# MDA を用いたサービス非依存インタラクションモデルの構築

天川美那†　　松浦佐江子‡
芝浦工業大学　大学院工学科　電気電子情報工学専攻†
芝浦工業大学　システム工学部　電子情報システム学科‡

## 1　はじめに

複雑なシステムを解決するための手段として，MDA(Model Driven Architecture)のプラットフォーム分割手法[1]を用いる事を提案する．

MDA のプラットフォーム分割により，システムの入力部を分離する事でシステムのロジックを変更せずに入力機器を変更した．しかしこの方法では，異なるシステムで同様の入力機器の切り替えを行いたい場合，システム作成のプロセスを初めからやり直す必要がある．

システムが安全に動作するためには環境（ユーザ・外部システム・ハードウェア機器）に対応したサービスの入出力の制約を漏らさず定義しなければならない．サービスのアルゴリズムの複雑さを解消し，信頼性を向上させるためにインタラクションとシステムのサービスとの責務を明確にするためのインタラクションモデルを提案する．

しかし，あらゆるシステムのあらゆるインタラクションをあるクラス群で抽象化する事はできない．そこで，本研究では図書の賃借を管理するシステム開発事例を通じて，責務の切り分けを明らかにする．具体的には，想定するインタラクションモデルに対して複数の要求に対応したインタラクションパターンを PSM として定義し，それを具体的な入出力デバイスに対応できるように変換してシステムを構築したので，この結果を報告する．

## 2　インタラクションの位置づけ

### 2-1　インタラクションとは

インタラクションはユーザとシステムの相互作用を指す．システムは入出力データの型や値の範囲，一度にやり取りされるデータの数等を厳密に規定されて作られる．この一連の取り決めをインタラクションパターンと呼ぶ．システムが正しく動作するためには，ユーザの入力がインタラクションパターンにマッチしていなければならないため，これを精査する必要がある．

### 2-2　システムにおけるインタラクション

図 2.1 のユースケースモデルのように，システムは線(関連)と丸(サービス)から形成される．インタラクションは関連の一種である．一般にインタラクションはサービスロジックに付随して定義され，システムのインタラクションパターンに依存する形で作られる．本研究ではインタラクション機構の中に入力精査の処理を設けてインタラクションパターンの差異を吸収し，サービスからインタラクションを切り離す事で依存を解消する．

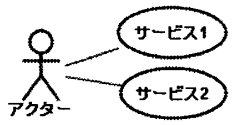

図 2.1　システム概略

## 3 図書管理システム

ケーススタディとして図書の賃借を管理するシステムを作成した[2]．提供するサービスは貸出・返却・検索の三種とそれに付随する副次的機能である．作成に際してサービスロジック部及び関連にあたるインタラクション部とユーザ認証部の三つのサブシステムを定義する．まずサービスをプラットフォームと捉え，それに依存しないインタラクションとユーザ認証の PIM (Platform Independent Model)を作成した．次に図書管理システムのサービスに合わせて具体化した PSM (Platform Specific Model)を作成し，サービス部のモデルと組み合わせて全体を構築した．また，入力機器としてキーボードとカード R/W 機構の FeliCa による二通りの実装を行った．

### 3-1　インタラクションモデルの問題点

図書管理システムにおけるインタラクションパターンは一入力につき一データを取得して精査し，アクションを一度だけ起動して出力を行うというものであった．作成したインタラクション PIM は上のインタラクションパターンに特化していた．また，このモデルでは入力精査をデータ型と入力桁数についてのみ行っていたが，インタラクションをパターン化するにはこれらの項目だけでは不足である．このため，作成したモデルを他のパターンについても汎化すべく改良を行った．

### 3-2　改良の指針

サービスロジックの責務とインタラクションの責務の境界を決める．2.1 項の定義に基づき，本研究ではインタラクションに入力受付・結果出力・入出力精査の機能を持たせた．この時，動的な情報を参照しないものをインタラクションの責務とした．例えば ISBN コードを利用するサービスは，これが 9 桁の数字である事を前提として定義される．当該図書が現在貸出中か否かについては，システム内で生成される図書データが複数のサービスによって参照・更新される事から，これはサービスにおいて精査すべきデータである．

上を踏まえ，インタラクションを再定義して実現

An MDA-based Service Independent Interaction Model
†芝浦工業大学大学院工学研究科電気電子情報工学専攻,
Graduate School of Engineering, Shibaura institute of technology
Department of electronic engineering and computer science
‡芝浦工業大学システム工学部電子情報システム学科,
Shibaura institute of technology Department of electronic information system

したのが図 3.1 のモデルである．まずアクションが直接入力・出力を持つのではなく，代わりにパラメータと実行結果を持つ．得られた入力をパラメータとして解釈し，一つずつの実引数に分解して精査を行う．このパラメータを元にアクションを起動して得られた結果を実行結果として保持・出力する．また，インタラクションが一つのサービスに対して起動させるアクションの系列を持つ．更に前回アクションの実行結果を事前条件として保持し，次回アクションのパラメータに与えて次回入力の補助とする．

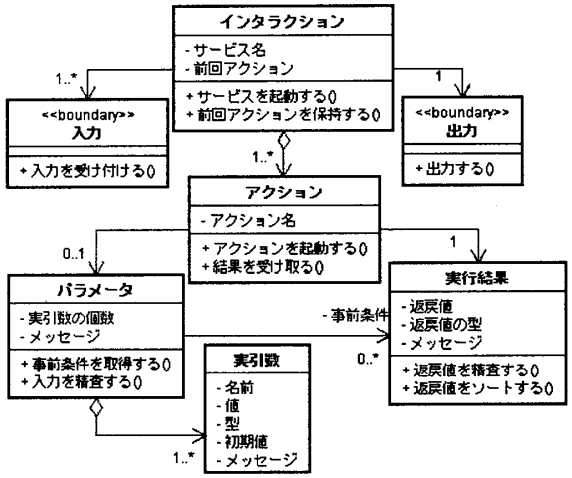


図 3.2 改良後のインタラクションモデル

具体例として貸出処理の場合を見る．サービス「貸出」で行われるアクションは図書検索・貸出状態変更・帯出者書き込みの三つである．この中でパラメータを必要とするアクションは図書検索であり，処理のためにユーザ ID と ISBN が必要になる．ここで二種の異なるインタラクションパターンを考える．

1. ID と ISBN を一つずつ別の入力として取得し，そのつど精査を行う．それぞれの適正な値が入力されるとデータの取得を終了し，パラメータをアクションに渡してアクションを起動する．
2. 一度の入力で複数のパラメータを取得し，それを一度分解して精査を行ってから貸し出す冊数の回数アクションを起動させて処理を行う．

## 4 実験結果

PIM を PSM に変換する．PSM は図 3.1 の PIM を継承し，メソッドの内容を定義して足りない属性を補う事で作成する．要求としてインタラクションパターンとシステムを実現するインターフェースの実装を踏まえる．

PIM→PSM の変更箇所を表 4.1・表 4.2 に示す．パターン項は二つのインタラクションパターンを CUI 及び web ブラウザの二種のインターフェースで実装した場合を指す．ただしパターン 1 と web ブラウザの組み合わせは性質上考えにくいため省略した．精査時の入力値解釈は入力されたデータの処理方法であり，「入力を受け付ける」「入力を精査する」の二つのメソッドで差異を実現する．その他の項目はモデル変換の際に各クラスに追加する要素である．

なお，セパレータは一度の入力で複数の値を得る際に値を区切るために使用するトークンを指す．web ブラウザで入力を行う場合はフォームの名前で値を識別するため，通常は使われない．

表 4.1 変更点一覧 1

| パターン | 入力方法 | 精査時の入力値解釈 |
|---|---|---|
| CUI1 | 標準入力 | 規定個数を満足するまで実引数を入力させる |
| CUI2 | 標準入力 | 二種のセパレータにより複数パラメータ・実引数に分解する |
| Web2 | 入力フォーム | フォームの名前と実引数の名前を照合する |

表 4.2 変更点一覧 2

| パターン | パラメータクラス属性 | 入力クラス属性 |
|---|---|---|
| CUI1 | 入力値の個数 | 変更なし |
| CUI2 | 実引数セパレータ | パラメータセパレータ<br>入力組数 |
| Web2 | 変更なし | 入力組数 |

パターン 1 の場合はパラメータクラスが入力された実引数を精査のあとで保持し，アクションを呼び出すのに必要な実引数が揃うまで入力を続ける必要がある．パターン 2 の場合，入力で得られたパラメータの数は入力クラスが数える．各パラメータの実引数についてパラメータクラスが精査し，値が正当であればアクションを呼び出すという処理が必要になる．更に拡張例として，図書検索アクションの前に現在の貸出冊数表示アクションを追加する事を考える．この場合，アクション固有のメッセージや精査内容を付加する必要がある．しかし入力を精査してアクションを呼び出し，その実行結果を次回入力の初期値またはメッセージとするという処理の流れは変わらないため，上に列挙した PIM→PSM の変換規則も変わらない．これはサービスの仕様からインタラクションが独立している事を示すものであると言える．

## 5 今後の課題

本稿では図 3.1 の PIM を二種のインタラクションパターンについて具体化するプロセスを述べた．この PIM が異なるパターンやインターフェースについてどの程度対応しうるのかを検証する．多様なインタラクションの全てのパターンを一つの PIM で表現するのは現実的でないと思われるため，この PIM の適用範囲を探り，境界を明示する必要がある．

## 6 参考文献

[1] MDA(Model Driven Architecture), http://www.omg.org/mda/, Object Management Group

[2] 天川美那，松浦佐江子：MDA におけるプラットフォーム分割手法の妥当性の検証，情報処理学会第 69 回全国大会，6L-6，2007