

図解でわかる

ソフトウェア開発のすべて

第3章 まずは開発計画から

計画・管理の重要性

- ❖ ソフトウェア開発プロジェクトの問題点
 - 技術的, 人間的側面
 - プロジェクトの状況把握
- ❖ 計画の重要性
 - すべき作業、必要な要因が明らかに
 - リスクが事前に検討できる
- ❖ 管理の重要性
 - 進捗状況を目に見える形で提示できる

プロジェクト計画の概要

- ❖ スコープ計画と管理
 - どこまでをシステムが担うか.
- ❖ スケジュール計画
 - WBS(段階的詳細化)
 - 進捗管理
 - スケジュール表
- ❖ リスクの特定と管理
 - 技術的スキルの不足
 - 要求事項の変更
- ❖ プロジェクト計画書の作成

ソフトウェア見積り技法①

- ❖ 概算法
経験をもとに工数やコストを類推する.
- ❖ 積算法
必要な作業を洗い出し、各々の工数を計算.
- ❖ 標準タスク法

(a) 標準タスクAの作業日数

複雑度 規模	単純	普通	複雑
小	1	2	3
中	1.5	3	5
大	2	4	7

(c) プロジェクトXにおける標準タスクAの作業工数

複雑度 規模	単純	普通	複雑
小	1×10	2×5	3×0
中	1.5×10	3×30	5×5
大	2×0	4×10	7×10

(b) プロジェクトXにおける標準タスクAの作業件数

複雑度 規模	単純	普通	複雑
小	10	5	0
中	10	30	5
大	0	10	10

$$10 + 10 + 0 + 15 + 90 + 25 + 0 + 40 + 70 = 260 \text{ (標準タスクAの総工数)}$$

プロジェクトXの総工数
= 標準タスクAの総工数
+ 標準タスクBの総工数
+

見積り技法 -COCOMO

❖ COCOMO(1970年代)

- LOC(Line of Code)で規模を測る.
- 基本, 中間, 詳細COCOMOがある.
- アセンブラCOBOLの頃に使われた.

開発工数 (人月) = $3.0(LOC)^{1.12} \times \text{調整要因}$

開発期間 (月) = $2.5 \times (\text{開発工数})^{0.35}$

見積り技法 -FP法

❖ ファンクションに分割

外部入力, 外部出力, 外部照合

内部論理ファイル, 外部インターフェイス

❖ 複雑度を3段階で評価

ファンクションポイント=基準値×(0.65+調整値/100)

	容易	普通	複雑
外部入力	3	4	6
外部出力	4	5	7
外部照合	3	4	6
内部論理ファイル	7	10	15
外部インターフェイス	5	7	10

- ・処理の複雑性
- ・データ通信
- ・オンライン更新
- ・オンライン更新
- ・分散データ処理

- ・複数サイトでの使用
- ・トランザクション量
- ・性能条件
- ・高負荷構成
- ・エンドユーザーの効率性

- ・変更の容易性
- ・システム運用性
- ・再利用性
- ・インストール容易性

見積り技法 -COCOMO II

- ❖ LOCで規模を推定
- ❖ FPを用いた規模推定にも対応
言語別に変換係数が用意されている.
- ❖ システムの再利用にも対応

$$ESLOC = \frac{ASLOC \{AA + AAF(1 + 0.02(SU)(UNFM))\}}{100}$$

ESLOC：再利用の規模調整

ASLOC：再利用規模

AA：再利用モジュールの評価と同化のための程度(0～8)

AAF：再利用モジュールの修正度

SU：再利用モジュールを理解するための工数増の程度(10～50)

UNFM：再利用モジュールを理解していない程度(0.0～1.0)

要点

- ❖ 事実を正確に把握する
 - ソフトウェアは目に見えない
 - 情報を選び好みしない
- ❖ コミュニケーションを図る
 - 活発なコミュニティは成功
- ❖ 変更を管理する
 - バージョン管理システム
 - 情報交換サイトの更新
- ❖ マイルストーンを管理する