ソフトウェア工学 136−29
（2002. 3. 8）

対象の一意特定識別記述可能なオブジェクト指向記述モデルの試み

畠山 正行†

オブジェクト指向（OO）に基づくモデリング方法は明示的に形式化されていないことが多く，方法論一般として習得或いは確立し難い．そこでOOモデリング方法を近似的に形式化した代替方法としてOO記述モデルを構築した。それは対象世界のモデル化すべき記述構成要素の種類とその構成を二つの対照表という形式に纏め、その両表の相互参照機構を組み込むことで自然にOOモデリングが実行されるという方法である。OO記述モデル一覧表が出す明示的・直截な指示に沿ってモデリング作業を進めて行けば、任意の詳細度でOOモデル記述が自然に出来上がる．記述実験結果は完備性と一意特定識別性を（証明は未だであるが、）充分に推定させ、簡潔で記述力の高いOOモデリング方法であることが分かった．今後、記述力の向上と検証を進める。

Object-oriented, Uniquely Identifiable Description Model of Target World

MASAYUKI HATAKEYAMA†

Abstract: In the present study, we will propose a kind of the "Object-oriented (hereafter OO) description model" that is formulated formally based on the OO modeling paradigm. This description model is a kind of the modeling framework, and is presented with the format of two tables, one is the kind list of the OO description elements, and the other is the constitution list of the each description element. The users can only make use of the description model tables and constitute the descriptions for the target world by mutually refering both tables each other. Since the serching processes and picking up the constitution elements are formally and explicitly designated by both tables, the users can easily find the descrioption elements, and describe the target model. We can conclude that this OO description model scheme is simple and useful, though the completeness and the uniquely identifiability in this OO description model have not yet been verified.

1. はじめに

本論文の狙いはオブジェクト指向（以下OOと略記）に基づくモデリング作業を明示的な形式を持たせた形で構成することにより，特に初期段階においては簡潔にモデリング作業を始めることが出来，最後には任意の詳細度にまで達することが出来る方法の提案の試みである．本論文では分析段階のモデリング作業を対象にする．また、現時点ではモデリング対象の世界を磯田の分類にいう真性実世界[1]に限定するものとする．

さて、モデリング方法と似て非なるものに、モデリングした結果を表現或いは記述するための表記法や言語がある。現状でも数多くの表記法[7),11)]や言語が提案されて利用されている．確かに表記法や言語の内部にモデリング方法の一部が含まれている*のであるが，モデリングそのものを「直裁かつ詳細に」その全容を提示したものは殆ど無い様である．その理由を推測するに，モデリングはその知的作業の原則・ルール・枠組みだけが形式化されて示されるだけのことが多く，作業が一意的形式では捉え難く，個人差も大きく，明示的な形式化・定式化として一般に認知された客観的な方法論を確立し難しいからであろう．また，各専門分野毎の特殊な知識やモデリングの考え方等が存在し，それが一般的なモデリング方法を捉え難くしているという事情もある．

ところで一方，OOは優れて「強い構造化手法」であり，モデリングにおいてもその特性は強く残せる．そこでOOモデリング作業自体を一般的な形で専門家が予め十分に行い，その作業結果として，記述すべき要素や各要素の記述構成等を抽象的かつ構造的表記法を用いて示し，ユーザはその構造的表記法に従ったモ

† 茨城大学工学部情報工学科
Department of Computer & Information Sciences, Faculty of Engineering, Ibaraki University

* 勿論，表記法や言語はその内部にOOモデルを持ち，モデリング方法を非明示的（Implicit）に内蔵していることは確かであるが，それを初心者や言語や表記法の非専門家に対しても気付いて当然である、あるいは気付け，というのは無理であろう．

デリングを行えばよい，とする方式を見出した．

それは，抽出される可能性のある構成要素を一覧表の形で示して「何をモデリングすべきか？」に応え，その要素の構成要素を一覧表の形で示して「どの様な構成要素が必要か？」に応え，両一覧表を相互参照で行き来する作業を繰り返すことで，OO モデリングが「自然に出来上がる」仕組みである．更に任意の詳細度にまで達する方法も両リスト内に予め埋め込んだ．つまり利用者の為すべき作業は二枚の一覧表内部に十分に詳細化されたので，個々の作業は OO を概念として理解している人ならば誰にでも分かる様に単純明快かつ直裁に明示化された作業手順の集約となる様に仕組んだ．

第 2 章で OO 記述モデルの仕組を導入し，離散モデリング及び再帰展開モデリングを提案し，最後に OO 記述モデルを述べる．第 3 章は検証・比較・評価と考察を，第 4 章で結論と課題を述べる．

2. OO 記述モデル

2.1 OO モデリングから OO 記述モデルへ

対象世界の再現シミュレーションを実現する OO モデリングを実現する条件は，対象世界の「何を（モデル化抽出すべき要素）」と「如何に」について一意特定識別 (Uniquely Identify) 出来ることであり，必要なだけ充分に対象を捕捉した分析記述になっていると判断出来る基準であると我々は考える．その「何を」の内容である対象世界の構成要素の「種類」とその各々の「要素自体の内部構成」を規定するのが OO 記述モデルである．

従って対象を OO パラダイムで必要なだけ充分に捉えるためにこの OO 記述モデルの精密な設計が予め十二分に必要である．そこで OO に基づいて予めモデリングを十二分に行い，各記述要素の「種類」と「構成」を予めほぼ充分に「意味的なリスト」として一覧表（リスト）にしてしまおう，またそれが可能である，というのが我々の考えである．つまり，OO モデリング方法の近似的代替としてある種の「OO 記述モデル」を工夫し，形式的に定式化して一覧表形式に凝縮させ，OO モデリングに置き換える訳である．

「何を」に対して「どの様に・如何に」記述すべきかは言語の責任と役割であるので，本論文では対象にしない.「何を」という部分と「如何に」という部分を分離し,「何を」という記述要素の種類と構成を言語に拘束されない形でリストで定義してしまおうというのが本論文の狙いである．しかし「何を」をリストアップするにも何か言語が要るのであるが，それについては自然日本語（以下，NJ と略記する）を充てることでその影響を最小限に留める，という方針とする．

よく知られている様に NJ は母国語でもあり，かつ全ての人工言語よりも一般的な意味では記述力・表現力があるので，対象世界のモデル化と記述には最も適している．勿論，欠点もよく知られており，その欠点（曖昧，OO に向かない）を克服して用いれば，最も適した言語であることは明らかである．この欠点を我々は OO という強い構造化モデリングパラダイムを導入することで自然言語の特性を充分に残したままで解決しようとしている．

2.2 OO 記述モデルの導入形成と完成条件

本論文ではモデリング方法を近似的に等価な OO 記述モデルで置き換えるため，まず予め OO モデリングパラダイムに沿って十分にモデル化を行う．そのためモデリングと記述という「作業過程」を，予め作成済みの記述構成要素の基本の全「種類」と，各構成要素の全ての「構成，組合わせ，要素間諸関係」をリスト表として挙げておき，両リスト間を相互参照しながら往復し，自身のモデル化に必要と判断される記述要素種類と構成を抽出し展開した記述にすれば,「結果として自然に」ユーザは OO に基づくモデリングとその記述を実行している様になるような戦略的な仕掛けを内蔵する設計とし，記述モデル表の形式に纏めた．

OO 記述モデルの構成は以下の様である．

(1) 記述要素**種類**の全リスト

(2) 各記述要素の基本**構成**全リスト

対象世界からモデル化・抽出すべき要素の種類は「種類リスト」から選択し，各要素種類の各々の具体的な記述要素は「構成リスト」から見出すという仕組みの設計にした⋆．全ての要素は自然日本語を前提にした⋆⋆一般化された用語で定義されている．即ち意味モデルとして構築され，要素意味も関係意味も持たせた要素であるので，その一般的な用語表現からモデル化抽出すべき要素を容易・的確に指すことが可能かつ十分容易になる．

⋆ リストは基本的，一般的なものに限定したので全ての分野で完全であるわけはない．本論文ではこの制約下で議論する．

⋆⋆ 本論文では自然日本語（以降 NJ と略）を前提とした．それは OO 記述モデルを「意味モデル」として設計したからである．それにより各ユーザはその意味を説明無しに了解・理解することが出来,「NJ 表現が意味するところの」種類とその記述構成要素を各対象世界から抽出することを確実かつ簡潔に可能にする．また，NJ であれば豊富な語彙とそれに付随する要素意味と関係意味を全ユーザの共有知識として前提に出来る．

上記の記述モデルにより充分な (完成した) オブジェクト指向記述を得るためには，両表に対して次の二つの条件を満たす改訂が更に必要である．それは，

1．記述要素の種類と構成の記述完備性
2．単語単位の一意特定識別の実現
（略して単一特識）

1．の記述完備性に関しては第３章で検証・考察し，単一特識は次節でその必要性と実現のアイデアを述べる．

2.3 記述モデルの単一特識の必要性と実現

OO 記述がより充分（完全に近い）であるためには，各記述構成要素に対する単語単位での一意特定識別の実現が必要である．それは各記述構成要素に対して各々個々にファセット番号を付することによってかなりな部分が満たされるが，それは記述の可読性・了解性を著しく落とす可能性がある．従って別の方法が必要である．そのアイデアとしては以下の方法により，記述モデルの単一特識を実現する．

1．後述の記述構成要素種類の集まりを明示する表１の個々のファセット番号毎に拡張された NJ 記述の特定文型を定義し，その特定文型から単語毎にその構成要素が指定されるべき仕様を特定する．これは OONJ[3]の一意特定識別文定義の拡張である．

2．ただし NJ の語順や意味規則は変更はせずに文型を定義する．各記述要素の構成要素の並びの順序 (後述の表２) は，そのまま特定文型の NJ 記述の順になる様に配置・並べておく．

3．各特定文型は単文とする．主語は自明なので省略可であり，形容詞は名詞とカプセル化し，副詞は動詞とカプセル化して記述する．単一特識のための特定文型は基本文型の特別な拡張文として定義する．

2.4 OO モデリングの詳細化

本論文では以下の三つの詳細化☆ （と統合化）モデリングとその OO 記述モデル化を採用した．

1．抽象化・具象化に基づく詳細化
2．離散化・分解と修復・再構成に基づく詳細化
3．再帰的な OO モデリングに基づく詳細化

特に指定したい場合には，

4．汎化・特化（上記１．の特別な場合）
5．集約・被集約（上記２．の特別な場合）

を用い，その他の特別な詳細化等には

6．その他の詳細化・統合化

上記の１．は対象世界の基本のモデリング作業そのものであり，OO 記述モデル表自体の導出・生成作業そのものである．そこで本章では２．と３．の項目を取り上げ，次の２つの節で提案する．これらの章で提案する OO モデリングの結果も OO 記述モデルに組み込まれる．

2.5 離散化・分解によるモデル化と修復・再構成

離散化や分解の対象は”モノ”と振舞いであるが，連続的な”モノ”の場合の離散化とは，”モノ”の離散化（これはドメインユーザ専決事項であるので議論しない）とそれに対応する 2.6.x 空間関連属性と，2.5.x 時間関連属性の二つの属性の離散化に帰着する．分解とは以前に集約に依って組み上げられた”モノ”を再び分けてそれぞれをモデル化単位に設定することを指す．

振舞いは”モノ”や属性に比べて，比較的に「詳細化」されることが少ない．その離散化とそれに伴う詳細化・具象化・抽象化，その修復・再構成をも扱う．振舞いの集約或いは汎化の階層構造は，振舞いを離散化された最小モデル化単位を設定し，OO 記述モデル表を利用して再帰的モデリングを形式的に行って直接的に任意の詳細度で記述できるようになる．最小単位まで分解された振舞いを再構成した記述を「OO 構造化振舞い記述」,「振舞いの OO 記述」と呼ぶことにする．

2.6 再帰展開モデリング

再帰展開モデリングとは，本論文では当初のモデリング後に記述要素を個々に更に何らかの意味で詳細化記述する必要が見出せたときに，**表 2-2** に沿って行うモデリングのことを指す．表１において予め拡張された全モデル化単位 (1.x，特に 1.2.x～1.6.x) を選んで改めて表１，表２の記述モデル表を用いた OO モデル化を行う．そして新たに OO モデル化したモデル化単位と元の記述要素とを表１のファセット番号 4.x の記述間相互関連を用いて繋ぐ，という方法である．

この方法は相互参照対を成す両表の記述モデルに OO 再帰展開モデリングの方法表 2-2 を組み込むこと

☆ **抽象化・具象化分析**とはターゲットのモデル化単位自体の内部を詳細・具象的な分析で集約 (集合を含む)・被集約，汎化・特化，その他関連する複数要素と見なせる記述要素を単独化し，または記述要素及びその数の追加，記述内容の増補・改訂を行った後に，相互関連の修復・再構成でモデル化を行うことであり，
離散モデル化とは，連続なモデル化単位を仮想的に切り離して離散化してモデル化単位として改めて設定し，離散化を修復するため相互関連を用いて修復することで再構成すると共に，前節で述べた OO 記述モデル表を当て填めて属性・振舞い・記述間関連等の諸々の付置記述要素☆☆を加える事であり，
再帰展開モデリングとは当初は付置記述要素とされていたものを改めてモデル化単位に指定し直し，前節で述べた OO 記述モデル表を全面的に当て填めて各付置記述要素の詳細設定を行うことである．

で，多様な対象世界に対する必要なかつ充分に詳細なOOモデル記述作業を可能にした．

2.7 OO記述モデル

前節までに提案したOO記述モデルを纏めて**表1**，**表2**に示す．表中に付された注が多いので纏めて一つの**表3**の形にして掲げた．また，OO記述モデルに予め既に埋め込まれているOOモデリングの標準的な手順を参考までに**表4**の様に示す．

3. 検証・比較・評価と考察

紙幅の関係で記述例の提示は省略した．

3.1 OO記述モデルの検証

OO記述モデルのモデリング機構

表1と表2は相互参照によってその定義を完成させると共に，ファセット番号4.0に示した記述間相互関連記述によって，再帰展開を含む詳細化を実現している．これによりユーザが必要と考える任意の詳細度や階層構造の記述を行うことが出来る機構に設計した．その結果一般的な対象世界のモデリング機構としてはOOモデリング方法を充分可能にすると評価した．個々の特殊な分野の特殊なモデル化以外ならば，モデリング機構としてこれ以上これ以外のものを見出すことが現時点では出来なかった．更にこの記述モデルは**表5**に挙げた理由から確かにOOモデリングの充分な代替となり得る充分な構成を持つことが論証された．これは次節で紹介する記述実験からも裏付けられた．

OO記述モデルの構成要素の完備性

本OO記述モデルは，必要とされる要素（種類，構成）はOOパラダイムに沿ってほぼ全て明示化・形式化して一覧表の形に組み込み，再帰展開モデリングの手順も含めて形式や方法の決まった詳細化作業に帰着された点がもっとも大きな特徴である．それは通常，各モデル化単位内部の構成要素に含まれず，別途記述される場合も多い要素も敢えて全て内部記述要素に含めたからである．具体的には相互関連，制約，時間及び空間関連を属性として記述させ，記述間相互関連を内部記述要素として新規に設けたからである．その結果，両表に登録された全ての記述要素と構成が形式的な議論の俎上に載るという大きな特長が得られた．この特長は3.4節の議論に連なる．

ただし両表のリストが充分或いは見落としが無いか（完全）については，我々の検討では見当たらなかったが，追加の可能性は残していると考える．以上から分かるのは，形式的な完備性は主張するには到達しなかったが，結果的には完備性が実現できていると言える結果が得られる筈であると推定される．少なくとも構成方針としては，表1にある各項目の抽象的な意味の解釈によってカバーされるという点と，再帰展開モデリングの方法の2つに依って実質的には完備性が得られる構成になっていると現時点では評価している．

表5　OO記述モデルの検証と特性の議論

*

OOモデリングの代替としての検証
1. 両記述モデル一覧表のモデリング方法自体がOOパラダイムに従って導出されていること．
2. 全ての記述要素種類に亘って「離散的な単位」，即ち，離散(化)モデルを保証していること．
3. その記述構成要素種類がOOの三種の構成要素("モノ"，属性，振舞い)を全て持つこと．
OO記述モデルの特性
4. OO記述モデルが「NJ記述の意味モデル」であることにより，モデル化に必要な記述要素を容易に想起・抽出を可能にしたこと．
5. 表形式で明示しているので捕捉性が高く，故に記述の完備性が高く，その検証も容易になった．
6. 必要な相互関連と相互作用の記述を従来と異なり全て明示的かつ形式的に記述構成要素化したこと．
7. 記述間相互関連定義と再帰展開なモデリング法を組み合わせて必要な詳細度や階層構造を持たせた詳細記述生成機構を備えていること．その実作業は形式の決まったモデリング手順であること．
8. モデリング機構付きの記述モデルであること．両記述モデル一覧表を行き来しての重ねての相互参照と，記述モデル内に埋め込んだ詳細化機構(表2-2)とにより，記述が自然に完成する仕組みを持つ．「相互参照」機構が実はOOモデリング作業の形式化であり，明示的・形式的・単純直線的に作業ができる仕組みでもある．
9. 駆動記述モデルを含めたことで，対象全世界の必要な動き・変遷を記述できる様にしたこと．
10. NJベースであるので，自然言語並みの記述性と了解性，及び分析・設計の仕様書を自然にそのままで兼用している高いドキュメント性が得られた．
11. 欠点はOOに対する概念の「型」を強制し，型に填まった考えを助長する側面があること．

3.2 適用実験とその評価

OOパラダイムの基本を理解し多少の記述経験のある

表1　OO対象記述モデルの汎化階層構成：記述要素種類の分類リスト

*

0 対象世界記述モデル (注 1) 0.1 対象記述モデル 5.x 駆動記述モデル
1 モデル化単位 (注 2) (注 3) 1.1 "モノ" (物理的存在，or 概念) 1.2.x 属性 1.3.x 振舞い 1.4.x 記述間相互関連 1.5.x 駆動記述モデル 1.6 離散化・詳細分解単位 (注 4) 1.6.1 最小構成"モノ"単位 1.6.2 最小離散化空間単位 (注 5) 1.6.3 最小分解属性単位 1.6.4 最小分解振舞い動詞 1.6.5 最小離散化時間単位Δt
2 属性 (注 6) 2.1 振舞い参照属性 (注 7) 2.2 相互関連属性 (構造属性) (注 2) 2.2.1 汎化・特化 2.2.2 集約・被集約 2.2.3 その他一般関連 (注 9) 2.3 相互作用属性 (注 10) 2.3.1 伝達作用 (働きかけ) 付置属性 2.3.2 応答 (反応) 付置属性 2.3.3 相互関連活性化付置属性 2.4 制約属性 2.4.1 一般制約 2.4.2 モデル化単位アクセス制約 2.4.3 属性アクセス制約 2.4.4 振舞い起動制約 2.5 時間的 (空間的) 関連属性 (注 11) 2.5.1 連接 (逐次) / 2.5.2 並列 2.5.3 分散 / 2.5.4 一般関連
3 振舞い (注 12) 3.1 内部振舞い 3.1.1 振舞い開始（発起） 3.1.2 一般内部振舞い 3.1.2.1 属性値参照振舞い 3.1.2.2 継続定常振舞い 3.1.2.3 属性値変更振舞い 3.1.3 条件分岐振舞い 3.1.4 反復振舞い 3.1.5 停止 3.2 相互作用伝達 (mp)(注 13) 3.2.1 伝達相互作用 (active mp) 3.2.2 返答・反応伝達 (passive mp) 3.3 相互作用 (振舞い)（＝3.1+3.2） 3.3.1 自発的働きかけ 3.3.2 相互関連活性化振舞い 3.3.3 制約属性活性化振舞い
4 記述間相互関連 (注 14) 4.1 詳細化・統合化関連 4.1.1 具象・抽象関連 4.1.2 離散化・修復再構成関連 4.1.3 詳細分解・再組立関連） 4.1.4 再帰展開・再帰統合関連 4.1.5 汎化・特化関連 (注 15) 4.1.6 集約・被集約関連 (注 15) 4.1.7 補足説明関連 (注 16) 4.1.8 その他一般関連 (注 17) 4.2 時間的あるいは空間的関連 4.2.1 連接 (逐次) / 4.2.2 並列 / 4.2.3 分散 / 4.2.4 一般関連 4.3 その他一般関連 (注 17)
5 駆動記述モデル 5.1 開始状況（初期化）記述 5.2 シナリオ駆動記述

卒業研究生（学部4年生），情報工学専攻の博士前期課程の学生，ドメインユーザの典型として数値流体力学（CFD）専攻の博士後期課程の学生を対象に記述実験を行った．彼らにはOONJ[3]というNJベースの記述言語をこのOO記述モデルと共に教え，数百行程度の記述例を与えて解説した後に，各自が選んだ対象世界に対して記述実験をして貰った．

その結果，ほぼ全員が「この一覧表の記述モデル表は具体的で了解性が高く，表の各要素を目標にモデル化すればよいので，OO記述を充分に行うことが出来，有用である.」とコメントを述べた．しかし同時に両表の用語が難解なのと，分析段階で一足飛びに両表の記述を実際に完全な形で実現する事は慣れの問題もあって複雑すぎて困難であるという意見が多く，予め段階的な記述法を工夫して組み込む必要もあることが分かった．

表２　OO 対象記述モデルの集約階層構成：各記述要素の構成

*

表２－１　OO 記述モデルの各記述要素の基本構成　(注 21)

(0 対象世界記述モデル) ≡ {(0.1 OO 対象記述モデル), (5 駆動記述モデル)}
(0.1 OO 対象記述モデル) ≡ {(1.x モデル化単位名), {(2.x 属性記述)$^+$, (3.x 振舞い記述)$^+$, (4.x 記述間相互関連記述)*} $^+$} $^+$　(注 23)
(2.1 参照属性記述) ≡ {(2.1 参照属性名), (2.1 参照属性値), (3.1.2.1 属性 (値) 参照振舞い名)}
(2.2.1 汎化属性記述) ≡ {(1.2.2.1 汎化モデル化単位名), (1.2.2.1 特化モデル化単位名)$^+$}
(2.2.2 集約属性記述) ≡ {(1.2.2.2 集約モデル化単位名), (1.2.2.2 被集約モデル化単位名)$^+$}　(注 22)
(2.2.3 一般関連属性記述) ≡ {(1.2.2.3 一般関連名), (1.2.2.3 相手モデル化単位名)$^+$}
(2.3 相互作用属性記述 (注 8)) ≡ {(2.3.x 相互作用属性名), (1.x 相互作用相手名)$^+$, (3.3.x 相互作用振舞い名), (2.x 伝達属性名)$^+$, (1.x 伝達モデル化単位名)$^+$}
(3.1.x 内部振舞い) ≡ {(3.1.x 内部振舞い記述 (名)), (2.1 参照属性記述)$^+$, (2.4.x 制約属性記述)*, (4.x 記述間相互関連記述)$^+$}
(3.2 相互作用伝達 (mp)) ≡ {(3.2.1 伝達相互作用名), {(3.2.1 伝達相互作用記述)*, (3.2.1 返答・反応相互作用記述)*}, (3.1.2.3 伝達属性値変更記述)*}
(3.3 相互作用) ≡ {(3.3.x 相互作用記述 (名)), (3.1.x 内部振舞い)*, (3.2.x 相互作用伝達)$^+$, (2.3.x 相互作用属性記述)$^+$, (2.4.x 制約属性記述)*} *, (4.x 記述間相互関連記述)*)
(4.x 記述間相互関連記述) ≡ {(4.x 相互関連名), (4.x 相互関連先)$^+$, (2.4.x 記述間制約記述)*}
(5 駆動記述モデル) ≡ {(5.1.x 開始状況記述)$^+$, {(5.2.x シナリオ駆動記述)$^+$, (2.4.x 制約属性記述)*, (4.x 記述間相互関連記述)*} $^+$}
(5.1.x 開始状況記述) ≡ {(2.1 参照属性名), (参照属性初期値)} *
(5.2.x シナリオ駆動記述) ≡ {(1.1 駆動主体名), (3.x 振舞い記述)} *

表２－２　再帰展開モデル　(注 24, 25)

(y.z 非終端記述) ≡ {(y.z 非終端記述)*, (x.x 終端記述)*, (1. x 非終端化モデル化単位記述)*}
(1. x 非終端化モデル化単位記述) ≡ {(0.1 OO 対象記述モデル), (4.1.x 記述間相互関連記述)*}

3.3 他研究との比較

多くの OO 分析/設計法[4)~10)]はその殆どがダイアグラム表記法を採用しており，NJ ベースでもなく，提示された記述例もやはり単純で短いものが殆どである．また本論文が目標としている分析段階の当初モデリングを如何にするかには殆ど触れておらず，モデル化後の表記・記述法を如何にするかに重点があるので，比較が困難である．

OO 言語や OO 表記法[11)]との比較について言えば，既にカテゴリーが異なるという点を無視しても，第１章の脚注で述べた様にモデリング方法については比較すべき対応するものを見出し難い．

OO 記述モデルの構成の個々の提案内容について言えば特記すべき新規性は持っていな．しかし提案全体を見れば，この様に形式化され簡潔で分かりやすい記述モデルの提案を筆者は寡聞にして知らない．表１の種類と構成の全リストを挙げ，強い相互参照性を持たせた形式の一覧表（勿論完全ではないが，）は知識に無く，内部に埋め込まれた詳細化のための再帰展開モデリングは表 1, 表 2 の自動的な詳細記述化を引き起こし，相互参照・自動的詳細記述化のサイクルに組み込まれて記述要素（種類，構成）が形式的に決まるという機構を組み込んだ結果になった．この様な構成と記述生成機構付きの OO 記述モデルは類例が少ないのではないか，というのが我々の見解である[4),6),7)]．

3.4 OO 記述モデルの特徴と利用・応用

OO モデリング論の出発点：本論文で提案した OO 記述モデルは OO 概念やモデリング作業を明示的に形式化した具体的な記述という形を持つ．従って OO に関する諸議論を行う際の共通基盤として最適なものの一つ，として使えよう．

記述言語設計への応用：我々の考えでは，OO 記述モデルは対象世界の「何を」記述するかを規定し，(記述)

表 3　OO 対象記述モデルの注リスト

(注 1) ファセット番号がちぐはぐなのは番号の階層をなるべく浅く取るための便宜的な番号付けの方法の結果である.

(注 2) .x に依り下位のファセット（分類) 番号を代表させる. 例えば 2.x は全属性を, 1.x は全ての要素種類を代表する.

(注 3) 拡張されたモデル化単位には本来のファセット番号の先頭に 1. を付置する. (例： 1.2.2.2 集約属性, 1.3.1 内部振舞い等). 付置記述構成要素それ自体も単独でモデル化単位となり得る. 各要素種類の構成は「構成リスト」参照.

(注 4) 単純・単一な動詞表現で捕捉可能, かつ記述者専決判断でこれ以上分けられない振舞い単位と設定したもの.

(注 5) 例えば, Δx, Δy, Δz, セル, メッシュ, ノード等の概念が挙げられる.

(注 6) 相互関連属性と相互作用属性を併せ相互関係属性と総称.

(注 7) 参照属性は内部振舞いに付置されて必ず参照される. その後で mp により外部に知らされる場合もある.

(注 8)OO のカプセル化は記述をグループ化して他と区別出来る一セットの形に明示する事を指し, 相互関連属性で扱う. 情報隠蔽の概念は含まない. OO の情報隠蔽は 2.4.x 制約属性で記述する.

(注 9) 上記の特定の分類に含まれないが, 上位階層の抽象度の分類としてはあり得る関連を全てここで扱う.

(注 10) 相互作用伝達付置属性とも言い, 相互作用記述に（正確には mp に) 付置されて伝達される属性である.

(注 11) 時間と空間属性そのものは通常は参照属性として定義する.

(注 12) データ駆動モデリングでは振舞いは属性値の変更（変化）であり, 振舞い駆動モデリングでは振舞いはモデル化単位の変化の捕捉である. 本記述モデルはデータ駆動モデリングの結果である.

(注 13)OO パラダイムのメッセージパッシング (mp) の概念を指す.

(注 14) 任意の二つの記述要素間の相互関連を示す. 継承と委譲, クラス (汎化) と汎化階層構造等も記述・表現可能.

(注 15) それぞれ, 4.1.1 及び 4.1.2 の特別な場合であり, 特にそう表現したい場合に使うことが出来る.

(注 16) 関係代名詞, 関係副詞, 或いは脚注に当たるものを別記する場合に用いる関連.

(注 17) 上記の特定の分類に含まれないが, 上位階層の抽象度の分類としてはあり得る関連を全てここで扱う.

(注 21) 肩添えの数表現記号：default は肩添え無しで 1, $()^*$：0 以上, $()^+$：1 以上, $()^0$：0 or 1.

(注 22) 相互関連 (構造) を持つ複数の構成要素記述 (名) の並びで表現され, 集約階層構造を成す.

(注 23) 表 1 の脚注と同様, .x とはその属する全ての要素種類を指すので表 1 を参照して選択する.

(注 24) 選択肢のある要素の場合や非終端の場合は再度「種類リスト」を参照する.

(注 25) 終端記述はその要素意味から記述内容 (名) を決め, 要素間相互関係は関係意味から決めて, NJ で記述する.

言語は「何を」によって抽出された記述要素を「『如何に』分かり易く単一特識させるか」を規定するものであると定義している. 100

OONJ と OODJ の記述力は既に検証されており, 従って OO 記述モデルはこれらの記述言語の仕様を含んだ上で更に拡張されたものであるから, OO 記述モデルを前提とした記述言語も間接的にではあるが OONJ, OODJ よりも高度な記述力であることが期待できよう. しかし, OO 記述モデルは既に（分かり易さを無視すれば,) 単一特識は実現しており, 全く新たな言語を設計する必要は無く, NJ を拡張した言語とし, オブジェクト指向プログラミング言語（OOPL）の詳細度と構造・機構に至るまでの「一意特定識別可能な構文規則」を持たせれば,「オブジェクト指向日本語（仮に OOJ と呼ぶ)」が出来ると考えている. この OOJ を設計するためには, もう少し精密な OO 記述モデルの規定が必要である.

2.3 節で述べた様に, 本論文中においては完備性と単一特識を保証する仕組みを提案することは出来なかった. しかしアイデアとしては提案されているので, 今後この方法の実現について追求する. もう一つ別の観点から言えば, 単一特識可能, というだけではなく, 簡潔かつ理解が容易な形で記述する方法が見出されれば, それは OO 記述モデルながら, 通常は（記述）言語が持つ事が多い機能まで備えてしまうことになる, とも言える. そのアイデア自体既に NJ を前提にした特定文型を提案しているという事もある. しかし, NJ はプログラミング言語や他の記述言語, 形式言語等とはユーザに占める位置付けが異なる特別な言語であり, NJ を前提にした OO 記述モデルは, 未だ言語を定義したとは言わなくて良いのではないかと考える.

4. 結論と今後の課題

結論： ある程度 OO に基づくモデリングや記述に馴れたドメインユーザや, 研究者, OO ソフトウェア開発技術者向けに, 真性実世界をモデル化記述する際の

表4　OO対象記述モデルの標準的モデリング手順

初期モデリング記述手順
（1）まず表1の「種類」リストを参照し，対象世界から着目記述要素をそのリストから選択して抽出する.
（2）それが新たに離散化や詳細分解すべき要素であれば，1.6.xをモデル化単位としてモデル化を行う.
（3）表2の「構成」表を参照して，該当する構成を確認し，再度必要な要素を抽出・モデル化・記述する.
（4）再度(1)に帰って対象世界の中から必要な記述要素全てを抽出記述するまで作業を繰り返す.
詳細化モデリング記述手順
（5）モデル化記述された要素の中で更に詳細化すべきものは，第2.2節で提案した三つの詳細化を行う. A．具象化・抽象化の場合は詳細化する要素自体を再分析し，上記の初期モデル記述段階の作業を行う. B．OO再帰展開モデリングの場合は，表2－1に従いx.xを改めてモデル化単位1.x.xと設定し，表1から2以降の付置記述要素を必要に応じて抽出し，表2の再帰展開適用と各要素の抽出・記述生成を行い，記述間相互関連で繋いだ上で「単独の」纏まったカプセル化記述する．ただし属性や振舞いは「独立」モデル化単位とはしない．再帰展開作業の終了はドメインユーザの専決判断. C．離散化と修復・再構成に基づく詳細化は1.6をモデル化単位にし，改めて表1に従って行う.
（6）上記のA．B．C．をモデル化の必要に応じて繰り返す．作業過程はしばしば多重ループを形成する.

OOモデリング方法の代替案として明示的に形式化・手順化されたOO記述モデルの方式を開発・提案した．記述言語OONJを流用した記述実験結果も良好であった．結論としてOOモデリングの応用記述現場にも有効・有用な記述モデル方式を形成できたと言って良いであろう．幾つかの段階分けを導入した記述支援環境ならば，その形式性や手順が明示的であることから初心者向けの方法としても期待できる.

今後の課題としては数百行から数千行規模での詳細かつ模範的な典型例の作成と公開，記述支援環境（ガイダンスシステム）の開発，それを通しての単語単位の一意特定識別を実現する具体的方法の実現，オブジェクト指向日本語記述言語の仕様（仮称OOJ）の設計等が挙げられよう.

参考文献

1) 磯田定宏，実世界モデル化有害論—オブジェクト指向モデル化技法の解明，電子情報通信学会論文誌，Vol.J83-D-I, No.9, pp.946-959, (2000).
2) 畠山正行，加藤木和夫，石井義之，オブジェクト指向記述日本語OODJとその記述環境，情報処理学会論文誌，Vol.41, No.9, pp.2567-2581, (2000).
3) 畠山正行，加藤木和夫，上田賀一，オブジェクト指向自然日本語OONJの設計と評価，第130回SE研究会研究報告，01-SE-130，pp.，(2001).
4) Wegner，P.，はやわかりオブジェクト指向，共立出版 (1992). 尾内理紀夫訳 (Concepts and Paradigms of Object-oriented Programming, key note Lecture in OOPSLA'89, OOPS MESSSENGER, Vol.1, No.1 (1990).).
5) 佐原伸，デザインパターン　オブジェクト指向分析/設計技法，ソフトリサーチセンター，(1999).
6) 所真理雄編，オブジェクト指向コンピューティング，第1・2章，岩波書店，(1993).
7) Rumbaugh，J. et. al.，Object-Oriented Modeling and Design，Prentice Hall，(1991).
8) 本位田真一，青山幹雄，深澤良彰，中谷多哉子，オブジェクト指向分析設計，共立出版，(1995).
9) P．コード，E．ヨードン著，羽生田栄一監訳，オブジェクト指向(OOA)［第2版］，(株）トッパン，(1993).
10) S．シュレィアー，S．J．メラー著，本位田真一，伊藤潔監訳，続・オブジェクト指向システム分析，啓学出版，(1992).
11) Booch，G. The Unified Modeling Language User Guide, Addison Wesley, Inc., (1999).