データベースシステム 111－2
（1997．1．21）

階層構造グラフを用いた半構造化データの段階的構造化手法の提案

森下 淳也*, 上島 紳一*, 大月 一弘†, 杉山 武司*

*姫路獨協大学情報科学センター, *関西大学総合情報学部, †神戸大学国際文化学部

あらまし：本稿では，利用者が様々な視点に応じてオブジェクトを取り扱い，視点に応じたビューを実現するため，視点に基づく属性を実行時に生成する機構，視点を切り替えることで異なるオブジェクトの属性構造を実現する機構，及び視点と視点に依存する属性構造を収納する機構を提案する．まず，視点と視点に依存する属性を収納するため，階層構造グラフを導入する．更に視点に応じた属性構造を持つオブジェクトのビューを与える仮想オブジェクト，及び同一観点から統一的に仮想オブジェクトを取り扱うためのスコープの概念を導入する．階層構造グラフに基づくオブジェクトを扱うことで，対象となるオブジェクトの多様な属性構造の実現，階層的な視点間の関係の表現を実行時に行える．次に，スコープ内の仮想オブジェクトと視点グラフからなる利用環境を規定する．この環境のもとで階層構造グラフを段階的に構築するための利用者の操作の間接的な制限，追加機能などを検討し，仮想オブジェクトへの属性付与のためのルール，グラフの精細化におけるルールを与える．これらにより，スコープの相互独立性が保たれ，複数の利用者による協調作業が可能となる．更に利用者の更新操作を矛盾なく階層構造グラフに表現するためのアルゴリズムを与える．

Incremental View Generation for Semi-structured Data Using Hierarchical Graph

Jun-ya MORISHITA*, Shinichi UESHIMA*, Kazuhiro OHTSUKI†,Takeshi SUGIYAMA*

*Information Science Center, Himeji Dokkyo University *Faculty of Informatics, Kansai University
†Faculty of Cross-cultural Studies, Kobe University

Abstract : In this paper, we give a mechanism to handle objects from various viewpoints and their *object views*. The mechanism consists of the functions that (i) generate viewpoint dependent attributes at run-time, (ii) realize multiple attribute structure of objects by swiching viewpoints, and (iii) store viewpoints and viewpoint dependent attributes. First, we introduce a hierarchical graph to store viewpoints and viewpoint dependent attributes. We define *virtual objects* that gives *object views* with personal purpose dependent attributes of specified objects, and also define *scopes* to handle virtual objects from various viewpoints in a unified manner. Users can generate virtual objects incrementally. By handling objects in the hierarchical graph, users can realize multiple arrtibute structure of objects, and hierarchical relationship among viewpoints. Then we give users' environment to access to hierarchical graphs. Users are given virtual objects in scopes and *viewpoint graph* derived from hierarchical graph. We discuss the indirect restriction of users'operations, additional functions in handling virtual objects. We also give a rule in the addition of attributes to virtual objects, and a rule in refining a hierarchical graph. Due to these rules, scopes become mutually independent, and cooperative work by multiple users becomes possible. Moreover, we give algorithms to generate hierarchical graph incrementally corresponding to users' operations in the users' environments.

1 はじめに

最近，科学技術データベースにおいて半構造化データに関する研究が注目されている [1, 2, 3, 4]. ここでは，特にシステムへの格納時に基本的な属性のみが与えられており，完全な属性構造が確定していないデータを取り扱う．この種のデータに対しては，利用者が様々な視点に応じてデータを取り扱い，視点に応じたビューを実現することが重要である．以下では，利用者が，利用時に様々な視点を与えたり，視点に基づいて新たに属性を付けたりしながらデータを取り扱うことを考える．

このような場合，視点は次の4つの性質を持つ．

- **視点に基づく属性の独立性**
  利用者が視点に基づき付与した属性は，視点に依存して取り出せる．
- **視点の多重性**
  同一データに対して異なる複数の視点から属性が付与できる．
- **視点の多層性**
  既に存在する視点を用いてより大きな視点からデータをとらえたり，1つの視点をより細分化した視点からデータを扱う．このため，視点間の関係が階層になる．
- **視点の部分性**
  同一視点に基づき複数のデータが取り扱われる．但し，1つの視点はすべてのデータを対象にするのではなく，着目する部分的なデータのみを対象とする．

本稿では，視点に基づく属性を実行時に生成する機構，視点を切り替えることで異なるオブジェクトの属性構造を実現する機構，及び視点と視点に依存する属性構造を収納する機構を提案する．

まず，視点と視点に依存する属性を収納するため，**階層構造グラフ**を導入する．利用者は，階層構造グラフ上で，視点に相当する**カテゴリー**を生成し，視点と対象とするオブジェクト（基底オブジェクト）の組み合わせでオブジェクトのビューを生成し，属性構造を与えることができる．これを**オブジェクトの仮想化**という．オブジェクトのビューは，階層構造グラフ上で**仮想オブジェクト**として与えられる．仮想オブジェクトを定義することで，利用者はオブジェクトのビューをオブジェクトと区別なく振る舞うオブジェクトとして取り扱える利点がある．

本稿で述べるオブジェクトの仮想化は，従来提案されているオブジェクトのビュー [5, 8] に比較して，利用者が実行時に与える属性を利用する点，実行時継承 [3, 6] による仮想オブジェクトを生成する点，視点を指定することで注目するオブジェクトのみが仮想化される点が異なる．

階層構造グラフでは，1つのオブジェクトに対して多種の仮想オブジェクトが定義されるが，これらはグラフ上には陽には表れない．同一観点から統一的に仮想オブジェクトを取り扱うため，**スコープ**の概念を導入する．スコープは視点階層における視点の切り替え操作を提供する．利用者の操作環境としては，階層構造グラフから導出されたスコープ内の仮想オブジェクトと視点グラフである．利用者は原則としてこの2つを用いる．

利用者は，この利用環境のもとで階層構造グラフを実行時に段階的に構築していく．本稿のアプローチでは，実行時の視点生成と属性構造の生成，実行時の視点グラフの生成を行うため，インスタンスベースモデルを用いる [3, 6]．また，このグラフ構造は複数の利用者が複数の目的に応じて独立に構築するため，既に構築されたグラフに影響を与えずに，グラフが追加される必要がある．このため，利用者の操作の間接的な制限，追加機能などを検討し，仮想オブジェクトへの属性付与のためのルール，グラフの精細化におけるルールを与える．更に利用者の更新操作を矛盾なく階層構造グラフに表現するためのアルゴリズムを与える．

このアルゴリズムは仮想オブジェクトの段階的な生成と仮想オブジェクトの管理に相当する．通常のビュー更新と比較して，階層構造グラフの構造を生成することがオブジェクトのビューを生成することに相当する点，オブジェクトビューを通して属性が新しく生成される点，その結果，スコープ内の仮想オブジェクトの属性構造に再び更新が伝搬される点が異なる [5].

本手法により，半構造化データなどの不完全データへの属性の補完，オブジェクトの持つ属性の仮想的な更新と隠蔽，利用者毎のデータベース作成，内容に応じた多重分類 [9] や複数の視点からの意味付け，複数の利用者による協調作業などを支援することが可能となる．

2 階層構造グラフと仮想オブジェクト

本節では，階層構造グラフを導入し，目的に応じた属性構造を持つ仮想オブジェクトの定義，及び視点間の関係の階層表現を行う．階層構造グラフは利用者の操作に応じて生成される．

図1に階層構造グラフの例を示す．階層構造グラフは，ノードと枝からなるサイクルのない有向グラフである．ノードと枝は，属性$\alpha_i, \beta_j, \rho_{kl}$を各々持つ（四角形で表される）．これらの属性は次のような意味を持つ．ノードの属性α_i, β_jは，ノード自身の性質を表す属性，枝の属性ρ_{kl}は上位ノードに依存した下位ノードの性質を表す属性である．例えば，α_2, β_5は各々C_2, D_5自身の性質を示す属性を示し，ρ_{25}はC_2に依存したD_5の性質を示す属性である．

半構造化データの構造化操作では，データを葉ノード(D_4, D_5, D_6)に置き，葉ノード以外のノードC_1, C_2, C_3や属性ρ_{kl}などを利用者が実行時に生成・更新する．以後，データをオブジェクトとして格納

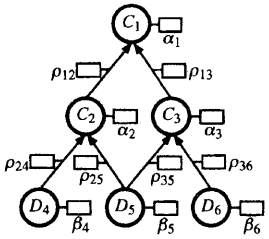

図 1: 階層構造グラフ

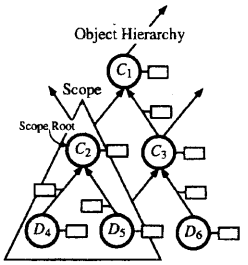

図 2: スコープ C_2

したものを**データオブジェクト**と言う．また，葉ノード以外のノードを**カテゴリー**と呼ぶ．

カテゴリーは1つの視点を表す．また，上位のカテゴリーは，下位のカテゴリーを包含（引用）する．

次に，階層構造グラフを用いて，オブジェクトの仮想的な属性構造を表すため仮想オブジェクトを定義する．

階層構造グラフ上で2つのノードを結ぶ有向グラフに**仮想オブジェクト**を一意的に対応させる．仮想オブジェクトの属性はパスに沿った属性を下向きに集約し，並置したものと定義する．この場合，属性名が重複する場合はそのまま重複して並置する．2点を結ぶパスが複数存在する場合は，すべてのパスに沿った属性を集約する．仮想オブジェクトの属性は，もとのオブジェクト自身が持つ属性に，あるカテゴリーから見た属性が加えられている．

例えば図 1で，C_2 と D_5 で定義される仮想オブジェクトを $\tilde{D}_5[C_2]$ と表記する．$\tilde{D}_5[C_2]$ の属性は，ρ_{25},β_5 となる．また，C_1と D_5で定義される仮想オブジェクトを$\tilde{D}_5[C_1]$ の属性は，2つのパス $C_1/C_2/D_5, C_1/C_3/D_5$に沿って継承された$\rho_{12},\alpha_2,\rho_{25},\rho_{13},\alpha_3,\rho_{35},\beta_5$ となる．同様にして，$\tilde{D}_4[C_1]$, $\tilde{D}_4[C_2]$, や，カテゴリー C_1に対する仮想オブジェクト$\tilde{C}_2[C_1]$ なども定義される．

階層構造グラフ上で，1つのオブジェクトに対して異なるカテゴリーを選択する毎に異なる仮想オブジェクトが定義される．仮想オブジェクトにより，もとのオブジェクトの異なる複数の属性構造 を仮想的に与えることができる．図 1では，D_5に対して $\tilde{D}_5[C_1]$, $\tilde{D}_5[C_2]$, $\tilde{D}_5[C_3]$ の3つの属性構造を与えることができる．これらは D_5に対する仮想オブジェクトであり，D_5のオブジェクトビューとみなせる．ここではオブジェクトの仮想化のみを考えており，スキーマの仮想化は考えていない．

階層構造グラフは，インスタンスベースモデルで実現できる [3, 6]．即ち，データオブジェクト，カテゴリー，枝はすべてインスタンスとして表し，グラフ上の属性はインスタンスの属性として表現する．また，上位/下位の関係にあるカテゴリー間の関係は，インスタンス間の super/sub 関係で表す．利用者が，あるカテゴリーとデータオブジェクト（或いは別のカテゴリー）を指定すると，上位から下位への属性の継承が行われる．このとき枝の持つ属性（以後，関係属性という）も同時に継承される．カテゴリーは通常の OODBMS (Object Oriented DataBase Management System) のクラスではない．以下では，階層構造グラフはインスタンスベースモデルで実現されているものとし，オブジェクト階層と呼ぶ．

3 スコープと利用者の操作環境

階層構造グラフ上では複数の仮想オブジェクトが定義できるが，仮想オブジェクトはグラフ上に陽に表れない．このため，利用者が階層構造グラフを直接参照しながら作業を行うことは容易でない．そこで同一観点から階層構造グラフ上の仮想オブジェクトを統一的に取り扱うため，**スコープ**の概念を導入する．スコープは利用者の環境を規定し，操作の簡便性を与える．

3.1 スコープ

階層構造グラフ上で，1つのカテゴリーを始点として下位方向へデータオブジェクトまで至るパスの存在するグラフの部分グラフを**スコープ**という．スコープを決める始点となるカテゴリーを**スコープ・ルート**と呼ぶ．以下では簡単の為，スコープ・ルート C_2を持つスコープを単にスコープ C_2と呼ぶ（図2）．図2では，C_2, D_5から出る有向枝が切れており，D_5へは C_2のみから継承が行われている．スコープは，スコープ・ルートと繋がらないパスを隠蔽することで，オブジェクト階層上の多重継承に対して**選択的継承**を行う．

利用者は複数のスコープを同時に利用するとが考えられるため，次の用語を準備しておく．利用者が現在指定しているカテゴリーを**カレント・ルート**，対象となる作業範囲を**カレント・スコープ**，カレント・スコープに含まれる全オブジェクトの集合を**対象オブジェクト**という．スコープを指定すれば，そのスコープ内に分類されている対象オブジェクトと仮想オブジェクトが1対1に対応する．スコープ C_2

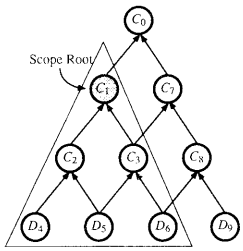

図 3: 視点グラフ

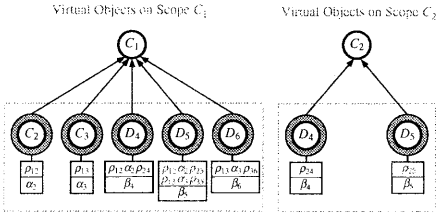

図 4: スコープ C_1, C_2による仮想オブジェクト

の対象オブジェクトはデータオブジェクトのみであり，D_4と D_5である．

3.2 利用者の環境

視点グラフとスコープ内の仮想オブジェクト

利用者のシステムの利用環境を以下のように定める．即ち，利用環境として，スコープで規定される仮想オブジェクト，及び多段化されたカテゴリーの形成する視点グラフとする．利用者は両者をもとに操作を行い，階層構造グラフを間接的に生成する．

階層構造グラフから属性を取り除いたグラフ，つまり視点間の関係を表すグラフを**視点グラフ**という（図 3）．

視点グラフ上で，カレント・スコープ C_1を指定すると，対象オブジェクトが C_1に対する仮想オブジェクトとして表示される（図 4左図）．カレント・スコープが変更されれば，同一の実オブジェクトに対しても，その仮想オブジェクトは変更される．図 4右図に，カレント・スコープを C_2に移動させた場合の仮想オブジェクトを示す．

図 4では，対象オブジェクトはカテゴリー，データオブジェクトの両者は区別なく仮想オブジェクトとして並列に表示される．各々の対象オブジェクトがカレント・カテゴリーに対して直接的/間接的に結合されているかどうかは陽には表れないが，間接的に結合されているものに関しては，上位に位置するカテゴリーの属性はすべて継承されている．

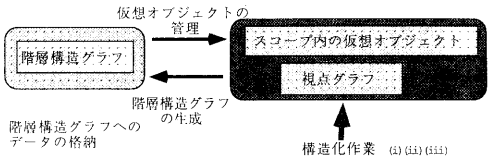

図 5: 階層構造グラフの生成

スコープの概念を用いることにより，利用者は，(1) 実オブジェクトと仮想オブジェクトの区別を強く意識しない，(2) 仮想オブジェクトの操作において階層を意識しない，などの利点が生まれ，利用者にとって操作が容易になる．

利用者の作業と階層構造グラフの更新

利用者は，上記の利用環境で，(i) カテゴリーの生成による仮想オブジェクトの生成，(ii) 仮想オブジェクトへの属性付与，(iii) 複数の視点を統合してより上位の視点を生成したり，1 つの視点をより細分化した視点からオブジェクトを扱う視点を生成する，などの作業を行うものとする．

利用者の環境での仮想オブジェクトは，階層構造グラフから導出されたものである．上記の作業は，スコープ内の仮想オブジェクトと視点グラフを通して行われるため，(i)(ii)(iii) の操作に対して，利用者が行う操作を間接的に制限したり，追加操作を促すような付加機能が必要となる．また，階層構造グラフを更新するアルゴリズムが必要である (図 5)．この点について次節で述べる．

スコープの中で仮想オブジェクトを通してある視点の上位の視点を切り替える場合でも，カテゴリーの下位に位置づけられるものは実オブジェクトである．実オブジェクトの上位/下位の関係が更新され，実行時継承の働きによって，それがまた，仮想オブジェクトとなる．これは利用者環境からの仮想オブジェクトを通してのオブジェクトビューの生成と仮想オブジェクトの管理に相当する．

4 階層構造グラフの段階的形成アルゴリズム

本節では，仮想オブジェクトへの属性付与，カテゴリーの新規生成，視点間の上位/下位の関係の切り替えなどの利用者の操作に応じて，段階的に階層構造グラフを形成する手法について述べる．

4.1 協調作業とスコープの影響範囲

本アルゴリズムでは，複数の利用者による協調作業を可能にするため，(R1) 他の利用者の作業に影響

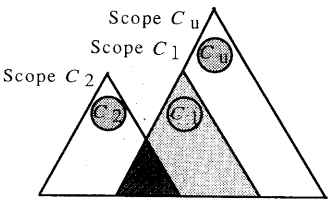

図 6: スコープの影響範囲

を与えない，(R2) 他の利用者の作業を参照・利用できる，ことを目標とする．

階層構造グラフでは属性の多重継承が許されるため，スコープの部分スコープが上位以外の他のスコープに含まれる場合がある（図 6）．スコープ内で行われる利用者の操作が，上位以外の他のスコープから見た仮想オブジェクトと視点グラフに影響を与えないためには，そのスコープにおける上位/下位の参照関係を保持する必要がある．

スコープ内で行われる上記の操作は，上位のスコープから見た視点部分グラフと仮想オブジェクトのみを変更するものとする．スコープ・ルート C_1 内の変更に対して，上位のスコープ C_u にのみ影響を与え，上位以外の他のスコープ C_2 に影響を与えないものとする．

4.2 仮想オブジェクトへの属性付与

まず，仮想オブジェクトに属性を付与することを考える．階層構造グラフの形状により，利用環境から付与しようとした属性の位置が階層構造グラフで一意に確定できなかったり，また付与できない場合があるため，次の規則，または追加操作が必要となる．

- 仮想オブジェクトの基底オブジェクトがスコープ・ルートに直接的に結合している場合．
  - データオブジェクトの場合は，直上の枝に付与する．
  - カテゴリーの場合は，直上の枝に付与できるが，配下にも継承されることを利用者に警告する必要がある．これには ECA ルール機構を用いる [7, 3]．
- 実オブジェクトがスコープ・ルートに間接的に結合している場合．
  原則的に付与することはできないため，補助手段として階層構造グラフを提示し，利用者が属性の位置を陽に指定する．

4.3 カテゴリーの新規生成とオブジェクトの登録

ここでは，2 つの構造化ルールを導入することでカテゴリーを新規生成し，オブジェクトを登録する場合についてアルゴリズムを与える．以後，データオブジェクトとカテゴリーをまとめて単にオブジェクトという．

[新規カテゴリー生成アルゴリズム]
カレント・ルートを C_x とし，カレント・スコープ内にカテゴリーを新規に生成する手順を述べる．まず，上位に親カテゴリーを持たないカテゴリー（ルート・カテゴリーという）を生成する場合について以下にアルゴリズムを与える．

1. 新規カテゴリー C_n の生成．
2. C_n の要素の候補となる実オブジェクト（候補オブジェクトという）の指定．
   カレント・スコープの中に含まれる実オブジェクト集合を指定するか，または，W を用いて，全実オブジェクト集合を指定する．利用者は実オブジェクトの指定を視点グラフを用いて行う．
3. 候補オブジェクトの仮想化
   カテゴリー C_y を指定し．実オブジェクトを C_y により仮想化する．C_y の選び方には以下の方法がある．
   (a) $C_y = C_x$ としてカレント・スコープ内の仮想オブジェクトを利用する．
   (b) $C_y \neq C_x$ として候補オブジェクトを C_y をスコープ・ルートとするスコープで見て，C_x とは別の観点から仮想化して見る．また，C_y 内のオブジェクトを更に別の C_z をスコープ・ルートとするスコープで見て，仮想化して見ることができる．
   (c) $C_y = C_t$ としてテンポラリカテゴリー C_t を一時的に設けて，候補オブジェクトとの間に属性を付与することで仮想化して利用する．
   (d) $C_y = W$ として全実オブジェクトを利用する．
4. 候補オブジェクトからの選択・抽出．
   仮想化されたオブジェクトの集合に対して，検索やマニュアル操作で C_n のサブオブジェクトとする実オブジェクトを候補オブジェクトから選択する．
5. C_n への登録
   視点グラフにおいて，ステップ 4 で選択した実オブジェクトを C_n のサブオブジェクトとして登録する．
6. 仮想化．
   C_n をカレント・ルートとして登録したオブジェクトを仮想化して属性を付与する．仮想化については後述する．[アルゴリズム終了]

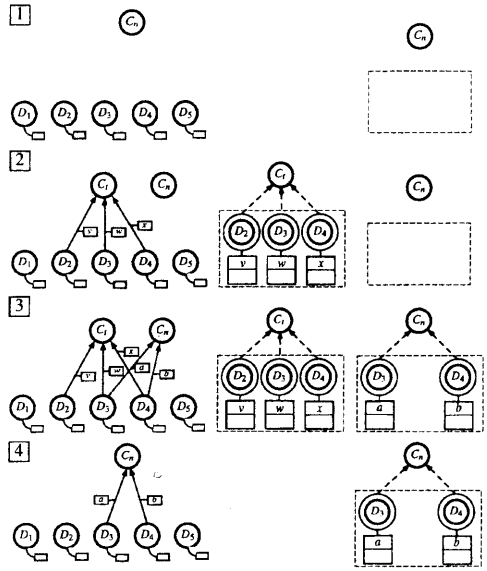

図 7: テンポラリカテゴリーを用いたルートカテゴリーの生成過程

[新規カテゴリー生成アルゴリズム] では，C_nをルートカテゴリーとして生成したため，それまでの階層構造グラフ上のどのスコープに対してもまったく影響を与えない．ステップ 2 において候補オブジェクトの選択範囲を拡げてもどのスコープにも影響を与えない．このアルゴリズムにより，カテゴリー C_n をスコープ・ルートとする新しいスコープが生成される．

図 7に仮想化されていないデータオブジェクトに対してテンポラリカテゴリー C_tを用いてルート・カテゴリー C_nを新規生成する場合の過程を示す．左図は階層構造グラフの変遷，中図と右図はそれぞれスコープ・ルート C_t，C_nから見た仮想オブジェクトの変遷を示す．利用者は，中図，右図の仮想オブジェクトを見る．アルゴリズムでは，システムに W とデータオブジェクトのみ存在する場合新規にカテゴリーを生成する場合，3(d) の場合にあたる．

また，図 8は，カレント・スコープを C_xとする時にルート・カテゴリー C_nを生成している．この図では，利用者がカレント・スコープ内の D_2，D_3 をスコープ C_yから仮想化し，更に D_2をスコープ C_zから仮想化した結果，候補オブジェクトとして D_2，D_3を選択したところを意味する．ルート・カテゴリー C_nにより新しいスコープが生成されている．アルゴリズム中では 3 (b) にあたる．次に，カレント・スコープ内に親オブジェクト C_pを持つカテゴリー C_nを新規に生成する場合について述べる．階層構造グラフでは属性の多重継承が許されるため，1 つのスコープの部分スコープがカレント・スコープの上位スコープ以外の他のスコープに含まれる場合がある．カレント・スコープ内で親オブジェクト C_pをカレント・ルート以外のカテゴリーを選択（$C_x \neq C_p$）すれば，他のスコープで見た仮想オブジェクトが変更される場合がある．これを回避するため，次のルールを定義する．

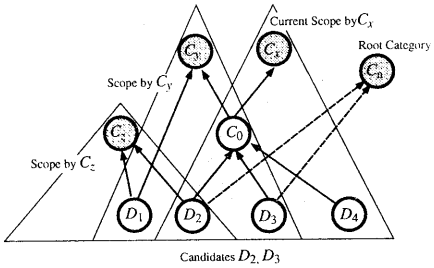

図 8: 複数のスコープを用いたルート・カテゴリーの生成

ルール 1 ルートカテゴリー以外のカテゴリーを生成する場合は，カテゴリーをカレント・ルートの直下に生成する．即ち，$C_p = C_x$とする．

[ルール 1] により上位以外の他のスコープに影響することなく，カテゴリーを生成できる．これにより**スコープの相互独立性**が保たれる．

つまり，C_nが親カテゴリー C_pを持つ場合は，上記のアルゴリズムで $C_p = C_x$とすればよい．アルゴリズムのステップ 3 以降は同様である．

本アルゴリズムで $C_x = C_y$と指定し，候補オブジェクトをカレント・スコープ内のオブジェクトのみを対象オブジェクトとする場合のアルゴリズムを特に**階層構造グラフの精細化**という．精細化は，スコープ内のオブジェクトをまず広い視点からのオブジェクトを仮想化して，次に，更に狭い視点からのオブジェクトを仮想化する場合に相当する（図 9 参照）．

上記のアルゴリズムで，利用者は仮想オブジェクトを操作するが，カテゴリーには仮想オブジェクトのもととなる実オブジェクトが登録され，付与された属性も実オブジェクトとして登録される．結果的に階層構造グラフが段階的に構成される．利用者は仮想オブジェクトを実オブジェクトと区別せずに取り扱うことができる．

本アルゴリズムは，段階的構造化の初期状態，即ち，階層構造グラフとしてデータオブジェクトと W のみしか存在しない状態，からのカテゴリーの生成法，また，2 つのカテゴリーの上位のカテゴリーの生成法も与えている．

[仮想化]

ステップ 4 で，カテゴリー C_n に対して登録したオブジェクトを C_nにより仮想化して属性を付与する．付与された属性を階層構造グラフ上に格納する場所

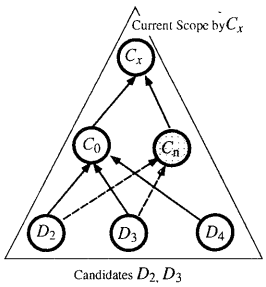

図 9: 精細化

は複数が考えられる．C_nがルート・オブジェクトの場合は，C_n，またはC_nの直上の関係オブジェクトの属性が考えられる．C_nが親オブジェクトを持つ場合は，更に，親オブジェクトの属性，C_nと親オブジェクトの間の関係オブジェクトの属性である．この中から位置を決定するには，付与される属性の意味を考慮する必要がある．ここでは，次のルールに基づき属性を格納する．

ルール 2 選択されたオブジェクトをC_nにより仮想化した場合，仮想オブジェクトに付与された属性は，階層構造グラフ上では選択されたオブジェクトの直上の関係オブジェクトの属性になる．

属性付与の方法としては，以下がある．

- C_x，C_y，C_zから仮想化したオブジェクトが持つ属性をすべて複製して関係オブジェクトの属性とする（この場合は，検索などの方法による自動化が可能），
- コピーした属性を更新する，
- 属性を全く付与しない．

属性の付与後は，更に，関係オブジェクトの共通の属性を意味に応じて，カテゴリーC_nや更にその上位の関係オブジェクトの属性に移動する操作などがある．また，検索を用いてC_nに登録するオブジェクトを選択した場合は，検索に用いた属性をC_nの属性としたり，検索自身をC_nの属性とすることも可能である．

4.4 階層構造グラフへの影響

前節のアルゴリズムの実行の結果，階層構造グラフが受ける影響は以下の通りである．

- C_n がルート・カテゴリーの場合
  他のスコープに基づく視点間の関係も仮想オブジェクトも全く影響を受けない．
- C_n が親オブジェクト C_p を持つ場合
  C_p の上位カテゴリーで定義されるスコープ内の仮想オブジェクトの数が増加する．また，各仮想オブジェクトは膨らむ．C_p の上位カテゴリー以外のカテゴリーから見た仮想オブジェクトは全く影響を受けない．即ちスコープの独立性が保たれる．
- 精細化の場合
  カテゴリーが1つだけ増加する．また，C_p の上位カテゴリーをスコープ・ルートとすれば，スコープ内の仮想オブジェクトの属性が増加する可能性がある．C_p の上位カテゴリー以外のカテゴリーから見れば，仮想オブジェクトは全く影響を受けない．

また，いずれの場合も，新規にカテゴリーを生成しても仮想オブジェクトの数が増加するだけで，減少しない．C_n が親オブジェクト C_p を持つことで，既存の階層構造グラフの属性を継承できる．また，属性付与の際に，既存の属性をそのまま利用できる．

このようにスコープを用いることで，利用者は作業範囲のオブジェクトを指定しながら，同時に他の利用者の作業を参照・利用したり，また，他の利用者が仮想オブジェクトに対して付与した属性を残すことができる．つまり段階的構築において既存の階層構造グラフの意味を保全することができる．

5 システム

本稿で提案した手法の有効性を確認するため，電子メールをデータとしてプロトタイプシステムを作成した（図10参照）．後部のウインドウに階層グラフがあり，カテゴリー0をスコープルートとした電子メール（仮想オブジェクト）の内容が最上のウインドウに表示されている．最上のウインドウで左部に属性，右側に仮想オブジェクトの内容が表示されており，利用者は仮想オブジェクトに対して属性付けや，基底データの内容の更新などの操作を行うことができる．このプロトタイプシステムは簡単のため，格納されるデータは現在はテキストのみであるが，階層構造グラフ上を自在に traverse でき，カテゴリーの生成/消去，またスコープを指定すればそのスコープで定まる仮想オブジェクトをすべて閲覧でき，属性について検索が可能である．

6 あとがき

本稿では，半構造化データの構造化を支援する枠組みとして，階層構造グラフを導入し，仮想オブジェクト，スコープを定義して利用者の操作環境を規定し，スコープ内の仮想オブジェクトと視点グラフで構成される利用者の利用環境からの操作のアルゴリズムについて述べた．

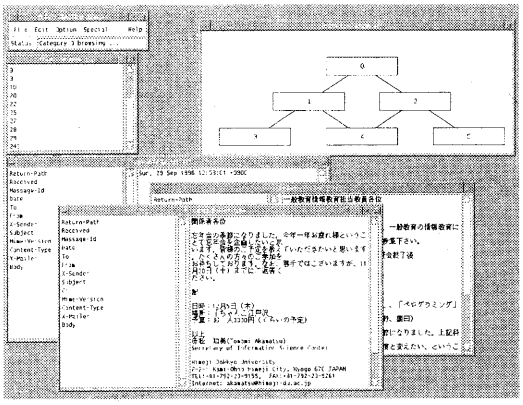

図 10: システム

利用者は，システムの利用時にアド・ホックに属性を生成するため，属性の数が増大するものと考えられる．これに対して，属性の管理機構が必要となる．このため，属性の索引表，知識ベースなどを利用することが考えられるが，この点については別稿で述べる．

参考文献

[1] IEEE Computer Society, "Special Issue on Scientific Databases," Bulletin of the Technical Committee on Data Engineering, vol.16, no.1, Mar. 1993.

[2] Zdonik,S., "Incremental Database Systems : Databases from the Ground Up," Proc. of the 1993 ACM SIGMOD International Conference on Management of Data, Washington DC, USA, pp.408–412, May 1993.

[3] Ueshima, S., Ohtsuki, K., Morishita, J., Qian, Q., Oiso, H. and Tanaka, K., "Incremental Data Organization for Ancient Document Databases,", Proc. of the Fourth International Conference on Database Systems for Advanced Applications(DASFAA'95), pp.457–466, Singapore, Apr. 1995.

[4] Shoens, K. Luniewski, A., Schwarz, P., Stamos, J., Thomas, J., "The Rufus System: Information Organization for Semi-structured Data," Proc. of the 19th VLDB Conference, Dublin, pp.97–107, Ireland, 1993.

[5] Kambayashi, Y., Peng, Z., "Object Deputy Model and Its Applications," (keynote Paper) Proc. of the Fourth International Conference on Database Systems for Advanced Applications(DASFAA'95), pp.1–15, Singapore, Apr. 1995.

[6] Tanaka,K., Nishio,S., Yoshikawa,M., Shimojo,S., Morishita,J. and Jozen,T., "Obase Object Database model: Towards a More Flexible Object-Oriented Database System," Proc. of the International Symposium on Next Generation Database Systems and Their Applications (NDA'93), pp.159–166, Sept.1993.

[7] McCarthy, D.R. and Dayal, U., *The Architecture of an Active Data Base Management System*, Proc. of ACM SIGMOD Symposium on the Management of Data, pp.215-224, Oregon, 1989.

[8] Abiteboul,S., Bonner, A., "Objects and Views," Proc. of the 1991 ACM SIGMOD, International Conference on Management of Data, Denver, Colorado, USA, pp.238–247, Feb. 1991.

[9] 山崎康雄, 古川哲也, 島崎真昭, "データベースを用いたデータ解析のための集合操作," 情処研報, DBS94-5, pp.39–48, Jul.1993.