正規化クラス図（ER図）作成のガイドライン –要のもの・こと間のハッセ図としてのクラス図–

金田 重郎[1,a)]　井田 明男[1]　森本 悠介[1]

概要：UML 静的モデル・クラス図に関するテキストは，数多く出版されており，海外の著名な作者によるテキストの翻訳も販売されている．しかし，これら多くのテキストでは，クラス図のシンタックスの説明はあるが，具体的に「どうやって，クラス図を描くのか」が意外に明示されていない．結果的に，市販テキストを用いただけでは，どの様にクラス図を描くのかを初学者が学ぶことは難しい．この問題点を解決するため，著者らは，クラス図が英語の認知構造を反映しており，それが，日本人初学者にクラス図を縁遠いものにしている一因であるとの仮説を示した．しかし，この既存ガイドラインをもってしても，「クラス図をどう描くのか」を具体的に示すガイドラインとしては不十分である．そこで，本稿では，クラス図の描き方のガイドラインを得ることを目的として，「クラス図とは何か」を明らかにする．具体的には，1対多関連は，時間的前後関係を制約するものであり，結果的に，多対多関連を用いない通常のクラス図は，処理プロセス間の時間的制約を示すハッセ図であることを示す．即ち，クラス図は，人間の頭の中にある概念を取り出すツールというより，対象ビジネスを構成する処理プロセスの時間的制約関係（材料—加工関係）を表現するツールである．クラス図を構成するクラスは，独立型と従属型に分かれるが，独立型では，他インスタンスとは無関係にインスタンスを生成できる．更に，従属型クラスには動作を表現するものが多く，クラス図を，中村善太郎の「要のもの・こと」モデルで理解できる．本稿では，以上の理解に基づくガイドラインを示し，そのガイドラインによって，実務家の間で知られるモデリング例題「花束問題」が分析できることを示す．

1. はじめに

UML 静的モデル（クラス図）の学習は，情報工学系学科では必須内容である．但し，UML に関する著名なテキストを見ても，クラス図のシンタックスについては書かれているが，クラス図を書くための具体的方法論については記述が乏しい．教員にとって，クラス図は学生に教え難いコンテンツである．結果的に，「どうすればクラス図を描けるのか」について，大学で十分に教えることは困難である．

クラス図（ER 図）は，要求分析や設計の基本的ツールである．「クラス図をどう描くのか」という方法論が，整理されなくて良いはずはない．この様な観点から，著者らは，既に，「クラス図は英語である」との分析結果を示し，日英の言語差が，日本人とクラス図を縁遠くしている原因の一つではないかとの問題提起を行った [1]．しかし，クラス図は英語であるとのガイドラインでは，具体的なクラス図（ER 図）を描くための指針としては不十分である．

[1] 同志社大学大学院・理工学研究科・情報工学専攻
Graduate School of Science and Engineering, Doshisha Universsity, Kyotanabe-city 610–0321 Japan
[a)] skaneda@mail.doshisha.ac.jp

そこで，本稿では，ER 図の作成例を分析し，西欧哲学の基本である「存在」に関する議論 [2][3]，中村善太郎の「要のもの・こと分析」の視点 [4] から，クラス図（ER 図）の構成方法についてひとつの試論を提起する．ここでは，従来は，静的な存在である「クラス」間の意味的関係とされて来た関連を，むしろ，対象業務を構成するプロセス間の時間的前後関係，即ち，時間的制約関係ととらえてガイドラインを示す．

以下，第 2 章では既存のクラス図（ER 図）を分析し，第 3 章では，クラス図の構成法をガイドラインとして提示する．第 4 章は，モデリングの専門家にはよく知られたモデリング対象「花束問題」に提案手法を適用した例を示す．第 5 章は，簡単な評価実験を示す．第 6 章はまとめである．

2. 既存クラス図の分析

2.1 クラス図の例

図 1 は，渡辺幸三による ER 図のテキスト [5] に記載された ER 図を，著者らがクラス図に書き直したものである．本稿におけるクラス図は，概念モデリングレベルを想定しているため，手続き（メソッド）については考察対象外と

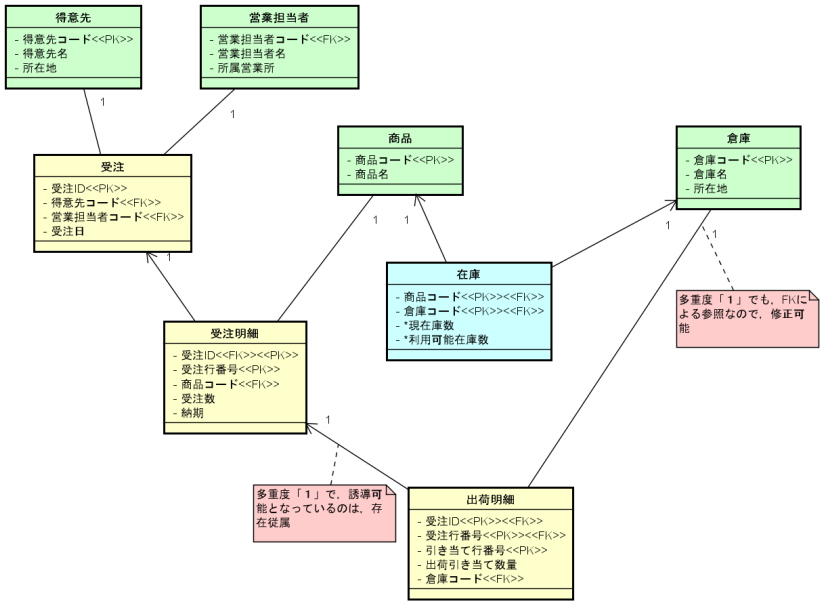

図 1　クラス図のサンプル

する．結果的に本稿の議論は，ER 図に関する議論と考えても良い．図 1 の ER 図を選んだのには，特段の理由はない．高度なスキルを持つモデリング専門家の作成サンプルを引用しているだけである．

但し，本稿では，クラス間の関連として 1 対多関連のみを利用する[*1]．また，議論を簡明化するため，多重度=0..1，は扱わない．本稿のクラス図では，1 対多関連であっても，図の簡明化のため，「多」の側の多重度は省略して描くことがある．

クラス図の分析に入る前に，同じ多重度=1 でも，以下の 2 種類があることを確認しておきたい．

- 多重度=1 で対象インスタンスが 1 個に限定され，挿げ替えできない場合．著者らは，この関係を「**存在従属**」と呼んでいる [6]．「多重度=1」側のインスタンスが存在しないと，「多重度=多」側のインスタンスを生成できないからである．渡辺はこれを「**親子関係**」と呼ぶ [5]．本稿では，UML クラス図の誘導可能性の矢印付き関連を流用して，→ でこの関連を表示する．→ の鏃側が多重度=1 となる．
- 多重度=1 であっても，相手のインスタンスを挿げ替えできる場合．渡辺はこれを「**参照関係**」と呼んでいる [5]．本稿では，この関係を UML の本来の関連である直線で図示する．多重度=1 は明示される．

本稿では，多重度=1 の方向（親子関係であれば → の「鏃」の存在する方向）を「**上流側**」，反対側を「**下流側**」と呼ぶ．つまり，→ の示す方向が，上流側に遡る方向である．

図 1 を見ると，以下の点に気付く．

(1) クラスには，(a) 多重度=1 の他クラスへの関連を持たないクラス（本稿では，これを「**独立型クラス**」と呼ぶ），(b) 少なくともひとつの多重度=1 で他クラスと関連したクラス（以下，本稿では，「**従属型クラス**」と呼ぶ）の 2 つがある[*2]．つまり，上流側に何もないのが独立型クラスである．独立型クラスと従属型クラスとの名称は，著者らの「存在従属クラス図 [6]」の用法に準じている．独立型クラスでは，当該クラスのインスタンスの状況にかかわりなく，新たなインスタンスを追加できる．

(2) 関連は，多重度=1 の方向を向いている有向リンクと解釈すると，図 1 のクラス図は全体として，下から上に向かう，非サイクル性有向グラフ（DAG: Directed Acyclic Graph）である．異なるクラス間には，半順序が成立する．関係モデルの関数従属性では，非推移的関数従属のみで，テーブル内の関数従属性を表現している．本稿が扱う 2 種類の「多対 1」関連 (以下，「$\succ$」で表現する) にも類似の性質があり，B$\succ$A では，クラス B のインスタンスが決定されると，必ず，1 個，クラス A のインスタンスが指定される．その意味では，$\succ$ で表現される関連は，関数従属性と類似している．そうであるなら，$\succ$ の関係にも，非推移性が期待される．クラス図を正規化して，保守性を向上させる為には，矢印付き関連は，直近のインスタンス (非推移的なインスタンス) にリンクしている必要がある．図 1 のクラス図でも，(A,B,C をクラスとして)，A$\succ$B$\succ$C

[*1] 1 対 1 は排除されない．多対多は使わない．多対多は，そのままでは関係モデルにマッピングできない為である．

[*2] 椿のリソース型エンティティがこの独立型クラスに近く，椿のイベント型エンティティと在庫型エンティティは，本稿の従属型クラスに近い [7]．尚，椿は他のエンティティとして，「断面」「要約」を示しているが，これらは VIEW，すなわち，派生属性で構成されるクラスなので，概念モデリングを対象とする本稿では，議論から除外する．

の有向な関連があれば，A≻C の有向な関連として表現するべきでは無い．これは，明らかに，既約な DAG であり，ハッセ図と呼ばれるものに等しい[*3]．

(3) 図 1 の例からも明らかな様に，クラス図は，全体として，最上部に独立型クラスが並べられ，その下には，従属型クラスが次々と業務の進捗に応じて展開する．従属型クラスの大半は，ある時点での動作を記録する内容である．時間経過とともに，属性値が変化するクラスは少ない．本稿では，あるタイミングにおけるデータ状態を保存するクラスを「**時点型**」，インスタンス生成後に，属性値が変化するクラスを「**期間型**」と呼ぶ．従属型クラスの大半が，動詞性名詞であり，過去のある時点での動作の記録（時点型）である．

尚，本稿では詳細な議論を行わないが，クラス図を描く際の，ひとつの課題は，関連の多重度=1 を親子関係にするのか参照関係にするのかである．図 1 の場合，「出荷明細」から「受注明細」へのリンクは変更される様なものではない，親子関係が妥当だろう．一方，「商品」の引き当ての対象である「倉庫」はあとから修正されることもあるので，参照関係を利用しているものと推定される．他のクラスを見てみたい．「受注明細」から見て，「受注」は，「受注明細」を束ねる働きをしており，変更できない関連であり，親子関係となる．一方，「商品」は処理に組み込まれるが，変更される可能性があり，参照関係の関連となる．「在庫」は，「商品」／「倉庫」の組み合わせが変更されると別の存在になるので，親子関係とする．「受注」は，担当者と相手先は変更される可能性があるので，参照関係としている．親子関係か参照関係かの議論は，「依存先へのリンクが後から変更される」可能性があるか否かに影響される．

親子関係の関連では，多重度=1 側のインスタンスの挿げ替えはできない．但し，親子関係の多重度=1 側にリンクされた上流側インスタンスを挿げ替える場合には，(1) 当該インスタンスよりも下流側にある（つまり，時間的に，当該インスタンスの生成後に，リンクさせて追記された下流側）全インスタンスを消去し，(2) 当該インスタンスから変更先インスタンスに制御を移し，(3) 新しい変更先インスタンスから下流側の該当するインスタンスを全て再構築すれば良い．つまり，参照関係におけるリンク先インスタンスの挿げ替えと同じことが，親子関係を用いて，理論的には実現できる．

以下の節では，上記の観察結果に対して，理論的一般化を行う．

2.2 哲学的視点から見た独立型クラスと従属型クラス

最初に問題となるのは，独立型と従属型の 2 つのクラスがそれぞれ何か，どうやって選ぶのかである．そもそも，（オブジェクトが）「存在する（ある）」とはどの様な意味なのであろうか．周知の様に，この問題は，哲学の根本原理であり，プラトン，アリストテレスの時代から議論されて来た [2]．

哲学では，「ある」という概念は 2 つに分裂する．質料（材料）によって規定される事実存在と，形相（質料を用いて作られた形）によって規定される本質存在である．事実存在は，“～ がある” という形で表現され，その事物が実際にあること，知覚可能であることを意味する．本質存在は，“～ は ～ である” という形で表現され，その事物の本質を意味する [8] [*4]．

図 1 の独立型クラスを見てみると，「得意先がある」「商品がある」と表現できる．独立型クラスは日本語の「もの」に近い．事実存在に該当すると考えたい．次に，従属型クラスを見てみる．従属型クラスは，「受注である」「出荷である」となる．プラトン，アリストテレスによれば，本質存在（形相）に対しては，設計図の様なもの（概念）に従って，材料（質料）を組み合わせた「制作物」と捉えていた様である [9]．プラトンの時代から，「ある」には「ものとして存在する場合」と「材料として事実存在を利用して制作された場合」の 2 種類あると言っている点は興味深い．

但し，上記の哲学的見方をクラスの種別判別にそのまま使うのは難しい．「他インスタンスの存在とは無関係にインスタンスを生成できるのが独立型クラス」と判断した方が分かり易い．

2.3 要のもの・こと

前述した通り，従属型クラスはしばしば動詞性名詞であり，ひとつのプロセスの実行を記録している．後述する様に，クラス図は，クラス間の前後関係を意味するものなので，非動詞性名詞の従属型クラスでも，本稿の手法では，プロセスとして表現することとする．具体的には，クラス名が非動詞性であっても，その名詞を生成するプロセスにすれば良い．例えば，「部品表を作る」である．非動詞性の名詞が表すオブジェクトを生成したタイミングがあるはずであり，プロセスの前後関係をそれで表現してゆくことになる．

以上の分析から，独立型クラスは「もの」，従属型クラスは「こと」である．独立型クラスを材料として，次々と加工が行われている．これは，中村の「要のもの・こと」モデルである．概念モデリングにおけるクラス図は「もの」と「こと」を描いている [4] と見なして良い様に思われる．

[*3] 本稿における「ハッセ図」という用語は，ハッセ図本来のグラフ表記とは関係なく，抽象概念としての有向非循環グラフの推移簡約を指す．

[*4] 日本語よりは，英語の方が，ここらの差が分かり易い．本質存在は，例えば，“This is a ～” における be 動詞であり，コピュラ (copula) である．これに対して，事実存在は，“There is a ～.” の be 動詞である．There are goods. には，商品がそこに存在しているとのイメージがあるが，These are goods. には，商品があるというイメージよりは，これらが商品とのイメージが強い．

情報処理システムに対しても，あたかも工場の様に，入力（材料）が加工されて出力（製品）となるプロセスとして捉えることは一般的である．図 1 のクラス図が，上から下へ向けて，つぎつぎと多段階に重畳しているのは，当然となる．例えば，受注明細には，商品と受注の 2 つのインスタンスへのリンクがある．受注があり，しかも，商品が決まっていないと，受注明細は生成できない．

また，「要のもの・こと」であることも重要である．本稿で議論しているのは，概念モデリングのためのクラス図であるから，VIEW や制御のためのオブジェクトは分析対象ではない．その意味でも，要のものへ，要の加工が処理されていなければならない．

2.4 1 対多関連が持つセマンティクス

本稿のクラス図では，関連として，1 対多関連を前提としている．この場合，2 つのインスタンス間にリンクがあるとして，インスタンス相互の生成タイミングの制約関係について考えて見たい．但し，「0..*」と「1..*」の「多」を考える．

定理 1. まず，多重度=1 側について考える．クラス A から関連がクラス B に向けて定義されており，クラス B 側の多重度が 1 であるとする．クラス A のインスタンスが成立するためには，少なくとも，クラス B のインスタンスが既に存在しているか，同時に生成される必要がある．

Proof. 親子関係では，クラス B のインスタンスが存在しないと，クラス A のインスタンスは生成できない．クラス A のリンク先インスタンスは，最低限度，1 個存在する必要がある．尚，参照関係の場合でも，リンクを最初に張る時の要件は，親子関係と同様である．切り替える際には，それまでのリンク先とは別のインスタンスを選択することになるので，選択先の新たなインスタンスについては，上記の定理は成り立たない．しかし，もともと，最初の 1 個目については，上記の定理が成立する必要があるため，結局，親子関係でも，参照関係でも，上記定理は成立する． □

定理 2. クラス A 側の多重度が「0..*」の場合には，クラス A のインスタンスの存在は，クラス B のインスタンスの存在に対して，必須ではない．また，1 対多関連の場合，クラス A のインスタンスが，クラス B のインスタンス生成時以降に，つぎつぎと追加されることもあり得る．即ち，「多」側からは，クラス A と B の間の時間的前後関係は規制をされない．

Proof. 自明のため省略．但し，これにより，1 対多関連がある場合の関連両端のインスタンスの生成タイミングは，多重度=1 側のみで制約されることが分かる． □

上記分析から，1 対多関連の場合には，1 側が親子関係でも参照関係でも，「多重度＝ 1 」側のインスタンスの生成タイミングより以前に，多重度が多の側のインスタンスは生成できない．

2.5 ハッセ図としてのクラス図

もう一つの観点から，クラス図を解釈したい．前節の分析から，1 対多関連を定義することは，自動的に，2 つのインスタンス間の時間的前後関係を規定する．結果として，クラス図はクラス間の時間的半順序関係（加工順序）を反映したハッセ図（既約の DAG）であることになる．この場合，多重度=多から 1 の方向に向けて，関連は方向性を有している．この点を論じる前に，若干の定理を示す．

定理 3. 少なくとも 1 対多関連の「1」の多重度が「1」であり，「0..1」ではない条件下では，あるインスタンスを含んで，リンクが閉ループとなることはない．従って，少なくともクラス図は，サイクルを持たない有向グラフ DAG である．

Proof. あるインスタンスを経由して，ループを形成しようとしてリンクを追記するときには，ループ先にあるリンク先のインスタンスが生成されていない（上流側インスタンスが生成されていない）ので，多重度= 1 に違反する．このため矛盾である． □

定理 4. 1 対多関連を前提とするなら，どのようなクラス図でも，関連の方向性には終了があるはずであり，結果的に，どの方向にも多重度= 1 の関連が存在しないクラス，即ち独立型クラスと，一般的な時点型／期間型の従属型クラスに分離する．

Proof. 自明のため省略 □

既約化することにより，クラス図はハッセ図となる．クラス図を描く場合には，A,B,C をクラスとして，「$\succ$」を多重度=多から多重度=1 への方向性を持つ関連とするとき，A$\succ$B$\succ$C となる関連と，A$\succ$C となる関連が並存することは許されず，A$\succ$C の関連を消去して，既約化を図るべきである．ハッセ図は，もともと，プロセスの時間的前後関係を半順序として表現するために利用されてきた．このため，クラス図も，プロセス間の時間的前後制約を表現するツールである[*5]．

以上の分析から，クラス図作成の示唆を得る．クラス図は既約な DAG(ハッセ図) として設計されるべきであり，1 対多関連しかないなら，クラス図は元々 DAG であり，既約にすればハッセ図となる．関連は，「2 つのクラスが，それらのインスタンスの存在の時間的前後関係で制約を受ける（これを因果関係の制約と呼ぶ）」と理解して接続するかどうかを判断すべきである．原因と結果，材料と加工品

[*5] この論理に従えば，他のモデリング手法，例えば，手島による概念データモデリング [14] においても，静的モデルを描く前に動的モデルを描くべきということになる．

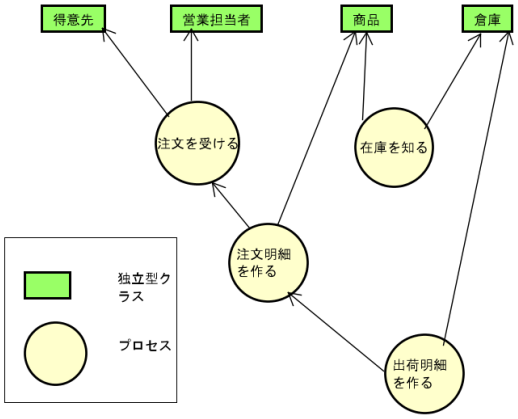

図 2　Resource-Process Dependency Diagram の例

と理解しても良い．逆に言えば，時間的順序制約を持つクラス相互間には，関連を定義するべきであり，時間的前後関係が定まらないクラス間では関連を引いてはならない．UML では，規定上，「意味的関連性があるところに関連を引く」としているが [10]，それは，本論文の理論的解析からすると，過度に曖昧な表現であり，むしろ，因果関係・時間関係の制約がある所に関連を引くべきである．

3. クラス図作成のガイドライン

3.1 ガイドライン

以上の分析から，以下の 4 つのガイドラインを適用して，クラス図を作成する．但し，モデリングの常として，以下の 4 ステップを一度だけ適用すれば良いわけではなく，完成するまでに，何度もガイドラインを適用して，検証することになる．

【STEP1:要の「もの」を抽出する】　概念モデリング段階のクラス図は，中村善太郎の「要のもの・こと」の見方で捉える．業務プロセスの中では，VIEW ではない，要の材料として利用されるオブジェクトを抽出して，独立型クラスとして，図の最上部に列挙する．物理的存在としての「もの」に近いイメージであり，「こと」により加工されるインプットである．独立型なので，他インスタンスの存在に無関係に生成し得る．この独立型クラスのオブジェクトは，後述のリソースープロセス依存グラフ（RPDD: Resource-Process Dependency Diagram) でも，インプットとして表示される．

【STEP2:業務プロセスの抽出】　従属型クラスは，基本的に，中村の言う「こと」である．従って，UML クラス図を，対象ビジネスの処理内容詳細も分からずに，いきなり記述することは本質的にできない．必須なのは，対象ドメインにおける「こと」，即ち処理プロセスとして，どのような処理が行われ，その処理相互に，どのような因果関係（時間順序）があるかの確認である．即ち，クラス図を描くためには，業務に含まれる処理プロセスの依存関係（原因結果関係）を分析者は知っていなければならない．

本稿では，クラス図作成の前に，まず，プロセスの抽出と時間関係の明確化のためにリソースとプロセス間の依存関係を示すグラフを用いる．本稿では，これを，**リソースープロセス依存グラフ**（RPDD: Resource-Process Dependency Diagram) と呼ぶ．図 2 は，図 1 のクラス図に相当する RPDD の例である．表記法として DFD[11] を流用している部分があるが，図の意味は大きく異なる．

図 2 の様に，RPDD の最上部には，処理の材料となるオブジェクトを並べる．STEP1 で選択した独立型クラスである．ここでは，DFD の源泉/吸収の四角と同じ記号を用いている．同様に，楕円が，加工を示すプロセスである．リンクの矢印は，DFD の情報の流れとは逆で，矢印の先の材料がなければ，プロセスは動けないことを示す．尚，図 1 の本稿のクラス図では，親子関係のみに矢印を用いていたが，RPDD では，1 対多関連の多重度=1 を矢印の鏃で示している．

ここで，一つの下流側のプロセスに対して，上流側プロセスが複数個存在することは許されない．矢印リンクは一種のポインタなので，あるインスタンスから上流方向に向けてリンクを辿る毎に，インスタンスを 1 個だけしか，特定できない．その結果，一つの下流側プロセスに対して，上流側の複数のプロセスが該当する様な場合，リンクを張ることができない．1 個のインスタンスから上流方向に複数のリンクがある場合，アプリケーションプログラムは，FK（Foreign Key）による検索等の対策をして，このリンクを辿る．

【STEP3:ハッセ図化とクラス図の生成】　上記 RPDD において，ネットワーク構造全体を，既約化してハッセ図となる様に描く．このハッセ図の構造はそのままクラス図にコピーされる．具体的には，2 つの「こと」（すなわち，従属型クラス）間に，時間的制約関係がある場合にのみ，（上流側のクラスのインスタンスを，当該クラスのインスタンスが加工する材料とみなし得る場合）2 つのクラス間にリンクを引く．逆に言えば，2 つのプロセス間に，時間的前後関係が存在する限り，リンクは引かなければならない．完成した，ハッセ図に従って，クラス図を描く．ここで，RPDD のプロセスは原則，時点型クラスとなる．RPDD のプロセスで，期間を持つ場合には，期間型クラスとする．

【STEP4:属性の設計と正規化】　クラスの属性値のライフタイムは，当該インスタンスのライフタイムと同一であることが必要 [12] である．また，クラス生成に際しては，Codd の関係モデルの概念をそのまま準用して，各クラスの属性は，当該クラスの PK(Primary Key) からみて，少なくとも BCNF(Boyce-Codd Nor-

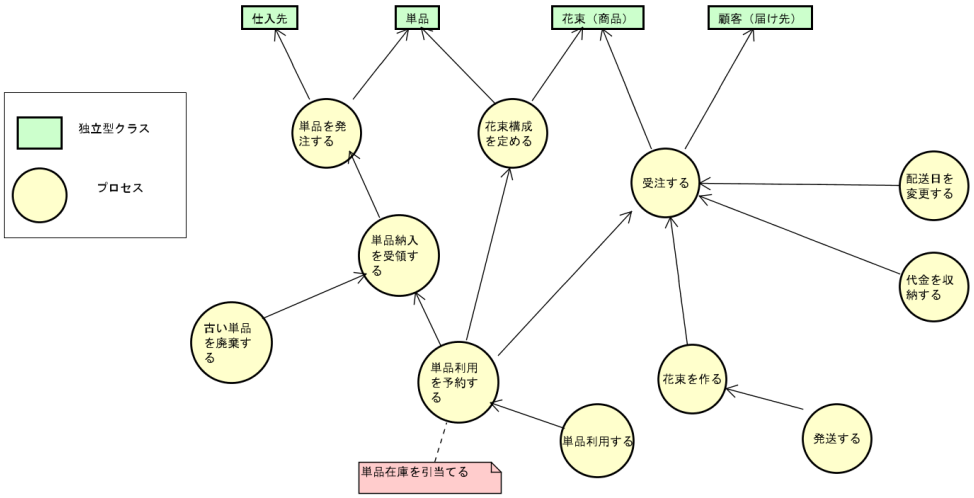

図 3　花束問題 RPDD 作成例

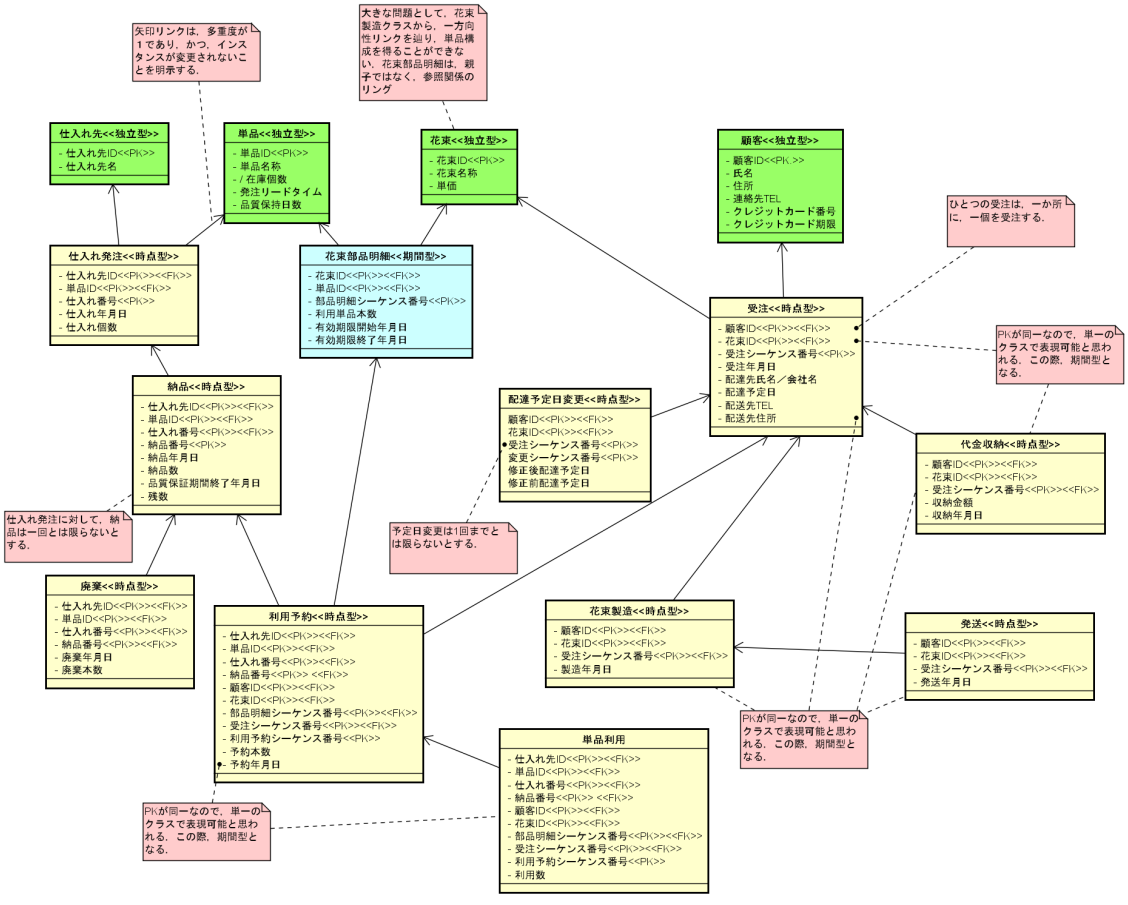

図 4　花束問題クラス図作成例

mal Form)，該当する場合には，第 4 正規形，第 5 正規形が要求される[*6]．

これらの視点は，クラス図の正規化において重要であり，属性を追加，検証して，クラス図を完成する．

尚，RPDD はあくまでも，業務処理プロセスの時間的制約関係を描いたものである．一方，クラス図はオブジェクトに着目してモデルを表現する．従って，RPDD の各処理プロセスが，結果的に，常に，クラス図のクラスに逐一対応する必要はない．複数の処理プロセスを，ひとつのクラスにおける属性値の変更として表現することは，十分に考えられ，クラス図もコンパクトになる．

3.2　提案手法の新規性と適用範囲

本提案手法は，「クラス図の 1 対多関連において，関連

[*6] 存在従属クラス図 [6] と同様，提案手法適用により，第 4 正規形迄が自動的に満たされる可能性があるが，本稿では未検討である．

の両端に存在するクラスの間には，時間的前後関係を制約として持たざるを得ない」との，著者ら独自の視点に立脚する点に最大のオリジナリティがある．この様に認識すると，クラス図は半順序関係（ハッセ図）と見なされる．ハッセ図としてのクラス図の構造は，原料（インプット）から次々と加工して下流側の処理へと回してゆく，中村の「要のもの・こと」モデル [4] と親和性が高い．そこで，「要のもの・こと」の視点の下に，RPDD の形で，要のことの流れを明確化し，それをそのままクラス図の構造として写し取るものである．これらの枠組み自体に新規性があると考えている．尚，本稿のガイドラインを正しく適用したならば，クラス図（ER 図）は，関係モデルに変換後[*7]，正規化されたものとなる．ハッセ図であるため，冗長な関連や派生属性は排除されているからである．

RPDD は，一見すると DFD に近い．しかし，RPDD と DFD は異なっている．DFD は表面的な業務の流れを表現するのに対して，RPDD は，中村善太郎の意味で「要のもの・こと」のみを記述しているからである．表面的な業務の流れではなく，本質的に必須の処理のみから RPDD を描く必要がある．この問題意識自体は，手島による概念データモデリング [14] に近い．手島の手法は，「要のもの・こと」のみを取り出した DFD と言う側面を持っている．但し，手島は，「要のもの・こと」は導入しているが，多重度をプロセスの前後関係としてモデル化に利用する問題意識はない[*8]．

本提案手法の適用範囲については，実用レベルのアプリケーションを対象として，評価・検討を必要とする．現状では，操作（メソッド）は検討に含めていない．また，同一クラスに関連のリンクが戻る自己参照型の関連については，リンクは同一インスタンスに戻っている訳ではないので，半順序関係を乱していない．自己参照型の関連も許容すると考えているが，今後の適用性の検討を待つ必要がある．更に，本稿では除外した，多重度「0..1」については，もともと，「存在しなくても良いリンク」なので，時間的前後関係の制約を含意できない．このため，本提案のガイドラインを適用した後，追加的にモデルに取り込めば十分と考えている．但し，本稿の提案はあくまで，ガイドラインである．このガイドラインのみでクラス図が書けるわけではない．例えば，is-a 関連を用いたモデリング等が，本提案のガイドラインから自動的に出てくるわけではない．しかし，それでも，モデリング担当者から余計な迷いを幾らかは取り去ることができると考えている．

尚，業務システムの処理を，「インプットを加工してアウトプットする生産工場モデルで捉える」ことが多い．しかし，本稿の主旨に沿えば，それは，「工場におけるの製造ラインの構造をたまたま情報システムに当てはめた」ものではない．データ中心設計を行うこと自体が，本質的に，そのようなソフトウェアの構造をもたらしている可能性がある．

4. 実践的課題への適用

本章では，分析対象をより実践的なものに改めて，上記のガイドラインを適用した結果を示す．課題は，佐野により提案されている「花束問題」である．単品の花を仕入れて，商品である花束を，単品の組み合わせで作る．但し，注文は，1 回にひとつの花束を，1 か所に配送する形で受けるのみである．また，単品の販売はなく代金はクレジットカードのみである．花束問題の記述は，文献 [13] を参照されたい．本稿では，付録 A.1 として，花束問題の仕様の中で，エンティティレイヤーの記述に相当する部分のみを添付している．

4.1 STEP1:独立型クラスの抽出

最初に仕様書から，材料となる独立型クラスを取り出す．この業務で加工される原料であり，当該クラスの他インスタンスの存在とは無関係に，インスタンスを追加できるものである．明らかに，仕様書から，以下の独立クラスがある．

仕入れ先，単品（花），商品（花束），顧客

4.2 STEP2:業務プロセスの分析と RPDD 化

材料を加工する要のプロセス「こと」を列挙してゆく．花束問題において，仕様書を見ても分からなかったのは，「商品（花束）」を作成する時点での，材料である単品の在庫をどう管理するのかであった．このため，以下の図 3 の RPDD では，「単品」の利用予約と，実際の単品利用を異なるプロセスとして表現している．

RPDD 作成では，単純に時間的前後関係のみを見てリンクを張れば良いわけではない．例えば，図 3 において，「単品利用を予約する」プロセスは，「花束を作る」プロセスの前に実施する必要がある．本来なら，「花束を作る」から，

[*7] クラス図（ER 図）のリレーショナルデータベーススキーマへの変換は，増永良文：リレーショナルデータベース入門 [第 3 版]–データモデル・SQL・管理システム・NoSQL–，サイエンス社，2017 年 2 月，1.6 節の記述に従う．

[*8] あるいは，「本提案手法では，クラス図作成の困難性を，RPDD に転嫁しており，『要のもの・こと』の下に業務を見直して RPDD を作成するのが困難ではないのか？」との見方があるかもしれない．もともと，日本語は，名詞（オブジェクト）よりも，動詞を中心とする言語である [15][16]．作文一つ取っても，我々日本人は，時系列的に文章を並べて最後に結論を書く教育を受けて来た [17]．日本人にとって，（英語ではそれが出てこないと文章が始まらない）名詞（クラス）で考えるよりも，むしろ，時間を追う分析をした方が，考えやすい事を期待している．逆に言えば，従来，何も構造的縛りがなかったクラス図に対して，ハッセ図で構造に縛りをかけ，時間軸方向にクラスを並べることをガイドして，初学者の迷いを無くすのが，本提案手法の主旨である．但し，1 対多の関連のみを考えるという条件下では，全てのクラス図はハッセ図である．この縛り（ガイド）はクラス図に何等かの制限を持ち込むものではない，というのが本稿の主張である．

「単品利用を予約する」へ向かって，矢印リンクを張りたい．しかし，張ることはできない．以下，その理由を確認する．

図 3 の左半分は「単品」の流れを汲み，右半分は「花束」の流れを汲んでいる．一つの花束には，複数種類の単品を必要とする．結果として，左から右への矢印リンクはターゲットが花束 1 個なので張ることができる．しかし，粒度の違いから，右（花束）から左（単品）へ矢印リンクを引くことはできない．クラス図 (図 4) の段階で考えると，花束を作る場合には，「花束」のインスタンスの ID を FK として持つ「花束部品明細」のインスタンスを，検索する必要がある．これを存在従属関係の関連における矢印の方向のみで辿ることはできず，AP（アプリケーションプログラム）は，矢印の方向とは逆方向にリンクを辿る必要がある．

4.3 STEP3:ハッセ図化とクラス図化

RPDD の段階でハッセ図となっていることが望ましいが，ハッセ図になっていない場合には，既約になっているかどうか，即ち，プロセス間のリンクが最短のプロセスに行っているか否かを見直して，既約とする．完成した RPDD の構造は，そのまま，クラス図に写し取る．この場合，従属型クラスは基本的に動作性の名詞で表現され，時点型となる．但し，RPDD の段階で，あらかじめ期間型が妥当とされていたプロセスは，期間型のクラスにより表現する．

各クラスの属性のライフタイムは，その属性が所属するインスタンスのライフタイムと一致する必要がある．この際には，クラスの PK と属性の関係は，正規化が行われていなければなければならない．図 4 には，得られたクラス図を示す．尚，クラス図の中で，花束部品明細のみ，期間型を用いている．これのみがイベントではなく，定常的な状態を表現しているためである．

表 1 評価実験の結果

被験者	1 回目演習結果（RPDD を教える前）		2 回目演習の結果（RPDD を教えた後）	
	利用クラス数	クラス図の完成状態	利用クラス数	クラス図の完成状態
A	7	×	13	○
B	10	△	10	○
C	11	×	14	○
D	9	×	16	○
E	6	×	7	△

×：未完成，△：周辺分を除いて完成，○：完成（基本的な構造が満足されている）

5. 評価実験

本提案のガイドラインの有効性を検証するためには，「初学者」を対象として，評価実験を行う必要がある．「初学者」とは，一通り UML について学ぶ中で，クラス図（ER 図）の作成経験はあるが，講義で演習した程度のスキルの学生を想定している．企業で初めてモデリング研修を受ける新入社員と考えても良い．そこで，情報工学系院生（1 年次生 4 名/2 年次生 1 名）を「初学者」と見なし，修士の授業時間を用いて，小規模な評価実験を行った．彼ら/彼女らは，学部 3 年次生でクラス図について既に習っている．しかし，一通りの練習しか経験していないので，モデリングスキルは十分ではない．正直なところ，5 名中 4 名は，クラス図を自由に書けない．

評価実験では，まず，1 回目の授業時間中に，課題の業務内容を詳しく説明した．その後，クラス図を作成してもらった．課題は，固定資産税業務の業務仕様を簡単化したものであり，付録 A.2 に業務記述を示す．次に，翌週の 2 回目の授業で，本提案手法の説明を行い，前週に作成したクラス図を修正してもらった．但し，1 回目，2 回目ともに，著者らは，学生からの質問には適宜答えた．

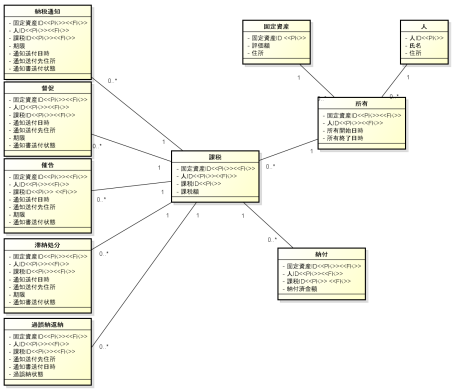

図 5 税金問題演習結果の例 (1)

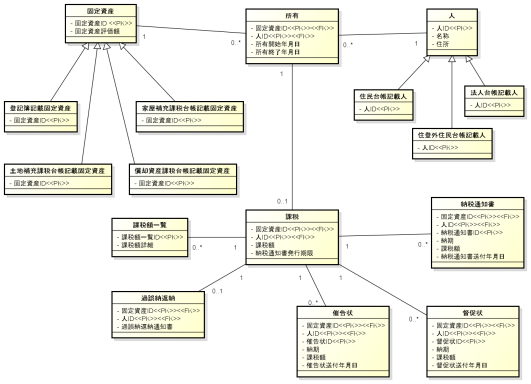

図 6 税金問題演習結果の例 (2)

表 1 に，提出されたクラス図の状況を示す．この例では，独立クラスは「人」と「固定資産」であり，ここから出発して RPDD を作成する必要がある．1 回目の授業では，明らかに「どこから手を付けてクラス図にしたら良いのか分からない」との迷いが 5 名中 4 名に観察された．他 1 名は自力でかなり描いていたが，完成したとは言えなかった．実

際，1 回目に作成されたクラス図中 1 名では，クラスの箱が書いてあるだけで，関連リンクが書かれていなかった．これに対して，2 回目では，全員，一定以上の完成度のクラス図となった．

2 回目の提出結果の一部を，図 5，図 6 に示す．誤りや不足クラスが見られる．しかし，その基本的構造は作成者に無関係に類似している．完成度の差異は，is-a 関連の使い方の知識の有無などにも依存しており，本提案のガイドラインのみで，機械的にクラス図が完成する訳ではない．時間数から見て，2 回目の完成度が高いのは当然である．仮にそうだとしても，少なくともこの評価例からは，担当者のスキルに関係無く，一定のクラス図が作成可能であることが示唆される．

6. 終わりに

本稿では，1 対多関連（1 対 1 を含む）のみを利用したクラス図は，業務処理プロセス相互の時間関係制約を表現するハッセ図であるとの見方を提起した．ハッセ図であるため，各クラスの正規化だけではなく，ハッセ図としての既約性が重要であることとなる．これによって，一種の「要のこと・もの」モデルとして，概念クラス図を構築できることを理論的に示した．また，これに基づくクラス図作成のガイドラインを提案した．

これらの理論的分析から以下のことが導出される．

(1) クラス図は，オブジェクト間の動的なリンクの時間軸を捨象して関連を定義した静的なものと言うより，時間的，即ち，因果関係にもとづく時間的な前後制約を表すものである．結果として，対象業務における業務プロセスとその前後関係が詳細まで分からないと，クラス図は書けない．

(2) 初学者には ER 図（クラス図）は難しいとされてきた．しかし，この分析から見る限り，初学者がクラス図を描けないのは，業務プロセスの詳細が分からない為であり，頭の中にあるクラス図（概念）を取り出すのが下手だからとは断定できない．この点は，実際に，教育内容に本稿の手法を入れて，評価する必要がある．

(3) クラス図を 1 対多関連で描くことにより，RDB(関係モデル) へのマッピングが容易となる．しかし，1 対多関連は，もう 1 つの効果として，対象ドメインにおける個々の業務プロセス間の順序関係を表現するハッセ図としての機能をクラス図に与えている．

(4) データフローダイアグラム (DFD) は優れた手法であるが，データフローに，VIEW レベルの内容まで混入する．本稿で提案したプロセス間のデータフローとしてクラス図を描くアプローチ（RPDD を用いる）は，結果的に，概念モデリングレベルの情報の流れのみを取り出してモデリングしていることになる．本稿の手法は，業務要求を分析する際における，DFD(データ・フロー・ダイアグラム) の限界を補完する，ひとつの方法になる可能性がある．

参考文献

[1] 金田重郎，井田明男，酒井孝真，熊谷聡志：日本語仕様文からの概念モデリングガイドライン－行為文と関数従属性に基づくクラス図の作成－，電子情報通信学会論文誌 D，Vol.J98-D，No.7，pp.1068-1082，2015 年 7 月
[2] 木田元：私の哲学入門，講談社学術文庫，2014 年 4 月
[3] 仲正昌樹：ハイデガー哲学入門–『存在と時間』を読む，講談社現代新書，2015 年 11 月
[4] 中村善太郎：もの・こと分析で成功するシンプルな仕事の構想法，日刊工業新聞，2003 年 11 月
[5] 渡辺幸三：データモデリング入門」日本実業出版社，2001 年 7 月，p.150，図 2
[6] 井田明男，金田重郎，熊谷聡志，藤本明莉：存在従属性に着目した論理要件ロバストなドメインモデルの作成-ドメインクラス図をユビキタス言語として用いるために-，情報処理学会論文誌，Vol.56, No.5, pp.1340-1350，2015 年 5 月
[7] 椿正明：名人椿正明が教えるデータモデリングの技，翔泳社，2005 年 11 月
[8] 仲正昌樹：前掲書，p.53，V 「現存在」と「存在」
[9] 木田元：前掲書，p.206，プラトンの制作的存在論
[10] OMG Unified Modeling LanguageTM (OMG UML), Infrastructure Version 2.4.1, 2011 年 8 月, 11.3.1 Association
[11] トム・デマルコ (著), 高梨智弘 (訳), 黒田順一郎 (訳)：構造化分析とシステム仕様＜新装版＞，日経 BP 社，1994 年 9 月
[12] 大木幹雄：ライフタイム分析に基づくクラス構造抽出の定式化と構造に関するデザインパターンの抽出実験，情報処理学会論文誌 45(6), pp.1554-1568, 2004 年 6 月
[13] 佐野初男　花束問題：（2018 年 5 月段階ではリンクが切れています）
https://www.benkyoenkai.org/2015/01/v12.html
佐野初男 花束問題解説：
https://drive.google.com/file/d/0BykMR9FfFyLOclV5VnFMVWJtMEU/view
[14] 手島歩三, 松井洋満, 南波幸雄, 安保秀雄, 小池俊弘：働く人の心をつなぐ情報技術―概念データモデルの設計，白桃書房，2011 年 6 月
[15] 三上章：象は鼻が長い，くろしお出版，1960 年 10 月
[16] 金谷武洋：日本語に主語はいらない，講談社選書メチエ，講談社，2002 年 1 月
[17] 渡辺雅子：納得の構造―日米初等教育に見る思考表現のスタイル―，東洋館出版社，2004 年 9 月

付　　録

A.1 花束問題仕様書（抜粋）[13]

文書化されているユーザ要件

///商品と在庫///

① 花束の組み合わせは事前に「商品」として決めうちされている．1 個の商品あたり，どの「単品（後述）」がどれだけ必要かも決められている．シングルレベルしかない部品表のようなもの．単品の在庫も含めて，保管場所は 1 箇

所で，これが増える予定もない．

② 花束の材料となるそれぞれの花は「単品」として管理される．「単品」はそれぞれ特定の仕入先から購入され，単品毎に品質維持可能日数が決められている．購入後にその日数を超えると結束には利用できずに廃棄されなければならない．尚，受注・出荷されるものは「商品」のみであって，単品がそのまま出荷されることはない．

///得意先と受注・出荷///

③ リピータを期待するので，得意先（個人のみ）情報を管理したい．届け先は毎回違う可能性があるが，前回の受注情報から届け先を簡単にコピーできるような機能は欲しい．

④ 1回の受注で，1箇所の届け先に対する1種類の商品1個を，「届け日」と「お届けメッセージ」，「お届け先電話番号」とともに受け付ける．出荷日は届け先に関係なく届け日の前日とする．

⑤ いったん受注を受けてから，届け日の変更が要望されることがある．その際には可能な限り変更に対応できるようにしたいが，指定日に出荷変更できないようならばその旨を顧客に直ちに伝えられるようでなければならない．

⑥ 単品を結束して商品（花束）にするための工程は十分に効率化されていて，材料さえあれば一瞬で結束可能とみなしてよい．したがって，出荷日当日に結束指示すれば出荷可能である．

///発注と入荷///

⑦ 単品を発注する際，単品毎に発注リードタイム（入荷されるまでにかかる日数）が異なる．発注リードタイムさえ越えていれば，どんな将来の入荷向けの単品も発注可能だし，入荷日の変更要望も受け付けてもらえる．

⑧ 「単品」毎に購入単位数が決まっている．たとえば，５０本必要だとしても，購入単位が１００本ならば１００本買わなければならない．尚，仕入先の供給能力は十分かつ，納期も正確とみなしてよい．

⑨ 発注の判断は，在庫推移（日別の在庫予定数）をみながら人間が行う．したがって，自動発注処理を考える必要はない．

///代金の扱い///

⑩ ＩＤの登録の際にクレジットカード情報を入れるため請求や入金に関しては考慮する必要はない．

⑪ 入荷の実績情報があれば処理できるので，仕入先への支払に関しても考慮する必要はない．

以上

A.2 税務処理問題（簡易版）

固定資産税システム仕様書（簡易版）

(1) 固定資産税は，1月1日現在の当該固定資産の所有者に課する．この場合，課税対象の所有者を納税義務者と呼ぶ．
(2) 所有者は法人，自然人の両方があり得る．但し，所有者は，ひとつの固定資産にひとりである．
(3) 所有者は，住民台帳，法人台帳あるいは，住登外住民台帳のいずれかに住所・氏名が記載されている．
(4) 固定資産の共有は認めない．
(5) 固定資産には，固定資産評価額が決められている．課税額は，この固定資産評価額により定められる．課税額詳細は，課税額一覧表に記載されている．
(6) 法が定める納税通知書発送期限までに，納税通知書を作成・送付する．
(7) 納税通知書に関しては，課税時点における所有者が納税義務者であり，納税義務者に納付通知する．
(8) 納税義務者は，納付を行うことができる．
(9) 納税通知を送っても，納期までに，未納付，あるいは，納付金額不足の場合には，督促状発送期限までに，督促状を送付する．
(10) 督促状を送っても，納期までに，未納付，あるいは，納付金額不足の場合には，催告状発送期限までに，催告状を送付する．
(11) 催告状を送っても，納期までに，未納付，あるいは，納付金額不足の場合には，滞納処分期限までに，滞納処分による強制執行（たとえば，預金の差し押さえ）を行う．同時に，滞納処分通知を行う．
(12) 納付金額が過誤により，課税額を上回る場合には，過誤納返納を行う．過誤納返納通知書を送付する．
(13) 納税通知書・督促状・催告状・滞納処分通知書の送付先は，住民台帳表・住民登録外表に記載されている現住所とは別の住所を本人の申し出により登録できる．この住所は，税種毎に登録できる．
(14) 納税通知書 → 督促状 → 催告状 → 滞納処分の順番は，前段階がないと後段階を出せないとするものではなく，税額の未納・不足によって，各処理は起動される．

以上