

オブジェクト指向データモデルのメソッドに対する 属性文法による形式化

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

GROOM-IIはオブジェクト指向データモデルGROOM(Grammatically Represented Object-Oriented Data Model)の拡張版であり、新たにメソッド概念が導入されている。GROOM-IIでは、メソッド仕様は属性文法により記述される。メッセージは属性文法の解析対象文であり、メソッド実行はメッセージに対する文解析プロセスとなる。上位/下位クラススキーマ関係を通じたメソッド定義の継承(inheritance)・再定義(overriding)は、メソッド定義の多形性(polymorphism)と同様に、対応する属性文法間での文の生成能力と変換能力とに関する条件によって定義される。

Formalization of the Method Concept in the Object-Oriented Data Model using Attribute Grammars

Hideki Sato * Nobuhiro Ito ** Tatsuya Hayashi **

*NIPPONDENSO CO., LTD.

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

GROOM-II, an extension of GROOM(Grammatically Represented Object-Oriented Data Model), newly incorporates the method concept into its modeling framework. In GROOM-II, method specifications are described using attribute grammars. A message is taken as a sentence to be parsed using an attribute grammar and a method execution is taken as a sentence parsing process. Inheritance and overriding of method definitions through super/sub class schema relationships, as well as polymorphism of method definitions, are specified based on the condition concerning sentence generation abilities and transformation abilities that the corresponding attribute grammars have.

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の限界・問題点(2)を克服するため、種々の高水準データモデルが提案されている。非正規関係データモデル(3),(4)、意味データモデル(5),(6)などの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モデル(7)に発展してきた。一方、テキストデータベースに対して形式文法(8)が初めて使われた(9)のを皮切りに、階層的な情報構造を備えたデータモデルに対して文法モデルが応用されてきた(10),(11)。

筆者らは、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)の提案を行った(12)。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、オブジェクトの情報はクラススキーマ上のオブジェクト木(導出木)として表現される。

GROOMは形式文法を基に、オブジェクトの情報構造に対する単純ではあるが、強力かつ統一的な記述の枠組を提供する。しかし、それが扱う範囲は、オブジェクト指向データモデルの構造的側面に限定されていた。本論文では、GROOMを拡張し、オブジェクト指向データモデルのふるまいの側面を取り入れたGROOM-IIを提案する。GROOM-IIでは、オブジェクトの構造を定義するクラススキーマにメソッド定義を導入する。メソッド仕様は、属性文法(attribute grammar)(13),(14)により記述される。メッセージは属性文法の解析対象となる文(sentence)であり、メソッド実行はメッセージに対する文解析プロセスとなる。

上位/下位クラススキーマ関係を通したメソッド定義の継承・再定義は、対応する属性文法間での文の生成能力と変換能力に関する条件によって定義される。同様に、メソッド定義の多形性(polymorphism)も、対応する属性文法間での文の生成能力と変換能力に関する条件によって定義される。

文献(9)-(11)のデータモデルは、GROOM-IIと同じ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、

かつ、メソッド概念のようなふるまい的側面を扱うことはない。GROOM(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、オブジェクト指向データモデルの構造的側面を扱っている。しかし、このデータモデルもメソッド概念を取り入れてはいない。文献(16),(17)は、オブジェクト指向パラダイムを使って属性文法の能力を拡張している。これに対して、GROOM-IIはオブジェクト指向データモデルのメソッド概念を属性文法により形式化するものであり、前者とは目的を異にしている。

本論文の構成は、以下の通りである。2. では、本論文で用いる属性文法とその能力について述べる。3. では、属性文法の応用によるメソッド表現を例を通して示す。4. では、GROOM-IIのクラススキーマとオブジェクト木を定義する。さらに、5. ではメソッド間での定義の継承・再定義および多形性について、6. ではメソッド実行と意味規則記述について述べる。最後に、7. で本論文のまとめと今後の課題を示す。

2. 属性文法

属性文法は、これまでに主にコンパイラ(18)、エディタ(19)などに応用されてきた言語処理のための形式的体系である。その基本原理の一つは、木構造上の属性値の純関数的な計算モデルにある。しかし、本論文では属性文法のデータベースへの応用を目的としており、このために属性文法に対する拡張が必要となってくる。以下に、本論文で用いる属性文法の定義を示す。

[定義1] 属性文法AGは、以下を満たす3項組 $\langle G, Y, F \rangle$ で定義される。

(1) $G = \langle V_N, V_T, S', P' \rangle$ は既約な文脈自由文法であり、AGの基底文脈自由文法と呼ばれる。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である。 $S' (\in V_N)$ は、開始記号である。 P' は、生成規則の有限集合である。

(2) G の非終端記号 $X (\in V_N)$ に対して、互いに素な2つの有限集合 $I-ATTR(X)$ と $S-ATTR(X)$ が付随している。 $I-ATTR(X)$ の要素を継承属性、 $S-ATTR(X)$ の要素を合成属性と呼ぶ。 X の属性 $\star$ 集

☆属性文法における属性概念とクラススキーマにおける属性概念とを区別するため、以降では前者をg-属性、後者をc-属性と記す。但し、属性文法における継承属性と合成属性、クラススキーマの開始属性はこの限りではない。

合の全体をATTR(X)で表す。すなわち、 $\text{ATTR}(X) = \text{I-ATTR}(X) \cup \text{S-ATTR}(X)$ である。AGのg-属性集合をYと表す。すなわち、 $Y = \bigcup_{X \in V_N} \text{ATTR}(X)$ である。また、Xのg-属性aをa(X)、g-属性aが取りうる値の集合をV(a)と表す。

(3) P'の生成規則 $p: X_0 \rightarrow w_0 X_1 w_1 \dots X_n w_n (n \geq 0, X_i \in V_N, w_i \in V_T^*, 0 \leq i \leq n)$ に対して、以下の(3.1), (3.2)の形式をとる意味規則の集合 F_p が付随している。但し、DBはデータベースの状態を表す。

(3.1) $\text{S-ATTR}(X_0) \cup \text{I-ATTR}(X_1) \cup \dots \cup \text{I-ATTR}(X_n)$ に含まれるg-属性a(X_k) ($0 \leq k \leq n$)を定義する意味規則は、次の形式をしている。

$a(X_k) := f(a_1(X_{i_1}), \dots, a_m(X_{i_m}))$ with DB
但し、fは $a_1(X_{i_1}), \dots, a_m(X_{i_m})$ を明示的なパラメータ、DBを暗黙的なパラメータとする関数である。

(3.2) データベース状態の更新を定義する意味規則は、次の形式をしている。

$\text{DB} \leftarrow g(a_1(X_{i_1}), \dots, a_n(X_{i_n}))$ with DB
但し、gは $a_1(X_{i_1}), \dots, a_n(X_{i_n})$ を明示的なパラメータ、DBを暗黙的なパラメータとする関数である。関数の計算結果は、データベースに対する更新として反映される。

$F = \bigcup_{p \in P'} F_p$ と定義し、FをAGの意味規則集合と呼ぶ。□

以降では、Gの開始記号S'のg-属性集合に関して、 $\text{I-ATTR}(S') = \emptyset$ 、 $|\text{S-ATTR}(S')| \leq 1$ が成立することを仮定する。すなわち、S'は継承属性を持たず、合成属性を高々1つだけ持つ。また、AGは任意の導出木において、その各ノードに付随したg-属性の値が循環定義に陥らない非循環属性文法に制限する。

次に、属性文法の生成能力と変換能力に関する定義を示す。

[定義2] 属性文法 $\text{AG} = \langle G, Y, F \rangle$ に関して、

(1) 生成能力：AGの基底文脈自由文法 $G = \langle V_N, V_T, S', P' \rangle$ によって生成される言語L(G)は、P'の生成規則を0回以上の有限回適用してS'から書き換えることができる $w (\in V_T^*)$ の集合で定義される。

(2) 変換能力：tをAGの任意の導出木とする。 $m1_{\text{AG}}(t, d)$ は、tの根ノードに合成属性が付随する場合には、データベース状態dの下でのtの根ノードの（唯一の）合成属性の値を表す。そうでなければ、 $m1_{\text{AG}}(t, d)$ は未定義値 $\perp$ を表す。また、 $m2_{\text{AG}}(t, d)$ はデータベース状態dの下でのtによるデータベースの更新状態を表す。もしtがデータベースの更新を行わなければ、 $m2_{\text{AG}}(t, d)$ はdを表す。この下で、AGによる変換は以下に定義される。但し、DBSはデータベース状態の集合を表す。

$T(\text{AG}) = \{ \langle w, d_1, v, d_2 \rangle \mid w \text{ is a string of terminal symbols derived by } t, d_1 \in \text{DBS}, v = m1_{\text{AG}}(t, d_1), d_2 = m2_{\text{AG}}(t, d_1) \in \text{DBS} \}$ □

[定義2]の属性文法による変換は、図1に分類される。ここで、TD(AG)は、データベースの更新を伴わない変換の集合である。TV(AG)は導出木の根ノードに付随する合成属性の値が未定義である変換の集合である。また、TS(d, AG)は、データベース状態dの下での導出木に対する終端記号列と根ノードに付随する合成属性の値の2項組の集合である。

以下に、2つの属性文法間の能力の関係を定義する。

[定義3] $\text{AG}_1 = \langle G_1, Y_1, F_1 \rangle$, $\text{AG}_2 = \langle G_2, Y_2, F_2 \rangle$ を属性文法であるとして、

(1) $L(G_1) = L(G_2)$ かつ $T(\text{AG}_1) = T(\text{AG}_2)$ であれば、 AG_1 と AG_2 は同じ能力を持つ。

(2) $L(G_1) \supseteq L(G_2)$ かつ $T(\text{AG}_1) \supset T(\text{AG}_2)$ であれば、 AG_1 は AG_2 よりも強い能力を持つ。 □

属性文法間の能力の関係は、以降でクラススキーマ間でのメソッド定義の継承・再定義および多形性の定義に用いられる。

3. 属性文法によるメソッド表現

オブジェクト指向データモデルにおけるメソッドのインタフェース仕様は、セレクトーとパラメータリストにより示される。例えば、今年の年齢を求めるメソッドのインタフェース仕様は、what-age()となる。このメソッドのインプリメンテーションは、オブジェクトのBIRTHDAY属性の値と現在の年とからオブジェクトの年齢を計算する。しかし、過去あるいは未来のある年におけるオブジェクトの年齢が知りたい時、このメソッドは使えない。このため、新たにメソッドwhat-age1(YEAR)を考えることになる。このメソッドは、オブジェクトのBIRTHDAY属性の値とパラメータYEARの値とから、パラメータYEARで指定

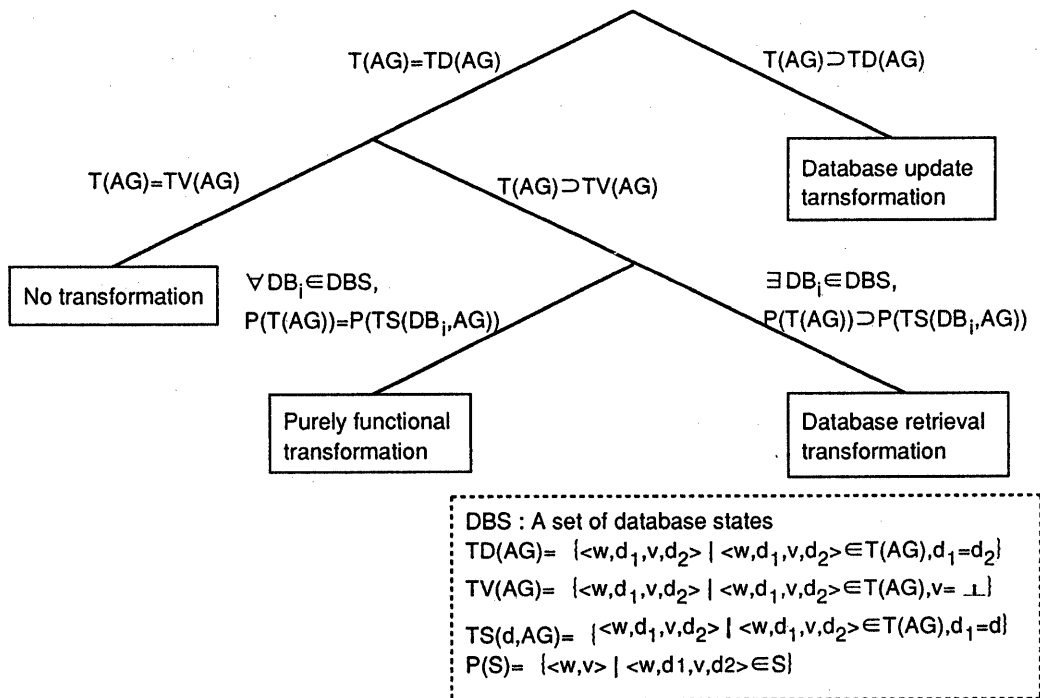


図1 属性文法AGによる変換の分類

された年におけるオブジェクトの年齢を計算する。

以上より、メソッドの仕様記述上の問題点は、次のようになる。

(1) 類似の機能要求に対して、異なるインタフェース仕様を持つメソッドが必要となる(例、what-age()とwhat-age1(YEAR))。

(2) 類似の機能要求を単一のメソッドのインタフェース仕様により扱う場合、パラメータの値を工夫して機能の使い分けを行うことになり

(例、what-age1(YEAR)のパラメータYEARの値が"don't care"の時、現在の年を表すことにする)、インタフェース仕様として不明確となる。

さらに、次の様な問題点も挙げられる。

(3) メソッドのインプリメンテーションは手続き的に与られるので、その仕様の作成・維持ならびに理解が困難である。

GROOM-IIでは、上記の問題点を解決するため、属性文法を使って、メソッドの形式化を行う。メソッドのインタフェース仕様は、属性文法による構文解析の対象となる文集合に対する基底文脈自由文法文法として与えられる。この点から、セレクターとパラメータといった区別は陽には現れない。また、類似の機能要求に対しては、

表1 属性文法とメソッド概念との対応

属性文法	メソッド概念
属性文法記述	メソッド仕様
基底文脈自由文法	インタフェース仕様 (セレクター、パラメータ)
g-属性集合、 意味規則集合	インプリメンテーション仕様 (メソッド本体)
文	メッセージ
文解析	メソッド実行
文解析結果 (g-属性計算結果)	メソッド実行結果

対応する文集合を包含する基底文脈自由文法を考えればよく、上記の問題点(1),(2)は生じない。問題点(3)に関しては、属性文法におけるg-属性集合/意味規則集合を使ったメソッドのインプリメンテーション仕様の形式化により解決を図る。表1に、属性文法とメソッド概念との対応を示す。

表2 非終端記号に付随するg-属性 (一部)

非終端記号	合成属性	継承属性
IS	age	—
NP	age	year
PN	obj	—
N1	age	year, birthday
AP	year	—
N2	year	—

表3 属性文法の規則 (一部)

構文規則	意味規則
IS → IP BV NP QM	age(IS)=age(NP) year(NP)=thisyear()
IS → IP BV NP AP QM	age(IS)=age(NP) year(NP)=year(AP)
NP → PN N1	age(NP)=age(N1) year(N1)=year(NP) birthday(N1)= obj(PN).birthday
PN → "year"	obj(PN)=self()
N1 → "age"	age(N1)=year(N1)- birthday(N1)
AP → PP N2	year(AP)=year(N2)
N2 → ... "1975" ...	year(N2)=value()

(注) a(N):非終端記号Nのg-属性a
obj.attr:オブジェクトobjのg-属性attr
self():メッセージのレシーバ・オブジェクト
の識別子を返す関数
value():終端記号の値を返す関数
thisyear():現在の年を返す関数

属性文法によるメソッド仕様の記述例として、
前記のwhta-gae()とwhat-age1(Year)との年齢を
計算するメソッドに対する属性文法を考える。表
2は、メッセージの文集合を規定する文脈自由文

法の非終端記号とそれに付随する合成属性、継承
属性の組の一部を示す。また、表3はメッセージ
の文集合に対応する意味規則の集合の組の一部を
示す。表2と表3における非終端記号ISは、開始
記号である。メッセージの戻り値は、属性計算の
結果として、開始記号ISに付随する合成属性
ageの値として与えられる。図2は、メッセージ
に対する構文木上でのg-属性計算の様子を示す。
図2の(a)はwhat-age()の呼び出しに対応してい
り、現在のオブジェクトの年齢を計算する。ま
た、図2の(b)はwhat-age1(1969)の呼び出しに
対応しており、1969年におけるオブジェクトの年
齢を計算する。

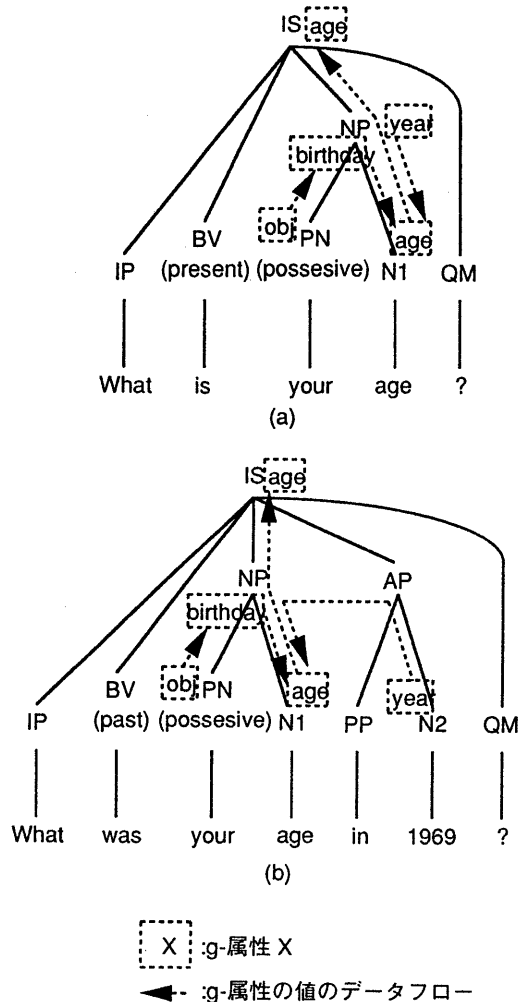


図2 メッセージに対する構文木上での属性計算

4. クラススキーマとオブジェクト木

クラススキーマは、オブジェクトの定義を与えるデータ構造である。GROOMのクラススキーマ定義は文献(12)に示されているが、そこではメソッド概念は含まれていない。以下では、GROOMのクラススキーマ定義を拡張し、メソッド概念を取り込んだGROOM-IIのクラススキーマを定義する。

[定義4] クラススキーマCSは、 $\langle N, A, O, C, M, S, P \rangle$ の7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。 M は、 $\langle MA, AG \rangle$ の有限集合である。 MA はメソッド・ドメインであり、 AG は MA に関連付けられた属性文法である。また、 $mas(M)$ は、 M におけるメソッド・ドメインの集合を表す。すなわち、 $mas(M) = \{MA \mid \langle MA, AG \rangle \in M\}$ である。さらに、 $S(\in N)$ 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 c -属性 $X(\in N)$ 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) $X \rightarrow w_1 \mid \dots \mid w_n$, 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 c -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、 c -属性 X が排他的な c -属性列 w_i から成ることを示す。

(2) $X \rightarrow Y^*$, ここで $X, Y \in N$ である。 X は、 Y の0回以上の繰り返しである。また、この形式の变形として、 $X \rightarrow Y^+$ (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) $X \rightarrow b$, ここで $X \in N, b \in (AUOUUCU mas(M))$ である。□

クラススキーマにおける c -属性、基本ドメイン、オブジェクト・ドメイン、定数、メソッド・ドメインの定義域は、以下の通りである。

[定義5] $CS = \langle N, A, O, C, M, S, P \rangle$ をクラススキーマとする。

(1) 基本ドメイン $a(\in A)$ 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 $o(\in O)$ 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。

む。

(3) 定数 $c(\in C)$ の定義域 $DOM(c)$ は、 $\{c\}$ である。

(4) メソッド・ドメイン $m(\in mas(M))$ の定義域 $DOM(m)$ は $\langle m, ag \rangle \in M, ag = \langle G, Y, F \rangle, G = \langle V_N, V_T, S', P' \rangle$ の時、 $S-ATTR(S') = \{y\}$ であれば $V(y)$ である。 $S-ATTR(S') = \emptyset$ であれば、 $DOM(m)$ は $\{\perp_m\}$ である。ここで、 $\perp_m$ は未定義値を示す下限要素である。

(5) c -属性 $X(\in N)$ 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(5.1)-(5.4)に定義される。

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

(5.2) $X \rightarrow Y^* (\in P)$ の場合、 $DOM(X) = 2^{DOM(Y)}$ である。また、 $X \rightarrow Y^+ (\in P)$ の場合、 $DOM(X) = 2^{DOM(Y)} - \{\emptyset\}$ 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。

(5.3) $X \rightarrow b (\in P), b \in (AUOUUCU mas(M))$ 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り、オブジェクトの情報を保持する。

[定義6] クラススキーマ $CS = \langle N, A, O, C, M, S, P \rangle$ 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、 $(NU(U DOM(a_i)) U(U DOM(o_j)) UCU mas(M))$ 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 $X_1, \dots, X_k$ であるとする。この場合、 P の要素であり、 $X(\in N)$ を左辺に持つ生成規則に関して、次の(4.1)-(4.4)のいずれかが成立する。

(4.1) 生成規則が $X \rightarrow w_1 \mid \dots \mid w_n$, 但し $w_i \in N^+, (n \geq 1)$ である時、 $w_i = X_1 \dots X_k, 1 \leq i \leq n$ 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、 $X_1 = \dots = X_k = Y$ 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 $X \rightarrow b$, $b \in (AUUC)$ である時、 $k=1$ かつ $X_1 \in \text{DOM}(b)$ である。

(4.4) 生成規則が $X \rightarrow m$, $m \in \text{mas}(M)$ である時、 $k=1$ かつ $X_1 = m$ である。□

[定義6]により、オブジェクト木はメソッド概念を取り入れるように拡張された。オブジェクト木の同形、オブジェクトの定義域、インスタンスの定義は、GROOMの定義と同じである（文献(12)の[定義5]、[定義6]、[定義7]参照）。

5. 継承・再定義と多形性

以下では、属性文法間の能力の関係をを用いて、メソッド定義の継承・再定義および多形性を定義する。

5.1 クラススキーマ間での属性定義の継承・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 scheme lattice)を導出する。GROOM-IIは、GROOMと同様に、クラススキーマ束において属性定義の継承・再定義、クラスの包含関係といった概念を提供する。

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 c -属性列、 c -属性集合、 c -属性、生成規則集合を表す。

(1) c -属性列 w を構成する c -属性の集合を返す関数

$$ns(w, N) = \{X_i \mid w = X_1 \dots X_k, X_i \in N, 1 \leq i \leq k\}$$

(2) c -属性 X を左辺に持つ生成規則の右辺の排他的な各 c -属性列を構成する c -属性集合族を返す関数

$$ws(X, N, P) = \{ns(w_i, N) \mid X \rightarrow w_i, 1 \leq i \leq n\}$$

以下に、上位/下位クラススキーマ関係を定義する。

[定義7] クラススキーマ $CS_1 = \langle N_1, A_1, O_1, C_1, M_1, S_1, P_1 \rangle$ がクラススキーマ $CS_2 = \langle N_2, A_2, O_2, C_2, M_2, S_2, P_2 \rangle$ 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 $CS_3 = \langle N_3, A_2, O_2, C_2, M_2, S_1, P_3 \rangle$ 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 $CS_4 = \langle N_4, A_4, O_4, C_4, M_4, S_4, P_4 \rangle$ 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)-(3.5)のいずれかが成立する。

(3.1) $X \rightarrow w_{4,1} \mid \dots \mid w_{4,n_4} \in P_4, w_{4,i} \in N_4^+, 1 \leq i \leq n_4$ の場合、 $X \rightarrow w_{1,1} \mid \dots \mid w_{1,n_1} \in P_1, w_{1,j} \in N_1^+, 1 \leq j \leq n_1$ 、かつ、 $ws(X, N_1, P_1) \supseteq ws(X, N_4, P_4)$ 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、 $i_1 \leq i_4, j_1 \leq j_4$ 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、 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、 $\text{DOM}(b_1) \supseteq \text{DOM}(b_4)$ である。

(3.5) $X \rightarrow m \in P_4, \langle m, ag_4 \rangle \in M_4$ の場合、 $X \rightarrow m \in P_1, \langle m, ag_1 \rangle \in M_1$ 、かつ、(a) ag_1 と ag_4 は同じ能力を持つ、あるいは(b) ag_1 は ag_4 よりも強い能力を持つ。□

[定義7]の(3.5)は、クラススキーマ CS_1, CS_2 におけるメソッド・ドメイン間の条件に関するものである。これが、文献(12)の上位/下位クラススキーマ関係の定義に対して、本論文で追加を行った部分である。(a)は CS_1 から CS_2 に対して、メソッドを定義する属性文法が継承される場合である。また、(b)は CS_1 のメソッドを定義する属性文

法の能力を制限して、 CS_2 でメソッド定義を再定義する場合に相当する。

5.2 メソッドの多形性

多形性はオブジェクトのクラススキーマを正確に指定することなく、そのオブジェクトのメソッドを呼び出す機能である。多形性の例として、図3のクラススキーマの階層構造上における、Employeeオブジェクトの給与計算メソッドを考える。ここで、Employeeオブジェクトの給与計算法は、職種によって異なる。すなわち、WageEmployeeオブジェクトは時間給であり、Salespersonオブジェクトは時間給分と歩合給分とを足した給与であり、Managerオブジェクトは固定給となっている。このように、多形性は同じ給与計算といったメッセージ構文に対して、複数の異なる形態のメソッド実行を仮定する能力を意味する。

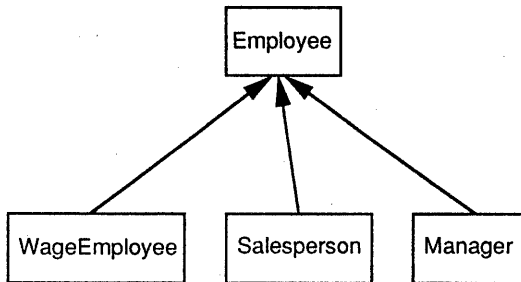


図3 従業員クラススキーマ階層

[定義8] 属性文法集合 $AGS=\{AG_1, \dots, AG_n\}$ が多形性であるための必要十分条件は、以下の通りである。但し、 $AG_i=\langle G_i, Y_i, F_i \rangle, G=\langle V_{Ni}, V_{Ti}, S_i', P_i' \rangle$ ($1 \leq i \leq n$)である。

- (1) AGSの各属性文法は、同じ生成能力を持つ。すなわち、 $L(G_1)=\dots=L(G_n)$ が成立する。
- (2) AGSの各属性文法は、同じ定義域を持つ。すなわち、すべての S_i' に $S-ATTR(S_i')=\{y_i\}$ であるような合成属性 y_i が付随すれば、 $V(y_1), \dots, V(y_n)$ が同じ定義域の部分集合となっている。そうでなければ、 $S-ATTR(S_1')=\dots=S-ATTR(S_n')=\emptyset$ が成立する。
- (3) AGSの各属性文法は、異なる変換能力を持つ。すなわち、すべての異なる $AG_i, AG_j (\in AGS)$ の対に対して、 $T(AG_i) \neq T(AG_j)$ が成立する。

□

[定義8]の(1)は、多形性の機能を提供するメソッド集合の各要素のインタフェース仕様が同じであることに関する条件である。(2)は各メソッドが実行結果の戻り値を持てば、それらが互換性のある(compatible)定義域を持つことに関する条件である。また、(3)は各メソッドが異なる変換を行うことに関する条件である。

6. メソッド実行と意味規則

以下では、メソッド実行のための演算子、メソッドの意味規則記述、文脈依存のg-属性計算に関して述べる。

6.1 オブジェクト操作言語

オブジェクト指向データモデルGROOMに対しては、オブジェクト操作言語GROOM/PL⁽¹²⁾が提案されている。GROOM/PLは、オブジェクトおよびオブジェクト木に対する問合せ・更新を行う原始演算子の集合である。本論文では、GROOM-IIに対するオブジェクト操作言語GROOM-II/PLを考える。GROOM-II/PLには、GROOM/PLが提供する原始演算子が備わっているとともに、メソッド実行のための次の演算子が追加される。

(1) issue-message(OBJT, NADDR, MSG)

オブジェクト木OBJTにおいてノードアドレス⁽¹²⁾NADDRが指すノードの子ノードに対してメッセージMSGを送り、メソッドの実行結果の戻値を返す。

6.2 メソッドの意味規則記述

メソッドのインプリメンテーション仕様を与える属性文法の意味規則記述においては、次の演算子を用いることができる。

- (1) 一般的計算機能用の演算子の集合
- (2) GROOM-II/PLの演算子の集合
- (3) 以下の特殊な演算子

self():当該メソッドを実行しているオブジェクトの識別子を返す。

node-address():当該メソッドを実行しているオブジェクトのオブジェクト木における当該メソッドのノードアドレス⁽¹²⁾を返す。

value():属性文法による文解析において、終端記号を返す。

6.3 文脈依存の属性計算

GROOM-IIでは、オブジェクトの情報はオブジェクト木として貯えられる。オブジェクト木は、

c-属性、基本ドメイン、オブジェクト・ドメイン、定数を階層構造の下に組織化する。この組織化は、メソッドに関しても同様に適用される。すなわち、メソッド・ドメインはオブジェクト木の葉ノードとして、生成規則に従う任意の箇所に出現する。これは、GROOM-IIのメソッド実行の特徴である文脈依存のg-属性計算(context-sensitive attribute computation)をもたらす。文脈依存のg-属性計算とは、メソッド・ドメインが出現するノードの周囲環境に応じてg-属性が計算されることをいう。図4に、文脈依存のg-属性計算の例を示す。図中のメソッドmをラベルとして持つ各ノードの位置を基に各ITEMのUNIT-PRICE, QTYの値を参照することが必要となる。これは、前記のself(), node-address()およびget-children-label(OBJT, NADDR)⁽¹²⁾などの演算子群を用いることで可能である。

7. おわりに

本論文では、オブジェクト指向データモデルGROOMを拡張し、メソッド・ドメインを取り入れたGROOM-IIを提案した。メソッド定義のために属性文法を用いたが、データベースへの応用を目的としている点で属性文法の基本原理の一つである木構造上のg-属性の値の純関数的な計算モデルからは逸脱している。このため、意味規則およびその上での変換能力の定義を拡張した。これらの準備の下に、メソッド定義の継承・再定義および多形性は、文の生成能力と変換能力に基づいた対応する属性文法間の条件として定義された。

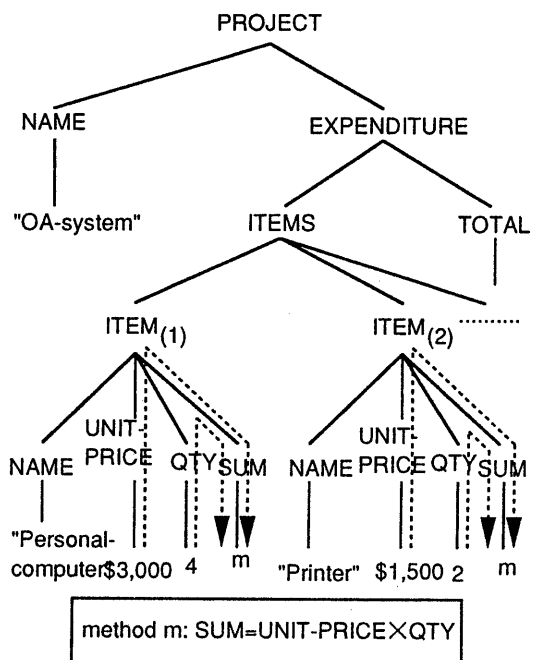
今後の課題は、(1)関数従属性などの制約記述の検討、(2)高水準問合せ言語の設計とその処理系の実装、(3)ビュー・導出データの検討などが挙げられる。

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

```
@PROJECT=<N,A,O,C,M,PROJECT,P>
P= {PROJECT→NAME EXPENDITURE,
    EXPENDITURE→ITEMS TOTAL,
    ITEMS→ITEM*,
    ITEM→NAME UNIT-PRICE QTY
SUM,
    NAME→String,
    UNIT-PRICE→Integer,
    QTY→Integer,
    SUM→m,
    ....}
M= {<m,ag>,.....}
```

(a) クラススキーマ @PROJECT



(b) オブジェクト木上のメソッド実行

図4 文脈依存の属性計算

- Relational Databases", ACM Trans. Database Syst., 13, pp.389-417 (1988)
- (5) Hull R., King R. : "Semantic Database Modeling: Surveys, Applications and Research Issues", ACM Comput. Surv., 19, 3, pp.201-260 (1987)
- (6) Peckham J., Maryanski F. : "Semantic Data

- Models", ACM Comput. Surv., 20, 3, pp.153-189 (1988)
- (7) Atokinson M., Bancilhon F., Dewit D., Dittrich K., Maier D., Zdonik S. : "The Object-Oriented Database System Manifesto", Proc. DOOD89, pp.40-57 (1989)
 - (8) 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)
 - (9) Gonnet G.H., Tompa F.W. : "Mind Your Grammar:a New Approach to Modelling Text", Proc. VLDB Conf., pp.339-346 (1987)
 - (10) Gyssens M., Paredaens J., Gucht D.V. : "A Grammar-Based Approach Towards Unifying Hierarchical Data Models", Proc. VLDB Conf., pp.263-272 (1989)
 - (11) Colby L.S., Gucht D.V. : "A Grammar Model for Databases", Techn. Rep. No.282, Comput. Sci. Dept., Indiana Univ. (1989)
 - (12) 佐藤秀樹, 伊藤暢浩, 林達也 : "文法表現によるオブジェクト指向データモデル", 情処学データベースシステム研報, DBS102-9, (1995-03)
 - (13) Knuth D. E. : "Semantics of Context-Free Languages", Mathematical Systems Theory, 2, 2, pp.127-145 (1968)
 - (14) 西野哲朗 : "属性文法の理論入門", 情報処理, 35, 3, pp.251-258 (1994-03)
 - (15) Levene M., Poulouvasilis A. : "An Object-Oriented Data Model Formalised Through Hypergraphs", Data & Knowledge Eng., 6, pp.205-224 (1991)
 - (16) Shinoda Y., Katayama T. : "Object Oriented Extension of Attribute Grammars and Its Implementation Using Distributed Attribute Evaluation Algorithm", Lecture Notes in Computer Science 461, Springer-Verlag, pp.177-191 (1990)
 - (17) Koskimies K. : "Object-Orientation in Attribute Grammars", Lecture Notes in Computer Science 545, Springer-Verlag, pp.297-329 (1991)
 - (18) 佐々正孝 : "属性文法によるコンパイラの記述例", 除法処理, 35, 4, pp.358-369 (1994-04)
 - (19) 今泉貴史, 篠田陽一 : "ソフトウェア環境への属性文法の応用", 情報処理, 35, 7, pp.651-657 (1994-07)