: On Extending the Functions of a Relational Database System, Proc. of ACM SIGMOD Conf., pp. 207-212 (1982).
15) Haynie, M. N.: The Relational/Network Hybrid Data Model for Design Automation Databases, Proc. of 18th ACM/IEEE Design Automation Conf., pp. 646-652 (1981).
16) Haynie, M. N.: The Relational Data Model for Design Automation, Proc. of 20th ACM/IEEE Design Automation Conf., pp. 599-607 (1983).
17) Katz, R. H.: A Database Approach for Managing VLSI Design Data, Proc. of 19th ACM/IEEE Design Automation Conf., pp. 274-287 (1982).
18) Katz, R. H.: Information Management for Engineering Design, Surveys in Computer Science, Springer-Verlag (1985).
19) Keller, K. H. et al.: A Symbolic Design System for Integrated Circuits, Proc. of 19th ACM/IEEE Design Automation Conf. (1982).
20) Kemper, A. et al.: An Analysis of Geometric Modeling in Database System, ACM Computing Surveys, Vol. 19, No. 1, pp. 47-91 (1987).
21) Kenneth, A. et al.: A Vertically Organized Computer-Aided Design Data Base, Proc. of 18th ACM/IEEE Design Automation Conf., pp. 595-602 (1981).
22) Kim, W. et al.: Object-Oriented Database Support for CAD, Computer-Aided Design, Vol. 22, No. 8, pp. 469-479 (1990).
23) Lee, Y. C. et al.: Integration of Solid Modeling and Database Management for CAD/CAM, Proc. of 20th ACM/IEEE Design Automation Conf., pp. 367-373 (1983).
24) Losleben, P.: Data Structures, Data Base and File Maintenance, in Digital System Design Automation, Breuer, M. A. ed. Computer Science Press (1975).
25) Maier, D. et al.: Object-Oriented Database Development at Servio Logic, IEEE Database Engineering, Boral, H. et al. ed. Vol. 4, pp. 294-301 (1985).
26) Miller, J. R. et al.: Architectural Issues in Solid Modelers, IEEE Computer Graph. & Appli., Vol. 9, No. 9, pp. 72-87 (1989).
27) Nash, D.: Topics in Design Automation Data Bases, Proc. of 15th ACM/IEEE Design Automation Conf., pp. 463-474 (1978).
28) Pistor, P.: Designing a Generalized NF2 Data Model with an SQL-type Language Interface, Proc. of 12th Very Large Data Bases, pp. 278-285 (1986).
29) Sidle, T. W.: Weaknesses of Commercial Data Base Management Systems in Engineering Applications, Proc. of 17th ACM/IEEE Design Automation Conf., pp. 57-61 (1980).
30) Spooner, D. et al.: Modeling Mechanical CAD Data with Data Abstraction and Object-Oriented Techniques, Proc. 3rd IEEE Conf. on Data Engineering, pp. 416-424 (1986).
31) Stonebraker, M. R.: Application of Abstract Data Type and Abstract Indexes to CAD Data Bases, Proc. of Engineering Applications Stream of 1983 Data Base Week, pp. 107-113 (1983).
32) Stonebraker, M. R.: The Design of POSTGRES, Proc. of ACM SIGMOD Conf., pp. 340-355 (1986).
33) Velez, F. et al.: The O_2 Object Manager——An Overview, Proc. of 15th Very Large Data Bases, pp. 357-366 (1989).
34) Wilkes, W.: Complex and Composite Objects in CAD/CAM Databases, Proc. 6th IEEE Conf. on Data Engineering, pp. 443-450 (1989).
35) Wong, S. et al.: A Computer Aided Design Database, Proc. of 16th ACM/IEEE Design Automation Conf., pp. 398-402 (1979).
36) Zdonik, S.: Object Management System for Design Environments, IEEE Database Engineering, Boral, H. et al. ed. Vol. 4, pp. 259-266 (1985).
37) Zintl, G: A CODASYL CAD Data Base System, Proc. of 18th ACM/IEEE Design Automation Conf., pp. 589-594 (1981).

(平成3年1月 29 日受付)



宇田川佳久 (正会員)

1954 年生. 1982 年東京大学大学院博士課程修了. 同年三菱電機(株)入社. 現在, 同社情報電子研究所主事. エンジニアリング・データベースなどの研究開発に従事. 工学博士. 電子情報通信学会, 人工知能学会各会員.