データベースシステム 103－6
（１９９５．５．２５）

JDMF/M-92の振る舞いモデル機能拡張 (JDMF/BM)

佐藤　英人

東京国際大学商学部

JDMF/BMは、静的な概念スキーマ記述のためのモデル機能JDMF/MODELに、動的振る舞い記述を加えたもので、属性定義とイベント受取定義の連携により、振る舞いを宣言的かつ簡潔に表現できる。本稿では、このJDMF/BMの考え方とオブジェクト定義の構文を概説し、次いで、それを用いた分析の枠組である協調場モデルを紹介する。

オブジェクトは他のオブジェクトとの相互作用によって状態変化する。協調場モデルは、オブジェクトが集まり相互作用する場である協調場に振舞記述を集約し、one behavior at one place を実現する。オブジェクトは参加する場によって、その性質が規定される。これらにより再利用性の高いオブジェクト部品が定義される。

Behavior Model Extension of JDMF/M-92 (JDMF/BM)

Hideto Sato

Tokyo International University

1-13-1 Matobakita, Kawagoe, Saitama 350-11 Japan
email: sato@tiu.ac.jp

JDMF/BM is a behavior model extension of JDMF/M-92 that describes conceptual schemas. It provides close relationships between attributes and receivable events, which serves declarative and simplified definition of behaviors. This paper summarizes the idea and syntax of JDMF/BM, and introduces a specialized model for behavior analysis, named Collaborative Field Model.

An object changes its states during interactions with others. Behavior descriptions are concentrated in Collaborative Fields where objects gather and interact with each other. This description can place one behavior at one place. Properties of an object are deduced from fields where it participates. By this modeling, highly reusable objects can be defined.

1．はじめに

JDMF/BMは、日本規格協会がまとめた概念スキーマ記述のための標準モデル機能JDMF/MODEL-1992［1］に、動的振る舞い記述を加えたものである。本稿では、2節で、JDMF/BMの概要を解説し、3節で、それを用いた分析の枠組である協調場モデルを紹介する。4節で、協調場モデルに基づくオブジェクトの部品化に言及する。

なお、JDMF/BMは日本規格協会情報資源スキーマ調査研究委員会で審議中のものであり、ここに紹介するものは筆者個人の意見であることをお断りしておく。また、3節の協調場の概念は、筆者が参加した東京電力プロジェクト［2］の成果をJDMF/BMに取り込んだものである。

2．JDMF/BMの概要

2.1　JDMF/BMの振る舞い概念

JDMF/BMは、以下の3つの概念をもつイベント駆動型状態モデルである。

（1）イベント

状態の遷移、アクションの起動、他のイベントの発行のきっかけとなるトリガーをイベントという。イベントは、発行者から受取手に伝える情報として、引数を持つ。

（2）状態

オブジェクトの属性値の組、及び、関連するリンクの接続形態を反映する指標を状態という。JDMF/BMでは、明示的な属性値変更、関連リンク変更を伴わずとも、イベントの送受やアクションの起動があれば、状態は変わりうるものと考える。

従って、状態は、受取可能なイベントの識別、あるいは、イベントの発行、アクション起動のタイミングを識別するためにオブジェクト毎に設定される指標である。

（3）アクション

イベントによって起動される操作をアクションという。JDMF/BMでは、オブジェクトの生成、属性値変更、関連リンク変更、他のイベントの発行などがアクションである。

この振る舞い概念は、アクションを簡略化しているが、ＯＭＴ［3］などで採用されているものとほぼ同様である。

```
class <クラス名> = subclassOf(スーパクラス名)
        attributes      <属性宣言>...
        states          <状態宣言>...
        behavior        <振舞宣言>...
        methods         <メソッド宣言>...
end.
```

図1　オブジェクトの構成

2.2　JDMF/BMのオブジェクト定義

上述の振る舞い概念を記述するために、JDMF/BMは、JDMF/MODELの「属性付オブジェクト」クラスの定義を図1のように拡張している。

図2は、振る舞いをもつオブジェクト定義の例である。ここでは、個々の商品の貸出の状況を記述するオブジェクト「商品貸出」を定義している。以下では、例を用いて、構成要素を説明する。

```
1   class 商品貸出 = subclassOf(AttributedObject)
2     attributes
3       契約:        貸出契約     := given on 商品貸出依頼;
4       希望商品:    商品イメージ := given on 商品貸出依頼;
5       貸出品:      商品         := given on 商品選定完了;
6       返却予定日:  日付表現     := given on 商品貸出依頼;
7       返却日:      日付表現     := Date() on 返却;

8       貸出中:      真偽値表現 := (state = 貸出中);
9       占有中:      真偽値表現 := 貸出中 or (state = 貸出可);
10      期限切れ:    真偽値表現 := 貸出中 and
                                   (Date() > 返却予定日);
11    states
12      キャンセル可能;
13      貸出品選定中              = substateOf(キャンセル可能);
14      貸出可                    = substateOf(キャンセル可能);
15      貸出中;
16      返却済:                   terminal;
17      キャンセル済:             terminal;
18    behavior
19      on 商品貸出依頼(契約, 希望商品, 返却予定日)
            then create and
                invoke 商品選定指示(self) and
                    state = 商品選定中;
20      on 選定完了(self, 貸出品)
            if state = 商品選定中 then
                    state = 貸出可;
21      on 貸出実施(契約)
            if state = 貸出可 then
                invoke 商品貸出実施(契約, 貸出品) and
                    state = 貸出中;
22      on 返却(貸出品)
            if state = 貸出中 then
                invoke 返却受取(self) and
                    state = 返却済;
23      on 選定不能(self)
            if state = 商品選定中 then
                invoke 商品貸出不可通知(契約, 希望商品) and
                    state = キャンセル済;
24      on キャンセル(契約)
            if state = キャンセル可能 then
                    state = キャンセル済;
25  end.
```

図2　振舞オブジェクトの定義例

（1）属性の宣言と式

属性宣言では、従来からある定義域の宣言に加え、属性値指定が加わり、導出属性の定義が可能になった。また、固有属性については、その値が定まるタイミングをイベント毎に指定する。

```
(1)導出属性の定義例
  金額: 金額表現 := 数量 * 単価;
        ↑      ↑      ↑
  属性名 定義域   属性値指定

(2)固有属性の定義例
  入庫量: 数量表現 := given on 入庫;
  棚卸日: 日付表現 := Date() on 棚卸実施;

(3)複雑な例
  在庫量: 数量表現
          := 0 on 新規登録
          := 在庫量 + 入庫量 on 入庫
          := 在庫量 - 出庫量 on 出庫;
```

図3　属性宣言の例

属性値指定の一般形は、次の通りである。

<式> on <イベント>

ここで、<式>は、属性を変数とする算術式、論理式、メッセージ式、あるいは、これらの組み合わせである。メッセージ式以外は、Pascalの記法に準拠している。これらは、常識でも理解可能なものであるので、詳細を省略する。メッセージ式、及び、メッセージ式を利用した問い合わせメソッドの例を図4に示す。

```
(1)メッセージ式の使用例
  上司の名前: 人名 := 所属．課長．名前；

  「所属」(当該オブジェクトの属性)している課の「課
長」(課の属性)の名前(課長である従業員の属性)

(2)問い合わせメソッドの使用例
  引当可能商品リスト: setOf(商品) :=
        商品．suchThat( x | x．品名 =注文商品名)；

  「商品」クラスのインスタンスで、その「品名」が「注
文商品名」と一致するものだけを集めた集合オブジェ
クトを属性「引当可能商品リスト」の値とする。
```

図4　メッセージ式と問い合わせメソッド

図2で、3～7行が固有属性の定義であり、8～10行が導出属性の定義である。

(2) 状態遷移

状態宣言と振舞宣言は、イベントに基づくオブジェクトの状態遷移を定義する。図2の11～24行の定義内容は、状態遷移図で書くと、図5のようになる。個々の宣言の詳細を説明する前に、この図に沿って、商品貸出オブジェクトの振る舞いを説明しよう。

状態遷移図では、グレーの矢印はイベントの送受を、黒の矢印は状態の遷移を表しており、丸四角は、状態を意味している。

上から見ていくと、「商品貸出依頼」というイベントが発生すると、「商品貸出」クラスはそのインスタンスを生成し、「商品選定指示」というイベントを送出して、「貸出品選定中」という状態になる。「貸出品選定中」という状態にあるとき、「選定完了」というイベントを受け取ると「貸出可」の状態になり、「選定不能」というイベントを受け取ると「商品貸出不可通知」というイベントを発行して「キャンセル済」という状態になる。以下同様である。

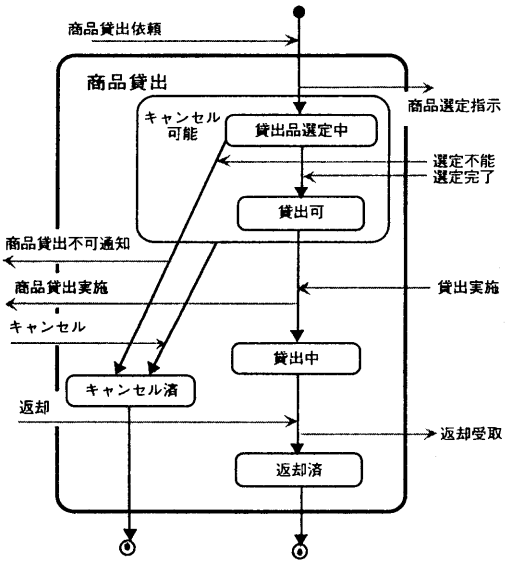

図5　状態遷移図

(3) 状態の宣言

状態宣言は、状態遷移記述に必要な状態を定義するものである。図2の12～17行がこれに当たる。「貸出品選定中」「貸出可」の2つの状態は、「キャンセル可能」という状態の部分状態である。13、14行目の「<状態>=substateOf(<上位状態>)」は、このことを示している。

また、16、17行目の「<状態>：terminal」は、その状態が終端状態であることを示している。

JDMF/BMでは、終端状態にあるオブジェクトが直ちに消滅するとは考えていない。終端状態にあるオブジェクトを削除するタイミングの決定は、実装上の問題である。

(4) 振舞の宣言

振舞宣言では、図6の(1)の形式で、オブジェクトが受け取れるイベントを宣言し、その受け取り条件と受け取ったときに起動されるアクションを記述する。

図6の(2)は、もっとも一般的な振舞宣言の例である。オブジェクトの状態が、「貸出中」の状態にあるとき、「返却」イベントが発生すると、オブジェクトはこのイベントを受け取り、アクションとして、「返却受取」というイベントを送出して、「返却済」という状態に遷移する。この振舞宣言は、図5の状態遷移図のグレーと黒の矢印に対応している。

なお、擬似変数 state は、オブジェクトのカレントな状態を表わしている。これは、当該オブジェクトの内部でだけ参照することができる。

```
(1)振舞宣言の一般形
  on <受取イベント> if <条件> then <アクション>;

(2)一般的な例
  on 返却(貸出品) if state=貸出中 then
  (受取イベント)      (前状態)
      invoke 返却受取(self) and state=返却済;
        (イベント送出)          (後状態)

(3)状態駆動型遷移
  if Time()=18:00 and state=開店中 then
          invoke 閉店 and state=閉店中;

(4)応答
  on 商品選定指示(貸出)
      if post 選定商品 is not null then
          invoke 選定完了(貸出, 選定商品);

(5)オブジェクト生成
  on 商品貸出依頼(契約, 希望商品, 返却予定日) then
      create and invoke 商品選定指示(self) and
          state=商品選定中;
```

図6　振舞宣言とその例

図6（3）の例は、受取イベントを持たず、ある条件が成立したとき、自動的に状態遷移が起きる例である。この例では、１８：００になったとき、開店していれば、閉店し、閉店中という状態になるという状態遷移を表している。

振舞宣言は、受取イベントと条件で識別される。従って、サブクラスで同じ受取イベントと条件をもつ振舞宣言を定義することにより、スーパクラスの定義を上書きすることができる。

（5）振舞宣言と属性宣言

JDMF/BMでは、振舞宣言と属性宣言に依存関係を持たせることにより、イベント受取条件とアクションの記述を簡潔にしている。

イベントの引数は属性またはオブジェクトの並びであり、次の２つの意味を持つ。

1）属性の更新値の提供
2）イベントの受け取り条件

どちらの意味で使用されるかは、属性宣言内の属性値指定におけるイベント指定に依存する。

例えば、図２の１９行の「商品貸出依頼（契約、希望商品、返却予定日）」では、この３つの引数の属性が３～７行で「given on 商品貸出依頼」となっている。従って、ここでは、属性更新値の提供であると解釈される。

これに対し、２１行の「貸出実施（契約）」では、属性「契約」の定義に「given on 貸出実施」がない。この場合は、引数「契約」の値が属性「契約」に一致する場合のみ、このイベントは受け取れるというイベントの受け取り条件を示している。

２０行の「選定完了（self、貸出品）」は、２つのタイプの引数を含むケースである。「self」（当該オブジェクト自身）は、イベント受け取り条件を、「貸出品」は属性の更新値を表している。

（6）属性の事前値と事後値

上述のように、属性値の更新を伴うイベントを受け取るとき、if 以下の受取条件の評価に当たって、属性値の事前値（更新前の値）を使用すべきか事後値（更新後の値）を使用すべきか問題になるケースがでてくる。このようなときは、属性に　pre（事前）、または、post（事後）の接頭辞をつけて区別する。

図６の（４）はそのような例で、引数「貸出」の値を属性「貸出」に代入し、その結果得られる導出属性「選定商品」の値が空でなければ、このイベントを受け入れ、アクションを実行する。

事後値を使用した場合でも、if 以下のイベント受取条件が満足されないときは、元の状態にロールバックするものとする。

（7）アクションの記述

アクションには、オブジェクトの生成、属性値の更新（関連リンクの変更を含む）、他のイベントの発行、メソッドの起動、事後状態の設定がある。このうち、属性値の更新については、上述のように、属性宣言で記述し、他のものは、振舞宣言の　then 以下で記述する。

オブジェクトの生成は、 create アクションで行う。すなわち、これを含む振舞宣言は、オブジェクトクラスが受け取るイベントを示している。図６の（５）は、そのような例である。

他のイベントの発行では、「 invoke　＜送出イベント＞」を、メソッドの起動では、「do ＜メソッド呼出＞」を　then　以下に記述する。事後状態は、「state = ＜後状態＞」で指定する。

（8）メソッド宣言

メソッド宣言では、メソッドの定義を記述する。メソッドは対応するメッセージに反応する手続または関数である。上述の例で、Date()、Time()、suchThat()　は、モデル機能が提供するメソッドである。

応用モデルでもメソッドを使用することはできるが、原則として、属性値の変更などの副作用を伴わない関数に使用を限定する。

2.3　JDMF/BMの特徴

JDMF/BMは、通常のイベント駆動型状態モデルと同様であるが、以下の特徴を持っている。

（1）振舞宣言と属性宣言の連携により、イベントの受取条件と属性値の更新アクションを、宣言的かつ簡潔に記述できる。

（2）イベントによらず、ある条件により起動される振る舞いを記述できる。これにより、ペトリネットなどの状態駆動型モデルのモデル結果も表現できる（図6(3)）。

（3）状態遷移を伴わないイベント送受を記述できる。これにより、メッセージ交換型モデルのモデル結果も表現できる（図6(4)）。

3．協調場モデル

3.1　モデル機能と部品化

2節で説明したJDMF/BMは、種々の振る舞い概念を1つの枠組の中で記述できる高い自由度をもつモデル機能である。しかし、自由度が高いということは、同じ1つの振る舞いを種々の形でモデル化できることを意味しており、モデル結果の部品化・再利用という観点から見るとき、必ずしも好ましいことではない。

このことは、単にJDMF/BMだけの問題ではなく、OMTをはじめとするオブジェクト指向方法論一般についても、同様にいえることである。

これは必ずしもモデル機能の欠陥を意味するものではない。むしろモデル機能を現実問題に適用する際の考え方・ガイドラインが未整備であるということに過ぎない。

ソフトウェアの部品化を考えるとき、データの部品化を追求したリレーショナルモデルにおける経験を活かすことができる。データは目的に応じて種々の形をとる。これをそのままデータベース化すると同じ事実が異なるデータに重複して記述されることになり、共有・再利用が困難になる。

データの正規化は、オブジェクト（実体）の直接の性質を表すデータをそのオブジェクト（実体レコード）にのみ帰属させることにより、"one fact at one place" を実現し、データ部品としてのオブジェクトの再利用性を高めるものである。

振る舞いについても、同様に考えることができる。振る舞いは、基本的にオブジェクト間の相互作用である。この相互作用をそれぞれのオブジェクトの側からモデル化すると、振る舞い記述の重複やオブジェクト間の強い依存関係が発生し、それぞれのオブジェクトを独立のソフトウェア部品として再利用することが困難になる。

振る舞いを、それが発生する相互作用場に、帰着させることにより、再利用性の高いソフトウェア部品を抽出することができる。すなわち、"one behavior at one place" の実現である。この第3節で説明する協調場モデルは、このような考え方に立って、その適用方法を限定したJDMF/BMのspecializationである。

3.2　協調場モデルの基本概念

（1）オブジェクトの振る舞いと協調場

振る舞いとは、オブジェクトの状態の変化である。しかし、一般的にオブジェクトは、それ単独でその状態を変化させることはない。他のオブジェクトとの相互作用によって状態の変化が生じる。このため、相互作用するオブジェクト相互に双対的な状態変化が生じる。

例えば、ビデオ店からビデオテープを借りるという場合を考えてみよう。顧客は「借りていない」という状態から「借りている」という状態に変化し、テープの方は「在庫中」という状態から「貸出中」という状態に変わる。

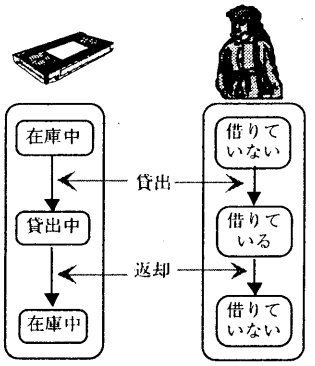

図7　複数オブジェクトの双対的状態遷移

このようにオブジェクト毎に状態変化を記述すると、記述の重複が生じ、オブジェクトの独立性が失われる。

オブジェクトが集まり、協調して状態を変える場を協調場（Collaborative Field）と呼ぶことにする。状態変化はこの協調場に記述し、個々のオブジェクトの状態は、自身が参加する協調場の状態から派生するもの（導出される状態）と考えれば、この状態記述の重複を避けることができる。

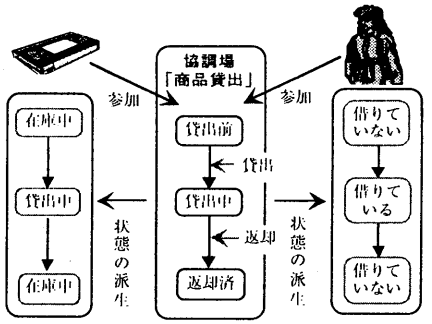

図8　協調場と状態の派生

この協調場を考えることの利点として、次の2点を挙げることができる。

1）1つの振る舞い（状態変化）を1ヶ所に記述できる。
2）1つのオブジェクトが複数の場に参加し、多数の役割を同時に演じるケースを容易にモデル化できる。

実際、われわれは多数の協調場に同時に参加し、それぞれの場に応じて、固有の状態や属性を持っている。われわれ自身とは何かといえば、それはこれらの場における役割の合成物に他ならない。

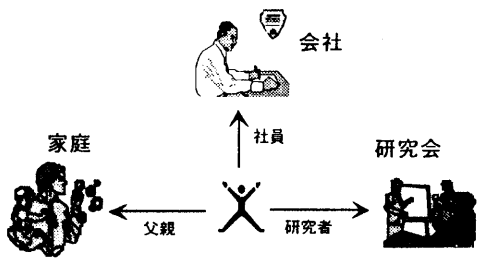

図９　多数の場への並行的参加

（２）前提オブジェクト

上で述べた協調場に対し、自身の固有の振る舞い（固有の状態変化）を持たず、協調場に参加することによって派生的に状態変化するオブジェクトを前提オブジェクト（Presupposed Object）と呼ぶ。図８の例でいえば、借手である「人」や借りられる「商品」が前提オブジェクトである。

（３）協調場とオブジェクト関連図

上で述べた協調場を、それより細かく分解すると－すなわち、参加者を少なくすると－意味を失ってしまうという限界まで分解すると、それは、正規化されたオブジェクト関連図におけるオブジェクト化された関連に帰着する。

（注）ここでの正規化はリレーショナルモデルの正規化と同じ意味である。

上のビデオレンタルの例でいえば、オブジェクト関連図は、図１０のように書くことができる。通常ビデオレンタルでは、１度に複数のビデオを借りることができ、その単位で清算する。この１件のレンタル契約は、顧客とレンタル店の関連であり、図で「貸出契約」と書かれたものに対応する。

他方、１本１本のビデオの貸出は、この契約に基づいて、顧客の希望するタイトル（商品型式）に対応するビデオ（商品）を貸し出すものであり、図で「商品貸出」と書かれたものに対応している。この場合、「貸出契約」という場が「商品貸出」という他の場の参加者となる。

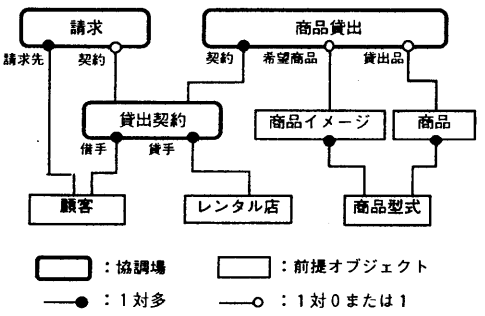

図１０　オブジェクト関連図と協調場

このビデオレンタルの例では、「貸出契約」、「商品貸出」、「請求」の３つが協調場であり、「顧客」「レンタル店」「商品」「商品イメージ」「商品型式」が前提オブジェクトである。

（４）協調場の振る舞いと判断場

第２節の図２のオブジェクト定義例は、上の協調場「商品貸出」を定義したものであり、図５の状態遷移図は、その振る舞いを書いたものである。

協調場の振る舞いを記述するとき、その場の参加者や場固有の性質だけでは決定できないような判断が必要になる場合がある。例えば、図５の「商品貸出」の振る舞いの例でいえば、要求のあったタイトル（商品型式）に対し、どのテープ（商品）を貸し出すかという「貸出品選定」という判断がこれに当たる。この判断を下すためには、在庫品全体の中から該当タイトルをもつテープを探し出すという操作が必要になる。

この種の判断は、より一般的には、注文に対する在庫引当と呼ばれるものである。在庫引当には、先入先出、後入先出などの多くのアルゴリズムが存在する。また、得意客を優先するなど、個々の企業に特有の業務ポリシーもありうる。

このように、応用毎に異なる可能性が高く、同一の応用でもしばしば変更されやすい判断は、再利用性を重視する協調場には含まない方が良い。

協調場モデルでは、このような判断業務を独立させ、それを専門に扱う判断場（Decision Field）という概念を導入する。判断場はオブジェクト間の関連の一種には違いないが、次のような特徴を持つ。

１）参加者の範囲が広く、しばしば集約的あるいは要約的情報を必要とする。
２）しばしば複雑なルールが関係する。これらのルールは業務上のポリシーに依存しており、しばしば変わりうるものである。
３）必要とする情報は、場の参加者の属性や参加者が関係する関連（協調場）の状態から演繹されうるものである。
４）固有の場の状態を持たない。

現実世界とのアナロジーでいえば、協調場はルーチンワークをこなすラインに相当し、判断場は諮問委員会のような意思決定のための会議に相当するものであるといえる。判断場は、要求に応じて参加者を集め、その情報をもとにある判断結果を下す。しかし、その判断結果を実行する（あるいは無視する）のは、現場である協調場の仕事である。

なお、判断場は状態遷移を持たないが、固有の属性やある種の振る舞いをもつことはありうる。例えば、既定の在庫水準を下回った品目を検出する在庫判断は、判断場の一種としてモデル化できる。このとき、参加者である在庫品の数量変化を監視し、水準を下回ったとき、それを通知するイベントを発行できる機能が必要である。また、一定間隔で在庫検査を行うような場合には、最後に行った検査の日付を記録する属性が必要である。

（５）場の相互作用

協調場モデルは、協調場と判断場で構成される。協調場は、イベントを受け取って、自身の状態を変化させ、イベントの発行によって、その状態変化を他の場に伝える。判断場は、問い合わせイベントを受けて起動され、判断（計算）を実行し、結果を返すイベントを発行する。このイベントの連鎖によって、システム全体の挙動が記述される。

この場の間のイベントの流れを図にまとめると、図１１のようになる。ここで黒の太枠で書かれた「貸出契約」「商品貸出」「請求」が協調場であり、グレーの太枠で書かれた「資格審査」「全件返却検査」「貸出品選定」「料金計算」が判断場である。

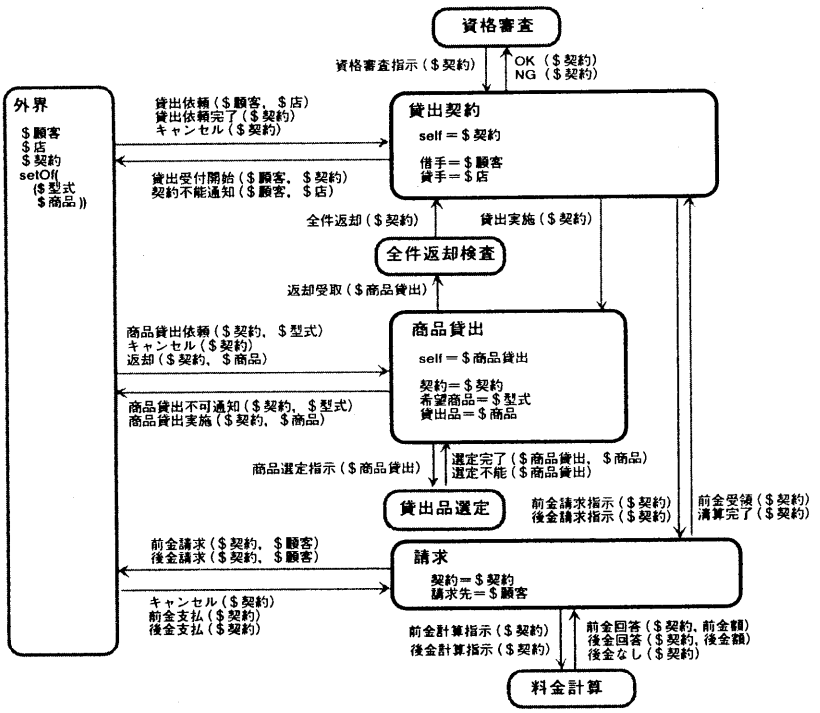

図１１　イベントフロー図

（６）外界

図１１で「外界」と書かれたものは、当該アプリケーションではモデル化の対象としていない協調場に対応するものである。対象世界が広がれば、これらも内部化されうる。

なお、当該アプリケーションの実装には、この外界が送受するイベントの実際の送受を受け持つユーザインタフェースの設計が必要になる。

```
＜前提オブジェクト＞
class ＜クラス名＞ = subclassOf(スーパクラス名)
        attributes    ＜属性宣言＞...
end.

＜協調場＞
class ＜クラス名＞ = subclassOf(スーパクラス名)
        attributes    ＜属性宣言＞...
        states        ＜状態宣言＞...
        behavior      ＜振舞宣言（状態遷移）＞...
end.

＜判断場＞
class ＜クラス名＞ = subclassOf(スーパクラス名)
        attributes    ＜属性宣言＞...
        behavior      ＜振舞宣言（応答）＞...
end.
```

図１２　協調場モデルのオブジェクト構成

３．３　協調場モデルのオブジェクト構成

協調場モデルの構成要素はそれぞれ、２節で述べたJDMF/BMのオブジェクトを図１２にあるように制限したものである。前提オブジェクトは状態・振る舞いを持たず、判断場は応答型の振る舞いのみをもつ。このように制限することにより、振る舞い記述の一意性を高めることができる。

４．オブジェクトの部品化

（１）協調場と部品化

協調場は、オブジェクト化された関連であり、関連参加者の存在に依存する。しかし、参加者の内部状態による違いを判断場で吸収することにより、参加者の内部状態に依存しないモデル化ができる。実際、図２の「商品貸出」場は、商品レンタルだけでなく、対価を伴わない図書館の貸出にも適用できる。

なお、協調場も一種のオブジェクトであるから、クラス階層化による部分修正が可能である。

（２）判断場と部品化

判断場は参加者の導出属性で構成される。その導出の過程を個々のオブジェクトに帰着すると、判断場に残るものは、複数のオブジェクトの属性間に成立するルール（ビジネスルール）だけになる。図１３の例では、８行目の追加借入可能かどうかというルールがこれに当たる。

```
「貸出契約」「レンタルビデオ会員」を前提とする判断場
1 class 資格審査 = subclassOf(DecisionView)
2   attributes
3     契約:          貸出契約 := given on 資格審査指示;
4     借手:          レンタルビデオ会員 := 契約. 借手;
5     借入予定本数: 本数表現 := 契約. 借入予定本数;
6     期限切れ借入あり: 真偽値表現
                          := 借手. 期限切れ借入あり;
7     追加借入可能本数: 本数表現 := 借手. 追加借入可能本数;
8     追加借入可能: 真偽値
                  := (追加借入可能本数 >= 借入予定本数)
                          and (not 期限切れ借入あり);
9   behavior
10    on 会員検査指示(契約)
        if post 借入可能 then invoke OK(契約);
11    on 会員検査指示(契約)
        if not post 借入可能 then invoke NG(契約);
12  end.
```

図１３　判断場の例

（３）オブジェクトのクラス階層

判断場で必要となる情報を個々のオブジェクトの導出属性に帰着すると、オブジェクトが参加する協調場に応じて、以下のクラス階層が形成される。

```
前提オブジェクト「人」
1 class 人 = subclassOf(PresupposedObject)
2   attributes
3     名前:          人名表現;
4     住所:          番地表現;
5   end.

前提オブジェクト「会員」
6 class 会員 = subclassOf(人)
7   attributes
8     会員番号:      識別子;
9     入会日:        日付表現;
10  end.

協調場「貸出契約」の参加資格者
11 class レンタル会員 = subclassOf(会員)
12  attributes
13    借入契約リスト: setOf(貸出)
                  := 貸出. suchThat( x | x. 借手 = self);
14  end.

協調場「貸出契約」と「商品貸出」の参加資格者
15 class レンタルビデオ会員 = subclassOf(レンタル会員)
16  attributes
17    最大借入本数: 本数表現;
18    借入リスト: setOf(商品貸出)
                  := 商品貸出. suchThat
                      ( x | x. 契約. 借手 = self and x. 貸出中);
19    借入本数:    本数表現 := 借入リスト. count;
20    追加借入可能本数: 本数表現
                  := 最大借入本数 - 借入本数;
21    期限切れ借入リスト: setOf(商品貸出)
                  := 商品貸出. suchThat
                    ( x | x. 契約. 借手 = self and x. 期限切れ);
22    期限切れ借入あり: 真偽値表現
                  := (期限切れ借入リスト is not null);
23  end.
```

図１４　前提オブジェクトのクラス階層

（４）協調場の参加資格者

図１４でオブジェクト「レンタル会員」は協調場「貸出契約」の参加資格者と呼ばれる。これは、その属性「貸出契約リスト」は、そのオブジェクトが参加する「貸出契約」場の属性から導出されるものであるからである。

ここで注意すべきは、この属性は協調場「貸出契約」の属性そのものではないということである。オブジェクトが参加する複数の「貸出契約」インスタンスの属性から導出されるものであり、個々の「貸出契約」の属性に帰着することはできない。

また、そのオブジェクトが、ある瞬間に、どの「貸出契約」場にも参加していないとしても、この属性は「貸出契約リストが空である」という事実を示し、意味を持っている。この理由から、参加者（あるいは役割オブジェクト）と呼ばずに、参加資格者と呼ぶ。協調場モデルでは、協調場への参加資格の獲得によってオブジェクトのdynamic classification［４］を記述する。

（５）導出属性の固有属性化

オブジェクトの固有属性と見られるものも、元をただせば、なんらかの協調場の属性から派生したものである。例えば、図１４で、オブジェクト「会員」がもつ属性「会員番号」「入会日」は、協調場「会員登録」の中で発生したものであり、それが場の参加者であるオブジェクトの属性であるかのようにみえるに過ぎない。

つまり、すべての属性は協調場で発生する。しかし、ある応用から見たとき、そこでモデル化の対象とならない協調場の属性は、その場の参加者の固有属性であるとみなされるのである。

（６）部品からの応用モデル構築の概要

協調場モデルでは、協調場毎に場とその参加資格者が部品化される。応用モデルのUoDに応じて、協調場が選ばれ、UoDの範囲外の協調場に起源をもつ導出属性は固有属性化される。このようにして応用モデルが構築される。

５．むすび

本稿では、一般性のある振る舞いモデルJDMF/BMを定義し、その特殊化として、協調場概念に基づく分析モデルを提示し、オブジェクトの部品化の道筋を示した。部品化されたオブジェクトからの応用モデルの構築とその実装は、今後に残された課題である。

参考文献

［１］情報資源スキーマ調査研究報告書、日本規格協会、1993.

［２］オブジェクト指向モデリング手法の研究、東京電力委託研究報告、1995.

［３］J.ランボー他（羽生田栄一訳）、オブジェクト指向方法論OMT、トッパン、1992.

［４］J. Martin & J. J. Odell, Object Oriented Analysis & Design, Prentice Hall, 1992.