ソフトウェア工学 103−12−5
（1995．3．9）

ドメイン分析・モデリングの利用法・研究法
ドメイン分析・モデリングとオブジェクト指向

吉田裕之　富士通研究所

1　はじめに

オブジェクト指向開発方法論は，近年ソフトウェア工学の分野で大きな注目を集めている．最初のブームの時には，オブジェクト指向はプログラミングの技法あるいはプログラミング言語が提供すべき概念として捉えられていた．現在は第二のブームと言える状況であり，システム分析・設計への適用方法がさかんに開発されている．さらにテスト工程などへ適用範囲を広げ，ライフサイクル全体に渡る一貫した開発方法論へと発展しつつある．

ドメイン分析・モデリングを，システム開発のためのインフラストラクチャの整備を目的としたものであって，特定のシステムの開発に着手する以前に行なうべき一つの開発工程であると考えれば，これにオブジェクト指向の考え方・技術・方法論を適用しようとするのは自然な考えであろう．

我々は，最も広く利用されているオブジェクト指向分析・設計法であるOMT[1]を使ったドメイン分析・モデリングを試みた[2]．ここではその試行から得た知見をおさらいし，また関連した他の研究開発動向を紹介する．

2　ドメイン分析・モデリングへのオブジェクト指向方法論の適用

OMTは元来システム分析及び設計工程を対象とする方法論であるから，ドメイン分析・モデリング工程への適用にあたって考慮を要する点が少なからずある．

ドメイン分析・モデリングを行なう目的を，第一にドメインの理解を助けること，第二にシステム分析・設計及び再利用の基盤を提供することと考え，この二つの観点からOMT適用のためのポイントを概説する．

2.1　ドメインの理解

まったく新規にシステム化を行なうドメインは稀であり，なんらかのシステム化がすでに行なわれていることが多いだろう．その場合，データベース，ファイルあるいは印刷された帳票などの形で様々なデータがすでに存在し，オブジェクト抽出の良い手がかりとなる．例えば図書館ドメインでは，利用者データベースや蔵書目録があり，これらから「利用者」「図書」というオブジェクトが抽出でき，そららが持つべきデータ属性も簡単に分かる．

しかしここで注意しなければならないのは，データをオブジェクトとして抽出しただけではドメインの理解にあまり役に立たないことである．(図書館ドメインの例で利用者や図書と言われて我々が直ぐに理解できるのは，我々が図書館ドメインをある程度すでに理解しているためであって一般には) passive なオブジェクトは，それを利用する active なオブジェクトとの関連を把握することで初めて理解できる．

そこで我々は「業務」を active オブジェクトと捉えることにした．業務オブジェクトを抽出し「データ」を表現する passive なオブジェクトとの関連を記述することによって，業務そのもの及びデータとその属性が持つ意味が理解できてくる．

また，ドメイン全体を理解するためには，トップダウンに順次分解して概要から次第に詳細な記述になるようにドメインモデルを分割することが必須である．この場合も，データを表現するオブジェクトよりは，業務オブジェクトに注目して詳細化するのが分かり易い．例えば「図書館業務」オブジェクトを考えると，それがさらに「貸出業務」や「利用者登録業務」などからなること (part-of 階層) や，「利用者登録業務」の中に特殊なケースとして例えば (障害者等の)「優先利用者登録業務」が存在すること (is-a 階層) などをモデル化できる．

上位の「図書館業務」を中心として概要を記述し，次に「貸出業務」などの下位のレベルの業務オブジェクトを中心に，他の業務オブジェクトやデータオブジェクトとの関連を記述することで，個別の業務やデータの理解がさらに深まる．

active なオブジェクトとして agent，すなわち実際にその行動を行なう (業務を担当する) 人をモデル化する，という方針も考えられないではない．例えば「図書館職員」オブジェクトを抽出し，これと他の passive なオブジェクトの関連を記述する．しかし，この場合の大きな問題点は階層的に分割できないことである．

オブジェクトのライフサイクルを捉えることによって理解はさらに深まる．通常，業務オブジェクトは生成されたり消滅したりすることがない．しかしその内

部では遂行する業務の手順にしたがって状態が変化し，特にクラス階層の下位の業務オブジェクトは短いサイクルの状態変化を日常的に繰り返している，と考えることができる．この様子はOMTの動的モデル(階層化状態遷移図)を用いてモデル化できる．

一方データオブジェクトには，一旦生成されるとほとんど状態変化をせずに長く生きながらえるものと，頻繁に生成・消滅が起こりその間も次々に状態変化を起こしているもの，の二種類がある．我々は前者を「管理対象オブジェクト」，後者を「管理情報オブジェクト」と呼んでいる．例えば「利用者」「図書」などは管理対象オブジェクトであって，「貸出」「発注」などは管理情報オブジェクトである．管理情報オブジェクトは，管理対象オブジェクトどうし，業務オブジェクトどうし，あるいは管理対象オブジェクトと業務オブジェクトの間の関連オブジェクトとして抽出される．このような観点がオブジェクト抽出の重要な手がかりになる．

オブジェクトを使ったドメイン理解のもう一つのポイントとして，(特に管理対象オブジェクトにおける)実体と記述の区別という問題がある．例えば，図書館には利用者がやってくるし，また利用者データベースには各利用者に関するデータが格納されている．そこで「利用者」という実体を表現するオブジェクトと，「利用者レコード」という実体に関する何らかの記述データを表現するオブジェクトを考えることができる．しかし，この区別はモデリングをかえって複雑にする虞がある．例えば「氏名」や「住所」といった属性が両者にはあって，対応するオブジェクトの間ではこれらは一致していないといけない，などと記述しなくてはならなくなる．

ところが，例えば"Rumbaugh, J., et al.: Object-Oriented Modeling and Design, Prentice Hall, 1991"という「図書」オブジェクトを考えてみると，これは記述オブジェクトであって，実体オブジェクトにはなりえない．なぜならばこの図書館はこの図書を10冊所有しているかもしれず，その場合，10個の実体を表現するオブジェクトが存在しなければならないからである．後者のオブジェクトを「蔵書」オブジェクトと呼ぶことにする．図書オブジェクトが持つ情報を蔵書オブジェクトに持たせてしまい，図書オブジェクトを廃止することは可能であるが，第一に情報の重複(この場合，タイトル，著者，出版社などの同一データを10通り持つ)が問題である．また「発注」オブジェクトがこれらのうちのどれかを参照する，という不自然さを生む．

図書オブジェクトはクラスであって，蔵書オブジェクトはそのインスタンスなのである，と考えることは可能である．しかし，相変わらず発注オブジェクトからの参照対象として図書オブジェクトは存在する必要があるために，クラスオブジェクトという概念を導入することになる．また，分析時点で確定しないクラスであるから，クラスを動的に生成するためにメタクラスという概念も導入する必要がある．これらの比較的高度な概念を安易に導入すると，オブジェクト指向の考え方に不慣れなドメインの専門家や開発者の理解をかえって妨げる虞がある．

結局，この例では図書オブジェクトと蔵書オブジェクトを区別するのが自然である．蔵書オブジェクトは図書オブジェクトを参照し，図書発注業務は図書オブジェクトを，貸出業務は蔵書オブジェクトを処理対象とすると考える．

オブジェクト指向を使ったモデリング技術の一つにパターンという考え方がある．モデル化する対象の中にしばしば現れるパターンとその自然な記述方法をあらかじめ整理しておけば，モデリングの手間を省くことができる．Coadは7つの典型的なパターンを紹介している[3]が，上記の図書/蔵書の例はこの中の最初のパターンであるitem descriptionパターンに相当する．

2.2 記述と再利用のための基盤

ドメイン分析・モデルリングの第二の目的は，開発するシステムの要求，仕様，設計を記述するためのコンテキストを提供し，またそれらの記述やソースコードなどの再利用の基盤を提供することである．

システム分析以降の開発工程でオブジェクト指向を採用することを前提とするならば，この観点からもドメイン分析・モデリングに同じ手法を適用するメリットは大きい．

第一にオブジェクト指向開発用に提供されている様々なツールをドメイン分析・モデリング工程にも流用できる．そして，同じツールで作ったドキュメントをベースにシステム分析・設計を進めることができる．

またドメイン分析・モデリング工程で行なった記述とそれ以降の工程での記述を，「オブジェクト」という捉え方をベースに一元管理できる(可能性が将来にはある)．

一元管理できれば，各工程でどのような判断が行なわれてどのようにオブジェクトの記述が詳細化/洗練化されたか追跡可能となるから，上流工程での記述の差異がソースコードのどのような差異となるかが予測し易くなり，上流工程での再利用が促進される．

しかしそのためには，ドメイン分析・モデリングを

進めていく際にシステムの特定の実現方法にできるだけ依存しないように心がける必要がある．特定の実現に依存するほど再利用性が劣化する．また，特定の実現に依存した判断を他の判断とできるだけ分離しておくことも必要である．

ドメイン分析とシステム分析の役割分担をあらかじめ決めておくことも必要であろう．勝手に先走った判断をしたモデルは再利用性が劣ると考えられる．

3 関連する研究開発動向

我々は，OMT法の考え方・表記法をドメインモデリングに適用した．ドメイン分析・モデリングへオブジェクト指向を適用した他の事例と，それぞれの特徴を紹介する．

ドメイン固有の方法論を新たに開発した事例 NOON (Nenkin Object-Oriented Navigator)[4]は，年金管理システムの開発のために考案された方法論で，OMT法をベースに拡張と変更を加えている．NOONでは，事務処理アプリケーションを業務領域とシステムメカニズムに分けた参照モデルを用いる．また，大規模でバッチ処理中心である年金管理システムの開発のために，機能によって業務領域を分割するなど，機能的な見方を重視しているのが特徴である．

ROOD(Realtime Object-Oriented Design)[5]は，通信ドメイン用に考案されたオブジェクト指向開発法で，移動体システムの開発に用いられた．ROODでは，通信ドメインに固有な参照モデルを用いることで，オブジェクト抽出や開発者間のコミュニケーションをスムーズに行うことに成功している．

OOBE (Object-Oriented Business Engineering) [6][7]は，分散化された企業全体 (the distributed enterprise) の情報システム構築におけるビジネス・モデリングとビジネス・プロセス・リエンジニアリングおよびビジネス・オブジェクトの設計のための方法で，1990年から米国のOpen Engineering社で開発されている．OOBEでは，オブジェクトをビジネス・オブジェクト，テクノロジー・オブジェクト，アプリケーション・オブジェクトの3つに分類し，それぞれの構築プロセスを定義している．ビジネス・オブジェクトについては，現在OMG(Object Management Group)のBOMSIG(Business Object Management Special Interest Group)でも検討が行われている．

オブジェクト指向以外の技術を併用した事例 DARPAのDSSA/ADAGE(Domain Specific Software Architectures / Avionics Domain Application Generation Environment)プロジェクトは，航空電子工学ドメインを対象に，ドメイン固有のソフトウェア・アーキテクチャとそれに基づくアプリケーション生成環境の研究を行っている[8] [9]．このプロジェクトでは，ドメイン分析からソフトウェア・アーキテクチャの構築および再利用に至る過程で，オブジェクト指向とそれ以外の様々な技術を併用している．

たとえばドメイン分析では，オブジェクト図とオブジェクトの状態遷移図はGE社のオブジェクト指向ツールであるOMToolを，ディクショナリおよびシステム全体の機能要件を記述するためのデータフロー図，状態遷移図はAscent Logic社のRDD-100(Requirements Driven Design tool)を用いている．また設計上の意志決定に関する決定木，理由付けにはgIBIS(graphical Issue Based Information Systems)を採用している．

4 おわりに

ドメイン分析・モデリングのもう一つの考え方として，なかなか実際的な効果をあげることができない汎用の開発方法論に対するアンチテーゼとして，特定ドメインにフォーカスしたより効果的な開発手法，ツール，部品などを整備しようというものがある．この立場からすると，汎用のオブジェクト指向開発方法論を適用することはナンセンスにさえ思える．

たしかにドメインの知識の中にはドメイン固有の考え方や表記法があって，OMTのそれをうまく適用できないか，適用してもかえって分かりにくいものもある．例えば高度な計算アルゴリズムの詳細をオブジェクトで記述することに意味はない．また現在のOMTの表記法では，同じクラスに属するインスタンスオブジェクト間の関係や，複数の観点からの分類構造が錯綜する時の組み合わせ制約を分かり易く記述できないことがある．

しかしながら，前述したようにオブジェクト指向の考え方を採り入れることで得られるだろうと期待されるメリットは非常に大きく，けして無視できないものである．結局の所，うまく使えるものは利用し，不足している考え方や表記法があるならばそのドメインに適しているもので補っていくのが，賢明な進め方ではなかろうか．

参考文献

[1] Rumbaugh, J., et al.: *Object-Oriented Modeling and Design*, Prentice Hall, 1991.

[2] 中山裕子, 吉田裕之: オブジェクト指向を用いた業務分析についての一考察, 情処研報, 本書.

[3] Coad, P.: Object-Oriented Patterns, Communication of the ACM, Vol. 35, No. 9, pp. 152-159, 1992.

[4] 広本治: 事務処理分野におけるオブジェクト指向開発の事例, オブジェクト指向開発の実践と課題, pp. 187-196, 情報処理学会連続セミナー「ビジネス・プロセス・リエンジニアリングのための最新情報テクノロジーの理論と実践」第4回テキスト, 1994.

[5] Yasuhara, R. : Software Design Method for Communication Systems - Real-Time OOD, Proc. of Telecom '92, pp. 147-151, 1992.

[6] Shelton, R.: Object-Oriented Business Engineering - A Framework & Discipline for Business Object, *Object World Expo / Tokyo '94 Conference Notes*, IDG ワールドエキスポ/ジャパン, (社) 日本オフィスオートメーション協会, 1994.

[7] Shelton, R.: The Distributed Enterprise, *Distributed Object Computing*, pp.13-15, SIG Publications, 1994.

[8] Tracz, W.: A Domain-Specific Software Architecture Engineering Process Outline, *ACM SIGSOFT Software Engineering Notes*, Vol. 18, No. 2, pp. 40-49, 1993.

[9] Tracz, W.: Domain Analysis Technology for Software Re-use, ソフトウェアツールシンポジウム '94 招待講演, (社) 情報サービス産業協会, (株) 情報技術コンソーシアム, 1994.