

# 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

関澤 俊弦

日本大学 工学部 情報工学科

# 復習

---

## ■ 開発工程（開発プロセス）

### □ ウォーターフォールモデル

- ・ 要件定義, 設計, 実装, テスト, 導入・保守

### □ プロトタイプモデル

### □ スパイラルモデル

### □ アジャイル

# 本日の講義内容

---

## ■ 構造化分析(教科書3章)

- システム要件定義時の問題点
- 構造化分析の必要性
- 構造化分析の手順
- 機能の階層化
- システム要件把握の実施例

## ■ 演習

- 構造化分析(現行システムの分析)

# 本日の講義内容

---

## ■ 構造化分析(教科書3章)

- システム要件定義時の問題点
- 構造化分析の必要性
- 構造化分析の手順
- 機能の階層化
- システム要件把握の実施例

## ■ 演習

- 構造化分析(現行システムの分析)

# システム要件定義時の問題点 (1)

## ■ 問題点

 p.33

- ユーザニーズの把握の難しさ
- データ収集能力
- 風土の違い
- 仕様書の理解のしにくさ
- あいまいな表現
- 変更の多発
- 感情とコミュニケーション
- 要件の不透明さ

システム要件把握の手順が  
明確になっていない  
→ 構造化分析

# システム要件定義時の問題点 (2)

---

## ■ ユーザニーズの把握の難しさ

- 大きなシステムになったとき, ユーザ自身でシステム要件を定義する例は少ない

- 開発者とユーザの関係は複雑

開発者: ベンダー

ユーザ: 企業の業務部門 (情報部門が介在して委託)

- 「ユーザ」とは誰か

- 最近はステークホルダー (利害関係者) が多いので, 気まぐれな要求やとんでもない装置の要求もある
- 真の意思決定者を的確に見つけることが先決

# システム要件定義時の問題点 (3)

---

## ■ データ収集能力

- ユーザ自身が自分の問題の解決法をよく知らないことがままある

## ■ 風土の違い

- 開発担当者は業務をよく知らない
- ユーザはコンピュータをよく知らない

## ■ 仕様書の理解のしにくさ

- 要件仕様書は一般にユーザには理解しにくい
  - SE(システム・エンジニア)は、ユーザ要件を業務中心ではなく、特定のハードウェアに関連して表現したり、ファイルの物理構造で表現したりする

# システム要件定義時の問題点 (4)

---

- あいまいな表現
  - 人はそれぞれ自分に都合のよいように解釈
- 変更の多発
  - 放っておくと変更は際限なく発生する
  - 1つのプログラムの変更が、他に大きな影響をおよぼすことも多い
- 感情とコミュニケーション
  - 感情的な態度はコミュニケーションを阻害する
- 要件の不透明さ
  - 最近のビジネス環境では、システム要件が時間とともに変動し、なおかつ短期間開発を余儀なくされる



# 本日の講義内容

---

## ■ 構造化分析(教科書3章)

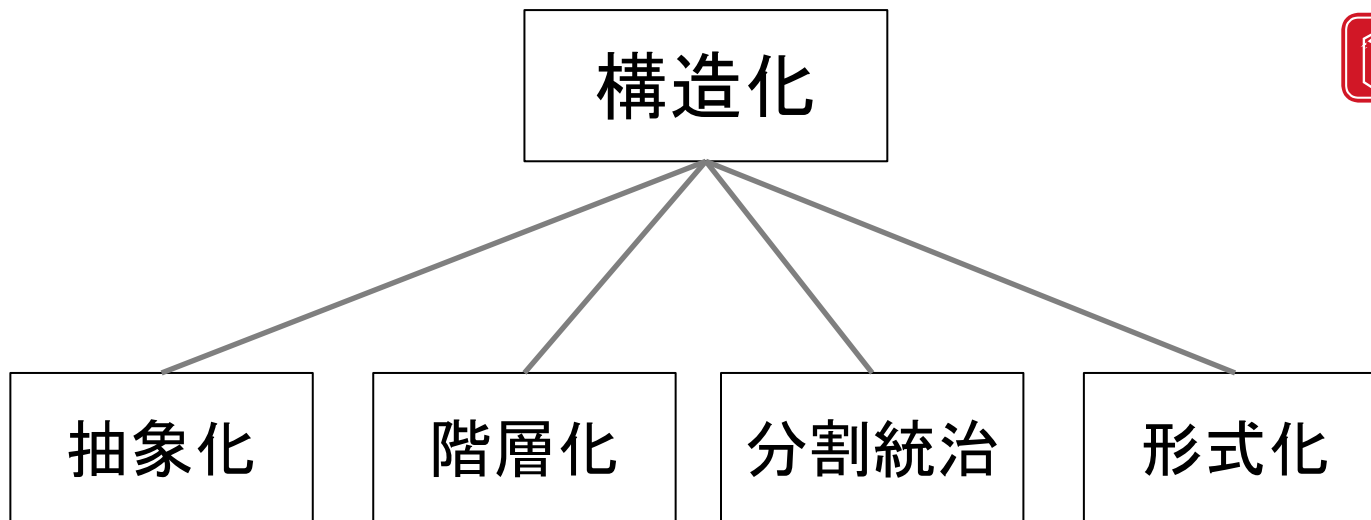
- システム要件定義時の問題点
- 構造化分析の必要性
- 構造化分析の手順
- 機能の階層化
- システム要件把握の実施例

## ■ 演習

- 構造化分析(現行システムの分析)

# 構造化分析の必要性 (1)

## ■ システムの複雑さを減少させる構造化の原理



p.38

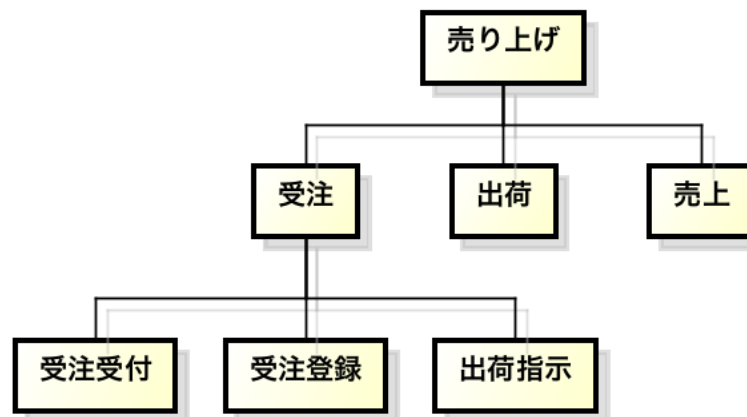
# 構造化分析の必要性 (2)

## ■ 抽象化の原理

- 抽象化とは、事実に関する細かな内容を捨象し、その事実の本質のみ説明すること

## ■ 階層化の原理

- システム機能を段階的に詳細化し、管理、権限の委譲、インタフェースの問題を考え易くする考え方

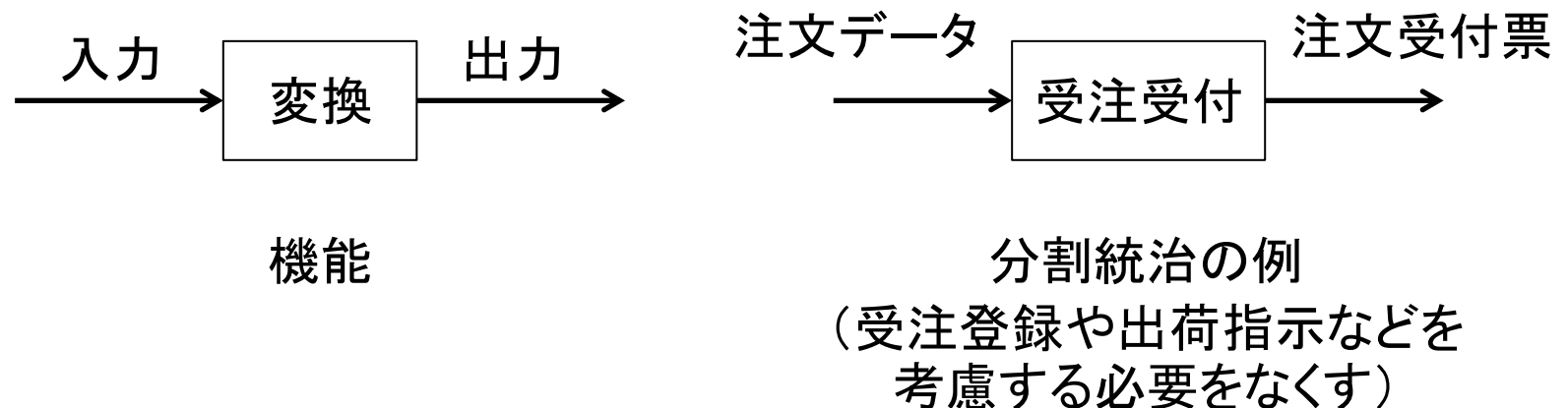


上位機能は下位機能を包括し、  
下位機能全体を抽象化

# 構造化分析の必要性 (3)

## ■ 分割統治(独立性)の原理

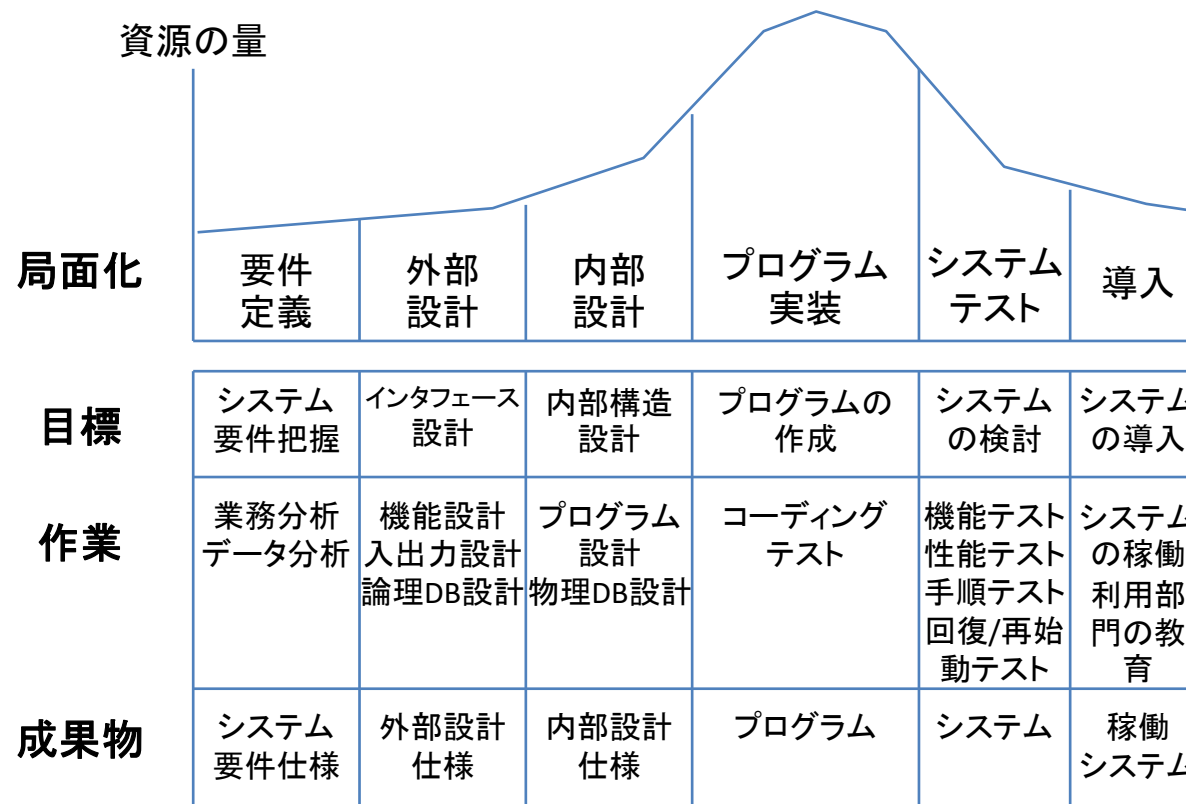
- 複雑な問題をいくつかのより小さな問題に分割して, 問題解決を図る考え方
- システムを機能的に入力, 変換, 出力へ分割



# 構造化分析の必要性 (4)

## ■ 形式化の原理

□ 共通の土台の上で議論を可能にする考え方



形式化の例  
(システム開発工程)

# 本日の講義内容

---

## ■ 構造化分析(教科書3章)

- システム要件定義時の問題点
- 構造化分析の必要性
- 構造化分析の手順
- 機能の階層化
- システム要件把握の実施例

## ■ 演習

- 構造化分析(現行システムの分析)

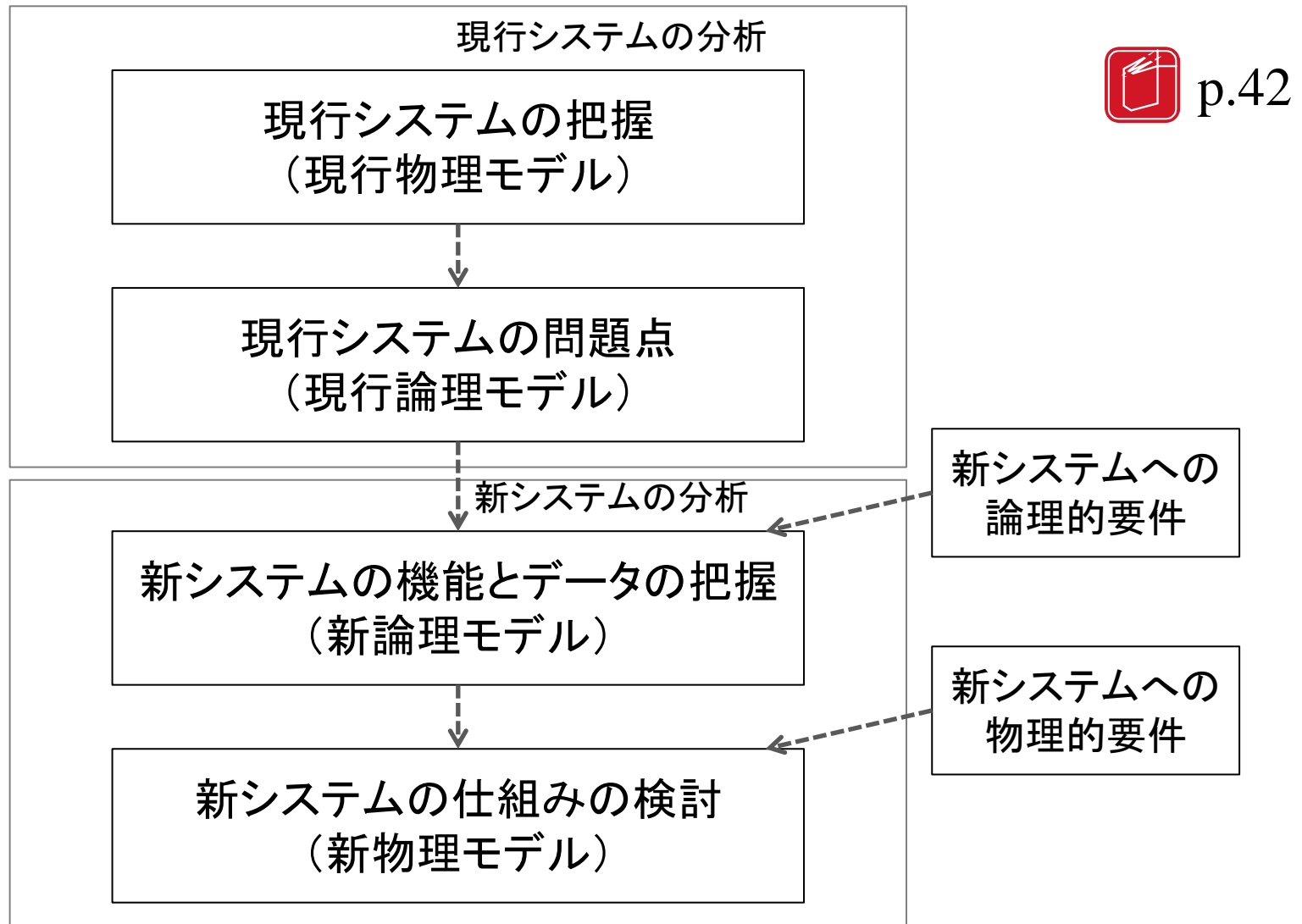
# 構造化分析・設計

## ■ 機能階層構造を作り出す手法の1つ

- 1) 要求仕様書に基づく問題の理解
- 2) 実現する機能のDFD(データフロー図)の作成
- 3) DFDから機能階層図(手続き)の作成



# 構造化分析の手順





# 現行システムの記述 (1)

---

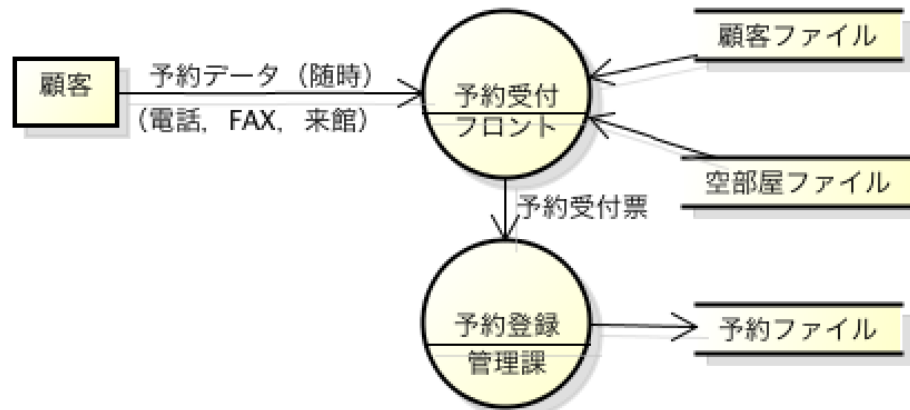
## ■ 現行物理モデルの作成

- 業務で行う機能とデータ, 機能の担当者, 実施タイミング, データの伝達媒体などをそのまま記述

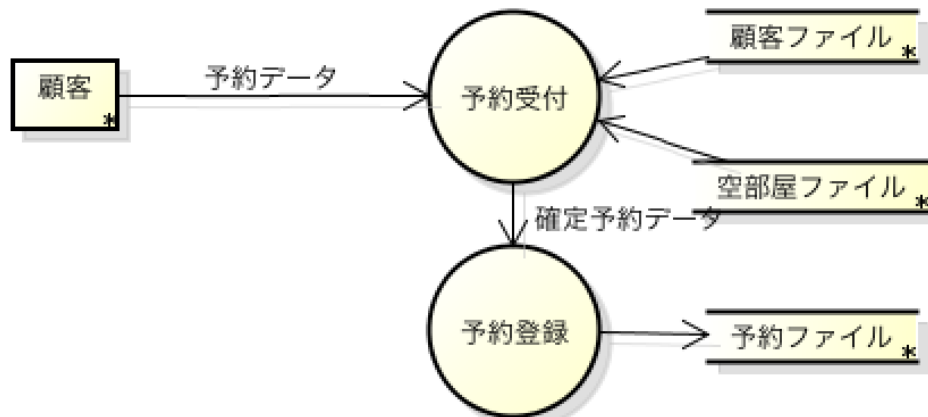
## ■ 現行論理モデルの作成

- 人, 組織, タイミング, 媒体といった種々の物理的制約を取除き, 業務遂行に本質的に必要な機能とそれに関連したデータのみを表現

# 現行システムの記述 (2)



ホテル予約業務の現行物理モデル



ホテル予約業務の現行論理モデル

業務遂行に本質的に必要な  
**機能**と関連した**データ**の表現

(DFDの記号の意味については  
Lesson04で取り上げる予定)

# 新システムの定義 (1)

---

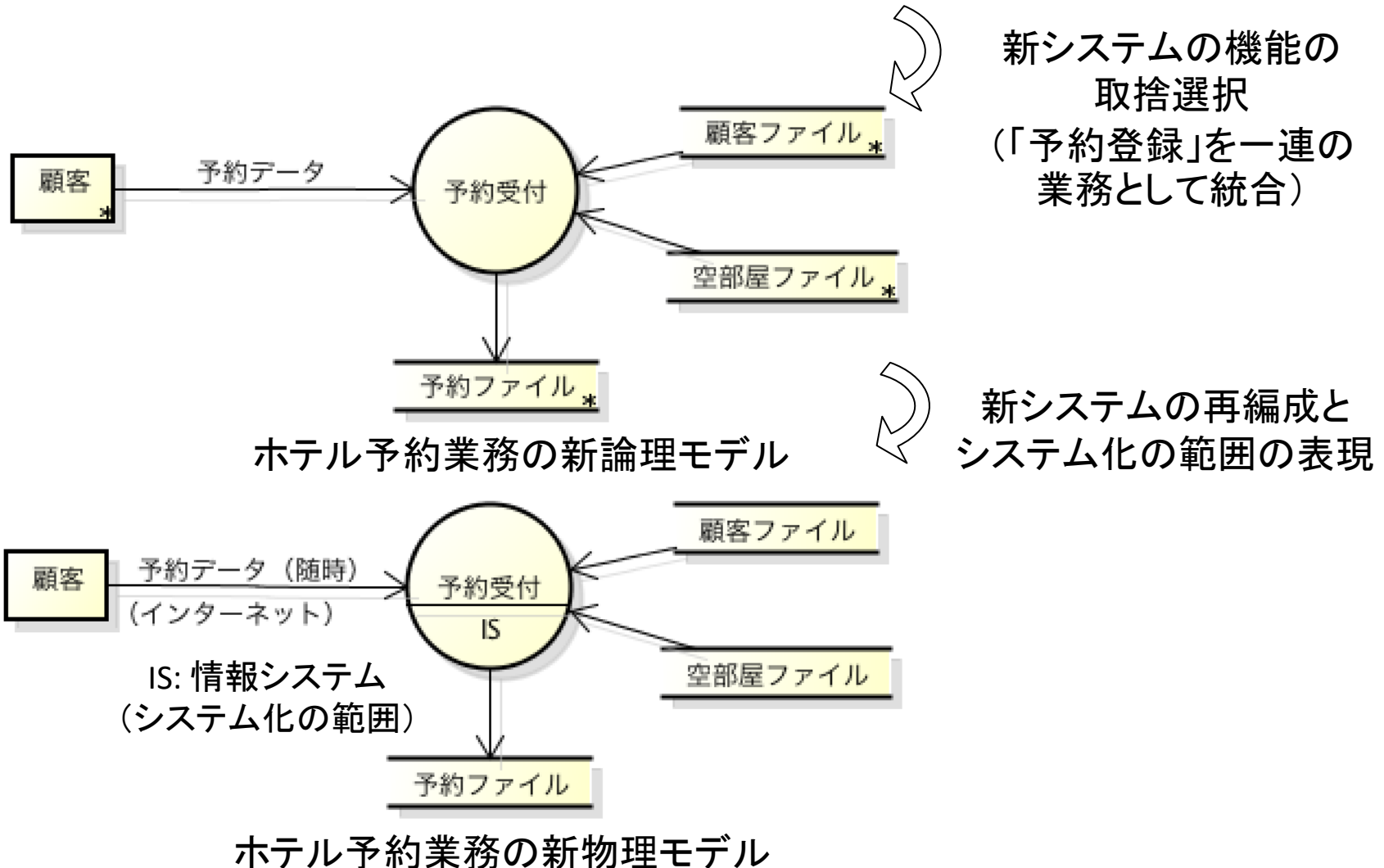
## ■ 新論理モデルの作成

- 新システムではどんな情報を必要とし, そのためにどんな処理機能が必要になるかを明確化
- 目的に照らし合わせて機能の取捨選択

## ■ 新物理モデルの作成

- 性能や媒体などの物理的要件や制約を考慮し, 新システム再編成とシステム化の範囲を明確化

# 新システムの定義 (2)



# 本日の講義内容

---

## ■ 構造化分析(教科書3章)

- システム要件定義時の問題点
- 構造化分析の必要性
- 構造化分析の手順
- 機能の階層化
- システム要件把握の実施例

## ■ 演習

- 構造化分析(現行システムの分析)

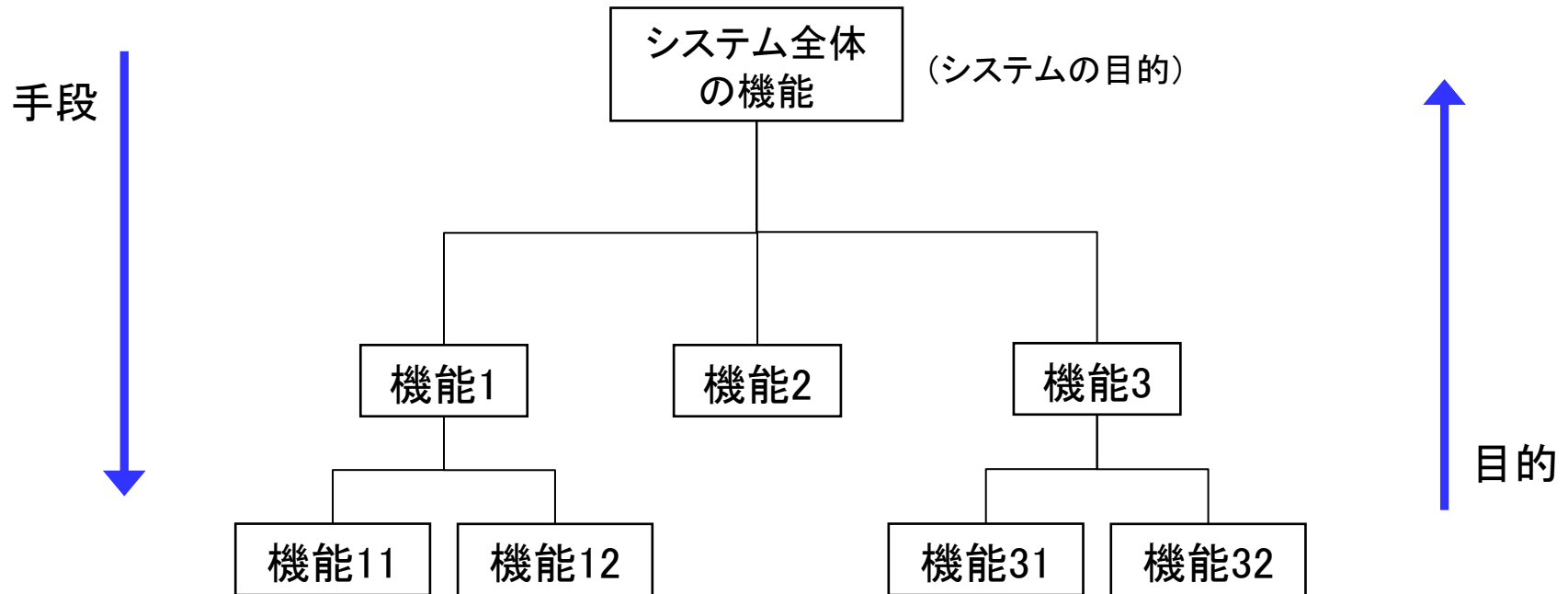
# 機能の階層化 (1)

---

- 独立のモジュール(手続き)に分割すること
  - 下位機能から順に分割統治をして開発計画を立てることができる
  - 2人以上で並行開発できる
  
- 手続き？
  - 繰り返し使えるプログラムの断片に手続き名(処理名または機能名)を付けたもの
  - 中身の詳細を知ることなく使えるという利点

# 機能の階層化 (2)

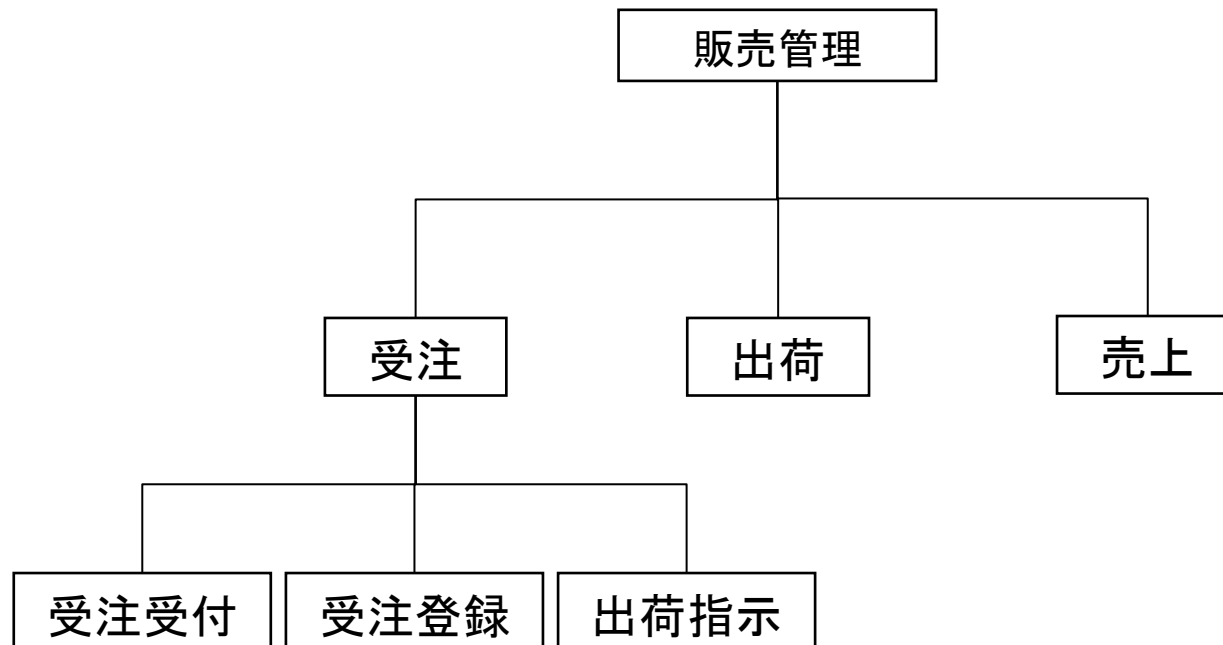
- 目的と手段の連鎖
  - 目的と手段で機能を階層的に展開すること



# 機能の階層化 (3)

## ■ 目的と手段の連鎖の例

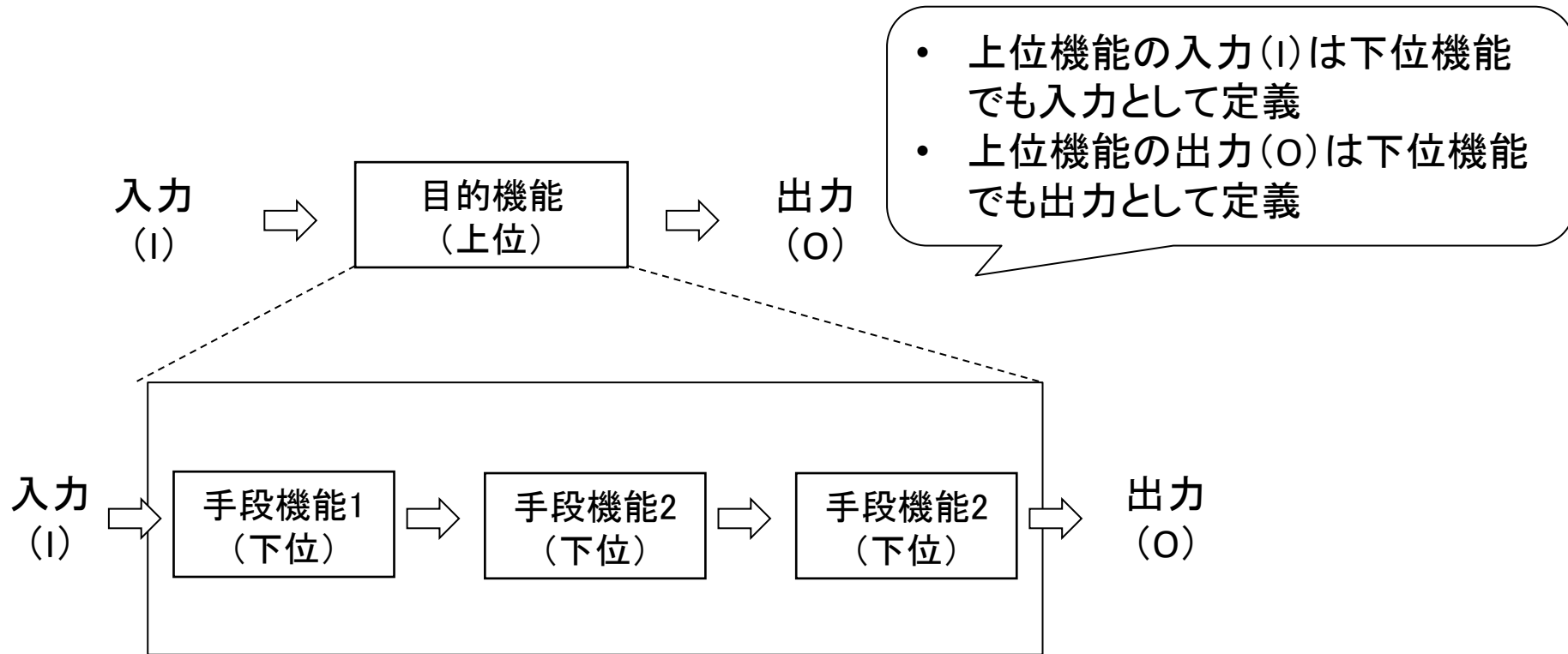
- システムの目的(販売管理)を達成するための機能(受注, 出荷, 売上)





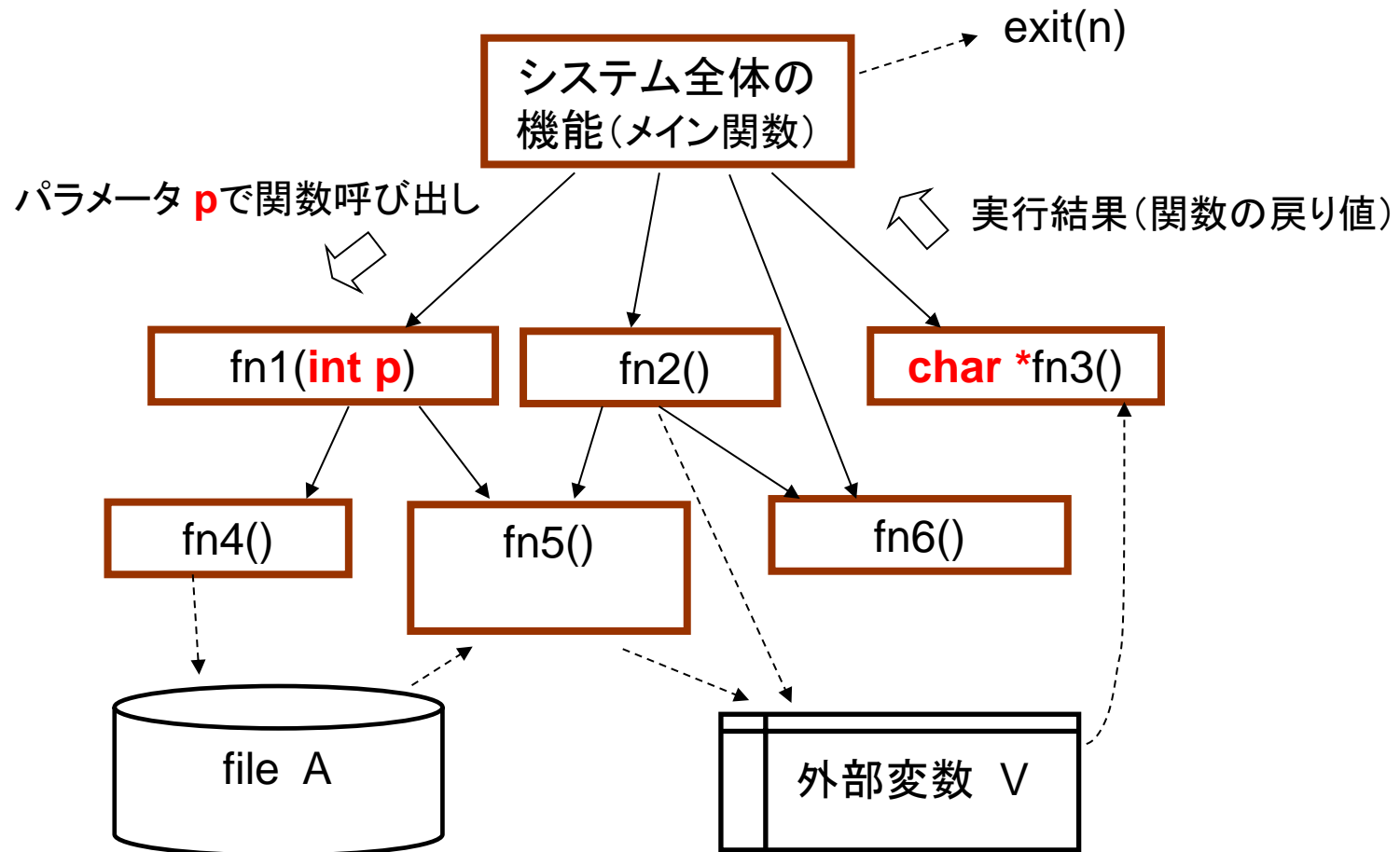
# 機能の定義 (1)

## ■ 機能の目的, 処理内容, 入出力



# 機能の定義 (2)

## ■ 各機能を関数として定義すると



# 本日の講義内容

---

## ■ 構造化分析(教科書3章)

- システム要件定義時の問題点
- 構造化分析の必要性
- 構造化分析の手順
- 機能の階層化
- システム要件把握の実施例

## ■ 演習

- 構造化分析(現行システムの分析)

# システム要件把握の実施例 (1)

---

## ■ 米国の大手生命保険会社の事例

- 1) ユーザ部門との協力体制を確立する
- 2) ユーザとの合同ミーティングで、まずシステムの概要の検討から開始する
- 3) 続くミーティングでリストされた個々の機能に1つずつ焦点をあてて検討する
- 4) システム機能階層構造(機能階層図)の作成
- 5) トップ・ダウンによる階層構造の見直し
- 6) ウォークスルーの実施

# システム要件把握の実施例 (2)

---

## ■ ユーザ部門との協力体制を確立する

- システムの基本的な理解を得るために、開発者側の要員がユーザの現行業務を観察し、調査できるような体制を**ユーザ側の協力のもとに確立**
- 合同ミーティングを短時間で頻繁に行う

## ■ ユーザとの合同ミーティングで、まずシステムの概要の検討から開始する

- 必要な出力(情報)を明確にする
- 検討結果として考えられる機能リストを作成する

# システム要件把握の実施例 (3)

---

- 続くミーティングでリストされた個々の機能に1つずつ焦点をあてて検討する
  - 業務を遂行するシステムの問題を個々の機能に1つずつ焦点をあてて明確にする
  - ユーザは機能に関する詳細情報を収集
  - 開発側の担当者は検討結果を**文章化**
    - ミニ機能階層構造(機能階層図)
    - 機能ごとのIPOダイアグラム(入出力の明確化と入力を出力に変換する処理を図的に記述)
    - (ミニ仕様とデータ辞書)

# システム要件把握の実施例 (4)

---

- システム機能階層構造（機能階層図）の作成
  - ミニ機能階層構造をより大きな階層構造に統合
- トップ・ダウンによる階層構造の見直し
  - 開発者の責任者が以下のような見直しを行う
    - ・ 本来の目的は達成されているか
    - ・ 機能の記述はその内容を正しく反映しているか
    - ・ 機能の従属関係は正しく定義されているか
- ウォークスルーの実施
  - 必ず**ユーザ側の要員**を参加させ、検討機能を明らかにした上で、ミーティングを行って文書化

# 本日の講義内容

---

## ■ 構造化分析(教科書3章)

- システム要件定義時の問題点
- 構造化分析の必要性
- 構造化分析の手順
- 機能の階層化
- システム要件把握の実施例

## ■ 演習

- 構造化分析(現行システムの分析)



# 演習3-1: ジュース自販機のフローチャート

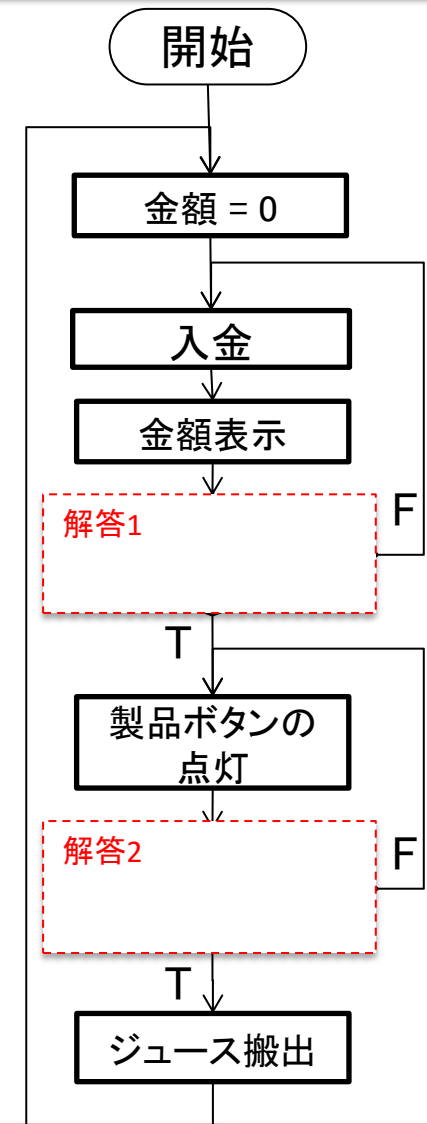
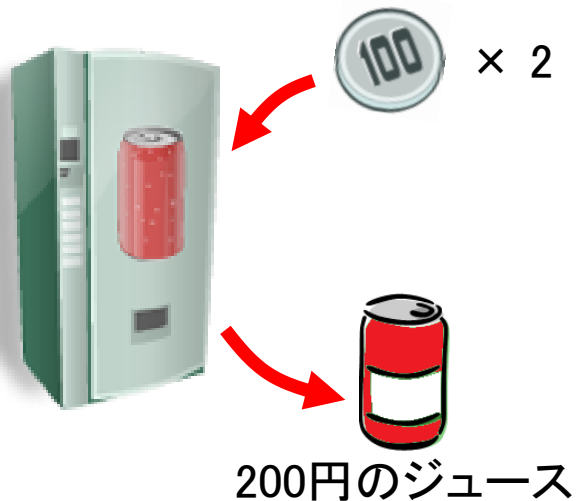
- 現行システムを分析する前に、以下のジュース自販機の仕様を参照し、右図のフローチャートをastah\*で描画せよ

- 前提条件:

- (1) 100円の硬貨のみ投入可能
- (2) 料金の返却処理なし

- 機能: 100円の硬貨を2回投入し、その後に製品ボタンを押すと、200円のジュース1本を搬出

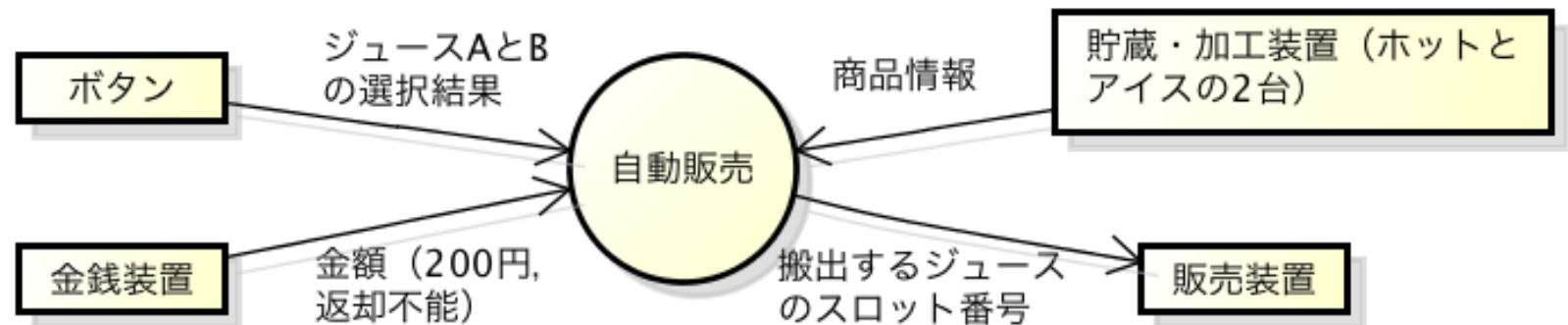
- ・ 金額が200円以上になると、「製品ボタンの点灯」を開始
- ・ 「製品ボタンの押下」によりジュース1本を搬出



## 演習3-2: ジュース自販機の現行モデル



- 以下の現行物理モデルを分析し、現行論理モデルを表すDFDを作成し、astah\*で描画せよ
  - 教科書p.44を熟読すること
  - 教科書と見比べて、下図のデータや外部実体の名前の中に物理的制約を示す表現がないか探してみる



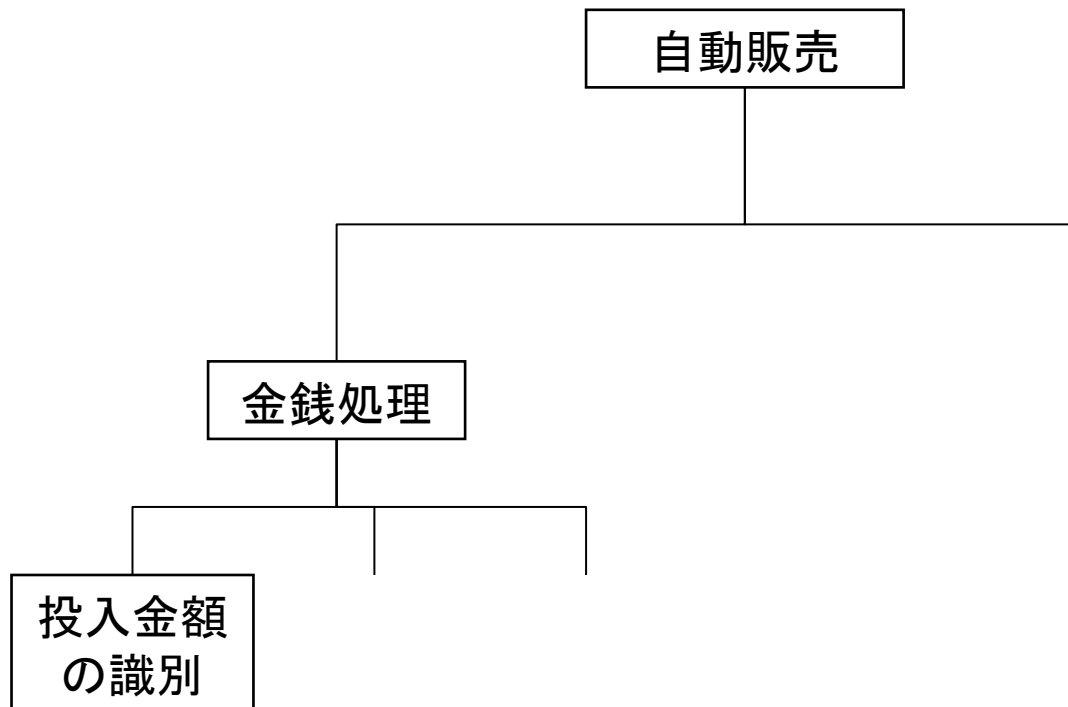
# 演習3-3: 自動販売の機能階層化

---

- 演習3-1のフローチャートと演習3-2のDFDを比較し, 現行システムを理解するうえでDFDの利点と欠点をそれぞれ箇条書きでまとめ, 自分のノートに記述せよ
  - DFDの利点
  - DFDの欠点

## 演習3-4: 自動販売の機能階層化

- 「自動販売」を実現するために必要な機能を追加し、以下の機能階層構造を自分のノートに記述せよ
  - この演習問題ではastah\*を用いて記述しなくてよい
  - 自動販売を2つ以上の機能(処理手順)に置き換えてみる



# 演習がおわったら

---

- 今回は提出物なし
- 次回のためにastah\* proのファイルを保存しておくこと
  - 例:「ソフトウェア設計法／lesson03\_1.asta」