

解 説



オブジェクト指向分析・設計

2. 例題によるオブジェクト指向のモデル化作法

——オブジェクト指向分析の落とし穴†——

中 谷 多哉子†† 永 田 守 男†††

1. は じ め に

オブジェクト指向は、データに着目したデータ中心¹²⁾や抽象データ型¹⁶⁾などの考え方から発生したパラダイムである¹²⁾。したがって、分析の中心は、実世界からオブジェクトを抽出し、オブジェクト間の関連を定義するオブジェクト・モデルの構築である。オブジェクト指向分析方法論として、Coad/Yourdon 法¹⁾、OMT 法 (Object Modeling Technique)²⁾、Shlaer/Mellor 法^{3), 4)}、James Martin の OOA & D⁵⁾ などがある。いずれの方法論も、実体関連図¹¹⁾から派生したオブジェクト・モデルを最初に構築する点では共通している。

また、オブジェクトの動的な振舞いを分析するために、いくつかのモデルも提案されている。これらのモデルから得られる情報は最終的に統合され、オブジェクト・モデルが完成する。

本稿では、2. で OMT 法の基本的な表記を使用しながら、実際にオブジェクト・モデルを構築する過程を示し、オブジェクト指向によって問題領域を捉える方法論を具体的に示すことを試みる。さらに、3. では分析で陥りやすい誤りと対策について説明し、4. で分析、設計の利用上の留意点を解説する。

2. 例題を用いた OMT 法の説明

手法を説明するために「図書館システム問題 (共通問題)⁶⁾」(以下、図書館問題)を例題として取り上げ、2.2 でオブジェクト・モデルの構築を進める。2.3 では、構築されたオブジェクト・モデルに対して、動的側面からの検討を行う。

† A Manner of Object-Oriented Modeling by Examples—A Trap of Object-Oriented Analysis— by Takako NAKATANI (Fuji Xerox Information Systems Co., Ltd.) and Morio NAGATA (Keio University, Dep. of Administration Engineering).

†† 富士ゼロックス情報システム

††† 慶應義塾大学理工学部

2.1 例題—図書館問題—

以下の事象が発生する小さな図書館データベース・システムを考えよ。

- (1) 本の貸出, および返却
- (2) 図書館への本の登録, および削除
- (3) 特定の作者の本, あるいは特定の分野の本に関するリストの作成
- (4) 特定の利用者が借りている本のリストの作成
- (5) 特定の本を最後に借りた人の検索

このシステムの利用者には、図書館の職員と図書館の利用者の2種類のタイプがある。(1), (2), (4)および(5)の事象は職員に限られている。ただし(4)については、自分自身で現在借りている本の検索だけは、図書館の利用者でも行えるようになっている。

データベースは、次の条件を満たさなければならない。

- 図書館のすべての本は、貸出可能か、すでに貸し出されている。
- ある本が貸出可能であり、同時に、貸し出されている、ということはない。
- 図書館の利用者は、1回にある決まった数以上の本を借り出すことはできない。

2.2 オブジェクト・モデルの構築と洗練

オブジェクト・モデルを構築するために、まず分析の視点を決め、次にこの視点から問題領域を観察してオブジェクトを抽出し、オブジェクト間の関連とオブジェクトの振舞いを定義する。さらに、構築されたオブジェクト・モデルを洗練するために、継承などの手段を利用する。

以上の手順を、例題を使って具体的に説明する。

2.2.1 分析の視点

オブジェクトを抽出するためには、まず分析の戦略として視点を定める必要がある。視点によ

て分析のモデルが異なるためである。ここでは、「図書館の情報検索を支援する」視点で例題を分析すると仮定し、解説を進める。

2.2.2 オブジェクトの抽出と定義

オブジェクトの抽出と定義の方法について説明する。OMT 法では、インスタンスとクラスの表記を区別している。インスタンスとは、実世界の「もの」に対応するオブジェクトで、個々の特徴を属性値にもっている。このインスタンスを戦略で定めた視点に基づいて分類し、抽象化したものがクラスである。

したがって、クラスは、インスタンスのデータ構造としての属性と、データに関する手続きである操作から構成される。

オブジェクトの抽出とは、実世界のインスタンスを視点に基づいて分類することであり、オブジェクトの定義とは、これらの分類されたインスタンスに共通な性質を、クラスの構造として定義することである。

インスタンスを表記するために、図-1 に示した2種類のアイコンを使用する。

また、クラスを表記するために、図-2 に示したクラス名のみアイコンと3段で表現するアイコンなどが提供されている。このアイコンには、1段目にクラス名、2段目に属性に関するデータ（属性名：型＝初期値）、3段目には操作に関するデータ（操作名（引き数名：引き数の型、引き数

(クラス名)

(クラス名)
属性値1
属性値2
属性値3

図-1 インスタンスの表記

クラス名

クラス名
属性名1:型1=初期値
属性名2:型2
属性名3
操作名1(引き数11:型11,引き数12:型12):型x
操作名2(引き数21:型y)
操作名3

図-2 クラスの表記

名：引き数の型、…): 返り値の型) を書く。

それ自身が情報をもち処理の対象物であるもの、また、事象の発生とともに状態変化があり、その後の事象に影響を与えるものは、オブジェクトの候補になる。この例では、「図書館」のように問題領域の場として実在するものも、オブジェクトの候補である。

オブジェクトを抽出する作業は、モデル化の対象となっている問題領域の専門家と、ソフトウェアの分析者との協力によって進められる。問題領域の専門家は、作られたモデルと実世界の相違を評価する役割をもち、分析者は、問題領域に対する理解を深めシステムに反映させる責任がある。

例題から抽出されたオブジェクトをデータの構造や役割によって分類し、クラスを定義しよう。

2.2.3 関連の定義

「図書館」、「図書」、「利用者」の間の関連を考えてみよう。図書館問題には図-3 に示すような、一つの図書館は複数の図書を蔵書し、おのおのの図書はたかだか1人の利用者に貸し出される実世界がある。このようなインスタンス間の関係をリンクと呼んでいる。図-4 は、図-3 の情報を抽象化したものである。インスタンスはクラスに、リンクは関連として表記され、また、リンクの多重度を表現するために、関連上に次のアイコンを使用している。

●: 「多 (0 以上)」の多重度

○: 「0 または 1」の多重度

表記なし: 「厳密な 1」

1+: 「1 以上」

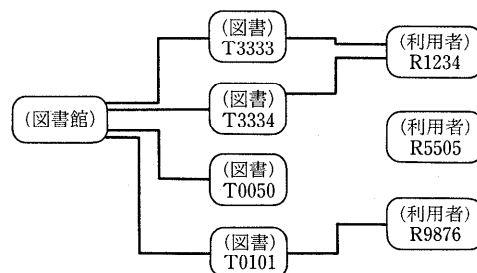


図-3 インスタンスとリンク

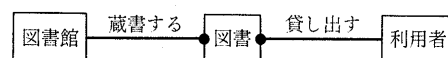


図-4 クラスと関連、多重度

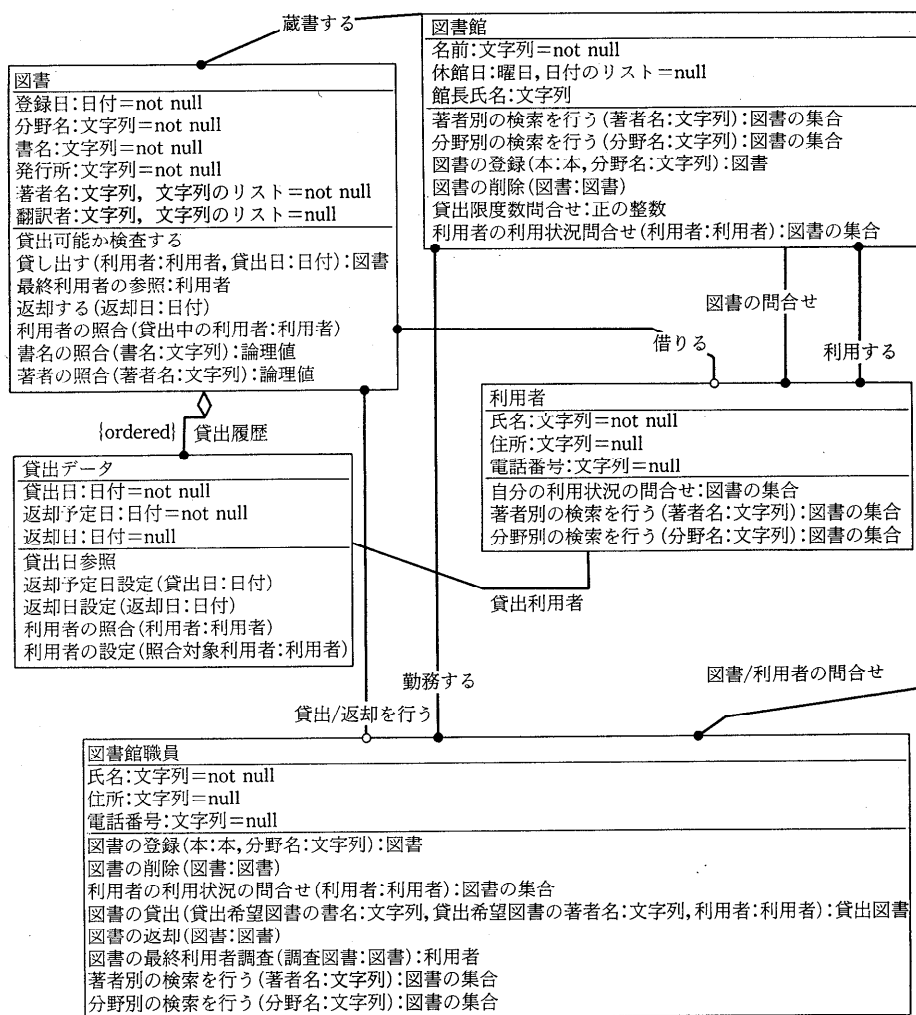


図-5 図書館問題のオブジェクト・モデル

その他:「3, 5, 7」など、多重度の数値

このようにして、実世界から抽出されたオブジェクトを抽象化してモデルを構築する。

次に、オブジェクト間の特別な関連である集約について説明する。集約は、全体一部分構造¹⁾(part-of)とも呼ばれている。表記は全体側の関連上に、◇を用いて他の関連と区別する。集約には、全体側のオブジェクトの状態が部分側のオブジェクトに伝播する性質があり、通常の関連とは区別できる。

2.2.4 操作の定義

クラスを構成する要素である操作は、オブジェクト指向プログラミング言語のメッセージに対応する。例題中に表れる動詞から、オブジェクトの役割、機能、振舞いを列挙してみよう。

「図書館職員」の役割は「図書登録/削除」, 「図書貸出/返却」であり, 「利用者」と「図書館職員」共に「著者別一覧参照」, 「分野別一覧参照」ができる。操作を書き込んだ全体のオブジェクト図を図-5に示した。

2.2.5 モデルの洗練

構築されたオブジェクト・モデルを洗練するための手段について説明する。

1) 継承による洗練

継承は、モデルがもつ共通概念の階層化を行うオブジェクト指向の重要な仕組みである。モデルの洗練は次の手順で進める。

(1) クラス間の構造、振舞いの類似点に着目し、共通部分をもつクラスを定義する。

ここで定義されたクラスが継承されるスーパー

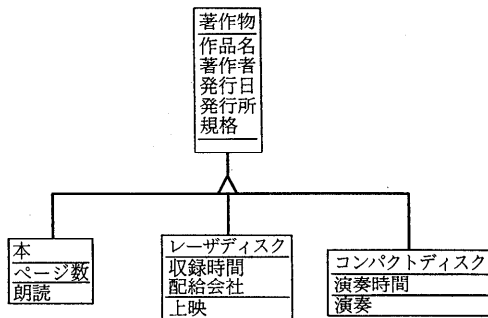


図-6 継承による概念の階層化

ラス¹²⁾になる。

(2) (1)で定義したクラスを共通部分とするスーパークラスに対して、違いだけをサブクラス¹²⁾で定義する。

たとえば、「本」「レーザーディスク」「コンパクトディスク」の共通部分をまとめて、「著作物」をスーパークラスとして定義できる。これらのクラス間を継承によって関連づけたモデルが図-6である。継承を表す場合、△を関連上に表記する。

2) 多対多問題の解決

図-7では、「図書館」と「利用者」の間の多重度が多対多になっており、二つのオブジェクトの対応関係が不明確である。OMT法では、1対多や多対多の関連を明確に定義する手段として、次の二つが提案されている。

(1) 関連にリンク属性を定義する

図-8は、「利用者が図書館を利用する」という事実から「人が図書館を利用する」というモデルへ一般化し、対応関係を「利用者番号」というリンクの属性で表現した。このリンク属性から、関

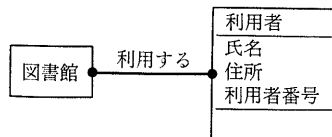


図-7 多対多問題

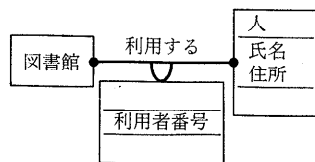


図-8 リンク属性を定義した関連

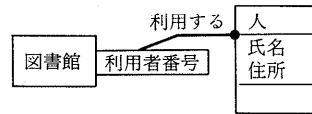


図-9 限定付き関連

連付けられているオブジェクト双方を管理する、新しいクラスを定義できる。

(2) 関連に限定子を定義する

1対多や多対多の対応関係を「限定子」というもので関連づける方法がある。図-9は「人」と利用している「図書館」との対応関係を「利用者番号」（限定子）で限定している。

以上の洗練の手順で、図-10のオブジェクト・モデルを構築することができた。

2.3 オブジェクト・モデル、動的モデル、機能モデルの統合

オブジェクトの動的な側面には、状態遷移と機能がある。動的モデルには、時系列上で進められるオブジェクト間のメッセージ送受信のタイミングと、メッセージ送受信によって起きるオブジェクトの状態遷移を表す。また、機能モデルには、モデル全体における情報の流れを表す。

1) 動的モデル

まず時間軸上で発生する各事象ごとにシナリオを記述する。このシナリオに従って、事象トレース図とオブジェクトの状態図を記述する。

(1) シナリオ

「図書の貸出」に関するシナリオの例を示そう。

〈図書の貸出〉

図書館職員は、利用者が貸出を希望する図書をデータベースから検索し、該当する図書が貸出可能か否かを調べる。

貸出可能な図書が検索できた場合、図書館の職員は利用者の貸出限度を検査する。限度内であれば貸出処理を進める。限度を超える場合は貸出処理を中止する。

次に、貸し出す図書の貸出カードへ利用者を登録し、利用者カードに貸出図書の記入を行う。

(2) 事象トレース図

事象トレース図には、横軸にメッセージ送受信を行うオブジェクト、縦軸に時間を取り、矢印でオブジェクト間のメッセージの発信状態を書く。

「図書貸出」のシナリオから作成した事象ト

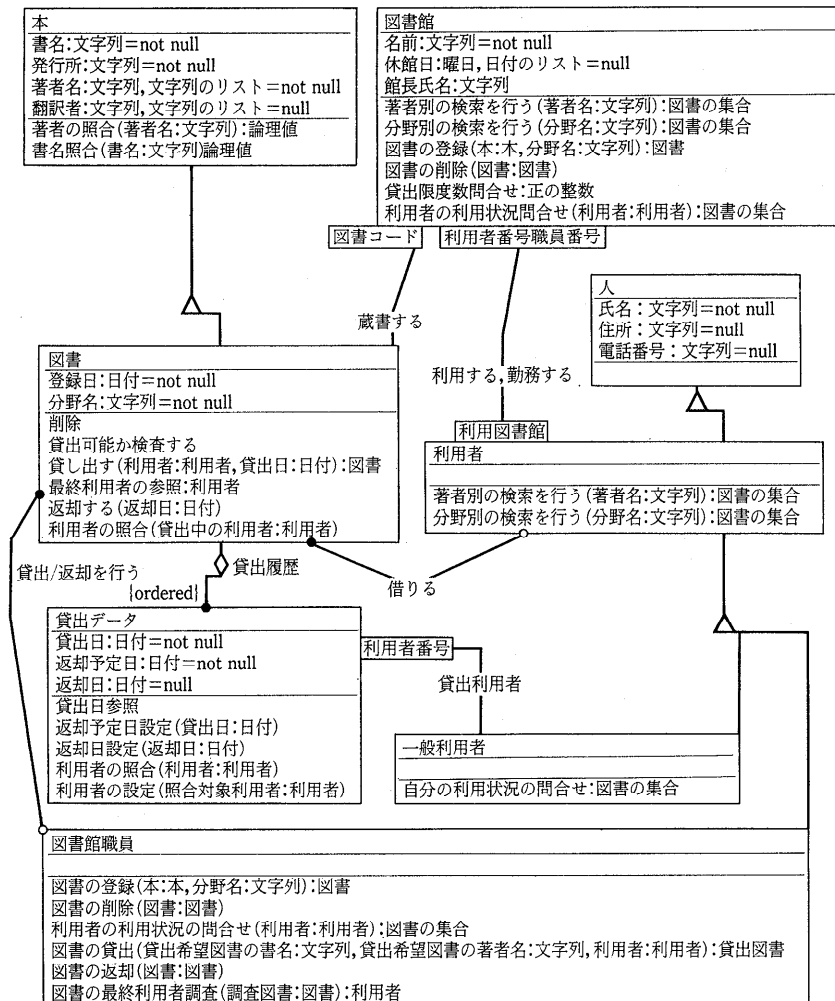


図-10 洗練されたオブジェクト・モデル

レース図を図-11 に示した。この事象トレース図から、オブジェクトの新たな役割も発見できる。

(3) 状態図

状態図には、オブジェクトの状態遷移と事象の関係が表される。図-12 では、貸出限度を超えた「貸出不可能利用者」の状態にある利用者は、「図書の返却」という事象が起き、「図書返却依頼」という状態を経ない限り「貸出可能利用者」にはならないことが示されている。

2) 機能モデル

機能モデルでは、データフロー図を適用する。

(1) データフロー図

処理を表すすべての丸(○)が、一つ以上のオブジェクトの操作に対応する。また、データの流から、その処理がどのオブジェクトの操作か、

そのときの引き数は何かを分析できる。

図-13 では、「貸出中にする」処理は、「図書館の本」のデータ更新を行うため、「図書館の本」自身の操作として定義されなければならない。

動的モデル、機能モデルの情報は、オブジェクト・モデルの属性や操作として定義される。最終的にこれらのモデルがすべてのシナリオでシミュレートできたとき、分析は終了する。

3. 陥りやすい誤りと対策

ここまで、分析過程の概要を説明したが、実際の分析にあたって陥りやすい誤りを分類して示し、その対策を示す。

3.1 モデル化における視点の違い

同一の対象であっても、分析の視点が異なるこ

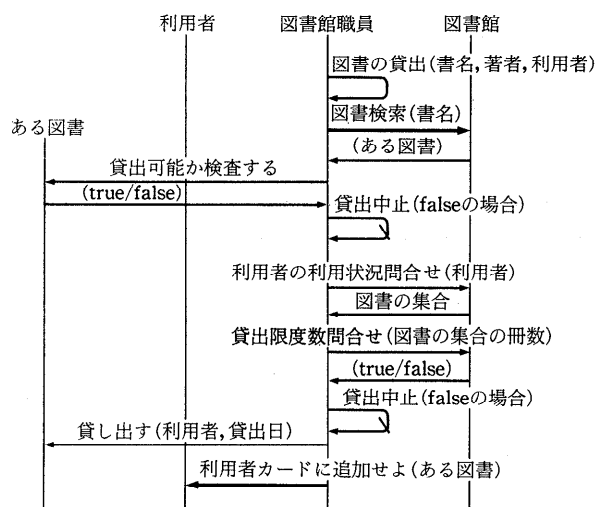


図-11 「図書の貸出」における事象トレース図

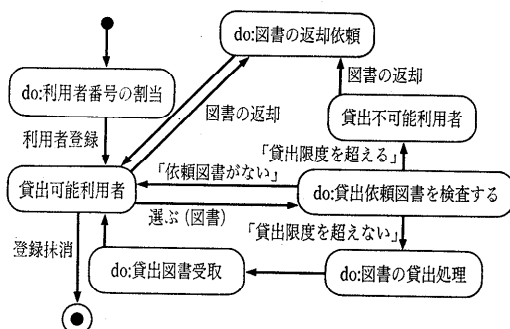


図-12 「利用者」の状態図

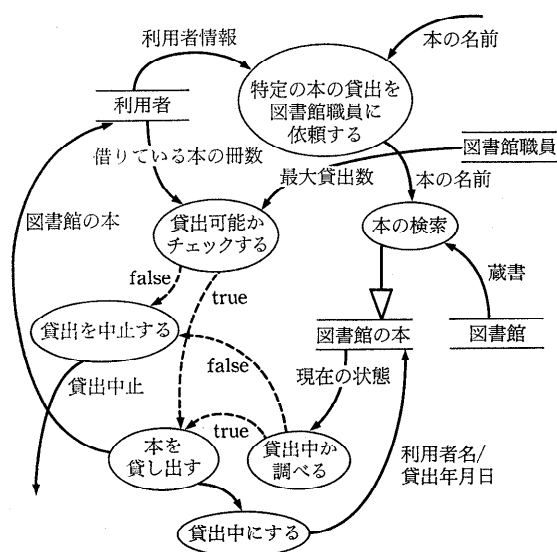


図-13 図書館問題のデータフロー図

とによって結果は異なる。たとえば、本稿の図書館問題の視点は「図書館の情報検索を支援する」であった。視点を「図書館職員の作業を支援する」に変更すると、図書の購入、整理、廃棄といった作業に関するオブジェクトも定義しなければならない。

3.2 実世界とモデル世界のオブジェクト

実世界のオブジェクトをどうモデル世界へ射影するか。これはオブジェクト指向分析の重要な課題である。何をオブジェクトとして定義するかは分析の場に立ち会った関係者の判断に委ねられる。

一般に、実世界で使われている名詞はオブジェクトの第1候補となる。しかし、図書館問題で使われている「データベース」は、オブジェクトとして定義しない名詞の例である。

この問題の「データベース」は、書籍管理を行うことを意味しているのであって、特定のデータベースを想定していない。図-10では、「データベース」は、「図書館」が「図書」を「蔵書する」関連として表現してある。実世界の対象物がモデル世界でオブジェクトとして定義されるかどうかは、分析の視点やその対象物の意味をどのように理解するかに依存する。

3.3 要求変更によるモデルの進化

利用者を取り巻く環境やシステム環境の状況変化によって、要求が変化することは避けられない。しかし、モデルを進化させることで、将来の類似の変化に対する変更を局所化できる。

モデルの拡張や洗練には次の種類がある⁷⁾。

- (1) オブジェクトの属性の削除、追加、型の変更
- (2) オブジェクトの操作の削除、追加、操作名および引き数の型の変更
- (3) クラス階層の変更。抽象クラスの追加、削除
- (4) 具象クラスの追加、削除

図書館が、図書以外のレーザーディスクやコンパクトディスクを貸し出すようになったという業務変更を想定してみよう。貸出対象オブジェクトに対して変更は加えられるが、「図書館職員」の「登録／削除、貸出／返却、問合せ」と

いった操作や、「図書館職員」と「利用者」の構造は変化しない。このように、モデルを進化させる場合は、同種の変更に対して頑健に対応できる構造をもたせるように配慮する。

3.4 再利用を考慮したモデルの進化

オブジェクト・モデルの再利用を考慮すると、オブジェクトの構造も変化することがある。3.3で述べた要求変更は同一アプリケーションに対するものであるが、再利用は類似アプリケーションへのモデルの適用である。この場合は、実装の類似性よりも、オブジェクトの仕様の類似性に着目して抽象クラスを定義したほうが良い。

分析時の限られた情報から、将来の再利用を予想することは困難であるが、最小限、継承構造が可読性を落とすようなモデル化の失敗は避ける必要がある。そのためには、実世界のインスタンスから定義したクラス間の継承に基づいて、将来の類似クラスへの再利用性を配慮するほうが良い。人間がもっている既存概念から抽象クラスを定義すると、不必要なクラス階層が定義され、可読性を落とす結果を招くこともある。

クラス・ライブラリのクラス階層は広く深い^{8),9)}、アプリケーションに特化したクラス階層は浅く、抽象クラスが少なくなる傾向がある¹⁴⁾。

3.5 オブジェクト間の構造

ここでは、オブジェクト間の構造を定義する場合の注意点について述べる。

3.5.1 継 承

継承の活用法と、多重継承と集約構造との取扱いを比較して解説する。

1) 活 用 法

継承は、操作の限定、追加、仕様を洗練するときを使うことができる¹⁰⁾。

(1) 限 定 の 例

「貸出禁止登録物」を表すために、「貸出禁止登録物」を継承して「貸出可能登録物」が定義されるモデルと、「貸出可能登録物」を特殊化し、「貸出を禁止」した「貸出禁止登録物」を定義する2種類のモデルを考えよう。実世界では、「禁帯出」のマークを見かけるが、このマークがない場合は貸出可能であるから、図書館登録物は一般的には貸出禁止物ではないと考える。これは限定の例で、後者のモデルに対応する。

(2) 追 加 の 例

「著作物」←「本」←「図書」の継承構造がこれにあたる。この継承は機能追加やコードの再利用に利用されている。

(3) 内部仕様を洗練するための継承の例

登録物←貸出物の継承構造における「貸出可能か」の操作の再定義はこの例である。

これはポリモルフィズムを利用し、外部仕様を変化させずにカプセルの内部仕様を変化させる。

以上の三つの継承は、すでに継承すべきクラスが存在している場合の議論である。前節で述べた再利用を考慮したモデルは、既存のモデルから三つの継承を活用するための再利用対象クラスを定義するものであった。

2) 多重継承と集約構造

継承は一般にモデルの理解を助けるが、複雑な継承は理解の弊害になることもある。多重継承では特にこの傾向が顕著である。

一方、多重継承はわれわれの日常の感覚に近いモデル化を実現する機構でもある。次にあげる実世界の解釈とモデルとを比較してみよう。

(1) 「勤労学生は、社会人でもある学生」

(2) 「勤労学生は、社会人と、学生を内包するオブジェクト」

(3) 「勤労学生は、社会人という側面をもっている学生」

(1)の例は図-14に示した。「勤労学生」は社会人と学生からの多重継承で定義されている。

(2)は継承を集約構造で言い替えたモデルである。(3)は継承と集約を組み合わせたモデルである。図-15に示すように、たとえば、「住所」という属性は継承している「学生」の属性を参照することになる。必要であれば、「社会人」の「住所」を参照する操作を「勤労学生」に定義することも可能である。

このように、分析者の視点を大きく変えずに、多重継承を使用しないモデルも構築できる。多重継承のかわりに集約構造を用いた場合、オブジェクトで定義される操作の数は増加する。しかし、集約されたオブジェクトから、部分オブジェクトの不必要な操作を隠すことも可能になる。

3) 集約構造と継承構造

集約構造と継承構造の二つの構造は、モデル化の過程では近い関係にある。しかし、継承では、

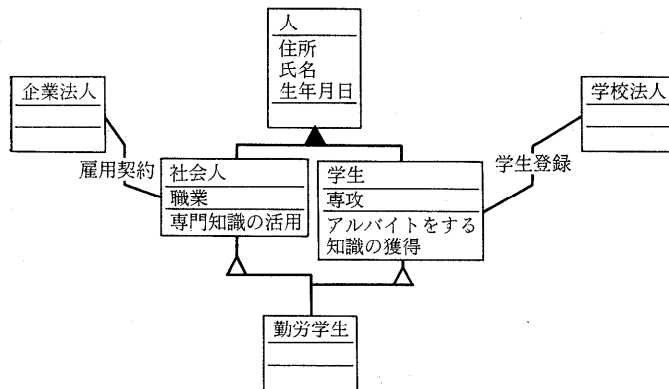


図-14 多重継承を用いた例

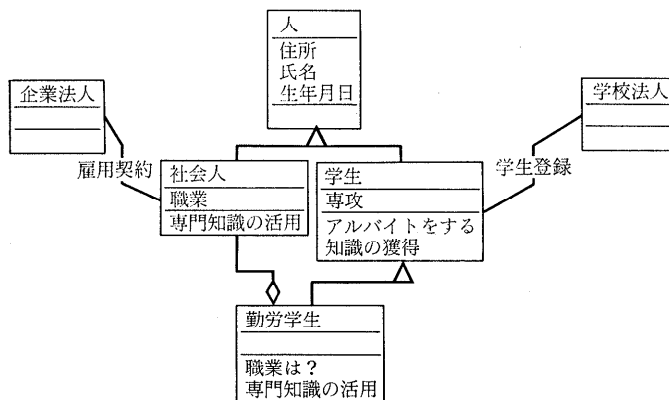


図-15 継承と集約を用いた例

サブクラスのインスタンスがスーパークラスのインスタンスの部分集合になっているのに対し、集約では全体オブジェクトから部分オブジェクトへの状態の伝播がある。

図-16を参照しよう。スタックA、Bどちらの構造が好ましいか。継承を用いた場合は、スタックAは、オブジェクトの意味がリストであることを表している。したがって操作の限定(図中“not...”)を行わないと、スタックAはリストに適用されるすべての操作ができることになる。一方、集約を用いたスタックBは内部構造として、データをリストで実現していることを表している。内部仕様は隠蔽されるため、スタックの意味を考えるとスタックBが正しい。

4) 継承の留意点

継承による生産性向上と、既存クラスの学習時間の増加とは、トレードオフである^{13),14)}。ちょうど、新規のシステムを既存システムの保守によ

て開発するようなものである。オブジェクト指向導入時の教育の問題とは、この学習時間を主に指している。

3.5.2 リンクと関連

複雑な実世界では、同じオブジェクト間に複数のリンクが存在する場合もある。

たとえば、「自動車」と「人」との間の関連を考えてみよう。両者の間には「所有する」「運転する」「乗る」などの複数の関連が存在する。これらの関連がすべて実装に必要なかといった決定は設計段階まで先送りしてよい。

3.6 オブジェクトの内部構造

定義された各オブジェクトの内部構造を構築する場合の留意点について解説する。

3.6.1 属性の定義

1) 属性と派生属性

図書館に蔵書されている図書の数、利用者が現在借りている図書のリストなどは、すべての蔵書の属性を検査することで得られる「図書館」の派生属性である。分析段階では、派生属性をオブジェクトの属性として定義しない。そのかわり、派生属性を計算結果などから得るための操作をオブジェクトの中で定義する。

3.6.2 操作の定義

1) 「個々」と「全体管理」

これは、オブジェクト・モデルの見方の問題である。図書館問題では、次のような要求がある。

(1) 「図書」の操作に“著者別一覧リストの作成”を定義する。

(2) すべての図書を管理している「図書館」に“著者別一覧リストの作成”を定義する。

オブジェクトは、ある操作を実行するときには、カプセル内の情報しかアクセスできない。また、オブジェクト図に描かれたクラスを表す一つのアイコンは、インスタンスの集合を表しているのではなく、一つのインスタンスを表現している。したがって、(1)は、一冊の「図書」が「図書館」のすべての「図書」に関する情報をアクセ

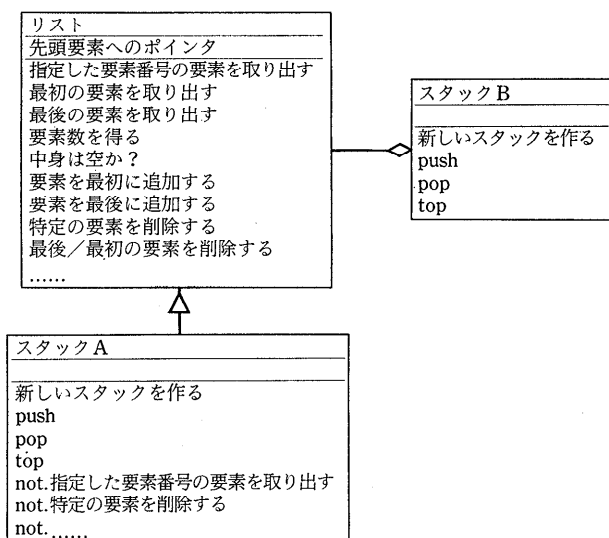


図-16 スタックの集約構造と継承構造

スすることを意味するモデルである。これは明らかに誤りである。

2) 属性の操作

オブジェクト図では、属性の初期化、代入、参照の操作は省略するのが一般的である。ただし、外部に対して公開する操作は表記したほうが良い。

3.7 モデル構造の評価

この節では、ここまで考えてきたオブジェクト・モデルを事象トレース図によって評価/検証する。評価結果から、オブジェクト・モデルをより適切なものへ修正できる。

1) 役割分担による評価

事象トレース図で、特定のオブジェクト間の頻繁なメッセージの通信は、オブジェクトの情報隠蔽を壊し、結合度が高いことを意味している。このような構造はシステムの信頼性を低下させる。

2) メッセージ送受信とオブジェクトの検証

事象トレース図でメッセージを受け取るオブジェクトには、そのメッセージの名前が操作名として定義されていることを確認する。

3) 操作と関連によるオブジェクトの検証

直接/間接の関連が存在しないオブジェクト間にはメッセージ送受信は存在しない。モデルの構造として評価を行う場合、必要なオブジェクトへ、関連で辿れるかを確認する。

図-17では「利用者」と「図書館」の間に、「利用図書館」という限定子付き関連がある。「利用

者」に定義されている二つの操作は、いずれも「利用図書館」を参照する。しかし、限定子を決定するための情報が定義されていないため、図書の検索時に、特定の利用図書館に対してメッセージを送信することができない。これを避けるためには、二つの操作に検索対象の図書館を引き数として与える必要がある。

4. オブジェクト指向分析、設計 利用上の留意点

1980年代後半より進められてきたオブジェクト指向分析・設計に関する研究は、特に、分析段階における問題領域のモデル化という点において、手続き中心の視点とは異なる視点をわれわれに提供してくれた。

オブジェクト指向によるモデル化が分かりやすいと言われるのは、ソフトウェア技術者にとってより、モデル化すべき問題領域の専門家にとってのようである。非ソフトウェア技術者が分析者とともにモデル構築を行いながら、曖昧さや実世界の問題点を自分自身で発見できるという長所をここで指摘しておきたい。

もともと分析と設計は明らかに区別すべき視点をもっている¹⁵⁾。問題領域の専門家は実世界をみているのに対し、分析者は実世界を「モデル化」という抽象化の視点で観察する。一方設計者は、分析者から受け取ったモデルをもとに、コンピュータにとって最適なモデルを構築することに興味をもっている。この視点の違いを明確に区別しながら開発の作業を進め、不十分な理解による

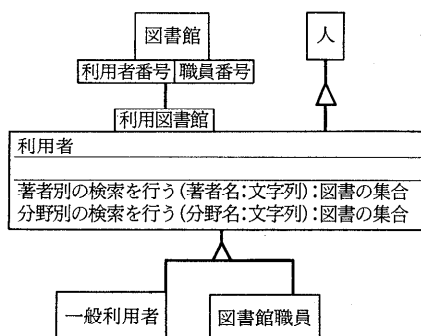


図-17 関連と引き数

必要以上に深いクラス階層，継承の失敗，不十分な役割分担を避けなければならない。

5. おわりに

本稿では，OMT 法の表記を用いてオブジェクト指向分析の作法を論じるとともに，分析時に注意すべき点について，例を示しながら説明した。

今後は国内でも，パイロット・システムへの適用が検討され，従来の開発方法論との融合を進めていくなど，実システムへの適用に対する検証がますます盛んに進められていくであろう。

オブジェクト指向は発展途上の技術であり，ソフトウェア開発上のすべての問題を解決したわけではない。しかし，実際に使いながら現在の方法論の問題点を探り，解決していきたいものである。

参 考 文 献

- 1) Coad, P. and Yourdon, E. 羽生田栄一監訳：オブジェクト指向分析 (OOA)，トッパン (1993).
- 2) Rumbaugh/Blaha/Premerlani/Eddy/Lorensen : 羽生田栄一監訳：オブジェクト指向方法論 OMT，トッパン (1992).
- 3) Shlaer, S. and Mellor, S. 本位田真一，山口 亨訳：オブジェクト指向システム分析，啓学出版 (1990).
- 4) Shlaer, S. and Mellor, S., 本位田真一，伊藤 潔訳：続オブジェクト指向システム分析，啓学出版 (1992).
- 5) Martin, J. and Odell, J. J.: Object Oriented Analysis & Design, Prentice Hall (1992).
- 6) "Problem Set for the Fourth International Workshop on Software Specification and Design", Software Engineering Notes, Vol. 11, No. 2 (1987).
- 7) Erradi, M., Bochmann, G. V. and Dssouli, R.: Framework for Dynamic Evolution of Object Oriented Specifications, Proc. Conf. on Software Maintenance, pp. 96-104 (1992).
- 8) Goldberg, Adele. 相磯秀夫監訳：Smalltalk-80 対話型プログラミング環境，オーム社 (1986).
- 9) Goldberg, A. and Robson, D. 相磯秀夫監訳：Smalltalk-80 言語詳細，オーム社 (1988).
- 10) Meyer, B. 二本厚吉監訳：オブジェクト指向入門，アスキー出版 (1990).
- 11) Chen, P.: The Entity Relationship Model—

Toward a Unified View of Data, ACM Transactions on Database Systems (1976).

- 12) Wegner, P.: Concepts and Paradigms of Object-Oriented Programming, OOPS Messenger Vol. 1, No. 1, ACM (1990).
- 13) Chapin, N.: Software Life Cycle, IEEE Proc. Conf. on Software Maintenance, pp. 6-13 (1988).
- 14) 中谷多哉子，春木良且：オブジェクト指向における効果的な部品の再利用，ソフトウェアシンポジウム '93，ソフトウェア技術者協会，pp. 6-14 (1993).
- 15) Davis, M. A.: Software Requirements, revision, OBJECTS, FUNCTIONS & STATUS, Prentice Hall (1993).
- 16) Cleaveland, J. C. 小林光夫訳：データ型序説，共立出版 (1990).

(平成 5 年 9 月 24 日受付)



中谷多哉子 (正会員)

1980 年東京理科大学理学部応用物理学科卒業。1990 年富士ゼロックス情報システム(株)入社。1994 年筑波大学大学院経営政策科学研究科経営システム科学専攻修士課程修了。現在，オブジェクト指向導入のための支援，モデル技術，利用者参画によるソフトウェア・プロセスの研究に従事。ACM，ソフトウェア技術者協会各会員。



永田 守男 (正会員)

1948 年生。1971 年慶應義塾大学工学部 (現在，理工学部) 管理工学科卒業。1973 年同修士課程修了。博士課程入学。1974 年管理工学科助手，その後専任講師，助教授を経て現在，教授。工学博士。慶應義塾大学日吉メディアセンター副所長を兼務。数式処理や自動的な推論といった数理的分野，それに人間の視点からのソフトウェア工学，ヒューマンインタフェース，グループウェアについて研究している。利用者が主導するソフトウェア構築にも興味がある。電子情報通信学会知能ソフトウェア工学研究専門委員会委員長，日本数式処理学会理事，日本ソフトウェア科学会，経営情報学会，ACM，IEEE など各会員。