

ソフトウェア開発の基本技術 (分析と設計)

ずっと受けたかった ソフトウェアエンジニアニング
の授業 1 p82~p123

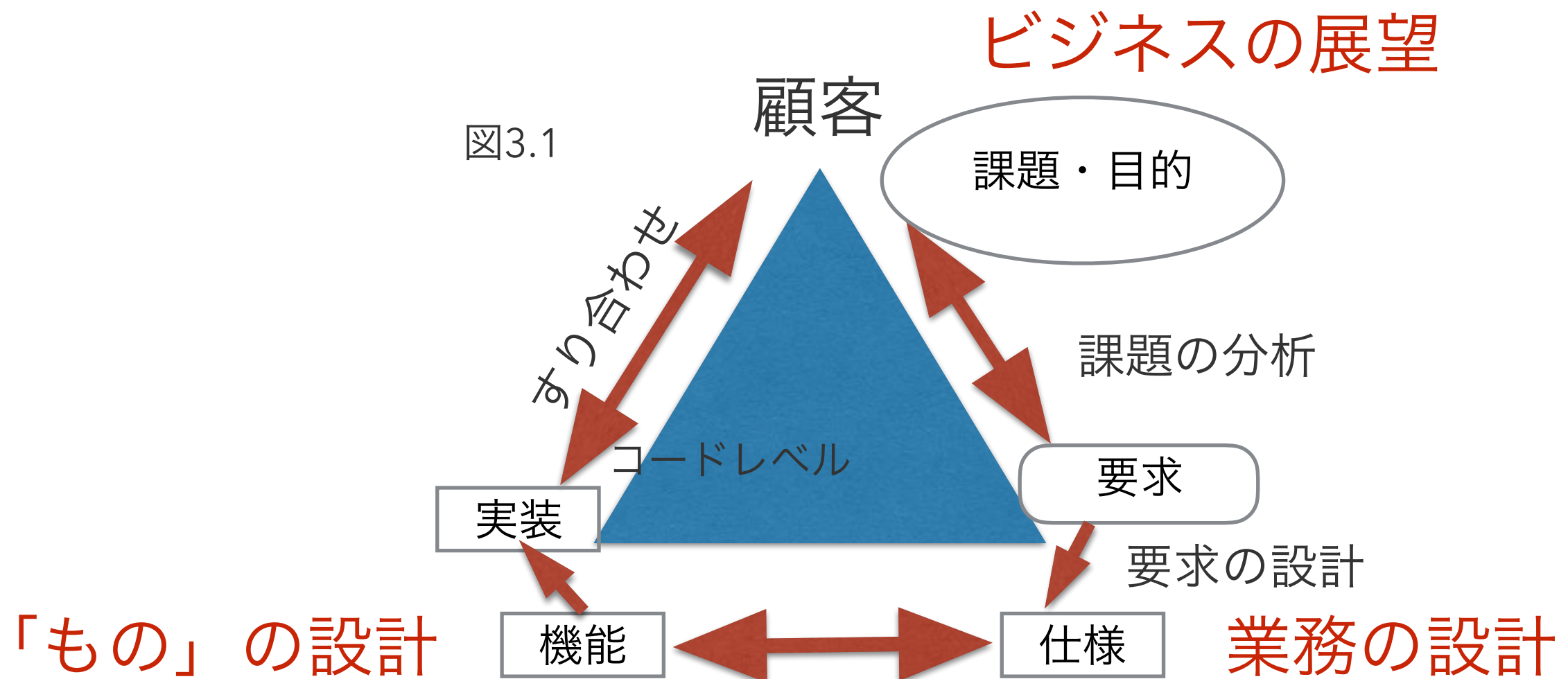
福井 克法

概要

- ・はじめに
- ・ソフトシステムアプローチ
- ・構造化分析・設計
- ・データ中心設計
- ・オブジェクト指向分析・設計
- ・モデルとモデリングについて
- ・まとめ

はじめに

- 分析 「ある事柄の内容・性質などを明らかにするため、細かな要素に分けていくこと」
- 設計 「対象物の構造・材料・製作法などの計画を図面に表すこと」



ソフトシステムアプローチ

- システムを作る目的

顧客が何を要望しているのか (What) を明らかにする。

「Whatはお客様が抱える課題」

- 問題解決のためのアプローチ

解決すべき問題が明確な場合は、解決するための効率的な方法(How)を求める。

ハードシステムアプローチ

オペレーションズリサーチ、システムエンジニアリング

解決すべき問題が不明確な場合 何であるか (What)を決める。

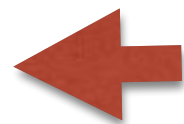
ソフトシステムアプローチ

ソフトシステム方法論 (SSM)

- SSM:Soft Systems Methodology)
- 組織における問題そのものを認識し、解決に向けて合意を形成するためのソフトシステムアプローチ
- SSMは7つのステージで構成されており各ステージを行き来し検討を進める。

● SSMの7つのステージ

- | | |
|------------------|--------------------|
| 1. 問題状況の把握 | 6. 実現可能で望ましい解決案の定義 |
| 2. 問題状況の表現 | 7. 問題を改善するための行動 |
| 3. 問題の定義 | |
| 4. 活動モデル（解決策）の構築 | |
| 5. モデルと現実の比較 | |



XYZ分析

分析の種類

- 分析の方法論としては2種類ある。

構造化分析とオブジェクト指向分析

- 構造化分析

お客様要望の目的や機能を考えることが基本

例：椅子を説明する場合、「人が座る」という目的を述べる。

- オブジェクト指向分析

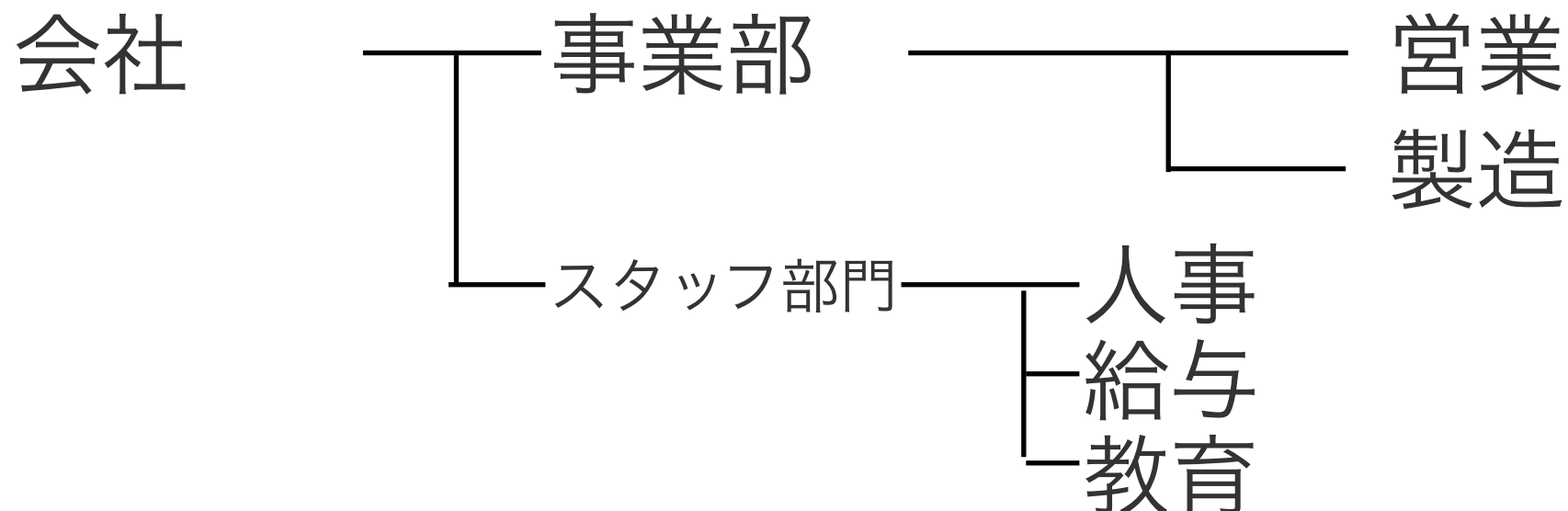
実際に存在するモノやことに着目するという考え方

例：それは、椅子というものであると述べる。どういう素材でできているか、どういう形かなど

構造化分析・設計

- 構造化とは
 - 構造化プログラミング

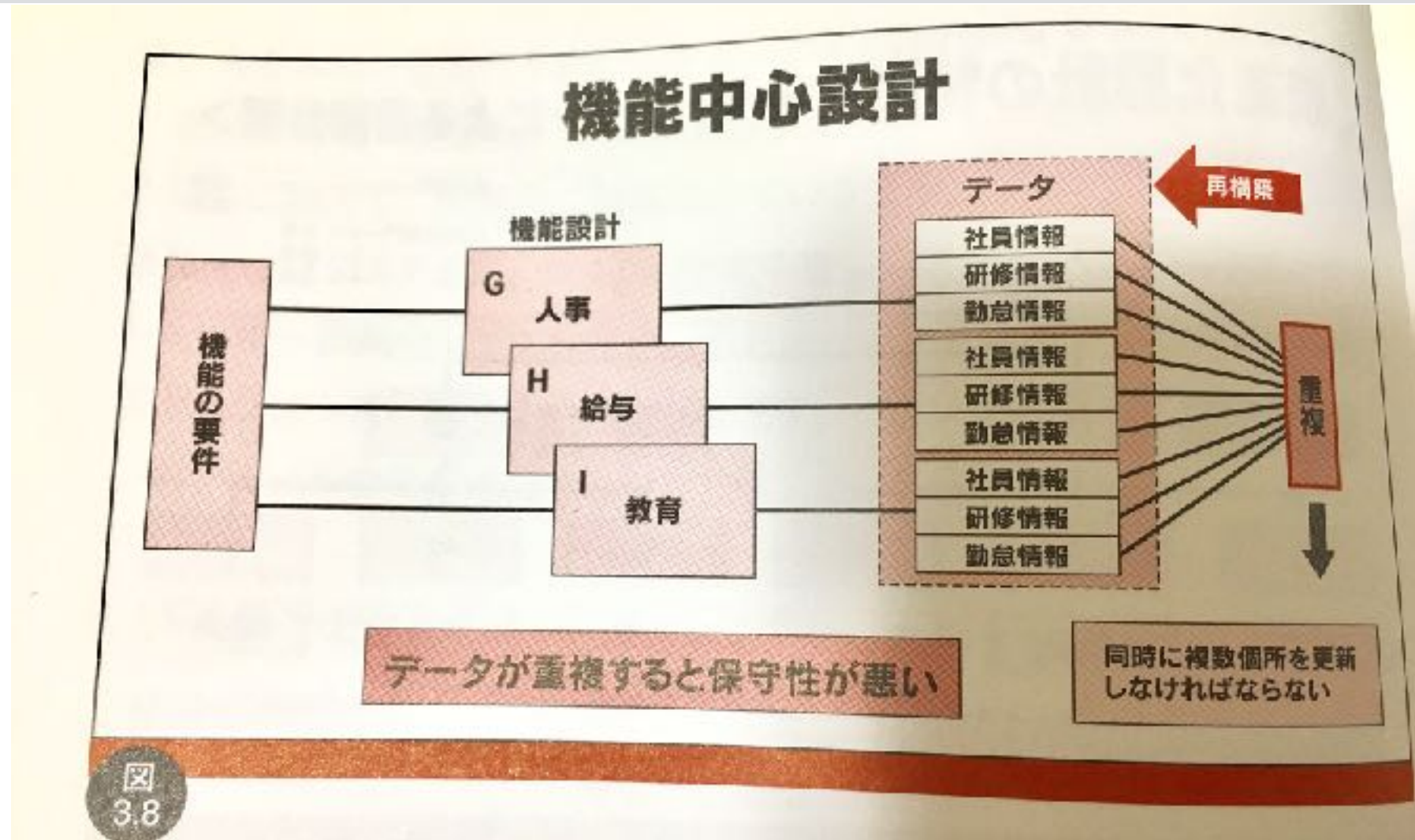
単純でわかりやすいプログラミングの組み合わせで実現すること
(順構造、分岐構造、繰り返し構造)
- 設計の基本的な考え方
 - 小さな要素に分解してそれぞれが独立したものであるものとして業務を進め効率化を図るという考え方。



構造化設計の特徴

- 構造化設計とは
全体の機能をトップダウンで機能階層（モジュール）に分けていくという考え方
- モジュールはプログラムの集まり。互いに独立、小規模、業務に対応。

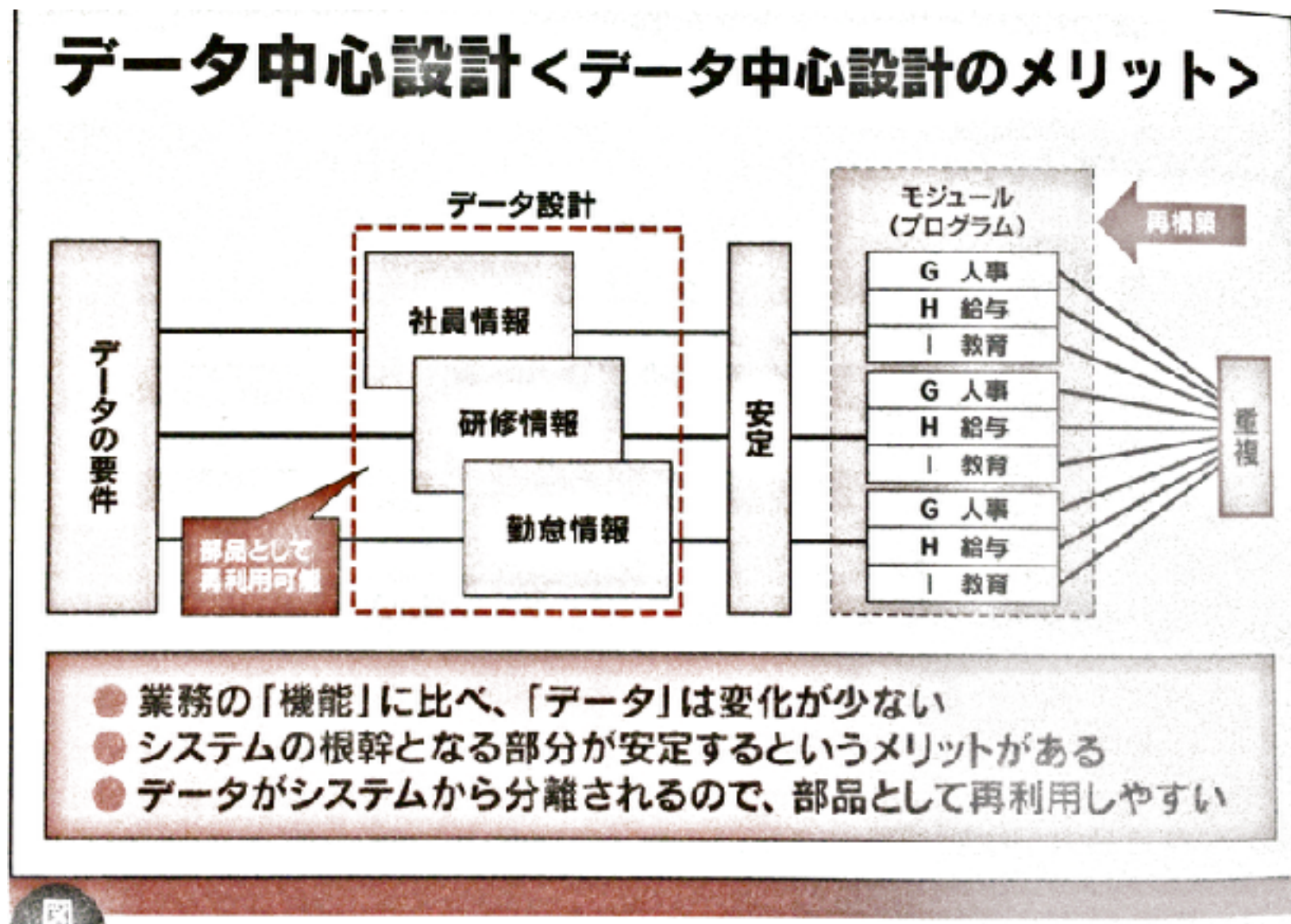
機能中心設計



- 欠点
 - データが重複すると保守性が悪い
 - 構造化設計では、データの重複を回避するため、全体のデータベースを再構築しなければならない。

データ中心設計 (DOA)

- データ分析からスタートする手法
- 機能ではなくデータが細分化され、それに対するモジュールが定義される。

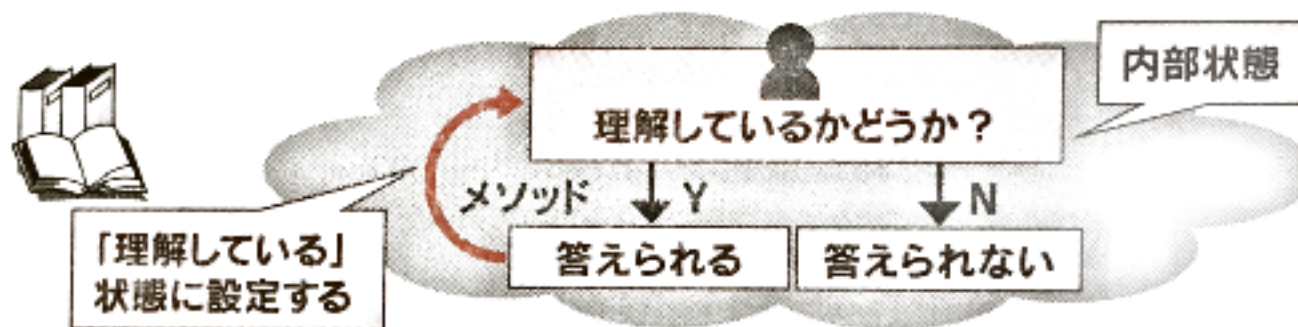


オブジェクト指向分析・設計

- オブジェクト指向
 - ・ 実際に存在するモノやコトに着目するという考え方
 - ・ 内部状態を有し、内部状態にはメソッドというインタフェースを介してのみアクセスできる特徴を持つ

オブジェクト指向設計の基本的な考え方

- オブジェクトは実世界を反映したもので、動作も表す
- オブジェクトは内部状態を有し、内部状態にはメソッドという特定の手段を介してのみアクセスできる



従来は現実の人間の仕事の仕方を記述するには単純過ぎた



従来のコンピュータ言語にはない大変便利な機能

オブジェクト指向の基本概念 1

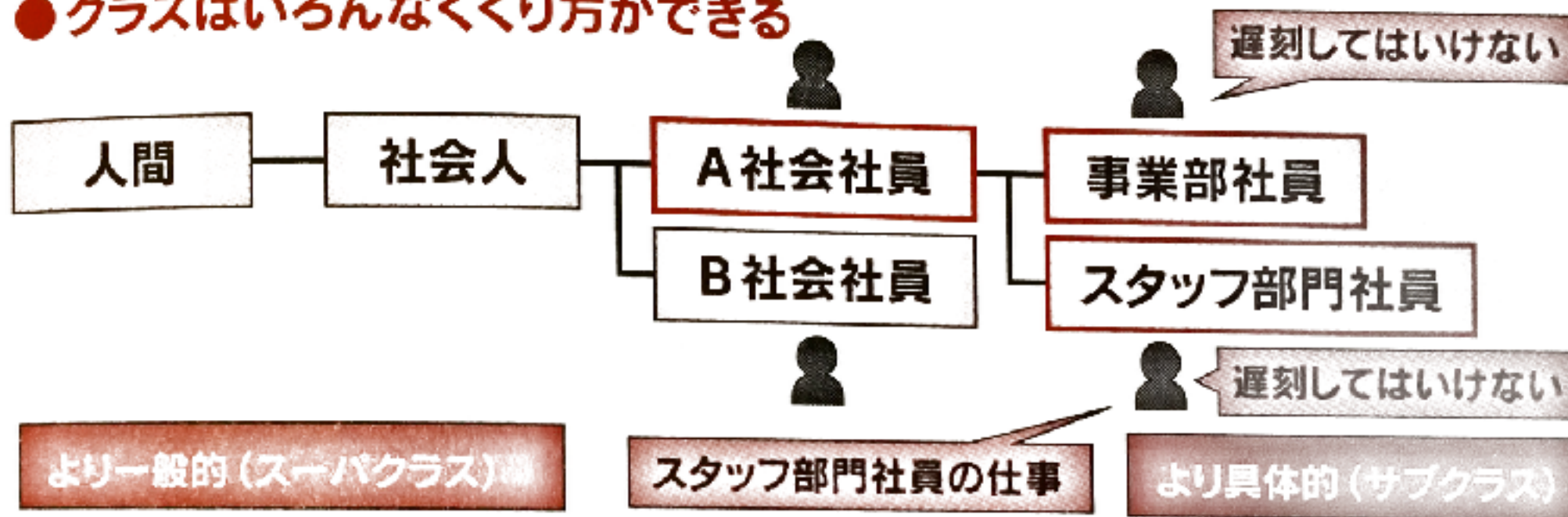
- カプセル化
 - オブジェクトの内部情報を外部に対して隠蔽する機能
- 抽象化（クラス・オブジェクト）
 - 仲間を1つにまとめること
- メッセージ
 - カプセル化によって内部情報を隠蔽したオブジェクトに処理をさせるために最低限必要な情報を送る。この情報を「メッセージという」
- ポルモーフイズム（多様性）
 - 異なるクラスをまとめて扱うこと
 - 異なるメソッドを異なるクラスに対して実行させること
 - メッセージを受けた方が振る舞いを変える

オブジェクト指向の基本概念 2

- 階層化と継承

オブジェクト指向設計の基本概念<クラスの階層化(継承)>

- クラスはいろんなくり方ができる



- 継承とは ... サブクラスがスーパークラスの性質を引き継ぐこと (逆は言えない)
- 差分プログラミングとは ... サブクラスはスーパークラスとの違いのみをプログラムすればよい

オブジェクト指向のまとめ

- オブジェクト思考は現時点で最も有力な技術である
- しかし、万能ではない。
- オブジェクト指向は、従来のソフトウェア開発の上に成立する。
- 最も望まれるのは、「プロジェクトの経験とオブジェクト技術を併せ持った技術者」である

モデルとモデリングについて

- 「具体的な個々の事象から枝葉を除き本質的な要素を抽出して抽象化したもの」

例： ER図、状態遷移図、状態遷移表などがある

利点

- 正確な設計が可能になる
- 設計が見えるようになり、レビュー効率が向上する
- コミュニケーションが円滑になる
- シミュレーションや検証をツール化できる
- 標準化が促進され、再利用が安易になる

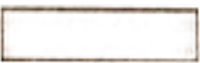
まとめ

- ソフトウェアの分析や設計を行うにあたって、どれか一つの手法のみを使うのではなく、様々な視点で要求を確認して分析・設計することが重要
- 開発者は、様々な視点で行った分析や設計が、全体として1つにまとまっているかどうかを念頭に置いておく必要がある。
- 1つの側面だけで設計すると、全体として使いづらくなったり、性能が悪くなったり、保守がしにくくなる場合がある。

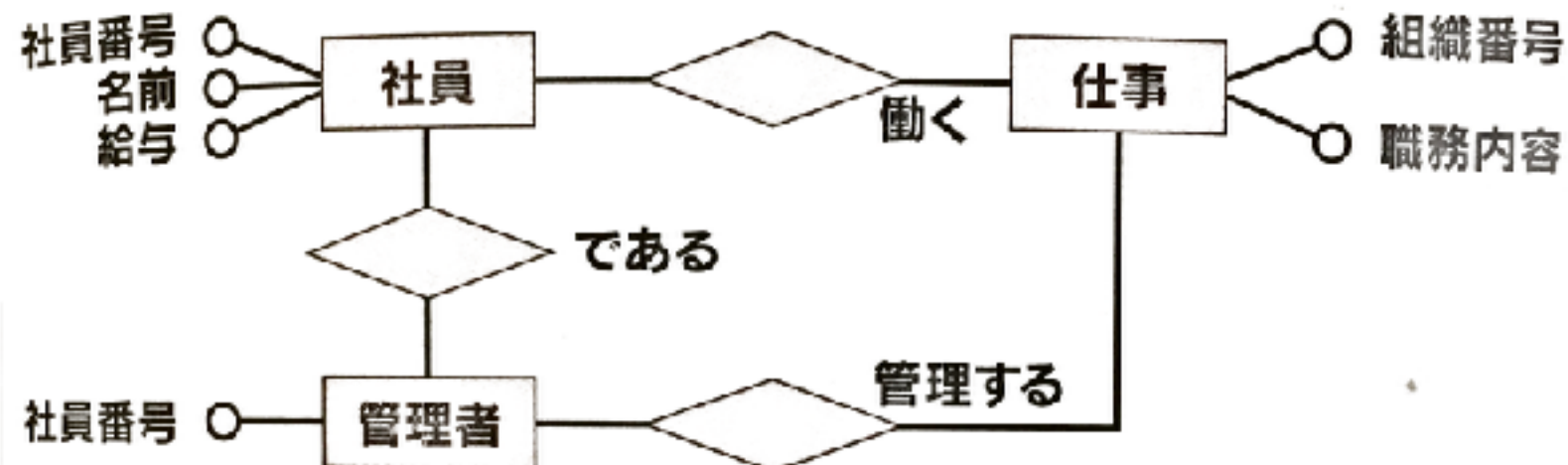
ER図の構成要素

DOAでは、データをER(Entity Relationship)ダイアグラムで明示的に表現する

- 実体 (Entity) は実世界を反映したもの
- 実体 (Entity) の間を関係 (Relationship) で結ぶ
- 実体 (Entity) と関係 (Relationship) には属性 (Attribute) がある

記号	名称	意味
	実体 (Entity)	データの実態を表す
	関係 (Relationship)	実体 (Entity) 間の関係を表す
	属性 (Attribute)	実体 (Entity) のもつ性質を表す

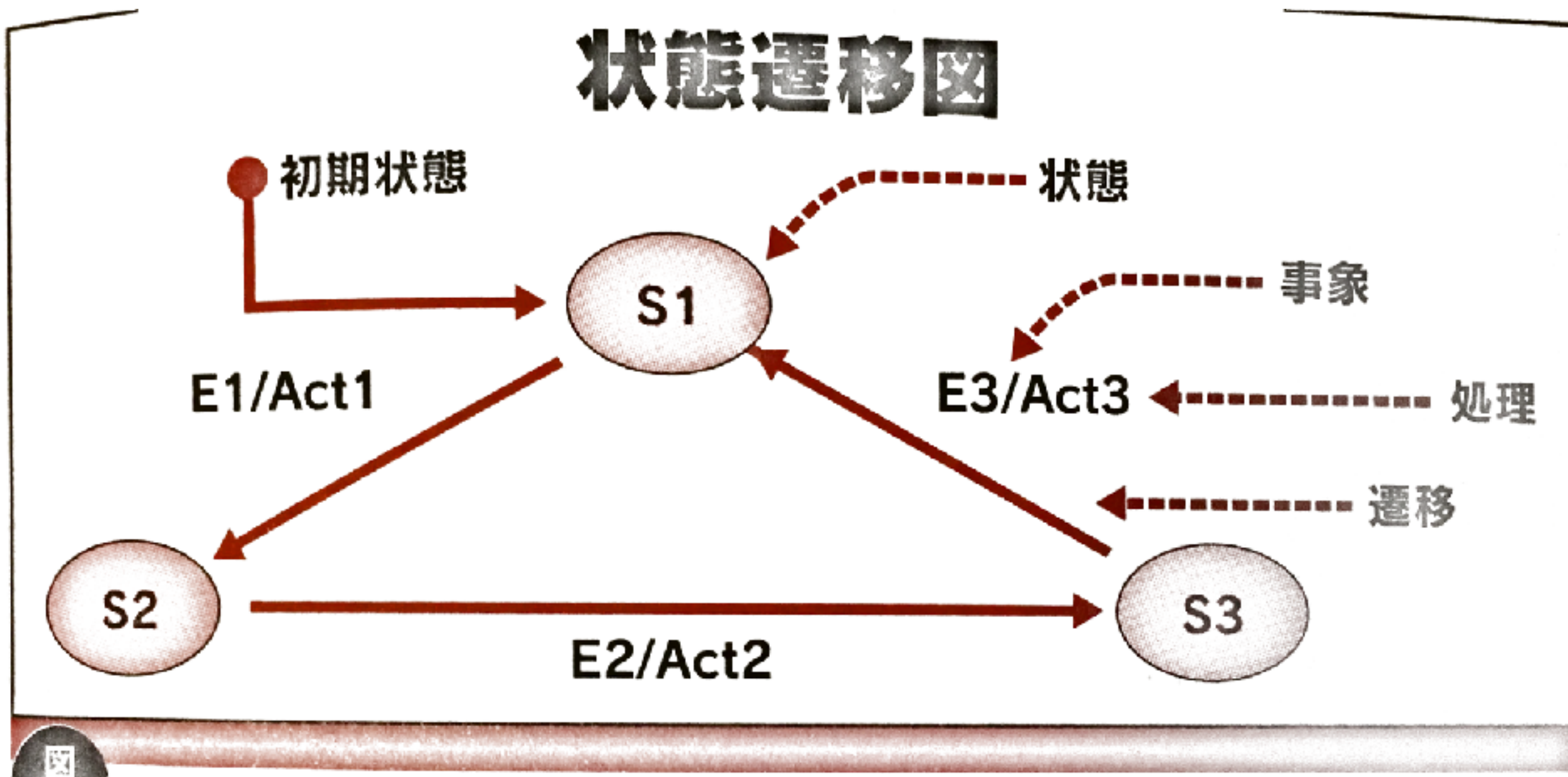
ER図の例



出典：『ソフトウェア開発の技法』(山村信吉著/三元社)

状态迁移图

状态迁移图



狀態遷移表

狀態遷移表

初期狀態

	S1	S2	S3	狀態
事象	E1	S2 Act1		
	E2		S3 Act2	
	E3		S1 Act3	遷移先 處理