



ソフトウェア設計法及び演習 ソフトウェア工学概論及び演習

大山 勝徳
日本大学 工学部



復習

■ オブジェクト指向

- オブジェクト指向開発
- オブジェクト
 - データ属性, メソッド
 - クラス, インスタンス
 - カプセル化と情報隠蔽
- オブジェクト間の関連



■ オブジェクト指向

- オブジェクト間の関連
- オブジェクト指向によるシステム分析

■ UML

- ユースケース
- クラス図

■ 演習

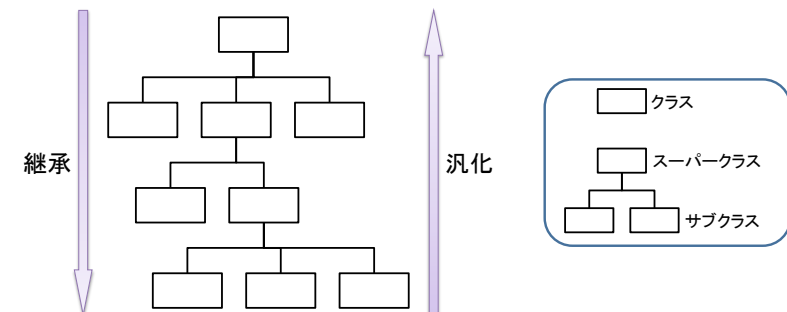


クラスの階層化と継承

クラスは階層構造をとる

■ 継承 (inheritance)

■ 汎化 (generalization)



クラスの階層化（再掲）

■ クラスの階層化のタイプ

□ INSTANCE-OF

- すべてのインスタンスは共通のデータ属性を持つ

□ IS-A

- 複数のクラス間で共通する特性を抜き出し、一般化したクラスを定義する（汎化）。
 - 汎化によりスーパー/サブクラスの関係が生じる

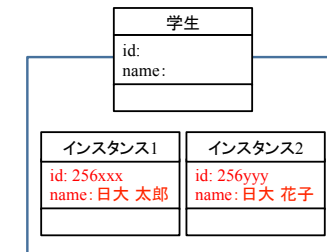
□ PART-OF

- 複数のクラスの集約により、別のクラスを構成するクラスを定義する（集約化）

INSTANCE-OF

■ すべてのインスタンスは共通のクラスのデータ属性を持つ

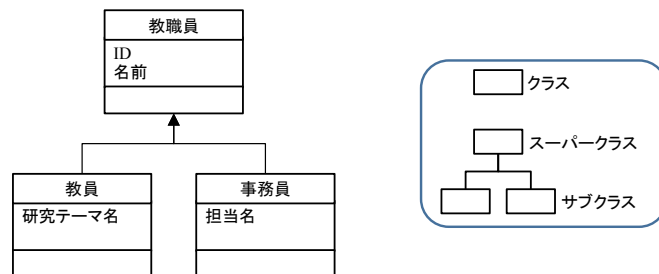
- データおよび操作に関して、共通の性質を持つインスタンスを集め、1つのクラスを定義する



IS-A

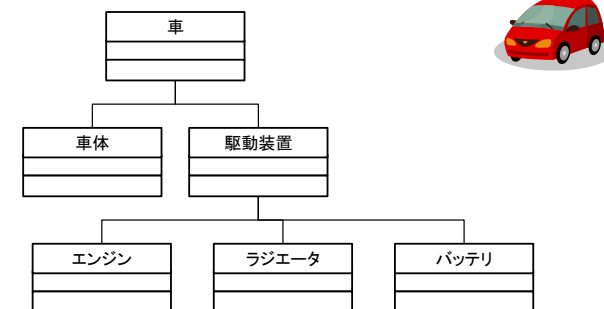
■ 複数のクラス間で共通する特性を抜き出し、一般化したクラスを定義する（汎化）。

- 汎化によりスーパー/サブクラスの関係が生じる



PART-OF

■ 複数のクラスの集約により、別のクラスを構成するクラスを定義する（集約化）



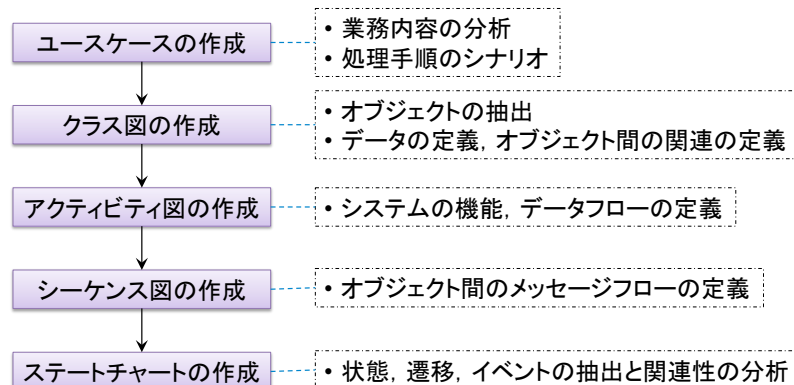
オブジェクト指向の要約

概念/用語	説明
オブジェクト(インスタンス)	データと操作を一体化させた実体
クラス	同じ特性を持つインスタンスをグループ化し、定義したもの
カプセル化	オブジェクトのデータと操作の一体化
メッセージ	メソッドの呼び出し
メソッド	オブジェクト・データを操作する手続き
継承(インヘリタンス)	スーパークラスの特性をサブクラスに引き継ぐ
多相化(ポリモーフィズム)	同じメッセージに対し、クラスの種類に応じた固有の応答を返すこと

 p.91

オブジェクト指向によるシステム分析

- 開発工程は(基本的に)ウォーターフォール
 - 分析→設計→実装→テスト



■ オブジェクト指向

- オブジェクト間の関連
- オブジェクト指向によるシステム分析

■ UML

- ユースケース
- クラス図

■ 演習

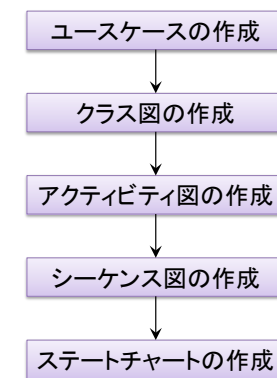
オブジェクト指向によるシステム分析


■ 静的側面

- ユースケース
- クラス図
- アクティビティ図

■ 動的側面

- シーケンス図
- ステートチャート



 モデルの表記法としてUMLを用いる

従来の手法の問題点

■ ウォーターフォールモデル

□ 問題点

- 作業区分が明確に決められている
 - 共同作業が困難
 - 工程間の情報共有が難しい
- 工程の途中で発見された問題への対処が難しい
 - 開発をやり直すと手戻りが大きい
 - 問題に対処せずに次の工程に進むと、正しく開発できない可能性がある



■ オブジェクト指向

- オブジェクト間の関連
- オブジェクト指向によるシステム分析

■ UML

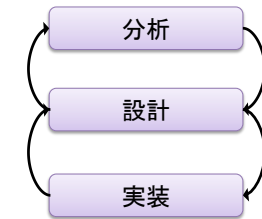
- ユースケース
- クラス図


■ 演習

【発展】設計開発フェーズ

■ 設計開発を次のフェーズに分割し、各フェーズで分析・設計・実装を行なう

- 分析
- 設計
 - システム設計
 - オブジェクト設計
- 実装



 古典的ですが、
OMT法を参考としています
(OMT: Object Modeling Technique)

UML

■ UML (Unified Modeling Language)

- オブジェクト指向モデリングの仕様記述言語
 - 図示による記述
 - 状況に応じて、様々な図(ダイアグラム)で記述する。
クラス図, ユースケース図, シーケンス図, ...
- 汎用のモデリング言語
 - ソフトウェア開発の設計・開発に適用可能
- OMG (Object Management Group)が管理
 - OMG: IT関連の標準を開発する非営利団体

UMLの背景

■ 従来の設計方法論

- 開発フェーズ毎に異なるモデルを使用
 - フェーズ間での関連の追跡が困難



■ オブジェクト指向の設計方法論

- 様々な方法論が提唱された
 - OMT (Object Modeling Technique) 法, Booch 法, ...

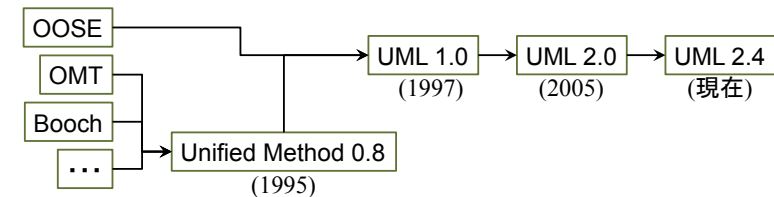


UML

UMLの背景

■ 歴史

- 様々なオブジェクト指向に基づく設計手法
 - OMT 法, OOSE 法, ...
- 手法の淘汰, UML へ統一化が進む
 - OMG による UML の標準化



■ オブジェクト指向

- オブジェクト間の関連
- オブジェクト指向によるシステム分析

■ UML

- ユースケース
- クラス図

■ 演習

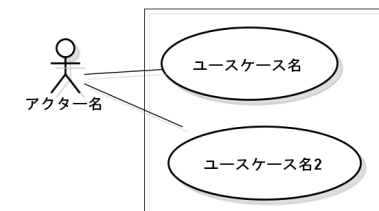
UML: ユースケース

■ システムの機能と外部環境を表わす図

- ユーザとシステム間の処理を記述する
 - ユーザの視点からシステムが見える

■ 作成手順

- システムの機能を抜き出す
- 外部環境を抜き出す
 - ユーザなど



UML: ユースケース – 表記法

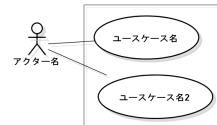


■ アクター

- 人間, 外部システムを表わす
- システムに何らかのイベントを発生させる主体

■ ユースケース

- システムの構成単位を抽象化したもの
- アクターとシステムの対話をモデル化する
- すべてのユースケースでシステム全体を表わす (ユースケース群)

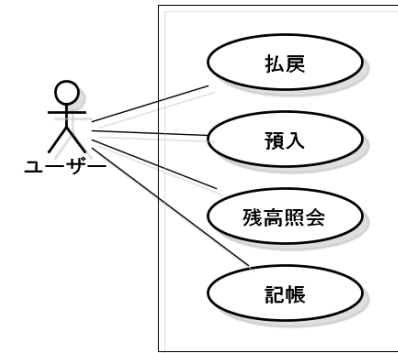


例: ATMのユースケース



■ ユーザがATMで使える機能

- 払戻, 預入, 残高照会, 記帳



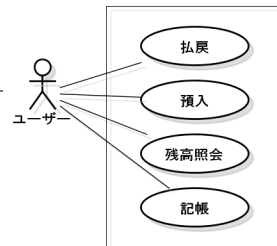
例: ATMのユースケース



■ 機能に対するシナリオ

「払戻」のシナリオ

1. ユーザはATMのメニューから「払戻」を選択する
2. ユーザはカードを挿入する
3. ユーザは暗証番号を入力する
4. システムは暗証番号を確認する
5. システムは該当の口座が存在することを確認する
6. ユーザは払戻額を入力する
7. システムは口座の残高を確認し, 払戻額よりも残高が多い場合, 出金する



■ オブジェクト指向

- オブジェクト間の関連
- オブジェクト指向によるシステム分析

■ UML

- ユースケース
- クラス図


■ 演習



UML: クラス図



- システムを構成するクラス, および, クラス間の関係を表現する図(ダイアグラム)
 - クラス, クラス間の関係の表現
 - システムの静的な側面の記述

 一般に, オブジェクト指向開発ではクラス図を中心として開発を進める

Jun. 8, 2015

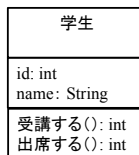
ソフトウェア設計法及び演習, Lesson08

25

クラス図: 表記法



- クラス
 - 長方形で表わす
 - 水平方向に3つの部分に分ける. 上から, クラス名, データ(属性)リスト, メソッドリストを記載する



Jun. 8, 2015

ソフトウェア設計法及び演習, Lesson08

27

UML: クラス図 – 作成手順



1. オブジェクトの識別
2. オブジェクトのデータ属性とメソッドの識別
3. オブジェクト間の関連の分析

Jun. 8, 2015

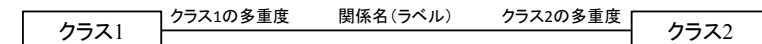
ソフトウェア設計法及び演習, Lesson08

26

UML: クラス図 - 表記法



- 関連
 - 関連とは, 協調するオブジェクト間の関係
 - あるオブジェクトが他のオブジェクトの属性や操作を使用するとき, オブジェクト間に関連があるという
 - クラス間の関連を, 接続線で表わす
 - 意味的な関連は接続線に近接する所に関連名を記入する
 - 多重度は接続線の両端に記入する



Jun. 8, 2015

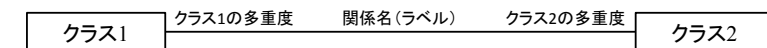
ソフトウェア設計法及び演習, Lesson08

28

UML: クラス図 - 表記法

■ 接続線の表記

関係	表記
関連	_____
汎化	◁_____
集約	◇_____
コンポジション	◆_____
依存	←-----
実現	◁-----

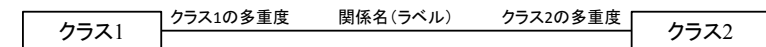


本講義では、関連と汎化、集約しか取り上げていません

UML: クラス図 - 表記法

■ 多重度の表記

表記	意味
1	1個
0..n または *	0以上
1..*	1以上
x..y	xからyの連続値 例) 1..n : 1以上 例) 2..5 : 2 から 5
x, y, z	離散値 例) 1, 3, 5 : 1 または 3 または 5

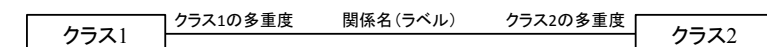


UML: クラス図 - 表記法

■ 関係名の表記

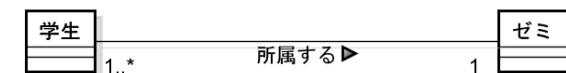
□ 関係名とは、意味を明確にするためのラベル

- 動詞で表記する
- 必要に応じて矢印(▶)を描き、解釈の向きを示す

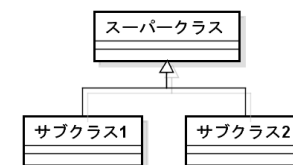


例: クラス図

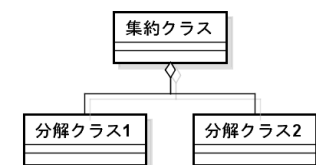
■ 関連(多重度, 関係名)



■ 汎化 (IS-A)



■ 集約 (PART-OF)




- オブジェクト指向
 - オブジェクト間の関連
 - オブジェクト指向によるシステム分析
- UML
 - ユースケース
 - クラス図
- 演習

演習8-2: 自動販売機

自動販売機を分析・設計する

- ユースケースの記述
 - ユーザと機能の抽出
 - シナリオの作成
- クラス図の記述
 - 対象の分析
 - オブジェクトの識別
 - 関連の作成



 本講義では、説明を重視し、仕様を明示しません。
また、データ属性とメソッドの識別の過程を省略します

演習8-1: Astah*のライセンス更新

- 更新(登録)手順
 - ポータルから、ライセンスファイルをダウンロードする("JUDE_License_User_Professional.xml")
 - Astah* Proをインストールしたディレクトリに、1.のファイルをコピーする
 - Astah*の[ヘルプ]-[バージョン情報]で、ライセンスが更新されていることを確認する



演習8-2-1: ユースケース

- 自動販売機の機能を抽出せよ
 - 購入に関する機能を含めること
 - (すべての機能で販売機全体を表わすように)
- ユースケースをAstahを用いて描け

演習8-2-2: シナリオ



■ 商品の「購入」に関するシナリオを記述せよ

「購入」のシナリオ

1. 購買者は、()に金銭を投入する
2. 購買者は、自動販売機のメニューから商品を選択する
3. 自動販売機は、選択された()をストックから()に搬出する
4. 購買者は、商品取出し口から()を取り出す

演習8-2-3: 対象の分析



■ 隣同士またはグループでヒアリングを行い、自動販売機を分析せよ

名詞

動詞

対象の分析



■ モデリングの対象を分析する

□ 対象を説明する**名詞・動詞**などに着目し書き出す

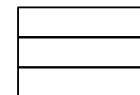
- 具体的な語を可能な限り書き出す
(付箋を用いてブレインストーミングをすると良い)
- 「名詞」「動詞」程度に分類するのみで十分

演習8-2-4: オブジェクトの識別



■ オブジェクトを識別し、クラス図をAstarhで記述せよ

クラス



オブジェクトの識別



■ 名詞から、オブジェクトを抽出する

□ (常識的に考えて) **オブジェクト化する必要がない語**を整理する

- 同義語 (例: 価格と値段)
- 抽象的すぎる名詞
 - 例: 設計開発の対象そのもの, なんでもできるオブジェクト
- 詳細すぎる名詞
 - 例: 「制御信号」など細かい実装手段

□ **属性や操作に分類できる語**を整理する

- 属性とするかクラスとするかの判断は経験が必要?!

演習8-2-5: 汎化



■ 演習8-2-4で作成したクラス図に, スーパークラスとサブクラス間の汎化関係を加筆せよ

汎化



■ 抽出されたクラスを, 意味のあるグループとしてまとめ, 抽象クラスを作成する

- **グループ**と考えられるクラスをまとめる
- グループを代表する(抽象)クラスを作成する
 - 抽象クラスも階層構造になる可能性があることに注意

演習8-2-5: 関連

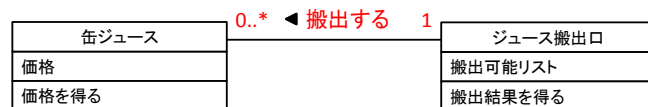


■ オブジェクト間の関連(汎化以外)をクラス図に加筆せよ

関連



- クラス間に「関連」をつけ、関係名(ラベル)や多重度を付ける
 - クラス間に関連の線を引く
 - 関連に関係名を付ける
 - 関係名は、分析で得られた動詞を参考とする
(一方の操作は、他のクラスと関係しない動詞)
 - 関連に多重度を加える



まとめ



- オブジェクト指向
 - オブジェクト間の関連
 - オブジェクト指向によるシステム分析
- UML
 - ユースケース
 - クラス図
- 演習