データベースシステム 102－9
（1995．3．14）

# 文法表現によるオブジェクト指向データモデル

佐藤秀樹* 伊藤暢浩** 林達也**

*日本電装株式会社
**名古屋工業大学電気情報工学科

形式文法を基礎とするオブジェクト指向データモデルGROOM (Grammatically Represented Object-Oriented Data Model)は、オブジェクト指向データモデルの構造的要件である複合オブジェクト概念に、オブジェクト識別性、クラス、属性定義の継承・再定義などの概念を取り入れている。GROOMでは、オブジェクトの構造は生成規則を使ったクラススキーマとして定義され、オブジェクトの情報はクラススキーマ上の導出木として表現される。クラススキーマ間の上位／下位クラススキーマ関係は、関連するクラススキーマ間に成立する条件により定義される。この関係に基づき、クラススキーマ間には属性定義の継承・再定義、対応するクラス間の包含関係が存在する。

# A Grammatically Represented Object-Oriented Data Model

Hideki Sato * Nobuhiro Ito ** Tatsuya Hayashi **

*NIPPONDENSO CO., LTD.
**Dept. of Electrical & Computer Engineering, Nagoya Institute of Technology

GROOM(Grammatically Represented Object-Oriented Data Model) is an object-oriented data model based on the formal grammars. Structural requirements of the object-oriented data model such as complex object, object identity, class, inheritance and overriding of attribute definitions are incorporated in GROOM. The structure of an object is defined as a class schema using a set of production rules and the information of an object is represented in the form of a derivation-tree over the corresponding class schema. Super/sub relationships are defined based on the condition that the related class schemas satisfy.

# 1．はじめに

関係データベース(Relational DataBase, RDB)[1]を代表とする従来型データベースシステムは、事務データ処理分野への応用をその対象としてきた。しかし、1980年代以降、CAD/CAM、CASE、オフィス情報システム、グループウェア、マルチメディアなどの新たな応用分野が登場してくると、従来型データベースシステムの限界・問題点[2]が指摘されるようになってきた。このため、関係データモデルなどの欠点を克服する高水準データモデルが求められている。

非正規関係データモデル[3],[4]は関係データモデルを拡張し、関係が他の関係から構成されるようにした。これにより直接的に階層オブジェクトの表現が可能となった。また、意味データモデル（実体関連モデル[5]、関数型データモデル[6]、SDM[7]、Format[8]、IFO[9]）は、データベースの構造にデータの意味を反映させることを目指してきた。このために、集約化(aggregation)、汎化(generalization)、抽象化(abstraction)、グループ化(grouping)などのモデリング概念の集合を提供する。この意味データモデルはオブジェクト指向プログラミング言語と融合し、データ間の複雑な関係とふるまいを表現するオブジェクト指向データベース[10]に発展してきた。

一方、上記のデータモデルでは扱い難い応用であるテキストデータベースに対して、形式文法[11]に基ずくデータモデル[12]が初めて提案された。このデータモデルでは、テキスト項目値の構造は生成規則に従う。提供される代数的問合せ言語は強力であるが、文字列操作向きのアドホック仕立てである。文献(13)では、階層構造に従う情報ベースに文法モデルを応用するための形式化が行われた。その上で、等価な記述力を持つ代数的言語と書換え言語が提案された。文献(14)では、複数のデータモデルを統合するための基盤として形式文法が位置付けられている。

本論文では、形式文法を基礎とするオブジェクト指向データモデルGROOM(Grammatically Represented Object-Oriented Data Model)を提案する。GROOMは、オブジェクト指向データモデルの構造的要件である複合オブジェクト概念に、オブジェクト識別性、クラス、属性定義の継承(inheritance)・再定義(overriding)などの概念を取り入れている。オブジェクトの構造は、生成規則を使ったクラススキーマとして定義される。オブジェクトは、<i,t>の対である。ここで、iはオブジェクト識別子(object identifier)であり、オブジェクトを一意にさす。tはオブジェクト木であり、オブジェクトの情報を保持するクラススキーマ上の導出木(derivation tree)である。クラススキーマ上の上位／下位クラススキーマ関係に従い、対応するクラス間にはオブジェクト集合の包含関係が存在する。さらに、２つのクラススキーマ間に成立する条件を基に、属性定義の継承・再定義が示される。また、オブジェクトの操作のため、object-at-a-timeレベルの原始演算子から成るGROOM/PL(GROOM/Primitive Language)が提供される。

文献(12)-(14)のデータモデルは、GROOMと同じく形式文法を基礎としている。しかし、いずれのデータモデルも値指向(value-oriented)であり、コンポーネント共有が行えない点でGROOMとは異なる。これに対して、GROOMは値指向データモデルの特徴を備えると共に、オブジェクト識別子によるオブジェクトの直接参照が可能であり、このことは生成規則で表現可能となっている。

GROOVY[15]は超グラフにより形式化されたオブジェクト指向データモデルであり、超ノードの深さレベルにより階層構造を表現する。一方、GROOMにおける階層構造は、導出木におけるノード間の親子関係に基づく。また、GROOVYの多値属性[15]は組構成子と集合構成子とを合わせており、スキーマ記述が曖昧となる。これに対して、GROOMでは組構成子と集合構成子は生成規則の形式によって区別される。さらに、GROOVYでは継承関係の概念はあるが、属性の再定義の概念はない。GROOMは、属性の再定義の多様な形態を提供する。

本論文の構成は、以下の通りである。２．では、オブジェクト指向データモデルの基礎としての文法モデルの有効性を例を通して示す。３．では、形式文法を基礎とするクラススキーマとオブジェクトの定義を与える。さらに、４．ではクラススキーマ束におけるクラスおよび属性定義の継承・再定義について述べる。オブジェクトに対する操作言語GROOM/PLは５．で提示する。最後に、６．で本論文のまとめと今後の課題を示す。

# 2．文法モデル

GROOMは形式文法を基礎とするオブジェクト指向データモデルであり、以下の考えに基ずく。

(1)　オブジェクトの情報構造（スキーマ）が、文法により簡便に記述できる。

(2)　オブジェクトの情報が、この文法上の導

出木として表現できる。

以下では、EMPLOYEEオブジェクトを例に、文法モデルの有効性を示す。図1はEMPLOYEEオブジェクトの情報構造を記述した生成規則の集合を、図2は図1の生成規則を使って導出されたEMPLOYEEオブジェクトの導出木T1を示す。

(1) 組構成子：生成規則R1は、左辺のEMPLOYEE属性が右辺の属性列の集約化であることを示す。

(2) 集合構成子：生成規則R2は、左辺のSUBORDINATE属性が右辺のSUB属性の0個以上の集合であることを示す。

(3) 汎化：生成規則R4は、左辺のDEPT属性が右辺のOVERSEAS-DEPT属性とDOMESTIC-DEPT属性の汎化であることを示す。

(4) 異種構造：生成規則R5は、左辺のOVERSEAS-DEPT 属性が右辺の異種構造を持つことを示す。

(5) オブジェクト識別子：生成規則R7は、左辺のSPOUSE属性が右辺のPERSONオブジェクトを直接参照するオブジェクト識別子を値として持つことを示す。オブジェクト識別子はオブジェクト共有を可能とする。

(6) 共有値：生成規則R8は、すべてのEMPLOYEEオブジェクトのCLASS属性が"regular"を共有値として持つことを示す。

(7) 再帰的データ：生成規則R1-R3は、EMPLOYEE-SUB構造を記述する。重要な点は、SUB自身がEMPLOYEEであることを簡潔に表現できることにある。

(8) 基本ドメイン：生成規則R9-R14は、左辺の属性がそれぞれ右辺の基本ドメインの要素を値として持つことを示す。

以上の様に、生成規則がオブジェクトの情報構造に対する単純だが、強力かつ統一的な記述の枠組であることがわかる。

## 3．クラススキーマとオブジェクト

GROOMの基礎をなすデータ構造はクラススキーマであり、実世界の実体を表すオブジェクトはクラススキーマ上で定義される。クラススキーマは属性集合により構成されるが、属性は埋め込み構造[3],[4]を取りえる。クラススキーマ上のオブジェクトは、<i,t>の対である。iはオブジェクト識別子であり、その一意性はオブジェクトが実世界の実体をモデル化可能であることに通じる。また、tはオブジェクト木である。オブジェクト木はオブジェクトに対して貯えられている情報であり、実世界の実体の性質をとらえる。

以下では、形式文法を基礎としてクラススキーマとその上でのオブジェクトの定義を与える。

R1: EMPLOYEE → CLASS NAME SPOUSE SALARY SUBORDINATES DEPT
R2: SUBORDINATES → SUB *
R3: SUB → @EMPLOYEE
R4: DEPT → OVERSEAS-DEPT | DOMESTIC-DEPT
R5: OVERSEAS-DEPT → DNAME COUNTRY | DNAME COUNTRY BRANCH
R6: DOMESTIC-DEPT → DNAME CITY
R7: SPOUSE → @PERSON
R8: CLASS → "regular"
R9: NAME → String
R10: SALARY → Integer
R11: DNAME → String
R12: COUNTRY → String
R13: BRANCH → String
R14: CITY → String

図1 EMPLOYEEオブジェクトに対する生成規則の集合

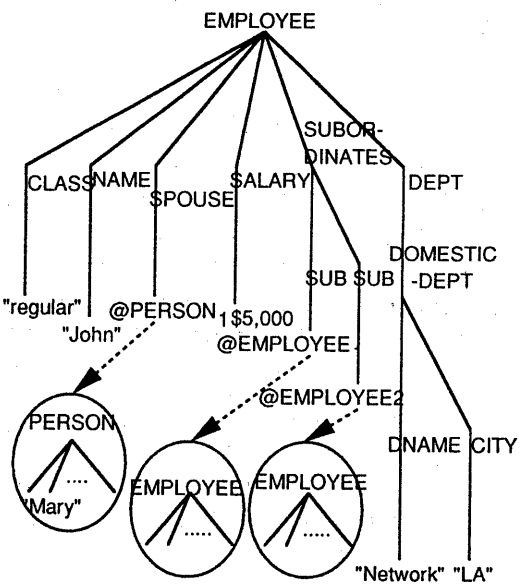


図2 EMPLOYEEオブジェクトの導出木T1

[定義1] クラススキーマCSは、<N,A,O,C,S,P>の6項組で定義される。ここで、N,A,O,Cは、それぞれ属性の有限集合、基本ドメインの有限集合、オブジェクト・ドメインの有限集合、定数の有限集合であり、これらは互いに素な集合である。S(∈N)は開始属性である。Pは生成規則の有限集合であり、すべての属性X(∈N)に対してXを左辺に持つ生成規則を1つ要素として持つ。生成規則の形式には、以下の3つがある。

(1) $X \rightarrow w_1 \mid \ldots \mid w_n$, ここで$X \in N$, $w_i \in N^*$ $(1 \leq i \leq n)$であり、各$w_i$において各属性は高々1回しか現われない。Xが開始属性Sの場合、n=1である。この形式は、属性Xが排他的な属性列$w_i$から成ることを示す。

(2) $X \rightarrow Y^*$、ここでX, Y∈Nである。Xは、Yの0回以上の繰り返しである。また、この形式の変形として、$X \rightarrow Y^+$（Yの1回以上の繰り返し）、$X \rightarrow Y_q^r$（Yのq回以上、r回以下の繰り返し）を含む。

(3) X→b、ここでX∈N, b∈(AUOUC)である。□

形式言語理論の分野で扱われる形式文法は、$G=<V_N,V_T,S,P>$の4項組で定義される。ここで、$V_N$は非終端記号の有限集合、$V_T$は終端記号の有限集合、$S(\in V_N)$は開始記号、Pは生成規則の有限集合である。クラススキーマにおいて、属性集合を$V_N$、基本ドメイン集合とオブジェクト・ドメイン集合と定数集合の和集合を$V_T$に対応付ければ、クラススキーマは基本的に文脈自由文法である。従って、文脈自由文法における既約な文法(reduced grammar)の概念が、クラススキーマにおいても定義できる。

[定義2] 既約なクラススキーマCS=<N,A,O,C,S,P>は、次の2条件を満たす。

(1) すべてのX∈Nに対して、$u, v \in N^*$が存在して、Pの生成規則を有限回適用してSをuXvに書き換えることができる。

(2) すべてのX∈(N-{S})に対して、w∈(AUOUC)*が存在して、Pの生成規則を1回以上の有限回適用してXをwに書き換えることができる。□

クラススキーマCSが既約ならば、CSのすべての属性は開始属性から到達可能であり、かつ基本ドメイン、オブジェクト・ドメイン、定数の列を生成することができる。言い換えれば、CSに無用な属性は存在しないことになる。

また、クラススキーマに対して、次の[定理1]が成立するが、その証明は文脈自由文法の場合と同様に行われる（文献(11)参照）。

[定理1] クラススキーマCSについて、CSと等価、かつ既約なクラススキーマCS'が存在する。また、CSからCS'を求めるアルゴリズムが存在する。□

次に、属性、基本ドメイン、オブジェクト・ドメイン、定数に対する定義域を示す。

[定義3] CS=<N,A,O,C,S,P>をクラススキーマとする。

(1) 基本ドメインa(∈A)に対する定義域DOM(a)はaである。DOM(a)は、未定義値を示す下限要素$\perp_a$を含む。

(2) オブジェクト・ドメインo(∈O)の定義域DOM(o)は、オブジェクト識別子の集合IDである。DOM(o)は、未定義値を示す下限要素$\perp_o$を含む。

(3) 定数c(∈C)の定義域DOM(c)は、{c}である。

(4) 属性X(∈N)の定義域DOM(X)は、Xを左辺に持つ生成規則の形式に応じて、以下の(4.1)-(4.3)に定義される。

(4.1) $X \rightarrow w_1 \mid \ldots \mid w_n (\in P)$, $w_i = X_{i1} \ldots X_{iki} (\in N^+)$, $1 \leq i \leq n$の場合、$DOM(X) = \bigcup_i (DOM(X_{i1}) \times \ldots \times DOM(X_{iki}))$である。但し、DOM(X)は、下限要素$\perp_i = <\perp_{i1}, \ldots, \perp_{iki}>$ $(1 \leq i \leq n)$を含む。

(4.2) $X \rightarrow Y^* (\in P)$の場合、$DOM(X) = 2^{DOM(Y)}$である。また、$X \rightarrow Y^+ (\in P)$の場合、$DOM(X) = 2^{DOM(Y)} - \{\phi\}$である。$X \rightarrow Y_q^r (\in P)$の場合、$DOM(X) = \{e \mid q \leq |e| \leq r, e \in 2^{DOM(Y)}\}$である。但し、DOM(X)は、下限要素$\perp_X$を含む。

(4.3) X→b(∈P), b∈(AUOUC)の場合、DOM(X)=DOM(b)である。□

オブジェクト木は、クラススキーマから生成される導出木である。以下に、その定義を示す。

[定義4] クラススキーマCS=<N,A,O,C,S,P>上のオブジェクト木Tは、以下の条件を満たす導出木である。

(1) Tの各ノードは、$(N \cup (\bigcup_i DOM(a_i)) \cup (\bigcup_j DOM(o_j)) \cup C)$の要素のいずれか一つをラベルとして持つ。但し、$a_i \in A$, $o_j \in O$である。

(2) Tの根ノードのラベルはSである。

(3) Tの内部ノード（葉でないノード）のラベルは、Nの要素に限られる。

(4) ラベルがXであるノードがk個の子ノードを持ち、かつ、それらのラベルが左から順に$X_1,...,X_k$であるとする。この場合、Pの要素であり、X(∈N)を左辺に持つ生成規則に関して、次の(4.1)-(4.3)のいずれかが成立する。

(4.1) 生成規則が$X \rightarrow w_1 \mid ... \mid w_n$, 但し$w_i \in N^+$, $n \geqq 1$である時、$w_i=X_1, ..., X_k$, $1 \leqq i \leqq n$であるようなiが存在する。

(4.2) 生成規則が$X \rightarrow Y^*$, $X \rightarrow Y^+$, $X \rightarrow Y_q^r$のいずれかであり、$X_1=...=X_k=Y$かつkが生成規則におけるYの繰り返し回数の条件を満足する。

(4.3) 生成規則が$X \rightarrow b$, $b \in (A \cup O \cup C)$である時、$k=1$かつ$X_1 \in DOM(b)$である。 □

2つのオブジェクト木が持つ情報が一致しているか否かを決定する上では、オブジェクト木の構造とノードのラベルが重要な関心事となる。このため、同形な(isomorphic)オブジェクト木を以下の様に定義する。

[定義5] 2つのオブジェクト木$T_1$, $T_2$が同形である（$T_1 \cong T_2$と記す）ための必要十分条件は、$T_1$と$T_2$のノード間にラベルと木構造を保存するような全単射関数が存在することである。 □

クラススキーマ上のオブジェクトの定義域は、以下の様に定義される。

[定義6] クラススキーマCS=<N,A,O,C,S,P>上のオブジェクトの定義域DOM(CS)は、ID×DT(CS)である。但し、DT(CS)は、クラススキーマCS上のオブジェクト木の集合である。DOM(CS)は、下限要素$<\perp_{ID}, \perp_{CS}>$を含む。 □

クラススキーマ上のインスタンスはオブジェクトの集合であり、その定義は以下の通りである。

[定義7] クラススキーマCS=<N,A,O,C,S,P>上のインスタンス$I_{CS}$はベキ集合$2^{DOM(CS)}$の要素であり、次の条件を満たす。

(1) $<i,t_1>,<i,t_2> \in I_{CS}$であれば、$t_1=t_2$である。

(2) $<i,t> \in I_{CS}$であれば、$i \neq \perp_{ID}$かつ$t \neq \perp_{CS}$である。

また、インスタンス$I_{CS}$でのオブジェクト識別子の集合$\{i \in ID \mid <i,t> \in I_{CS}\}$を$OBJ(I_{CS})$と記す。 □

クラススキーマCS=<N,A,O,C,S,P>上のインスタンス$I_{CS}$は、以下の関数$VAL_{CS}$:ID→DT(CS)を導出する。

(1) すべての$<i,t> \in I_{CS}$に対して、$VAL_{CS}(i)=t$である。

(2) $i \notin OBJ(I_{CS})$であるようなすべてのi∈IDに対して、$VAL_{CS}(i)=\perp_{CS}$である。

[例1] 以下に、図2のEMPLOYEEオブジェクトに対するクラススキーマを示す。尚、R1,...,R14は、図1の生成規則を表す。

@EMPLOYEE=<$N_1$,$A_1$,$O_1$,$C_1$,EMPLOYEE,$P_1$>

$N_1$={EMPLOYEE,SUBORDINATES,SUB,
DEPT,OVERSEAS-DEPT,
DOMESTIC-DEPT,SPOUSE,CLASS,
NAME,SALARY,DNAME,COUNTRY,
BRANCH,CITY}

$A_1$={String,Integer}

$O_1$={@PERSON,@EMPLOYEE}

$C_1$={"regular"}

$P_1$={R1, ... , R14}

また、オブジェクト<1,T1>は、クラススキーマ@EMPLOYEE上のインスタンスの要素である。ここで、T1は図2に示す導出木である。

## 4. クラスと属性定義の継承・再定義

クラススキーマ間には、上位／下位クラススキーマ関係が存在する。クラススキーマの集合はこの関係の下での半順序集合であり、クラススキーマ束(class scchceme lattice)を導出する。GROOMはクラススキーマ束において、次の概念を提供する。

(1) 属性定義の継承と再定義：下位クラススキーマは、上位クラススキーマの属性定義の継承、上位クラススキーマの属性定義を特殊化した属性の再定義を行う。

(2) クラスの包含関係：クラススキーマには、その上でのインスタンスであるクラスが対応する。上位クラススキーマに対応するクラスは、下位クラススキーマに対応するクラスを包含する集合である。

以下に、上位／下位クラススキーマ関係の定義に用いる関数ns(w,N),ws(X,N,P)を示す。但し、w,N,X,Pは、それぞれ属性列、属性集合、属性、生成規則集合を表す。

(1) 属性列wを構成する属性の集合を返す関数
$ns(w,N)=\{X_i \mid w=X_1...X_k, X_i \in N, 1 \leqq i \leqq k\}$

(2) 属性Xを左辺に持つ生成規則の右辺の排他的な各属性列を構成する属性集合族を返す関数
$ws(X,N,P)=\{ns(w_i,N) \mid X \rightarrow w_1 \mid ... \mid w_n \in P, 1 \leqq i \leqq n\}$

以下に、上位／下位クラススキーマ関係を定義

する。

[定義8] クラススキーマ$CS_1$=<$N_1$,$A_1$,$O_1$,$C_1$,$S_1$,$P_1$>がクラススキーマ$CS_2$=<$N_2$,$A_2$,$O_2$,$C_2$,$S_2$,$P_2$>の上位クラススキーマ（等価的には、$CS_2$は$CS_1$の下位クラススキーマ）であるための必要十分条件は、以下の通りである。

(1) $S_1 \neq S_2$, $S_1 \notin N_2$, $S_2 \notin N_1$

(2) $S_1 \rightarrow w_1 \in P_1$, $S_2 \rightarrow w_2 \in P_2$の時、$ns(w_1,N_1) \subseteq ns(w_2,N_2)$である。

(3) $CS_1$, $CS_2$を基にクラススキーマ$CS_3$=<$N_3$,$A_2$,$O_2$,$C_2$,$S_1$,$P_3$>を構成する。ここで、$N_3=(N_3-\{S_2\}) \cup \{S_1\}$,$P_3=(P_2-\{S_2 \rightarrow w_2\}) \cup \{S_1 \rightarrow w_1\}$である。$CS_3$に対する既約なクラススキーマを$CS_4$=<$N_4$,$A_4$,$O_4$,$C_4$,$S_4$,$P_4$>とする。この時、すべてのX($\in N_4$)に対して、Xを左辺に持つ生成規則の形式に応じて、次の(3.1)-(3.4)のいずれかが成立する。

(3.1) $X \rightarrow w_{4,1} \mid \ldots \mid w_{4,n4} \in P_4, w_{4,i} \in N_4^+, 1 \leqq i \leqq n_4$の場合、$X \rightarrow w_{1,1} \mid \ldots \mid w_{1,n1} \in P_1, w_{1,j} \in N_1^+, 1 \leqq j \leqq n_1$、かつ、(a)$ws(X,N_1,P_1)=ws(X,N_4,P_4)$であるか、(b)$ws(X,N_1,P_1) \supset ws(X,N_4,P_4)$である。

(3.2) $X \rightarrow Y^* \in P_4$, $X \rightarrow Y^+ \in P_4$, $X \rightarrow P_{q4}^{r4} \in P_4$のいずれかであり、Yの繰り返し回数が$i_4$回以上、$j_4$回以下の場合、$X \rightarrow Y^* \in P_1$, $X \rightarrow Y^+ \in P_1$, $X \rightarrow P_{q1}^{r1} \in P_1$のいずれかであり、Yの繰り返し回数が$i_1$回以上、$j_1$回以下であり、かつ、(a) $i_1=i_4$, $j_1=j_4$であるか、(b) $i_1<i_4$, $j_1 \leqq j_4$あるいは $i_1 \leqq i_4$, $j_1<j_4$である。

(3.3) $X \rightarrow o_4 \in P_4, o_4 \in O_4$の場合、$X \rightarrow o_1 \in P_1, o_1 \in O_1$であり、かつ、(a) $o_1$, $o_4$に対応するクラススキーマが同一であるか、(b) $o_1$に対応するクラススキーマが$o_4$に対応するクラススキーマの上位クラススキーマである。

(3.4) $X \rightarrow b_4 \in P_4, b_4 \in (A_4 \cup C_4)$の場合、$X \rightarrow b_1 \in P_1, b_1 \in (A_1 \cup C_1)$、かつ、(a)$DOM(b_1)=DOM(b_4)$であるか、(b) $DOM(b_1) \supset DOM(b_4)$である。

□

[定義8]の(3.1)-(3.4)の各(a)の場合が、クラススキーマ$CS_1$から$CS_2$に対する属性定義の継承の場合である。また、各(b)の場合が$CS_1$の属性定義を特殊化して$CS_2$で属性の再定義を行う場合である。この属性の再定義に関して、(a)は属性Xを構成する代替的な属性列を制限する場合であり、構造特殊化(structure specialization)と呼ぶ。(b)は属性Xを構成する属性Yの繰り返し回数を制限する場合であり、繰り返し特殊化(repeating specialization)と呼ぶ。(c)は属性Xのオブジェクト識別子の集合を制限する場合であり、定義域特殊化(domain specialization)と呼ぶ。(d)は属性Xの値の集合を制限する場合であり、値特殊化(value specialization)と呼ぶ。表1は、クラススキーマ間での属性の再定義の例である。また、図3はクラススキーマ間属性集合上の関係を示す。

$CS_1$(=<$N_1$,$A_1$,$O_1$,$C_1$,$S_1$,$P_1$>)
IS A SUPER CLASS SCHEMA OF
$CS_2$(=<$N_2$,$A_2$,$O_2$,$C_2$,$S_2$,$P_2$>).

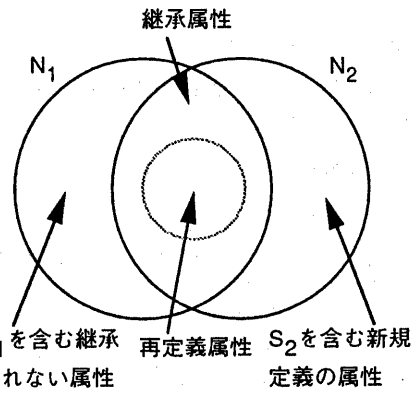


図3 クラススキーマの属性集合間の関係

クラススキーマ束におけるデータベースの定義に用いる関数を以下に示す。但し、OBJ-Tはオブジェクト木、L,$L_1$, ... ,$L_k$はラベルを表す。

(1) project(OBJ-T, L→$L_1$ ... $L_k$)：ラベルLを持つOBJ-Tのノードに関して、その子ノードが左から順にラベル$L_1$,... ,$L_k$を持つノードとなるオブジェクト木を返す。

(2) rename(OBJ-T, $L_1$, $L_2$)：OBJ-Tに対して、ラベル$L_1$のノードのラベルを$L_2$に変更したオブジェクト木を返す。

以下に、クラススキーマ束におけるデータベースを定義する。

[定義9] クラススキーマ束CSL上のデータベー

表1　クラススキーマ間での属性の再定義(overriding)の例

| 種類 | 構造特殊化 (structure specialization) | 繰り返し特殊化 (repeating specialization) | 定義域特殊化 (domain specialization) | 値特殊化 (value specialization) |
|---|---|---|---|---|
| $CS_1$ | MAJOR→<br>MATHEMATICS \|<br>PHYSICS \|<br>ELECTRONICS \|<br>MECHANICS \|<br>PSYCHOLOGY \|<br>..... $\in P_1$ | REQUIRED-SUBJECT→<br>SUBJECT* $\in P_1$ | SUBJECT→<br>@SPECIALIZED-SUBJECT<br>$\in P_1$ | DEPARTMENT→<br>String$\in P_1$ |
| $CS_2$ | MAJOR→<br>ELECTRONICS \|<br>MECHANICS$\in P_2$ | REQUIRED-SUBJECT→<br>$SUBJECT_{10}^{20} \in P_2$ | SUBJECT→<br>@ENGINEERING-SUBJECT$\in P_2$ | DEPARTMENT→<br>"Engineering"$\in P_2$ |
| 再定義の内容 | $CS_2$では、専攻が工学部の対象に制限されている. | $CS_2$では、必修科目数が具体的な数の範囲に制限されている. | $CS_2$では、科目が専門科目から工学系専門科目に制限されている. | $CS_2$では、全オブジェクトの学部名が”工学部”である. |

$CS_1$:@STUDENT=<$N_1$,$A_1$,$O_1$,$C_1$,$S_1$,$P_1$>

$CS_2$:@ENGINEERING-STUDENT=<$N_2$,$A_2$,$O_2$,$C_2$,$S_2$,$P_2$>

$CS_1$ IS A SUPER CLASS SCHEMA OF $CS_2$.

ス集合DBは、次の2条件を満たすクラス集合である。

(1)　すべてのクラススキーマCS(∈CSL)に対して、クラス$I_{CS}$(∈DB)が1つ対応する。

(2)　$CS_2$(=<$N_2$,$A_2$,$O_2$,$C_2$,$S_2$,$P_2$>)が$CS_1$(=<$N_1$,$A_1$,$O_1$,$C_1$,$S_1$,$P_1$>)の下位クラススキーマであるような、すべてのクラススキーマ$CS_1$,$CS_2$(∈CSL)と対応するクラス$I_{CS1}$,$I_{CS2}$(∈DB)に対して、以下の(2.1)、(2.2)が成立する。

(2.1)　$OBJ(I_{CS2}) \subseteq OBJ(I_{CS1})$

(2.2)　すべてのオブジェクトo=<i,t>(∈$I_{CS2}$)に対して、rename(project(t,$S_2 \rightarrow w_1$),$S_2$,$S_1$)$\cong$ $VAL_{CS1}$(i)である。但し、$S_1 \rightarrow w_1 \in P_1$である。□

[定義9]の(2.1)は、$I_{CS2}$が$I_{CS1}$の下位クラスであることを保証する。また、(2.2)は、オブジェクトo(∈$I_{CS2}$)が同じオブジェクト識別子を持つ持つ$I_{CS1}$に属するオブジェクトから性質を継承することを保証する。

[例2]　次の3つのクラススキーマを考える。

@EMPLOYEE=<$N_1$,$A_1$,$O_1$,$C_1$,EMPLOYEE,$P_1$>

([例1]に定義したクラススキーマ)

@MANAGER=<($N_1$-{EMPLOYEE})∪
{MANAGER},$A_1$,$O_1$,$C_1$,MANAGER,($P_1$-{R1})
∪{MANAGER→CLASS NAME SPOUSE SALARY SUBORDINATES DEPT RANK, RANK→String}>

@ENGINERR=<($N_1$-{EMPLOYEE})∪
{ENGINEER},$A_1$,$O_1$,$C_1$,ENGINEER,($P_1$-{R1})
∪{ENGINEER→CLASS NAME SPOUSE SALARY SUBORDINATES DEPT SPECIALITY,SPECIALITY→String}>

@MANAGERと@ENGINEERは@EMPLOYEEの下位クラススキーマであり、@EMPLOYEEの属性定義を継承している。<1,T1>, <1,T2>は、各々、@EMPLOYEEと@MANAGERのインスタンスの要素である。但し、T1, T2は、それぞれ図2、図4に示す導出木である。併せて、<1,T2>は@EMPLOYEEのインスタンスの要素でもあり、以下の様に[定義9]の(2.2)の条件が成立している。

rename(project(T2,MANAGER→CLASS NAME SPOUSE SALARY SUBORDINATES DEPT), MANAGER,EMPLOYEE)
$\cong VAL_{@EMPLOYEE}(1)$

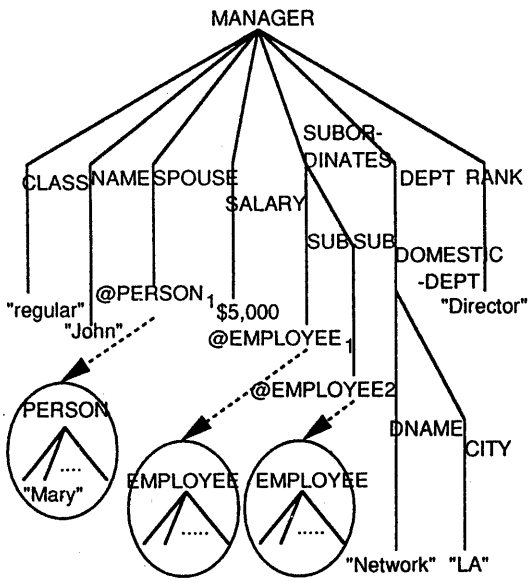


図4　MANAGERオブジェクトの導出木T2

さらに、クラススキーマ関での多重継承(multiple inheritance)の例を示す。以下のクラススキーマ@EMP-MGRは@MANAGERと@ENGINEERの下位クラススキーマであり、@MANAGERと@ENGINEERの属性定義を継承する。

@ENG-MGR=<($N_1$-{EMPLOYEE})∪
{ENG-MGR},$A_1$,$O_1$,$C_1$,ENG-MGR,($P_1$-{R1})∪{
ENG-MGR→CLASS NAME SPOUSE SALARY
SUBORDINATES DEPT RANK SPECIALITY,
RANK→String,SPECIALITY→String}>

## 5．オブジェクト操作言語

ここでは、オブジェクトに対する問合せ・更新を行う操作言語GROOM/PLを提案する。GROOM/PLは、1つのオブジェクトあるいは1つのオブジェクト木を操作する原始演算子の集合から成る。GROOMでは、より高水準な操作言語を提供する予定であるが、GROOM/PLはこれらの上位の言語群を実現するための下位層としての機能を担う。

GROOM/PLでは、パラメータとしてオブジェクト木のノードを指すノードアドレス(node address)を必要とする演算子がある。このため、まずオブジェクト木に対するユニーク木を定義し、その下でノードアドレスを定義する。

[定義10]　オブジェクト木tに対するユニーク木unique(t)は、以下を満足する。

(1)　unique(t)は、tと同じ構造を持つ。

(2)　unique(t)のノードのラベルは、次の(2.1),(2.2)により規定される。

(2.1)　tのノードnのラベルlが兄弟ノードのラベルの中で重複する場合、ノードnに対応するunique(t)のノードのラベルを$l_i$とする。但し、iは、nはラベルが重複する兄弟ノードの左からi番目のノードであることを示す。

(2.2)　tのノードnのラベルlが兄弟ノードのラベルの中でユニークである場合、ノードnに対応するunique(t) のノードのラベルをlとする。　□

次に、オブジェクト木のノードを指定するノードアドレスを定義する。

[定義11]　オブジェクト木tのノードnを指定するノードアドレスは、unique(t)における根ノードからノードnに対応するノードに至るパス上のノードのラベル系列である。　□

GROOM/PLの演算子は、オブジェクトの問合せ・更新、オブジェクト木の問合せ・更新に分類される。オブジェクトに対する演算子とオブジェクト木に対する演算子とを分けた理由は、GROOM/PLレベルでは更新処理の過程において一時的にクラススキーマ記述に合わないオブジェクト木が現われることを許すためである。これは、トランザクション処理の途中では矛盾したデータベース状態(inconsistent database state)の出現を許すことに考え方として同じである。

### 5．1　オブジェクトの問合せ

(1)　get-object-tree(CNAME,OID)

クラス名CNAMEを持つクラスからオブジェクト識別子OIDが参照するオブジェクト木を返す。

(2)　value-to-object(CNAME,COND)

クラス名CNAMEを持つクラスから条件CONDを満足するオブジェクトの集合を求め、この集合の要素を順次走査するスキャンの識別子を返す。ここで、CONDは2項条件または単項条件の論理積リストである。2項条件は＜ノードアドレス、関係演算子、値＞で表され、ノードアドレスが指すノードが基本ドメインの要素をラベルとして持ち、かつ、その子ノードのラベルと値との間の関係演算子で示す条件を表す。関係演算子は、"=","≠","<",">","≦","≧"を含む。単項条件は＜ノードアドレス、単項演算子＞で表され、ノードドレスが指すノードの存在に関する条件を示す。単項演算子は"existent","nonexistent"を含み、それぞれノードの存在、非存在を表す。

(3) get-obect-identifier(SID)
スキャン識別子SIDが示すスキャンが指すオブジェクトのオブジェクト識別子を返し、スキャンが次のオブジェクトを指すように進める。

5．2 オブジェクトの更新
(1) create-object(CNAME)
クラス名CNAMEを持つクラスに属するオブジェクトを生成し、そのオブジェクト識別子を返す。オブジェクト識別子が参照するオブジェクト木は、根ノードのみから成る。
(2) drop-object(CNAME,OID)
クラス名CNAMEを持つクラスからオブジェクト識別子OIDを持つオブジェクトを削除する。
(3) change-object(CNAME,OID,NEW-OBJT)
クラス名CNAMEを持つクラスに属するオブジェクト識別子OIDを持つオブジェクトのオブジェクト木をNEW-OBJTに更新する。この時、オブジェクト木NEW-OBJTがクラススキーマ定義に合うか否かがチェックされる。

5．3 オブジェクト木の問合せ
(1) get-children-label(OBJT,NADDR)
オブジェクト木OBJTにおいてノードアドレスNADDRが指すノードの子ノードのラベルのリストを返す。
(2) get-children-no(OBJT,NADDR)
オブジェクト木OBJTにおいてノードアドレスNADDRが指すノードの子ノードの数を返す。

5．4 オブジェクト木の更新
(1) insert-node(OBJT,NADDR,CHD-LBL)
オブジェクト木OBJTにおいてノードアドレスNADDRが指すノードの子ノードとしてラベル・リストCHD-LBLの各ラベルを持つノードを挿入する。ノードの挿入箇所は、既に存在している最も右の子ノードの右の位置となる。
(2) delete-node(OBJT,NADDR)
オブジェクト木OBJTにおいてノードアドレスNADDRが指すノードを根ノードとする部分木を削除する。
(3) change-label(OBJT,NADDR,NEW-LBL)
オブジェクト木OBJTにおいてノードアドレスNADDRが指すノードのラベルをNEW-LBLに変更する。
(4) move-subtree(OBJT,NADDR1,NADDR2)
オブジェクト木OBJTにおいてノードアドレスNADDR1が指すノードの最も右の子ノードとなるようにノードアドレスNADDR2が指すノードを根ノードとする部分木を移動する。
(5) copy-subtree(OBJT,NADDR1,NADDR2)
オブジェクト木OBJTにおいてノードアドレスNADDR1が指すノードの最も右の子ノードとなるようにノードアドレスNADDR2が指すノードを根ノードとする部分木を複製する。
(6) reorder-subtree(OBJT,NADDR,
NEW-ORDER)
オブジェクト木OBJTにおいてノードアドレスNADDRが指すノードの子ノードの並びをラベルの順序が左からNEW-ORDERとなるように変える。NEW-ORDERの各ラベルは子ノードのラベルであるが、子ノードのラベルがNEW-ORDERのラベルである必要はない。
図5に、オブジェクト木に対する更新演算子の系列の適用例を示す。

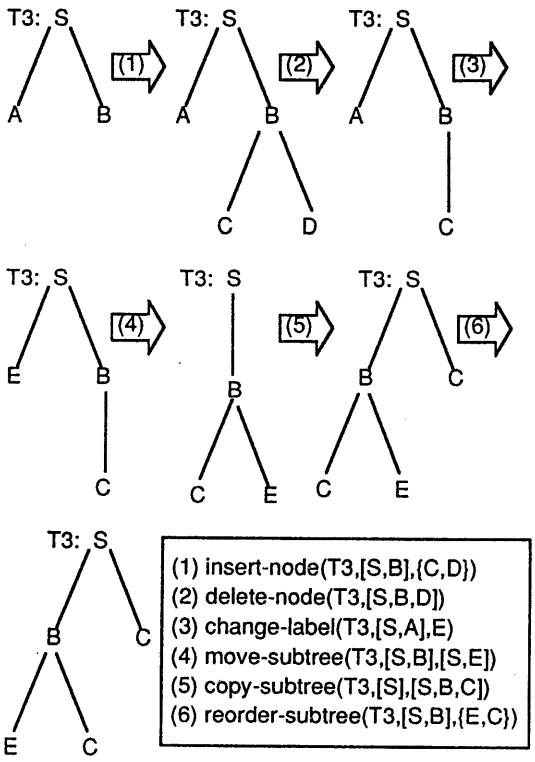


図5 オブジェクト木の更新例

## 6．おわりに

本論文では、形式文法を基礎とするオブジェクト指向データモデルGROOMを提案した。GROOMは値指向のデータモデルの特徴を備えると共に、オブジェクト指向データモデルの構造的要件（複合オブジェクト、オブジェクト識別性、クラス、継承・再定義など）を満足する。オブジェクト木は木構造が本来持つ構成関係に加えて、オブジェクト・ドメインの導入によりオブジェクト木間の関係も表現できるように拡張されている。この点が、GROOMをオブジェクトの情報構造に対する単純だが、強力かつ統一的な枠組としている。

今後の課題は、(1)GROOMにメソッド概念を組み込むこと、(2)関数従属性などの制約記述の検討、(3)高水準問合せ言語の設計とその処理系の実装、(4)ビュー・導出データの検討などが挙げられる。

## 参考文献


(1) Codd E.F. : "A Relational Model of Data for Large Shared Data Banks", Commun. ACM, 13, 2, pp.377-387 (1970-02)

(2) 田中克己 : "オブジェクト指向データベースの基礎概念", 情報処理, 32, 5, pp.500-513 (1991-05)

(3) Dadam P. et al. : "A DBMS Prototype to Support Extended NF2 Relations", Proc. SIGMOD Conf., pp.356-364 (1986)

(4) Roth M.A., Korth H.F., Silberschatz A. : "Extended Algebra and Calculus for Nested Relational Databases", ACM Trans. Database Syst., 13, pp.389-417 (1988)

(5) Chen P. : "The Entity-Relationship Model: Toward a Unified View of Data", ACM Trans. Database Syst., 1, pp.9-36 (1976)

(6) Shipman D. : "The Functional Data Model and the Data Language DAPLEX", ACM Trans. Database Syst., 6, pp.140-173 (1981)

(7) Hammer M., Mcleod D. : "Database Description with SDM:A Semantic Database Model", ACM Trans. Database Syst., 6, pp.351-386 (1981)

(8) Hull R., Yap C. : "The Format Model:A Theory of Database Organization", J.ACM, 31, pp.518-537 (1984)

(9) Abiteboul S., Hull R. : "IFO:A Formal Semantic Database Model", ACM Trans. Database Syst., 12, pp.525-565 (1987)

(10) Atokinson M., Bancilhon F., Dewit D., Dittrich K., Maier D., Zdonik S. : "The Object-Oriented Database System Manifesto", Proc. DOOD89, pp.40-57 (1989)

(11) 福村晃夫, 稲垣康善 : "オートマトン・形式言語理論と計算論",岩波講座・情報科学6, 岩波書店 (1982)

(12) Gonnet G.H., Tompa F.W. : "Mind Your Grammar:a New Approach to Modelling Text", Proc. VLDB Conf., pp.339-346 (1987)

(13) Gyssens M., Paredaens J., Gucht D.V. : "A Grammar-Based Approach Towards Unifying Hierarchical Data Models", Proc. VLDB Conf., pp.263-272 (1989)

(14) Colby L.S., Gucht D.V. : "A Grammar Model for Databases", Techn. Rep. No.282, Comput. Sci. Dept., Indiana Univ. (1989)

(15) Levene M., Poulovassilis A. : "An Object-Oriented Data Model Formalised Through Hypergraphs", Data & Knowledge Eng., 6, pp.205-224 (1991)