
第1章 ソフトウェア開発の概要

- 1.1 ソフトウェアとソフトウェア工学
- 1.2 プロセスモデル
- 1.3 ソフトウェア開発のアプローチ
- 1.4 演習問題

この章のねらい

この章では、ソフトウェア工学の必要性、代表的なプロセスモデルとアプローチを紹介します。

1.1 ソフトウェアとソフトウェア工学

ソフトウェアという用語は、プログラムやデータベースなどコンピュータ処理の実施主体という狭い意味で用いる場合と、それらにプログラム仕様書や操作マニュアルなどのドキュメント類、更には操作する人まで含めた広い意味で用いる場合があります。

JIS では、ソフトウェアを以下のように定義しています。

ソフトウェアの定義(JIS X 0001)
情報処理システムのプログラム, 手続き, 規則及び関連文書の全体又は一部分

1.1.1 ソフトウェアの種類

一般的な、ソフトウェア(狭義)の分類は以下の通りです。

■ アプリケーションソフトウェア

ユーザの個別の利用目的を実現するソフトウェア。具体的には表計算やワードプロセッサなどの汎用性の高いソフトウェア、特定の用途のためにオーダーメイドで開発された給与管理システムなどの業務ソフトウェアが含まれる。

■ ミドルウェア

アプリケーションソフトウェアに共通する機能を提供するソフトウェア。代表的なミドルウェアには、データベースシステムを支えるデータベース管理システムや、Web システムを提供する Web サーバなどがある。

■ 基本ソフトウェア(OS)

すべてのアプリケーションソフトウェアやミドルウェアに共通するメモリ管理、プロセス管理、ファイルシステムなどの機能を提供するソフトウェア。

■ ファームウェア

ソフトウェアの指示(機械語)をハードウェア(回路)に伝達するためのソフトウェア(マイクロプログラム)。

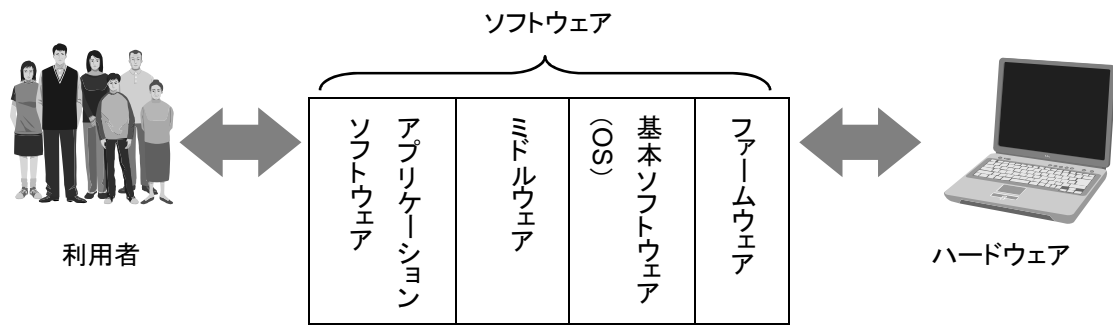


図1-1 ソフトウェアの種類

1.1.2 ソフトウェア工学の必要性

■ 一般的なものづくりとソフトウェア開発の比較

ものであれ、ソフトウェアであれ、ユーザの要望に基づき、何かを開発する際の基本的な流れは同じです。しかし、ソフトウェア開発には特有の難しさがあります。

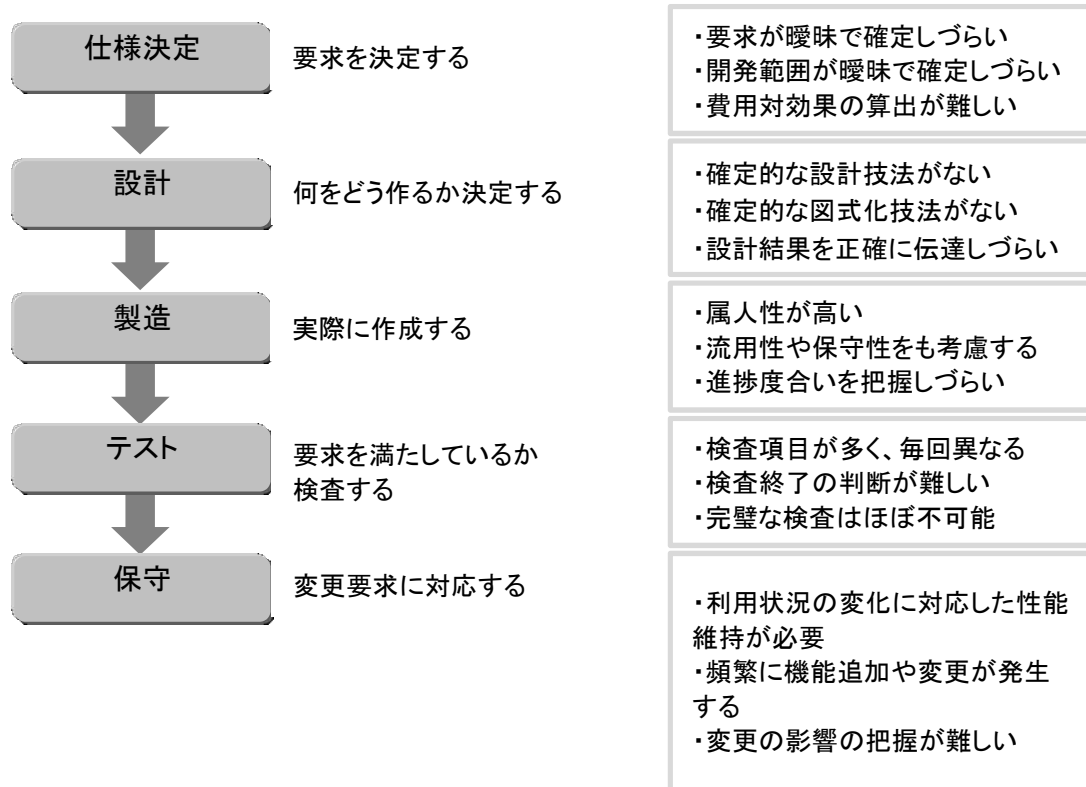


図1-2 一般的なものづくりと比較したソフトウェア開発の難しさ

■ よいソフトウェア開発とは

ソフトウェア開発の成否を評価する視点として、以下のQCDがあります。

- ◆ Q (Quality 品質)
利用者の要求を満たし、欠陥がないこと。
- ◆ C (Cost 原価)
ソフトウェア開発のコストを予算内に収めること。
- ◆ D (Delivery 納期(引渡し))
ソフトウェアを利用者が求めるタイミングで提供すること。

ソフトウェア開発では、利用者と開発者がQCDレベルを事前に共有し、開発者がそのレベルをクリアする取り組みが求められます。

■ ソフトウェア工学の目的と必要性

学術研究や軍事利用など、限られた分野で利用されていたコンピュータは、1960年代に入ると、ビジネスや公共サービスなど、幅広い分野で利用されるようになりました。これにともない、ソフトウェア開発の質・量に対する要求は飛躍的に高まり、ソフトウェア技術者の人員不足やソフトウェアの質の問題が深刻化しました。

この問題を解決するためにソフトウェア工学が誕生しました。ソフトウェア工学の目的は、それまで個人に依存していたソフトウェア開発の技法や手順を体系的に整理し、誰が担当しても、良い品質(Q)のソフトウェアを、低コスト(C)で、短期間(D)に開発することです。

ソフトウェア工学とは

ソフトウェア開発管理工程の全局面において、工学的な技法や方法論を導入し適用することによって、ソフトウェアの生産を質的および量的に向上させていくための実務的学問分野。

ソフトウェア工学が目指すもの

1. 生産性の向上 — 自動化の追及 —
2. 属人性の排除 — 具体的方法論の確立 —

第4章 構造化分析技法

- 4.1 構造化分析とは
- 4.2 DFD
- 4.3 構造化分析の流れ
- 4.4 データディクショナリ
- 4.5 ミニ仕様書
- 4.6 演習問題

この章のねらい

この章では、構造化分析の概要と手順、DFD、データディクショナリ、ミニ仕様書の特徴を紹介します。

4.1 構造化分析とは

ソフトウェア開発における構造化とは大きな機能を細分化し、小さな機能の有機的な集合にすることです。たとえば、顧客からの注文を受け付ける「受注」機能は、「顧客確認」「在庫引当」「受注データ登録」「受注伝票印刷」などの機能に細分化できます。

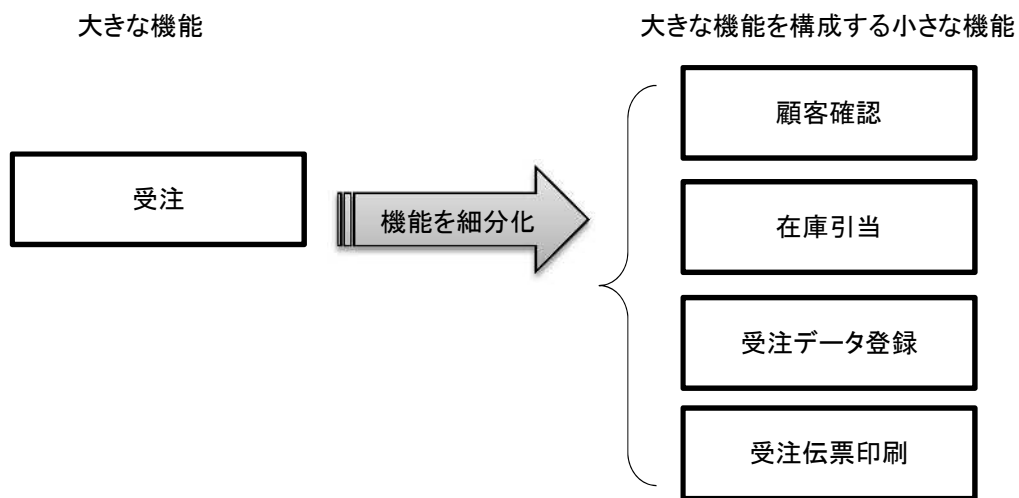


図4－1 機能の構造化の例

構造化分析とは、ソフトウェアの機能をデータの流れに着目して洗い出し、段階的に詳細化しながら、ソフトウェア全体の機能構造を明確化する手法です。

構造化分析の代表的な技法は、デマルコ氏(T.DeMarco)によって提案されたDFD(Data Flow Diagram)、データディクショナリ、ミニ仕様書の3つです。

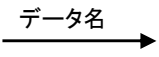

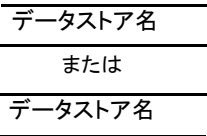
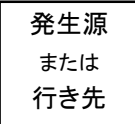
4.2 DFD

DFDはソフトウェアのデータの流れに注目し、機能とデータの間係をわかりやすい図として表現する技法です。

4.2.1 DFDの表記法

DFDは以下の4つの記号を使って記述します。

DFD の記号

記号	名称	意味
	データフロー	データの流れを表します。データの種類を明確にするために、矢印の上または下にデータ名を記述します。
	機能 (プロセス)	データの加工や変換などを行う機能を表します。処理の内容を表す機能名を記述します。
	データストア	データの蓄積を表します。データストアのほとんどはファイルやデータベースで、永続的に保存されることを意味します。
	外部 (源泉／吸収)	対象業務(システム)の範囲外にあり、システムにデータを投入したり(源泉)、システムがデータを出力する対象(吸収)を表します。顧客や他の関連システムなどに相当します。