ソフトウェア工学

06：UMLモデリング（Ｉ）－
ユースケースモデリングとユースケース駆動型開発

理工学部 経営システム工学科
庄司　裕子

前回の復習：考えてみよう！

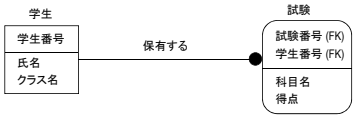
- 個人成績表に、学生番号、氏名、クラス名という個人情報と、試験番号、科目名、成績（得点）という成績情報が記載されているとする
- これをERモデリングして、ER図を書いてみよう

ヒント：

- 「学生」、「クラス」、「試験」という独立エンティティ（「もの」を表す）と、「成績」、「所属」という依存エンティティ（関係を表す）を用意する

2

成績表をそのまま表現したERD

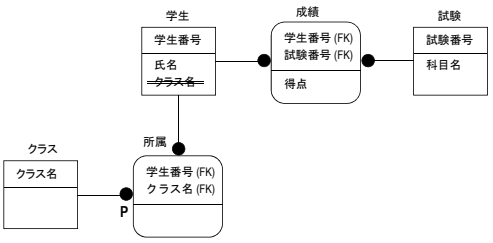


3

「成績」エンティティを導入したERD

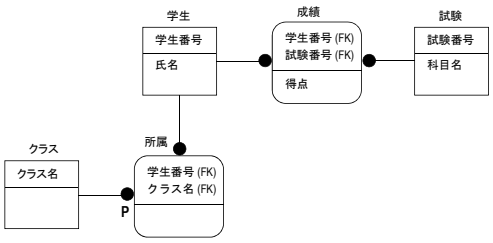
4

「所属」エンティティを導入したERD



5

最終的なERD



6

今回のテーマ

UMLモデリング（Ｉ）
– ユースケースモデリングとユースケース駆動型開発
- 開発プロセスにおける位置づけと目的
- ERモデリング
- オブジェクト指向分析／設計（OOAD）
- UML

7

UMLモデリング（Ｉ）– ユースケースモデリングとユースケース駆動型開発

⇨UMLとUMLダイアグラム
- ユースケース モデリングの基礎
- ユースケース モデリングの役割
- ユースケース関連要素の関係
- ユースケース駆動型開発

8

UML（Unified Modelling Language）とは

- 1990年代に乱立していた主なOOAD方法論の概念と表記法を統一したもの
  - OMT法（James Rumbaugh）
  - Booch法（Grady Booch）
  - OOSE/Objectory法（Ivar Jacobson）
  - その他

☞ソフトウェア システムの成果物を仕様化、視覚化、作成、文書化するための汎用のビジュアル モデリング言語

9

UMLダイアグラム

- システムの静的（構造的）側面
  - クラス図
  - オブジェクト図
  - パッケージ図
  - コンポーネント図
  - 配置図
- システムの動的側面
  - ⇨**ユースケース図**
  - シーケンス図
  - コラボレーション図
  - アクティビティ図
  - ステートチャート図

10

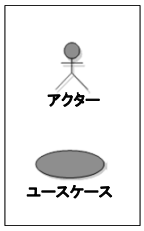
ユースケース モデルとは

- システムに求められる機能（ユースケース）とその外部環境（アクター）との相互作用をアクターの視点で表すモデル
- 要求分析／定義、システム分析／設計、およびテストには同じユースケース モデルが一貫して使用される

☞ユースケース モデルをビジュアルに表現したものがユースケース図

11

ユースケース モデルの主要概念

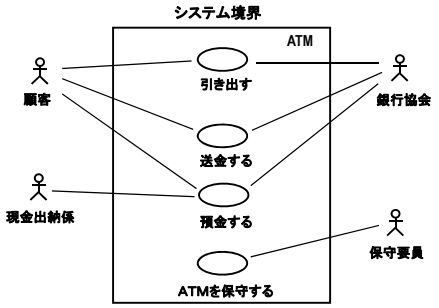


UMLでの表記法

- **アクター**は、システムとやり取りする外部の実体（人間とは限らない）
- **ユースケース**は、システムによって実行されるアクションのシーケンスを定義し、結果としてそれと認められるような価値ある結果をアクターにもたらすもの

12

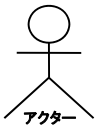
ユースケースモデルの例: ATM



13

アクターは…

- **システムの一部ではない**(**境界**を定義)
- システムのユーザが果たすことのできる**役割**を表す
- **人間**、**機械**、または**別のシステム**を表すことができる
- システムと**能動的に情報を交換**する場合がある
- **情報を提供する**場合がある
- **情報を受動的に受け取る**場合がある



14

アクターの発見に有効な質問

- 情報の提供者、利用者、削除者は誰か?
- この機能を使うのは誰か?
- どの要求に誰が関心を持つか?
- システムが組織内のどこで使われるか?
- システムのサポートと保守は誰が行うか?
- システムの外部リソースには何があるか?
- 他のシステムがこのシステムとやり取りするのに何が必要か?

15

ユースケースは…

- システムの個別の利用方法について、その通常動作とそれに関連する可能なさまざまな使用パスを1つにまとめて記述したもの(ユースケース クラス)
  - ユースケース クラスに属する個々の利用方法(ユースケース インスタンス)は**シナリオ**と呼ばれる
  - ユースケースの識別と記述は、インスタンスではなくクラスのレベルで行う
- 完全に実行されたり、まったく実行されなかったりすることがある
- アクターにより開始される

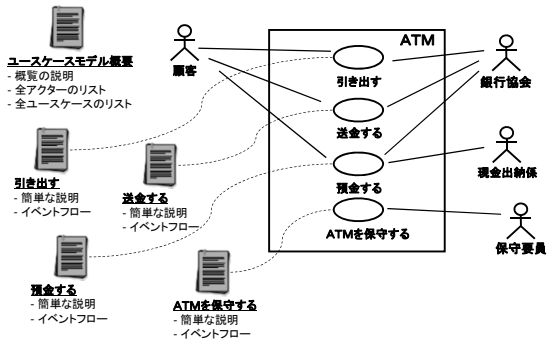
16

ユースケースの発見に有効な質問

- アクターがシステムで実行する主なタスクは?
- アクターはシステム内のデータの作成、保存、変更、削除、読みとりを行うか?
- アクターは急な外部的変更をシステムに通知する必要があるか?
- システム内の特定の出来事をアクターに通知する必要があるか?
- アクターはシステムの起動と停止を実行するか?

17

ユースケースはテキスト中心



18

ユースケース内容の記述

- ユースケース モデルはユースケース図だけではない
- 個々のユースケースについて記述したドキュメントがなければ、ユースケース モデルは実効性がない
- UMLには、ユースケース定義ドキュメントの形式は規定されていない
  - ⇨顧客およびユーザに理解してもらえる記述形式が必須

19

ユースケース定義のテンプレート（ラショナル統一プロセスでの例）

<ユースケース名>

1. 概要
ユースケースの役割

2. イベントフロー
ユースケースの振る舞い（基本フロー＆代替フロー）

3. 関係
・ユースケースが関係する、アクターのコミュニケーション関連、
・ユースケース同士の包含関係、拡張関係

4. 特殊要件
このユースケースに関係する機能外要求の集合

添付
・ダイアグラム
・事前条件の説明（オプション）
・事後条件の説明（オプション）
・(グラフィカル) ユーザインターフェイスの図

ユースケース定義ドキュメントでは、ユースケース モデル内の個々のユースケースに関する情報を記述する。

20

ユースケース定義ドキュメントに記載すべき内容

- 概要
  - そのユースケースの目的を2、3文で記述
- ☞イベントフロー（基本フロー＆代替フロー）
  - そのユースケースがシステム内で何を行うか
  - いつどのように開始し終了するか
  - いつアクターと相互作用するか、どのような情報が交換されるか
- 関係
  - このユースケースが他のユースケースに対して備えている関連をすべて列挙し、必要であればそれらの概要も付ける

21

ユースケース定義ドキュメントに記載すべき内容（続き）

- 特殊要件
  - イベントフローでは扱われないが、設計に影響を及ぼし得る要求（応答時間などのパフォーマンス特性、他）
- 補足資料
  - ユースケース図
    - このユースケースに含まれるすべての関係を示した図
  - 事前条件（オプション）
    - ユースケース開始時にシステムが満たすべき条件を説明した文
  - 事後条件（オプション）
    - ユースケース終了時にシステムが満たすべき条件を説明した文
  - （グラフィカル）ユーザ インタフェースの図
    - ユースケースを明確にする手書きのスケッチ
    - ユーザ インタフェースのプロトタイプ画面のハードコピー

22

ユースケースの記述での留意点

- 平易な言葉を使う
- 用語集を用意して、同じ用語を一貫して使用する
- システムが行うこととユーザが行うことを明確に区別する
- 各セクションには番号と名前を付けて、レビューを容易にする
- ☞記述は、コンピュータのためのものではなく、人間が読むためのものである

23

ユースケース モデリングの役割

- 要求獲得（要求分析／定義）
  - システムの機能と振る舞いについての要求を顧客またはエンド ユーザから獲得するための構造化された方法を提供する
- 開発の反復計画（反復型開発）
  - プロジェクト全体の見積りと開発プロセスの立案の根拠を与える
- システムの妥当性検証
  - システムの設計の妥当性を検証する手法やシステムのテスト手法として使える

24

要求獲得にどのように役立つか？

- システムの機能について顧客の同意を得やすい
- システムと相互作用するのが誰であるかを発見できる
- システムに備えるべきインタフェースを発見できる
- 要求が欠落していないかどうか確認できる
- 開発者が要求を正しく理解しているかどうか確認できる

25

ユースケース モデリングの落とし穴

- オブジェクト指向に反したシステムを作ってしまう危険性がある
  - ユースケースに注意を払いすぎる余り、システム アーキテクチャと静的オブジェクト構造を見失うかもしれない
  - ⇨反復の注意深い管理で回避可能
- 設計を要求と取り違えてしまう危険性がある
- 要求を見落とす危険性がある
  - まずアクターを見つけ、次にそれらのアクターに必要なユースケースを見つけるというプロセスで、すべての要求が明らかになるわけではない

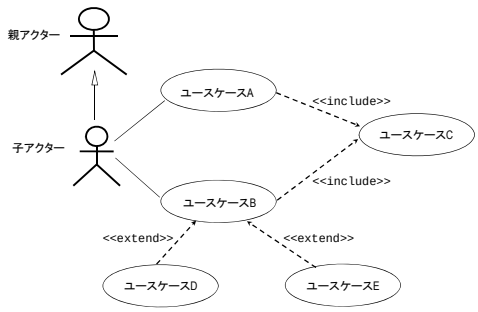
26

ユースケース間／アクター間の関係

- ユースケース間の関係
  - **包含**：複数のユースケースに共通した振る舞いを、独立したユースケースとして分離し、他のユースケースに組み込む
  - **拡張**：あるユースケースに大きく異なるシナリオが含まれている場合、それらを1つの主ユースケースと従属ユースケースに分け、適宜振る舞いを切り替える
- アクター間の関係
  - **汎化**：複数のアクターを1つのアクターに抽象化

27

ユースケース間／アクター間の関係のUMLによる表記法



28

ユースケース駆動型開発の思想

- ユースケースを開発プロセスの最も重要な側面として重視する
- 要求獲得が済んで設計に入ったら捨てられるのではなく、変更の追跡や反復の定義などを目的として、プロジェクト全体を通して使用される

☞ユーザ／顧客の要求から焦点を逸らさないようにするのに役立つ

29

まとめ：ユースケースモデリングとユースケース駆動型開発

- ユースケースモデリング
  - システムの機能を、外部の人やシステムとの相互作用に着目してモデリング
- ユースケース： システムの果たすべき機能
- アクター：システムと関わる外部の人やもの
- ユースケース駆動型開発
  - ユースケースを最重要視
  - アクターやユースケースを上手に発見・定義することが重要

30