# オブジェクト指向モデルを用いた要求獲得の支援

斉藤康彦† 本位田真一††

標準的な業務知識をオブジェクト指向モデルを用いて表現した業務モデルは，分析者が要求者から要求を獲得するためのコミュニケーションの基盤となる．しかし，人間の認知限界を超える大規模な問題領域においては，業務モデルを理解すること自体が難しい．そこで，本論文では，錯綜した業務モデルを大局的に理解することを支援するツールを提案する．本ツールのメカニズムは，次のように説明される．業務モデルは，混沌とした状態にある組織である（入力）．本ツールの利用者である要求者と分析者は，組織に対して刺激を与える（入力）．すると，これに反応して，組織の構成要素が局所的に協調することによって，大局的なパターンが発現する（出力）．刺激が変化すれば，反応も変化するので，利用者は，さまざまなパターンを対話的に発現させることができる．このとき，要求者と分析者の間に，コミュニケーションが発生する．したがって，分析者は，パターンの観察と要求者とのコミュニケーションを通して，組織の性質を理解するようになる．本ツールを図書館業務に適用する実験では，図書館システムに対する要求を獲得する上で有効なパターンを探索することができた．このパターンには，局所的な構造の集積の中に隠された大局的な構造を可視化することによって，問題領域の見通しを良くする効果があることを確認した．

# Supporting Requirements Acquisition Using Object-Oriented Models

YASUHIKO SAITO† and SHINICHI HONIDEN††

To support requirements acquisition, an object-oriented business model representing standard knowledge in a problem domain is used as a basis for communication between clients and analysts. However, if the domain is large, it is difficult to understand the business model itself. In this paper, a tool for understanding complex models is proposed. The mechanism of the tool can be explained as follows. A business model is a chaotic organism (input). Users, namely clients and analysts, stimulate the organism (input). The organism responds, that is, a global pattern emerges from local interactions among components of the organism (output). Since a change of stimulus causes a change of response, it is possible to form various patterns interactively. As a result, the tool facilitates communication between the clients and the analysts. Thus, by observing the patterns and by communicating with the clients, the analysts understand the features of the organism. We applied the tool to analysis of a library system and explored the patterns implying the requirements. The patterns are useful in understanding the library domain, because they reveal the global structures underlying the accumulation of the local structures.

## 1. はじめに

ユーザから要求を抽出する際の主要な問題として，問題領域に依存しない要求仕様のモデルが存在しないこと，したがって，分析者が問題領域を理解する負担が大きいことが指摘されている[7]．これは，問題領域が小さい場合や，分析者が問題領域を熟知している場合には，問題にならない．本研究では，大規模な問題領域を大局的に理解することが重要であるという立場から，オブジェクト指向分析[2),11),15)]における要求の獲得を支援するツール[12)]を提案する．

オブジェクト指向分析のプロセスは，要求の獲得のフェーズと要求モデルの定義のフェーズから構成されるが，本研究では，前者に焦点を当てる．このフェーズの目的は，対象とするシステムにおいて「何をオブジェクトとするか，それはどのようなオブジェクトか」を概観し，システムの責任範囲を明らかにすることである．そのためには，システム化の対象とする業

† 情報処理振興事業協会 (IPA)，
(株)アイネスより出向．
Information-technology Promotion Agency, Japan (IPA),
Also with INES Corp.
†† 情報処理振興事業協会 (IPA)，
(株)東芝より出向．
Information-technology Promotion Agency, Japan (IPA),
Also with Toshiba Corp.

務に関する情報を収集し整理する必要がある．これは，ドメイン分析[9]を行うことである．特に，オブジェクト指向ドメイン分析は，オブジェクトを的確に認識する上で有効である[4]．従来は，業務マニュアルや既存のシステムのドキュメントなどから，関連する情報を収集していたが，本研究では，これらの情報源の代替として，業務モデルを位置付ける．要求モデルが計算機システムのモデルであるのに対して，業務モデルは，実世界の業務のモデルである．業務モデルには，過去に開発したいくつかの同じようなシステムの間で，共通性の高い業務知識が反映される．したがって，業務モデルは，問題領域には依存するが，対象システムには依存しない知識を表現するドメインモデルである．

オブジェクト指向モデルに基づく標準的な業務モデルを，要求者と分析者の間のコミュニケーションの基盤として利用することによって，標準からはずれた知識やインフォーマルな情報を捕捉しながら，対象システムにおけるオブジェクトを的確に認識することができる．しかし，問題領域の規模が大きくなると，既存の業務モデルを直ちに利用できないことがある．なぜならば，人間の認知限界を超えるような業務モデルを理解すること自体が難しいからである．そこで，この問題を解決するためのツールを提案する．

本ツールは，業務モデル，および，本ツールの利用者である要求者と分析者によって与えられたプロセス木にしたがって，系統樹を生成する．ここで，プロセス木は，問題領域を分割する手順を表現し，系統樹は，業務の構成要素の布置を表現する．本ツールによって規定されることになる，業務モデル，プロセス木，系統樹の間の関係は，以下の隠喩で説明される．

業務モデル：混沌とした状態にある組織
プロセス木：組織に対して与えられる刺激
系　統　樹：刺激に対する反応

すなわち，刺激に反応して，組織の構成要素が局所的に協調することによって，大局的なパターンが発現する．刺激が変化すれば，反応も変化するので，利用者は，さまざまなパターンを対話的に発現させることができる．このとき，要求者と分析者の間に，コミュニケーションが発生する．したがって，分析者は，パターンの観察と要求者とのコミュニケーションを通して，組織の性質を理解するようになる．

本論文は，次のような構成になっている．第2章では，要求獲得支援ツールのメカニズムを説明する．第3章では，本ツールの基本的な利用方法を説明する．第4章では，本ツールを図書館システムの分析に適用した実験について報告する．第5章では，多変量解析，活性拡散，ドメイン分析との関連を議論する．

## 2. 要求獲得支援ツール

### 2.1 業務モデル

業務は，以下の要素から構成される．

- object：業務の中に存在する，見たり触れたりすることができる物である．
- relationship：一般化—特殊化構造や全体—部分構造などの object 間の対応，および，ある object から他の object への作用である．
- attribute：object を特徴付ける性質である．

これらの要素の間に，以下の二項関係を定義する．

- OR 構造：$OR(x, y)$ は，object $x$ と relationship $y$ の間に関連性があることを示す構造である．
- OA 構造：$OA(x, y)$ は，object $x$ と attribute $y$ の間に関連性があることを示す構造である．

したがって，業務は，各要素を節とし，各二項関係を枝とするネットワークとして表現される．

### 2.2 パターン形成

本ツールは，業務モデル全体をサブモデルに分割するために，業務モデルのネットワークの節をクラスタリングする．次いで，中心—周辺という関係に基づいて，各クラスタの内部を再構造化する．これによって，錯綜した業務モデルの中に隠されたパターンが可視化される．

#### 2.2.1 節のクラスタリング

節をクラスタリングするために，次のような節間のシグナル交換を考える．節 $v$ に隣接する節の集合を $A[v]$ とする．節 $v$ の時刻 $t$ における状態を $P[v]_t$ とする．節 $v$ が時刻 $t$ $(\geq 1)$ に受信するシグナル $s$ の個数は $x \in A[v] \wedge P[x]_{t-1} = s$ を満足する節 $x$ の個数である．節 $v$ が時刻 $t$ までに受信したシグナル $s$ の個数の累計を $R[s, v]_t$ とする．ただし，$n$ 個のクラスタを生成しようとする場合，$P[v]_t \in \{0, 1, 2, \cdots, n\}$，$s \in \{1, 2, \cdots, n\}$ である．ここで，以下の規則にしたがって，$P[v]_t$ を決定する．

$$P[v]_t = \begin{cases} s, \ if\ R[s, v]_t = \max(R[k, v]_t) \text{ を満足する唯一の } s \text{ が存在する．} \\ \qquad \text{ただし，} k \in \{1, 2, \cdots, n\} \text{ である．} \\ 0, \ otherwise \end{cases}$$

この規則では，各節が受信したシグナルの個数に基づ

く多数決によって，次の状態を決定する．多数決では，最大多数が一意に決定しなければ，結論を出せない．そこで，結論を出せない中立的な節 $v$ については，$P[v]_t=0$ とする．

すべての節の暗黙値を，$P[v]_0=0$ とする．terminal node とは，節間のシグナル交換の起点となるいくつかの節であり，初期値として，任意のシグナル $s$ が設定される．すなわち，terminal node $v$ の初期値を，$P[v]_0=s$ とする．このとき，複数の terminal node に対して，同一のシグナルを設定することもできる．ここで，シグナル交換を開始すると，ある時刻 $e$ において，すべての節の状態がそれ以上変化しなくなるので，$P[v]_e$ の値が同じ節の集合をクラスタとする．厳密には，ネットワークが平衡状態に到達しないこともあるが，極めて稀であるため，実用上の問題は生じない．

このメカニズムでは，次のような考え方で業務モデルを分割する．生成されるクラスタは，terminal node の影響力の範囲を示している．すなわち，各クラスタは，terminal node をリーダとする勢力であり，リーダからの指令に服従するメンバのグループである．したがって，平衡状態にあるネットワークから，均衡した状態にある勢力を抽出することができる．

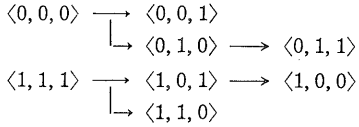
### 2.2.2 系統樹へのマッピング

節のクラスタの内部構造として，次のような系統樹を考える．

1. 系統樹は，深さ $d$ と幅 $w$ によって決定する．
2. 系統樹の節を以下のような項の組で記述する．
   $\langle t_1, t_2, \cdots, t_d \rangle$ $(t_i \in \{0, 1, 2, \cdots, w-1\})$
   節は，$w^d$ 個だけ存在する．
3. origin は，記述中のすべての項の値が等しい節である．origin は，$w$ 個だけ存在する．
4. origin を除く節について，$t_i \neq t_1$ である最大の $i$ を $p$ とすると，$t_p=t_1$ であり，それ以外の項の値が当該節と等しい唯一の節が存在する．この節と当該節の組が親子である．origin を除く節の親は，一意に決定する．このことは，系統樹が無交叉であることを保証する．
5. origin は，第1世代である．親が第 $k$ 世代であるとき，子は，第 $k+1$ 世代である．origin の第1項の値を $a$ とすると，origin の第 $k$ 世代は，その記述中に $d-k+1$ 個の $a$ を含む．

子は，親の主要な特徴を継承する．したがって，ある系譜における下位の世代の節は，上位の世代の節の特徴の一部を継承する．世代は，origin からの距離を表す．世代が下るにしたがって，節の特徴が origin から遠ざかり他の系譜に近づく．

$d=3$，$w=2$ の系統樹を以下に示す．




節のクラスタは，次のように系統樹にマッピングされる．前述のクラスタリングを，

1) OR 構造と OA 構造で構成されるネットワーク
2) OR 構造のみで構成されるネットワーク
3) OA 構造のみで構成されるネットワーク

上で行い，それぞれの分類結果を，ネットワークの節ごとに，$\langle c_1, c_2, c_3 \rangle$ の形で保持する．すなわち，節 $v$ について，1) の結果の $P[v]_e$ を $c_1$ とし，2) の結果の $P[v]_e$ を $c_2$ とし，3) の結果の $P[v]_e$ を $c_3$ とする．この $\langle c_1, c_2, c_3 \rangle$ を，深さが3で，幅がクラスタの個数である系統樹の節であると考えて，ネットワークの節を系統樹上に布置する．これが本ツールの出力であり，業務の大局的な構造を表現する．

この構造は，ネットワークの節をクラスタリングし，各クラスタに属する節を，それがクラスタの中心にあるか／周辺（他のクラスタ）によっているか，という観点から配置したものである．中心からの距離は，系統樹の世代によって示される．したがって，シグナル交換の経路を変更したときに，その節がどのクラスタに属するかによって決定する．これは，指令伝達の経路を変更したときに，そのメンバがどう寝返るか，すなわち，勢力への忠誠の度合を測っているといえる．OR 構造のみで構成されるネットワーク，および，OA 構造のみで構成されるネットワークに経路を変更することによって，各クラスタを object と relationship のみに着目して分割した後，さらに，その中を object と attribute のみに着目して分割することができる．こうして得られた節のグループは，中心からの距離にしたがって配置される．

## 2.3 プロセス木

プロセス木は，業務モデルを段階的にサブモデルに分割していく手順を表現する木構造である．プロセス木 branch の生成規則を以下に示す．

```
⟨branch⟩ ::= ⟨nest⟩ | ⟨key⟩ |
             ⟨branch⟩⟨branch⟩
⟨nest⟩   ::= (⟨key⟩⟨branch⟩)
⟨key⟩    ::= ⟨class⟩.⟨identifier⟩|
```

$\langle key\rangle + \langle key\rangle$

クラスタリングを実行する単位は，トップレベルと nest 中の branch である．このとき，並置された各 branch は，生成するクラスタに対応する．すなわち，branch が key である場合には，key の指定にしたがって初期設定し，branch が nest である場合には，nest 中の key の指定にしたがって初期設定して，クラスタリングを行う．key には，terminal node の identifier と class を指定する．これによって，状態の初期値として，節 identifier に値 class が設定される．terminal node 以外の節には，値 0 が設定される．こうして生成されたクラスタは，系統樹にマッピングされる．

同じクラスタに属するすべての節 $v$ は，$P[v]_e$ の値が同じなので，この値によって，生成されたクラスタが識別される．key が '+' で連結されている場合には，各 key の class によって識別される複数のクラスタが併合される．すなわち，それらのクラスタの和集合としてのクラスタが生成される．

nest は，key の class によって識別されるクラスタ，あるいは，それらを併合したクラスタに属する節のみを抽出して，再度，クラスタリングを実行することを意味する．このとき，中立的な節もクラスタリングの対象として抽出する．

プロセス木の例を以下に示す．

```
(1.v1+1.v2
    1.v1
    (2.v2+3.v4
        1.v5 2.v6))
 2.v3
```

これは，以下の手順を表現している．

1. 第 1 レベル「(1.$v_1$+1.$v_2$ 〜) 2.$v_3$」では，節 $v_1$ と節 $v_2$ に値 1，節 $v_3$ に値 2 を設定してクラスタリングを行い，class 0, 1, 2 のクラスタを生成する．class 0, 1 のクラスタに属する節を抽出する．
2. 第 2 レベル「1.$v_1$ (2.$v_2$+3.$v_4$ 〜)」では，節 $v_1$ に値 1，節 $v_2$ に値 2，節 $v_4$ に値 3 を設定してクラスタリングを行い，class 0, 1, 2, 3 のクラスタを生成する．class 0, 2, 3 のクラスタの和集合に属する節を抽出する．
3. 第 3 レベル「1.$v_5$ 2.$v_6$」では，節 $v_5$ に値 1，節 $v_6$ に値 2 を設定してクラスタリングを行い，class 0, 1, 2 のクラスタを生成する．

## 3. ツールの利用方法

### 3.1 利用条件

本ツールは，大規模な問題領域に対して適用される．目安としては，業務モデルにおける構成要素の数が 100〜1000 程度を想定している．

本ツールを用いることによって，与えられたモデルを業務の実状に適合するように洗練していくことができる．したがって，本ツールは，標準的なモデルを予め構築しておける業務に対して効果的である．

しかし，オブジェクト指向モデルに基づいて，業務が素直に記述できない場合には，本ツールを利用することが難しい．すなわち，その問題領域がオブジェクト指向分析に適していなければならない．

### 3.2 利用手順

本ツールの利用手順を以下に示す．

```
暫定的な業務モデルを定義する
暫定的なプロセス木を定義する
系統樹を生成する
while 系統樹が適切でない, do
    if プロセス木が適切でない, then
            プロセス木を修正する
    if 業務モデルが適切でない, then
            業務モデルを修正する
    系統樹を生成する
```

ここで，本ツールの機能である「系統樹を生成する」以外の各作業には，要求者と分析者の間のコミュニケーションが不可欠である．特に，系統樹が適切でない場合に，プロセス木を修正すべきか，業務モデルを修正すべきかの切り分けは，要求者の知識と業務の実状に大きく依存する難しい問題である．本ツールは，適切な業務モデルが得られるように，コミュニケーションを誘導しているといえる．なぜならば，プロセス木が問題領域を要約し，系統樹が業務の構成要素を分類することによって，業務モデルが大局的に理解しやすくなるからである．

### 3.3 プロセス木と系統樹の解釈

問題領域を理解する上で鍵になると考えられる object を terminal node に指定する．したがって，terminal node を段階的に指定する手順を表現するプロセス木は，問題領域の段階的な分割に関する仮説である．プロセス木の末端部に生成される系統樹を評価することによって，仮説が適切であるかどうかを確認する．最初から完結したプロセス木を与えるのではな

くて，系統樹の形を観察しながら，プロセス木を徐々に成長させていくこともできる．

系統樹は，業務を構成する多数の要素を階層構造に整理する．第1世代の要素は，問題領域における中心概念であり，第2世代の要素は，第1世代の要素に対する周辺概念であり，第3世代の要素は，第2世代の要素に対する周辺概念である．また，系統が分岐することによって，同一の世代においても，複数のグループが形成される．このとき，業務の観点から素直に解釈できる系統樹は，適切であると評価される．

### 3.4 オブジェクトの認識

オブジェクト指向分析では，対象システムにおけるオブジェクトと，対象業務におけるオブジェクト，すなわち，業務モデルにおける object が，素直に対応すると考えられるが，必ずしも1対1に対応するとは限らない．本ツールは，業務モデルを基底にして，対象システムにおけるオブジェクトを認識することを支援する．

対象システムにおけるオブジェクトを的確に認識するためには，トップダウンの視点とボトムアップの視点が必要である．本ツールは，次の理由から，オブジェクトを認識する上で有効である．プロセス木における表現の視点は，トップダウン的であり，業務モデルにおける表現の視点は，ボトムアップ的である．本ツールが規定する両視点の間の相互干渉によって，業務モデルとプロセス木から系統樹が導かれる．したがって，本ツールを用いて探索された適切な系統樹は，トップダウンの視点からもボトムアップの視点からも妥当な，対象業務の構成要素の組織構造である．この組織構造は，対象システムにおけるオブジェクト，および，その機能や属性を認識する上でのヒントになるといえる．

## 4. 図書館システムの分析

### 4.1 業務モデル

図書館業務のモデルに対して，本ツールを適用する実験を行った．本実験で用いた図書館業務のモデルを**図1**に示す．構成要素間を結ぶ線は，OR 構造と OA 構造を示す．たとえば，「図書館が利用者を登録する」という事実を記述する場合には，object “図書館”，object “利用者”，relationship “登録する” について，

*OR*（図書館，登録する）
*OR*（利用者，登録する）

を定義する．“利用者” は，“利用者番号”，“氏名”，“住所”，“性別”，“年齢”，“区分”，“登録年月日” などの attribute によって特徴付けられるので，

*OA*（利用者，利用者番号）
*OA*（利用者，氏名）
*OA*（利用者，住所）
*OA*（利用者，性別）
*OA*（利用者，年齢）
*OA*（利用者，区分）
*OA*（利用者，登録年月日）

を定義する．また，relationship における接尾辞の番号は，同一の動詞による異なる要素間の異なる関係を区別する．たとえば，“貸し出す-1” と “貸し出す-2” では，前者が図書館から利用者への貸し出しを意味し，後者が図書館の間の相互貸借を意味する．

実際の分析では，過去の分析の結果に基づいて作成した標準的なモデルを，業務の実状に適合するように，逐次，変更していくことになる．しかし，本実験では，図書館学の文献[16]を参考にして「机上」で作成したモデルを，途中で変更することなく用いている．すなわち，本実験の目標は，与えられた業務モデルについて，適切なプロセス木と系統樹の対を探索することである．なお，要求がある程度はっきりしている場合にも，本ツールを利用することができるが，本実験では，対象業務を大局的に理解した上で要求を明確にしていく場合を想定している．したがって，探索の前提となる特定の要求条件を考えないものとする．

### 4.2 プロセス木と系統樹

本実験において得られた適切なプロセス木の例を以下に示す．

1. 書店
(2. 図書館＋3. 利用者
1. 利用者
(2. 資料 1. 図書館 2. 資料))

これは，以下の手順を表現している．

1.「1. 書店 (2. 図書館＋3. 利用者 ～)」の実行
   1) “書店” と “図書館” と “利用者” のクラスタが生成される．
   2) “書店” のクラスタが系統樹にマッピングされる．
   3) “図書館” と “利用者” のクラスタの和集合に属する節が抽出される．
2.「1. 利用者 (2. 資料 ～)」の実行
   1) “利用者” と “資料” のクラスタが生成される．
   2) “利用者” のクラスタが系統樹にマッピングさ

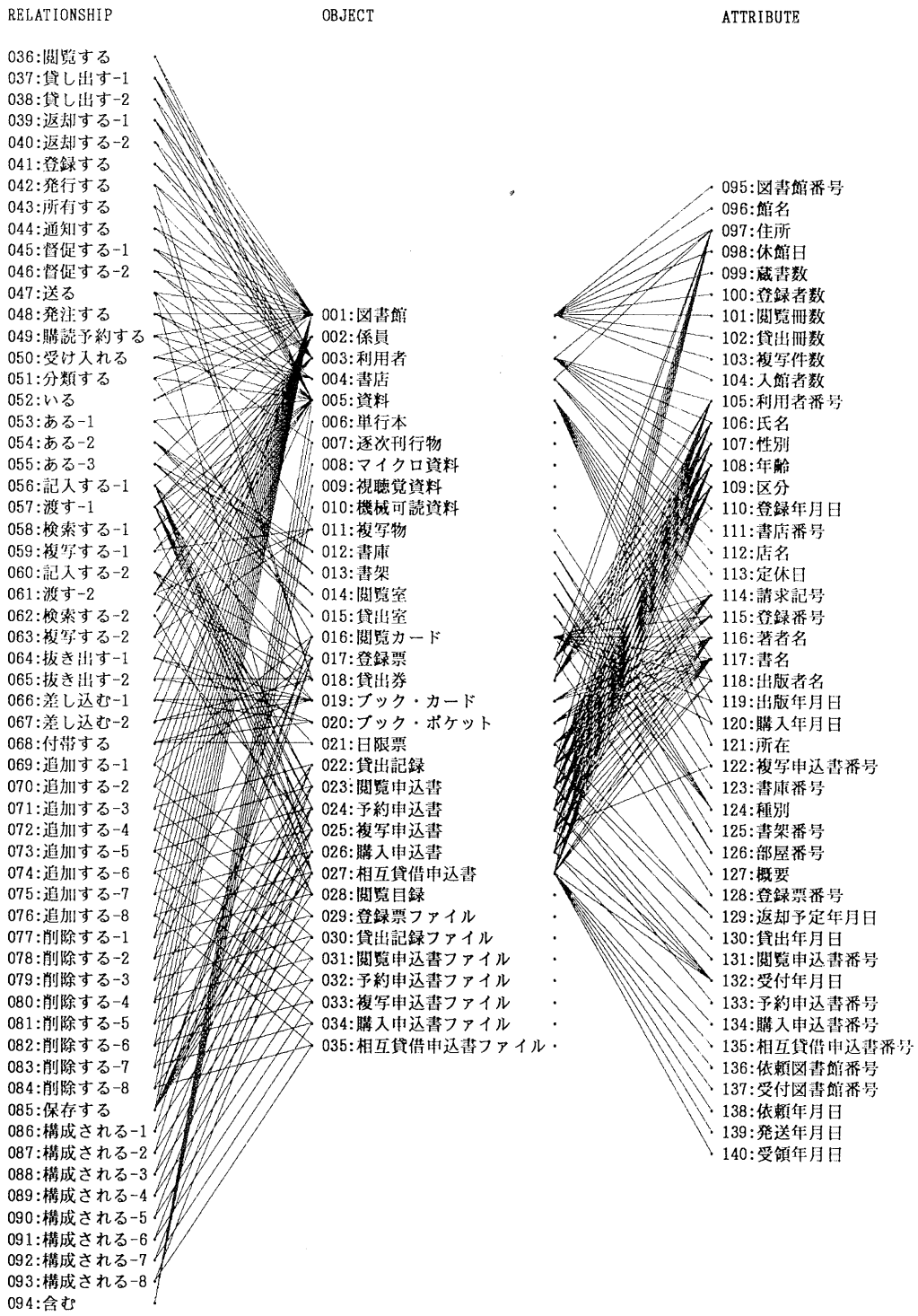



**図 1** 図書館業務のモデル
Fig. 1 A model of the library domain.

```
<1, 1, 1> 003:利用者
<1, 1, 1> 017:登録票
<1, 1, 1> 018:貸出券
<1, 1, 1> 023:閲覧申込書
<1, 1, 1> 024:予約申込書
<1, 1, 1> 026:購入申込書
<1, 0, 1>     097:住所
<1, 0, 1>     105:利用者番号
<1, 0, 1>     106:氏名
<1, 0, 1>     107:性別
<1, 0, 1>     108:年齢
<1, 0, 1>     109:区分
<1, 0, 1>     110:登録年月日
<1, 0, 1>     116:著者名
<1, 0, 1>     117:書名
<1, 0, 1>     128:登録票番号
<1, 0, 1>     130:貸出年月日
<1, 0, 1>     131:閲覧申込書番号
<1, 0, 1>     132:受付年月日
<1, 0, 1>     133:予約申込書番号
<1, 0, 1>     134:購入申込書番号
<1, 0, 1>     135:相互貸借申込書番号
<1, 0, 1>     136:依頼図書館番号
<1, 0, 1>     137:受付図書館番号
<1, 0, 1>     138:依頼年月日
<1, 0, 1>     139:発送年月日
<1, 0, 1>     140:受領年月日
<1, 1, 0> 029:登録票ファイル
<1, 1, 0> 031:閲覧申込書ファイル
<1, 1, 0> 032:予約申込書ファイル
<1, 1, 0> 034:購入申込書ファイル
<1, 1, 0> 041:登録する
<1, 1, 0> 042:発行する
<1, 1, 0> 043:所有する
<1, 1, 0> 044:通知する
<1, 1, 0> 056:記入する-1
<1, 1, 0> 057:渡す-1
<1, 1, 0> 070:追加する-2
<1, 1, 0> 072:追加する-4
<1, 1, 0> 073:追加する-5
<1, 1, 0> 075:追加する-7
<1, 1, 0> 078:削除する-2
<1, 1, 0> 080:削除する-4
<1, 1, 0> 081:削除する-5
<1, 1, 0> 083:削除する-7
<1, 1, 0> 085:保存する
<1, 1, 0> 087:構成される-2
<1, 1, 0> 089:構成される-4
<1, 1, 0> 090:構成される-5
<1, 1, 0> 092:構成される-7
<1, 2, 1> 022:貸出記録
<1, 2, 1> 025:複写申込書
<1, 2, 1> 027:相互貸借申込書
<1, 2, 0>     030:貸出記録ファイル
<1, 2, 0>     033:複写申込書ファイル
<1, 2, 0>     035:相互貸借申込書ファイル
<1, 2, 0>     071:追加する-3
<1, 2, 0>     074:追加する-6
<1, 2, 0>     076:追加する-8
<1, 2, 0>     079:削除する-3
<1, 2, 0>     082:削除する-6
<1, 2, 0>     084:削除する-8
<1, 2, 0>     088:構成される-3
<1, 2, 0>     091:構成される-6
<1, 2, 0>     093:構成される-8
```

図 2 “利用者”の系統樹
Fig. 2 A genealogical tree of “user”.

れる（図 2）.

3) “資料”のクラスタに属する節が抽出される.

3.「1. 図書館 2. 資料」の実行

1) “図書館”と“資料”のクラスタが生成される.

2) “図書館”のクラスタが系統樹にマッピングされる（図 3）.

3) “資料”のクラスタが系統樹にマッピングされる（図 4）.

このプロセス木の例によれば，この問題領域で鍵になる object は，“書店”，“利用者”，“図書館”，“資料”である．したがって，対象システムにおいても，これらが重要なオブジェクトになると考えられる．その場合に，たとえば，“書店”をシステム化の範囲に含めない，と判断することもできる．

次に，本実験において得られた系統樹の例に基づいて，図書館業務を理解する上でのポイントを示す．

1. “利用者”の〈1, 1, 1〉に集まる要素は，“利用者”に直接的に関与する書式である．これらが“利用者”の核となる．一方，“利用者”の〈1, 2, 1〉に集まる要素は，これらの書式よりも，やや“資料”よりであるといえる．
2. “資料”の〈2, 2, 2〉に集まる要素は，“資料”に付帯し，物理的に“資料”を構成する．これらが“資料”の核となる．
3. “利用者”の〈1, 0, 1〉には，“利用者”の諸属性が集まる．このうち，“住所”，“利用者番号”，“氏名”，“性別”，“年齢”，“区分”，“登録年月日”は，もとより“利用者”を特徴付けるものであ

```
<1, 1, 1> 001:図書館
<1, 0, 1>     095:図書館番号
<1, 0, 1>     096:館名
<1, 0, 1>     098:休館日
<1, 0, 1>     099:蔵書数
<1, 0, 1>     100:登録者数
<1, 0, 1>     101:閲覧冊数
<1, 0, 1>     102:貸出冊数
<1, 0, 1>     103:複写件数
<1, 0, 1>     104:入館者数
<1, 0, 0>         123:書庫番号
<1, 0, 0>         124:種別
<1, 0, 0>         126:部屋番号
<1, 1, 0>     012:書庫
<1, 1, 0>     014:閲覧室
<1, 1, 0>     015:貸出室
<1, 1, 0>     049:購読予約する
<1, 1, 0>     052:いる
<1, 1, 0>     054:ある-2
<1, 1, 0>     055:ある-3
```

図 3 “図書館”の系統樹
Fig. 3 A genealogical tree of “library”.

るが，それ以外は，関連するオブジェクトから移入されたものである．

4. “資料” の 〈2, 0, 2〉 には，“資料” の諸属性が集まる．このうち，“返却予定年月日” は，関連するオブジェクトから移入されたものである．
5. 他のオブジェクトから “利用者” および “資料” に移入された属性は，“利用者” が “資料” にどのように関わるかについての記録である．これらの属性は，概ね，“利用者” よりである．“著者名” と “書名” は，もとより “資料” を特徴付けるものであるが，“利用者” と “資料” の関わりにおいて本質的な情報であり，これらが “利用者” よりであることは，妥当であるといえる．
6. “利用者” の 〈1, 2, 0〉 の要素は，親である 〈1, 2, 1〉 の要素の下位に位置する．また，“資料” の 〈2, 1, 0〉 の要素は，親である 〈2, 1, 2〉 の要素の下位に位置する．しかし，複数の ‘0’ を含む節の要素は，当該クラスタに属していても，内部構造における位置付けに関して，あまり当てにならない．たとえば，“図書館” の 〈1, 0, 0〉 の要素は，親である 〈1, 0, 1〉 の要素の下位に位置するとは考えにくい．

```
< 2,  2,  2 >  005 :資料
< 2,  2,  2 >  019 :ブック・カード
< 2,  2,  2 >  020 :ブック・ポケット
< 2,  2,  2 >  021 :日限票
< 2,  0,  2 >      114 :請求記号
< 2,  0,  2 >      115 :登録番号
< 2,  0,  2 >      118 :出版者名
< 2,  0,  2 >      119 :出版年月日
< 2,  0,  2 >      120 :購入年月日
< 2,  0,  2 >      121 :所在
< 2,  0,  2 >      127 :概要
< 2,  0,  2 >      129 :返却予定年月日
< 2,  0,  0 >          122 :複写申込書番号
< 2,  0,  0 >          125.:書架番号
< 2,  1,  2 >      016 :閲覧カード
< 2,  1,  0 >          028 :閲覧目録
< 2,  1,  0 >          051 :分類する
< 2,  1,  0 >          069 :追加する-1
< 2,  1,  0 >          077 :削除する-1
< 2,  1,  0 >          086 :構成される-1
< 2,  2,  0 >      002 :係員
< 2,  2,  0 >      006 :単行本
< 2,  2,  0 >      008 :マイクロ資料
< 2,  2,  0 >      009 :視聴覚資料
< 2,  2,  0 >      010 :機械可読資料
< 2,  2,  0 >      011 :複写物
< 2,  2,  0 >      013 :書架
< 2,  2,  0 >      036 :閲覧する
< 2,  2,  0 >      047 :送る
< 2,  2,  0 >      053 :ある-1
< 2,  2,  0 >      058 :検索する-1
< 2,  2,  0 >      059 :複写する-1
< 2,  2,  0 >      060 :記入する-2
< 2,  2,  0 >      061 :渡す-2
< 2,  2,  0 >      062 :検索する-2
< 2,  2,  0 >      063 :複写する-2
< 2,  2,  0 >      064 :抜き出す-1
< 2,  2,  0 >      065 :抜き出す-2
< 2,  2,  0 >      066 :差し込む-1
< 2,  2,  0 >      067 :差し込む-2
< 2,  2,  0 >      068 :付帯する
< 2,  2,  0 >      094 :含む
```

**図 4** “資料” の系統樹
Fig. 4 A genealogical tree of “material”.

系統樹には，局所的な構造の集積の中に隠された大局的な構造を可視化することによって，問題領域の見通しを良くする効果があるといえる．図 1 と図 2-図 4 を対比させれば，この効果は明らかである．

## 5. 関 連 研 究

### 5.1 多変量解析

クラスタ分析[1]，因子分析[14]，多次元尺度法[5] などは，多数の観測対象によって表現される世界を大局的に理解するために，観測対象をグルーピングする多変量解析の手法である．そこで，業務の構成要素を，システム化の対象とする世界，すなわち，問題領域における観測対象として捉えると，これらの手法は，問題領域を理解するためのツールになる．たとえば，図書館業務のモデルにおいて，節間の最短経路を距離とし，最長距離法を用いてクラスタ分析を行った結果を**図 5** に示す．クラスタ分析では，ボトムアップの視点のみからデータを整理する．因子分析や多次元尺度法も，その点では同様である．これに対して，本ツールでは，トップダウンとボトムアップの視点からデータを整理する．加えて，

1) 中心概念—周辺概念という関係を表現する
2) プロセス木と相補的である

という特徴がある．これらの特徴は，オブジェクトを認識する上で有利であるといえる．

多変量解析が観測対象間の距離に基づいて組織化を行う手法であるのに対して，観測対象間の関係に基づいて組織化を行う手法に，ISM (Interpretive Structural Modeling)[18] や部分的関係からのクラスタリング[13]などがある．前者は順序関係，後者は類義関係と反義関係を扱う．しかし，業務のモデル化の手法としては，どちらも適用範囲が限られている．本ツールは，オブジェクト指向モデルに基づくネットワーク構造を対象とするので，適用範囲が広い．

### 5.2 活 性 拡 散

本ツールにおけるクラスタリングは，離散的な活性

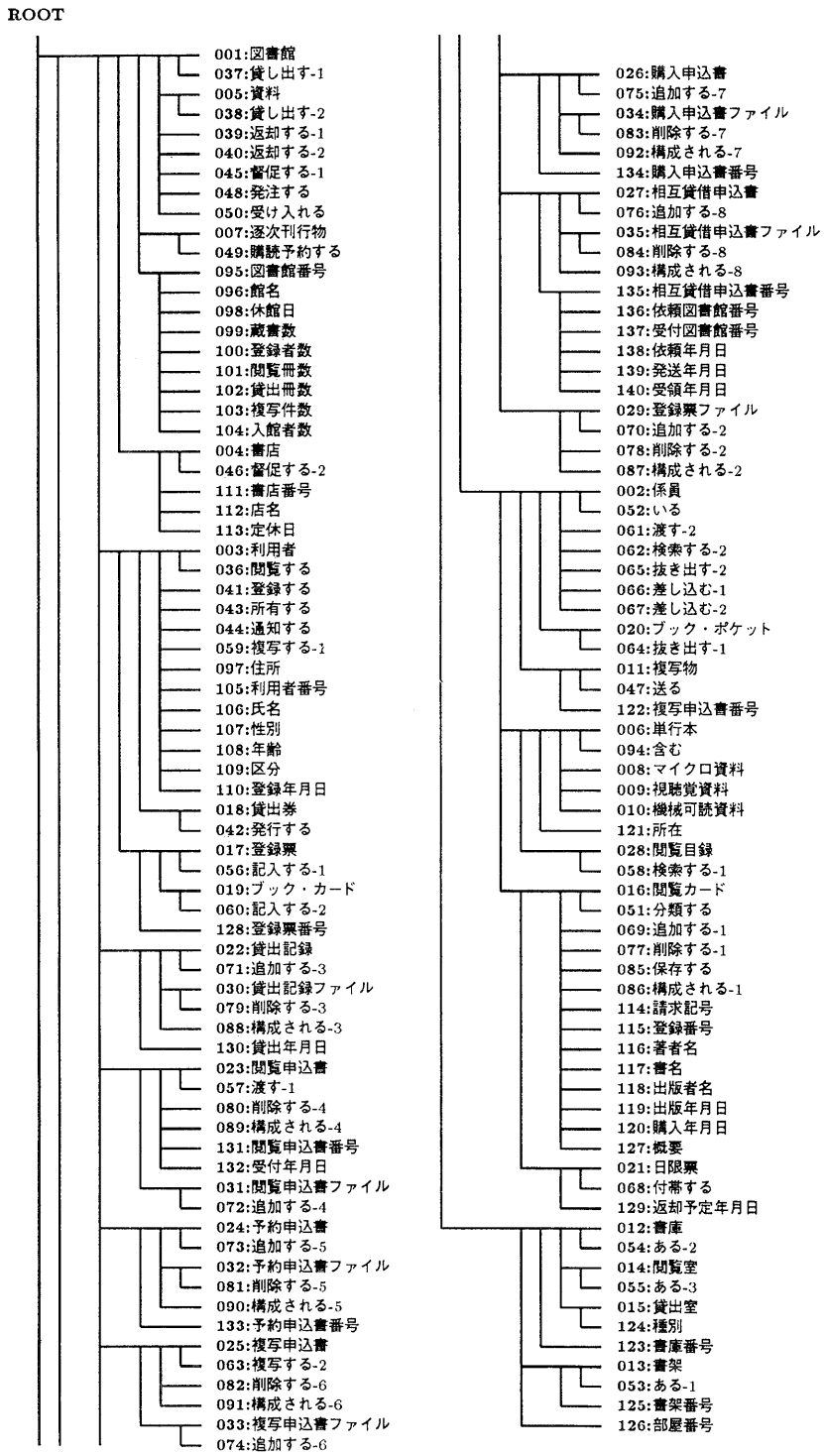



**図 5** 図書館業務の樹形図
Fig. 5 A dendrogram of the library domain.

拡散[3]として捉えられる．活性拡散は，意味記憶のモデルである．概念が節に対応し，概念間の連想関係が枝に対応するネットワークにおいて，ある概念について処理が行われると，対応する節が活性化され，枝を通じて，その活性化が他の節に拡散していく．このとき，すべての節が活性化されてしまわないように，拡散の過程で，活性度の減衰が起こる．D. Waltz と J. Pollack は，ネットワークの枝に興奮性／抑制性の区別を設け，抑制性の枝のために活性度の減衰が起こるモデルを提案した[17]．このモデルは，自然言語の文の意味解釈における文脈効果を説明することができる．すなわち，文をネットワークとして表現し，文脈を節の活性度の初期パターンとして表現すると，活性拡散によって，多義的な文の解釈が文脈に依存して決定する．これに対して，本ツールでは，ネットワークの枝に興奮性／抑制性の区別を設けずに，複数の種類のシグナルを伝達する．活性度に相当するものは，分類のクラスに対応する離散的な値をとる．また，terminal node を指定することによって，文脈が与えられ，文脈に依存して，クラスタリングが動的に変化する．

### 5.3 ドメイン分析

ドメイン分析は，プロダクトの系統的で効果的な再利用のための技術としても注目されている[10]．ドメイン分析によって得られたドメインモデルは，再利用可能なプロダクトの管理体系，すなわち，ライブラリに反映される．これに対して，本研究における業務モデルの目的は，要求獲得の支援という観点から，要求者と分析者の間のコミュニケーションの基盤を提供することである．そこで，オブジェクト指向モデルに基づく業務モデルによって，モデルと実世界とを素直に対応させている．また，時間的な変動が小さく，異なる組織間で共通性が高い部分をモデル化の対象とし，不足する情報は，要求者とのコミュニケーションを通じて捕捉することを前提としている．

継承やメッセージ通信などのオブジェクト指向プログラミングの概念を，積極的に取り入れたモデルは，要求モデルへの移行が容易であるため，ドメインモデルの構築がライブラリの構築に直結する[6]．したがって，プロダクトの再利用という面で有利である．反面，モデルの形式が複雑になり過ぎて，理解や変更のためのコストが大きくなるので，問題領域によっては，要求者との円滑なコミュニケーションが妨げられることがある．また，計算機に不慣れな業務担当者が，オブジェクト指向プログラミングの概念を理解することは期待できない．本研究における業務モデルは，簡略化したオブジェクト指向モデルであり，これによって，コミュニケーションの問題を回避する．

R. Prieto-Diaz と P. Freeman は，プロダクトの再利用を目的とするドメインモデルの形式として，ファセット分類のスキーマを提案した[8]．ファセットとは，問題領域に関連する用語の集合である．いくつかのファセットのそれぞれに属する用語を組み合わせて，プロダクトの特徴を記述することによって，プロダクトを体系的に管理する．しかし，ファセット分類では，トップダウンの視点から問題領域を整理するので，部分から全体を漸進的に理解していくことが難しい．本研究における系統樹は，分類を表現している点で，ファセット分類と同じである．しかし，系統樹は，業務モデルやプロセス木と相補的に，トップダウンとボトムアップの視点から問題領域を整理することによって，問題領域の漸進的な理解を支援する．

## 6. おわりに

ソフトウェアの要求仕様を獲得するためのツールの枠組を提案した．本ツールは，要求者と分析者の間のコミュニケーションを通じて，問題領域を大局的に理解し，対象システムにおけるオブジェクトを的確に認識することを支援する．本ツールを図書館システムの分析に適用する実験では，比較的良好な結果が得られた．

本論文では，ドメイン分析の結果であるドメインモデルを利用するという観点から述べてきた．しかし，ドメインモデルを構築するためには，まず，問題領域を理解する必要があるので，ドメインモデルを構築する際にも，本ツールを利用することができる．今後の課題は，過去の開発の事例に基づく標準的な業務モデルの構築，あるいは，プロダクトの再利用を目的とするドメイン分析などのために本ツールを利用する方法論を検討することである．

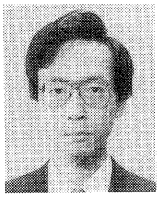
**謝辞** 本研究は，産業科学技術研究開発制度「新ソフトウェア構造化モデルの研究開発」の一環として情報処理振興事業協会（IPA）が新エネルギー・産業技術総合開発機構から委託をうけて実施したものである．また，有益な助言をいただいた査読者の方々に感謝する．

## 参考文献

1) Anderberg, M. R.: *Cluster Analysis for Applications*, Academic Press (1973).

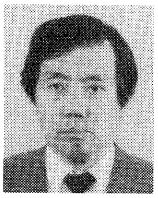
2) Coad, P. and Yourdon, E.: *Object-Oriented Analysis*, Yourdon Press (1990).
3) Collins, A. M. and Loftus, E. F.: A Spreading Activation Theory of Semantic Processing, *Psychological Review*, Vol. 82, pp. 407-428 (1975).
4) Firesmith, D. G.: *Object-Oriented Requirements Analysis and Logical Design*, John Wiley (1993).
5) 林知己夫, 鮑戸 弘(編): 多次元尺度解析法, サイエンス社 (1976).
6) Johnson, R. E. and Foote, B.: Designing Reusable Classes, *Journal of Object-Oriented Programming*, Vol. 1, No. 2, pp. 22-35 (1988).
7) 折原良平, 荒木 大, 西村一彦: 仕様獲得 vs. 知識獲得, 情報処理, Vol. 33, No. 6, pp. 605-611 (1992).
8) Prieto-Diaz, R. and Freeman, P.: Classifying Software for Reusability, *IEEE Software*, Vol. 4, No. 1, pp. 6-16 (1987).
9) Prieto-Diaz, R. and Arango, G.: *Domain Analysis and Software Systems Modeling*, IEEE Computer Society Press (1991).
10) Prieto-Diaz, R.: Status Report: Software Reusability, *IEEE Software*, Vol. 10, No. 3, pp. 61-66 (1993).
11) Rumbaugh, J. et al.: *Object-Oriented Modeling and Design*, Prentice Hall (1991).
12) 斉藤康彦, 本位田真一: オブジェクト指向モデルを用いた業務分析に基づく要求獲得の試み, 情報処理学会研究報告, 93-SE-95-8 (1993).
13) 斉藤康彦, 東条 敏, 古宮誠一: 部分的関係からのクラスタリング: 直観的データ解析の枠組, 情報処理学会論文誌, Vol. 34, No. 11, pp. 2354-2365 (1993).
14) 芝 祐順: 因子分析法, 東京大学出版会 (1979).
15) Shlaer, S. and Mellor, S. J.: *Object-Oriented Systems Analysis*, Prentice Hall (1988).
16) 図書館情報学ハンドブック編集委員会 (編): 図書館情報学ハンドブック, 丸善 (1988).
17) Waltz, D. L. and Pollack, J. B.: Massively Parallel Parsing: A Strongly Interactive Model of Natural Language Interpretation, *Cognitive Science*, Vol. 9, pp. 51-74 (1985).
18) Warfield, J. N.: *Societal Systems: Planning, Policy and Complexity*, John Wiley (1976).

(平成6年4月 4 日受付)
(平成6年7月14日採録)



**斉藤 康彦** (正会員)

1961年生. 1984年早稲田大学教育学部卒業. 同年, (株)協栄計算センター (現, (株)アイネス) 入社. 1991年より情報処理振興事業協会 (IPA) に出向. 新ソフトウェア構造化モデル研究本部に所属. 要求分析法の研究に従事. 人工知能学会会員.



**本位田真一** (正会員)

1953年生. 1976年早稲田大学理工学部電気工学科卒業. 1978年同大学院理工学研究科電気工学専攻修士課程修了. 工学博士. 同年(株)東芝入社. 現在, 同社研究開発センターシステム・ソフトウェア生産技術研究所に所属. 1989年より早稲田大学非常勤講師を兼任. 1991年東京工業大学大学院非常勤講師. 主として, ソフトウェア工学, 人工知能の研究に従事. ソフトウェアの基礎理論に興味を持つ. 1986年情報処理学会論文賞受賞. 著訳書「ソフトウェア開発のためのプロトタイピング・ツール」(共著),「KE 養成講座(2)エキスパートシステム基礎技術」(共著),「オブジェクト指向システム分析」(共訳) など. 日本ソフトウェア科学会理事. 日本ソフトウェア科学会, 人工知能学会, IEEE, AAAI 各会員.