ソフトウェア工学 116－5
（1997．9．11）

OO'97 オブジェクト指向モデリングワークショップ報告

鯵坂恒夫[†1] 池田健次郎[†2]
中谷多哉子[†3] 野呂昌満[†4]

OO'97 の一セッションとして開催したモデリングワークショップについて，課題やモデル例，および議論の一端を紹介する．

OO'97 Workshop Report on Object Oriented Modeling

TSUNEO AJISAKA,[†1] KENJIRO IKEDA,[†2] TAKAKO NAKATANI[†3]
and MASAMI NORO[†4]

The OO'97 symposium had a workshop session focusing object oriented modeling. This paper reports the exercise problem, its modeling examples, and some issues discussed at the workshop.

1. はじめに

オブジェクト指向 '97 シンポジウムでは，昨年に引続き，具体的例題を対象に実際のモデリングの過程や結果を比較検討するワークショップが，セッションのひとつとして設けられた．このワークショップでは，例題要求を分析しソフトウェアシステムとして実現できるような形にモデル化するエクササイズを通して，モデリングに一般的に有用な規則や知識を抽出，整理することを目的とする．クラスやメソッドの同定及びそれらの相互関係が明確になるような抽象度，粒度でモデリングを行なった結果を示しあい，個々のモデルの長所／短所について議論を行なった．

モデルの優越性は

- 要求項目に対応すべきシステム要素の抜けを検知しやすい
- 複数の解釈／選択可能性や変更要求に対応しやすい
- 再利用性が高い

といった性質で議論することとし，性能や耐故障性など，より詳細な設計が必要となる評価には今回は立ち入らないこととした．

†1 和歌山大学
†2 NEC
†3 東京大学
†4 南山大学

2. 課　題

ワークショップで用いた例題は，缶入り清涼飲料水の自動販売機を制御するソフトウェアである．対象となる自動販売機の詳細を以下に示す．まず「3 購買者インタフェース」から見れば，われわれが日ごろ街角でよく利用する自動販売機であることがわかるが，「1 対象機の機器ハードウェア構成」や「4 販売者インタフェース」に記述されているように，販売管理のための機構や機能も備わっている．自動販売機の製造メーカでは，多品種生産に対応するため，ハードウェア構成のモジュール化を進めており，「2 対象機の制御 CPU 構成」に示されるようないわば分散システム的な構成が前提条件となっている．より詳細な制約条件は「5 機器制御における制約条件」にまとめた．

1　対象機の機器ハードウェア構成

- 1.1　商品を格納するラックが内部に複数個あり，各ラックは温冷両用である．温熱・冷却機と温度センサが各ラックに取り付けてあり，ラック内の温度が適正温度に保たれるように制御される．
- 1.2　販売ボタンが複数個あり，販売可能表示ランプと売切表示ランプが内蔵されている．
- 1.3　紙幣投入口，硬貨投入口，プリペイドカード読みとり機が各１個ある．
- 1.4　投入された金額から販売済代金をさし引いた現在残高の表示器がある．

1.5 販売された商品の取り出し口が1個あり，取り出し口に商品が残存するかどうかを検査するセンサがある．

1.6 釣銭取り出し口が1個ある．

1.7 釣銭切れおよび釣銭払出し動作中を表示する表示器がある．

1.8 返金ボタンが1個ある．

1.9 懸賞ルーレット機がある．

1.10 販売管理用の内部キーボードがある．

1.11 売上に関するデータを蓄積するデータベースが内部にある．

1.12 内部データベースの内容をメーカーへデータ転送するための専用無線回線がある．通信は独自の通信規約による．

2 対象機の制御 CPU 構成

2.1 各ラック毎に独立した CPU があり，それぞれ商品の取り出し口への送り出しとラックの温度を制御する．

2.2 複数個の販売ボタンは，ひとつの独立した CPU で制御される．

2.3 紙幣，硬貨，プリペイドカードは，それぞれ個別の機器によって処理され，各機器に対応する独立した CPU で制御される．

2.4 販売商品取り出し口のセンサは，独立した CPU で制御される．

2.5 返金ボタンは硬貨処理機器を制御する CPU で制御される．

2.6 内部データベースは独立した CPU で制御される．

2.7 専用無線回線は，独立した CPU でデータ授受と制御が行なわれる．

2.8 以上の全 CPU を管理するマスタ CPU がある．

2.9 CPU 間相互の通信は，相手を特定した virtual circuit 通信で行われる．

3 購買者インタフェース

3.1 代金投入：紙幣投入口，硬貨投入口，プリペイドカード読みとり機に手で投入する．紙幣は千円札のみ，硬貨は 10 円以上のものが使用可能である．プリペイドカードと紙幣，硬貨の併用が可能である．

3.2 残高確認：投入された金額から販売済代金をさし引いた現在残高を表示器で確認する．

3.3 価格表示：各販売ボタンにオフライン（紙ラベルで）表示される．

3.4 売切表示：当該販売ボタンに売切表示ランプで表示される．

3.5 商品選択：投入金額に応じ販売可能表示（ボタンが光る）されたボタンを押す．販売動作中は当該商品のボタンが点滅する．

3.6 商品取得：販売商品取り出し口から手で取り出す．

3.7 釣銭取得：釣銭取り出し口から手で取り出す．

3.8 釣銭確認：釣銭切れおよび釣銭払出し動作中を表示する表示器で確認する．

3.9 返金要求：返金ボタンを押すと，その時点での残高が釣銭取り出し口に返金される．

3.10 懸賞：懸賞ルーレット機の表示で確認する．当れば任意の商品がさらに1本選択，購入できる．

3.11 利用時間：販売可能時間の設定があって，その時間外は購入できない．

4 販売者インタフェース

4.1 販売管理用の内部キーボードを用いて，
各商品の価格，各ラックの温冷切替え，ラックとボタンの対応（多対多対応設定可能），販売機への最大投入金額，販売可能時間
を設定できる．

4.2 販売管理用の内部キーボードを用いて，指示された期間に販売した（特定のあるいはすべての）商品の本数と売上金額を，データベースへの問い合わせにより知ることができる．

4.3 定期的に内部データベースの内容をメーカーへデータ転送することができる．

5 機器制御における制約条件

5.1 同一商品が複数ラック納まっている場合，各ラックから均等に販売されるように制御する．

5.2 温商品は摂氏 70 度以上，冷商品は摂氏 4 度以下を保つ．

5.3 次の場合，ボタン操作はできない．
当該商品売り切れの時，販売動作中，釣銭払出し / 返金動作中，代金投入動作中

5.4 次の場合，金銭は受け付けない．
全商品売り切れの時，販売動作中，釣銭払出し / 返金動作中，

最大投入金額を越える時

5.5 プリペイドカード，紙幣，硬貨を併用する際の優先順位はこの順である．

5.6 プリペイドカードに釣銭は払出さない．

5.7 硬貨は連続投入可能であるが，その際の読み飛ばしが決して起こらないことを保証する．

5.8 釣銭は最小枚数の硬貨で払出す．

5.9 投入金額の範囲内で一定時間内に連続販売ができる．ただし，商品は取り出し口から１本ずつ取り出されていなければならない．販売可能表示等は１本販売する毎に変化する．

5.10 データ転送中に販売事象が起こった場合，販売を優先してデータ転送は中断し，販売動作終了後再開する．

5.11 地震，火事，外部からの強い衝撃ですべての機能を停止して，電源を自動的に切断する．

3. OMT法によるモデリング

例題として与えられた自動販売機の制御ソフトウェアを、OMT法を用いて分析した。

本章では、OMT法で分析する際の方針と作業内容を述べた後に、作成された分析モデルとその性質について考察する。

3.1 分析の方針

一般的にどの様な分析手法を用いようとも、分析の方針を決めておかないと意味の無い分析モデルを構築する可能性が高くなる。特にOMT法はモデル構築の際の制限も少なく自由度が高い手法なので、分析の方針を決めておくことが重要である。

今回は以下の方針で分析することにした。

3.1.1 どこまでを分析とするか

オブジェクト指向で分析する場合には、どこまでで分析と設計の区切りをつける事が難しいが、今回は単純にwhatのみを記述したモデルを作成できた時点を分析の終わりとした。

具体的には、与えられた問題記述文にはCPUに関するものを除き実装に関する記述は無いと考え、問題記述文から直接読み取れるクラスのみで分析モデルを構築することにした。また、オブジェクト間のやり取りの種類(メッセージ通信、関数呼びだし等)も区別しないこととした。

3.1.2 ハードウェアの取り扱い

今回の例題はハードウェアの制御ソフトなので、制御対象としてのハードウェアの構成要素をクラスとするのは自然であると考える。

そこで、基本的に問題記述文中に出て来る制御対象のハードウェア構成要素は全てクラスとした。それに対し、制御手段として存在するCPUはクラスとしないことにした。また、直接制御に関わらないものもクラスとしないことにした。

3.1.3 変更可能性 / 再利用性への取り組み

変更可能性や再利用性を考える場合、システムのどの部分に仕様変更の可能性があるのか、どの部分 / どの様な単位で再利用を行うのかと言う事が分からなければならない。そこで、変更可能性と再利用性について仮定を置いて方針を決めた。

まず、システム外部とのやり取りがあるインタフェース部分は変更可能性が高いと仮定し、仕様変更に対する対応はインタフェース部分を重点的に検討することにした。

また、再利用は主にハードウェアのモジュール単位で行われると考えられるが、今回の問題記述で示されている各CPUの制御する範囲が自動販売機システムのハードウェアのモジュールとして一般的であると仮定し、ハードウェアのモジュール構造を分析モデルのクラス構造に作り込むことにした。

3.2 分析作業

実際の分析作業は、OMT法に準拠する方法で行った。具体的には、問題記述文中の名詞句、動詞句に注目してクラス図を作成し、動作例としてのシナリオを作成して各クラスの状態図を作成している。作業手順は以下の通りである。

1 クラスの識別
2 関連付け
3 属性の識別
4 シナリオの作成
5 操作の識別
6 クラス毎の状態図作成

当然のことながら、これらの作業は逐次的に一度だけ行うのではなく、必要に応じて複数の作業を並行して行ったり、何度も繰り返したりする。

以下、各作業の詳細について説明する。

3.2.1 クラスの識別

問題記述文中より名詞句を抽出してクラス候補とし、その中から曖昧なもの、冗長なもの、属性になりそうなものを削除してシステムを構成するクラスを抽出した。

ハードウェア構成要素も基本的にクラスとしているが、方針で述べた様に制御手段としてのCPUや、制御に直接関係しない硬貨取り出し口の様なものはクラスに

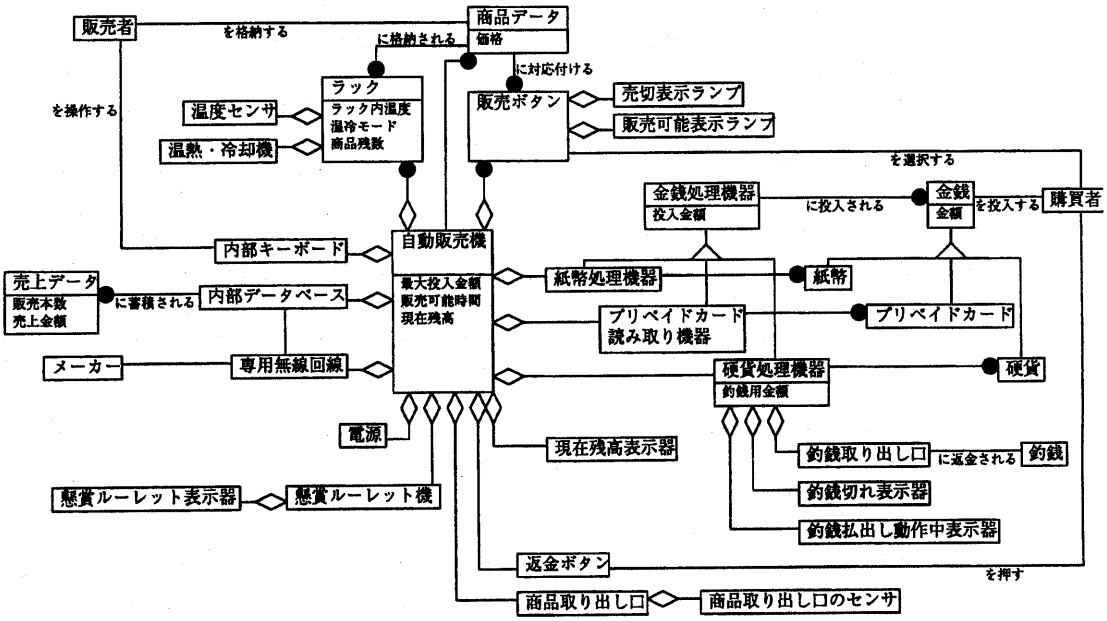

図1　クラス図 (全体)
Fig. 1　class diagram

はしていない。

また、購入者や販売者等のクラスはシステム外にあるクラスであるが、モデルの理解を助ける目的で残してある。

3.2.2　関連付け

動詞句に注目する等して、問題記述文中から読み取れる通常の関連を抽出した。

また、自動販売機のハードウェア構成を集約関係で表現した。これは問題理解の為に問題記述を素直に表現しているだけであり、集約関係であることは本質ではない。

更に、クラスの持つ操作や属性がある程度識別されたされた段階で、クラスの持つ操作や属性に注目して継承関係の抽出を行った。

3.2.3　属性の識別

各クラスに対して、問題記述文中から明らかに読み取れるものを属性として識別した。今回の例題では、あまり複雑なデータを取り扱わないので、特別な作業は行っていない。

3.2.4　シナリオの作成

幾つかの典型的な場面を想定してシナリオを作成した。作成されたシナリオは問題記述文中の処理記述文と突き合わせて、動さの抜けをチェックした。

なお、例外的な処理のシナリオも含めてしまうとモデルが複雑になり他手法との比較が困難になるので、今回の分析では主に正常系の処理だけを対象とし、例外処理的なものは対象外としている。

3.2.5　操作の識別

シナリオの作成にあわせて操作の識別を行った。今回はオブジェクト間のやり取りの種類は区別しない事にしたので、便宜的にオブジェクトが受け取るメッセージは全て操作とした。最終的にどれを操作にするかは、各メッセージのやり取りをどの様な実装方法で実現するか判断する段階で決められる。

また、クラス構成や各クラスの状態遷移が大まかに決まった段階で、操作の構成 / 配置の見直しを行った。

3.2.6　クラス毎の状態図作成

作成されたシナリオを元にして、クラス毎の状態図を作成した。

今回は、各クラス毎の独立性を高め再利用し易くする為に、各ハードウェア構成要素の状態は対応したクラスで管理する様にした。

3.3　分析結果

3.3.1　作成されたモデル

- クラス図

  クラス図の全体を図1に示す。クラス図はハードウェア構成要素をのクラスを中心にして構成されており、その周囲に関連するクラスが存在している。

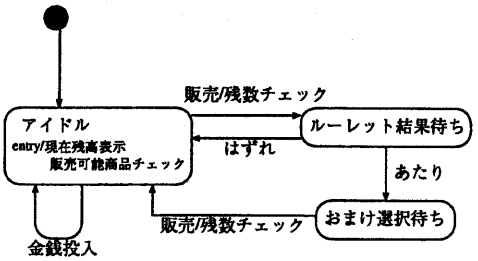

図2 状態図(自動販売機クラス)
Fig. 2 state diagram(vending machine class)

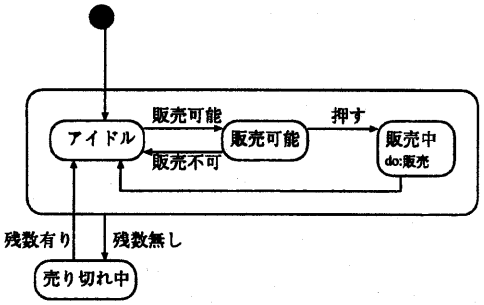

図3 状態図(販売ボタンクラス)
Fig. 3 state diagram(item button class)

紙幣処理装置クラスやプリペイドカード処理装置クラス、硬貨処理装置クラスは、どれも一定の金額の金銭を受け取る処理装置として類似の属性や操作を持っているので、これらを抽象化して金銭処理機器クラスが作成されている。

- 状態図自動販売機クラスと販売ボタンの状態図を図2と図3に示す。 各機器の状態は、図3に示す様に各々の機器に対応したクラスで管理する方針で状態図を作成してあるので、自動販売機全体を制御する自動販売機クラスの状態図自体は図2に示す様に非常に簡潔なものになっている。

3.3.2 仕様の抜け検出についての考察

各作業を行った時に、幾つかの仕様の抜け(ほとんどが動作の未定義)を検出した。主なものは以下の通りである。

- クラス図作成(関連付け)時
  - 販売ボタンとラックの対応関係には制約は無いのか?
- シナリオ作成時
  - 取り出し口に商品が残っている時に、金銭を投入したり販売ボタンを押したらどうなるか?
  - 釣銭切れ時に受け付ける操作やその動作は?
  - 販売可能時間以外に受け付ける操作やその動作は?
- 状態図作成時
  - 紙幣処理装置は複数の紙幣を受け付けるのか?
  - ルーレットで当たった時に返金ボタンを押すとどうなるか?

多くのものは、シナリオ作成や状態図作成の時点で検出された。これは、一般にデータ構造の定義よりも振る舞いの定義の方が厳密さを必要とすることや、自動販売機システムが取り扱うべきデータは少なく、ほとんど自明であること等が理由であると考えられる。

3.3.3 仕様変更への対応についての考察

同じような操作や属性をもったクラスが複数出てきた金銭を処理する機器については抽象クラスを作成できたので、仕様変更に対して対応しやすいと考えられる。

それ以外の部分に対しては抽象クラスを作成しなかったが、各クラス自体のもつ操作や属性が大きく変わらない範囲の仕様変更であれば、同様に抽象クラスを作成しておく事により対応出来ると考えられる。

3.3.4 再利用性についての考察

問題記述文中に出現する名詞句に注目することにより、ハードウェアのモジュール構成を分析モデルに作り込むことができたので、ハードウェアモジュールの構成が大きく変わらない限り、多くのクラスの再利用が可能であると考えられる。

また、各機器の状態をその機器に対応したクラスで管理している点も再利用性を高くしていると考えられる。

4. ユースケースに基づくモデル化

ユースケースの記述からオブジェクトを抽出し,オブジェクトモデルの構築を試みた.全体の分析手順は次のようになる.

1 問題理解のための漫画作成
問題を漫画で記述することで,問題領域に存在する事物を抽出できる.ここで抽出した事物がそのままオブジェクトとなるとは限らないが,問題の概要を理解する助けとなる.図4に,今回作成した漫画を示した.

2 システムの利用者を特定
与えられた問題では,購入者と販売者がこのシステムの利用者である.

3 利用者の状態遷移図を記述
ユースケースを記述するためには,システムと協調作業を行うアクターを特定しなければならない.アクターは,システムを利用する利用者に着目し,利用者が取り得る状態に対応して定義される.だから,アクターは利用者の状態遷移図を作

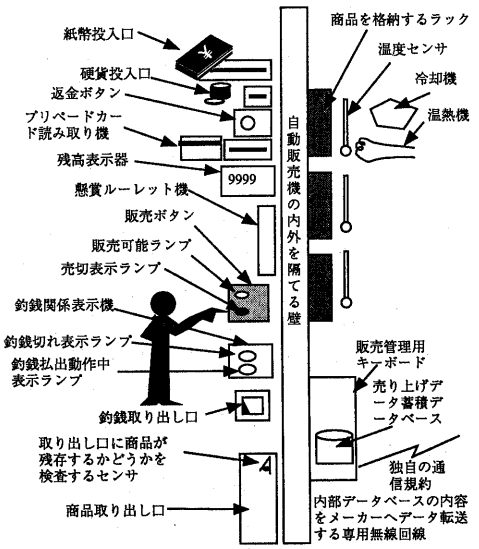

図4 問題の漫画を用いた記述

Fig. 4 Description of the problem by a picture

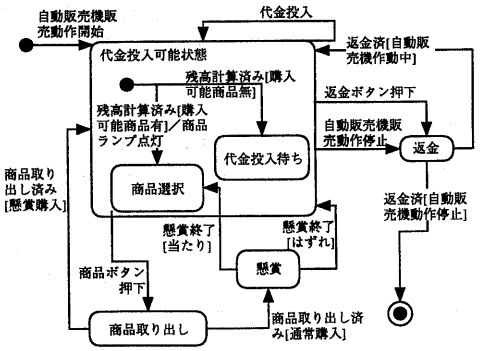

図5 購入者の状態遷移図

Fig. 5 State Transition Diagram of a purchaser

成することで抽出できる．図5に購入者の状態遷移図を示した．ここで，ある状態から次の状態へ遷移するときに発生する事象は，ユースケースの事前条件や事後条件となる．

4 アクターごとのユースケースを定義

ユースケースは，利用者とシステムをオブジェクトと捉え，相互のメッセージ送受信を時系列で記述するものである．ユースケースを単純化するためには，基本的な相互作用と異常が発生する場合の相互作用を分割して書く必要がある．したがって，ユースケースには，アクター，そのユースケースが起動するための事前条件および終了するための事後条件，基本的な相互作用を説明する基本系列，異常な相互作用を説明する代替系列を記述する．

5 ユースケースからオブジェクトを抽出

ユースケースは利用者とシステムの対話と解釈することもできる．ユースケースからオブジェクトを抽出するためには，利用者と対話を行って利用者が望むシステムのサービスを提供している「もの」を考える．これがオブジェクトの候補である．

6 抽出したオブジェクトをもとにした，オブジェクトモデルの構築

役割をもったオブジェクトをもとに，OMT法の表記を用いてオブジェクトモデルを構築する．OMT法のオブジェクトモデルの組み立て方法については，本稿の目的から外れるので省略する．

4.1 ユースケースの定義例

次に主なユースケースの例を示す．

ユースケース名: 代金投入

- アクター: 代金投入する購入者
- 事前条件: 自動販売機作動中，商品送出し中でない，および商品取出し済み，懸賞中でない，返金中でない
  注）これらの条件は，代金投入可能であることを表わす．
- 基本系列:
  - 紙幣投入口，硬貨投入口，プリペードカード読み取り機に手で貨幣を投入する．
  - 残高表示器に表示されていた額に，新たに投入された金額が合計されて表示される．
  - 投入された額に応じて購買可能な商品の商品ランプが点灯する．
- 事後条件: 販売可能な商品の商品ランプ点灯
- 代替系列:
  - 硬貨および紙幣が投入されて，合計金額が最大投入金額を超えた場合は，新たに投入された貨幣をそれぞれの返金口から返却する．
  - 千円札と認識できた紙幣以外の紙幣あるいは紙などの不良貨幣が投入されたときは，それらを投入口から返却する．
  - 500円，100円，50円，10円硬貨と認識されなかった異物は釣銭取り出し口から返却する．
  - 有効期限切れ，残高のないプリペードカードは投入口から返却する．
  - 商品を選択したくなくなったとき→返金ユースケースへ
  - 自動販売機作動停止になる時，xxxの条件が満たされたら返金ユースケースへ

ユースケース名: 商品選択

- アクター: 商品選択する購入者
- 事前条件: 購入可能な商品の商品ランプ点灯
- 基本系列:
  - 購入希望の商品の点灯している商品ボタンを押す．
  - 当該商品以外のボタンが消灯する．
  - 当該商品ボタンが点滅して販売動作中であることを示す．
  - 当該商品ボタンが消灯して販売動作が完了したこと

を示す.
  - 残高表示器の表示が，販売した商品の額だけ引いた残高に変更される.
  - 販売商品取り出し口に当該商品が排出される.
  - 販売商品取り出し口から商品を手で取り出す.
- 事後条件: 商品取り出し口には商品が残されていない
- 代替系列:
  - 釣銭切れで，購入希望商品の販売価格よりも大きい紙幣，硬貨が投入されていたときは，商品を販売せず，それぞれの返金口から今投入されたお金を返却する.
  - 代金投入による商品選択の場合は，基本系列完了後懸賞ユースケースへ
  - 代金投入による商品選択の場合で，代金を追加する場合は代金投入ユースケースへ
  - 懸賞による商品選択の場合は，xx 秒間だけ商品選択が可能であり，xx 秒経過または商品選択の基本系列完了後，代金投入ユースケースへ

ユースケース名: 返金
- アクター: 返金を要求する購入者
- 事前条件: 代金投入可能状態である
- 基本系列:
  - 返金ボタンを押す.
  - 釣銭払動作中のランプが点滅する.
  - 投入されたプリペードカード，紙幣，硬貨から順番に購入した商品の代金が引かれ，残高がそれぞれの返金方法に則って返却される.
  - 返金に伴って，残高表示が減算される.
  - 残高表示がゼロになる.
  - 釣銭払動作中のランプが消灯する.
- 事後条件: 残高表示がゼロである
- 代替系列:

ユースケース名: 懸賞
- アクター: 懸賞を実施する購入者
- 事前条件: 商品取り出し済みで，直前の商品を通常購入した
- 基本系列:
  - ルーレットが回り，その間，商品ボタンが点滅する.
  - ルーレットが止まる.
  - 商品ボタンが消灯する.
  - 懸賞の結果が表示される.
- 事後条件: 懸賞の結果が当たり，はずれのいずれかであることが確定している
- 代替系列:
  - 懸賞結果が当たりの場合→懸賞当選ユースケースへ
  - 懸賞結果がはずれの場合→懸賞落選ユースケースへ

ユースケース名: 懸賞当選
- アクター: 懸賞に当選した購入者
- 事前条件: 懸賞が終了し，懸賞結果が当たりである
- 基本系列:
  - 「当たり」ランプが点灯する.
  - 懸賞購入可能な商品ランプが点灯する.
- 事後条件: 商品選択ユースケースへ移る
- 代替系列:

ユースケース名: 懸賞落選
- アクター: 懸賞にはずれた購入者
- 事前条件: 懸賞が終了し，懸賞結果がはずれである
- 基本系列:
  - 「はずれ」ランプが点灯する.
  - xx 秒後「はずれ」ランプが消灯する.
- 事後条件: 代金投入ユースケースへ移る
- 代替系列:

ユースケース名: 商品価格設定
- アクター: 商品価格を設定する販売者
- 事前条件: 自動販売機が販売動作停止状態にある
- 基本系列:
  - 販売動作不可能状態となる.
  - ラックと商品ボタンの対応関係を設定する.
  - 商品ボタンに対応する商品名と価格を設定する.
  - 販売動作可能状態となる.
- 事後条件: 商品が販売されたら該当する商品の価格を販売実績として記録できる
- 代替系列:

ユースケース名: 最大投入金額設定
- アクター: 販売機への最大投入金額を設定する販売者
- 事前条件: 自動販売機が販売動作停止状態にある
- 基本系列:
  - 販売動作不可能状態となる.
  - 販売機への最大投入金額を設定する.
  - 販売動作可能状態となる.
- 事後条件:
- 代替系列:
  - 設定された最大投入金額が，xx 円以下または yy 円以上である場合は設定を無効とし，販売動作不可能状態のまま変化しない.

ユースケース名: 販売可能時間帯設定
- アクター: 販売機の販売可能時間帯を設定する販売者
- 事前条件: 自動販売機が販売動作停止状態にある
- 基本系列:
  - 販売動作不可能状態となる.
  - 販売機の販売可能時間帯を設定する.
  - 販売動作可能状態となる.
- 事後条件:
- 代替系列:
  - 設定された時間帯は販売動作開始時刻と販売動作停止時刻とし，開始時刻が停止時刻を超えた場合は，設定を無効とし，販売動作不可能状態のまま変化しない.

ユースケース名: 販売実績入手
- アクター: 販売機の販売実績を入手する販売者
- 事前条件: 自動販売機が販売動作停止状態にある
- 基本系列:
  - 販売動作不可能状態となる.
  - データベースから，商品名，通常販売個数，懸賞販売個数の組みを商品種別数を販売実績として，販売実績を規定のプロトコルに則って送信する.
  - データベース内の販売実績を消去する.
  - 販売動作可能状態となる.
- 事後条件:
- 代替系列:
  - 通信が中断された場合は，規定の手順で販売実績を再送信する.

ユースケース名: 異常事態発生
- アクター: 異常事態が発生した販売機
- 事前条件: 自動販売機が作動中で，異常事態を検知する
- 基本系列:
  - 異常事態を知らせるアラームを鳴らす.

- 代金投入口を禁止する．
- 作動中止状態となる．
- 販売者は異常事態を解消し，作動状態とする．

- 事後条件: 作動状態となる
- 代替系列:
  - 販売実績送信中は，送信を中止する．
  - 返金中は，排出中の貨幣，プリペードカードを排出する．
  - 商品排出中は，排出中の商品を排出する．
  - 懸賞実施中は，懸賞を終了し，強制的に「はずれ」にする．

4.2 オブジェクトの抽出

オブジェクト抽出の例として，各ユースケースの基本系列をもとに，利用者とシステムの対話を記述してみた．

- 代金投入ユースケースより

利用者「このお金を受け取って下さい」→代金投入
　システム「受け取りました」→受け取ったのは代金投入口
  - 「受け取ったお金は○○円です」→表示したのは残高表示器
  - 「あなたが購入できる商品は××です」→商品ランプ点灯

- 商品選択ユースケースより

利用者「私はこの商品を選択します」→商品ボタン押下
　システム「あなたが希望した商品はこの商品です」
　　→商品ランプ点滅→ボタンとラックの関係がわかる
  - ガチャン（商品排出）
  - 「お金がまだ余っています」→販売した商品の価格から残高を計算できる

- 返金ユースケースより

利用者「残りを返してください」→返金ボタン押下
　システム「返します」→釣銭払動作中のランプ点滅
  - ガチャン（返金）→返金額をどの貨幣で返金するかがわかる

以上の対話から，システムを構成するいくつかのオブジェクトとその役割を抽出する．たとえば，代金投入口がお金を受け取ったのであれば，代金を受け取り，その金額を判定する役割を持っているオブジェクトが代金投入口になる．また，押下されたボタンから商品を特定して排出するためには，ボタンと商品が格納されているラックとの関係を知っているオブジェクトがいなければならない．さらに，販売した商品の金額を伝えられるオブジェクトが存在しなければ，販売した商品の価格から残高を計算することはできない．これは同時に，投入された代金の合計金額を知っているオブジェクトが存在することも意味する．残高に表示されている金額を返金する責任を持っているのは返金ボタンである．

オブジェクトの抽出の作業とは，このように，責任を任せるオブジェクトを発見し適切な名前をつけることである．名前にはボタンや表示器という名前は，単機能を提供するオブジェクトのように見えるので，オブジェクトが持つ責任を表現できるような名前がよい．だから，

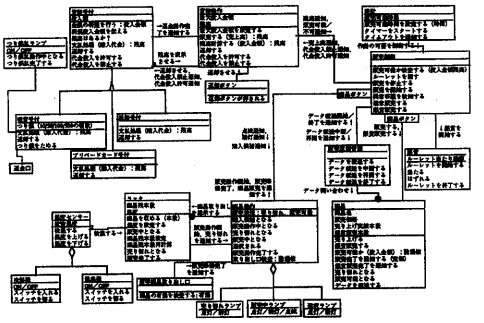

図6 ユースケースを基に記述したオブジェクトモデル
Fig. 6 Object Model defined from usecases

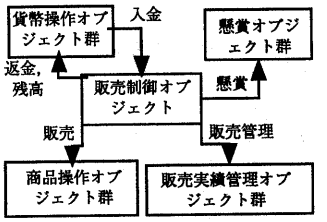

図7 モデルのアーキテクチャ
Fig. 7 Architecture of the model

最初に作成した漫画に登場したアイコンの名前が必ずしもモデルのオブジェクト名になるとは限らない．たとえば，残高表示器を，投入された貨幣を管理し，投入口を制御するオブジェクトとしてモデル化するのであれば，その名前は「貨幣操作」とした方が良いだろう．

4.3 オブジェクトモデルの作成

ユースケースから抽出したオブジェクトを用いて作成したオブジェクトモデルを図6に示した．このモデルを簡略化してアーキテクチャがわかるように書き換えたモデルが図7である．図7は，次の5つのオブジェクト群から構成されている．まず，投入された貨幣を管理し，返金，残高を求める役割を持つ貨幣操作オブジェクト群，ラック内の商品の状態を制御し，押下されたボタンに対応する商品を排出する役割を持つ商品操作オブジェクト群，懸賞を実施し結果を求める役割を持つ懸賞オブジェクト群，販売実績の記録と転送を行う販売実績管理オブジェクト群，以上のオブジェクト群間の協調作業を運行する役割を持つ販売制御オブジェクト群である．

4.4 手法の評価

ここまで示した手法を次の評価点で評価する．

- 要求項目に対応すべきシステム要素の抜けを検知しやすいか
- 複数の解釈 / 選択可能性や変更要求に対応しやすい

か

- 再利用性が高いか

4.4.1 システム要素の抜けの検知しやすさ

ユースケースを用いると，利用者とシステムの対話を時系列で定義することになるので，要求項目に対応すべきシステム要素の抜けを検知することが比較的容易にできる．先に示したユースケースの太字部分が，ユースケースを記述するときに検知したシステム要素の抜けと思われた部分である．ここで補ったユースケースの記述は，実際には顧客と相談しなければ決定できない．

4.4.2 変更要求への対応しやすさ

図6のオブジェクトモデルが図7の5つの部分に分かれていることを説明した．それぞれの部分はカプセル化されているため，それぞれのカプセルは，さらに効率的な設計モデルと置換可能である．たとえば，商品操作オブジェクトには，売切れランプ，販売可能ランプ，温度を表わすランプなど，多くのランプがカプセル化されている．そのため，実際にどのようなボタンを付加してラック内の商品の状態を表わすかは設計の自由であり，この部分を変更しても，他のオブジェクトには変更の影響が及ばない．

設計段階では，再利用部品を考慮したモデルへの変換も必要となるが，このシステムのアーキテクチャに変更の必要がない限り，変更要求には柔軟に対応できるだろう．

4.4.3 再利用性の高さ

再利用性については，他の切符の自動販売機やたばこの自動販売機といった類似システムのモデルを検討していないので，あまり議論できない．しかし，ほとんどの自動販売機に，図7で示したアーキテクチャを適用できるはずである．たばこや切符の自動販売機では懸賞オブジェクトは必要ないが，このような部分的な削除のような変更の労力はそれほど多くはないだろう．

5. ソフトアーキテクチャに基づくモデル

良いアーキテクチャを持つソフトウェアは良い品質(変更し易さ，理解し易さ，保守性が高い等の性質)を持つと言われている．生産性の向上や再利用の鍵の1つとして，これまでに，このソフトウェアアーキテクチャに関して，多くの研究が行なわれてきた[1)~8)]．われわれは，5年超にわたる自動販売機制御ソフトウェアの開発に関する研究を通して，領域特定のソフトウェアアーキテクチャ(Domain Specific Software Architecture)である自動販売機3層モデル(Vendinga Machine 3-Layered Model)を定義した．ここでは与えられた仕様を自動販売機3層モデルに基づいて分析・設計した例を示す．

5.1 自動販売機3層モデル

自動販売機3層モデルの概略を図8に示す．図にあるように，自動販売機3層モデルは自動販売機制御ソフトウェアは3つの層から構成されていると仮定している．それぞれを応用層，制御層，モデル層と呼ぶ．

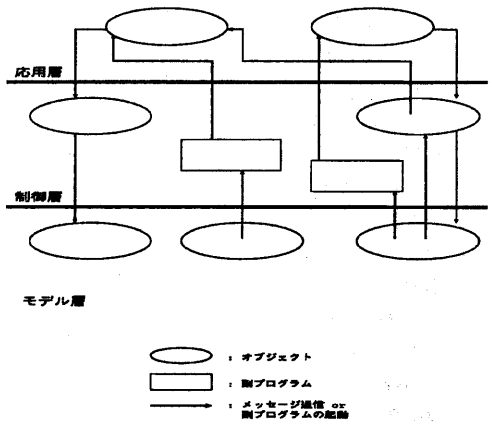

図8 自動販売機3層モデル

それぞれの層の構成要素はオブジェクトか副プログラム(手続きまたは関数)である．すべての構成要素がオブジェクトでない理由は以下のとおりである．自動販売機は使用者等の行動(例えばお金を入れるなど)をきっかけに内部の処理が起動される機械である．このような機械では通常ハードウェアがイベントを発生させソフトウェアの処理のきっかけを作る．並行オブジェクト指向の計算モデルがこれらのソフトウェアの実現に適したモデルではあるが，自動販売機のハードウェアにその処理を行なうだけのCPUを搭載することは費用の面から無理がある．並行オブジェクト実行の仕組みを実現できない場合は，イベントを処理・加工する副プログラムを作成する必要がある．自動販売機3層モデルはこれらの制約等を素直に採り入れた形式で考えたモデルである．

応 用 層

制御ソフトウェアの論理をこの層で実現する．この層のオブジェクトと副プログラムが通信し合いながら論理を実現する．この層の構成要素は下位の制御層から起動される．

制 御 層

単機能のハードウェアを実現する層．この層のハードウェアは実際のハードウェアの一部または全部の仮想ハードウェアである．この層はハードウェアの変更ならびに応用論理の変更を吸収する．

モデル層

実際の自動販売機のハードウェアを構成要素として実現する層.

5.2 自動販売機3層モデルに基づく分析

まず, 仕様から識別できるオブジェクトを以下に示す:

- 札処理装置.
- 硬貨処理装置.
- プリペイドカード処理装置.
- 現在残高表示装置.
- つり銭表示装置.
- 選択ボタン.
- 販売中表示器.
- 売り切れ表示器.
- ラック.
- 温度センサ.
- データベース.
- 通信ポート.
- 温熱機.
- 冷却機.

図9は, これらのオブジェクトを3層に配置したものである. 図にあるように, モデル層のオブジェクトは実際に自動販売機のハードウェアに存在するものであり, 通常これらは制御層にある単機能のハードウェアモデルに分割できる. 例えば, 実際の自動販売機のハードウェアに存在する販売ボタンには, ボタンと売り切れ表示器, 販売可能表示器が一体となって組み込まれている. 制御層の, 販売可能ランプ, 売り切れ表示ランプ, ならびに販売ボタンは, モデル層の販売ボタンを単機能に分解したものになる. また, モデル層のラックは単機能に分解すると, 制御層のラック, 温度センサ, およびヒータまたはクーラの3つになる. さらに, モデル層の表示器は制御層の現在残高表示装置とつり銭表示装置に分割できる.

応用層では, 自動販売機を金庫, 商品貯蔵庫, データベースの主要3構成要素に分割している. これらのオブジェクトに金銭処理, 商品選択, およびデータベース処理のデータと応用論理を実現する.

応用層のオブジェクトは制御層の単機能ハードウェアモデルを介してハードウェアモデルとメッセージを交換する. 図10に硬貨投入のさいのメッセージの授受を示す. 実装のさいには, 制御層の硬貨処理装置は副プログラム (硬貨入力処理) となる. このように, 分析の段階ではオブジェクトとして認識したものを, 必要に応じて, 副プログラムまたはオブジェクトとして実装する. 副プログラムとなる可能性のあるものは, 応用層のオブジェクトに限られる.

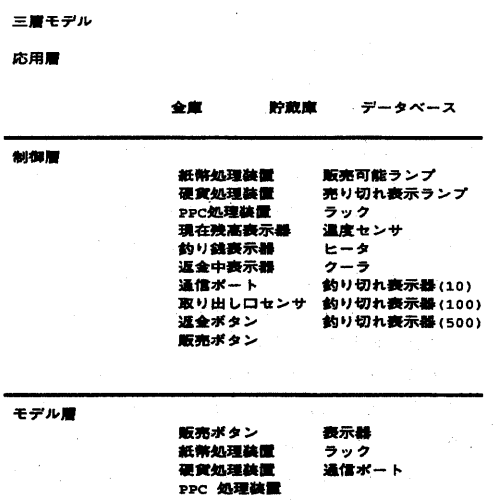

図9 自動販売機3層モデルへのオブジェクトの配置

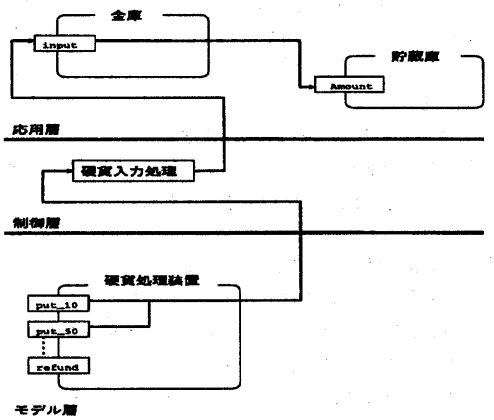

図10 メッセージ交信

5.3 まとめ

本節では, ソフトウェアアーキテクチャに基づく分析結果を簡単に述べた. このアーキテクチャに基づく実際のソフトウェアの書き直しとアプリケーションフレームワークの実装プロジェクトが現在進行中である.

6. おわりに

このほか, 会場からもいくつかモデル提案があった. この例題は制御系の問題なので, 大きなデータストアが見当たらない. したがって, オブジェクトの同定や機能モデルの展開にとまどうところがあると思われる. これに対して, 時間的, 空間的に小さいデータであっても, 複数の機能から参照されなければならないものをデータストアとして抽出し, それを核としてオブジェクト,

メソッドを組み上げるという方策が示された．☆複数のデータストアの集約のしかたやメソッドの配置に対する設計判断がこのあとに必要であるが，CPU 構成などの外部的制約条件を考慮しながら，オブジェクト間フローの極小化をはかって分割を決める．

また，ユースケースをイベント入出力系列としてとらえ，これをもとに状態遷移モデルをまっさきに固めるというモデリング法も提案された．☆☆今回の例題のような制御系の問題では自然なプロセスかもしれない．状態遷移モデルをつくるにあたって，内部イベントとそれを発生する内部アクションの認識が必要となるが，これと外界とのデータ送受を行う入出力アクションの振り分けによってオブジェクト分割をすすめる．分割の指針を与えるのはやはりオブジェクト間の通信コストであって，マスタ CPU の有効性およびそこに割り付けるべき機能もそれによって決まる．

以上いろいろなアプローチによるモデリングを試みたが，モデリング結果のゆらぎ（クラスの同定 / 分割，クラス間関係の設定，メソッドの同定 / 配置）は予想したほど大きくなかった．ハードウェア的な既定制約条件の網がかなり強くかかった問題であったことがその主因であると考えられる．ビジネスモデル（問題ドメイン，人間系，コンテクストモデル）を用いるかシステムモデル（システムドメイン，機械系，内部モデル）を用いるかという違いや，あるいはひとつのモデル内でのそれらの混同も見られたが，この例題の場合，これら両系統のモデル要素間の対応性がかなりよいため，重大な問題となっていない．逆にいうと，アプリケーションを形成する実体や関連に対する制約条件を精密化することが重要で，モデリングは（設計につなげる具体化を提供することにもまして）それを助けることを第一義とすべきではないかと思われる．

参 考 文 献

1) R. Allen, *et. al*, "Formalizing Architecture Connection," *Proc. 16th ICSE*, 1994.
2) T. Arano, *et. al*, "A Computer Network Management System Paltform based on Distributed Objects," *6th IFIP/IEEE DOSCOM'95*, 1995.
3) F. Buschman, *et. al*, *Pattern-Oriented Software Architecture – A System of Patterns*, Wiley, 1996.
4) J. J. Donovan, *Business Re-engineering with Information Technology*, Prentice Hall, 1994.
5) D. Garlan, *et. al*, "Exploiting Style in Architectural Design Environments," *Proc. ACM SIGSOFT'94 Symposium on Foundation of Software Engineering*, 1994.
6) G. E. Krasner and S. T. Pope, "A Cookbook for Using the Model-View-Controller User Interface Paradigm in Smalltalk-80," *J. of Object-Oriented Program*, Vol. 1, No. 3, pp.22-49, Aug./Sep. 1988.
7) P. B. Kruchten, "The 4+1 View Model of Architecture," *IEEE Software*, Vol. 12, No. 6, pp.42-50, Nov. 1995.
8) M. Shaw, *Software Architecture, Perspective on an Emerging Discipline*, Prentice Hall, 1996.

☆ 沢田篤史氏（京大）による．
☆☆ 満田成紀氏（和歌山大）による．