オブジェクト指向言語（2010年度）
オブジェクト指向分析・設計

桑原 寛明
情報理工学部 情報システム学科

オブジェクト指向分析

オブジェクト指向モデリング

- 機能的側面：システムの機能を定義
  - ✓ ユースケース図：システムの提供する機能と利用者の関係
- 静的側面：システムの構造を定義
  - ✓ クラス図：クラスの構造(属性や操作)とクラス間の静的な関係
  - ✓ オブジェクト図：ある時点でのオブジェクトの状態とオブジェクト間の関係
- 動的側面：システムの振る舞いを定義
  - ✓ 相互作用図
    - ー シーケンス図：オブジェクト間の相互作用の時系列
    - ー コミュニケーション図：オブジェクト間の相互作用のリンク
  - ✓ アクティビティ図：作業の順序と並行性
  - ✓ 状態機械図：オブジェクトの状態とイベントによる状態遷移
- 物理的側面：システムの物理的制約を定義
  - ✓ コンポーネント図：コンポーネントの構造と依存関係
  - ✓ 配置図：システムにおける物理的な配置

オブジェクト指向モデリング(cont'd)

- 概念モデル：ドメイン分析(業務状況分析)により作成
  - ✓ 対象業務の世界を構成する概念と概念間の関係を表すモデル

会員とは?
入会とは?

- 要求モデル：要求文書から作成
  - ✓ 顧客がシステムに望む事柄を表すモデル

オンライン会員
登録をしたい

- 分析モデル：概念モデル + 要求モデルから作成
  - ✓ 実装方法に関知せずに，対象業務についてシステム化する方法を表すモデル
- 設計モデル：分析モデルから作成
  - ✓ システム化する事柄と，その実装方法を詳細に表すモデル

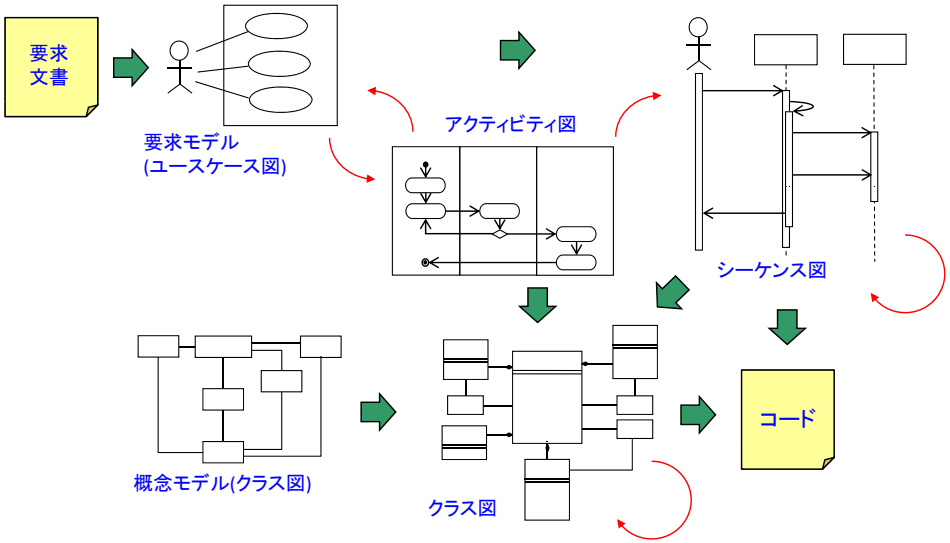
画面で氏名や住所
を入力して，…

- 実装モデル：設計モデルから作成
  - ✓ プログラムソースコード，バイナリプログラム
  - ✓ プログラムの計算機上の配置

Web経由で氏名を会員
オブジェクトに格納して，
…

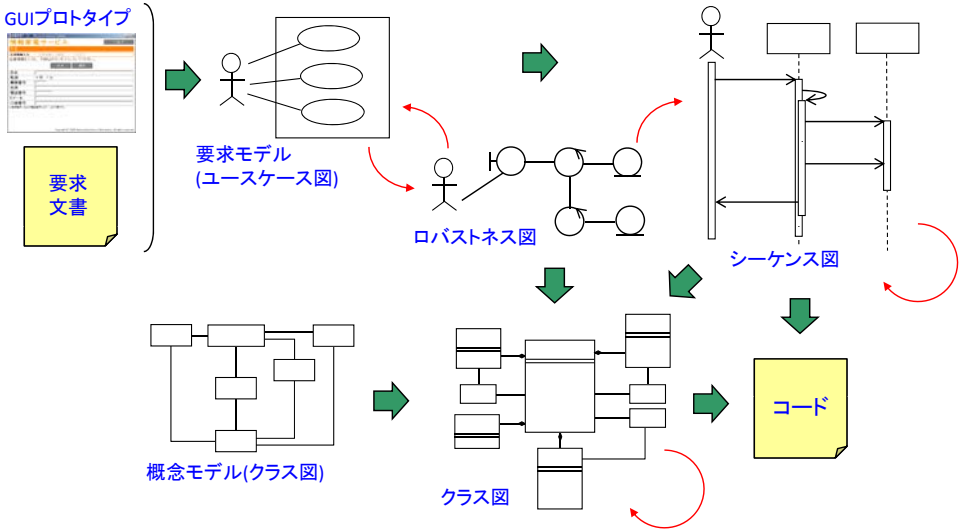
```
class Member {
  …
```

モデリングプロセス

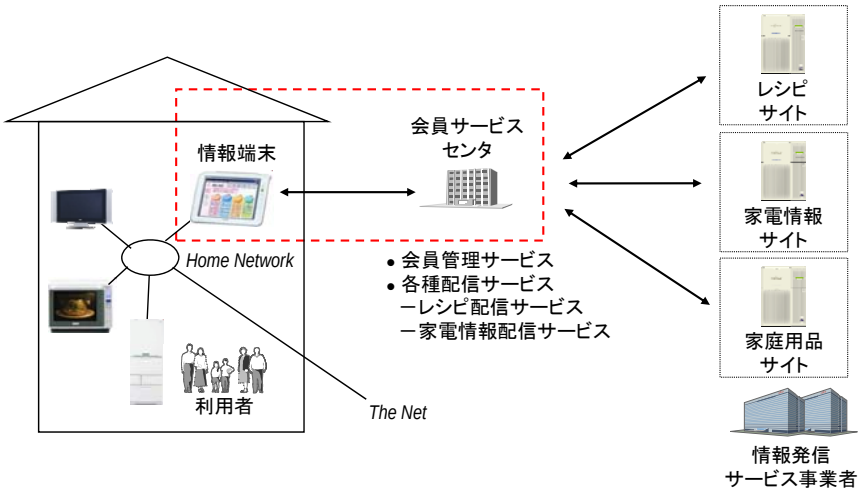


OOモデリングプロセス(ICONIX)

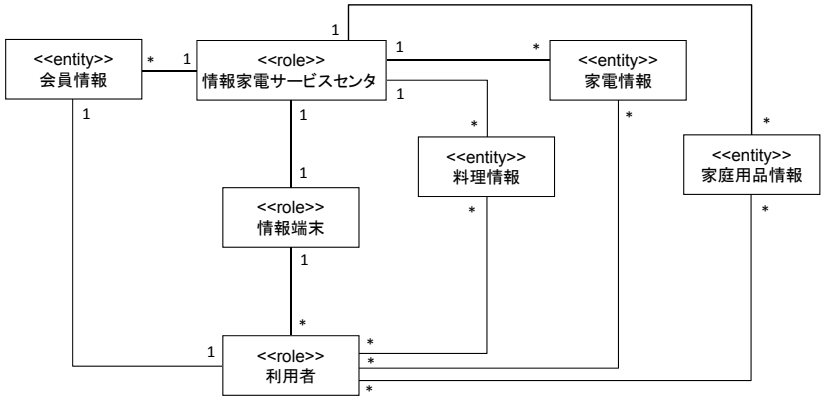
- ICONIX社が提案するBooch法，OMT, OOSEを統合したアプローチ



例題：情報家電サービス

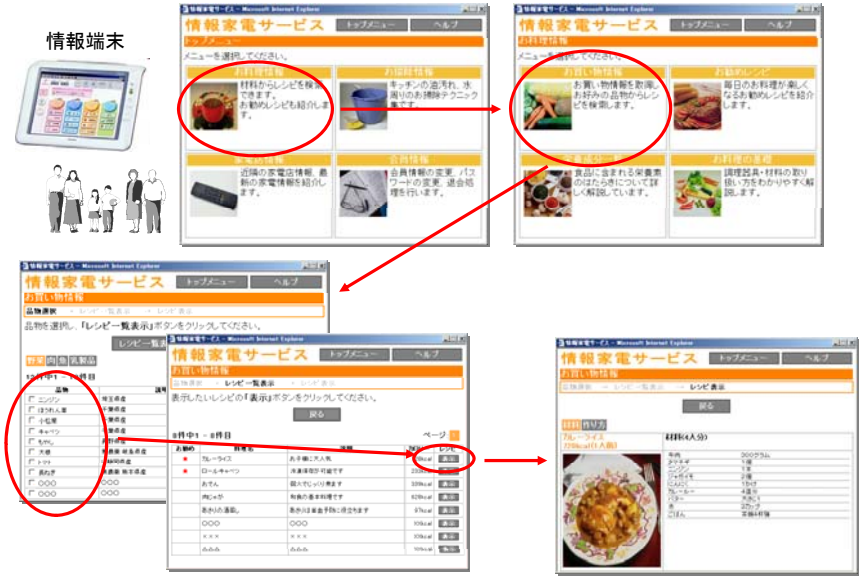


概念モデル

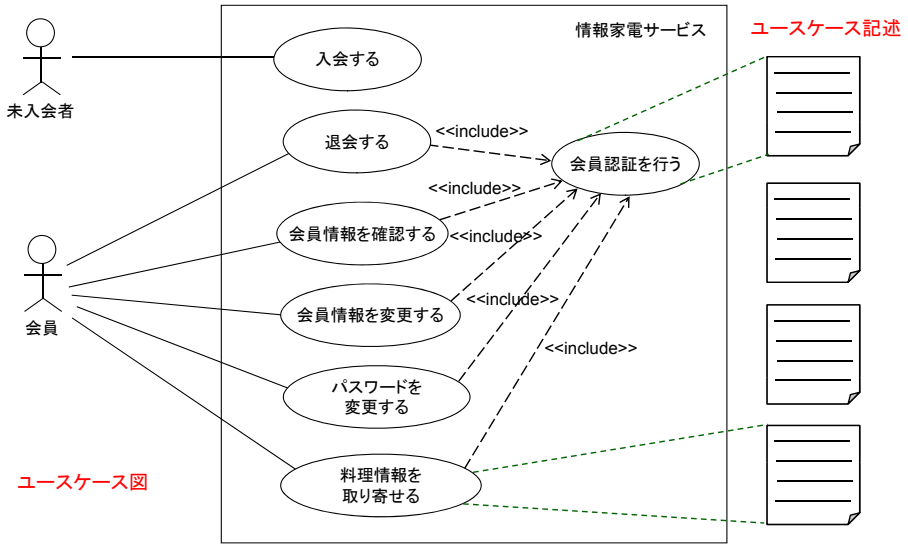


利用画面遷移イメージ(1)

利用画面遷移イメージ(2)



要求モデル



ユースケース記述：入会する

Name 入会する
Initiator 未入会者
Goal 情報家電サービスの会員になる

Main Success Scenario
1. 情報家電サービスに接続し，「入会」を選択する.
2. 新規登録に必要な情報を入力する.
   2.1 氏名を入力する.
   2.2 郵便番号を入力する.
   2.3 住所を入力する.
   2.4 電話番号を入力する.
   2.5 Eメールアドレスを入力する.
3. システムは入会を受付け，未入会者に会員番号を発行する.
4. システムは，会員番号と初期パスワードを表示する.

Extensions
3. 既に入会済みである（氏名とEメールアドレスに基づく）.
   a. システムは既に入会済みであるメッセージを表示する.
   b. 失敗.
3. 未入力項目が存在する.
   a. 未入力項目を指摘する.
   b. 2に戻る.

ユースケース記述：退会する

Name 退会する
Initiator 会員
Goal 情報家電サービスの会員をやめる

Main Success Scenario
1.情報家電サービスに接続し，「トップメニュー」を選択する.
2. 会員認証を行う.
3.「会員情報」を選択する.
4.「退会」を選択する.
5. システムは退会を受付け，退会処理を行う.
6. システムは退会完了メッセージを表示する.

Extensions
3. 認証に失敗した.
  a. 失敗.

ユースケース記述：会員情報を確認する

Name 会員情報を確認する
Initiator 会員
Goal 会員の登録情報を確認する

Main Success Scenario
1. 情報家電サービスに接続し，「トップメニュー」を選択する.
2. 会員認証を行う.
3.「会員情報」を選択する.
4.「会員情報確認」を選択する.
5. システムは該当する会員を検索し，会員情報（会員番号，氏名，
  郵便番号，住所，電話番号，Eメールアドレス）を表示する.

Extensions
3. 認証に失敗した.
  a. 失敗.

ユースケース記述：会員情報を変更する

Name 会員情報を変更する
Initiator 会員
Goal 会員の登録情報を変更する.

Main Success Scenario
1. 情報家電サービスに接続し，「トップメニュー」を選択する.
2. 会員認証を行う.
3.「会員情報」を選択する.
4.「会員情報変更」を選択する.
5. 変更する情報を入力する.
6. システムは変更を受付け，変更後の会員情報を表示する.

Extensions
3. 認証に失敗した.
  a. 失敗.

ユースケース記述：パスワードを変更する

Name パスワードを変更する
Initiator 会員
Goal 会員のパスワードを変更する.

Main Success Scenario
1. 情報家電サービスに接続し，「トップメニュー」を選択する.
2. 会員認証を行う.
3.「会員情報」を選択する.
4.「パスワード変更」を選択する.
5. 変更するパスワードを入力する.
6. システムは変更を受付け，パスワード変更メッセージを表示する.

Extensions
3. 認証に失敗した.
  a. 失敗.

ユースケース記述：料理情報を取り寄せる

Name 料理情報を取り寄せる
Initiator 会員
Goal 買い物情報やレシピなどの料理情報を閲覧する.

Main Success Scenario
1. 情報家電サービスに接続し,「トップメニュー」を選択する.
2. 会員認証を行う.
3.「お料理情報」を選択する.
4. システムは料理情報を表示する.

Extensions
3. 認証に失敗した.
   a. 失敗.

ユースケース記述：会員認証を行う

Name 会員認証を行う
Initiator Included only
Goal 会員番号とパスワードに基づき会員を認証する.

Main Success Scenario
1. 会員番号を入力する.
2. パスワードを入力する.
3. システムは，会員情報の会員番号とパスワードを照合する.
4. 認証成功.

Extensions
4. 会員番号が存在しない.
   a. 認証に失敗した旨のメッセージを表示する.
   b. 認証失敗.
4. 退会利用者
   a. 既に退会している旨のメッセージを表示する.
   b. 認証失敗.
4. パスワードが一致しない.
   a.認証に失敗した旨のメッセージを表示する.
   b. 認証失敗.

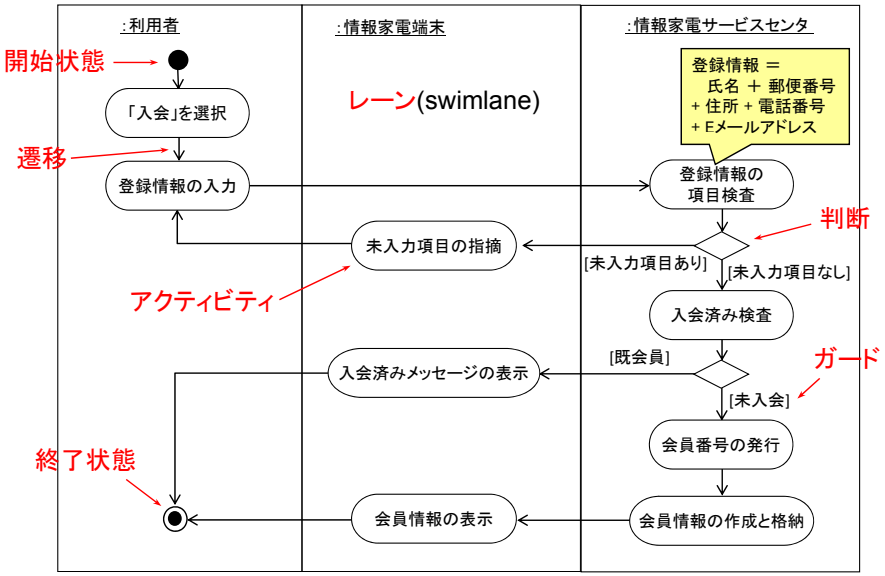
アクティビティ図(activity diagram)

- アクティビティ間の依存性(並行性, 順序) を表現
- 一連の作業を完結するために必要な個々のアクションとその結果を表現
- ワークフローを表現
- イベントによらない自動的な状態遷移だけを記述した特殊な状態図

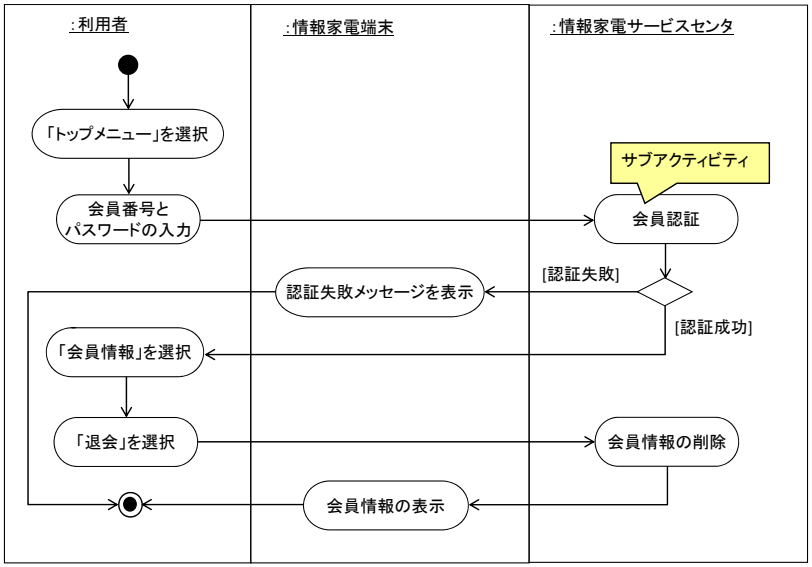
✓ アクティビティ(activity): 何かを実行中の状態

✓ アクション(action): 実行される作業やアクティビティ

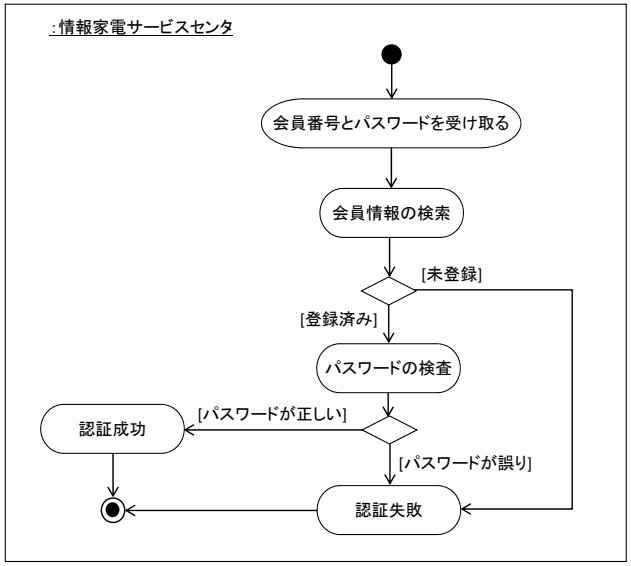
分析モデル：入会する



分析モデル: 退会する

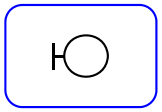


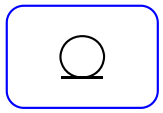
分析モデル: 会員認証を行う



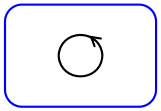
ロバストネス分析(Robustness Analysis)

- 境界オブジェクト <<boundary>>
  - アクタがシステムとの通信に利用
  - インタフェース
  - 画面，ダイアログ，メニューなど
  - インタフェースごとに定義
  - 入出力項目が属性





- 実体オブジェクト <<entity>>
  - ドメインモデルに登場
  - システム内部で半永久的に存在するデータとその振る舞い
  - データベースのテーブルやデータベース設計の入力
  - データ項目が属性

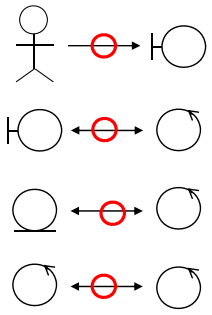


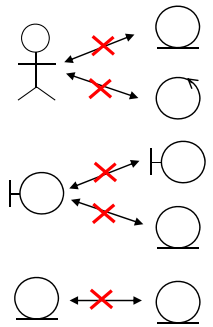
- 制御オブジェクト <<control>>
  - 境界オブジェクトと実体オブジェクトを接続
  - コントローラ
  - ユースケース，ユーザのアクション，操作(ICONIX)に対応
  - 属性なし

ロバストネス図

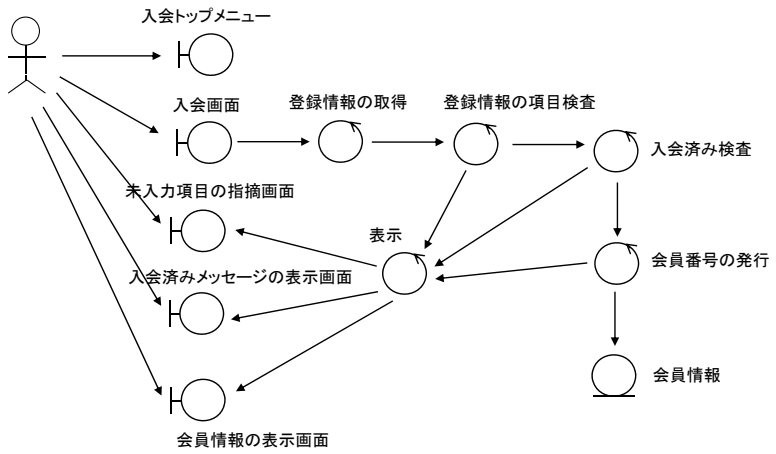
原則:

1. アクタは境界オブジェクトとのみ通信可能
2. 境界オブジェクトは制御オブジェクトとのみ通信可能
3. 実体オブジェクトは制御オブジェクトとのみ通信可能
4. 制御オブジェクトは境界オブジェクト，実体オブジェクト，制御オブジェクトと通信可能，かつ，アクタとは通信不可能

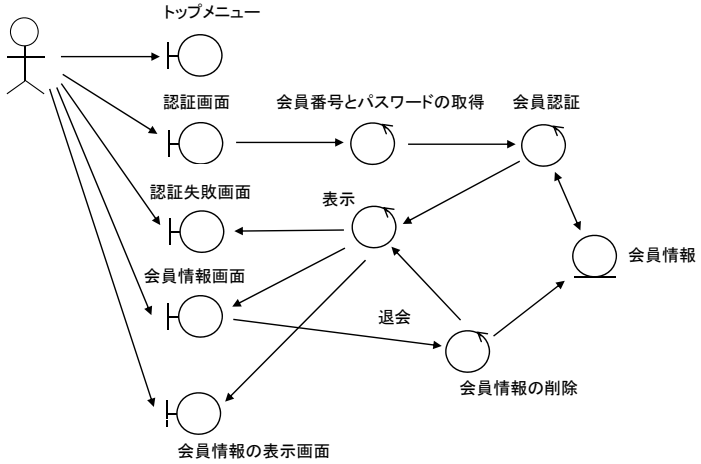




分析モデル: 入会する



分析モデル: 退会する



オブジェクト指向設計

オブジェクトの抽出

人間あるいは装置に関する:
利用者 (未入会者, 会員)
情報家電端末
情報家電サービスセンタ

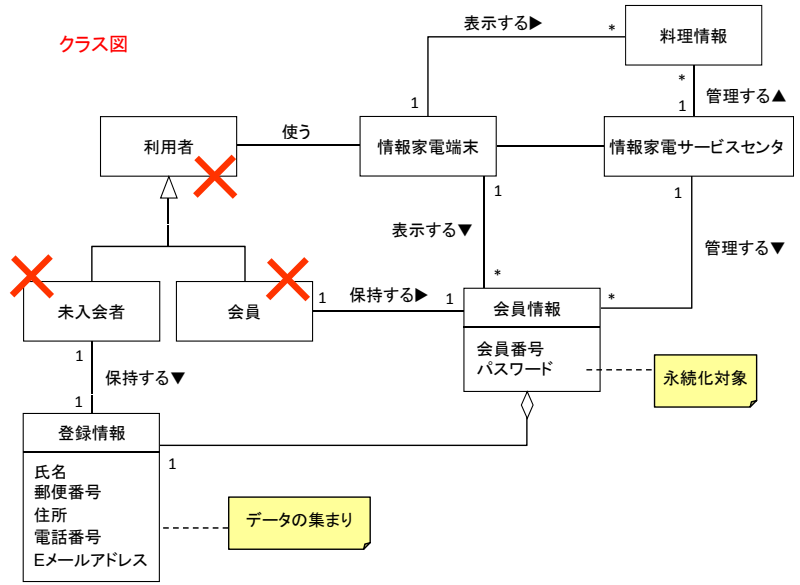
データに関する:
登録情報 = 氏名 + 郵便番号 + 住所 + 電話番号 + Eメールアドレス
未入力項目
会員情報 = 会員番号 + パスワード + 登録情報

画面に関する:
トップメニュー, 入会トップメニュー, 認証画面, ...

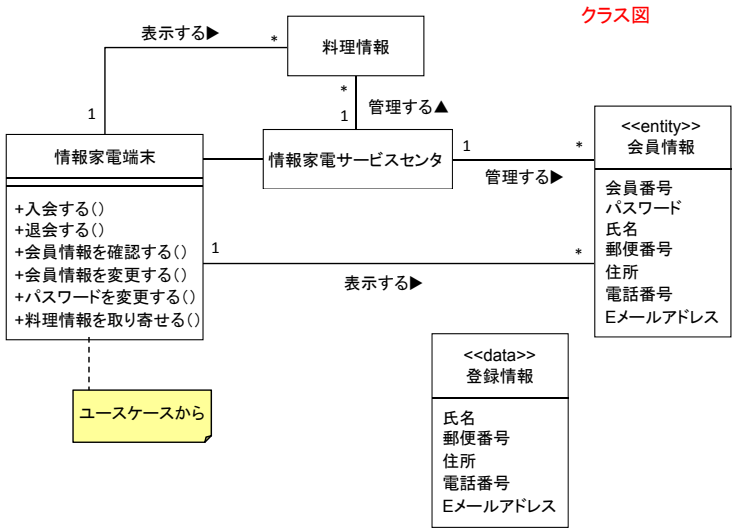
メニュー項目に関する:
「入会」,「退会」,「会員情報」,「会員情報確認」,「会員情報変更」,「パスワード変更」,
「お料理情報」,「お掃除情報」,「家電店情報」, ...

メッセージに関する:
入会済みメッセージ, 認証失敗メッセージ, パスワード変更メッセージ,
退会している旨のメッセージ, ...

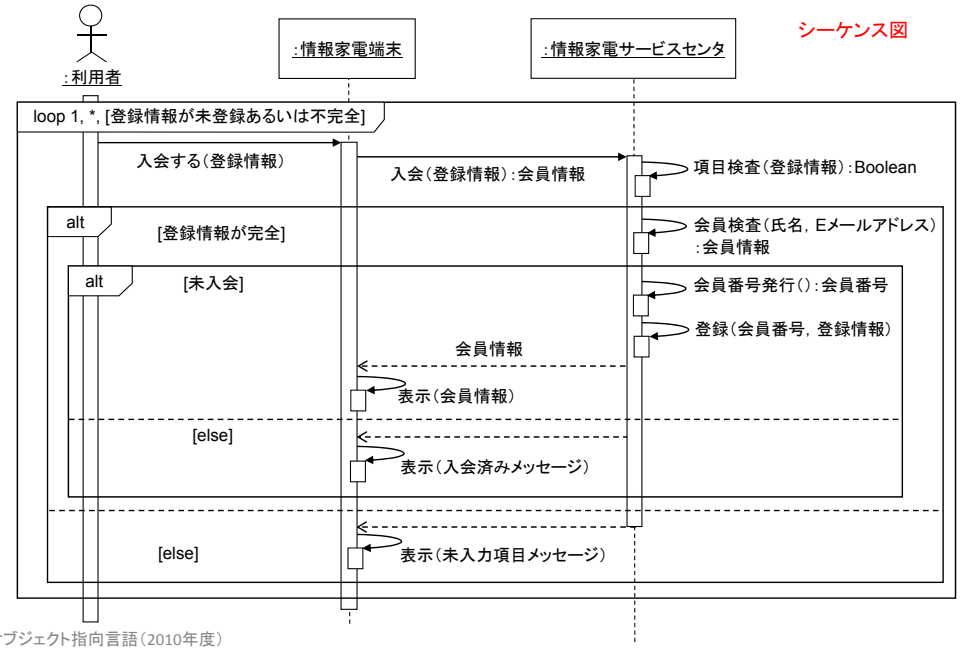
分析モデル: 概念モデル → 分析モデル



分析モデル: クラス図の洗練



分析モデル: 振る舞いの定義(1)



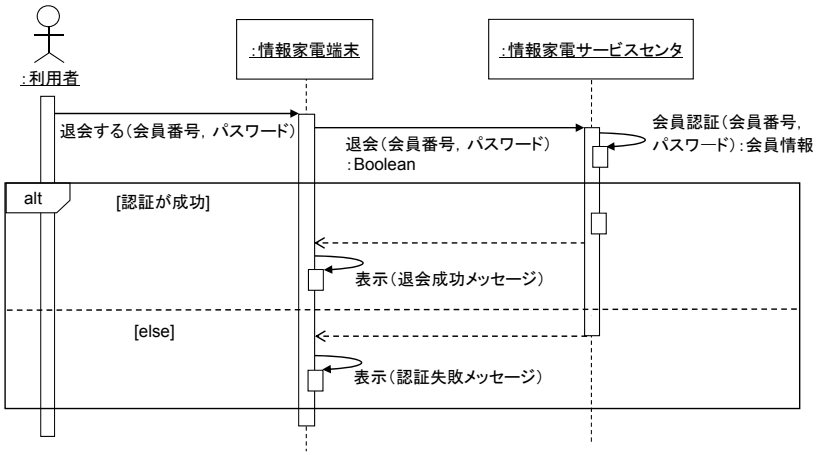
分析モデル: 機能の定義

情報家電サービスセンタの機能

Name	入会()
Informal Description	会員を新規に登録する.
Constraints	—
Receives	登録情報
Returns	入会成功:会員情報 入会済み:null 未入力項目あり: 会員情報において該当項目がnull
Changes	情報家電サービスセンタ
Assumes	—
Result	入会済みでないとき, 唯一の会員番号を持つ新しい会員情報が作成される. 会員の会員番号, パスワード, 登録情報が会員情報に格納される. 会員情報が返却される.

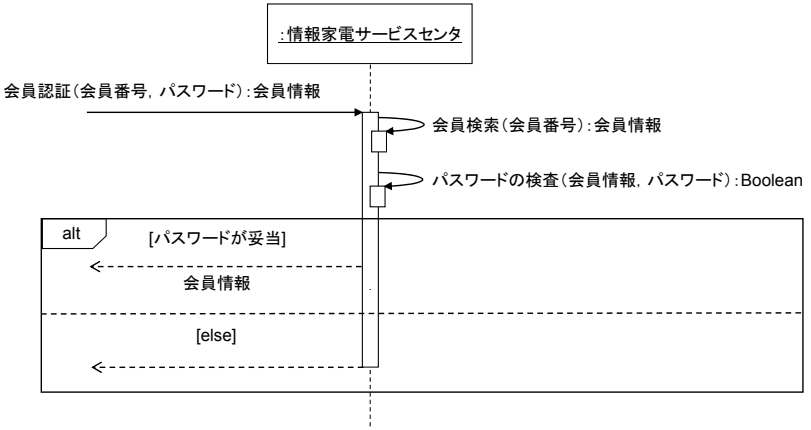
分析モデル：振る舞いの定義(2)

シーケンス図



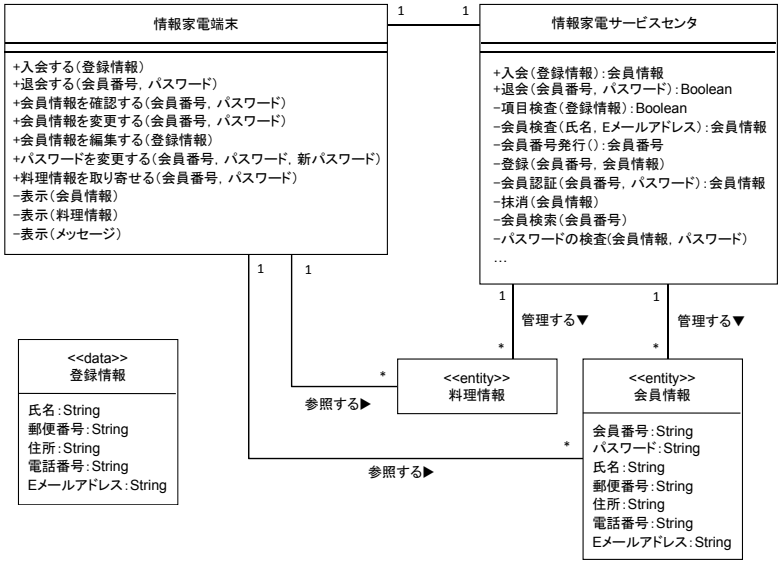
分析モデル：振る舞いの定義(3)

シーケンス図



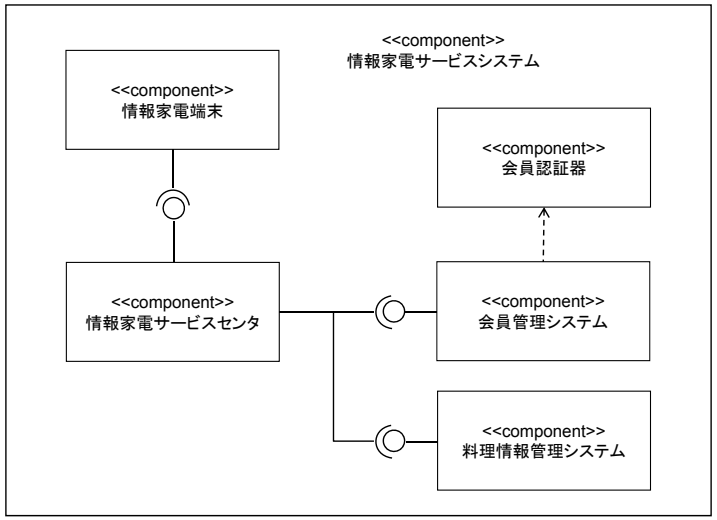
分析モデル：操作の追加

クラス図



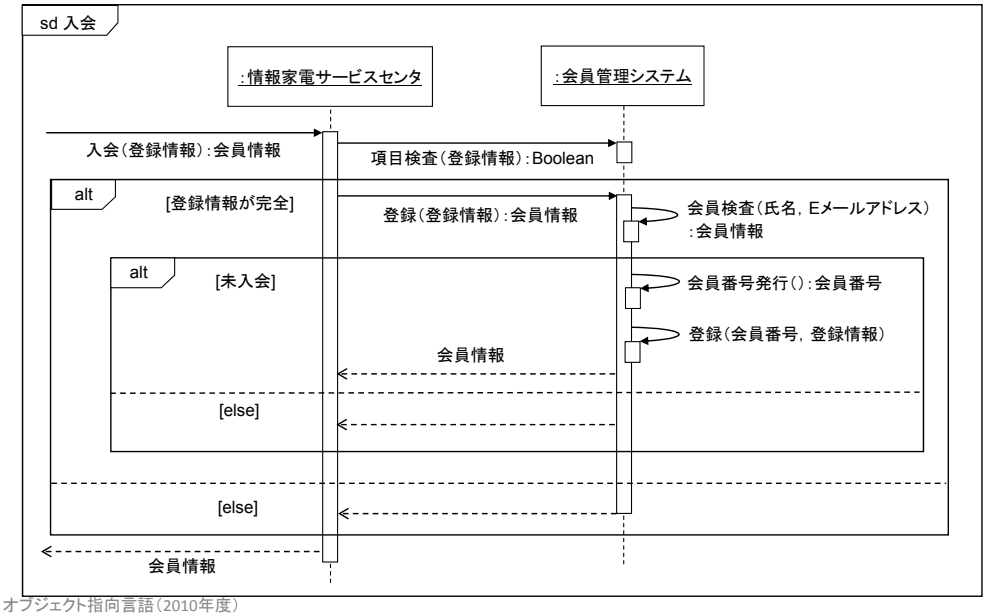
分析モデル：アーキテクチャの定義

コンポーネント図



分析モデル：振る舞いの洗練(1)

シーケンス図



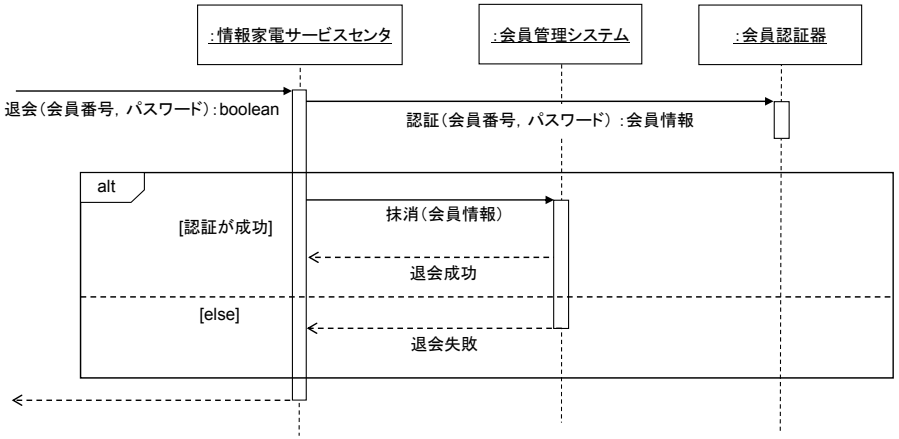
分析モデル：機能の洗練

会員管理システムの機能

Name	登録()
Informal Description	会員情報を会員情報リストに登録する.
Constraints	会員数はIntegerの範囲を超えない.
Receives	会員番号:String 登録情報
Returns	登録成功:会員情報
Changes	会員管理器
Assumes	会員情報コレクションが存在する. 登録情報が規定フォーマットに準じている.
Result	唯一の会員番号を持つ新しい会員情報が作成される. 氏名, 郵便番号, 住所, 電話番号, Eメールアドレスが会員情報に格納される. 作成した会員情報が会員情報コレクションに追加される. 退会フラグがfalseに設定される. 会員数が1つ増加する.

分析モデル：振る舞いの洗練(2)

シーケンス図



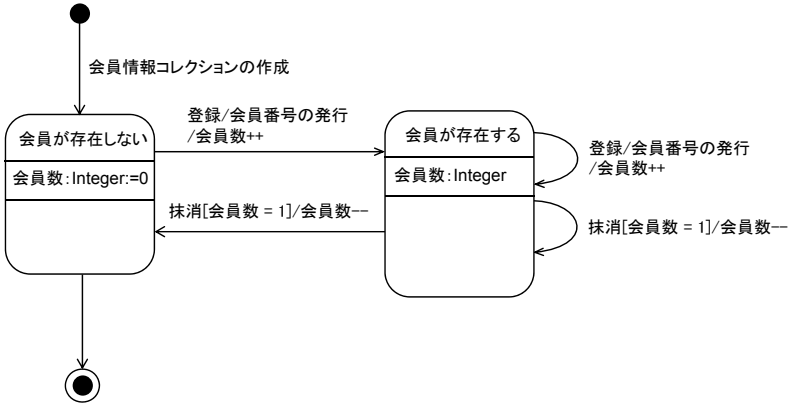
分析モデル：機能の洗練(2)

会員管理システムの機能

Name	抹消()
Informal Description	与えられた会員情報の退会フラグをセットする.
Constraints	—
Receives	会員情報
Returns	—
Changes	与えられた会員情報
Assumes	会員情報が有効である. 会員情報の退会フラグがfalseである.
Result	会員情報の退会フラグがtrueとなる. 退会フラグ以外の情報はそのまま保存される. 会員数が1つ減少する.

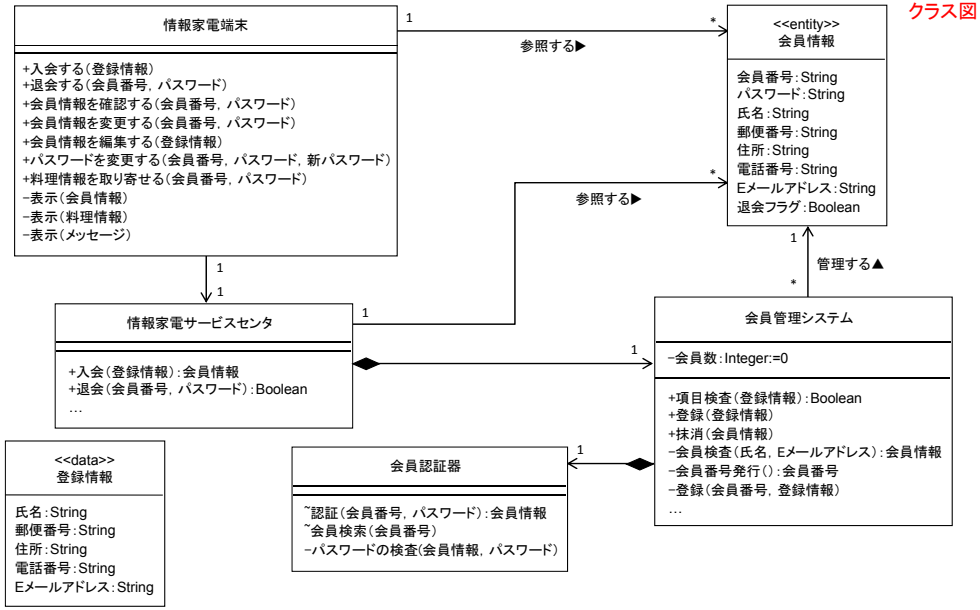
分析モデル：状態の定義

状態図



分析モデル：クラス図の洗練

クラス図

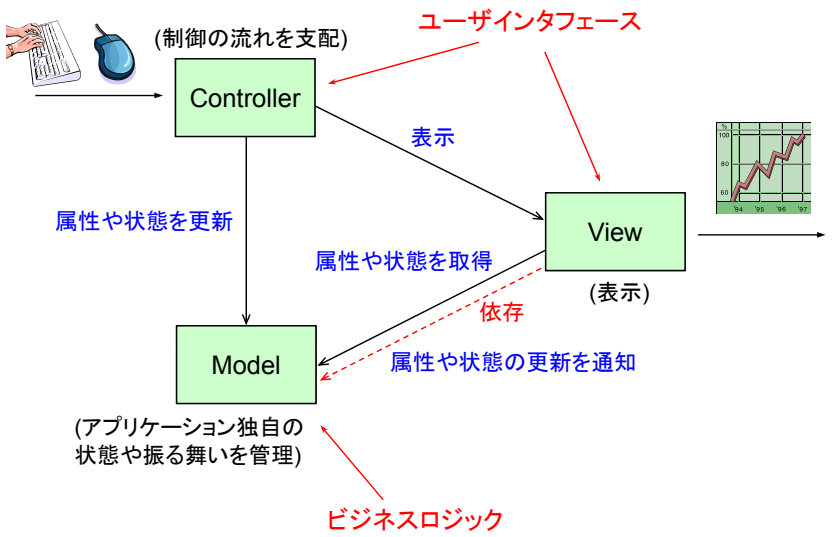


アーキテクチャの比較

役割 / アークテクチャ	ユーザインタフェース層	ビジネスロジック層	データアクセス層
MVC	View Controller	Model	
ロバストネス分析 (Jacobson)	Boundary	Control	Entity
EAアークテクチャ (Fowler)	Presentation	Domain	Data Source

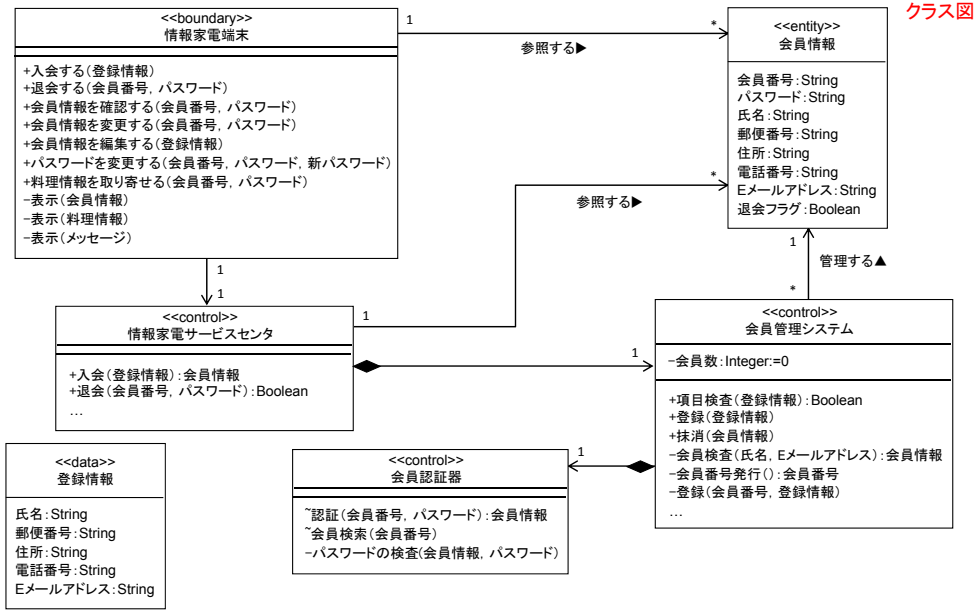
EAアークテクチャ：Enterprise Application Architecture

MVCアーキテクチャ



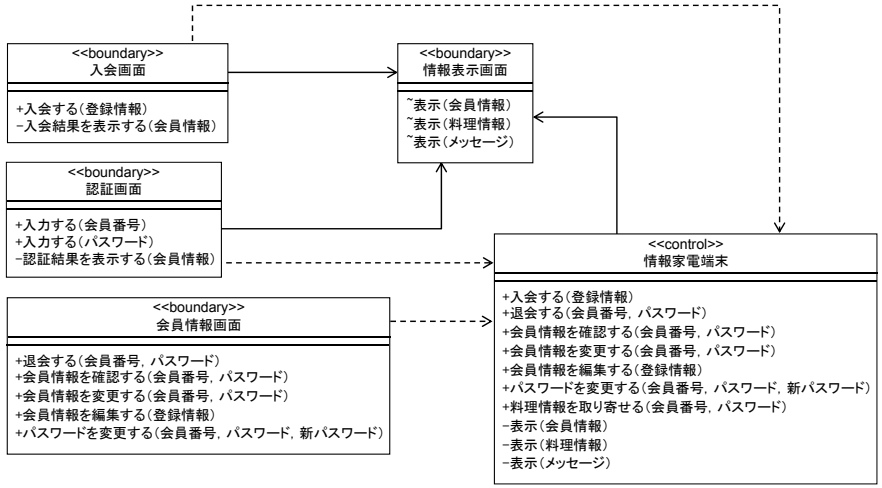
設計モデル

クラス図



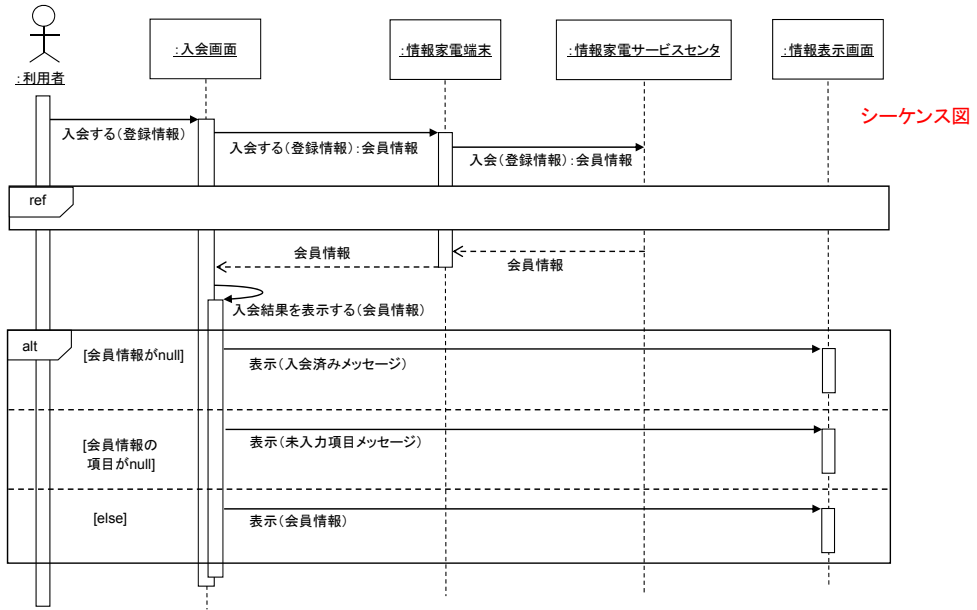
設計モデル: ユーザインタフェース層の洗練

クラス図



設計モデル: 振る舞いの定義

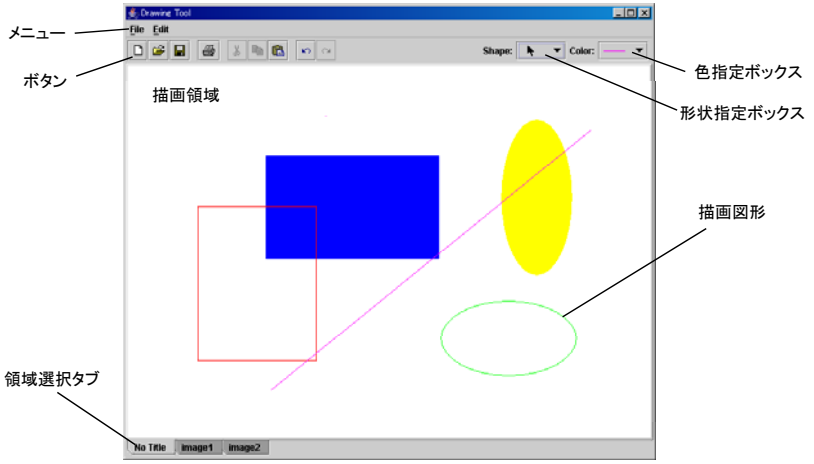
シーケンス図



例題: 図形描画ツールの開発

要求文書(1)

(a) 図形描画ツールは，領域選択タブで切り替えられる複数の描画領域（キャンバス）を持つ．
利用者は，マウスを用いて，描画領域に図形を描くことができる．
描画する図形の形状は，形状指定ボックスで行う．また，図形の色は，色指定ボックスで行う．
利用者は，描画領域および図形領域に対して，メニューおよびボタンにより特定の操作を行うことができる．

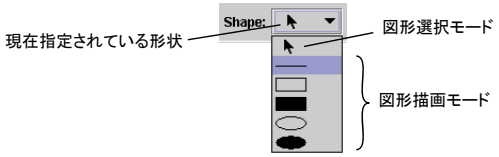


要求文書(2)

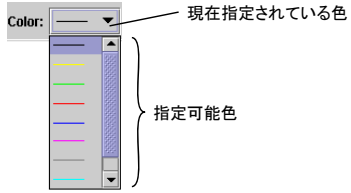
(b) 利用者は，描画領域上の図形に対して，以下の操作を実行できる．

描画 … マウスの左クリックで図形の始点を指定し，ドラッグ & ドロップで図形の終点を指定する．描画中の図形はラバーバンドで表現される．
選択 … 形状指定ボックスにおいて図形選択モードが指定されているとき，描画図形上でマウスの左クリックにより，図形を選択できる．図形の外部で，左クリックすることで，選択を解除できる．
移動 … 図形の選択中に，図形上でマウスの左クリックを行うことで，図形をつまむことができ，ドラッグ&ドロップで図形を移動させることができる．

(c) 形状指定ボックス：図形選択/描画モードの切り替えと，描画する図形(線，長方形，楕円など)を指定する．



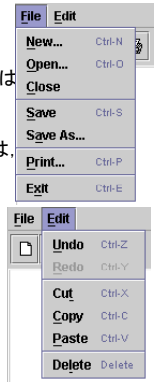
(d) 色指定ボックス：描画する図形の色(黒，黄，緑，赤など)を指定する．



要求文書(3)

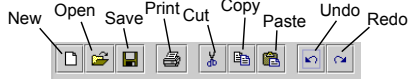
(e) 利用者は，描画領域および図形領域に対して，メニューおよびボタンにより以下の操作を行うことができる．

1. Fileメニュー：
New … メインウインドウにおいて，新規に描画領域を作成する．
Open … ファイルから描画領域（図形）を読み込む．
Close … 最前面の描画領域を閉じる．描画領域に変更が存在する場合，変更の保存あるいは破棄の確認が行われる．
Save … 最前面の描画領域（図形）を保存する．
SaveAs … 最前面の描画領域（図形）を別のファイル名を付けて保存する．もとの描画領域は，メインウインドウから削除され，新しい名前の描画領域が最前面に呼び出される．
Print … 最前面の描画領域をプリンタに印刷する．
Exit … 図形描画ツールを終了する．変更を含む描画領域に対しては，変更の保存あるいは破棄の確認が行われる．



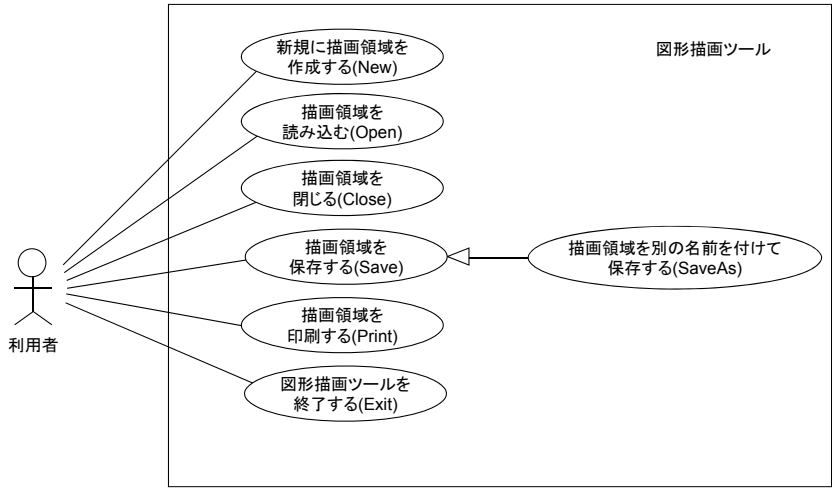
2. Editメニュー：
~~Undo … 最前面の描画領域に対して，直前の操作を元に戻す．~~
~~Redo … 最前面の描画領域に対して，直前の取り消しをやり直す．~~
Cut … 選択した図形を切り取り(削除し)，その図形をクリップボードに格納する．
Copy … 選択した図形をクリップボードにコピーする．図形の削除は行わない．
Paste … クリップボードに格納されている図形をもとの図形の右下に挿入する．
Delete … 選択した図形を削除する．クリップボードに格納されている図形は更新されない．

~~(f) ボタン：以下に示すボタンを押すことで，Fileメニューおよび Editメニューにおける一部アクションを実行できる．~~

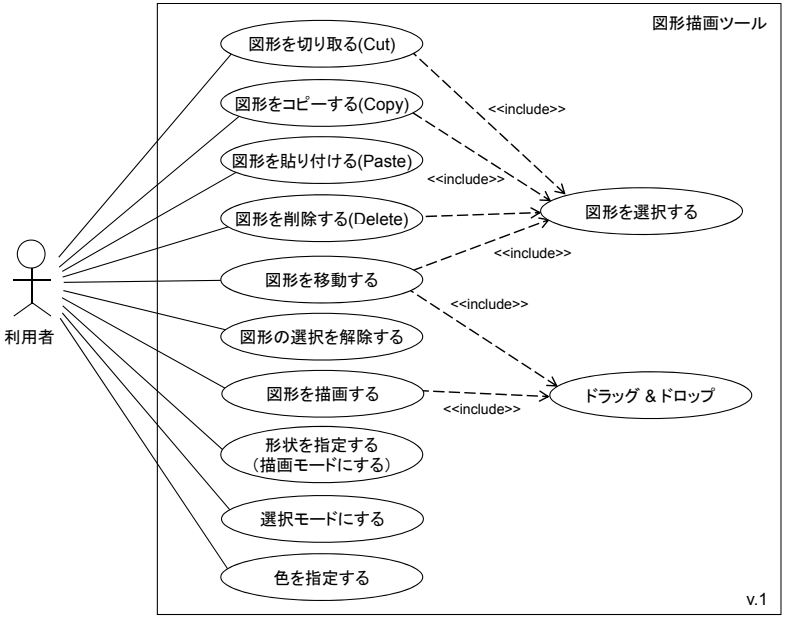


(g) 領域選択タブ：タブを選択することで，その描画領域を最前面に呼び出すことができる．

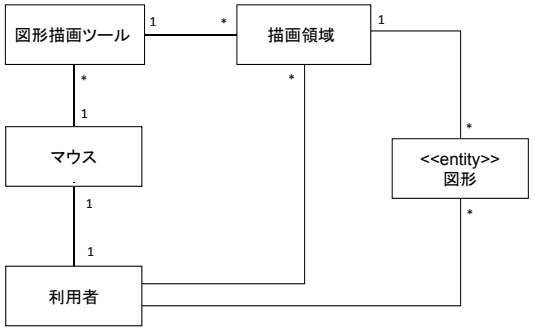
要求モデル：ユースケース図



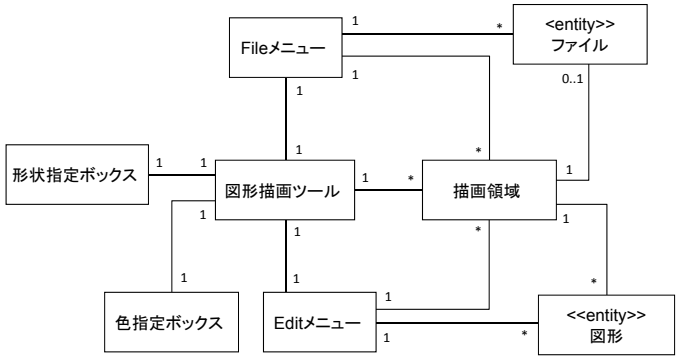
要求モデル: ユースケース図 (cont'd)



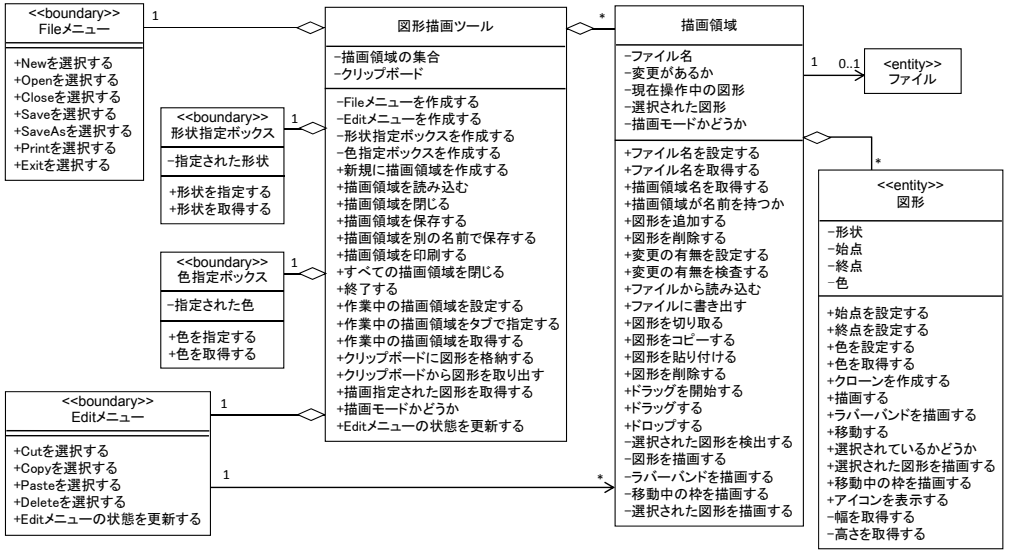
概念モデル



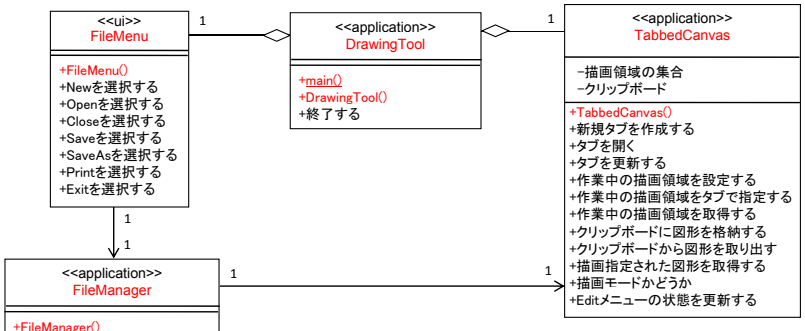
分析モデル



設計モデル

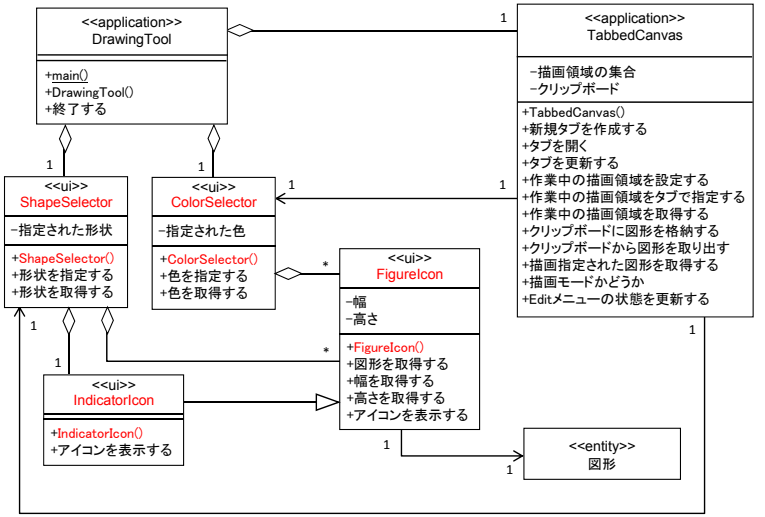


詳細設計モデル(1/4)



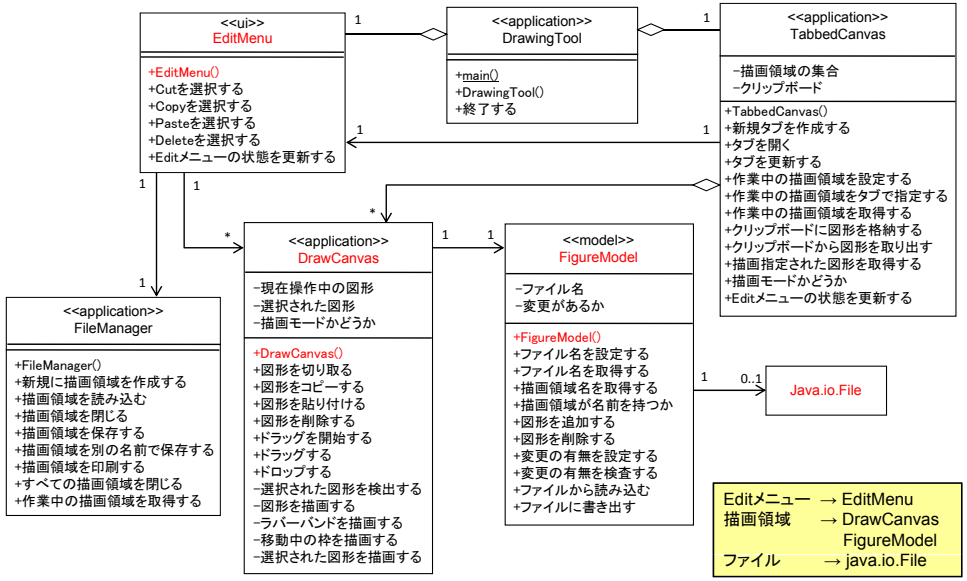
Fileメニュー → FileMenu
図形描画ツール → DrawingTool, FileManager, PrintManager, TabbedCanvas

詳細設計モデル(2/4)



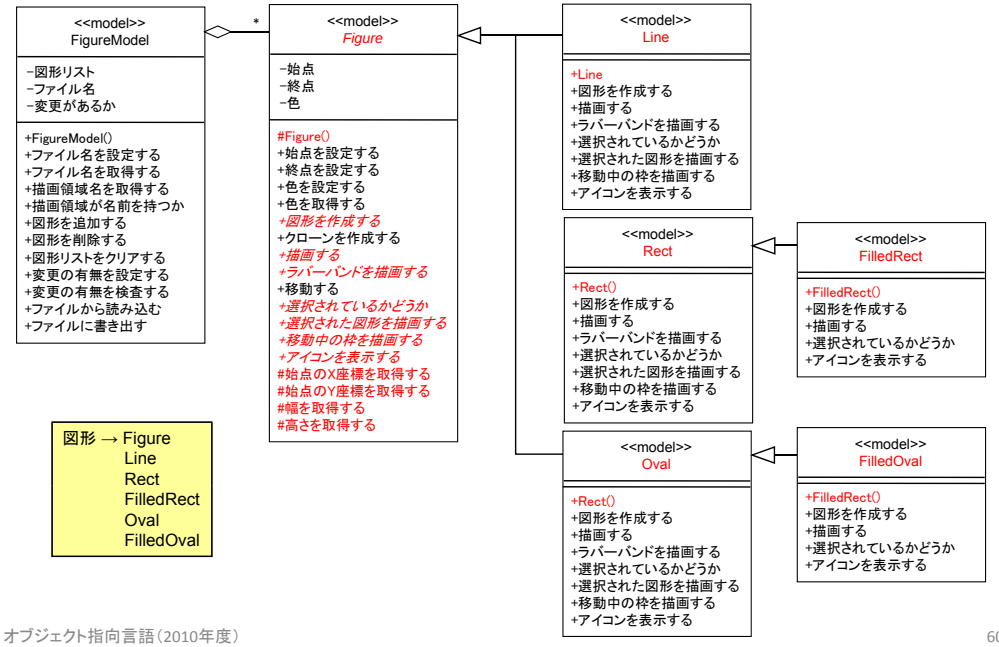
形状指定ボックス → ShapeSelector, FigureIcon
色指定ボックス → ColorSelector, FigureIcon, IndicatorIcon

詳細設計モデル(3/4)



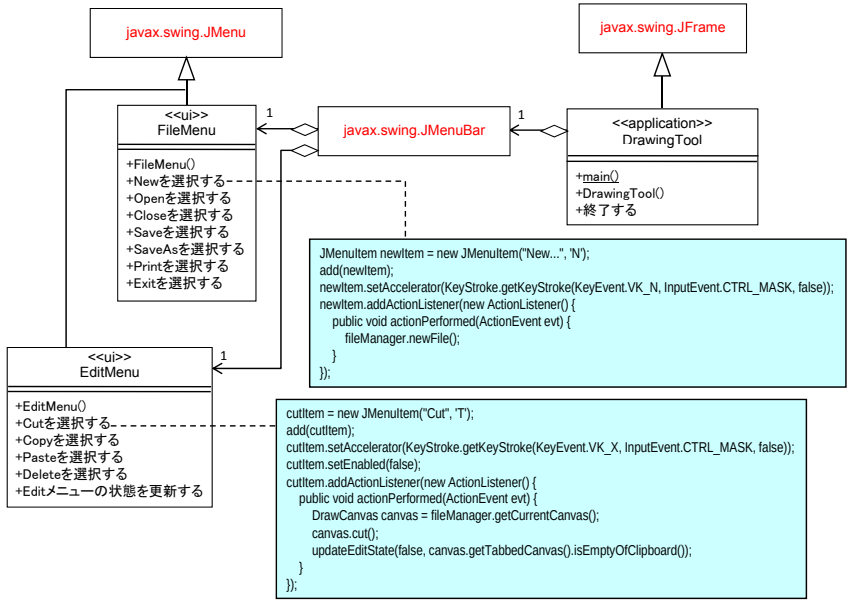
Editメニュー → EditMenu
描画領域 → DrawCanvas
FigureModel
ファイル → java.io.File

詳細設計モデル(4/4)

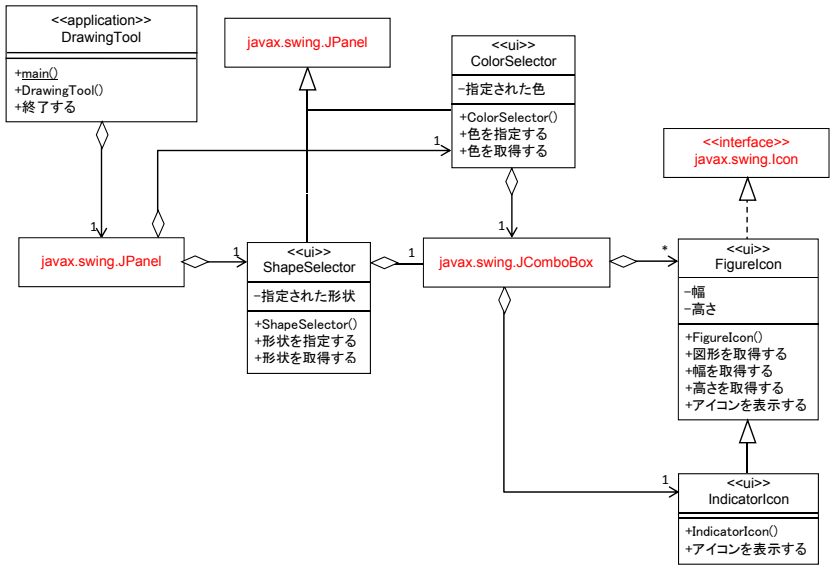


図形 → Figure
Line
Rect
FilledRect
Oval
FilledOval

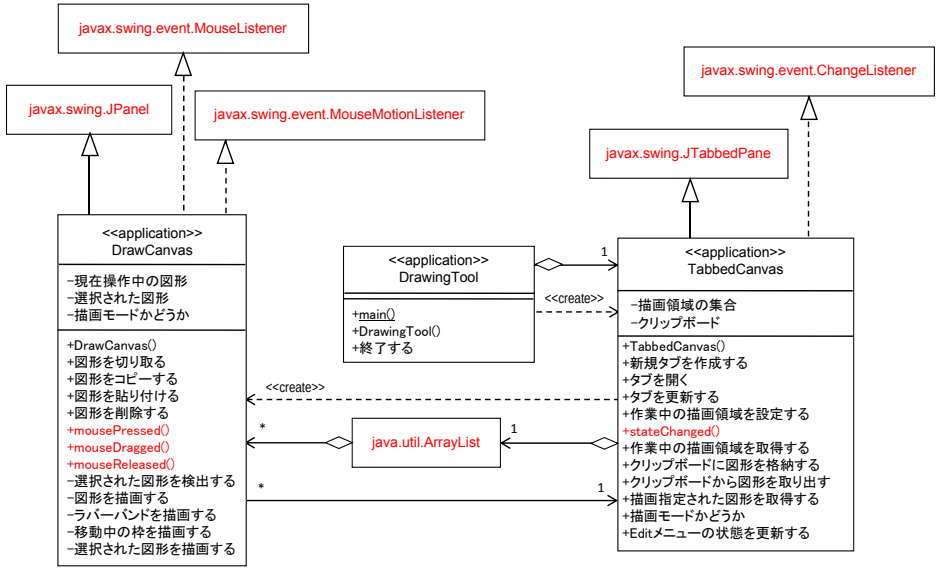
詳細設計モデル(Java: 1/4)



詳細設計モデル(Java: 2/4)



詳細設計モデル(Java: 3/4)



詳細設計モデル(Java: 4/4)