オーム社/雑誌局

ISBN4-274-50068-3

C3055 ¥1524E

9784274500688

1923055015241

定価(本体1524円【税別】)

IPA 独立行政法人 情報処理推進機構
ソフトウェア・エンジニアリング・センター

SEC-TN05-001

PRINTED WITH SOY INK

R100 100%リサイクル用紙を使用しています

SEC BOOKS

ソフトウェア開発見積りガイドブック
～ITユーザとベンダにおける定量的見積りの実現～

独立行政法人 情報処理推進機構
ソフトウェア・エンジニアリング・センター 編

Ohmsha

SEC BOOKS

ソフトウェア開発見積りガイドブック
～ITユーザとベンダにおける定量的見積りの実現～

独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター 編

Ohmsha

はじめに

ソフトウェア・エンジニアリング・センター(以下，**SEC**と略記)は，定量的な見積りに関する重要ポイントのまとめとして，小冊子「**ITユーザとベンダのための定量的見積りの勧め**」(以下，「**見積り小冊子**」と略記)を2005年４月に発刊しました。

本ガイドブックは，見積り小冊子を受けて，システム開発プロジェクトにおけるソフトウェア開発の見積りに焦点を当て，具体的な方法およびノウハウを紹介するものです。構成は大きく二つに分かれており，第１部が「総論」，第２部が「見積り手法の事例集」となっています。

第１部では，見積り小冊子で示した見積り活動に関する基本的な考え方や心得的な内容を実際の活動につなげ，さらに向上を図るための方法について示しています。システム開発プロジェクトにおいて，ユーザおよびベンダ企業の双方の関係者のそれぞれが果たすべき役割の重要性とプロジェクトを成功に導くための協力的な関係の構築を主眼として，第１部では，次の２点を中心に解説します。

① ITユーザとベンダが見積り結果について互いに納得できるために持つべき共通認識について(「第１章　合意できる見積りとは」)
② プロジェクトマネジメントおよび組織の観点からみた，見積り能力の向上について(「第２章　見積り能力の向上」)

想定読者は，**表１**に示すA)～D)のすべてを対象としています。②については，ソフトウェア開発プロジェクトのマネジメント・運営を行う観点から示されている事項も多く，ユーザ企業にとっては，やや詳細すぎる印象もあるかも

表 1

想定読者
A）ユーザ企業のトップマネジメント プロジェクトマネージャ システム部門のメンバ 社内改善メンバ(企画メンバ)
B）ユーザ企業の契約担当者 ベンダ企業の営業担当者
C）ベンダ企業のプロジェクトマネージャ
D）ベンダ企業の社内改善メンバ(企画メンバ)

しれませんが，ユーザ企業にとっても見積り能力の向上は必要であり，その意味で自社の仕組みを見直すための参考にできます。

第 2 部では，見積り活動の事例集として，見積りの具体的な手法(実際に企業で実践されている見積り手法)を紹介し，見積り精度向上に向けての各社の取り組み事例，当該手法の導入に当たっての留意点を示します。第 2 部は，個別の方法に興味のある方が，一つ一つの事例を独立して参照できるように構成しております。

なお，見積り能力とは，精度の高い見積りを組織横断的に再現性高く実施することの度合いを指していますが，見積りにおける精度とは，数学的な意味での精度ではなく，「妥当な」見積りの実現度合いを高めることを指しています。「妥当」であるとは，「マネジメント可能な範囲に見積り値を収める確度を高め

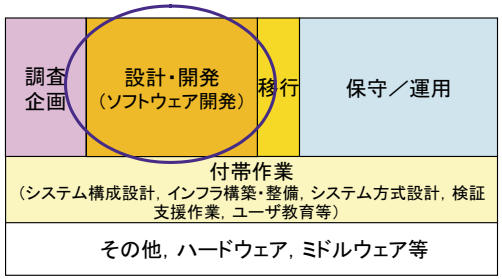


図 1

る」こと，すなわち，プロジェクトの健全な運営が可能な見積りであることを意味しています(第 1 部 第 2 章 2.2.1項を参照)。

本ガイドブックのスコープは次のとおりです。

○ システム開発全般を視野に入れますが，特にソフトウェア開発の見積りの具体的な方法について解説を行います(**図 1** 参照)。

○ ビジネスアプリケーションを中心としたソフトウェア開発を対象とします。ただし，本書で示されたガイドは，組込みソフトウェアなど他の分野でも十分に応用可能です。

○ 見積りとは，規模，工数，工期，品質(信頼性，性能など)，コストなどのさまざまな要素を広く対象とするものです。本書では，特に規模，工数，工期，コストの見積りに焦点を当て，その具体的な考え方および方法を示します。品質は，それぞれの関係に影響を及ぼす要因として，とらえています。

最後に，見積り小冊子でも示したことですが，見積りに当たっての成功要因は，次に尽きると考えております。本ガイドブックでは，その具体的な方法について示します。

- ユーザとベンダは，必要な情報を互いに出しあったうえで，その妥当性を互いに確認しあう。
- ユーザは，見積りを確定するうえで必要な情報を可能な限り示し，ベンダは見積りの根拠を明確にする。
- プロジェクトには，常にリスクがつきものであることをお互い理解し，ユーザとベンダ双方が協力し，リスクを極力早期に小さくする。

2006年 3 月

見積り手法部会　一同

本ガイドブックの読み方

☆ユーザ企業のトップマネジメント，プロジェクトマネージャ，システム部門のメンバ，社内改善メンバ(企画メンバ)

第１部の第１章を読み，プロジェクトにおける見積りの基本事項を理解してください。そして，第２章を参考にして，自組織での見積り能力の向上方法を検討してください。また，第２部の各事例を参照し，自組織の見積りの向上に役立ててください。

☆ユーザ企業の契約担当者，ベンダ企業の営業担当者

最初に，第１部第１章の1.4節の箇所を読み，続いて，その背景として第１章の最初から通読してください。第２部の各事例を参照し，見積りの根拠として示されている考え方を把握してください。

☆ベンダ企業のプロジェクトマネージャ

第１部の第１章から第２章まで通読してください。また，第２部の各事例を参照し，見積りの根拠として示されている内容を活用してください。なお，知識確認として，第１部の第３章をお読みください。

☆ベンダ企業の社内改善メンバ(企画メンバ)

第１部の第１章から第２章まで通読してください。第２章を参考にして，組織的な見積り活動の成熟度向上に取り組んでください。あわせて，第２部の各事例を参照し，自組織との共通点や差異を把握して，実際の見積りモデルの作成や既存のモデルの改善を行うとともに，見積り活動の成熟度向上に役立ててください。なお，知識確認として，第１部の第３章をお読みください。

見積り小冊子(2005年４月発行・本文54ページ)

☆小冊子「ITユーザとベンダのための定量的見積りの勧め」

見積りの重要性について改めて意識を喚起し，見積り精度向上のための考え方，取り組みなどの基本的な内容を概説書としてまとめたものです。ソフトウェア開発にとどまらず，システム全体の開発も視野に入れて論じています。組織の状況に応じた適切な見積り手法があるという意識喚起を行い，見積り方法の考え方を示しております。

本ガイドブックを読む前に，小冊子に目を通しておくと，ガイドブックの全体像がわかりやすくなります。

目　　次

第 2 章　見積り能力の向上

第 3 章　見積り手法の一般的事項と今後の課題

第 2 部　見積り手法の事例集

第 2 章　野村総合研究所手法

第 3 章　TIS手法

第 4 章　日立システムアンドサービス手法

第５章　日立製作所手法

第６章　ファンクションスケール法(富士通)

第７章　日本ユニシス手法

第８章　日本IBM手法

第９章　ジャステック手法

SECでの見積り手法に関する実証実験

第１章　CoBRA法

第２章　EASE協調フィルタリング法

目　次

第1部
総　　論

第1章　合意できる見積りとは

1.1　見積りの実情と課題

システム開発の見積りにおいては，過去から変わることなく，当初に想定した見積りと実績に大きな乖離があり，結果としてプロジェクトが予定期間に予定コストで終了しないばかりか，品質的にも問題があるといった例が存在します。昨今のソフトウェア技術や開発技術の進展など，見積りに当たって考慮しなければならない開発に対する影響要因も増えており，その状況が顕著になっています。特に，ユーザ企業におけるシステムが現行業務の電子化といった単純なシステム化から，ビジネス戦略に基づいたシステム化という位置付けに変わったことにより，見積りの重要な根拠であるシステムの要件を決定しづらくしたり，ビジネスシステムの展開までの時間(Time to Market)の短縮，新規技術の導入や他システムとの連接といったリスク要因が増加したりして，見積りのむずかしさに拍車をかけています(**図1.1**参照)。

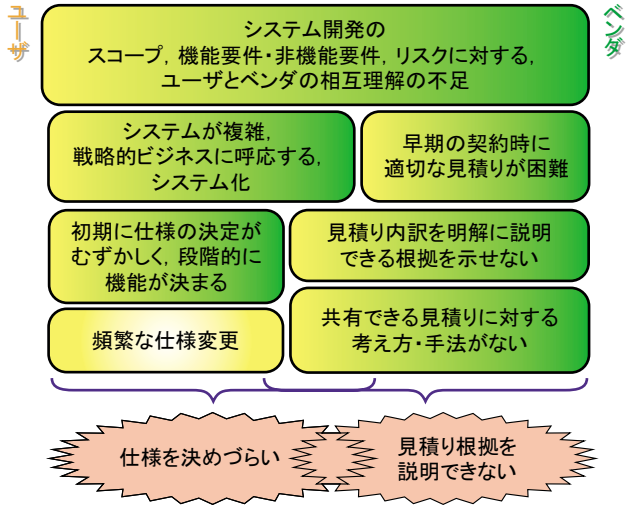


図1.1　見積りを取り巻く状況

1.1.1　要求・要件があいまいな状況での見積り

システム開発プロジェクトにおいては，「システム化の方向性」の段階など，ソフトウェアのライフサイクルでいえば，早期の段階で見積りを行う必要があります。特に，ユーザでは予算の概算のために必要とされます。しかし，この段階では，人事システム，給与システムまたは配送制御システムなど，どのようなタイプのシステムを構築するかといったレベルの一般的な情報にとどまり，要求・要件の詳細は不明確にならざるを得ません。

また，ビジネス戦略に直結したシステムの場合，どのようなシステムを構築すればよいか自体が不明確な場合も多く，ユーザ自身が十分にシステムのイメージを持っていない場合もあります。また，そのような場合は，ベンダ側でユーザの業務に関する知識不足を補うなどの必要が生じてきますが，不明確なシステムイメージをユーザから引き出すには，ベンダ側に，より高度な知識・スキルが要求されることになるため，ユーザ要求を引き出すに至らず，要件を確定できないといった状況も生まれます。

このような状況から，要求・要件があいまいなままプロジェクトを進めることにより，設計・製造段階にいたっても仕様変更が頻繁に発生して，見積りの前提が変わり続ける状況が続く場合も少なくありません。

いずれの場合でも，仕様があいまいな状況で見積り，予算を定められ，それに縛られ，後で確定する仕様と見積りが合わなくなるといった結果になりがちです。

1.1.2　ユーザとベンダとの間の納得感の欠如

システム開発においては，実現する機能を明確にすることに加えて，要求品質といった非機能要件やプロジェクトに対する制約条件も考慮しなければなりません。一方，最初に述べたとおり昨今考慮しなければならない事項・条件が多岐にわたり，複雑なものになっています。そのため，関係者の間でシステム開発の見積りの「根拠」をお互い理解できない，または，そもそも見積ろうとしている対象(見積りのスコープ。1.3.1項参照)について，見ているところが違うといった相互の理解において，くい違いが生じる例が増加しています。たとえば，ユーザが見積りに「移行」の費用が含まれていると考えているのに，ベンダ側ではソフトウェアの「開発」だけが対象と考えているといった例は，珍しくありません。

このような状況の背景には，見積りの根拠や説明の理解において，双方でコミュニケーションが，十分に取れていない問題があります。はなはだしい場合は，ベンダが開発全体一式の価格のみを提示するにとどまり，ユーザはその妥当性を判断する術(すべ)を持たないという，コミュニケーションの前提となる情報提供が，ほとんど行われない場合もあります。

ユーザとベンダが，相互に納得できる観点から見た課題をまとめると，次のとおりです。基本的には，システム開発に関する理解およびコミュニケーションに関するものです。

（1）ソフトウェア規模の尺度に関する「ことば」の共通理解

例：ファンクションポイント(FP)，ソースコード行数，ユースケースなど。それぞれに基づいた見積り結果が，何を表しているのかを，共通に理解できることが必要となります。

（2）ソフトウェア開発にかかわる「影響要因」がコストへ及ぼす影響についての認識の共有，抜けの防止

高い品質のシステムは，価格が高くなるなど，システムの信頼性，セキュリティ，複雑度，先進性などの要件やユーザ・ベンダ間の意思疎通などの要因により，通常の水準との差に応じて，コストが上下することを関係者で共有する必要があります。そして，考慮しておくべき事項に抜けがないか，関係者で確認することが必要です。さらに，このような影響要因について，コントロールできるのは誰か，関係者のお互いの役割は何かを，共通に理解しておく必要があります。

（3）見積りにおける不確実な要素をプロジェクトにかかわるリスクとして関係者で共有し，相互協力によるコントロール

プロジェクトにかかわるリスク(要求内容の不確実性，新規技術の導入リスクなど)は，プロジェクトの関係者で共有し，プロジェクトの破綻をきたさないように協力しあう必要があります。

（4）システムの価格の評価

現在，製作コストに対応する対価は，工数を指標とする事例が多く見られますが，そうした価格の決め方に対し，古くから疑問の声があがっています。

システムがもたらす付加価値(経営的効果)に基づいて設定するなど，脱工数を目指した方法も実践されてきていますが，まだその方法が確立しているとはいえません。

システム価格の評価においては，よりよい方法の模索段階であることを理解

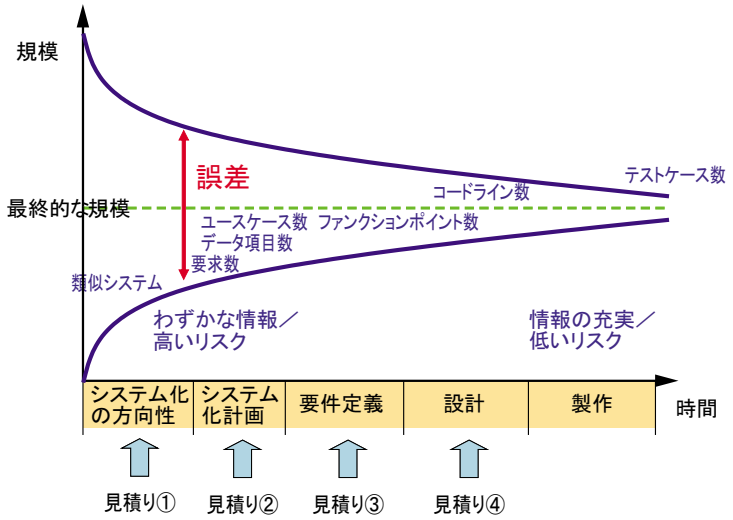
し，関係者がそのことを意識した上で，お互いに納得のできる価格の決め方を合意することが重要です。

次項以降では，このような状況の中で，ユーザおよびベンダの関係者がどのように見積りに取り組むべきなのかを示していきます。

1.2　見積りにかかわるリスク

1.2.1　見積り時期とリスク

見積りを実施する際に，最初に問題となるのが見積り時期です。見積り時期が早ければ早いほど，プロジェクト全体に不明確な点が多く存在し，最終的な規模と見積り規模との「ぶれ」幅は大きくなります(1)。不明確な部分があると，どうしても実際にできあがるものから比べて，見積りの「ぶれ」が出てしまうことは避けようがありません。一方，早い時期での見積りは，予算獲得などのおおよその計画を立てるために必要であることも事実です。解決策は，「わから



（注）文献：Barry Boehm著の“Software Engineering Economics（Prentice-Hall社）”の図に基づきSEC作成

図1.2　見積り時期とリスク

(1)　Boehmによると1/4倍～4倍の範囲に散らばる（図1.2の左端で）との例を示している。

ないものはわからないもの」としてリストアップし，さらには不確実性の評価（どの程度のぶれ幅になるのか）を行って，関係者の間で共有しておき，適切なタイミングで見直すことが必要です。

特定された「わからないもの」は，見積りが確定していない範囲としてユーザとベンダとで相互に認識を合わせておき，確定するまでモニタして双方でコントロールすることが，見積りの妥当性を確保するための基本的な対策となります。不確実なもののコントロールは，関係者すべての協力が必要ですが，特にユーザが最終的な判断の鍵を握っているので，これらの活動においてユーザがリードすることが望まれます。

1.2.2　プロジェクトの特性による見積りへの影響要因

見積りにおいては，開発するシステム（プロダクト）だけでなく，プロジェクトの前提条件，プロジェクトにかかわる組織（ユーザとベンダ）の特徴，その組織内のプロセス，さらには開発チームや個々の開発メンバの特性など，さまざまな要因を考慮する必要があります。特に，ソフトウェア開発は自動化されている部分が少ないため，開発は人的な要因から大きな影響を受けます。**図1.3**

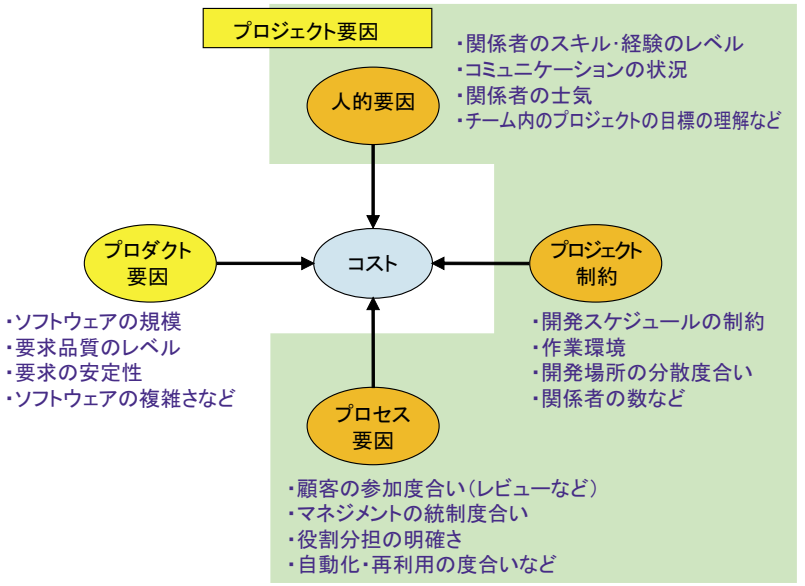


図1.3　見積りへの影響要因例

に影響要因例を示します。

「プロダクト要因」は，要求される品質，要求の安定性，複雑さなどの作成するシステムの特性に直接関係します。コストに大きな影響を及ぼす要因として，システムの規模もプロダクト要因のひとつとして挙げることができます。

一方，システムそのものではなく，システムを開発するプロジェクトの特性による要因は「プロジェクト要因」としてまとめられます。「プロジェクト要因」は，プロジェクトにかかわるメンバやチームの状況にかかわる要因(「人的要因」)，プロジェクト遂行に当たっての制約(「プロジェクト制約」)，プロジェクトのマネジメントの統制度合いなどのプロセスの状況にかかわる要因(「プロセス要因」)の３つに分類できます。

これらの影響要因のレベルも，機能要件などと同様，ユーザとベンダなどの関係者で妥当なものに調整でき，結果的にはリスクや開発コストの増減をコントロールできます。逆に，プロジェクトの影響要因とリスクを関係者が共有していないと，それらをモニタし，プロジェクトをコントロールすることができません。本ガイドブックで示す内容は，ユーザとベンダなどの関係者が，相互に協力して取り組んで，効果が上がることを念頭においてください。

では，これらのリスクや影響要因をどのように把握し，妥当な見積りを実現すればよいのでしょうか。次項では，そのガイドを示します。

1.3　妥当な見積り

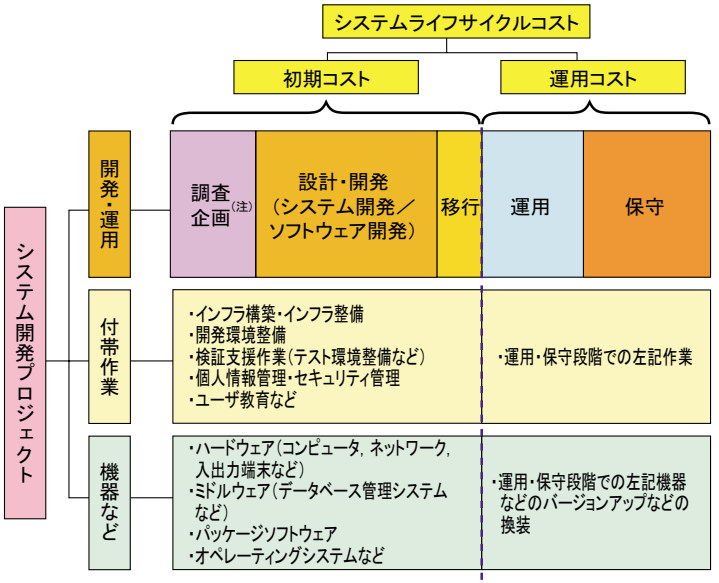
1.3.1　見積りのスコープの共有

見積りに当たってユーザとベンダ間で最初にすべきことは，システム開発にかかわる成果物および活動のうち，どの範囲を見積るのか，見積りのスコープを明確にし，あらかじめユーザとベンダとで，共通に認識しておくことです。

システム開発プロジェクトは，システムの「開発・運用」(ソフトウェアの開発・運用を含む)を中心として，「機器など」の調達，開発・運用を支援するものとしての「付帯作業」から構成されます(**図1.4**に例示)。

(1)　システムライフサイクル全体の明確化

「システムライフサイクル全体の明確化」とは，システムのライフサイクルを通じて，必要となる作業を明確にすることです。すなわち，図1.4に示す開発・運用および付帯作業の構成要素のうち，何を対象に見積っているのかをユーザ



（注）システム化の方向性から要件定義

図1.4　システム開発プロジェクトの構成要素

とベンダでの認識のずれをなくすことが必要です。

(a) 調査・企画　システム導入の効果検討，最新技術の調査，システム機能検討など。

(b) 設計・構築　システム要件に基づくソフトウェア機能の設計，モジュール設計，コーディング，テストなど，システム構築に関連する活動。

(c) 移　行　システムの拠点への展開や切り替え，それに伴うデータの切り替えなど。

(d) 保守・運用　顧客に納品されたシステムの機能改善(追加・改良)，バグ修正，是正処置などの活動やシステムの定常的運用，障害対応などの活動。

なお，上記の作業は，誰がその作業を行うのか，担当者をはっきりさせなければなりません。たとえば，調査・企画はユーザ，ベンダのどちらに作業責任があるのか，データ移行の主体として，どちらが主導をとり，どちらに責任があるのか，といった点です。この内容は，契約や仕様書などで明確にされる必要があります。

（２）　見積り対象システム要件の明確化

「見積り対象システム要件の明確化」とは，端的には，「何を，どのような条件で構築するか」，という点を明確にすることです。その内容をまとめて**表1.1**に示します。

対象システムの明確化には，システムの機能範囲，システムの機能要件，品質要件などの非機能要件の明確化があります。これは，第１章 1.1.1項の（１）で示した要求・要件の明確化にほかなりません。第１章 1.3.3項では，ソフトウェアの要求・要件をソフトウェアの規模として把握する方法の詳細を示します。

表1.1　見積り対象システム要件の明確化

分　　類			概　　要
機能範囲			システム化する範囲とシステム化しない範囲の境界の明確化
機能要件			システムに要求される機能
非機能要件	品質要件		システムに要求される品質
	技術要件	ハードウェア	ソフトウェア開発に必要な機器，納品後の稼働環境に必要な機器，通信機器など
		ソフトウェア	ソフトウェア開発に必要なパッケージソフト，オペレーティングシステム（OS），ツールなど
	その他の要件		運用・操作要件，移行用件，付帯作業など

1.3.2　見積りの手順

システム開発では，図1.4に示したような構成要素に対して，それぞれの規模や作業量などを見積ることになります。ここでは，ソフトウェア開発に絞って定量的な見積りの手順を示します。ソフトウェア開発以外の付帯作業および機器などは，過去のデータに基づいて，類推法または作業などの積み上げによる方法が一般的です(2)。

なお，本ガイドブックでは，製作コストの観点からソフトウェアのコストをとらえています。

（１）　ソフトウェア開発における見積りの基本形

見積り方法には，さまざまなものがありますが，基本は**図1.5**に示す手順で

(2)　第１部 第２章 表2.1参照(p.51)。

行います。図に示すとおり，見積りの前提として，まず要件の洗い出しがあります。要件に基づいて，成果物の規模を見積ります。次に，工数と工期を見積ります。工数は，規模と最も大きな相関関係にあるとして，なんらかのベースとなる関係式を用いて，規模から工数を見積ります。また，同様に工数と工期に大きな相関関係があると考えて，関係式を設定し，工数から工期を見積ります。

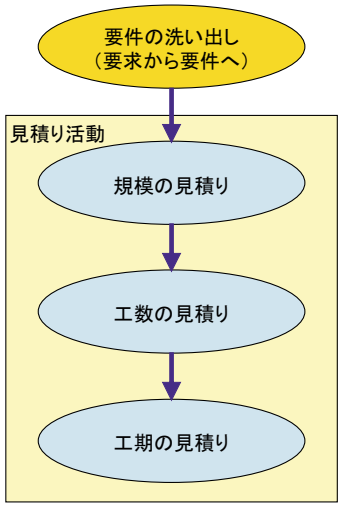


図1.5　見積りの基本的な手順

なお，図1.5では，先に要件が決まる場合の手順を示していますが，実際には，工期が最重要な制約条件となる場合もあり，そのときには工期から可能な工数を得て，さらに構築可能な規模を見積る手順になります。同様に工数が制約条件となり，規模や工期を見積る場合もあります。いずれにせよ，要件，規模，工数，工期の間に，なんらかの関係をあらかじめ設定しておいて，いずれかから他のものを見積ることが基本となります。

（２）　見積りの個々の活動

図1.6は，図1.5の具体的な内容を示したものです。図に示すとおり，要件の洗い出しは，機能の要件と非機能要件（品質要件や技術要件など）を定義することを対象とします。ここで洗い出した機能要件および非機能要件は，規模の見積り，工数の見積りの入力になります。

機能要件は，システムがユーザに提供する機能を指します。そのまま直接システムの規模を見積るための入力となります。

一方，非機能要件は，工数見積りにおける生産性の変動要因として評価します。

非機能要件(品質要件や技術要件など)は，機能を実現するに当たっての目標値として設定するものです。たとえば，高い信頼性を要求されたり，高い性能を要求されたりする場合は，その要求を実現するために通常よりも多くの工夫や工数を必要とし，その分，生産性を下げることになります。品質要件の例として，信頼性要件や性能要件があります。具体的には「365日24時間システムをストップすることなく稼働すること」(信頼性)や，「レスポンスタイムを３秒以内とする」(性能)というように規定されます。技術要件の例では，システムの実現方式などがあります。これらは，個々のシステム開発において，要求されるレベルを通常のレベル(類似のシステムの要求レベルで最も頻度の高いもの)と比較し，個々のシステム開発における実現可能な生産性を評価するために用います。

なお，品質要件や技術要件などを「規模」として評価・換算する見積り手法もあります(第１章 1.3.3項(７)参照)。

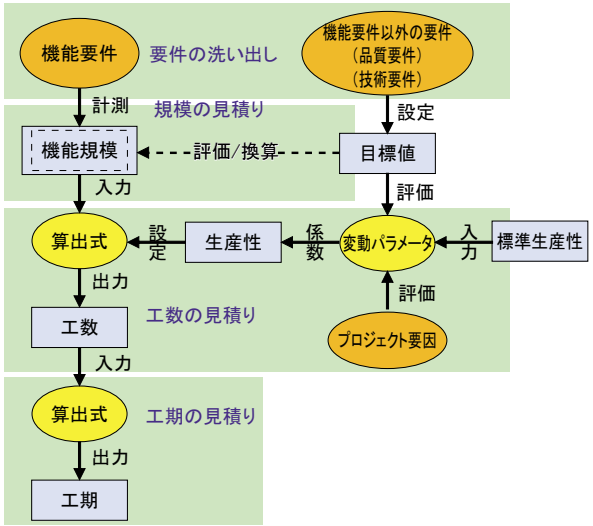


図1.6　個々の見積りの基本手順のブレークダウン

(３)　ユーザとベンダのそれぞれの役割

上記の個々の活動に対して，ユーザおよびベンダのそれぞれの役割があります。

まず，ユーザはシステム開発でのすべての意思決定の主体であり，機能要件や非機能要件の内容は，ユーザが決定します。図1.6に示すとおり，機能要件の内容や品質要件・技術要件などは，システムの規模や開発の生産性を決定します。したがって，システムの規模や開発の生産性は，ユーザがコントロールできます。機能要件，品質要件，技術要件などを取捨選択し，その内容のレベルを調整することによって，最終的な工数またはコスト(ひいては価格)の低減や工期の短縮を図ることができます。

逆に，ベンダ側は，そのような要件をユーザが判断・確認することをシステム開発のプロフェッショナルとしてサポートする必要があります。これは，ユーザすべてがシステム開発に慣れているわけでないことが背景にあります。例えば，ユーザにとって，システムの技術的難易度など，非機能要件のうち，システム構築の知識がないと判断できないものは，ベンダのサポートが不可欠です。

サポートには，図1.6に示すような見積りの手順，機能要件や非機能要件の内容とそれぞれの見積りへの影響を説明することが含まれます。つまり，手順と必要な情報を示すことにより，ユーザの意思決定を支援します。また，機能要件や非機能要件に漏れがないかといった確認も重要な支援のひとつです。システム開発では，ユーザとベンダとが共同して作り上げていく点を改めて認識し，見積りも互いの考えについて，コミュニケーションをとることが重要です。

なお，見積りに関して，ベンダ側の説明能力の向上だけでなく，ユーザが見積りの妥当性を判断する能力の向上も，今後ますます必要になると考えられます。システム開発を低コストで無理なプロジェクトにすることなく，また，高コストでむだな投資をしないためには，ベンダの能力のみに頼るのではなく，ユーザ側でも妥当性が判断できなければなりません。システム調達のリスクを軽減しつつ，必要なシステムを適正な価格で入手するうえで，ユーザにとっても見積り能力の向上は重要な課題です。

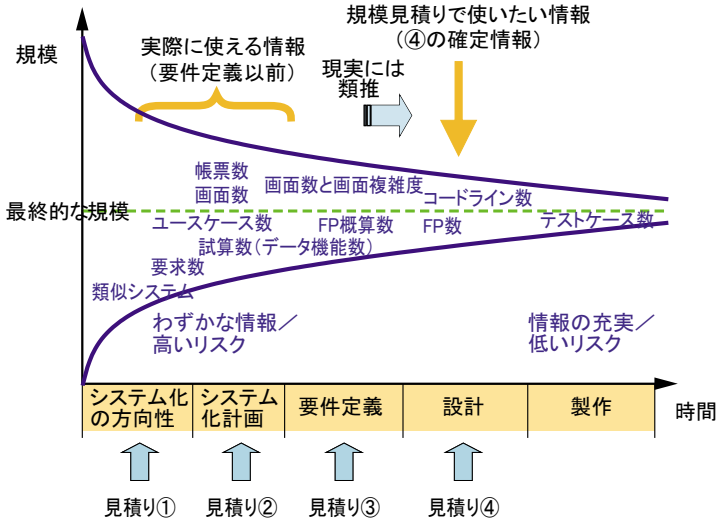
見積り能力の向上については，第２章で詳細を述べます。

以下の各項では，規模の見積り，工数・コストの見積り，工期の見積りについて詳細を示していきます。

1.3.3　規模見積り

（１）　必要な情報と実情

規模見積りを行う場合，どのようなシステムを構築するか(機能要件，品質要件，技術要件など)が確定した情報に基づいて見積ることが理想です。**図1.7**でいえば，見積り④の設計段階での情報の利用が望まれます。



FP：Function Point

図1.7　見積り時期と得られる情報例

しかし，予算の確保や計画実行の可否判定などのために，「システム化の方向性」の段階で見積るコストは，ユーザ企業にとって重要な判断材料であり，見積り④の段階まで待つわけにはいきません。一方，システム開発の早期の段階で見積るため，必然的に不確実な要素が残ることを避けられず，見積り結果の利用に当たっては，そのリスクを常に念頭におく必要があります。

（２）　見積り時期と利用可能な情報との関係

では，早期の段階での見積りは，どのようにすればよいのでしょうか。

ソフトウェアの開発方法によって，早期に比較的高い精度で得られる情報があります。例えば，データ中心アプローチでソフトウェアが開発される場合では，データに関連する情報が最初に確定します。一方，Web系のシステムでは，

画面からユーザと詰めていくことも行われ，その場合には，画面に関する情報が比較的早い時期に確定します。

このようにシステム開発の対象や方法に応じて，早期に確定する情報があります。少しでも早い時期に見積るためには，それらの情報を利用して，見積ることが必要となります。以下に，その方法をいくつか示します。

（３）　早い段階での規模の推定方法

(a)　データモデルから全体規模の推定　まず，データモデルから最終的な全体規模を推定する方法があります。開発の早期の段階のデータモデルからデータ項目の数などを読み取り，その数に基づいて最終的な規模をファンクションポイント(Function Point。以下，FP)で推定する方法です。

FPは５つの機能種別(ILF，EIF，EI，EO，EQ)を計測して規模を算出する尺度です(第３章 3.1.1項参照)。一般的なエンタープライズ系のシステムであれば，最終的な規模(FP)に対して，FPの５つの機能種別のうち，データに関する機能(ILFとEIF)が，類似のシステムの間で，ある程度一定の割合となることが，経験的に分かっています。逆にいうと，FPを尺度にして，規模の見積りを行う場合に，FPの各機能(ILF, EIF, EI, EO, EQ)のそれぞれが，おおよそどのような割合を示すかを，あらかじめ過去のプロジェクトデータから求めておけば，④の段階のFPを早い段階で推定ができます。この考え方は，NESMA試算法と呼ばれる手法(第３章 3.1.1項(4)参照)でも利用されています。

データ中心アプローチでは，データ分析を中心に要件を洗い出していくことから，データ項目が早期に確定します。例えば，「E-Rダイヤグラム」が早い時期に作成され確定する場合，当該ダイヤグラムのエンティティの数から，FPのデータファンクション(ILF，EIF)を求めることができ，さらに，全体のFP

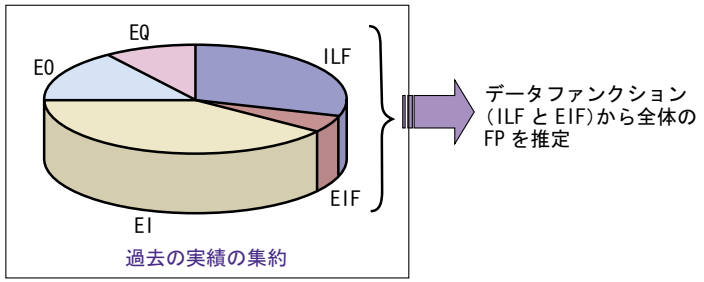


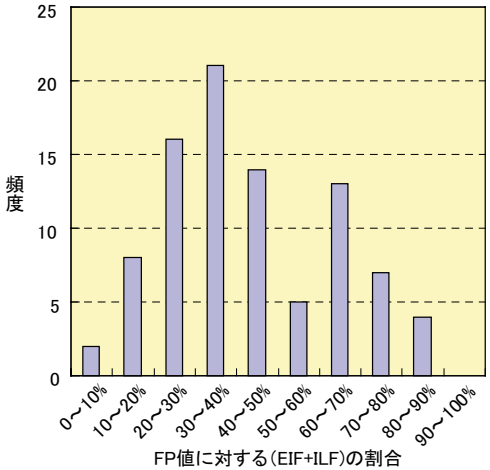
図1.8　FP値の推定例

を推定できます。

なお，機能種別の構成比率は，企業やプロジェクトの特徴(例：適用業種(金融，流通，製造)など)で違いが生ずる可能性があり，機能種別構成比率を規模推定に使用できるか否かの分析を組織的に行っておく必要があります。

コラム：SECデータベースを使った推定例

データモデルから全体FPを推定する方法の具体的なイメージをつかんでもらうためにSECデータベース(3)の例を示します。SECで収集した実績プロジェクトデータからデータファンクション(ILFとEIFの和)が全体FPに占める割合を求めると，平均は43%です。つまりこの例では，ILFとEIFの和を0.43で割れば，全体のFPが推定できます。ただし，ここで重要なことは，あくまで要件の一部の情報を使っての推定なので，誤差が含まれる点です。実際，**図1.9**に示すとおり，上記の割合は，30～40%の場合の頻度が一番高くて両側に裾を引き，広がった分布になっています。



(備考) 20～30%は，「20%以上30%より小さい」を示しています。
(出典) SECデータベース(2005年度版)の90件のプロジェクトデータ(新規開発でFPおよびその各機能のデータがあるもの)に基づく。

図1.9　FP値に対する(EIF+ILF)の割合の分布

図1.10には，データファンクションから全体FPを見積るために，上記の係数0.43を割合として採用した場合のFP値の推定誤差の分布を示します。この場合の平均誤差は40%弱です。これから示唆されることは，例えば概算として1 000FPと見積った場合，最終的には平均的に1 400FPになる可能性があることです(4)。

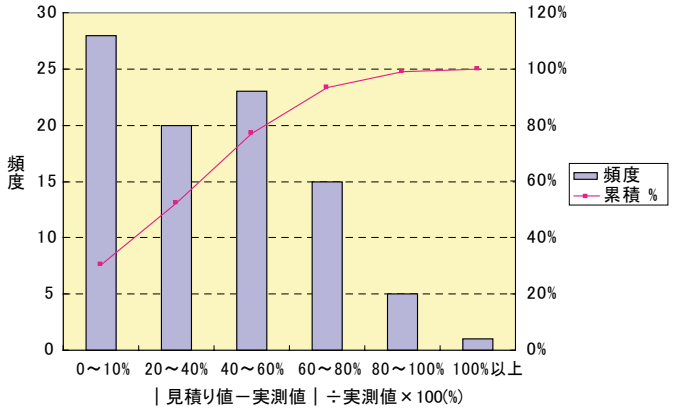


図1.10　FP値の推定誤差例

なお，図1.10の折れ線は累積度数の全体に占めるパーセンテージを示していますが，これからある特定の誤差の範囲内に収まる可能性を計算できます。例えば，最終規模が6割の誤差以内に収まる可能性は約80%です。逆に，見積り値を超えてしまう確率を20%に抑えたい場合は，見積り値に対して6割のぶれをリスクとして上乗せするという規模の設定ができます(1.6倍を見込んでおけば，その値を超える確率は20%に抑えられるといいかえられます)。

このように，組織的にデータをとって分析しておけば，早期の段階での見積りとその誤差および誤差内に収まる確率を予想することができます。なお，分布に対する精査として，さらに分布の広がりを狭めるように分析を進めることが考えられます。例えば，上記の例で分布に山が2つあると判断して要因を探り，2つに分けて割合を分析しなお

(3)　2004年度にベンダ企業15社から1 009件のプロジェクトの実績データを収集したもの「ソフトウェア開発データ白書2005」参照。
(4)　反対に600FPになる可能性もあります。

すなどの方法がとれます。いずれにせよこの方法はあくまで概算なので，上記の事項に留意して，見積り値には誤差が避けられないことを認識したうえで利用することが重要です。

(b)　画面に関する情報からの工数や全体規模の推定方法　画面数からの推定は，ソフトウェアの全体規模または工数(コスト)に対して画面の数や複雑度などが高い相関を持っていることに着目し，あらかじめ過去のデータに基づき関係を設定しておき，利用するものです。

SECデータベースの例からも，規模の単位の一つであるFPと画面数とは高い相関があるとの結果が得られています。特に，Webが中心のシステムでは，この相関が高い結果が得られています。また，規模の尺度はFPばかりではありません。過去データとして，ソースコード行数(SLOC：Source　Lines　Of Codes)を蓄積している場合は，規模としてFPではなく，SLOCを採用し，SLOCと画面数との関係を見ることができます。

この推定方法は，データ項目からの場合と同様，システムの部分的な情報から全体規模を簡便に推定しようとするものです。

画面数がわかっている段階では，画面数からFPに相当する指標値を推定し，画面の詳細(複雑度)がわかった時点で，画面上にあるボタンやリストボックスなどのオブジェクトをカウントすることで，FPとの相関を高めようとするものです(**図1.11**参照)。詳細は，第2部 第6章を参照してください。

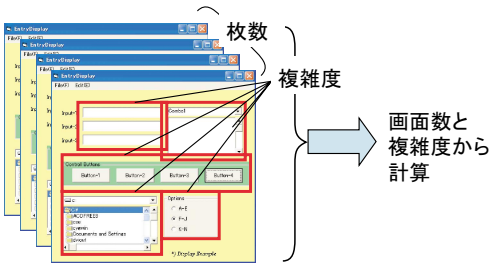


図1.11　画面からの規模推定

(4)　確定していない部分の把握と認識共有

以上，詳細な情報が得られていない時点で，部分的な情報から最終的な規模を見積る例(規模推定方法)を示しましたが，本項の最初でも述べたとおり，不

確定な部分があるので見積りには誤差が避けられません。

そこで，次の点についてユーザとベンダとの間で認識共有しておくことが重要です。

- どの情報に基づいた概算であるか
- 確定していない部分が何であるのか
- 確定していないことによる発生する誤差範囲
- 確定していない部分が確定する時期
- 誤差が発生した場合での対処の取り決め(再見積りの実施，機能の縮小・変更など)

これらを合意した上で，プロジェクトの進行中，不確定なもの(確定したものの変化を含む)をモニタし，必要に応じてコントロールする必要があります。

(5)　要件の追加・変更の共有，モニタリングおよびコントロール

要件の不確実性とともに，規模が変動する大きな要因として要件(機能要件，非機能要件)の変更があると，でき上がり規模が変動するとともに，作成したものを棄てる(手戻る)ことになり，コストに影響します。また，たとえ結果的にでき上がり規模が変動しなくても，棄却・追加が発生し，コストに影響を及ぼす場合があり注意が必要です。要件変更に関してユーザの判断として次の事項を確認することが必要です。

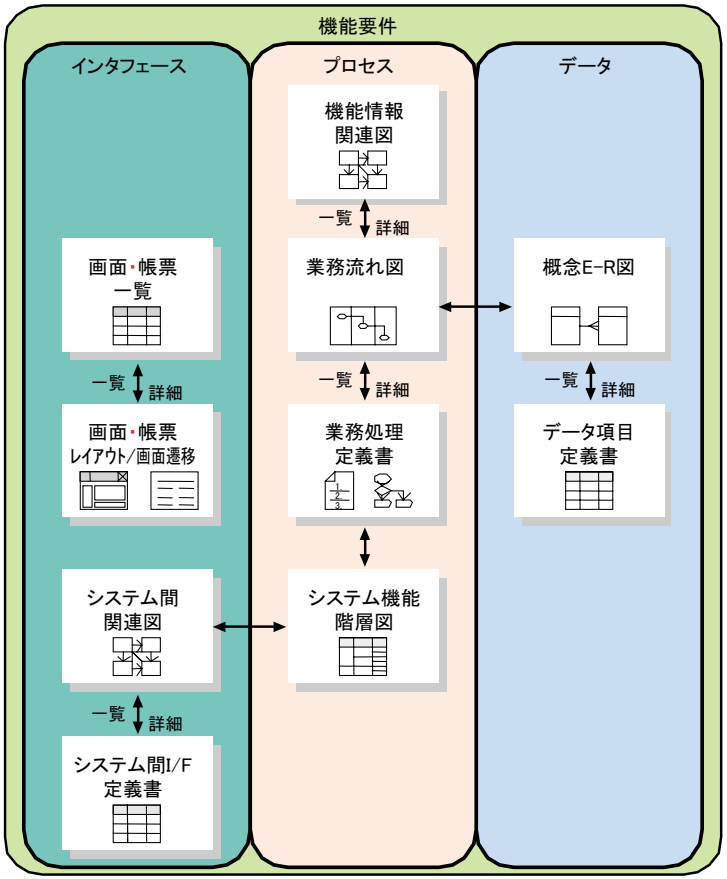
(a)　要件変更の基点　提案書，企画書，要件定義書など，変更の基点となる情報を明確にします。また，あわせて変更管理を開始する時期(例：要件定義書の検収日など)を決定します。

(b)　要件変更として扱う条件　要件変更に関するユーザ側の責任，ベンダ側の責任を整理して，ユーザ側の責任の要件変更を対象とします(切り分け基準，双方の交渉窓口，交渉タイミング，最終承認者なども決めておきます)。

(c)　要件変更の量(追加量，棄却量)と反映する時期　要件変更を反映する時期(設計段階，テスト段階など)により，手戻る対象の成果物が異なり，影響するコストが異なるので，どの時期の変更かを確認して，コストなどへの影響を合意します。

(6)　システム要件として決めるべき事項

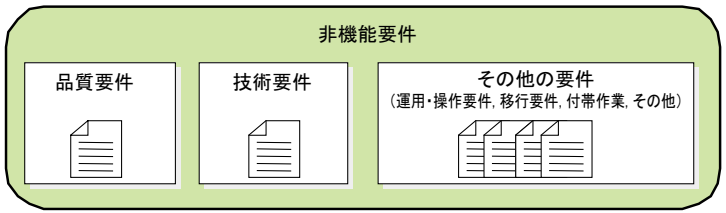
見積り④で確定しているべき事項は，**図1.12**および**図1.13**にまとめるとおりです。これらは，小冊子「**経営者が参画する要求品質の確保**」に基づいたものです。図1.12は，どの部分が明確になっているのかをチェックする際に利用でき



E-R：Entity-Relationship　　I/F：Interface

（出典）小冊子「経営者が参画する要求品質の確保」に基づき作成

図1.12　機能要件の全体像



（出典）小冊子「経営者が参画する要求品質の確保」に基づき作成

図1.13　非機能要件の全体像

ます。例えば，「システム化の方向性」の段階では，どんなシステムを構築するのかといったレベルで，図の一番上にある機能情報関連図がわかっている程度でしょう。段階が進むにつれて，Web系であれば画面に関係する情報が確定していく（図の左列），また，データ中心アプローチであれば，データに関係する情報が確定していく（図の右列）ことになります。図1.12の箱の概要を**表1.2**～**表1.4**にまとめます。図1.13に示す非機能要件の概要は，**表1.5**にまとめます。詳しくは，小冊子「**経営者が参画する要求品質の確保**」（平成17年4月，オーム社刊）を参照してください[(5)]。

表1.2　機能要件（プロセスにかかわる情報）の概要

名称（大項目） / 名称（小項目）	内　容	具体的事項
機能要件（プロセス）		
機能情報関連図	業務機能間の情報（データ）の流れを明確にする	業務機能間を流れる情報（データ）と向き（方向）を関連図で表現する
業務流れ図	業務がどのような組織・手段・手順で処理されるかを明確にする	業務処理機能，業務処理担当部署（担当者），処理手段（手作業・コンピュータ作業），処理手順（流れ）をフロー図で表現する
業務処理定義書	業務流れ図上の各業務処理機能の内容を明確にする	業務処理機能ごとの ・インプット ・プロセス（業務処理のルール，手順，内容など） ・アウトプット をHIPOもしくはフローチャートなどで定義する
システム機能階層図	業務機能を実現する情報システムの機能を明確にする	情報システムの機能を大分類，中分類，小分類の階層で定義する

HIPO：Hierarchical Input Process Output

（出典）小冊子「経営者が参画する要求品質の確保」に基づき作成

（5）　さらに，小冊子を発展した成果として，要件として定義しなければならない内容の具体的な例について，2006年度にSECから提供する予定です。

表1.3　機能要件(データにかかわる情報)の概要

名称(大項目) / 名称(小項目)	内　　容	具体的事項
機能要件(データ)		
概念E-R図	情報システムにおける概念レベルのデータ構造を明確にする	情報システムを実現するためのデータ構造をエンティティとリレーションシップで表現する
データ項目定義書	データ項目の属性を明確にする	エンティティごとにデータ項目名，説明，データ型，けた数，編集方法（コード参照など）を一覧表で定義する

E-R：Entity-Relationship

（出典）小冊子「経営者が参画する要求品質の確保」に基づき作成

表1.4　機能要件(インタフェースにかかわる情報)の概要

名称(大項目) / 名称(小項目)	内　　容	具体的事項
機能要件(インタフェース)		
システム間関連図	検討対象のシステムと，既存システムまたは周辺システムとのデータの流れを明確にする	各システム間で受け渡されるすべてのデータの流れをフロー図で表現し明確にする
システム間インタフェース定義書	検討対象のシステムと，既存システムまたは周辺システムとのデータのやり取りを明確にする	各システム間で受け渡されるデータごとに以下を明確にする ・主なデータ項目 ・データ量 ・受け渡し手段，媒体など
画面・帳票一覧	検討対象のビジネス機能で必要となる画面・帳票を業務フローごとに洗い出し，画面・帳票一覧として整理し，基本的なビジネスデータの所在を明確にする	画面・帳票ごとに以下を明確にする ・画面・帳票名 ・利用目的 ・利用者 ・様式 ・利用頻度など
画面・帳票レイアウト	各画面・帳票のレイアウトサンプルを集め，整理し，基本的なビジネスデータを収集することで，画面・帳票を処理する業務システムの設計条件を明確にする	既存の画面・帳票については，修正の有無をレイアウト上で明確にする。新規の画面・帳票については，レイアウトサンプルを作成することが望ましい
画面遷移	各画面での入力や選択に応じた画面間の切換えなどの制御を明確にする	全画面の体系と相互の関連および切換えについて階層図やネットワーク図などで明確にする

（出典）小冊子「経営者が参画する要求品質の確保」に基づき作成

表1.5　非機能要件の概要

名称(大項目) / 名称(小項目)		内　　容	具体的事項
非機能要件			
品質要件		システムに対する品質に関する要件	以下の項目(JIS X 0129による)に対する目標値 ・機能性(相互運用性，セキュリティ，標準適合性など) ・信頼性(障害許容性，回復性など) ・使用性(理解性，習得性など) ・効率性(時間効率，資源効率：レスポンスタイム，資源使用量など) ・保守性(解析性，変更性など) ・移植性(環境適応性など)
技術要件		ソフトウェアの開発，維持管理，支援および実行のための技術・環境に関連した要件	例えば，以下の項目 ・システム実現方法 ・システム構成 ・システム開発方式(言語など) ・開発基準・標準 ・開発環境
その他の要件	運用・操作要件	安定したシステム運用を行うための，検討対象のビジネス機能を実行するシステムについての運用要件と操作要件	例えば，以下の項目 ・システム運用形態 ・システム運用スケジュール ・監視方法・基準 ・SLA(障害復旧時間など) ・災害対応策(DR)，業務継続策(BC) ・保存データ周期・量 ・エンドユーザ操作方法など
	移行要件	現行システムから新システムへの移行対象，移行方法などの移行に関する要件	例えば，以下の項目 ・移行対象業務 ・移行データ量 ・移行対象プログラム ・移行対象ハードウェア・移行手順 ・移行時期など
	付帯作業	システム構築に付帯する作業に関する要件	例えば，以下の項目 ・環境設定 ・端末展開作業 ・エンドユーザ教育 ・運用支援など
	その他	上記に該当しない要件	例えば，以下の項目 ・コスト，納期の目標値 ・電力量 ・作業環境 ・フロア面積など

（出典）小冊子「経営者が参画する要求品質の確保」に基づき作成

（7）　規模に対する変動要因

ここまで，機能要件から得られるソフトウェアの規模の求め方を述べましたが，非機能要件の要求レベルをソフトウェアの規模に反映する例もあります。図1.6の規模見積りでいえば，非機能要件を評価し，規模に換算する例に相当します。

具体的な例を**表1.6**に示します。第2部 第9章で示されているものです。日本情報システム・ユーザー協会でもユーザ企業により検討がなされています。

ユーザとベンダの双方で，個別のソフトウェア開発で求められている品質のレベルをチェックして，今回のソフトウェアの要求レベルは通常のレベル(類似のシステムの要求レベルの最も頻度の高いもの)から，どの程度高いのか，低いのかを確認しあい，そのレベルの高低を決定し，当該ソフトウェアの規模に換算します。

表1.6　規模へ反映する品質要件の例

主特性	副特性	評価の観点
機能性	合目的性	利用者/利害関係者の広がり，コンテンジェンシー対応，不正移行データ対応などの該当事象数
	正確性	正確性(検証)にかかわる標準テスト密度を基準としたテスト項目への要求水準
	接続性	他システムとの接続によるコード変換，フォーマット変換数
	セキュリティ	対応が必要なセキュリティ実現機能数，ただし，機能要件に定義されている部分は除く
信頼性	成熟性	故障低減に必要な実現機能数
	障害許容性	異常検知に必要な機能数
	回復性	再開処理に必要な実現機能数
使用性	理解性	理解性向上(機能など)のためのプレゼンツールなどの作成対象数
	習得性	習得性向上(使い方など)のためのマニュアルなどの作成対象数
	操作性	操作性向上(心理的/肉体的配慮，運用やインストール容易性など)のための実現機能数
保守性	解析性	解析に必要な実現機能数
	変更作業性	作成する保守用ドキュメントの数
	試験性	試験に必要な機能数

1.3.4　工数・コスト見積り

（1）　工数・コスト見積りの基本的な活動

工数・コストを規模から見積る場合，生産性の単位は，見積り対象が工数かコストかに応じて，人月/規模(単位規模あたりの開発にかかる工数(人時など)，金額/規模(単位規模あたりの開発にかかる金額)など，さまざまな設定が可能で，いずれの場合でも，組織における過去のプロジェクトデータなどに基づいて，次の2つの活動を行い，計算方法を確立しておくことが必要です。

(a) ベースラインの設定　プロジェクトや組織における過去のプロジェクトデータなどに基づいて，組織における見積りの基準値となるベースラインを設定します。

(b) ベースラインからの変動要因の設定　実際の見積りでは，上記のベースラインを基準として，さまざまな変動要因を評価して，変動値を求めます。そのためには，組織横断的に個別のプロジェクトで共通となる変動要因を洗い出し，その影響度合いの基準と個別のプロジェクトでの評価方法を確立しておきます。

なお，規模と工数・コストに関するデータは，ソフトウェア開発を実施しているベンダ企業側で蓄積されているのが現状ですが，ユーザ企業側においてもデータの蓄積を通して，これらの2つの設定を行うことで，ベンダ企業の提示する内容に対する比較検討が可能となり，見積り内容に関するコミュニケーションの深化につながります。

以下にそれぞれの詳細を示します。

（2）　ベースラインの設定

規模と工数，規模と金額などの関係のベースラインは，基本的に組織における過去の実績データの分析に基づき設定する必要があります。

規模と工数の間の関係式としては，工数と規模が正比例(**工数 ＝ α × 規模**)するもの，工数と規模の累乗が比例(**工数 ＝ α × 規模$^{\beta}$**)するものが主に利用されています。さらに複雑な関係式が利用されることもあります(コラム参照)。

組織でどの関係式を用いるかは，

- 第一には，当該組織で収集しているデータの範囲でもっとも誤差が少なくなる関係式かどうか
- 第二には，組織でどの関係式が最も現場で納得されるか

という点から判断します。

なお，上記のような基本的な関係は見積りの基準値を求めるためのベースラインとなるものです。

コラム（規模と工数・コストの関係）

☆比例（線形）の関係

工数と規模は基本的には比例するとして計算するものです。規模が比較的狭い範囲に収まっているときに採用されます。

工数 ＝ α × 規模　　ただし，α は定数

☆非線形な関係

非線形の関係の背後にある考え方は，規模が大きくなるとより多くの工数・コストが必要となるというものです。

COCOMO法などで採用されている関係式であり，次のとおり規模のβ乗に工数・コストが比例すると表されます。

工数 ＝ α × 規模$^{\beta}$　　ただし，α および β は定数

☆さらに複雑な関係

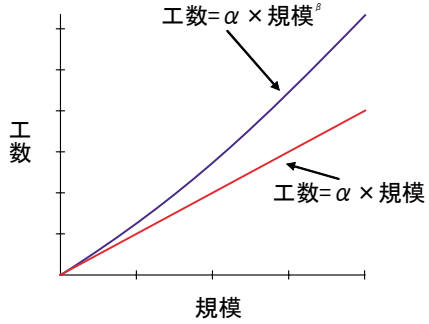
工数と規模の関係として，工数と規模の両者について，対数の対数をとった関係で，過去プロジェクトデータを統計分析して，次の関係を求めたものもあります。

log（log（工数））＝ α × log（log（規模））

この関係は，次のものと同じです。

工数 ＝ *a* ×（10$^{\log(\text{規模})^{b}}$）^{c}　　ただし，*a*，*b*，*c* は定数

これは，小規模から大規模まで，規模によらずに適用できることを目指したモデル式です。詳細は，第2部 第8章に示す事例を参照してください。



コラム（ベースライン設定時の注意事項）

ベースラインを設定する際には，以下の項目を明確にする必要があります。このあたりの定義をきちんと行っておかないと，データの収集時に違った解釈がなされたりして，見積りモデル作成時の分析に苦労することになります。

・ベースラインの基礎となる数値

（1）規模の求め方（単位，どのように数えるか）

（a）FPの場合：どの計測手法を使用するか（簡易計測法を適用するか）。

（b）SLOCの場合：マクロは展開するか。コメント行を含めるか。

（2）工数・コストを求める範囲（どの部分のデータを使うか，求めるか）

（a）工数の場合：管理工数を含むか。一括委託分の工数をどう扱うか。

（b）コストの場合：どの部分の工数に対するコストか。人件費単価はどう考えるか。間接費を含むか。

・ベースラインの適用範囲

（1）開発ライフサイクルのうちどこからどこまでを含むか。

（2）どういう分野のソフトウェアに適用できるか。

（a）業務分野

（b）システム基盤の技術分野

（3）　ベースラインからの変動要因

（a）　ユーザ制御要因とベンダ制御要因　常日頃経験されるように，上記のようなベースラインだけでは，プロジェクトの見積りを説明しきれません。個別のプロジェクトのさまざまな特性から影響を受ける，ベースラインからの変動を考慮した見積りが必要です。

生産性に影響を与える変動要因とは，要求品質のレベル，プロジェクトの特性，プロセスの確実さ，人のスキルレベルなど，ソフトウェアを構築するためのコストに大きな影響を与えます。同じ機能のものを作成するにも，変動要因が異なれば，必要な工数も大きく異なります。典型的な例としては，要求品質のレベルです。これは，機能性，信頼性，使用性，効率性，保守性，移植性として示されるものです（JIS X 0129（ISO/IEC 9126）に基づく）。

ベースラインからの変動要因を設定するに当たっては，ユーザ側とベンダ側のどちらに原因があるかが重要です。変動要因は，ユーザ側がコントロールできるものと，ベンダ側がコントロールできるものに分けることができます。以下，ユーザがコントロールできる変動要因を「ユーザ制御要因」，ベンダのものは「ベンダ制御要因」と呼びます。

見積りを互いに納得するためには，基本的にユーザ制御要因について共通認識を持つことが重要です。ベンダ制御要因は，契約時の見積りではユーザが直接考慮するものではありません(6)。ただし，いうまでもなくプロジェクトマネジメントの観点からは非常に重要なものです。見積り自体に実現性が乏しい場合は，プロジェクトが失敗するリスクが高く，ベンダおよびユーザの双方が損害をこうむります。ユーザ・ベンダ相互に変動要因および変動要因の設定に関するリスクを明らかにして，相互の責任を確認し，協力してモニタし，コントロールすることが大切です。

（b）　ユーザ制御要因の例　**表1.7**および**表1.8**にユーザ制御要因の具体的な例を示します。この例は第2部に示すジャステックのもので，日本情報システム・ユーザー協会でもユーザの視点から検討されています。

ユーザ制御要因の考え方は，以下のとおりです。

ユーザ制御要因のひとつとしてユーザの「業務ナレッジ」がありますが，これ

（6）　端的な例として，ベンダ側の開発チームのスキルが低く，ベースラインよりもコストがかかってしまう場合，ユーザにとっては，そのコスト増は，納得できないものです。

表1.7　ユーザ制御要因の例（1／2）

主特性	副特性	評価の観点
業務特性	業務ナレッジ	顧客の開発対象業務に対する業務ナレッジが生産性に及ぼす影響
ソフトウェア特性	安定度/信頼度/使用度	システム/製品となる他社作成ソフトウェアもしくはCOTSの安定度・信頼度
ハードウェア特性	安定度/信頼度/使用度	システムもしくは製品となるハードウェアの安定度・信頼度
コミュニケーション特性	顧客窓口特性	意思決定能力（期限遵守，決定事項のくつがえる度合）
	工期の厳しさ	基準工期（月）＝2.7×（人月）$^{1/3}$に対し▲30％限度とした短期化度合
	コミニュケーション基盤	開発拠点分散，資料など情報共有，電子媒体・システム具備など物理的基盤充実度
	レビュー体制	むだなレビュー（重複多段階など）の排除およびレビュー効率向上への工夫度合
開発環境特性	開発手法/開発環境	開発手法・環境（ソフト/ハード/ツール）の信頼性，占有率などを考慮した使用実績
	テスト手順書水準	テスト手順の具体化度（操作手順＆入出力の具体化の要求水準）
工程入力情報特性	業務関連資料	必要資料の具備状況（正確性，信頼性を含む）および使いやすさ（検索性，理解性）
	他システム関連資料	必要資料の具備状況（正確性，信頼性を含む）および使いやすさ（検索性，理解性）
	規約・標準化関連資料	必要資料の具備状況（正確性，信頼性を含む）および使いやすさ（検索性，理解性）
顧客の協力特性	役割分担特性	顧客がベンダに協力する度合および顧客とベンダとの役割分担の明確性
改造・再構築特性	既存システムの練度	改造対象の母体または再構築する元のシステムに関係する顧客の熟練度
	既存テスト環境流用水準	既存のテストリソースの流用度合
	母体調査ツールの水準	調査ツール機能（絞込み，モニタリング，リバースなど）の数
	既存母体の品質	正確性（潜在バグなど），解析性，環境適用性などの事象数

COTS：Commercial Off The Shelf

表1.8　ユーザ制御要因の例(2/2)

主特性	副特性	評価の観点
機能性	合目的性(要求仕様の網羅性)	要求の記述水準および網羅性。要件定義については新規性，方針明確性，ステークフォルダの多様性などを考慮
	正確性	正確性(検証)にかかわる標準レビュー工数(各工程10%)を基準にした要求水準
	接続性	基準単位(100KSLOC)に対する社内/社外システムとのインタフェース先の数
	整合性	整合をとる社内/社外の規格・基準の数，全体適合性やグローバル化対応も含む
効率性	実行効率性	実行効率に対する一般的要求水準(注)の最適事例を基準にした要求水準
	資源効率性	資源効率に対する一般的要求水準(注)の最適事例を基準にした要求水準
保守性	解析性	ソースコードの解析性をコード化規約に定めるコメントに対する要求水準により評価
	安定性	ソフトウェア変更に対しシステム品質維持可能とする水準をライフサイクル目標年数の長さにより評価
移植性	環境適用性	多様なハード，ソフト，運用環境に適用させる要求の水準
	移植作業性	環境を移す際に，必要な労力を低減させる要求の水準
	規格準拠性	移植性に関する国際/国内規格または規約を遵守する要求水準
	置換性	使用環境/条件を変更せずに他のソフトウェア製品と置き換えて使用可能とする要求の水準

(注)　類似のシステムの要求レベルで最も頻度の高いものに基づいて，あらかじめ設定したもの。

は顧客の開発対象業務に関する業務ナレッジが，生産性に及ぼす影響を示したものです。例えば，上流工程やシステムテスト工程では，中核メンバ構成と当該業務ナレッジの保有度合いにより，プロジェクト方針策定や意思決定などのスピード，さらにはアイドリングや手戻りの発生度合いなど，生産性への影響度を評価し，その結果を価格に反映します。ときとして，生産性の低下を補うために，ベンダ側で要求を明確にしたり，必要な情報を提供したりする活動が発生する場合があります。もちろん，その活動は，当該工程の見積りに追加作

業として，付加することになりますが，それによりユーザ側の生産性が上がり，トータルとして工数削減できれば，ユーザとして，このような支援をベンダに要求することで，全体の生産性をコントロールすることができます。

プラットフォーム(ハードウェア，ソフトウェア)の安定性・信頼性は，当該プラットフォームにおける効率的な開発手法の保有度合いや使用実績の有無などにより評価します。使用実績が十分ある場合は，ハードウェアやCOTS(Commercial Off The Shelf)の特性を把握し，その対策が既知であるがゆえに生産性が高くなります。逆に，その特性および対策が既知でない場合は生産性が低下します。

また，ユーザは，ビジネス機会獲得のためにシステム開発・稼働までの時間の短縮(工期の圧縮)を求めますが，工期の短縮は短い期間に作業が集中するので，多くの場合，増員を伴い，立上げ期間不十分などの要因により，生産性の低下を招きがちです。そのため工期については，このようなリスクの軽減対策や生産性の低下に対する影響度合いを評価した上で決定する必要があります。

これらは，ユーザ制御要因について，ユーザが，それぞれの項目を工夫・改善することで，生産性の向上が可能になることを示唆しています。このことは非常に重要であり，ユーザが工数・コストをコントロール(改善努力)できることを意味します。

上記の例で，中核メンバの当該業務ナレッジを高まる方向に改善すれば，方針策定や意思決定のスピードが高まり，アイドリングや手戻りの発生を抑えられます。また，プラットフォーム(ハードウェア，ソフトウェア)の安定性・信頼性は，プラットフォームの試行を行うなど，事前に問題点を把握し，効率的な開発手法を確立することにより，生産性が向上し，工数・コストの低減が図れます。

表1.7および表1.8は，ユーザが工数・コストの見積りのコントロールをするためには，どこをコントロールすればよいのかを示したリストです。各組織において，このリストを参考にして，ユーザと議論し，必要なユーザ制御要因について作り上げてください。また，このリストの適用の可否を検討し，可能であると判断されれば，そのまま活用する方法もあります。

(c)　変動要因の定量化の方法　変動要因は，(2)で求めたベースラインの関係式に対して，変動分として反映させます。次の式により反映させることが基本です。

工数 ＝ α × 規模 ×（１＋変動分）

ただし，変動分＝(変動要因の影響度)の総和

変動要因を洗い出したのち，その影響度を定量化することになりますが，その方法は，過去のデータからある特定の変動要因以外の条件が同じデータを抽出し，変動要因の度合いの違いによる影響度の違いを統計的に導き出すものがあります(例：第３章 3.1.2項および，第２部 第９章 参照)。また，影響度合いをひとつの定数にすることがむずかしいと考える場合，広がりのある確率分布として設定して，リスク(見積り値を超える確率)を予測する方法もあります(例：CoBRA法。SECでの見積り手法に関する実証実験の第１章参照)。

自分の組織に見合った方法(適用が可能と考えられる方法)を選択し，始めることが重要です。

（４） 変動要因に関するユーザとベンダとの調整プロセス

要求される品質のレベルを設定する場合と同様に，ユーザとベンダ間で，個別のソフトウェア開発プロジェクトでの変動要因のレベルをチェックして，今回のソフトウェアは，個々の変動要因の基準(類似のソフトウェア開発のプロジェクト実績データから得られる基準値)から，どの程度高いのか，低いのかを確認しあい，そのレベルに応じて生産性の高低を評価し，見積りへ反映することで，見積りの妥当性を確保する必要があります。

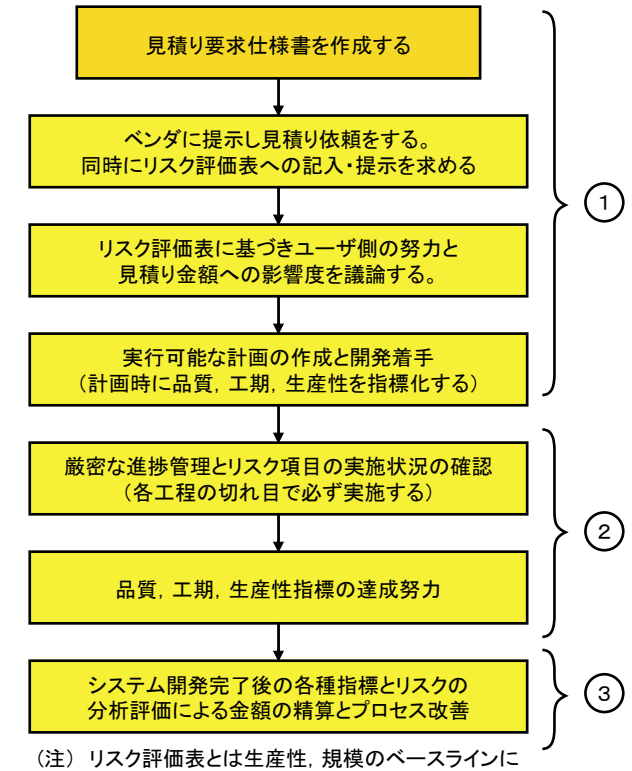
ユーザがコントロールできる生産性変動要因の調整のプロセスを**図1.14**に示します。表1.7および表1.8に示したような項目をもとに，ユーザ・ベンダ間で前提を合意し，見積りを行います。ユーザ制御要因の中には，プロジェクトの見積り時に，ユーザが確約できないものも当然存在しますが，見積りに際して前提条件として仮決めした上で，これらのリスクとして共有します。プロジェクトの進行段階では，その前提をモニタして変化があるか否かを把握して，変化がある場合には，必要に応じてユーザ・ベンダ間で前提を再確認するプロセスを踏みます。

プロジェクト終了時には，契約金額の確定・精算などが発生しますが，これは個別の契約の内容によります。

互いに納得できる見積りのためには，以上に示してきた事項をユーザ・ベンダ間で密にコミュニケーションし，前提の把握と設定，その変化のモニタとコントロールすることにつきます。

図1.14において，①は契約段階，②はプロジェクト進行段階，③はプロジェ

クト終了段階を示しています。①では，規模見積りおよび変動要因の見積りを実施し，ユーザとベンダとの間で了解します。②では，互いに了解した内容の変化をモニタし，必要に応じてコントロールする段階です。③は，最終的に①で示した条件に変化がないかなどを確認して，契約によっては，この段階で契約金額を精算する場合もあります。



（注）リスク評価表とは生産性，規模のベースラインに影響する変動要因(環境変数表)である。

(出典)システム・リファレンスマニュアル(SRM：JUAS)

図1.14　ユーザ・ベンダ間の調整プロセス

1.3.5　工期見積り

（１） 工数と工期の関係

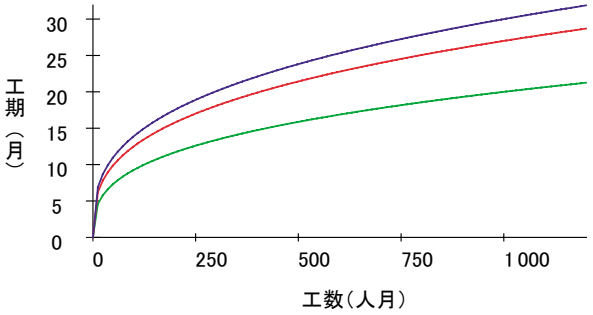
工期は，工数と高い相関関係があるといわれており，一般的な関係として次

のものが示されています(例：COCOMO法)。

$$工期 = \alpha \times \sqrt[\beta]{工数} \qquad ただし，\alpha = 2.0 \sim 3.0，\beta = 3.0前後$$

傾向として，工期(T_D)は，工数(*Effort*)のおおよそ3乗根に比例するという傾向があります。これまでα=2.0～3.0と報告されています。最近の結果では，日本情報システム・ユーザー協会の報告(7)でα=2.7となっています。α=2.0の場合，α=2.7の場合およびα=3.0の場合の具体的な様子を**図1.15**に示しました。

$$T_D = 2.0 \times \sqrt[3]{Effort}, \quad 2.7 \times \sqrt[3]{Effort} \quad または \quad 3.0 \times \sqrt[3]{Effort}$$



グラフの上から，α=3.0，α=2.7，α=2.0の場合

図1.15　工数と工期の関係

(2)　SECデータベースの例

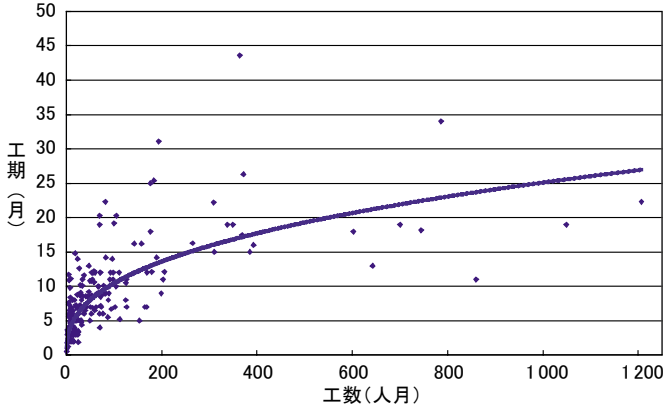
実際のプロジェクトデータからも上記のような工数と工期の関係(傾向)が見られます。2005年3月のSECのプロジェクトデータから，工数(人月)と工期(月)の関係をグラフ化したものを**図1.16**に示します。ここでは，工期が工数の3乗根ではなく，一般の累乗根に比例するとして，最もフィットする曲線を求める方法をとっています。

データは「新規開発」のプロジェクト(211件)を用い，累乗根の回帰曲線を求め，式としては次のとおり表されます。累乗が0.38であり，3乗根(約0.33乗)に比較的近いものとなっています。

(7)　2004年度ソフトウェア・メトリックス調査

$$工期 = 1.8 \times 工数^{0.38}$$

この関係式が成り立つ背景は，コラム欄を参照してください。



（注）データベース内の平均値として，1月は166時間/月と換算

図1.16　SECデータベースに基づく工数と工期の関係(回帰曲線)

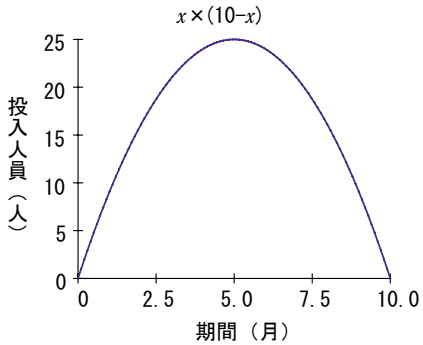
コラム(工数と工期の関係のモデル)

(1)　3乗根の関係の場合

3乗根に比例する場合を例にとって，工数と工期の関係の背後にある理由を見てみましょう。まず，工数はプロジェクトの期間中の要員の割り当てを時間で積分したものとモデル化できます。実際，工期＝$\alpha \times \sqrt[3]{工数}$の式は，プロジェクト開発中における「人員の割当てが開発の半ばで人数が最も多くなる」分布，具体的には**図1.17**に例として示すような2次関数を仮定すると導かれます。人員の分布を$M(t)$とすると，tの関数式は，次のとおり表現されます。

$$M(t) = k \times t(T_D - t)$$

ただし，T_Dは，プロジェクトの終了時までの時間(工期)です。kは人員の能力に応じた適当な係数です。



（備考）例として期間は10ヶ月とした

図1.17　ソフトウェア開発中の人員の割当ての様子

プロジェクトで必要な工数(*Effort*)は，この関数を時間(期間)で積分したものになるので，*Effort*は次のとおり求まります。

$$Effort = \int_0^{T_D} M(t)\,dt = \int_0^{T_D} kt\,(T_D - t)\,dt$$

$$\therefore Effort = k\left[T_D\frac{t^2}{2} - \frac{t^3}{3}\right]_0^{T_D} = k\left(\frac{T_D^3}{2} - \frac{T_D^3}{3}\right) = k\,\frac{T_D^3}{6}$$

この式を変形すると，工期(T_D)と工数(*Effort*)の関係は，次のとおり導かれ，工期が工数の３乗根に比例する結果が得られます。

$$T_D = \sqrt[3]{\frac{6\times Effort}{k}} = \sqrt[3]{\frac{6}{k}}\ \sqrt[3]{Effort}$$

人員の割当ては，実際にはもっと複雑ですが，図1.17のような人員の割当てを仮定することで，工数と工期のシンプルな関係が得られます。

（２）　SECデータベースから得られる関係の場合

SECデータベースから得られる関係について，人員配分はどのような式になるのかを導いてみると，次式が得られます。

$$M(t) = 1.25\times t^{0.82}\ (T_D^{0.82} - t^{0.82})$$

図1.18にT_D＝10ヶ月の場合のグラフを示します。２次関数よりも少し左側に偏ったものになっています。

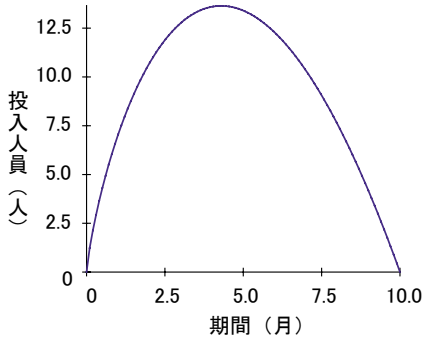


図1.18　SECデータベースの関係式から導かれる人員割当ての概算式

なお，工期が10ヶ月の場合，ベースとなっている工数は，おおよそ90(人月)です。また，モデル上は「ピーク時人数」がおおよそ13人または14人である(図のピークは13.6人)との結果が得られます。もちろん，以上の工数と工期の関係はベースラインとして設定されるものであり，この値に対して他の要因などを加味して，工期やピーク時人数などが予測されることになります。

1.4　妥当な契約

ここまで，規模見積りと見積り時期との関係，プロジェクトにかかわるリスクや変動要因の工数・コスト見積りへの反映またはそれらのリスクや変動要因をコントロールすることに関して述べました。本節では，これらのリスクについて，ユーザとベンダの両者が協調して，プロジェクトの特徴に応じた契約タイミング・契約形態を設定・選択することにより，回避または軽減する方法を提示します。

最初に契約に際してのユーザとベンダそれぞれの経済原理・行動原理を明らかにした上で，コスト構造から見てコスト低減を図るには，どうすべきか示すとともに，契約によるリスク解消の糸口と契約の具体的な事例を示します。

1.4.1　契約金額交渉

（１）　契約形態の分類

わが国で頻繁に採用される契約形態として，一括請負型契約と支援型契約（委任契約や準委任契約のような労務提供型の契約）があります。

一括請負型契約ではプロジェクトの結果が支払の対象となるので，成果物に対する責任は受注者であるベンダ側が負うことになります。一方，支援型契約ではその契約の履行のために費やされた時間または労力が支払の対象であり，達成された成果ではありません。成果物に対する責任は発注者であるユーザ側が負うことになります。なお，きわめて当然のことですが，上記いずれの場合であっても，ベンダはプロフェッショナルとしての十分な責任を果たすのが，大前提となっています。

支援型契約の場合，ユーザは契約の中でベンダに対して要求する労力レベルの規定，ならびにそのパフォーマンスの監視などを適切に実施しなければなりません。また，コスト低減を図ろうとした場合，一括請負型契約であれば，契約前にユーザとベンダの両者がある程度合意していますが，支援型契約ではおろそかになる場合が多く見受けられます。支援型契約でも契約前にユーザとベンダの両者が互いに意識して調整し合意することが望ましいと考えます。これらは往々にして見落とされがちな点なので，注意が必要です。

なお，以下では一括請負型契約を対象として，その具体的な進め方などを示すこととします。

（２）　契約金額と見積りとの関係

一括請負型契約における価格は，各企業・各団体によって多少の違いはあるものの，基本的には次の三つの要素から構成されると考えられます。

① コスト
② リスク分（不確実なコスト）
③ 役務とは別に確保される利益分

契約に際して，ユーザとベンダはそれぞれ異なる原理に基づき行動することになります。ユーザには，機能，品質，ビジネス効果などといった自分達が持つ要求を実現し，かつ可能な限り発注額を低く抑える経済原理が働きます。一方，ベンダには，契約金額の中で適正な利益を確保する行動原理が働きます。

図1.19の価格構造を考えると，契約金額の削減を図る場合に，ベンダ側がとりうる対応は，次の三つのいずれかとなります。

① 利益分の圧縮
② リスク分の縮小
③ コストの縮小，ひいては規模（機能）や工数の縮小

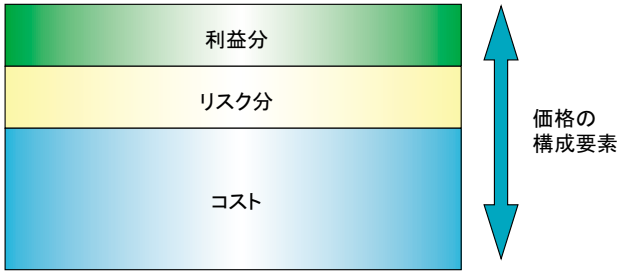


図1.19　一括請負型契約における価格の基本構造

これらの項目をどういった優先順位で，どの程度行うかは，ベンダ各社の判断となりますが，妥当性を欠いた無理な契約額の削減は，結果としてユーザ側にも不利益をもたらすことにつながります。

（３）　コスト低減の方策

ユーザ・ベンダ間の契約金額の調整において，ユーザ側の価格削減の方策として，時間単価の削減要求があげられます。しかし，これはベンダ側にとっては努力の余地が残されていない上，無理な要求か否かを判断する根拠を失ってしまうことになります。

図1.20は，価格削減の方策として時間単価ではなく，妥当性を判断できる方法について示したものです。これは，本来変動要因とされている「規模の変動分（ΔS）」「生産性の変動分（ΔP）」を，ユーザ・ベンダ間で調整するものです。

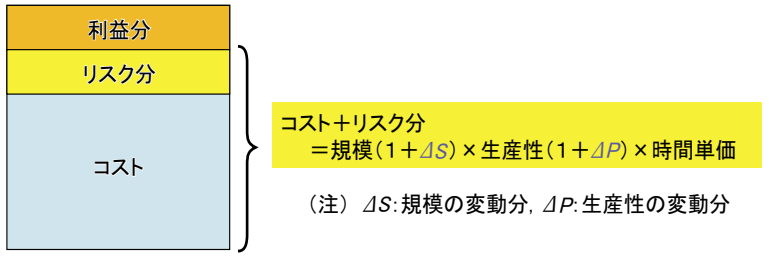


図1.20　コスト低減の方策

契約に際して，見積りは契約金額決定のための重要なインプットとなることはいうまでもありません。ユーザとベンダの両者が，合意できる見積り根拠と金額設定があってこそ，妥当な契約は成立し得るといえます。

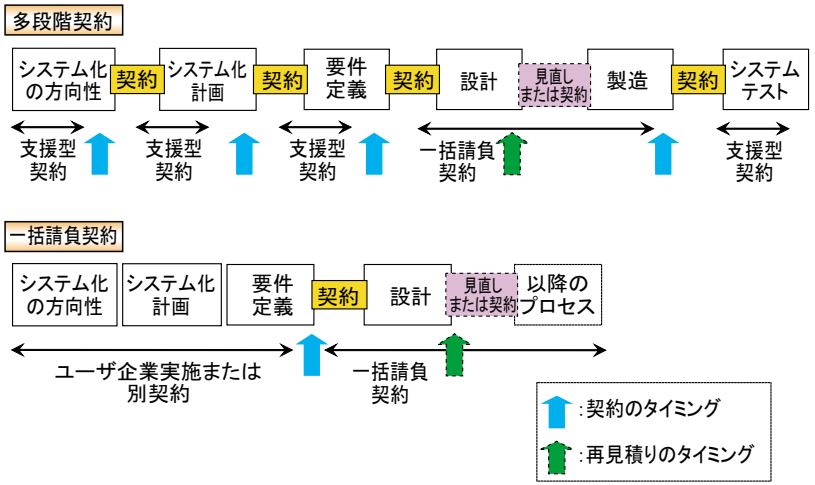
1.4.2　契約によるリスク解消の糸口

（1）　多段階契約の採用

ソフトウェア開発において，最初の開発工程である「システム化の方向性」レベルの基本的な考え方は決まっているものの，「システム化計画」「要件定義」にあたる部分は，最初から作業を開始する必要がある場合，開発工程ごとに契約を締結していく多段階契約を選択するのも一つの方法です。

多段階契約では，契約作業にかかわる手間は増大しますが，開発途中で発生しがちな仕様変更の影響を抑えつつ，その時点で明確になった部分の反映が可能であるため，比較的大規模なプロジェクトに適しています。

図1.21の上部は，「システム化の方向性」「システム化計画」「要件定義」がそれぞれ終了した時点で契約を締結する場合を示したものですが，契約のタイミングは，プロジェクトの特性に応じて，適宜設定することができます。1.4.1項



（注）図の矢印は，すべてのタイミングで契約すべきとするものではなく，タイミングとして多くの選択肢があることを示すもの。

図1.21　多段階契約と契約・再見積りのタイミング例

の（1）でも触れたように，成果物に対する責任の所在などを考えると，ユーザ・ベンダ間において成果物に対する責任の合意がとれる場合には，一括請負型契約，そうでない場合は，支援型契約を採用することが，現実的と思われます。したがって，例えば，「設計」より後の製造にかかわるプロセスは，一括請負型契約，「設計」以前ならびに「システムテスト」以降は，支援型契約を採用する方法も，多段階契約の有効な活用です。

（2）　実費償還型契約の採用

（1）のほかにも，ユーザ・ベンダ双方のリスクを軽減させる方法として，実費償還型契約があります。実費償還型契約とは，契約上定められた金額の範囲内で，発生したコストに対する実費が支払われるものです。

実費償還型契約は，さらにいくつかの契約形態に細分化されますが，比較的採用事例が多いのは，動機付け型契約と報償型契約の二つです(8)。動機付け型契約は，予定コストに対する実際コストの差分を，超過，余剰に関係なくユーザ・ベンダ間で分担して負担するものです。一方，報償型契約は，契約開始時に設定された基本額に加えて，ユーザが常に成果物の品質などを評価し，その結果に基づいて追加の報酬をベンダに支払うものです。

いずれもプロジェクトの過程において，ユーザ側には，ベンダのパフォーマンスを評価する能力が要求されますが，自分たちにとって，本当に使いやすいシステムを適正なコストで実現することが可能となります。一方，ベンダ側にも自らのパフォーマンスを説明できる能力が求められます。この契約の特徴を最大限に生かすためには，ユーザとベンダとの間の密なコミュニケーションが必須条件になります。

（3）　具体的なリスク軽減の契約事例

動機付け型契約・報償型契約を採用することによって，ユーザとベンダの間で積極的なコミュニケーションが図られ，結果的に不確実な部分が明確になり，リスクが減少するとともに，効率の良い仕事が行われ，双方にとって良い結果を導いた例を以下に示します。

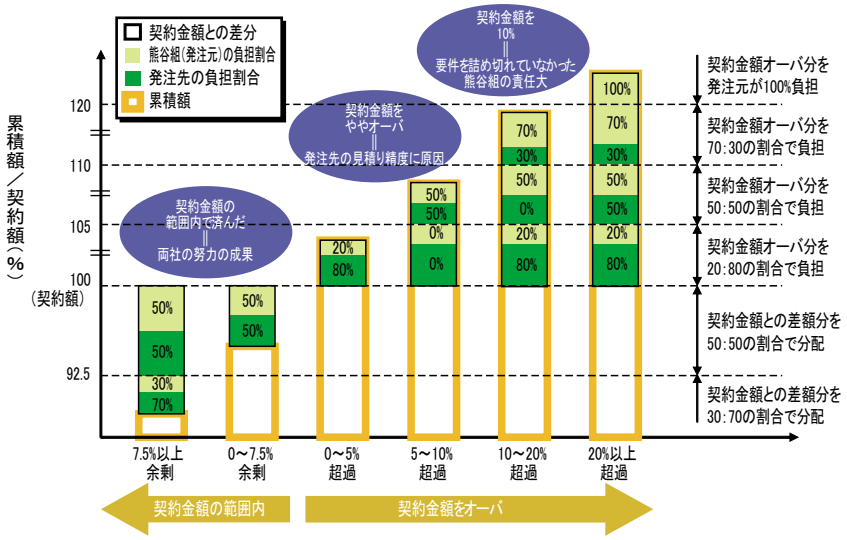
(a)　国内の事例　建設業界での事例として，建築原価管理システムの開発

（8）　分類上，実費償還型契約ではない動機付け型契約，実費償還型契約ではない報償型契約も存在することに注意が必要です。詳しくはコラムを参照してください。

（9）　ここでの内容は，株式会社熊谷組　高木副社長へのインタビュー（2005年11月SEC実施）に基づきます。

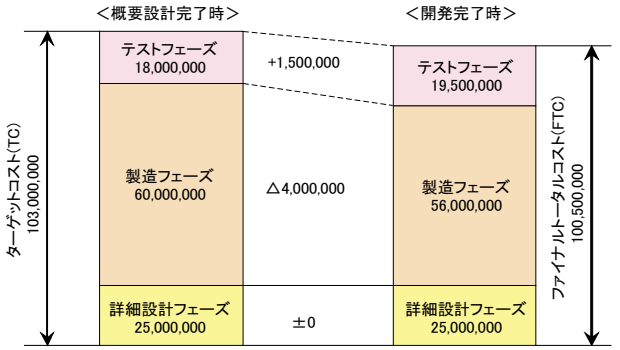
プロジェクトにおいて，パートナリング方式という動機付け型契約に類似した契約形態を採用し，コスト低減とともにプロジェクトを成功裡に完了した例があります[9]。特徴的なのは，パートナリング方式の導入に際し，赤字の大きさに応じて，ユーザ側が負担する金額の比率を工夫した点です(**図1.22**参照)。**図1.23**および**図1.24**に示した例では，詳細設計フェーズは，当初の見積りコスト(ターゲットコスト)と実際にかかったコストが同額でしたが，製造フェーズでは，約７％弱コストを抑えることに成功しています。この場合，ベンダとユーザ双方の努力の結果で5:5の比率で配分します。一方，テストフェーズでは８％強のコストオーバとなっています。この場合，5%分まではベンダとユーザが8:2，5%を超える分については5:5の割合でそれぞれ超過分を負担します。

図1.23および図1.24に示す計算例では該当しませんが，表中の「配分率」欄にもあるとおり，コストが20%以上オーバした場合，その超過分は，ユーザ側が赤字分を全額負担します。これは，ユーザ側にとって要件を確実に管理しなければならない，という意識を喚起させる効果があります。逆に，赤字が５％未



（出典）日経コンピュータ2005年９月５日号，p.100の図をもとに作成

図1.22　実費償還型契約における損益配分の例



（出典）株式会社熊谷組資料よりSEC作成

図1.23　実費償還型契約における損益配分の計算例(１)

項目		計算式	フェーズ			合計
			詳細設計	製造	テスト	
ターゲットコスト	(TC)		25,000,000	60,000,000	18,000,000	103,000,000
ファイナルトータルコスト	(FTC)		25,000,000	56,000,000	19,500,000	100,500,000
差分	(α)	(FTC−TC)	−	−4,000,000	1,500,000	−2,500,000
比率	(ρ)	(α/TC)	−	−6.67%	8.33%	−2.43%

区分	比率	配分先	配分率	詳細設計	製造	テスト	全体
ペインシェア	20%≦ρ	発注元	100%	−	−	−	0
		発注先	0%	−	−	−	0
	10%≦ρ <20%	発注元	70%	−	−	−	0
		発注先	30%	−	−	−	0
	5%≦ρ <10%	発注元	50%	−	−	300,000	300,000
		発注先	50%	−	−	300,000	300,000
	0%< ρ <5%	発注元	20%	−	−	180,000	180,000
		発注先	80%	−	−	720,000	720,000
ゲインシェア	−7.5%< ρ <0%	発注元	50%	−	−2,000,000	−	−2,000,000
		発注先	50%	−	−2,000,000	−	−2,000,000
	ρ ≦−7.5%	発注元	30%	−	−	−	0
		発注先	70%	−	−	−	0
シェア金額計		発注元		−	−2,000,000	480,000	−1,520,000
		発注先		−	−2,000,000	1,020,000	−980,000
請求金額				25,000,000	60,000,000	16,480,000	101,480,000

(計算式) 100,500,000+(2,000,000−1,020,000)−25,000,000−60,000,000

（出典）株式会社熊谷組資料よりSEC作成

図1.24　実費償還型契約における損益配分の計算例(２)

満であれば，ベンダ側の見積り精度に原因があるとし，計算例のとおりベンダが，赤字額の８割を負担します。一方，開発原価と固定費の合計が当初見積りより安く済んだ際の値引き率も，黒字額(余剰額)によって変えています。当初見積りよりも安く開発できた場合は，その分ベンダの取り分が増えるような比率を設定して，ベンダのやる気を引き出すことを目的としたものです。このような考え方は，「ペインシェア/ゲインシェア」と呼ばれています。

図1.23および図1.24は，計算の例ですが，実際のプロジェクトの結果としては，実装の終了段階で,400万円強の黒字が出たため，この範囲内で仕様変更や追加開発を実施し，テストを終えたプロジェクトの完了段階で，契約金額を約９万円下回る0.1%弱の黒字を出したと報告されています。

この例では，ベンダにとって，実際にかかったコストを支払ってもらえる安心感があります。一方，ベンダがプロジェクトの進捗状況を明確にすることを基本として，ユーザ・ベンダの両者が協力してプロジェクトの状況(かかったコストなど)を確認しあう仕組みが必要です。互いにコストを明らかにすることによって，不必要な機能などの検討や削除につながり，コスト増加要因への対策をとるなどのフィードバックコントロールがかかるようになり，結果として最適なシステムの構築とともにコスト低減につながります。

(b)　海外の事例　イギリスで実施された北海の石油プラント建設プロジェクトでは，不確定要素が大きく，受注者でリスクを背負うことが非常に困難と判断されたため，CRINE(Cost Reduction in the New Era)という動機付け型契約の一種が採用されました。この結果，受注者側のAndrew　Allianceでは同規模プロジェクトと比較して30%のコスト(約8 300万ドル)の削減を達成し，かつ予定の６ヵ月前にプロジェクトを完了させることができたと報告されています。

また，米国国防総省の飛行機メンテナンス訓練システムでは，報償型の契約が適用されました。これは，新規に調達される戦闘機が納品される90日前に，その機種に対応するメンテナンス訓練システムが，納品されることが，必須とされており，効率的なコミュニケーションを行えることが，重視されたのが主な理由です。結果として，各機種用のシステム全体で90%の顧客満足度が得られるとともに，2年間にわたり，納期よりも前にシステムの提供が行われたと報告されています。

コラム(契約形態の考え方)

契約形態の考え方の包括的な例として，米国政府が調達時に使用するFAR(Federal　Acquisition　Regulation)による分類があります。これによると，まず支払形態の観点から

① 定額型契約(Fixed Price)
② 実費償還型契約(Cost Reimbursement)
③ 間接作業型契約(Level Effort)

の三つのカテゴリーに大別されます。わが国で比較的頻繁に採用される一括請負型契約は，①の定額型契約，支援型契約は③の間接作業型契約に属することになります。

さらに，FARでは結果として超過あるいは余剰となったコストの扱いをどのようにするかによって，分類を行っています。その分類が1.4.2項の(２)でも触れた「動機付け型契約(Incentive　type)」や「報償型契約(Award　type)」です。**表1.9**のFPIとCPIFが前者，FPAFとCPAFが後者に該当します。

表1.9　契約形態による分類

支払形態による分類	契約分類	支払の対象
定額型契約 (Fixed Price)	FFP(Firm Fixed Price)	プロジェクトの結果
	FPI(Fixed Price Incentive)	
	FPAF (Fixed Price with Award Fees)	
実費償還型契約 (Cost Reimbursement)	CPFF (Cost Plus Fixed Fees)	プロジェクトの結果と要した実費
	CPIF (Cost Plus Incentive Fees)	
	CPAF (Cost Plus Award Fees)	
間接作業型契約 (Level Effort)	T&M (Time & Material, Labor Hour)	時間，あるいは労力
	FFP-LET (FFP-Level of Effort)	
	Cost-Plus-Fee Term	

：動機付け型契約　　：報償型契約

第２章　見積り能力の向上

第１章では，ユーザとベンダ間で納得できる見積りを実現するためには，どうしたらよいかをクローズアップして解説をしました。本章では，第１章のような見積りを実践できるようになり，さらにその能力を向上させるためには，個々の企業において何を行う必要があるかを示します。

2.1　見積りの重要性

システム開発プロジェクトの見積りの妥当性は，ユーザ企業とベンダ企業の双方にとって，必要なシステムを妥当な価格で入手したり，一定の利益を確保する点で，プロジェクトの成否に直接つながります。低すぎる見積りでは，ベンダ企業において必要なアウトプットの達成が困難になります。一方，高すぎる見積りは，ユーザとベンダとの間の信頼を損なう原因になりかねません。この見積りの妥当性を確保するために互いに何をするべきかは，第１章で見てきたとおりです。

ところで，見積りは契約時において重要なだけではありません。見積り結果は，プロジェクトの実施内容や方法まで規定する最も基本的なデータです。見積りの結果，契約額が決まると，プロジェクトマネージャは，その額からプロジェクトを組み立て，開発完了時には，妥当な利益を確保しつつ，ユーザの要望を満足させるソフトウェアを構築する必要があります。つまり，プロジェクトの予測だった見積りが目標(ゴール)となり，見積りに応じたプロジェクト成果をいかに出していくかという課題になるわけです(**図2.1**参照)。

さらに，見積り活動は１つのプロジェクトに閉じたものではなく，予測と実績との差異分析を通して次のプロジェクト，さらには組織全体のプロジェクトの見積り精度向上に向けてのフィードバックの基となります。見積り精度の向上は，単に見積り技法の問題ではなく，組織として，どのようにプロセスを構築し，取り組んでいるかという点につながっています。

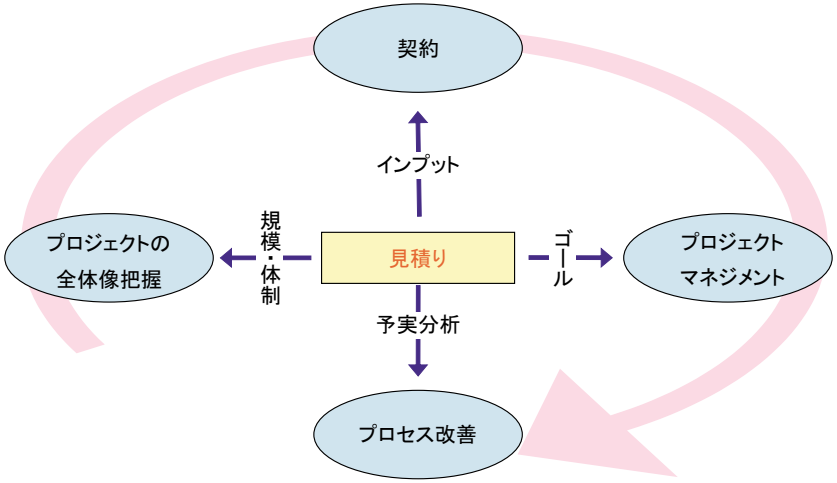


図2.1　見積りの重要性

2.2　見積り活動

2.2.1　見積りとプロジェクトマネジメント

見積りの結果は，唯一の正しい値があると考えるのではなく，プロジェクトのゴールを目指して，関係者が，プロジェクトをコントロールすることにより，プロジェクトの実績を見積り値に近い値に収められることが重要です。そのためには，見積り値を，プロジェクト遂行上，コントロール可能な許容範囲内に収めなければなりません(**図2.2**の赤い箱で示される範囲)。また，見積り値は，単に数字が示されるものではなく，見積りに当たっての前提条件やリスクなどの発生に備えた対応策とあわせて示されるべきです。

見積りは単独の活動として扱うのではなく，次のとおり，常にプロジェクトマネジメントと一体として対策を考えることが肝要です。

- 過去のプロジェクトで実証された定量的能力および規模，工数，工期などの関係式に基づき見積る。
- 見積り結果をプロジェクトマネジメントの基礎情報として活用する。
- 規模，工数，工期などの関係式は，継続的に評価して改善する。

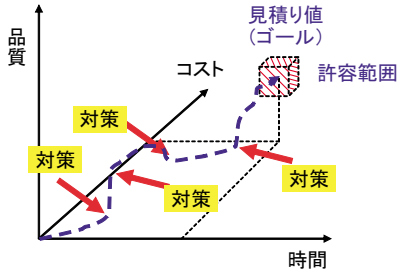


図2.2　見積りをゴールとしてモニタリング/コントロール

2.2.2　見積り活動のサイクル

見積りは，個別のプロジェクトの活動ですが，組織的に一貫性を持たせるために，見積り手法やデータ収集・分析などの見積り手順が組織内で確立している必要があります。その手順に基づいて見積り，見積り結果をインプットのひとつとして，プロジェクトマネジメント(進捗マネジメント，プロジェクトリスクマネジメントなど)を実施します。プロジェクトの終了後に見積り値(計画)と実績との差異分析が行われ，その結果がプロジェクト活動または組織活動あるいは見積りモデルにフィードバックされます。

見積り活動は，次の活動からなります。

(１)　見積り手順の確立

見積りについて組織で一貫した手順を確立する。

(２)　見積りの実践

確立した見積り手順を実践する。

(３)　見積り結果(計画)と実績の差異分析

計画時の見積り結果とプロジェクト完了時の実績値との間の差異分析を通して，差異が生じた根本原因を特定する。

(４)　差異分析結果のフィードバック

差異分析結果に基づいて改善対策を検討し，対策を展開します。差異分析の結果反映される対象として，次の二つがあります。

① プロジェクトマネジメントへのフィードバック

② 見積り手順(見積り手法を含む)へのフィードバック

(５)　共通要因に基づくプロセス改善

複数のプロジェクトで共通な課題を見積り，実績評価の繰り返しに基づき分

析し，対策を検討して，プロセス改善を展開します。

図2. 3に見積り手法の構築からフィードバックまでのサイクルを示します。また，次項以降には，それぞれの活動の詳細を示します。

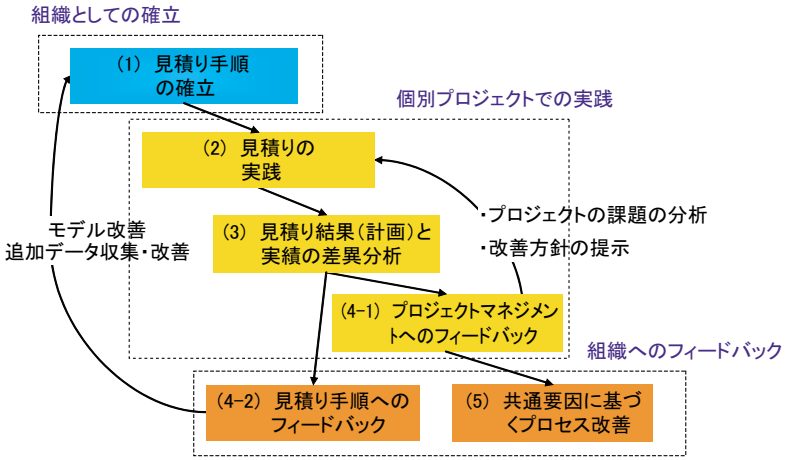


図2. 3　見積り手法の構築から活用，フィードバックのサイクル

2. 2. 3　見積り手順の確立

組織で一貫した手順を確立するために，以下のことを実施します。

（１）　見積り対象の設定

見積り対象の構成要素のうち，どの部分を見積るかを設定します。第１章の図1. 4に示した構成要素に対して，何(規模，工数，コスト，工期など)を見積るかです。

（２）　見積り対象と見積り手法

見積りする対象にふさわしい見積り手法を選定します。見積り手法には，①類似システムとの比較・類推，②積み上げによる方法，および③パラメトリックなモデルによる予測という３つに分類できます(**表2. 1**参照)。同じ類型のシステムを繰り返し開発している組織において，一貫性を持って過去のデータを収集・分析していると，①と②の方法の精度が高い特徴がありますが，反面，根拠を説明しづらく，再現性，客観性に乏しい欠点があります。一方，③はデータや経験から導かれるパラメータなどの設定により，繰り返し検証が可能であり，原因追求などの効率化という点でメリットが高いが，関係式において，不

表2. 1　見積り手法の特徴

分類	概　要	特　徴
① 類推法(10)	過去の類似プロジェクトの実績を基礎に見積る方法	長所 ・簡便で，初期見積りに適している。 短所 ・過去の実績プロジェクトについてその背景などのコンテキスト(制約，特徴など)が明らかでないと適用が困難である。 ・過去のプロジェクトの関係者以外に説明するときに，客観性に欠ける。 前提条件 ・実績のデータベースが必要である。 ・短所を補うために，過去のプロジェクトの実施者が見積ることが望ましい。
② 積み上げ法(10)	プロジェクトの成果物の構成要素を洗い出し，それぞれに必要な工数などを見積って積み上げる方法	長所 ・同じ類型のシステムを繰り返し開発している組織において精度が高い。 短所 ・プロジェクトの構成要素としての作業項目をWBS(Work Breakdown Structure)，成果物の構成要素としてのサブシステムやコンポーネントなどを事前に洗い出しておく必要があり，不確実なものが多い初期見積りではむずかしい場合がある。 ・構成要素の網羅度と厳密度で見積り精度が変わる。 ・類似のプロジェクトの関係者以外に説明するときに，客観性に欠ける。 前提条件 ・過去の実績に基づいて構成要素を分類・整理しておく必要がある。 ・それぞれに対して規模や工数などの実績データが必要である。
③ パラメトリック法	工数などを目的変数として，説明変数に規模や要因などを設定し，数学的な関数として表す方法	長所 ・再現性があり客観的である。 ・関係式の前提が明確である。 ・条件を変えることによりシミュレーションが可能である。 短所 ・関係式において不確定な変数があると見積りの誤差は大きい。 前提条件 ・実績のデータベースが必要である。 ・過去の実績データベースに基づいて関係式を定式化し，実証している必要がある。

(10)　詳細については，例えば，「見積りの方法」(真野，誉田著，日科技連出版社(1993年))参照。

確定な変数があると，見積りの誤差は大きくなる欠点があります。一般には，①，②および③を相互補完的に使用します。また，いずれの見積り方法でも実績のデータベースが必要です。

実績のデータベースを効果的に活用し，定量的なアプローチを見積りの活動で実践するためには，③のように実績に基づいた関係式を定式化して利用することをぜひ行うべきです。開発プロジェクトの作業に影響を及ぼすと考えられるもの(例：規模，新規性，難易度など)をあらかじめ洗い出しておき，実績データを蓄積して，概算的な見積り式を用意することです。見積り値と実績値との差異分析において特に威力を発揮します。

表2. 2には，見積り対象とそれぞれに対する見積り手法例を示しています。ソフトウェア開発以外のものは，作業ベースまたは製品ベースの積み上げが，現在のところ妥当な方法と考えられます。ただし，パラメトリックの手法の場合と同様，実績データを蓄積して，見積りに活用可能なように分析しておく必要があります。

見積り手法を選択する際には，見積り対象を見積るための尺度として，何がふさわしいかを決定します。尺度は，規模であれば，ファンクションポイント，ソースコード行数，オブジェクト数，データ項目数，画面数・帳票数，文書量，その他，対象に応じて計測可能なものがあります。

表2. 2　見積り対象と見積り手法

見積り対象	見積り手法例	備　　考
調査企画	・類推法 ・積み上げ法	規模，新規性，ビジネスコンサルティング的な要素があるかどうかの考慮
設計・開発(ソフトウェア開発)	・類推法 ・積み上げ法 ・パラメトリック法	本ガイドブックで主に解説している対象
移　行	・類推法 ・積み上げ法	移行データ量，導入箇所などの数などを考慮
保守/運用	・類推法 ・積み上げ法 ・パラメトリック法	第3章 3.2.1項参照
付帯作業	・類推法 ・積み上げ法	作業内容の難易度や必要な人数などを考慮
機器など	・価格×数量 ・積み上げ法	ハードウェア，ソフトウェアの単価と個数の見積り。また，バージョンアップやパッチを当てる作業の可能性を考慮。

(3)　ソフトウェア開発に対する見積り手法

(a)　規模見積りの尺度　ソフトウェアの規模を測る尺度としては，現在広くソースコード行数とファンクションポイントが利用されています。機能要件からソースコード行数またはファンクションポイント数を見積ることになりますが，ファンクションポイント数は，機能に基づく計測方法が定められています(第3章 3.1.1項参照)。

(b)　見積り時期に応じた規模見積り　見積り時期に応じたデータ収集および見積り手法を検討しておく必要があります。すでに第1章でも述べたとおり，開発システムの特徴に応じて，プロジェクト初期の段階から明確になるデータがあり，それを活用して初期見積りを行い，プロジェクトが進むにつれて詳細化または明確になるデータを活用して，見積りを精緻化することになります。

現実に用いられる手法として，データモデルから規模を見積る方法と画面から見積る方法があります。

表2. 3にそれぞれの方法をまとめて示します。

表2. 3　見積り時期と規模見積り例

	システム化の方向性	システム化計画	要求定義	設　計	備　考
NESMA試算法など)(データモデルからの類推)	①類似システムのFPから開発するシステムのFPを類推する。	②抽象データモデルからFP超概算を求める。	③E-Rダイヤグラムのエンティティから FP試算を求める。 ④E-Rダイヤグラムのエンティティと DFDのバブルから FP概算を求める。	⑤ほぼ全体像が明確になるので，基本設計書の情報からフルセットのFPを求める。	第1章および 3.1.1 項参照
ファンクションスケール法(画面からの類推)	①類似システムの画面数からファンクションスケールを類推する。	②画面数から簡易ファンクションスケールを求める。	③画面数と複雑度から詳細なファンクションスケールを求める。		第2部第6章参照
その他，ジャステックにおける方法	①類似システムのソースコード行数から，規模(ソースコード行数，作成する設計書の量など)を類推する。	②データモデルや画面数から，規模(ソースコード行数，作成する設計書の量など)を類推する。	③要件定義書の記述量から，規模(ソースコード行数，作成する設計書の量など)を求める。	④ほぼ全体像が明確になるので，基本設計書の量から，規模(ソースコード行数，作成する設計書の量など)を求める。	第2部第9章参照

DFD：Data Flow Diagram

(c)　工数・コスト見積り　パラメトリックな方法は，第1章 1.3.4項に示したとおり，規模と工数・コストのベースラインの関係と変動要因を設定します。

(d)　工期見積り　パラメトリックな方法としては，第1章 1.3.5項に示したとおり，工数から見積るのが一般的です。

(e)　複数見積りによる相互検証　ソフトウェア開発において一般に「銀の弾はない」(用語解説参照)といわれるとおり，見積り手法にも絶対のものはありません。① 類推法，② 積み上げ法および ③ パラトメリック法の三つの方法で，相互検証することが，現実的な実践です(**図2.4**参照)。

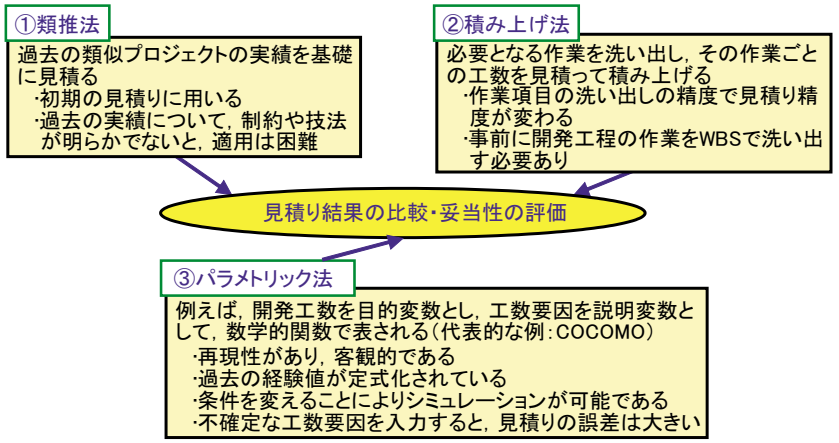


図2.4　見積り結果の相互チェックの例

(4)　プロジェクトデータ収集体制の確立

定量的な見積りの成功のためには，プロジェクトデータの精度の高い測定が前提となります。収集データが決定した後は，いかに実態を反映したデータが収集できるかがカギとなります。開発現場にデータの精度の重要性を理解してもらうとともに確実なフィードバックによるデータ収集のインセンティブを維持することや，データ収集の負荷を低減するため，ツールなどの利用の工夫を図ることが必要となります。後でも述べますが，データを単に収集するだけでなく，予測値と実績値の差異分析を行い，フィードバックの仕組みも確立しなければなりません。

なお，見落とされがちなものとして，予測値と実績値を比較分析するためには，見積り時点での記録(予測値とその前提条件など)の保持があるので留意する必要があります。

2.2.4　見積りの実践

確立した見積り手順に従い，システム規模，工数，工期などを見積ります。

(1)　要件の洗い出しと記述

第1章「合意できる見積りとは」でも述べたとおり，見積りに当たって生じるぶれの最大の原因は，要件(機能要件および非機能要件)が不明確であることです。全体のすべてが確定しなくても，ある程度確からしい部分的な情報が明らかになった時点で見積ることも，現実的には必要があります。そのためには，まず洗い出した要件を記述して，確からしさを評価して，見積りを行う上で必要な情報を定義します。これは，見積りにインプットする要件のベースラインを定義することになります。

(2)　規模見積り

見積りにインプットする情報は，見積り時期により変わります。見積り時期に応じて利用できるインプット情報に基づき，採用する見積り手法を選択して，見積ることになります。すでに述べたとおり，採用する見積り手法は，①類推法，②積み上げ法，③パラメトリック法の中から選択して，主として見積りに使用する手法と，見積り結果の比較・妥当性の評価に使用する手法とに区分けします。

(3)　変動要因の設定

規模見積り後または並行して，プロジェクトの特性やリスクなどに起因するベースラインからの変動率を変動要因ごとに設定(見積り)し，その総和を求めます。

(4)　工数・コスト，工期見積りの実施

上記の情報をインプットとして，組織として採用した見積りを実施します。このとき，複数の見積りの結果をに基づき，プロジェクト計画をシミュレートします。そして，それぞれの計画の良否を比較して，いずれを採用するか決定します。一方，大幅な違いが生じた場合は，必要に応じて，見直しを行うことになります。パラメータや分割した作業の個別の見積りなどをチェックします。

2.2.5　計画と実績の差異分析

次のプロジェクトの改善につなげるためには，個別プロジェクトで学んだことをまとめ,将来にその知見を活用できるように整備しておくことが重要です。見積り活動では，見積り結果(計画)と実績値との間の差異は，数値として明確にわかるものであり，その原因を探り，差異の発生を次回から抑える方策を検討することが肝要です。

（１）　計画と実績の差異分析のためのプロセスの確立

差異分析を行うためには，実施した見積りをいつ振り返り，どういう知見を得，それをどこに活かすのかを整理し，見積りと差異分析のプロセスを体系的に確立する必要があります。差異分析を行う時期は，プロジェクトの進行中とプロジェクト終了時の２種があります。

プロジェクト進行中は，その進行に伴って，リアルタイムに計画と実績の差をチェックし，見積り時に設定した変動要因の発生状況を把握することで，当初見積りの誤差を補正し，見積り自体をブラッシュアップします。

ブラッシュアップした見積りは，当初の見積りと同じく，その時点から先のプロジェクトマネジメントのインプットとして使用します。こうした活動は，プロジェクトのマイルストーンにあわせて再見積りとして実施することも，日常的なプロジェクトマネジメントとして実施することもあります。組織的なプロセス確立の観点から考えると，再見積りの実施時期を組織の規定として定めることが考えられます。また，後の差異分析で使用できるように，記録すべきデータと保管すべき情報を定めておくことも必要です。

プロジェクト終了時の差異分析では，プロジェクト進行中に実施した見積りにおける見積り誤差の把握と，その原因の分析を行います。見積り誤差は，単純に見積り値の変動幅だけでなく，見積った規模，工数算出時に使用したベースラインの値，見込んだ変動要因とその変動率，前提とした事項の妥当性(前提とした事項で間違っていたことは何か。なぜ間違ったか)など，見積り手順の個々の要素について誤差を把握し，誤差の発生原因を追究することが必要です。また，プロジェクト終了時の差異分析では，プロジェクトの実績をプロジェクトデータとして保存し，将来の見積り時に参照できるようにするとともに，他のプロジェクトの実績データとあわせて統計分析し，ベースラインの妥当性の見直しも実施します。

プロジェクト進行中の差異分析をどう設定するかは，各組織の性質によってさまざまですが，プロジェクト終了時の差異分析は，どのような組織でも最低限実施する必要があります。

これらの詳細なプロセスを示したのが**図2.5**です。

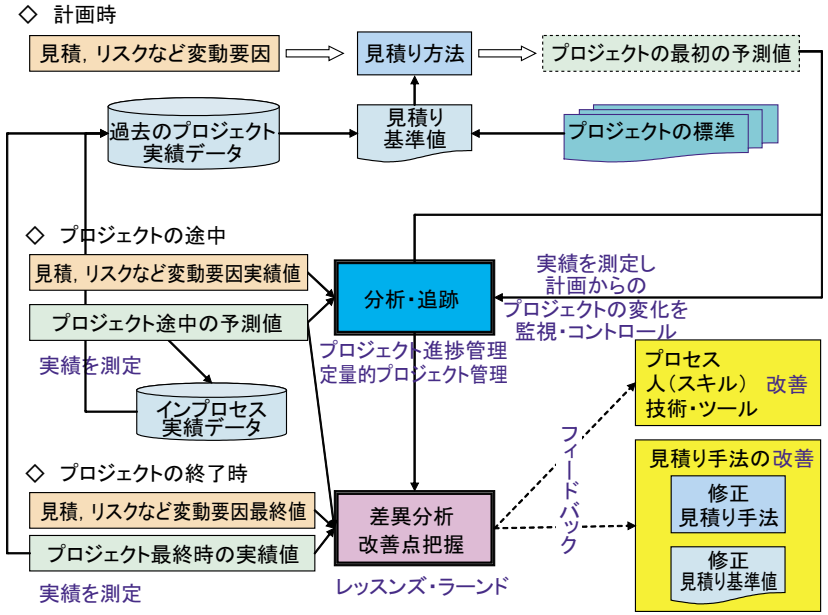


図2.5　組織的な取組みの例

（２）　分析可能なデータ精度の確保

見積り手法を考える上で重要な視点として，見積り値と実績値を比較して差異があった場合に，どうしてその差異が生じたのかが分析可能であることが重要です。

例えば，積み上げ法の場合を考えてみましょう。積み上げに分割したものがあった場合に，それぞれの根拠を見積り時に設定するわけですが，実績において全体が違っていた場合に，どの部分に違いがあったのかを検証できるかどうかが重要です。積み上げ時の分解の細かさと同じレベルの実績データを取得できることが前提となります。WBSなどに分解して，見積りを行ったときに，原因分析には，個々のWBSの内容が，妥当だったかどうかを確認するためには，実績値としても，見積りに使ったデータと同じ分類のものが，取得されていな

ければ検証は不可能です。

パラメトリック法の場合でも同様です。ベースラインと変動要因から基本的に見積りを求めますが，設定した変動要因がどうだったかのレベルまで実績値で確認できることが必須です。

見積り手法で利用したデータに応じた，データ測定・分析の仕組みが，プロジェクト遂行時においても構築されている必要がわかります。

2.2.6　差異分析結果のフィードバック

差異分析結果に基づいて，改善対策を検討し，展開します。**図2.6**に示すとおり，差異分析の結果，反映される対象として，個別プロジェクトに対するフィードバックと見積り手順(見積り手法を含む)そのものに対するフィードバックの両者があります。このようにして，適正な評価を行い，改善を続けることで，開発現場の協力を確実なものとすることができます。

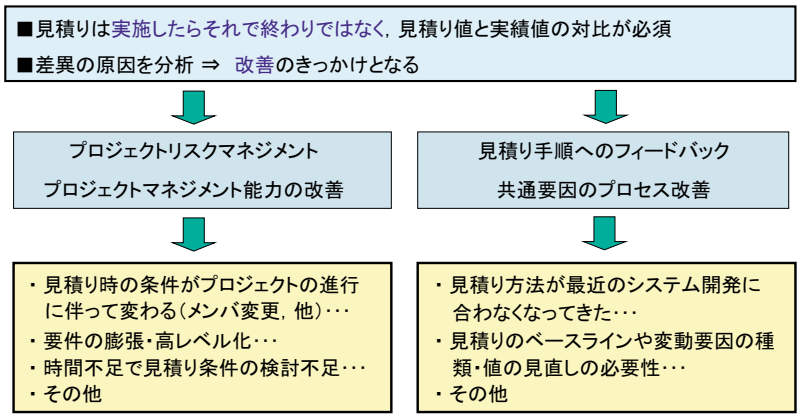


図2.6　差異分析からのフィードバック

(1)　プロジェクトマネジメントへのフィードバック

見積り手法は妥当であったが，プロジェクト(プロセス，人，技術・ツール)やプロジェクトマネジメント，見積り実施者のスキルに問題があり，実績値と見積り値に大きな乖離があった場合。

(2)　見積り手法へのフィードバック

開発を取り巻く環境変化のため，見積り手法(モデル)自体が実態に合わなくなった場合。

2.2.7　共通要因に基づくプロセス改善

複数のプロジェクトで共通の課題，すなわち，組織としての課題は，見積りの繰り返しから見えてきます。例えば，どのプロジェクトでも，チームやプロジェクトマネージャのスキルが大きくぶれ幅に影響を与える状況であれば，特定のプロジェクトではなく，組織横断的に人材の教育を考慮する必要があります。

共通要因に基づくプロセス改善とは，そのような見積り活動の結果から見える組織全体の共通要因について，組織全体で改善するための対策を講じて，改善します。なお，この活動自体は見積り活動ではありませんが，見積りのぶれ幅の減少のために必須な組織的活動です。

2.3　見積り活動に関する組織成熟度

2.3.1　見積り成熟度の考え方

前項で示したとおり，見積りの精度向上は，差異分析とその結果のフィードバックを通して実現することが本質であることがわかります。精度向上を図るためには，見積りの実施(Do)，結果のチェック(Check)および差異があった場合の対策(Act)のサイクルを回すことが必須です。

一方，前項に示した見積りにかかわる活動すべてを，どの組織でもすぐに始められるものではないことも明らかです。組織としての活動が確立していない場合，上記のPDCAの活動を確実にまわすのはむずかしく，ある程度一貫性のある組織的な活動が確立しており，実践され，体制が維持されている必要があります。そのように考えると，自然と見積りの精度向上は，組織的な成熟度と関係があることが理解できます。

図2.7に，見積りから見た組織の段階的な成熟度(Estimation Maturity Level 以下，EMLと略記)を示します(11)。図にも示していますが，「再現性のある見積り」までが個々のプロジェクトごとの見積り手法の確立に主に関係し，EML３以上は組織として一貫した見積り手法を確立し，その見積り精度向上のためのプロセス・体制の確立に関係します。**表2.4**には，それぞれの段階の概要をまとめています。

(11)　見積りから見た組織の段階的な成熟度(見積り成熟度：Estimation Maturity Level)とは，本ガイドブックで提唱する考え方です。

この考え方は，CMMIにおける５段階の成熟度の概念[12]が，ソフトウェア開発に限らず，成熟度を判断する上で非常に汎用的な考え方であることから活用したものです。見積りに関する組織の成熟度が向上することをCMMIの成熟度の概念で整理したものです。

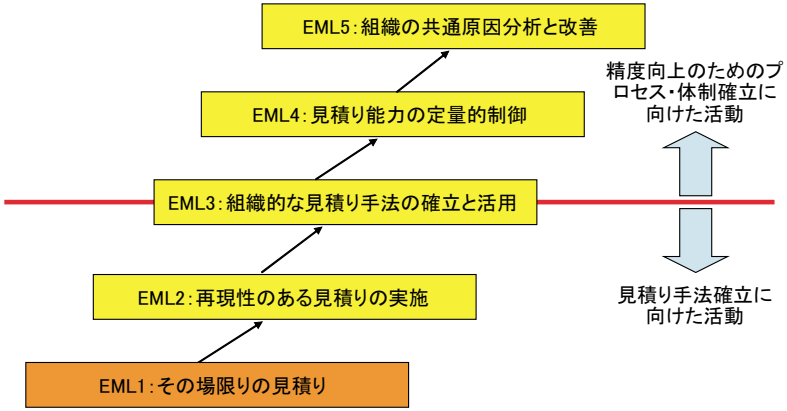


図2.7　見積り成熟度

表2.4　見積り成熟度の概要

EML	タイトル	説　　明
1	その場限りの見積り	プロジェクトマネージャなどが自らの直感に基づいて，見積りを行っている状態（属人的で再現性がない）。
2	再現性のある見積りの実施	プロジェクトマネージャが過去に経験したプロジェクトデータまたは経験的知識を収集・蓄積して，手順化し見積りを行っている状態（反復可能であるが，経験のないプロジェクトには適用できない）。
3	組織的な見積り手法の確立と活用	複数の有識者の知見に基づき組織的な見積り手法を確立し，継続的に活用するための体制（一貫したデータ収集，教育・普及，ツール）を整え，組織全体が統一された見積り手法を活用し，改善している状態。
4	見積り能力の定量的制御	プロジェクトの実行中に見積り能力を定量的に計測および監視し，プロジェクトの実績を許容範囲内に収めるようにコントロール可能な状態。また，見積り値と実績値との差異分析などを通して，プロジェクトの改善につなげるとともに，定期的に組織的な見積り手法を改善している状態。
5	組織の共通原因分析と改善	組織内のプロジェクトの見積り結果（ベースラインおよび変動要因）の分析に基づき，組織に共通の強み・弱みを把握し改善目標を定量化し，定量的に改善を制御している状態。

また，見積りに関する成熟度が向上するイメージを**図2.8**に示します。見積りに関する組織成熟度が向上するに従って見積り幅が狭まり，誤差も減少していきます。

なお，見積り成熟度の考え方を示したのは，各組織において，見積り活動の向上を図る上で，現状がどのような段階であり，次のステップに進むには，何をすればよいのかというガイドを示すことが目的です。見積り能力を評定することが目的ではありません。また，新たなモデルを構築することを行おうとするものではなく，2.3.2項「見積り成熟度の概要」に示すとおり，CMMIなどのすでに確立されたモデルを参照して，見積りに関連する活動をピックアップし，何をするべきかの指針を示すものです。

2.3.2　見積り成熟度の概要

ここでは，前項で紹介した成熟度の考え方に基づいて，見積り手法の構築と精度向上に向けての具体的な活動内容を示します。**図2.9**は，見積り成熟度とCMMIのプロセス領域とを対比させたものです。この対応に基づいて，各見積り成熟度の各段階は，どのような状態であるのか，また，次のステップに進むには，どのような活動をするべきか概観します。

（１）　レベル２（EML２）

EML２は，チームや個人ベースで過去に類似したプロジェクトについては，再現性のある見積りが実現している状態です。手順を文書化し，手順に従い実践し，結果を評価してフィードバックしている状態です。EML1との決定的な相違は，手順を文書化し，手順に従って実践していることです。EML３との決定的な相違は，PDCAサイクルがプロジェクト内にとどまっていることです。

---

(12)　CMMIにおける成熟度に関する成熟度レベルは，次のとおりです。
- 成熟度レベル１（初期）：通常，プロセスは場当たり的で無秩序である。
- 成熟度レベル２（管理された）：要件が管理され，かつプロセスが計画され，実施され，測定され，そして制御されることを確実なものにしている。
- 成熟度レベル３（定義された）：プロセスは，特性が十分に明確化され理解され，そして標準，手順，ツールおよび手法の中で記述されている。
- 成熟度レベル４（定量的に管理された）：組織およびプロジェクトは，「品質およびプロセス実績の定量的目標」を確立し，プロセスを管理する基準として使用する。
- 成熟度レベル５（最適化している）：組織は，プロセスに固有な変動の共通原因に関する定量的な理解に基づいて，プロセスを継続的に改善する。

なお，CMMIは，アメリカカーネギーメロン大学のサービスマークです。

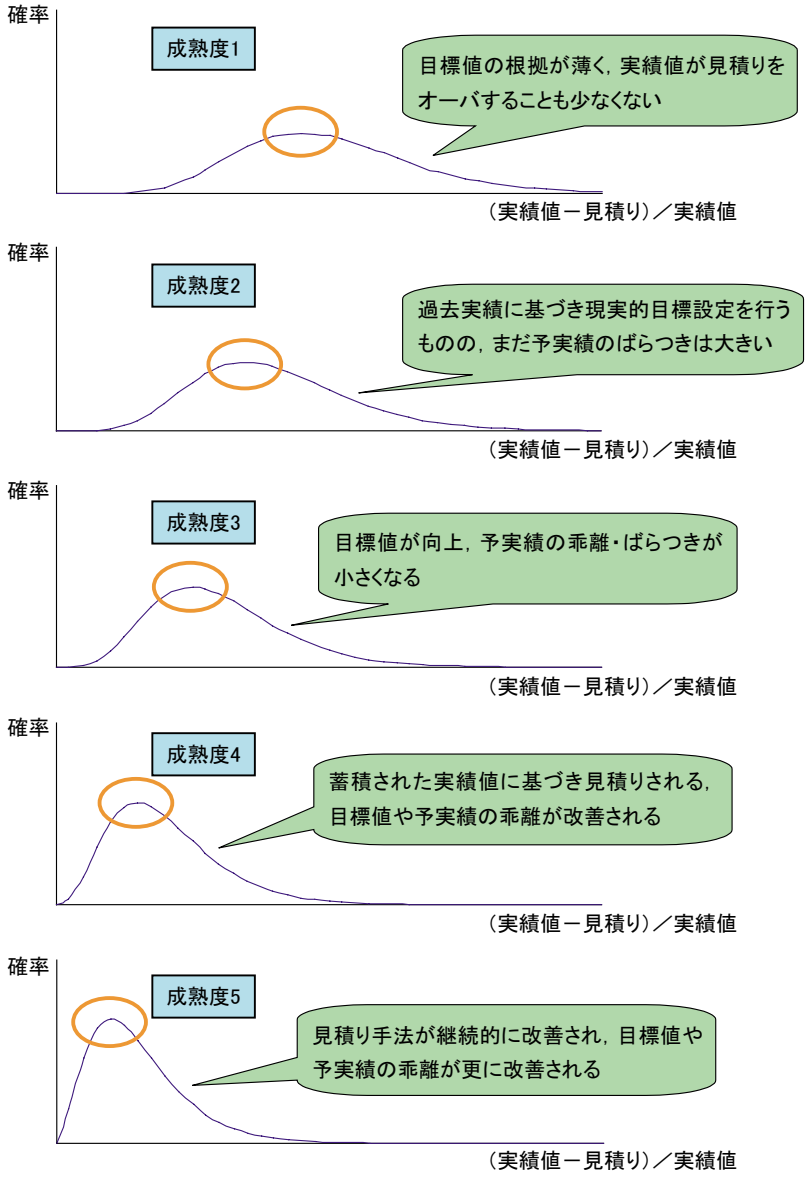


図2.8　成熟度レベルごとの見積り能力の向上のイメージ

関連するプロセス領域 ＼ 見積り成熟度		ML2				ML3							ML4		ML5	
		要件管理	プロジェクトの計画策定	プロジェクトの監視と制御	測定と分析	要件開発	組織プロセス重視	組織プロセス定義	組織トレーニング	統合プロジェクト管理	リスク管理	決定分析と解決	組織プロセス実績	定量的プロジェクト管理	組織改革と展開	原因分析と解決
EML1	その場限りの見積り															
EML2	再現性のある見積りの実施	○	○	○	○	○					○					
EML3	組織的な見積り手法の確立と活用	○	○	○	○	○	○	○	○	○	○	○				
EML4	見積り能力の定量的制御	○	○	○	○	○	○	○	○	○	○	○	○	○		
EML5	組織の共通原因分析と改善	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

（備考）EML: 見積り成熟度レベル；ML:CMMI 段階表現の成熟度レベル

図2.9　見積り成熟度と関連のあるプロセス領域（例）

CMMIで対応するプロセス領域としては，「要件管理」および「要件開発[13]」，「プロジェクトの計画策定」，「プロジェクトの監視と制御」，「測定と分析」および「リスク管理[14]」です。**表2.5**に各プロセス領域と見積りにかかわる活動を示します。

（２）　レベル３（EML３）

EML３は，各プロジェクトで使用している手順を統合・標準化して，組織的に見積り手法を確立して活用している状態です。EML２との相違点は，組織として首尾一貫したプロセスを定義して実践している点です。**表2.6**に示すとおり，組織プロセスに関連するプロセス領域に，フォーカスが当たっています。

EML３で新たに実装するプロセス領域には，大きく分けて，組織としての見積りプロセスにかかわるプロセス領域（「組織プロセス重視」，「組織プロセス定義」），プロジェクトマネジメントにかかわるプロセス領域（「統合プロジェク

(13)　CMMIでは，「要件開発」はレベル３に位置づけていますが，要件を明確にすることの見積りにおける重要性に鑑みて，実質的には，この二つのプロセス領域は，表裏一体の関係にあり，レベル２の段階から重要視されるべきものと考えて，見積りの成熟度では，レベル２に対応させています。

(14)　CMMIでは，「リスク管理」はレベル３に位置づけていますが，これはSW-CMMとの整合性を図るための処置の意味合いが強いものと認識して，見積りの成熟度では，レベル２に対応させています。

表2.5　EML 2 における活動例(レベル 2)

プロセス領域	見積りにかかわる活動(プラクティス)例
要件管理	見積りの前提となる要件を文書化して特定し，要件の変更を識別して，適宜見積りに反映する。 「要件管理」は，見積りの主要なインプット情報である要件の管理を行う最も重要なプロセスである。要件の管理は最初に実現されなければならないプロセスである。 【関連する見積り活動】 (1)見積りの実践ー①要件の洗い出しおよび記述
要件開発	ユーザの目標を満たすために必要な要件を引き出し，見積りのインプットとなる要件(機能要件および非機能要件)の漏れや誤りを予防する。 【関連する見積り活動】 (2)見積りの実践ー①要件の洗い出しおよび記述
プロジェクト計画策定	見積りの手順を定めて,定めた手順に則りプロジェクトの範囲･内容を明確にして，客観的かつ論理的根拠に基づき見積り，その結果を根拠とともに文書化して，プロジェクトの利害関係者の間で見積り内容を合意し共有する。見積り結果は，「プロジェクトの監視と制御」における目標設定に活用する。 【関連する見積り活動】 (1)見積り手順の確立 (2)見積りの実践
プロジェクトの監視と制御	計画に照らしてプロジェクトを監視しプロジェクトをコントロールする。また，必要に応じて見積りを変更する。場合によっては，見積り手法を変更することもある。 ・プロジェクトの進捗に従って，見積りと実績の差を監視し，計画を達成するようにプロジェクトをコントロールする。 ・プロジェクトの進捗，実績，課題を定期的にレビューする。 ・レビュー結果に基づき，必要に応じて見積り内容を変更する。 ・プロジェクトの実績を，再利用可能な状態に整理して保管する。 【関連する見積り活動】 (3)計画と実績の差異分析 (4)差異結果のフィードバック
測定と分析	見積りおよびプロジェクトの監視に必要なデータを収集，分析し，プロジェクト管理に提供する。 ・プロジェクトデータの収集・分析計画を作成する。 ・規模，工数見積りの基となるデータを収集・分析し，提供する。 ・見積りの妥当性を確認する。 ・プロジェクトの監視利用するデータを収集・分析し，提供する。 【関連する見積り活動】 (1)見積り手順の確立 (3)計画と実績の差異分析 (4)差異結果のフィードバック
リスク管理	見積り(プロジェクトの計画)のリスクを把握・監視して，コストオーバなどを防止する。プロジェクトの計画をプロジェクトの範囲・内容を明確にして，客観的かつ論理的根拠に基づき策定することにより，リスク源の把握を容易にし，さらに，網羅性を高めることができる。 【関連する見積り活動】 (2)見積りの実践 (3)計画と実績の差異分析

(注) 表中の【関連する見積り活動】の番号(1)～(5)は,2.2節 2.2.2項の番号に対応

表2.6　EML 3 における活動例(レベル 3)

プロセス領域	見積りにかかわる活動(プラクティス)例
組織プロセス重視	組織として首尾一貫したプロセスが引き続き有効であることを保全するために，組織のニーズに基づき，組織レベルで活用されるプロセス(見積り手法も含む)を改善する。 ・組織目標にかなう見積り手法を維持する。 ・定期的かつ必要に応じて見積り手法を評価し改善する。 【関連する見積り活動】 (1)見積り手順の確立 (2)見積りの実践 (3)計画と実績の差異分析 (4)差異結果のフィードバック (5)共通要因に基づくプロセス改善
組織プロセス定義	組織として首尾一貫したプロセス(見積り手法を含む)を確立する。また，見積り手法は，組織レベルで蓄積したデータを活用し確立している。 ・プロジェクトの実績値を組織の資産として活用できるようにする。 ・組織レベルで活用されるプロセス(見積り手法を含む)を確立する。 ・プロジェクトの特性にかなうように，プロセスを調整する基準を保持して，見積り手法を調整する。 【関連する見積り活動】 (1)見積り手順の確立 (3)計画と実績の差異分析 (4)差異結果のフィードバック
組織トレーニング	組織として首尾一貫したプロセス(見積り手法を含む)の習得・活用のためのトレーニングを行う。 【関連する見積り活動】 (1)見積り手順の確立 (2)見積りの実践
統合プロジェクト管理	組織として首尾一貫したプロセス(見積り手法を含む)をプロジェクトに適用するとともに見積りに利害関係者の関与を得る。 ・組織レベルで共有するプロセスおよび見積り手法をすべてのプロジェクトで活用する。 ・組織で共有する実績データを見積りに活用する。 ・プロジェクトメンバ，顧客など利害関係者が見積りに関与し，見積り内容を共有し，必要に応じて見直すなどの調整をする。 【関連する見積り活動】 (2)見積りの実践
決定分析と解決	プロジェクトにとって重要な判断を公式な判定基準に基づき比較検討し，決定する。 ・見積り時期，類似データの有無，組織やプロジェクトのおかれた状況などを総合的に判断して，複数の見積り手法から最も適したものを採用する。 ・要件や実装方法の重要な判断を，組織やプロジェクトのおかれた状況などを総合的に判断して，複数の案から最も適したものを選択する。 【関連する見積り活動】 (2)見積りの実践

(注) 表中の【関連する見積り活動】の番号(1)～(5)は,2.2節 2.2.2項の番号に対応

ト管理」)トレーニングにかかわるプロセス領域(「組織トレーニング」)があります。「決定分析と解決」は，プロジェクトにとって重要な判断の精度を高めるもので，複数の見積り手法または見積り結果に対する判断プロセスや要件や実装方法の判断プロセスにかかわり，判断ミスによる要件のぶれやリスクの判断ミスを防止することになります。

(3)　レベル4(EML4)

EML4は，プロセスの能力(見積り能力を含む)を定量的に把握し，これを利用してプロジェクトおよびプロセスを定量的に監視・制御して，プロジェクトの目標ならびにプロセスの目標を確実に達成できている状態です。EML3との相違点は次の2点です。

ひとつは，定義した組織として首尾一貫したプロセスの能力を安定させて定量的に把握していることであり，もうひとつは，この定量的な能力特性に基づいて，プロジェクトおよびプロセスの監視を定量的に行うことにより，標準化および効率化し，目標の達成確率を高めていることです。

EML2およびEML3の段階で，すでに予実分析を通して，見積り能力の向上を図ることを実施していますが，その活動の集大成として，能力(誤差など)を定量的に安定するまで精錬し，これを活用して予実分析を標準化して効率化している状態です。この状態では，プロジェクトの実績として正確な測定値が収集され，統計的に分析され，組織の共有データベースとして活用可能になっています。すなわち，経験・知見などの定性的なナレッジ等は，定量的なデータに基づく定量的な特性と結び付けて整理されており，経験・知見などの定性的なナレッジなどの活用による改善度合いの予測を実現し，その結果，見積りの予測可能性が格段に高まっています。見積り手法は，プロジェクトの進行中において，プロジェクトの結果を予測する目的にも利用されます。

表2.7に関連プロセス領域と見積りにかかわる活動の概要を示します。

(4)　レベル5(EML5)

EML5は，組織の共通原因分析と改善(戦略的改善を含む)を定量的に計画し，監視・制御して，確実に実現している状態です。見積り活動の観点からみたEML4との相違点は次の2点です。

ひとつは，組織の首尾一貫したプロセスの改革(戦略的改善)を，その効果を定量的に予測して目標設定し，効果を実証した後に，組織内に展開して，確実に効果を獲得することであり,その過程で組織的に試行することも含まれます。

表2.7　EML4における活動例(レベル4)

プロセス領域	見積りにかかわる活動(プラクティス)例
組織プロセス実績	組織の首尾一貫したプロセス(見積り手法を含む)の能力(見積り能力でいえば誤差など)を定量的に安定するまで精錬して，その能力を定量的に把握する。 ・見積り能力を測定する適切な尺度(例：予実績乖離など)を決定し，プロセスを改善・調整して安定させる。 ・見積り能力(見積り活動にかかわわるプロセスを含む)のベースライン(現在の能力)を確立(安定させること)する。結果として，見積り能力は予測可能となる。 【関連する見積り活動】 (2)見積り手順の確立 (3)計画と実績の差異分析 (4)差異結果のフィードバック (5)共通要因に基づくプロセス改善
定量的プロジェクト管理	プロセスの定量的能力に裏づけされた見積り手法に基づき，プロジェクトの見積りの妥当性を高めるとともに，プロセスの定量的能力特性に基づき，プロジェクトの監視を定量的に行うことにより標準化および効率化し，プロジェクトマネジメントの状況を把握し，改善につなげる。 ・達成可能なプロジェクトの目標を設定する。 ・目標の達成状況を定量的に管理し，プロジェクトの実績を見積り値の許容範囲内に収まるようにコントロールする。 ・統計的管理に基づき適宜，見積り手法を是正・改善し，関連データを蓄積し，プロジェクトで発見した経験・知見などの定性的なナレッジなどとともに組織に提供する。 【関連する見積り活動】 (1)見積りの実施 (2)計画と実績の差異分析 (3)差異結果のフィードバック (4)共通要因に基づくプロセス改善

(注) 表中の【関連する見積り活動】の番号(1)～(5)は,2.2節 2.2.2項の番号に対応

戦略的改善とは，組織内で見積り活動を実践した結果ではなく，組織の戦略的な目標やニーズに基づいて，新たな見積り手法を導入することや，新たな開発技術を導入することを想定しています。

もうひとつは，プロセスおよびプロジェクトを実施した結果から組織の共通課題およびプロジェクトの共通課題を抽出して,問題の解決を定量的に計画し，監視・制御して，確実に実現することおよびその結果を組織全般にわたる継続的改善へのインプットとすることです。

改善効果の定量的な測定および監視は，プロセスの安定した定量的能力特性に基づいているので，EML4の次の段階に位置づけられています。

見積り手法は，ソフトウェア開発プロセスの定量的な能力特性に基づいています。したがって，見積り活動は，見積り手法の改善だけでなく，ソフトウェ

ア開発プロセスの改善と密接にかかわっていますので，EML５は見積り手法にとどまらずこれを利用して，ソフトウェア開発プロセスの定量的な組織的改善を実現している段階と理解してください。

表2.8に関連プロセス領域と見積りにかかわる活動の概要を示します。

表2.8　EML５における活動例（レベル５）

プロセス領域	見積りにかかわる活動（プラクティス）例
組織改革と展開	組織の首尾一貫したプロセスの改革（戦略的改善）を，定量的に計画し展開して，展開状況を監視・制御して，改善の効果を測定・評価する。 ・組織の首尾一貫したプロセスの改革（戦略的改善）計画を立案する。 ・見積り能力を向上させるための改善計画を立案する。 ・改善策を実施，その効果を定量的に評価する。 【関連する見積り活動】 （５）共通要因に基づくプロセス改善
原因分析と解決	見積りの変動要因の傾向やプロセスやプロジェクトの予実差異の原因について定量的能力特性を活用して分析し，プロジェクトや組織内におけるプロジェクトマネジメントなどのプロセスの共通課題を発見し，プロジェクトの問題解決や見積り手法などの組織のプロセスそのものの改善に活用する。 ・見積りの変動要因の実績の分析や予実績乖離の原因の分析を通して，組織の共通課題を抽出し，対策を講じる。 ・是正されたことを定量的に評価する。 【関連する見積り活動】 （３）計画と実績の差異分析 （４）差異結果のフィードバック （５）共通要因に基づくプロセス改善

（注）表中の【関連する見積り活動】の番号（1）～（5）は，2.2節 2.2.2項の番号に対応

2.3.3　見積り成熟度向上における組織サポートの重要性

見積り能力の向上をめざすには，個々のプロジェクト活動としてとどまるものではなく，最終的には組織的な活動として定着させることが目標ですから，最初から組織的に取り組むことが望ましいことになります。成熟度をEML２に上げてから組織的な取り組みを開始するという順序ではなく，まず経営陣のサポートおよびバックアップする組織・体制を整えてください（**図2.10**参照）。プロセス改善で強調されるとおり，組織活動で実際の改善効果を出すためには，経営層のコミットメントが必須となります。次に，組織として一貫した見積り手法の導入および組織の見積り活動を実装します。

また，見積り手法には初歩的な手法から定量的な見積り手法までさまざまな手法があります。いきなり高度な見積り手法を導入すると，見積り手法の実装に必要な実績データがない場合や見積り手法を適用する各プロジェクトマネージャなどが使い切れないことが多く，結果的に形骸化することになり，かえって逆効果になることがよく見られます。各企業の見積りの成熟度に応じて適用可能な見積り手法は異なりますので，各企業の状況などに応じて始められるところから始めることが肝要です（図2.10参照）。

その上で，見積り活動に関する組織の成熟度の達成目標の設定にEML2，EML3，EML４といった見積り活動に関する組織の成熟度を活用して，向上を図る段階を追って取り組みます。

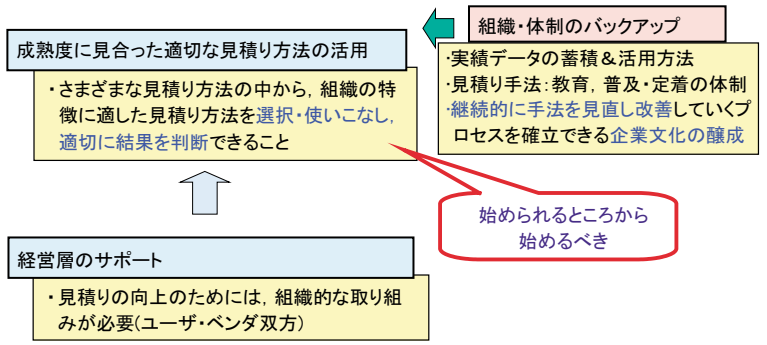


図2.10　体制・役割分担・企業文化

2.4　組織の状況に応じた見積り活動

見積り活動に関する組織の成熟度の各段階（EML２からEML５）は，組織の現状の状態を判断して，最終目標まで改善するための道しるべを設定することを目的としたものです。

2.4.1項では，見積りの成熟度（EML２からEML５）を参考にしながらマイルストーンを設定するための考え方を説明します。また，見積り成熟度の各段階に対応した見積り手法の確立度合いのイメージをまとめています。2.4.2項以降には，ステップアップの具体的な内容の紹介として，EML２からEML５の各段階で，見積り活動として実現すべき主要な事項を，段階のステップアップごとに説明します。

2.4.1　ステップアップの考え方

組織的な見積り活動をしていない状態は，見積り成熟度ではEML１と呼ん

でいます。まず，この状況から次のステップEML 2への移行は，組織内のプロジェクトが確実に見積り活動を実施することが定着するように，組織がバックアップすることが基本になります。次に，組織の見積り活動のPDCAサイクルを確実に実施することが定着するように組織として注力し，EML 3の段階に移行します。各プロジェクトをバックアップする体制と組織の見積り活動のPDCAサイクルを推進する体制とは重複する部分が多いので，大きな組織ではEML2, EML 3と段階を区切って実装することをお薦めします。

EML 3の段階で，見積り活動の実装は一通り完了しますので，後はEMl4, EML 5と順に見積り活動をレベルアップしていきます。EML 3の段階で，見積り手法の実装に必要な正確な実績データが十分にそろっていて，見積り手法で使用する関係式や，関係式に使用する各種係数(生産性など)の基準値のぶれ幅が，比較的小さい場合は，EML 4とEML 5に同時に取り組んでも混乱することはないでしょう。

以上の見積り活動のステップアップは，それぞれの見積り活動のレベル(プロセスおよびプロダクトの測定・データの収集プロセスなどの確立度合い)，

表2. 9　見積り活動の確立度合いの評価軸

評 価 軸		説　　明
プロジェクト特性への適応性		さまざまな類型のプロジェクトの特性にあうように調整可能な度合い。
見積り時期への適応性	上流工程	システム化の方向性など超上流工程および上流工程の時点で得られる情報に基づいて，見積りを行える度合い。
	下流工程	設計工程などの下流工程の時点で得られる情報に基づいて，見積りを行える度合い。
客観性(説明可能性)		プロジェクトの関係者(プロジェクトメンバ，ユーザなど)が理解できる用語を用いて，客観的に論拠と見積り結果とを関連付けて，説明できる度合い。
複数手法による評価・検証		複数の見積り手法を併用して，それぞれの手法の弱点を補完して，評価，検証している度合い。
工夫・改善の反映可能性		ユーザの工夫・改善，ベンダの工夫・改善を見積りの論拠として設定でき，見積り結果に反映できる度合い。
プロジェクトのコントロールへの活用可能性		プロジェクトの進行中に，見積り(プロジェクトの目標)達成できるか否かを確認するための項目(ドキュメント量，生産性など)の目標値を見積り結果として得られている度合い。
信頼性	誤差など	見積り手法自体の誤差の大きさ，実証の度合いなど。
	利用者の信頼度合い	見積り手法の有効性に対する利用者の信頼度合い。見積り手法の論拠と経験･知見などと合致度合い，実際の利用体験に基づく見積り手法の実証度合いなどが関連する。

見積り手順・手法の確立度合い(見積り手法のレベルなど)，見積り活動を推進する体制，教育(体制や内容)の観点からまとめることができます。**表2. 9**に見積り活動の確立度合いの評価軸を示します。また，この評価軸に基づいて，各見積り成熟度の各段階で実現する水準を概観したのが**表2. 10**です。

表2. 10　見積り成熟度と実現状況

ギャップ ＼ 評価軸			EML 2	EML 3	EML 4	EML 5
見積り手法の確立度合いのイメージ	プロジェクト特性への適応性		△(過去の経験に依存)	○(組織の経験は網羅)	◎(未経験のものでも予測可能)	◎(未経験のものでも予測可能)
	見積り時期への適応性	上流工程	△(過去の経験に依存)	○(実証性に難有)	◎(実証されている)	◎
		下流工程	○(偏りがある)	◎(網羅している)	◎	◎
	客観性(説明可能性)		△	○(説明可能な手法)	◎(実証されている)	◎
	複数手法による評価・検証		△(欠けている場合がある)	○(妥当性に難有)	◎(定量的に実証されている)	◎
	工夫・改善の反映可能性		△(客観性に乏しい)	○(実証性に難有)	◎(定量的に実証されている)	◎
	プロジェクトのコントロールへの活用可能性		△(客観性に乏しい)	○(実証性に難有)	◎(実証されている)	◎
	信頼性	誤差など	×(バラつき大)	△(安定性低い)	○(初期安定化)	◎(精錬している)
		利用者の信頼度合い	×(経験しないと信じられない)	△(理屈はわかるが半信半疑)	○(客観的に実証し説明されている)	◎(積極的に活用している)
体制			推進の中心人物および推進ワークショップの開催	組織的な推進組織の確立	同左	同左
教育内容			・見積りの重要性の理解 ・見積りに関連する各種の尺度およびその関連 ・見積り手法の利用訓練	・組織の首尾一貫した見積り手法の実証結果および利用訓練 ・データ収集・分析・活用の理解	・統計的手法のリテラシーの向上	——

2.4.2　EML１からEML２へのステップアップ

（１）　実現すべき見積り活動

EML１とは，見積り活動が必要に応じたその場限りのものにとどまっている段階です。EML１からEML２へのステップアップは，その状況からの脱却を指します。根拠のある再現性のある活動を実現することです。端的には「見積り手順の確立」を実現し，プロジェクト内部でPDCAサイクルを回して改善します。

（a）　見積り活動を推進する体制の整備　EML２の最初の段階では，次のステップをにらみながら，見積り活動を小さく回して，繰り返し可能な活動の基本的なことが実施されるように心がけます。必要に応じて，一時的な組織としてワークショップを開催し，調整するようにします。**図2.11**には，EML３へ進む前に実現しておくべき，見積りに関する活動の内容を示します。

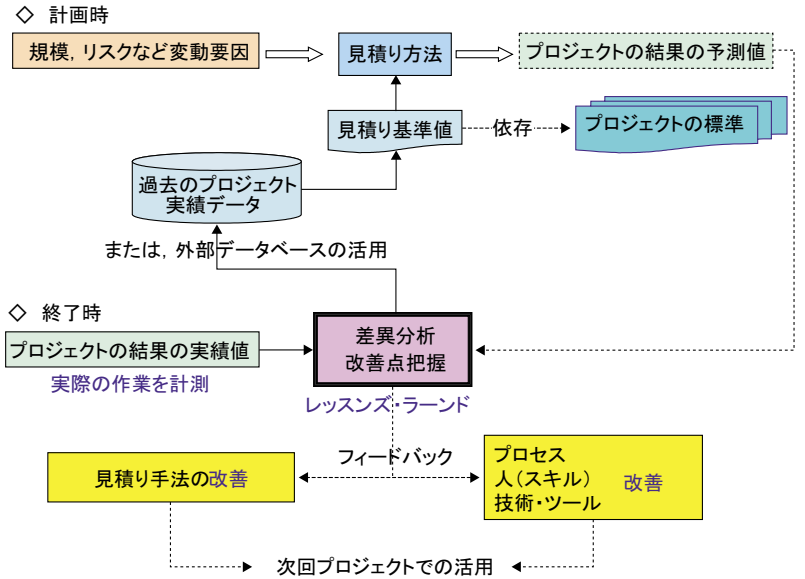


図2.11　見積り実施のための最低限のプロセス

（b）　見積り手順の確立　見積り手法は，プロジェクトの関係者の経験・知見を集約して，少なくとも各プロジェクトで一貫して使用する見積り手法および手順を文書化するとともに，プロジェクトの実績データを収集・蓄積し活用する体制および手順を定めます。

見積り手法および手順は，最初から組織で首尾一貫して使用するものと整合しているとなおよいですが，必須ではありません。可能な範囲で，取り組んでください。

（c）　見積りの実践

（ア）　要件管理プロセスの確立：見積りの妥当性を高めるために最も重要なこととして，要件を明確にする点を常に重視する必要があります。以上の活動の前提条件として，要件をできるだけ明確にすることがあります。

　要件管理・要件開発については，小冊子「経営者が参画する要求品質の確保」に示される事項をユーザおよびベンダなどの開発の関係者がいかに明確にしていくか，という点に尽きます。

（イ）　見積りの文書化および合意：見積りをプロジェクトマネジメントで活用するために，その根拠とともに文書化し，ユーザ，ベンダの利害関係者と合意して共有します。なお，見積りは，見積りのインプットである要件とともに管理して，要件の変更など，必要に応じて見直すなどの調整をします。

（d）　計画と実績の差異分析　EML２の段階では，最低限，見積り結果とプロジェクト終了時に判明する実績値との比較・分析を行い，プロジェクトの実績データベースに登録します。このために，あらかじめ計画と実績の差異分析を行うプロセスを定めて文書化してください。

また，プロジェクト途中においてモニタリングとコントロールの体制を整えて，途中段階で予定と実績の差異を分析・コントロールします。途中段階での予定と実績の差異分析は可能な限り定量的に行います。

プロジェクト遂行過程における予実分析を実現するためには，2つのことを実現する必要があります。ひとつは，プロジェクト遂行時におけるデータ収集のための仕組みの構築です。もうひとつは，見積り値と実績値が対応しており，比較可能であることです。

（e）　差異分析結果のフィードバック　プロジェクトの実績データが増加するに従って，これを活用して見積り手法を実証して修正するとともに，見積り結果の粒度を詳細にしていきます。自組織で，10件程度のデータを収集したところで，おおよその傾向は見えてくると思います。その時点で，収集するデータの種類を再検討します。ここで，再び一時的なワークショップを開催します。

（2）　ステップアップのための留意事項

（a）　見積り手法の初期設定　最初のステップは，実績データの蓄積などがない状況から定量的な見積りにつながる最初の段階として，初期の見積り手法の定義を行うことです。最低限集められるデータ(または情報)を集め，実施可能な見積り手法および見積り手法を適用する手順を確立することになります。見積り手法を文書化していなくても，プロジェクトマネージャは過去の経験と知見に基づいて，見積りを行っている場合がほとんどです。実績データの蓄積が乏しい場合は，プロジェクトマネージャの経験と知見で補い，見積り手法を設定します。その方法はいくつかあります。

（ア）　外部から既知の見積り手法を導入する：プロジェクトマネージャの経験と知見から見積り手法を設定するといっても，一から見積り手法を構築するのは大変な労力が必要です。このような場合に，外部から既知の見積り手法(COCOMO法など)を導入して活用しますが，プロジェクトで使いこなすには，プロジェクトの文化に応じて咀嚼して再定義しなくてはなりません。このため，見積り手法の枠組みにプロジェクトマネージャの経験や知見をマッピングして，日常プロジェクトで使用している共通の言葉で書き直します。また，プロジェクトマネージャの経験は，非機能要件などの影響度については部分的にしか捉えられていない場合には，既知の見積り手法の枠組みに当てはめることにより，欠落している経験を補完できる可能性があります。その上で，プロジェクトマネージャの過去の経験と照らし合わせて，まずは定性的に傾向が一致することを確認します。次に，プロジェクトマネージャが覚えている範囲でいくつかのプロジェクトのマクロな実績を当てはめて，見積り手法を調整します。

たとえば，プロジェクトマネージャが機能要件から概算で規模やコストなどを見積っていたとします。これを見積り手法として設定することを目標として，画面数・帳票数・データ項目数を収集します。これらは，機能要件のすべてを表す指標ではありませんが，第1章で説明したとおり，全体規模を見積もるために活用可能です[15]。このとき，検証するデータが部分的にしかない場合は,過去のプロジェクトデータから得られる傾向(全体に占める部分情報の割合)が重要ですが，自組織にはそのようなデータがないことが多いので，次善の策として，外部のプロジェクトデータベースの結果を活用することもできます。

規模，工数，コスト，工期間の関係は，基本的には回帰分析などを行って線形(正比例)の関係を最初のベースラインの関係として設定することができます。

さらに，見積り時期によっては，非機能要件などは，部分的にしかわからない場合もありますが，このような場合の取り扱い方は，見積り手法の利用方法として，見積りの手順に整理します。

（イ）　直感が鋭いプロジェクトマネージャの知識を利用して，見積りモデル作成する手法を活用する：組織として統一した見積り方法がまだ確立していない場合であっても，個々のプロジェクトマネージャが長い経験の間に直感とはいえ，見積りの鋭い勘を磨いている場合も少なくありません。このような場合は，これらの直感が鋭いプロジェクトマネージャの知識を利用して，どのようなデータを収集し，見積りモデルを作成すればよいかについて，CoBRA法のような見積りモデル作成手法を活用することが考えられます(SECでの見積り手法に関する実証実験の第1章参照)。

（b）　プロジェクトデータの収集・蓄積手順の設定　見積り手法の初期設定を行ったのち，見積り手法を自組織の実際のプロジェクトで活用して，その実績データを収集してプロジェクトの実績データベースを構築するとともに，これを分析して見積り基準値，見積り手法を修正したり，見積りのために参照したりします。

このとき，重要なこととして，以降，見積り手法にフィードバックして精錬していくために収集するデータ項目は，必要十分であること，見積りの参考とするためにプロジェクトの類型や特性を識別できること，などを検討し決定しておく必要があります。また，信頼できる正確な情報を収集する基盤を整えます。これを推進するために，一時的な組織として推進組織を設けて，どのようなデータを収集したらよいかを，組織内のワークショップ形式で実施することが，薦められます。

最初に構築する見積り手法が,実データによる裏づけが乏しいことなどから，見積り結果もプロジェクト全体で捉えるなど，粒度が粗くならざるを得ませんが，いずれは見積り結果をプロジェクトマネジメントに利用する，超上流工程

(15)　第1章では，早期の見積り段階での見積り方法として示しましたが，ここでの意味合いは，組織の成熟度が，まだそれほど高くない場合での簡易な方法としての活用が可能ということです。

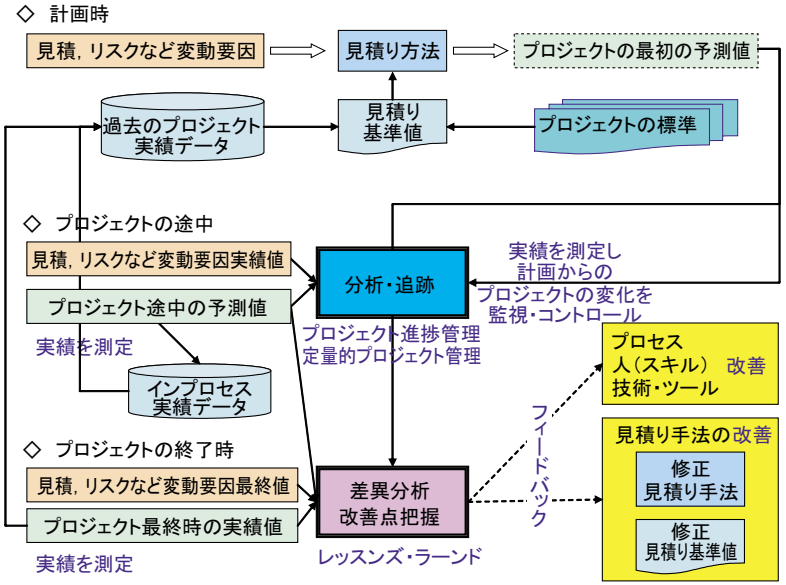
における見積り手法を整備する，などの最終的な目標を考えて，収集するデータ項目を決定し，収集し始めることです。

FPを例にすると，先々，上流工程でFPを推定することを念頭に，概算としてFPを推定するためのデータを収集し始めることです(1.3.3(３),3.1.1(４)参照)。また，プロジェクトに要求される非機能要件などで層別できるようにします。適用業種や業務などの分野を後で分類できるようにしておくことも重要ですが，このステップで重要なのは，あまり欲張らないことです。

2.4.3　EML２からEML３へのステップアップ

（１）　実現すべき見積り活動

EML２からEML３へのステップアップは，EML２の段階でプロジェクトごとに確立した見積り手法や手順を体系化するとともに整理統合します。また，見積り活動を組織的な活動(プロジェクトを横断した活動)にレベルアップして，組織のPDCAサイクルとして定着させます。



(注) 図2.5の再掲

図2.12　見積り実施のためのプロセス

(a)　見積り活動を推進する体制の整備　組織的に一貫したデータ収集体制をはじめとして，一貫した見積り活動の維持・改善をミッションとした推進組織の確立が必要となります。背景としては，組織的な見積り手法の確立には組織全体の状況の把握と統括する必要があることと，データの収集・分析および活用のためには，組織共有のデータベースを設置する必要があり，個別の部門または個人的な活動ではほとんど不可能であることがあります。

また，一貫性のある見積り活動を維持するために確立した見積りに関する背景知識，活用方法についての教育を実施する中心的な促進組織による首尾一貫した教育の提供が必要です。

(b)　見積り手順の確立

(ア)　見積り手法の体系化および整理統合：EML２の段階で確立したプロジェクトごとの見積り手法について，組織横断的な観点から体系化して，整理統合します。体系化の観点には，見積り対象，見積り手法の分類(類推法，積み上げ法，パラメトリック法など)，適用可能な見積り時期，適用可能なプロジェクトの特性などがあります。その上で，見積り手法を標準化して組織で一貫して使用するいくつかの見積り手法に統合します。この際，見積り手法をそれぞれのプロジェクトの特性にあうように適用するためのガイドもあわせて整備します。

(イ)　プロジェクトデータベースの確立：組織横断的な観点から，プロジェクト共通のデータ項目とプロジェクトの特性に応じたオプション的なデータ項目を明確にして，精緻なものにして，組織として一貫して利用するプロジェクトデータベースを構築します。あわせて，データを収集する安定した支援環境を用意し，プロジェクトのデータ収集の負荷に対して，十分な支援を整備します。

また，プロジェクト遂行時における差異分析に利用するデータ項目を整理して，プロジェクト遂行時における差異分析を実現します。

(c)　計画と実績の差異分析　組織内の全プロジェクト実績の統計的な分析により，ベースラインの関係および変動要因とその影響度を導き出し，さらに外部のプロジェクトデータベースとの比較などを実施して，基準値などの統計データ，その結果得られる知見をまとめて，組織内の全プロジェクトにフィードバックします。なお，統計データは，プロジェクトの類型などに応じて区分して，プロジェクトが利用しやすいようにします。

また，それぞれのプロジェクトが，類似のプロジェクトの実績を参照して，参考にできるように，データ収集・分析に中心的な役割を果たす組織を作り，プロジェクトの収集データの精度向上を図り，組織的な経験的な実績データが共通データベースに蓄積し，利用環境を整備する必要があります。

(d)　差異分析結果のフィードバック　EML 2では，一時的なワークショップを開催して，見積り手法を実証して修正するとともに，見積り結果の粒度を詳細にしていましたが，EML 3では専門組織を設置して推進します。

さらに，各プロジェクトに確実にフィードバックするために，一時的なワークショップを開催します。

(e)　共通的要因に基づく改善　プロジェクト実績データとあわせて，失敗事例，成功事例および見積り手法などに対する改善提案を収集する活動を行います。この結果および組織内の全プロジェクト実績の統計分析の結果から，プロジェクトに共通する課題およびその要因を特定して，改善策を立案して組織内に展開します。

この活動でも，専門組織を設置して推進し，組織内に確実に展開するために，一時的なワークショップを開催します。

2.4.4　EML 3 からEML 4 へのステップアップ

(1)　実現すべき見積り活動

EML 3 からEML 4 へのステップアップは，プロジェクトの実績を見積りした許容範囲内に収めるようにコントロール可能にすることに主眼がおかれます。EML 3 で，プロジェクトの進行過程での差異分析を実現していますが，EML 4 ではプロジェクトの実績を目標とする見積り値の許容範囲内に収めるようにプロジェクトマネージャがプロジェクトのモニタを通して，コントロールできる状況にします。

(a)　見積り活動を推進する体制の整備　データ収集・分析に中心的な役割を果たす組織を設け，収集データの精度向上を図る必要があります。

(b)　見積り手順の確立

(ア)　見積り手法の精錬：プロジェクトの実績データの統計的な分析により，見積り手法自体を精錬し，精度の向上(モデルの誤差の縮小)を図ります。

(イ)　プロジェクトデータベースの精錬：組織の共有データベースに，失敗事例，成功事例などの定性的なナレッジをその定量的効果とともに蓄積して，プロジェクトが利用できるようにします。

また，プロジェクトのベースラインなどの統計データに，ばらつき(誤差)を加えて提供します。

(c)　見積りの実践　マイルストーンごとに開発規模，工数，期間，投入人数などを設定することはもちろん，許容範囲の設定，プロジェクトを監視するための測定項目および限界値などを設定できる状況になっている必要があります。

(d)　計画と実績の差異分析

(ア)　定量的コントロールの実践：プロジェクトの途中で，プロジェクトを定量的に監視して差異原因の特定に活用し，処置結果の効果を定量的に監視して，プロジェクトをコントロールします。また，見積り手法を含む，プロジェクトの実績分析から得られた知見(パラメトリックな関係式など)を用いて，プロジェクトの途中の実績に基づいて，プロジェクトの結果を予測し，予防的に処置します。

(イ)　収集データ項目の精錬：EML 4 からEML 5 にステップアップする前に，プロジェクトから収集するデータ項目を整理統合して，スリムにします。収集データ項目の中には，EML 4 までステップアップするためには重要であったが，達成した後には，重要性が低くなるものもあります。これらをそぎ落とします。

(2)　ステップアップのための留意事項

組織として首尾一貫して，プロジェクトで作成するドキュメント，ソフトウェアの開発プロセスおよびプロジェクトマネジメントプロセスを定めて，これを守ることにより，これらを修正しながらプロジェクトの実績のぶれ幅を許容できる範囲内に収まるように精錬し，生産性や品質などを定量的に監視できるようにします。見積り手法自体も，プロジェクトの実績データの統計的な分析により，精度の向上を図ります。

また，見積り値は，マイルストーンごとに開発規模，工数，期間，投入人数などを設定できる状況になっている必要があります。

2.4.5　EML 4 からEML 5 へのステップアップ

（1）　実現すべき見積り活動

EML 5 では，EML 4 までに構築した仕組みを活用して，組織全般の活動としてCMMIで示す「組織改革と展開」および「原因分析と解決」を実現することです。見積り活動の視点からは，「組織改革と展開」の一環として見積り活動の改善，「原因分析と解決」の一環として見積り活動から得られる知見の活用の実現です。

（2）　ステップアップのための留意事項

見積り活動は，見積り手法の改善だけでなく，ソフトウェア開発プロセスの改善と密接にかかわっていますので，EML 5 は見積り手法にとどまらず，これを利用して，EML 4 の達成までに投下した投資を回収する段階と理解してください。

第3章　見積り手法の一般的事項と今後の課題

3.1　見積り手法の一般的事項

3.1.1　ファンクションポイントについて

（1）　ファンクションポイント（Function Point）法とは

ファンクションポイント法（以下，FP法と略す）とは，1979年にAllan J. Albrechtが提唱したソフトウェア規模の計測手法です。

FP法は，開発基盤に影響されないソフトウェア規模計測尺度を目指して開発され，データの入出力など，ユーザに見える機能に着目し，機能数とその重みにより，ソフトウェア規模を定量化します。

しばしば誤解されるところですが，FP法は工数や価格を見積ることを目的とした尺度ではありません。ソフトウェアの大きさ（規模）を測るための尺度です。しかし，例えば，車の大きさを表すのに，排気量や積載重量，車高など，さまざまな尺度があるように，ソフトウェアの規模についても多様な定義が可能です。FP法は，基本的にソフトウェアの機能量に着目して定量化する手法ですが，「機能」の範囲がどこまでか，人によって解釈が異なる可能性があります。そのため，尺度として広く流通するには，明確な定義と国際的な標準化が必要になります。

（2）　標準化動向

国際標準では，1998年「ソフトウェアの測定－機能規模測定[16]」という規格において，機能規模測定（Functional Size Measurement）の概念が定められました。この定義において，機能規模測定とは，「利用者機能要件を定量化して得られるソフトウェアの規模を計測すること」とされており，利用者機能要件とは，「利

(16)　ISO/IEC 14143　ソフトウェアの測定 － 機能規模測定
第1部：概念の定義　（ISO/IEC 14143-1:1998，JIS X 0135-1:1999）
第2部：ソフトウェア規模測定手法のJISX0135-1:1999への適合性評価
（ISO/IEC 14143-2:2002，JIS X 0135-2:2004）

用者の要求を満足するためにソフトウェアが実現しなければならない利用者の業務および手順」とされています。利用者機能要件は，利用者要件の一部であり，技術要件，品質要件などを除きます。したがって，機能規模測定法としてのFP法は，利用者要件の中の利用者機能要件だけを定量化するものといえます。

具体的な計測手法は，FP法の標準化団体として，IFPUG（International Function Point User Group）が1986年に発足し，計測マニュアルの改訂や計測技術の普及を行っています。IFPUGが定めた計測手法をIFPUG法と呼び，世界的に最も多く使用されています。しかし，IFPUG法も万能ではなく，適用困難な分野が存在するため，それを補うために，MKⅡ法，NESMA法，COSMIC-FFP法などの新しい計測手法が多数提案されています。これらの手法は，以下のように，個別に国際規格化が進んでいます。

ISO/IEC 19761：2003 COSMIC-FFP法
ISO/IEC 20926：2003 IFPUG法
ISO/IEC 20968：2003 MKII法
ISO/IEC 24570：2005 NESMA法

（3） IFPUG法の計測手法概要

現在，世界で，また，わが国でも最も多く使われているFP計測手法であるIFPUG法につき，その計測手法の概要を以下に解説します。なお，IFPUG法の計測手法の詳細は，IFPUG発行のFunction Point Counting Practice Manual（以下CPMと略す）に記されています。

（a） 機能の種別　IFPUG法で機能（function）として計測するものは，**図3.1**のようになります。

機能種別には，内部論理ファイル（ILF：Internal Logical File），外部インタフェースファイル（EIF：External Interface File），外部入力（EI：External Input），外部出力（EO：External Output），外部照会（EQ：External Inquiry）の5

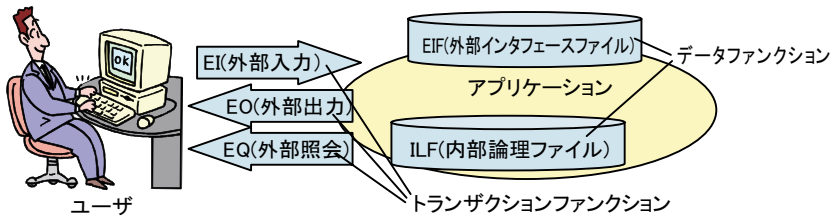


図3.1　ファンクションポイントの機能種別

種があります（以下，それぞれ略称で示す）。

このうち，ILFとEIFはデータファンクション（Data Function）であり，EI，EO，EQはトランザクションファンクション（Transaction Function）です。

データファンクションは，データに関する機能であり，アプリケーションソフトウェア（以下アプリケーションと略す）が使用するデータのまとまりを機能とします。ILFとEIFの識別は，アプリケーション内部で，そのデータが維持管理（データの追加，更新，削除など）されるかどうかで行います。維持管理されるものがILF，維持管理されず参照のみのものがEIFです。ここでいうファイルとは，「データのまとまり」の意味であり，物理的な電子ファイルの意味ではないことに注意してください。

トランザクションファンクションは，データ処理に関する機能であり，アプリケーション境界を超える一連のデータ入出力を機能とします。アプリケーション境界とは，そのアプリケーションの範囲を示す境界で，ユーザとアプリケーションとの境界，アプリケーションと他のアプリケーションの境界に設定します。例えば，「人事情報システム」や「給与計算システム」などの業務的なシステムの範囲が，アプリケーションの境界に該当します。こういうアプリケーションの境界を超えたデータの出入りが，トランザクションファンクションとなります。

トランザクションファンクションの種別（EI，EO，EQ）の識別は，処理の主目的・副目的によって行います。EIは主に外部からのデータ入力によるデータ更新（ILFの維持管理）が主目的です。EO，EQは共にユーザへの情報提供を主目的とします。EOとEQの違いは，なんらかの処理ロジックを通して情報提供をするものがEO，単純なデータ検索により情報提供をするものがEQです。

（b） 機能の粒度　機能規模測定において，1機能単位の大きさを粒度といいますが，その大きさ（粗さ）をどのくらいにするかが，その手法を特徴づけます。なぜならば，粒の大きさは機能数の多さに直接影響するからです。IFPUG法では，1機能の大きさは，1）ユーザの視点からみて，2）論理的な，3）業務活動としての最小単位，と定義されています。

データファンクションの場合，上記の3条件に基づき，「論理的なデータのまとまり」を1機能とします。すなわち，論理データ設計におけるエンティティなどが，1機能に該当します。

トランザクションファンクションの場合，同じく上記条件に基づき，要素処

理(elementary process)と呼ばれる「ユーザにとって意味があり，業務の最小単位」を１機能とします。ここでいう最小単位とは，「自己完結し，アプリケーションの業務を矛盾のない状態に保つ」範囲を指します。以下に機能として，計測するものと，しないものの例を示します。

【例】

- 機能として計測するもの：「新入社員登録」「社員情報検索」などの業務機能，「社員エンティティ」などの論理データ
- 機能として計測しないもの：「メニュー遷移」「入力用ダイアログBOX起動」など，業務として完結していない「社員情報画面」などの物理的な画面，「給与情報バックアップファイル」などの物理データ

上記のように，計測するのはあくまでも論理的な機能単位であり，物理的な画面や物理ファイルではありません。この特徴によって，IFPUG法で計測されたFP値は，技術要件に影響されず，言語やアーキテクチャが異なっても，機能規模(Functional Size)が同じであれば，同じ値を示すといえます。

(c)　FP値の算出　粒度の考え方に従い洗い出した機能の一覧を，FP値として定量化するためには，洗い出された個々の機能に対し，重み付けを行う必要があります。もちろん，１機能が１点でもよいのですが，個々の機能の重さに傾斜をつければ，より精緻な値を得ることができます。

IFPUG法では，重みは寄与度(contribution)と呼び，機能種別(ILF，EIF，EI，EO，EQの５種)と個々の機能の複雑さ(complexity：低，中，高の３種)とのマトリクスで決定します。例えば，複雑さが「低」のILFならば何点というように点数が決まります。

複雑さは，その機能がいくつのデータ項目を持つか，また，いくつのファイルにアクセスするかなどで決定します。具体的には，データファンクションではレコード種類数(RET：Record　Element　Type)とデータ項目数(DET：Data Element Type)，トランザクションファンクションでは，関連ファイル数(FTR：File Type Referenced)とデータ項目数のそれぞれ２種類の数値を計測し，両者のマトリクスで決定します。マトリクスの値は，CPMに詳細に決められています。

こうして決めた個々の機能の重みを合計すると，アプリケーション全体のFP値が算出できます。FP値算出の仕組みを**図3. 2**に示します。

このように算出したものを未調整ファンクションポイント(Unadjusted

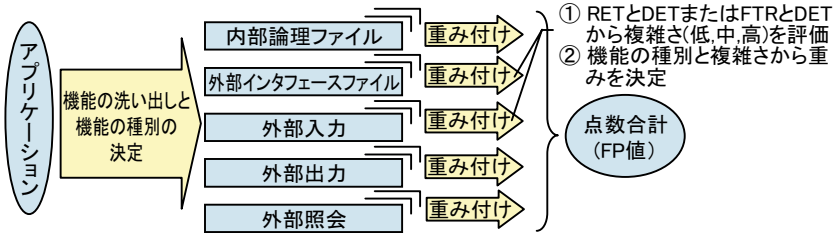


図3. 2　FP値算出の仕組み

Function　Point)と呼びます。IFPUG法では，これに続けて，調整要因の決定，調整済ファンクションポイントの算出を行いますが，この部分は，ISO/IEC 14143で規定された機能規模の範囲ではないので，説明を省略します。IFPUGおよびJFPUG(Japan Function Point User Group)からも「調整要因以下の手順はオプションとする」旨の見解が示されており，わが国でも多くの企業が，未調整ファンクションポイントを最終FPとして使用しています。

(４)　開発初期段階におけるFP計測手法(NESMA法)

IFPUG法では，前述のように，関連ファイル数(FTR)やデータ項目数(DET)を計測対象とするため，開発工程がある程度進み，詳細なシステム設計が確定した段階でないと，正確な計測ができません。そのため，NESMA(Netherlands Software Metrics Association)から，開発初期段階におけるFP計測手法が２種類，提案されています。

(a)　FP概算法(The estimated function point count)　基本的な数え方は，IFPUG法と同じです。開発工程が進まないと正確な数値を入手できない，FTRやDETの計測を避けるため，機能の重み付けにおける複雑さの判定を以下のように簡略化します。

- データファンクションの複雑さを全て低にする
- トランザクションファンクションの複雑さを全て中にする

この計測手法は，複雑さの計測を簡略しているだけであること，かつ，複雑さの分布は計測対象アプリケーションによる偏りが少ないことから，正式のIFPUG法計測値との乖離が小さいことで知られています。

(b)　FP試算法(The Indicative function point count)　この計測手法では，IFPUG法の計測法を用いて，データファンクション(ILF，EIF)の個数のみを計測します。その結果を以下の計算式にあてはめFP値を求めます。

FP試算値＝35×ILFの個数＋15×EIFの個数

35，15という係数は，ILF，EIFのそれぞれに平均的に付加されるトランザクションファンクションの数を仮定し，その仮定に基づく機能セット全体のFP値合計を算出した値です。例えば，ILFでは，3つのEI（ILFへの追加，変更，削除）と2つのEO，1つのEQが付加すると仮定しています(17)。

仮定に基づく計測であるため，仮定が実態に合わない場合は，当然，乖離は大きくなります。NESMA法の国際規格（ISO/IEC 24570）でも，IFPUG法による計測値と大きな差異（50%以上のずれ）が出るケースがあり，使用時には十分な注意が必要です。しかし，機能種別の構成比率を仮定して，ある部分（この場合ではデータファンクション）の計測から全体を試算する発想そのものは，いろいろなケースに応用することができます。たとえば，

FP＝α×（ILFの数）＋β×（EIFの数）

と一般化して，αおよびβのパラメータの妥当性を個別の組織の過去の実績データから導き出すことが考えられます。

（c）　開発の進行に合わせたFP計測法の選択　データ中心アプローチの場合，まずデータ項目を決め，その後に「画面・帳票一覧」を決める手順をとります。このとき，FP試算法とFP概算法を用いて，順次，FPの精度を高めていくことができます。

まず，データ項目が明らかになった時点で，FP試算法を用いて，データファンクションの数に，上記の係数を掛けて規模を見積ります。

次に，「画面・帳票一覧」が明らかになった時点で，FP概算法を用いて，さらに詳細な見積りを行います。画面，帳票に対する要件が決まる段階では，外部入力，外部出力および外部照会の数は決まりますが，画面の詳細なレイアウトやどのファイルを参照するかなどは，さらに設計が進むまで確定できません。そのような場合に，FP概算法により，複雑さの判定を簡略化してしまえば，データファンクション（ILF，EIF）の数とトランザクションファンクション（EI，EO，EQ）の数から，概算のFPを得ることができます（**図3.3**参照）。

なお，IFPUG法に基づく詳細なFPは，さらにデータ項目数，テーブルの構造（レコード構造），処理時の操作データ項目数，各処理において読み書きするテーブル数などが判明した時点で確定することになります。

(17)　第3正規化が終了した場合には，25×ILF＋10×EIFという関係式が示されています。

表3.1には，FP法において示されている試算，概算の方法をまとめています。

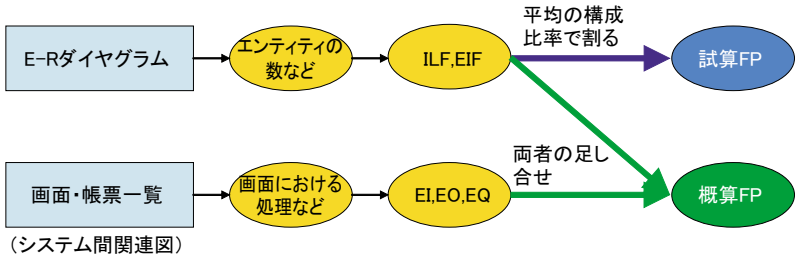


図3.3　部分的な情報からのFP値の推定の方法

表3.1　FPにおける簡易手法から詳細化手法の例

見積り時期	システム化計画～要件定義	要件定義～設計	設計（外部設計）
FP法	FP試算法（NESMA試算法） ・データファンクション（2つの機能）の数を基にFPを推定する。 ・試算FP=35×（ILFの数）＋15×（EIFの数） ・応用として，統計値に基づき，基準とするファンクションと係数を別途設定することもできる。	FP概算法（NESMA概算法） ・5つの機能を基にFPを概算する。 ・各機能の複雑度は評定せずに，データファンクションは複雑度『低』，トランザクションファンクションは複雑度『普通』と設定。	IFPUG法 ・5つの機能おのおのの複雑度評価から算出した重みを加算し，FPを計測する。
FP計測に必要な情報	・データ設計により，内部論理ファイル（マスタやDBなど）と外部インタフェースファイルの数が知られていること。 ・E-R図のエンティティの数などが有効。	・システムで実現する機能（5つの機能）が全て抽出されていること。 ・ただし，複雑度は決定できていなくてもよい。	・システムで実現する機能（5つの機能）が全て抽出されていること。 ・おのおのの機能の複雑度が決定していること。 ・データファンクション：レコード種類数とデータ項目数 ・トランザクションファンクション：入出力時に読み書きするデータファンクションの数と入出力データ項目数

E-R：Entity Relationship

3.1.2　COCOMO法について

（1）　概　要

COCOMO（Constructive COst Model）法は，アメリカのBoehm教授により提案された工数見積りモデルです。1981年に最初のモデルが発表されましたが，1997年には汎用化し適用範囲を広げた第2バージョン（COCOMO II法）が提案されています。ここでは，初期のモデルを中心に，ベースラインと変動要因に関する主要な考え方を紹介します。COCOMO法が前提としている事項など，詳細については，参考文献[16]，[17]をご参照ください。

初期のモデルは63個のソフトウェア開発プロジェクトのデータに基づいて開発され，ソフトウェア開発の実態にあわせて基本COCOMO，中間COCOMO，詳細COCOMOの3つから構成されています。

基本COCOMOは，ソースコード行数（SLOC：Source Lines Of Codes）により見積った開発規模を基にシステムの開発工数を見積る単一変数モデルです。中間COCOMOは，開発規模と合わせて，システムの開発特性を反映した15個の工数変動要因を変数として導入しています。詳細COCOMOは，中間COCOMOと同様に，開発規模と15個の工数変動要因を変数とするうえに，モジュールレベルでの見積りを可能にし，また開発フェーズごとに変動要因の調整ができるようにしています（**表3.2**参照）。

表3.2　COCOMOの3つのモデルの比較

モデル / 比較項目	基本COCOMO	中間COCOMO	詳細COCOMO
開発モード	組織モード 半組込みモード 組込みモード	同左	同左
見積り単位	システム	システム コンポーネント	システム サブシステム モジュール
コスト要因	なし	15個	4個（モジュール） 11個（サブシステム）
開発フェーズ	特に考慮していない	同左	製品設計 詳細設計 コーディング 統合テスト

なお，COCOMO法の考え方は，規模をSLOCに制限するものではなく，実際，COCOMO II法では，規模の単位をSLOCに制限していません。また，COCOMO II法では，変動要因が22個に増加しています。

また，COCOMO法では，プロジェクトの開発形態の違いが生産性の違いとして現れると考え，3つの開発モードに分けて，モデルの係数を設定しています。それぞれのモードの内容は，**表3.3**に示すとおりです。

モードは，開発形態によりモデル式のパラメータの値を層別し設定するもので，見積り精度の向上を目的とするものです。

（2）　モデル式

3つのモデルのそれぞれのモデル式は，**表3.4**に示すとおりです。

表3.3　COCOMO法における開発モード

開発モード	概　　要
組織モード	自社開発のような熟練度の高い開発形態を指し，少人数の開発チームで在庫管理システムや科学技術計算用パッケージなどの業務アプリケーションを開発する場合。
組込みモード	開発チームが大規模であり，種々の厳しい制約条件の下で，ハードウェア，ソフトウェア，運用手順などの複雑さがからみ合っている場合。たとえば，要求仕様への厳しい一致性や開発技法とテスト技法に関する規制が求められ，航空管制システムなどのミッションクリティカルなシステムを開発する場合。
半組込みモード	組織モードと組込みモードの中間に位置するモードである。

表3.4　COCOMO法におけるモデル式

種　類	モデル式	備　考
基本モデル式	$E = a \times \mathrm{Size}^b$ $D = c \times E^d$	$a > 0$，$b > 1$ $c > 0$，$0 < d < 1$ E：開発工数（人月） Size：開発規模（行数） D：開発期間（月）
中間モデル式 詳細モデル式	$E = E_{norm} \times \sum_{i=1}^{15} \alpha_i = a \times \mathrm{Size}^b \times \sum_{i=1}^{15} \alpha_i$ $D = c \times E^d$	$a > 0$，$b > 1$ $c > 0$，$0 < d < 1$ E：開発工数（人月） E_{norm}：名目工数（人月） Size：開発規模（行数） α_i：コスト変動要因iの工数乗数 D：開発期間（月）

（3）　コスト変動要因

COCOMO法で示されているコスト変動要因は，**表3.5**および**表3.6**に示すとおりです。ただし，これらはCOCOMOII法で示されているものです。表3.5に示すものは，初期設計から，設計が終了した開発工程以降でも共通の変動要因です。表3.6と**表3.7**に示すものは，設計が終了した開発工程以降での要因です。

表3.5　COCOMO II法における変動要因（1／3）

規模要因	略号	概　　要	VLow	Low	Nominal	High	VHigh	ExHigh
問題領域経験度	PREC	開発組織が開発対象システムと同様のシステムを開発した経験がどの程度あるか。その分野の知識がどの程度あるか。	全く先例なし	ほとんど先例なし	いくらか先例あり	だいたいなじみあり	おおかたなじみあり	完全になじみあり
プロセス柔軟性	FLEX	ソフトウェアに対する要件，外部とのインタフェース仕様などに対する拘束力。開発側でどの程度変更可能か。	厳格に準拠	場合により緩和	いくらか緩和	一般的な準拠	いくらか準拠	一般的な目標
リスク管理レベル	RESL	リスク管理のためのさまざまな活動をどの程度行っているか。	ほとんど行われていない	いくらか行われている	しばしば行われている	だいたい行われている	ほとんど行われている	完全に行われている
チーム強度	TEAM	利害関係者（ユーザ，開発者，発注者，保守担当者など）間の協力と開発ビジョンの共有の程度。	非常に困難な相互作用	いくらか困難な相互作用	基本的に協力的な相互作用	おおかた協力的な相互作用	高度に協力的な相互作用	シームレスな相互作用
プロセス成熟度	PMAT	SEI（Software Engineering Institute）の成熟度モデル（CMM：Capability Maturity Model）でのレベル。	Level 1	Level 2 下位	Level 2 上位	Level 3	Level 4	Level 5

（備考）　VLow：非常に低い，Low：低い，Nominal：中位，High：高い
VHigh：非常に高い，ExHigh：きわめて高い

表3.6　COCOMO II法における変動要因（2／3）

分類	コスト要因	略号	概　　要	VLow	Low	Nominal	High	VHigh	ExHigh
プロダクト要因	信頼性要求度	RELY	一定期間にわたって意図した機能を実行し続けることに対する要求度合い。失敗が起きたときの影響度合い。	軽微な損失	簡単に復旧可能な小規模の損失	復旧可能な中規模の損失	財政上の大規模な損失	人命にかかわる損失	
	データベースサイズ	DATA	データベースの大きさ。プログラムサイズに対する比率。			D/P＜10	10≦D/P＜100	100≦D/P＜1000	1000≦D／P
	製品の複雑度	CPLX	制御，数値計算，機器依存性，データ管理，ユーザインタフェースなどへの要求の複雑さ。	単純	いくらか複雑	適度に複雑	複雑	かなり複雑	きわめて複雑
	再利用性	RUSE	開発するソフトウェアに求める再利用性のレベル。		再利用を意図しない	プロジェクト間での再利用	プログラム間での再利用	プロダクトライン間での再利用	プロダクトライン間での再利用
	文書化	DOCU	開発過程での文書作成の量。	要求されない	部分的な要求	適当	過度	非常に過度	
プラットフォーム要因	実行性能制約	TIME	システムが使用可能な実行時間のうち，実際に使用すると予想している時間の割合。			T≦50%	70%	85%	95%
	メモリ制約	STOR	ソフトウェアのメモリ使用量のメモリ全体に対する割合。			M≦50%	70%	85%	95%
	プラットフォーム安定性	PVOL	開発するソフトウェアに対するプラットフォーム（ハードウェア，OS，データベースなど）の変化の速さ。		メジャー変更：12ヶ月 マイナ変更：1ヶ月	メジャー変更：6ヶ月 マイナ変更：2週間	メジャー変更：2ヶ月 マイナ変更：1週間	メジャー変更：2週間 マイナ変更：2日	

（備考）　VLow：非常に低い，Low：低い，Nominal：中位，High：高い
VHigh：非常に高い，ExHigh：きわめて高い

表3.7　COCOMO II法における変動要因（3／3）

分類	コスト要因	略号	概　　要	VLow	Low	Nominal	High	VHigh	ExHigh
人的要因	分析者の能力	ACAP	分析者の分析・設計能力，コミュニケーション能力，協調性の度合い。	下位より15%	35%	55%	75%	90%	
	プログラマの能力	PCAP	プログラマの能力，コミュニケーション能力，協調性の度合い。	下位より15%	35%	55%	75%	90%	
	アプリケーションの経験	AEXP	開発対象のソフトウェアシステムと同じアプリケーションの経験の度合い。	2ヶ月以下	6ヶ月	1年	3年	6年	
	プラットフォームの経験	PLEX	プラットフォーム（ユーザインタフェース，データベース，ネットワークなどを含む）に関する経験の度合い。	2ヶ月以下	6ヶ月	1年	3年	6年	
	言語/ツール経験	LTEX	プログラミング言語とソフトウェアツール（要件分析，設計作図，構成管理，ライブラリ管理，一貫性チェックなどを行う）の経験の度合い。	2ヶ月以下	6ヶ月	1年	3年	6年	
	定着率	PCON	プロジェクトの参加者が継続して従事している度合い。	48%以上要員補充	24%	12%	6%	3%	
プロジェクト要因	ツール充実度	TOOL	開発に利用するソフトウェアツールがサポートする機能の充実度と，複数ツールの統合の度合い。	単体のツール	少し統合された単純なCASE	基本的なライフサイクルツール	統合された強力なライフサイクルツール	よく統合された強力で成熟したライフサイクルツール	
	コミュニケーション	SITE	開発拠点の地理的な分散の度合いと，開発拠点間のコミュニケーション手段のレベル。	電話，郵便	電話，FAX	電子メール	広帯域の電子的コミュニケーション	同左およびビデオ会議	双方向マルチメディア
	スケジュール	SCED	標準的な開発期間に対して，どの程度の開発期間を要求されるか。	標準の75%	85%	100%	130%	160%	

（備考）　VLow：非常に低い，Low：低い，Nominal：中位，High：高い
　　VHigh：非常に高い，ExHigh：きわめて高い

開発工程以降での要因は，次の4つに分類されます。

①　プロダクト要因　　②　プラットフォーム要因

③　人的要因　　④　プロジェクト要因

個々の変動要因に対して影響度合い（コスト変動要因の工数乗数：表3.4のα_i）を設定し，モデル式から見積り値を求めます。各変動要因は，影響度を最高で6段階に分けています。変動要因によっては，「低い」から「非常に高い」の4段階に設定するものもあります。そして，各段階に対して影響度合いを乗数（割合い）として設定します。中位（Nominal）は常に1.00とします。

「信頼性要求度」を例にとると，「非常に低い（VLow）」から「非常に高い（VHigh）」まで5段階に分けます。乗数は過去のプロジェクトデータからそれぞれ決定します。例えば，非常に低い：0.75，低い：0.88，中位：1.00，高い：1.15，非常に高い：1.40のように設定します。

では，COCOMO法における具体的な計算方法を簡単に見てみましょう。例として，表3.4の詳細モデル式で，a：3.26（FP/人時），b：1.14，Size：5 000FP，信頼性要件：高い（1.15），製品の複雑度：高い（1.15），分析者の能力：高い（0.86），ツールの使用度合い：高い（0.91）とした場合（それ以外の変動要因は，中位，つまり影響度は1.00とします），次のとおりの結果が得られます。

$$E = E_{norm} \times \sum_{i=1}^{15} a_i = a \times \mathrm{Size}^b \times \sum_{i=1}^{15} a_i$$
$$= 3.26 \times 5\,000^{1.14} \times (1.15 \times 1.15 \times 0.86 \times 0.91)$$
$$= 55\,587.7 \text{（人時）}$$
$$= 370 \text{（人月）}$$

3.2　見積り手法の今後の課題

3.2.1　ライフサイクルコストの見積りについて

（1）　ライフサイクルコスト

基幹系のシステムなど，長期にわたり稼働するシステムにおいては，最初の新規開発でのコストとともに，実際に運用段階に入った後での修正，更新，機能追加などのコストを含めたライフサイクルコストがIT投資の観点から重要です。現在，ソフトウェア開発コストの半分以上，あるいは8割程度は保守であるといわれています。これは，ほとんどの企業において，既存システムが存在しており，それとの関係がまったくない新規のシステム開発が想定しにくい状況が背景にあります。

図3. 4に２つのパターンを示しています。図の左側は，初期コストを抑えて，運用コストが高くなったパターン，図の右側は，初期コストは大きいものの，運用コストを抑えたパターンです。長い運用・保守期間があるシステムの場合，初期コストをかけても運用コストが下がれば，トータル，すなわち，ライフサイクルコストで低くなるほうが有利です。

では，後者のパターンにするには，どうしたらよいのでしょうか。

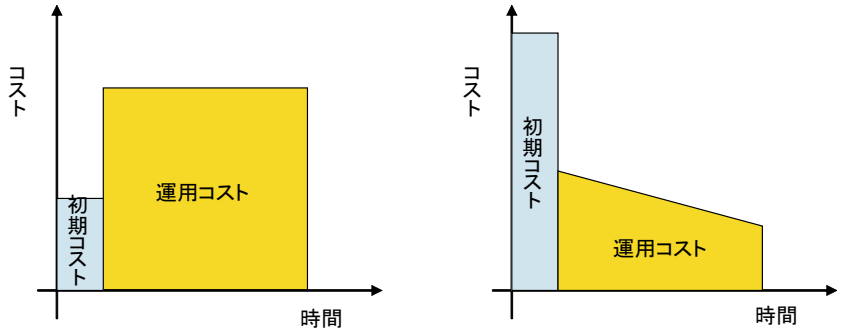


図3. 4　ライフサイクルコストのパターン

（２）　運用・保守のコスト見積り

図3. 5に保守における要求定義の様子を示します。図の左側に示した「要求」を新規のシステム開発における要求とします。新規のプロジェクトとして，要求から要件へ，要件に基づいて設計，製造，さらにテストなどの検証を経て，リリースされます。リリース後にシステムの運用が開始されると，ユーザの利用に基づいて新たな要求が生じたり，ビジネス環境の変化や社内の組織の変化などにより既存の機能を変更する必要が生じたりします。新たな要求をシステムに実現するところが保守に相当しますが，新規の場合との大きな違いは，既存のシステムが存在することです。

図3. 6には，保守における費用構造を示しています。保守費用は大きく３つの要素から構成されます。

①　現行理解
②　機能実現
③　テスト

保守の場合に特徴的なのは，新たな要求を機能として実現する前に，既存のシステム（母体）の理解が求められることです。ここでは，母体の規模，母体の

理解容易性，母体の保守性の高さ，既存部分の習熟度，変更箇所の分散度が費用増減の大きな要因となります。

テストも保守の場合，既存システムとの関連部分について留意する必要があ

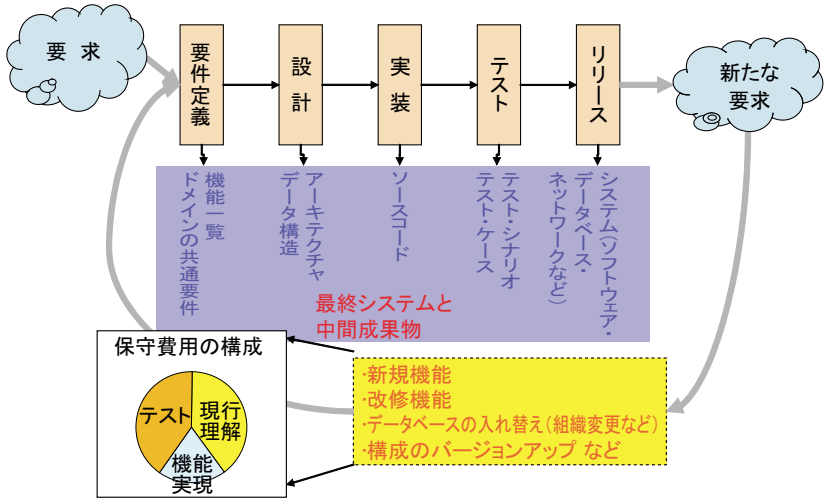


図3. 5　保守における要求定義

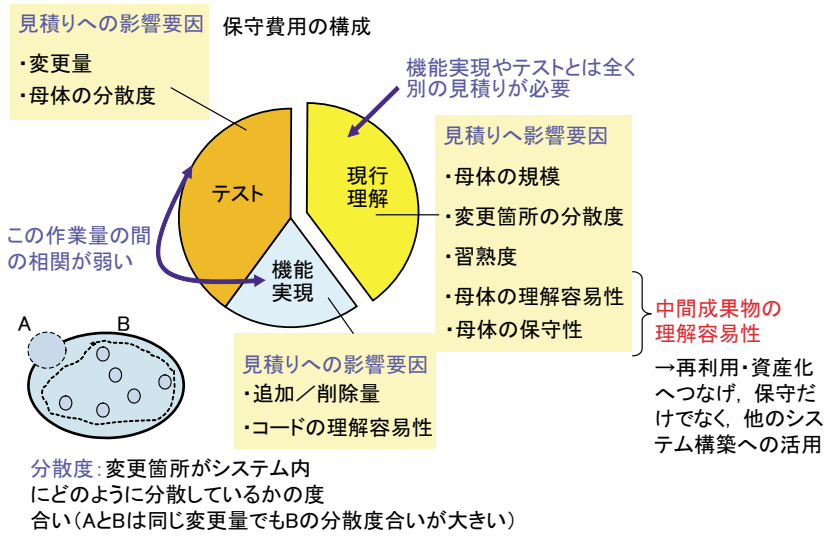


図3. 6　保守における費用構造

ります。機能の変更量や追加量が一番大きな要素となります。また，新たに追加した機能が既存のシステムの各所の機能と関係する(分散度が高い)場合は，テスト工数が増加する影響もあります。新規の場合と大きな違いは，機能量とテスト量の相関が低い点です。

また，機能実現の部分は，新規の場合と似たアプローチがとれますが，既存の機能を変更する場合には，規模の数え方などに違いがあります。

以上のとおり，保守コストを下げるためには，現行システムの理解容易性を高める工夫や機能の追加や変更時におけるテストを容易にする工夫が重要です。図3.4の右のパターンにするためには，そのような工夫を初期に行えるように投資する(コストをかける)といったことが重要になります。

保守コスト，ひいては，ライフサイクルコストに関する見積りにおいて，上記のような考え方が考慮される必要がありますが，本ガイドブックでは，指摘するにとどめ，今後具体的な見積り方法について検討したのち，改めて公表していきたいと考えています。

3.2.2　アジャイル的開発の見積りについて

最近のビジネス環境では，仕様をあらかじめ固定することが困難である場合が少なくないことを受け入れて，要求変化にじん速に対応可能な開発方法論などが注目を集めています。

それらは，アジャイル，インクリメンタル，エボリューショナル，スパイラルなどといった考え方で提唱されていますが，規模からコストなどの見積りの方法はそのままでは利用することができません。開発パラダイムが多様化することは，それぞれの立場で着目する視点，アクティビティ，手順，基準も異なることであり，見積り手法の入力，コスト要因，出力項目，計算の方法も工夫しなくてはなりません。

アジャイル型を含め最近の開発パラダイムで多く採用されているインクリメンタル(段階的拡充)手法では，プログラムを常に実行可能な状態に保ち，確認・拡充していく方法を採っています。各段階に要する期間は，数日から数週間と短いのが通例です。こうすることによって，仕様変動のリスク，技術的実現性のリスクを回避することができます。段階を重ねるごとに生産性は向上し，製品やサービスのリリースという位置づけでのプロジェクトの完了も，市場や経営上の視点から決定されることが多くなります。こういった場合には，プロジェクト期間をあらかじめ見積り手法の入力として設定することも，むずかしくなってきます。

本ガイドブックでも，要求がまだ確定していない場面での規模の見積り方法をはじめ，上記のような状況への対応方法を示していますが，基本的には，仕様が確定するまでの簡便な方法として位置づけています。

要求仕様が変化をすることを前提に，見積りにその変化を組み込んでいく方法は，まだ更なる検討を要するものであり，今後具体的な見積り方法について検討したのち，改めて公表していきたいと考えています。

3.2.3　「見える化」と見積り活動の関係

「第2章 見積り能力の向上」での主張は，「見積り手法だけで見積り精度の向上は実現できるものではなく，プロジェクト途中段階でモニタリングとコントロールを行い，プロジェクト終了後は，予定と実績との差異分析を通じた差異の原因の追究と改善を繰り返すことにより，実現する」ものです。

見積り活動は，プロジェクトの最初で見積り値を出して終了というものではなく，プロジェクトの終了までのアクティビティすべてと一体となったものです。見積り活動は，プロジェクトの途中段階でのマネジメント，特にプロジェクトの状況を把握し，予定とのずれを監視する観点から「見える化」の一部として組み込まれる必要があることがわかります。

本ガイドブックは，全体工数や全体工期の見積りを中心にして，エンジニアリング的なアプローチを解説したものです。プロジェクトの途中段階での予定の設定(マイルストーンごとの工期，工数の予測)などには踏み込んでいません(もちろん，本ガイドブックで示した考え方は，それらへ活用可能です)。

「見える化」の活動のインプットとして，プロジェクトのマイルストーンや各マイルストーンでの成果物(ドキュメント，テストケース，品質の状況など)をいかに見積り，設定し，モニタリングとコントロールを行うかは，プロジェクトにとっての大きな課題です。

SECでは，本ガイドブックで示した「見積り手法」とは別に，「定量データ分析」「見える化」というテーマで研究開発を実施しております。これらは，「定量的アプローチ」として統一するものですが，今後，本ガイドブックで示した内容とともに，定量的アプローチとして，広く実践を研究開発し，IT産業に対する問いかけと，実践のための材料を提供し続ける所存です。

第２部
見積り手法の事例集

第１章　事例集の活用方法

ここでは，ソフトウェア開発の見積りについて，先導的な取り組みを行っている，各社の事例を紹介します。

各事例は，原則として，以下のような構成となっています。

（１）　**取り組みの背景**：見積り活動に関する各社のこれまでの取り組み，当該見積り方法を利用するに至った経緯などについて記述しています。

（２）　**見積り方法**：当該見積り方法におけるモデルの説明(入力パラメータ，出力，基本的なアルゴリズム・方式など)を記述しています。

（３）　**見積り方法の前提条件**：当該見積り方法を利用するために必要となる諸条件(見積り時期，見積り対象，見積り活動，併用見積り手法，体制・役割分担・企業文化など)について記述しています。

（４）　**精度向上のための活動**：当該見積り手法の継続的な改善活動(見積り値と実績値との差異分析，フィードバックなど)について記述しています。

（５）　**実施実績**：当該見積り方法を実際に適用した分野(業種，システム・プロジェクトの特徴)，適用した見積り時期とその精度などについて記述しています。

（６）　**当該見積り方法の優位点と課題**：当該見積り方法のアピールポイント，今後の課題，利用に当たって留意すべき点などについて記述しています。

各事例を順番に読んでいくのもよいのですが，以下に該当する方は，各社上記項目を比較しながら，読み進めるのも効果的です。

EML１の企業の方：各社事例のうち，特に(１)～(３)および(６)を参考にしながら，自分に合った見積り手法を見つける。

EML２の企業の方：各社事例のうち，特に(２)～(３)および(６)を参考にしながら，組織的な見積り手法確立へのステップアップのヒントを得る。

EML３～EML４の企業の方：各社事例のうち，特に(３)～(４)を参考にしながら，組織的な見積り手法の具体的な活用・見積り能力の定量的な

制御へのヒントを得る。

なお，ここで紹介している事例は，次のとおりです。**表1.1～表1.4**には，各事例の概要(見積り時期，見積り対象，入力データなど)を示しています[(1)]。

これらの事例はあくまで各社が利用している見積り方法の１つという位置づけであり，それぞれ社内で当該手法のみを用いていることを示すものではないことに注意してください。

★ FP法をベースとした事例

ー野村総合研究所手法

NESMA法の実践

ーTIS手法

生産性変動要因の分析と絞込みにより精度向上と理解容易性を高める。

ー日立システムアンドサービス手法

機能要素見積り法を中核としたFP法の適用実践

ー日立製作所手法

見積り時期に応じた機能量からの見積り

ーファンクションスケール法(富士通)

見積り時期に応じた画面に基づくファンクションスケール(Function Scale)に基づく見積り。ファンクションスケールは，富士通で提唱する規模尺度。

★ COCOMO法をベースとした事例

ー日本ユニシス手法

COCOMO法を活用した見積りの妥当性の検証，リスクの事前把握

★ 独自モデルによる事例

ー日本IBM手法

過去プロジェクトデータをベースラインにしたパラメトリックモデルの構築と活用

ージャステック手法

独自の生産管理に基づく見積りモデル

(1)　見積り時期に応じて複数の方法が示されている企業については，表1.1～表1.4において黄色い網掛けが施してあるものを中心に事例を紹介しています。

特にEML１やEML２の企業の方が，各社見積り手法を参考にしようとする場合は，次にあげる３つのポイントに着目するとよいでしょう。

★ポイント１：見積りのベースとなる情報(材料)として，どういったものをそろえることができるか。
★ポイント２：第１部 第１章 図1.7における見積り①から見積り④のうち，どのタイミングで適用したいのか。
★ポイント３：過去の見積りデータの蓄積があるか。

まず，ポイント１を考えることによって，おおまかにどういったタイプの手法を適用できそうかが分かります。例えば，データ項目についての情報であればFP法，画面・帳票についての情報であればファンクションスケール法などといった具合です。

次にポイント２では，ポイント１で考えた情報を確定値として用意する必要があるのか，あるいは，過去の類似プロジェクトなどをもとにした推定値を使用することになるのかが明らかになります。

ポイント３では，第１部 第１章 1.3.4項で述べたようなベースライン・変動要因の設定や，ポイント２でいう推定値などとして使用するには，どうすればよいかが分かります。過去データの蓄積があれば，それらの分析結果をベースとすることができますし，もしそういったデータがない場合でも，最初はSECデータベースをはじめとした外部データを使用して，それからデータの測定・蓄積を進めていくことも可能です。

この事例集が，読者の皆さんが自分にあった見積り手法を見つけたり，あるいは現在使用している見積り手法のカスタマイズやブラッシュアップを行ったりする際の一助となれば幸いです。

表1.1　各社見積り

手法名	対象分野	見積り時期	見積り対象	入力データ	見積り式（パラメータ）	各手法の適用実績
野村総合研究所手法	汎用	システム化計画→見積り②	工数・期間	想定画面・帳票数		・NESMA法は3年程度。FP自体は4～5年。4年前，品質監理部門により，現場のFP計測を支援するキャンペーンを実施。 ・一定規模以上のプロジェクトは社内の公式のレビューにおいてFP適用も含めて見積りをチェック。
		概要設計，基本設計→　見積り③，見積り④	工数	FP(注１)，生産性(注２)	工数＝FP×生産性	
			期間	期間/工数相関データ(注２)		
TIS手法	C／S系(Web系含む)	企画提案段階	工数	RFP，顧客打合せ資料，類似プロジェクト		・1997年にFP法導入開始。1998年FP見積り基準値を設定し，1999年よりFPによる見積り検証を義務化。 ・ファイル・入出力で見積り可能な新規開発(機能追加を含む)のプロジェクトであり，かつ規模が10人月を超えるものが適用対象。
		要件定義後→　見積り③，見積り④	規模[FP(IFPUG法)]，工数	標準WBS，FP，規模変動係数(注１)，生産性基準値(注２)，生産性変動係数(注３)	積上げ法による見積り結果を，FP法(IFPUG法)による見積り結果と比較することで検証(注４) ・開発工数＝システム規模(FP)／個別プロジェクトの生産性 ・システム規模(FP)＝計測した機能規模×規模変動係数 ・個別プロジェクトの生産性＝生産性基準値(FP/人月)×生産性変動係数	

手法概要（１／４）

経営層や現場の理解の得やすさ	適用コスト：習得時	適用コスト：実際の見積り算出時(規模計測時)の所要時間	適用を可能にする前提となる，組織体制(社内の機能)	適用を可能にする前提となる，蓄積したデータ・情報の種類	備　考
・FPは普及から，2～3年程度であり，まだ完全には行き渡っていない。 ・FPの計測法や概念は行き渡りつつある。プロジェクト完了時に見積り・実績ともに報告されるようになっている。 ・初めて計測する人には支援も行っている。 ・部門ごとにFPのキーマンが出てきている。				・全社的な実測値DBを整備。生産性，期間/工数相関データなどについては，類似プロジェクト見積り時の入力データとして使用。	(注１)概要設計では推定値を使用 (注２)全社的な実測値DBから得られる統計値
・適用する局面を「新しい技術分野の新規開発」に限定し，その局面に対応した生産性変動要因５つに絞り込むことで，生産性の設定や算出式が基礎知識をもたない開発要員にも理解しやすい平易なものとなっている。				・生産性基準値は，半年ごとに見直しを実施。	(注１)要件の確定度を評価して設定 (注２)開発規模に応じて設定 (注３)影響度が高いと思われる要因の上位５種を抽出・分類し，ガイドを作成 (注４)FP法見積りによる検証は，要件定義後，外部設計後，プロジェクト完了後の計３回実施

表1.2　各社見積り

手法名	対象分野	見積り時期	見積り対象	入力データ	見積り式（パラメータ）	各手法の適用実績
日立システムアンドサービス手法	汎用	システム化計画段階 ・機能が把握できない段階は「FP試算法」を適用→見積り①，見積り②	FP，工数	画面数(又は概算)，帳票数(又は概算)，ファイル数(又は概算)，類似システム規模，予算，基準生産性，プロジェクト特性	・FPについては本文参照。工数は以下のとおり。 ・プロジェクト全体工数＝ソフトウェア開発工数＋その他工数 ・ソフトウェア開発工数＝開発規模(FP)／プロジェクト生産性 ・プロジェクト生産性＝開発パターン別基準生産性×該当プロジェクトの特性	・1996年６月　にJFPUGに加入し，IFPUG法を導入。その後，FP計測技術者育成，生産性データの蓄積と分析等を推進。 ・1998年には，当社の生産効率管理にFP法を採用。また，FP法適用経験を踏まえ，当社独自の「機能要素見積法」及び，機能洗出し定義支援ツール「機能整理FP算出ツール」を開発・適用。 ・現在は一定規模以上新規開発プロジェクトは原則FP法適用必須。
		要件定義段階・機能一覧レベルが把握できる段階は「FP概算法(機能要素見積法)」を適用 →見積り②，見積り③	FP，工数	テーブル一覧，機能一覧，機能要素，基準生産性，プロジェクト特性		
		設計段階・関連ファイル数や入出力項目数がほぼ把握できる段階はFP詳細法(IFPUG法)を適用 →見積り③，見積り④	FP，工数	テーブル一覧とデータ項目数，機能一覧と関連ファイル・テーブル数，データ項目数，基準生産性，プロジェクト特性		
日立製作所手法	アプリケーションソフト	予算化前(仕様がほとんど見えていない段階)(超概算見積り) →見積り①，見積り②	FP SLOC(注１)	画面数，帳票数	基準値法 ・各見積り対象データの算出は，それぞれの入力データと見積り対象データの換算テーブルを用いて行う。 ・FPからSLOCの算出には，開発言語，アプリケーションアーキテクチャなどによる換算係数を使用。 ・SLOCから工数の算出時には，要求される信頼性や開発ツールなどによる換算係数を使用。 ・“基準値”は実績データをもとに組織的に見直す	・1995年より試行開始。大規模なプロジェクトや顧客がFPでの提示を求められるケースではFPを使用。 ・情報・通信グループで全面的に採用したのは2003年から。制度化したのは2004年から。大規模新規プロジェクト中心に適用。 ・顧客には，FPを求められればFPを提示。 ・見積り時のFP計測件数は100以上。実績値のFP計測はサンプリングしたもののみのため，数は少ない
			工数 工期	SLOC(注１)		
		引合/受注段階(仕様があいまいな段階) 開発開始前(概算見積り) →見積り②，見積り③	FP SLOC(注１)	マスタ/機能の個数(DBテーブル数，画面数，要素機能の個数)の実測または推測数		
			工数 工期	SLOC(注１)		
		要件定義完了時等，仕様が明確な段階(基本設計後)(確定見積り) →見積り③，見積り④	FP SLOC(注１)	マスタや機能の内容（複雑さを表す情報，つまりIFPUG法で規模を実測または予測）		
			工数 工期	SLOC(注１)		

手法概要(２／４)

経営層や現場の理解の得やすさ	適用コスト：習得時	適用コスト：実際の見積り算出時(規模計測時)の所要時間	適用を可能にする前提となる，組織体制(社内の機能)	適用を可能にする前提となる，蓄積したデータ・情報の種類	備　考
・幹部指示により，一定規模以上の新規開発プロジェクトへは原則FP法適用必須として推進中。	・1.5日～２日のFP法定期教育実施。1000名以上教育済。 ・FP概算法については，1時間程度の教育で適用可能。	・ツール適用により適用時間の短縮を図っている。	・適用推進専門部門がある。	・FP法適用開始時（1996年）からFPデータ・工数データ等のプロジェクトデータを蓄積している。	見積時の情報の詳細さに応じて適用手法を選択し，規模・工数等を見積る粒度を揃えた機能要素を選択指定するだけで機能の定義ができる「機能要素見積法」を中核として規模見積を推進している
・Webシステムなどの場合，SLOC(ステップ数)だと価値を表せないということを顧客にも経営層にも理解が得られるようになってきた。 ・全社的な展開がしやすくなった。 ・見積り根拠が明確となり，理解を得やすくなった。	・マニュアルやガイドラインが提供され，必要なときに学習できる。 ・管理者層，担当者層それぞれに向けた教育(座学，eラーニング)を常時開設している。	・FPの標準的な計測時間を計測ガイドに記載している。 ・概算見積りと確定見積りで計測時間が3～5倍違う。つまり，IFPUGで見積る場合の1/3や1/5で概算見積りできるようにしている。	・見積りはSE部門で作成し，PMOメンバがチェックする。 ・見積り作業時に，PMOメンバが各種見積り技法の活用法や細かいノウハウについてSE部門をサポートすることもできる。 ・見積りレビュー会議がある。リスクが高い案件はレビュー前に専任グループが入って内容を確認する。 ・高リスクの判断基準＝金額，未経験分野，初めての顧客，要件の厳しさなど。　PMBOKベースのチェック項目でリスクを点数付けしている。	・類似する実績情報(規模，工数，プロジェクト特性，プロジェクト運営上の工夫点など)や見積り情報(規模，工数，プロジェクト特性，プロジェクトリスクなど)を活用する。 ・「類似」の判断に際しては，同じ業種，同じシステム(業務，アーキテクチャ)程度の粒度で判断する。	(注１)並行して，ソフトウェア開発とは直接規模的関連を持たないがシステム構築上必要な作業についても洗い出し，工数算出時には考慮に入れる

表1.3　各社見積り

手法名	対象分野	見積り時期	見積り対象	入力データ	見積り式（パラメータ）	各手法の適用実績
ファンクションスケール法（富士通）	Webシステム	要求定義段階（概算FS見積り） →見積り③	概算FS	機能の実装難易度ごとのFS基準値，業務処理数（注１）	概算FS=Σ（当該ランクの概算FS基準値×当該ランクの業務処理数） ランク：機能の実装難易度（A～Dの４段階）	・2005年度から適用を開始。 ・確認できているのは20～30件程度。見積りガイドラインそのものは1 000件以上ダウンロードされているので，実際はもう少し多いかもしれない。
		基本設計後（FS見積り） →見積り④	標準FS	各画面のFS	標準FS=Σ（各画面のFS）	
			見積りFS	標準FS，変動要素による調整係数（注２）	見積りFS＝標準FS×α α：変動要素による調整係数	
日本ユニシス手法	汎用	要件定義前，論理設計後 →見積り②～見積り④	工数（トップダウン方式）	SLOC（注１），FP（注１）	規模をFP/SLOC（注２）で出し，生産性基準を考慮して最終的な工数を算出する	
			工数（ボトムアップ方式）	標準WBSまたは類似システムのWBS		

手法概要（３／４）

経営層や現場の理解の得やすさ	適用コスト：習得時	適用コスト：実際の見積り算出時（規模計測時）の所要時間	適用を可能にする前提となる，組織体制（社内の機能）	適用を可能にする前提となる，蓄積したデータ・情報の種類	備　考
・経営層ならびに現場の反応はおおむね良好。 ・社内見積りのガイドラインとしてイントラネットからダウンロード可能な状態にしているが，使用するかどうかは各事業部の判断に任せている。	・最初のレクチャーとして30～40分程度（スプレッドシートなど）。	・１画面５分で見積れることを目標にしている。		・2005年度から適用を開始したばかりなので，実績情報の蓄積はこれからである。	（注１）クライアント画面からの要求によりDBに対する登録・更新・削除・検索を行い，その結果を返す一連の流れを１業務処理とカウントする。 （注２）開発業務の難易度，品質要求のレベル（性能要件，操作性要件など），エラーチェックのバリエーション，ビジネスロジックの複雑度などが考慮される。
・見積りを行う担当者がそれぞれ一番自信を持っている方法を利用しようというスタンス。 ・WBSについては，会社としての標準WBSを用意している。各部署がそれに基づきカスタマイズしている場合がある。 ・実際にPMが前回・前々回の開発（同じ業務分野のシステム開発）で使用したWBSを再利用することが多い。			・PMOを1998年に組織化。	・過去のプロジェクトデータをDB化。要件定義前において，類似プロジェクトから類推した見積りを算出する。	複数方式（トップダウン方式（FP・COCOMOII）と従来のボトムアップ方式）での見積り実施による妥当性向上。 （注１）要件定義前の場合は，推測値を使用。 （注２）規模をSLOCで出した上でCOCOMOIIを適用する方法も現在評価中。

表1.4　各社見積り

手法概要（４／４）

手法名	対象分野	見積り時期	見積り対象	入力データ	見積り式（パラメータ）	各手法の適用実績	経営層や現場の理解の得やすさ	適用コスト：習得時	適用コスト：実際の見積り算出時（規模計測時）の所要時間	適用を可能にする前提となる，組織体制（社内の機能）	適用を可能にする前提となる，蓄積したデータ・情報の種類	備　考
日本ＩＢＭ手法	汎用	要件定義（ドメインモデルでデータ項目が見えた段階），基本設計時 →見積り③，見積り④	工数	開発サイズ	・工数＝a×｛10＾（Log（開発サイズ））＾b｝＾c ・a,b,c：定数	・過去12年間のSI案件（１万件超）に適用。 ・見積りツール（注１）を開発し，社内イントラネット上のサーバで稼働するWebアプリケーションとして提供中。	・PMは複数手法で見積るように指示されており，本手法はそのうちの１つという位置づけ。	・PM研修の一環として一般的な見積り技法（FP/COCOMO他）を基礎知識として学習。その上で社内ツールの利用方法を習得。	・入力対象情報の調査収集作業はプロジェクト規模に依存。これを除けば，入力・試行錯誤の時間を含めても極めて短時間で結果を算出可能（見積り計算自体はツールで処理するため）。	・プロジェクト管理（含む見積もり）のレビューを行う品質保証組織。特に積算・見積り漏れの検証や新技術への対応方法ガイドなども当該組織で行う（注２）。	・開発負荷は，サイズを見積る尺度であれば何でも利用可能（DSLOCである必要はなし）。	（注１）開発サイズから見積り負荷を得る機能のほかに，投入可能な負荷から制限すべきサイズを得る機能も持っている。 （注２）実際には品質保証（QA）組織がレビューを行う際に他社でいうPMO・SEPG的役割を兼任しており，各種レビューを通じて見積り結果の検証も行っている。
ジャステック手法	汎用	要件定義後（以降各工程で適用可） →見積り④	規模（各工程の生産物量），生産性（各工程の生産物の単位量当たりの工数またはコスト），工数（各工程の生産物別）	次のいずれかを選択。 ①基本設計書の規模（文字数） ②類似システムまたは再構築対象のソースコード行数 ③要求定義書に記述されているシステムへの入力，出力および機能	・工数（コスト）＝生産物量×生産性 ・生産物量＝標準生産物量×（１＋Σ生産物量環境変数（注１）） ・生産性＝標準生産性値×（１＋Σ生産性環境変数（注２））	・20数年前より適用。その都度改良を加えている。 ・生産物量の単位は当初，文字数ではなくページ数を単位としていた。品質特性や環境特性の導入も途中からユーザの声を反映し，改善してきた。	・会社設立当初より，生産性原理に基づき「量に基づく客観的な価格設定をしたい」「量に基づく客観的な技術者の能力評価をしたい」ということでトップダウンで推進。 ・手法は汎用的に適用。全てのプロジェクトが同じ形式で作成され，横並びで比較可能であることが大前提。	・新入社員研修の一環としてPSPを実施。２週間程度の期間で自分たちの計画を作成し，見積り結果がどれくらいの精度を持っているかという感覚をつかんでもらう。	・新規顧客の案件は時間がかかる。初回の開発を通して確実に実績が蓄積され，差異の原因も明らかになり，顧客との認識の統一も図られる。したがって，２回目以降の開発では，見積りの所要時間は飛躍的に低減する。	・生産管理システムの全体像を「標準開発計画書」として，開発部門が作成。これを受けて，営業部門が「標準見積り書」を作成。開発部門はさらに内容を深堀りして「PRJ実行計画書」を作成。 ・標準開発計画書が作成されると，SEPG部署（開発部門のプロセス改善推進部署）のベテランが必ず目を通す。	・適用開始時よりDBに蓄積。ただし，環境が変化するスピードは速いので，あまり古いデータは参考にならない。２～３年分あれば十分。	（注１）生産物量環境変数は，あらかじめ用意された要因リスト（品質特性変動要素評価表）から個別のプロジェクトごとに評価。生産物量の単位は，ドキュメントであれば文字数（KC），コードはKLOC。 （注２）生産性環境変数は，あらかじめ用意された要因リスト（品質特性変動要素評価表と環境特性変動要素評価表）から個別のプロジェクトごとに評価。

第２章　野村総合研究所手法

2.1　取り組みの背景

情報システム開発プロジェクトでは，見積りに基づき要員計画が立案される。不正確な見積りは，要員不足を引き起こし，品質の低下や納期の遅れにつながりがちである。的確な見積りは，プロジェクトマネジメントの第一歩であり，プロジェクト計画にとっての核となる基本数値である。

野村総合研究所（以下「NRI」と呼ぶ）では，以前のメインフレームの中心の時代には，主にCOBOLのプログラムステップ数を用いた規模見積りを行っていた。しかし，オープン系システムが主流となり，開発言語がC/C++，VisualBasic，その他のGUI（Graphical User Interface）指向の開発言語に変わっていく中で，新たな見積り手法の必要性が認識されるようになった。プログラム言語に依存しない見積り手法としてファンクションポイント法（FP法）には早くから注目していたが，情報システムの変化に対応して本格的に取り組むこととした。

FP計算で必要となるデータファンクションや，トランザクションファンクションは，設計の初期段階では，精密な計算ができないことは，FPについての調査段階から認識していた。FP適用のモチベーションを高めるためには，設計の初期段階から適用可能とすることが必要であり，少ない情報からFPを推定する手法を検討した。

2.2　見積り方法

（１）　簡易FP法

IFPUG法に基づきFPを求めるには，データ項目の数を知る必要がある。基本設計が終了すれば，データベースやトランザクションのデータ項目が確定するが，その前段階である概要設計では，データ項目の定義は完全ではない（工程の定義は後述）。したがって，FPの推定が必要になる。

データ項目が確定していない段階での簡易FP法として，NESMA（Netherlands Software Metrics Association）で開発された，FP試算法（The indicative function point count），FP概算法（The estimated function point count）がある。これらは，少ない情報でIFPUG法を簡単に行うための方法である。

URL：http：//www.nesma.nl/japanese/index.htm

①　FP試算法（オランダ方式と呼ばれている）

－ILF（内部論理ファイル），EIF（外部インタフェースファイル）を決定する。

－FP試算値＝35×ILFの数＋15×EIFの数

－１つのILFには平均３つのEI（外部入力）（内容は追加，変更，削除）と２つのEO（外部出力），1つのEQ（照会）があると仮定。また，1つのEIFには１つのEOと，1つのEQがあると仮定している。

②　FP概算法

－ILF，EIF，EI，EO，EQを識別する。

－ILF，EIFは複雑度を低，EI，EO，EQは平均とみなし，IFPUG法に基づきFPを算出する。

（２）　工数，期間の見積り

FPを計測した後の見積りの流れを**図2.1**に示す。

図2.1　工数，期間の見積り

標準工数は，

標準工数＝機能規模/生産性

で計算される。生産性の単位はFP/人月。生産性については，一般的に適用可能な標準生産性が設定されている。その設定方法については後述する。

プロジェクト工数は，標準工数に対して，そのプロジェクトの特徴を考慮して調整を行った工数である。このときCOCOMO IIのコストドライバの項目を参考に，それぞれの影響度合いを検討し，工数調整を行う。項目としては技術者の経験，再利用の度合い，期間などがある。そのほかに，顧客の業種，適用業務も考慮する。また，プロジェクト工数に対して，工程別工数比率を適用することで，工程別の工数を設定する。工程別工数比率は，**表2.1**の形式で提示している。

表2.1　工程別工数比率（イメージ）

	概要設計	基本設計	詳細設計～単体テスト	連結テスト	総合テスト
工数比率の目安（%）	10	15	45	15	15

ここで，詳細設計～単体テスト，連結テストは，ソフトウェア設計，プログラミング，ソフトウェアテストに相当。総合テストは，システムテストに相当する。

期間と工数の関係について，過去の実績に基づく分析を行っている。この期間と工数の関係を利用して，プロジェクト工数から期間を見積もる。

顧客要件として完成時期が設定されることもある。その場合，期間を前述したコストドライバとして扱い，必要に応じて工数に反映する。

2.3　見積り方法の前提条件

（1）　見積り時期，見積り対象

見積りはシステム化計画，概要設計，基本設計の各工程で行う。見積り対象は，規模，工数，期間。システム化計画では，類似法で見積りを行う。概要設計段階では，積み上げ法，類似法に加えて，FPによる規模見積りを行う（**図2.2**参照）。

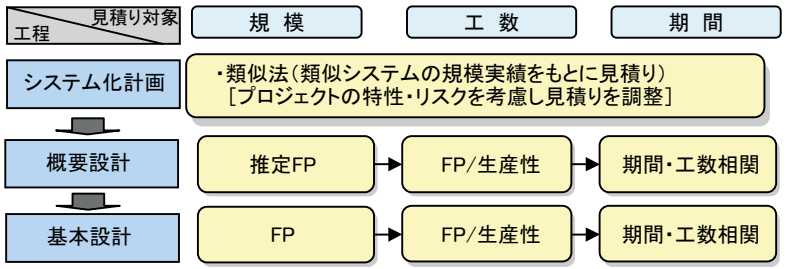


図2.2　見積りの概略

ここで，各工程の概要は次のとおり。

- システム化計画：システムの企画とプロジェクト計画の立案を行い，プロジェクト開始の承認を得る。

- 概要設計：要件定義工程に相当。新システムの要求仕様をまとめる。実現すべきシステムの構造，主要な入出力，データベースなどを決める。
- 基本設計：システム設計工程に相当。要求仕様に基づきプログラム分割までを行う。画面・帳票・データベースなどの項目を確定する。

なお，これらの工程は，NRIの開発標準である「NRI標準フレームワーク」で定義されている（概要は参考資料に掲載）。

見積りは，各開発プロジェクトにて実施する。品質監理部門が見積りの支援を行うことがある。

（2）　概要設計における見積り

概要設計では，システム全体にわたり構成の概要を設計する。ここでデータベース一覧，テーブル一覧，E-Rダイヤグラム（Entity-Relationship diagram）などを作成するので，ILF，EIFをカウントできる。また，機能一覧も作成するので，EI，EO，EQをカウントできる。

個々のデータ項目は未確定であっても，これらの情報をもとにFP概算法を使用して，FPを推定することが可能である。

FPの推定結果を検証する目的で，NRIでのFP実績値をもとに，機能別構成比率を計算している（**図2.3**参照）。

この例では，流通業務システムを対象として，比率を計算している。

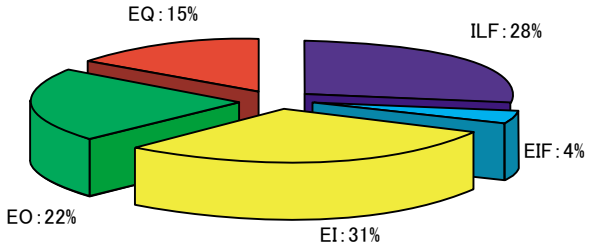


図2.3　機能別構成比率（流通系）

FP概算法で見積ったファンクションの数が，この比率からみて妥当かどうかの評価を行う。

なお，業種別の比率を計測しているのは，業種によりデータ項目の多寡や，入出力の特徴の違いが考えられるためであるが，現時点の実績では，大きな違いはない。

（３）　基本設計における見積り

基本設計では，データベースの項目やトランザクションの項目が確定するので，正式のFP法（IFPUG法）を使用する。このためにFP算出シートを用意しており，必要なデータを入力することで，FPが自動計算される。

機能別構成比率は，基本設計においてもFP計測結果の妥当性確認に使用する。特定のファンクションの値が機能別構成比率と大きく異なる場合，設計における機能の抽出が不十分な可能性がある。

（補足）　前述のような見積り手順の適用は，2.4節で説明する精度向上のための活動を定常的に実施していることが前提となっている。

2.4　精度向上のための活動

（１）　データの収集と分析

前述のFP推定方法は，実プロジェクトの実績値がベースになっている。生産性に関しても実績に基づき設定する必要がある。このような，実績値を収集・分析する活動を行っている（**図2.4**参照）。

プロジェクト終了時にはプロジェクト完了報告書の作成が義務付けられてい

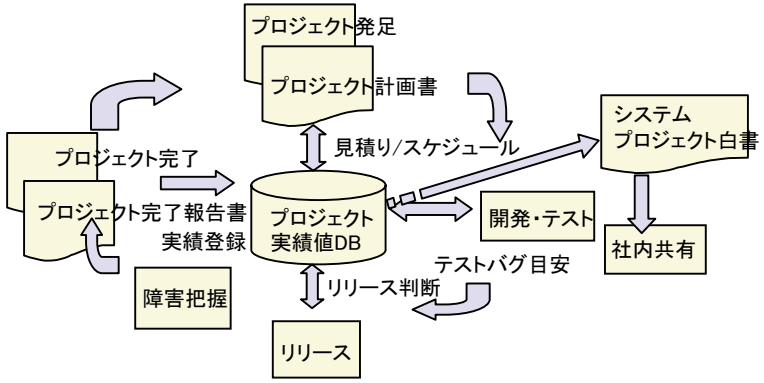


図2.4　プロジェクト実績値の収集

る。プロジェクト完了報告書には，開発規模，工数，スケジュールなどの各種実績値を記入する。見積りと実績との差異分析も記入する。その他，障害の数などの品質に関する情報や，教訓・ノウハウなどの定性的な情報も含まれる。

プロジェクト完了報告書から抽出した実績値は，プロジェクト実績値データ

ベースに登録される。プロジェクト実績値として収集されるデータを**表2.2**に示す。

表2.2　収集データ

収集するデータ	プロジェクト特性	定性的評価
◆ 開発規模（FP，KLOC） ◆ 成果物量（ドキュメントページ数） ◆ 開発期間 ◆ 開発工数 ◆ コスト ◆ テスト項目数 ◆ バグ数 ◆ 納品後の障害数 その他	● 開発形態（新規，機能追加，他） ● 顧客（業種，他） ● 適用業務（金融，流通，他） ● 開発基盤 ● 受注形態 ● 開発言語 ● 開発手法・技法 その他	○ 顧客との関係 ○ プロジェクトから得られた教訓・ノウハウ ○ メンバの育成 その他

これらの実績値が分析され，毎年，システムプロジェクト白書として社内に公開される。

システムプロジェクト白書は，組織の資産としてプロジェクトの実績を蓄積し，データを分析して現場にフィードバックすることを目的に作成している。また，NRIの品質マネジメントシステム（NRI-QMS）で設定されている「設計・開発プロセス」の品質目標の達成状況の評価にも使用される。主要記載内容は，次のとおりである。

システムプロジェクト白書の主要記載内容

＜実績データ＞
- ・プロジェクトのプロフィール
- ・プロジェクト規模・期間の分布，年度別推移
- ・開発工数と期間の相関
- ・予実績分析（予実績乖離）
  - －コスト，規模，工数
- ・その他

＜分析＞
- ・生産性関連指標（**表2.3**参照のこと）
- ・品質関連指標
  - －システム規模と不良発生密度の関連
  - －テスト数と検出不良の関連など

表2.3　各種生産性関連指標

項　目	説　　明
全体生産性	全社の生産性(FP/MM)。最小値，最大値，中央値，平均値
アプリケーション種別生産性	業務系，情報系の区分けによる生産性
顧客業態別生産性	金融，証券，流通などの業種別生産性
工程別工数比	概要設計，基本設計などの工程別に，全体に対する工数の比率を設定したもの

システムプロジェクト白書の生産性関連指標を類似プロジェクトの計画時に参照する。

（２）　推進体制

品質監理本部の生産性向上推進部がプロジェクト実績値に基づく分析と，プロジェクト白書の作成を実施する。見積り手法の指導も実施している。

また，品質監理本部には，全社のプロジェクトの状況を把握し，リスクの早期発見や対策の推進を行うプロジェクト監理部がある。プロジェクト監理部では，個別プロジェクトの支援も行う。その過程で，実際にプロジェクトの見積りに参加し，共同で積み上げ法や類似法による見積り，FP計測を行っている。

一定規模以上のプロジェクトは社内の公式設計レビューにおいて，見積りの確認が行われるが，その際にFPの計測結果についても確認する。

2.5　実施実績

（１）　簡易FP法の適用実績

適用にあたり業種は問わない。FPの測定は全社で実施するよう指導している。

全社的なプロジェク実績値の収集と分析結果の公開は,5年程度の実績がある。あわせてFPの実プロジェクトへの適用も開始している。FPの推定手法については，3年程度の実績がある。

簡易FPの習得はレクチャー・Q&Aなどで１人当たり２時間程度。FP概算法によるFPの計測は，概要設計が完了していれば，規模にもよるが１～２時間程度で実施可能。

（２）　顧客への見積り説明

顧客への見積り根拠の説明のしかたは顧客により異なる。仕様追加による規

模の増加も含め，全てFPで顧客に説明することはあるが，そのようなケースはまだ少ない。FPで説明できない場合は，FPをSLOCに換算して説明することが多い。FPからSLOCへの換算は，蓄積した実績データをもとに換算値を設定している。

2.6　当該見積り方法の優位点と課題

（１）　優位点

①　FPにより，開発言語に依存しない見積りが可能
②　FPを詳細なデータ項目の確定が行われる以前の段階でも適用可能
③　プロジェクト実績値に基づいた見積りの検証を行うことで，見積り精度が向上可能

（２）　留意すべき点

①　難易度などの調整は人に依存しがち
②　環境設定やユーザテスト，教育などの工数は，別途見積りが必要
③　システムの一部に再利用を含む場合の見積りは困難

（参考資料）　NRI開発標準フレームワーク

●主　旨●

「NRI標準フレームワーク」は，NRIグループで実施するシステム構築において，実施すべき活動(アクティビティ)および実施すべき作業(タスク)および作成すべき成果物を標準ガイドラインとして提供するもの。

また，再利用や参考となる雛型や部品および管理・設計・開発の実施ノウハウなどを蓄積するライブラリを全社的に提供する。

●主な内容●

（１）　標準フレームワーク工程定義：全体を概観したもので，システム構築の工程，アクティビティ，タスクを網羅する定義書。

（２）　アクティビティチャートおよび作成手順フロー：システム構築における作業の単位を定義し，おのおのについてインプット，作業内容，アウトプットを明示し，実施する順番を明確に示すフローチャート。

（３）　ドキュメント：成果物として社内で活用されている資料を提供。内容説明，詳細イメージ，ダウンロードファイル，事例ファイルなど。

第 3 章　TIS手法

3.1　取り組みの背景

バブル経済の崩壊後，企業のIT投資においてもシビアな投資効率が求められ，情報化による新たなビジネスの創出競争が激化する一方，投入資源の徹底した抑制が求められるようになった。この傾向は，インターネット全盛となった現在も変わらない。

新しいビジネスチャンスを得るために，また高い開発効率を実現するために，基盤技術は革新を重ねる。しかし，新規性の高い分野におけるソフトウェア開発は，プロジェクトの遂行に高いリスクを伴う。リスクを軽減するためには，先行事例を集め，できる限りすみやかに，その分野における知識体系をくみ上げる必要がある。見積りに関しても，その分野における標準的な生産性と生産性の変動要因を見きわめ，精度の高い見積りを実施することが，プロジェクトを的確にコントロールするための必要条件となる。

当社では，この必要性に応えるため，新規性の高い技術分野（1990年代であればクライアント/サーバ型システム，現在であればWeb系システムなど）の新規開発プロジェクトを対象に，従来型の積み上げ見積りと，過去事例の分析から算出した見積りとの比較検証を行い，見積り精度の向上に努めている。

3.2　見積り方法（モデル）

（1）　見積りの目的

この事例において，見積りモデルの導入目的はプロジェクトコントロールの精度向上であり，顧客交渉における利用は，スコープに入らない。顧客交渉はこのモデルのスコープ外で実施されており，プロジェクトが契約上の（または提案書上の）見積り工数をすでに持っている状態を前提としている。

そのため，このモデルでは，プロジェクトが目標とする上記の「契約上の見積り工数」を，過去の事例から得られる標準工数により検証することが目的に

なる。「契約上の見積り工数」は基本的に従来法による見積りにより算出されており，標準工数による検証は，基盤技術から独立した規模比較のためにFP値を用いる。これを図式化すると，以下のように表すことができる。なお，この比較検証のことを社内では「見積り検証」（**図3.1**）と呼んでいる。

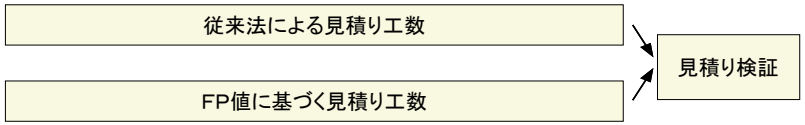


図3.1　見積り検証

ここでいう従来法とは，主に積み上げ法のことを意味するが，こちらの見積り手法は，ここでは省略する。

（2）　見積り時期

上記のように，この見積りモデルを用いるのは，プロジェクトが顧客と概算見積りを済ませ，ある程度開発が進んだ段階である。FPを規模計測の尺度としていることもあり，この見積りモデルは，①要件定義が明確になった段階，②外部設計が明確になった段階に使うのが効果的である。また，プロジェクトの完了時には，標準工数を算出するため，実績値（FP，工数，他）を収集する必要がある。この関係を図式化すると，**図3.2**のようになる。

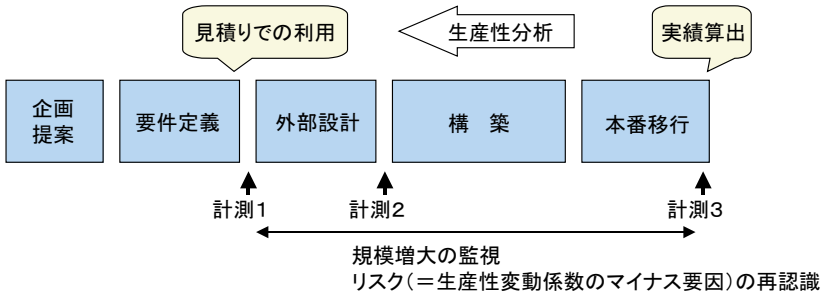


図3.2　FPの計測タイミングと見積り時期

計測３で得た実績値を用いて生産性を分析して標準工数を求め，計測1,2における見積りに利用する。

見積りにおいては，

①　規模を見積ることにより，規模増大の監視を行い，

②　生産性変動要因を評価することで，リスク，すなわち生産性変動係数のマイナス要因の洗い出しを行うことになる。

なお，「外部設計」は，システム設計工程の一部であるが，「アプリケーションの入出力が確定した時期」であることを明確にするために，「外部設計」という名称を用いている。

（３）　見積り対象

この見積りモデルで対象とするのは，アプリケーションの機能規模と相関する範囲の工数である。**図3. 3**では，「対象工程工数」とされた部分（太枠内の部分）が該当する。「付帯工数」は，従来法（積み上げ法）で見積った値を使用する。このため，「付帯工数」は，「（１）見積りの目的」で述べた見積り検証の対象にはならない。

また，「（４）見積り手順」で述べるように，工数を求めるための手順の一貫として，アプリケーションの機能規模と対象工程の生産性も見積りの対象になる。

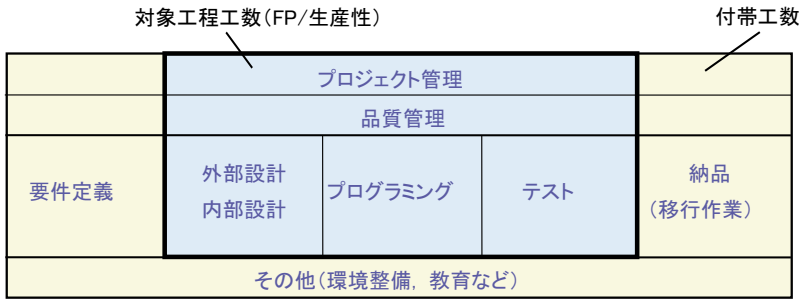


図3. 3　工数見積りの対象範囲

（４）　見積り手順

FP値と標準生産性による見積りの具体的手順を以下に述べる。

①　機能規模を見積る

１）アプリケーションの機能規模を計測する。

要件定義書/外部設計書をもとに，IFPUG法の計測ルールに基づき，未調整FPを計測する。

IFPUG法の調整係数はISO 14143のFunctional Size Measurementの定義に合わないため，使用しない。よって使用するのは，調整済FPではなく未調整FPである。

なお，要件定義段階では計測に必要な情報はそろっていないことが多いので，複雑度が不明の場合は，デフォルト値を使用する社内ルール（NESMAの概算法：The estimated function point countと内容的には同じ）を採用している。

２）規模変動係数を見積る。

システムの完成時点で規模がどの程度膨らむかを見積る。

見積時点の要件/設計の確定度をもとに，その時点の計測FP値（未調整FP）を１とすると，何倍になるかを予想する。

理想的には，要件定義，外部設計の各工程で，100％の完成度を得て，次工程に進むべきではあるが，現実のプロジェクトは，必ずしも開発領域の全てが，十分な完成度を得られるわけではない。よって，全体の中のどれだけの領域が未完成であり，その結果のどの程度の機能漏れが考えられるかを開発者の内観をもとに見積る。例えば，全体の２割が40％の完成度とすると，機能増加は１＋0.2×（1÷0.4－1）で1.3倍と見積ることができる。

また，外部設計完了時点で完成率がほぼ100％であったとしても，未経験分野の開発であれば，予想外のデータ参照や開発すべき機能の前提となる機能の見落としなどが発生し，開発完了時点までに機能規模は数％～数10％増加する。こういったいわば自然増加もあわせて，ここで見積る。

どの程度の規模増加があったかは，プロジェクト完了時点で情報を集計し，次回の見積りに対し，フィードバックをかける。

３）見積り規模を算出する。

得られた計測規模に規模変動計数をかけて，プロジェクト完了時の見積り規模とする。以下に式を示す。

見積り規模（FP）＝計測した機能規模（FP）×規模変動係数

②　プロジェクトの生産性を見積る

１）システム規模に応じた標準生産性を求める。

社内の標準生産性から，上記の見積り規模（FP）に該当する値を選ぶ。

規模の増大に伴う生産性の低下を加味し，標準生産性は規模が増加に従って低くなるように設定されている。よって，見積り規模に応じた生産性を今回の標準生産性として選択する必要がある。この場合の規模は，規模変動計数をかけた「見積り規模」であり，「計測した機能規模」ではないことに注意する。

２）個々の生産性変動要因を評価する。

今回のプロジェクトの状況に基づき，生産性変動要因を評価する。

当社の場合，経験の多いプロジェクトマネージャに対するヒアリングの結果，特に生産性に対して，影響が高いとされた上位４つの変動要因と「その他の変動要因」という１カテゴリの計５つの変動要因を設定している。このそれぞれにつき，今回のプロジェクトにおける影響度を評価する。

生産性に対する影響度であるから，その要因により生産性が，どの程度上がるのか(または下がるのか)を影響のない場合を±０％とし，何％向上(または低下)するかという形で評価する。例えば，類似システムの構築経験者の配置で生産性が２割上がるのであれば＋20%，新しい技術要素の利用で生産性が1割下がるのであれば－10%となる。

このとき，部門長(アカウントマネージャ)，プロジェクトマネージャ，開発現場のリーダ，サブリーダでは，見解が異なることが多い。個人の意見ではなく，上位者と開発者が話し合って，できるかぎり多くの視点から，影響度を分析し，プロジェクトとしての統一見解を作成することが重要である。

３）生産性変動係数を算出する

上で求めた生産性変動要因の評価値を全て加算する。式を次に示す。

生産性変動係数＝(生産性変動要因１の評価値(%)
＋生産性変動要因２の評価値(%)
＋生産性変動要因３の評価値(%)
＋生産性変動要因４の評価値(%)
＋その他の生産性変動要因の評価値(%))÷100

生産性に対する影響率であるので，本来は掛け合わせるのが正しいが，計算が複雑になるので，近似的に加算で代用している。プラス要因とマイナス要因は相殺されるので，プラスとマイナスが適度に釣り合っている場合は，問題がないが，マイナスに偏りがあり，生産性変動係数の合計が－50%を大きく下回る場合は，正しい見積りを得ることができなくなる。ただし，この場合は，プロジェクトはきわめて不利な状況にあるので，プロジェクトの前提を見直す必要がある。

４）見積り生産性を算出する。

１)で求めた標準生産性と，3)で算出した生産性変動係数から，当該プロジェクトの見積り生産性を算出する。見積り生産性の算出式を以下に示す。

見積り生産性(FP/人月)＝標準生産性(FP/人月)×(1＋生産性変動係数)

③　見積り工数を求める

①で求めた見積り規模と②で求めた見積り生産性から，見積り工数を求める。式を以下に示す。

見積り工数(人月)＝見積り規模(FP)÷見積り生産性(FP/人月)

④　従来法による見積りと比較する(見積検証)

得られた見積りを従来法による見積り(プロジェクトがすでに持っている見積り)と比較する。工数だけでなく，規模をどう見積っているか，生産性変動要因をどう評価したかについても，あわせて検証する。各見積り要素に対する両者の見解が食い違う部分があれば，その差がなぜ生じたかを追求する。どちらかの見積りに見積り要素の考慮漏れ(生産性変動要因の見落としなど)があれば，その部分を見直して再見積りを行う。

ここでの目的は見積りの精度を上げることであるので，見積りを一致させる必要はない。最終的にどういう見積りを採択するかは，全ての材料を吟味して，プロジェクトが決定する。

マイナス評価の生産性変動要因は，プロジェクトの阻害要因である。見積りに阻害要因を見込むことは，阻害要因の発生リスクに対し，対策費を積んでいるといえる。予算を積み，リソース投入により対応することもリスク対策だが，リスクそのものを除去する(例えば「未経験技術の採用」というリスクに対し，「経験ある技術への転換」という手を打つなど)ように対策することもできる。見積り検証では，単に工数見積りをするだけではなく，潜在するリスクを洗い出し，対策を明確にし，プロジェクトコントロールを強化することが目的である。

3.3　見積りモデルを支えるための社内活動

(１)　標準生産性の分析

標準生産性を得るために，各プロジェクトの情報をプロジェクト完了報告書とともに，全社で収集，分析する。

売上１千万以上または工数10人月以上のプロジェクトは，全て完了報告書を作成し，生産技術部門に提出する。収集する情報は，見積りモデルにおける入

力，出力の各パラメータの全て，すなわち，計測規模，規模変動計数，選択した標準生産性，生産性変動要因の評価，生産性変動係数，見積り工数と，これらの実績値，すなわち，実績規模，実績生産性，実績工数，生産性変動要因の実績値である。

生産技術部門において，これらの数値の統計処理を行い，標準生産性を社内にフィードバックする。また，規模変動計数，生産性変動係数の変動幅なども分析し，フィードバックをしている。

（２）　生産性変動要因の設定

当見積りモデルの構築に先立ち,社内の生産性変動要因について調査を行い，独自のカテゴリを設定した。調査は，経験豊富なプロジェクトマネージャに対し，以下の項目をヒアリングした。

ⅰ）当社内のプロジェクトにおいて，生産性に影響の高い要因はなにか。

ⅱ）それらの要素は，常に影響があるのか，特定の場合にのみ影響があるのか。

この調査結果をもとに，常に影響が高い４要因を当社の生産性変動要因として設定し，その他の要因を，「その他の変動要因」にて評価することとした。

ただし，ここで設定した変動要因は，3.5節の（１）で後述するように，使用局面を限定して設定しているため，他の局面で使用する場合には，別途ヒアリングを実施し，再設定する必要がある。

（３）　見積り支援

当見積りモデルは，生産性変動要因や，規模変動計数など，開発者の内観にたよる部分が多い。そのため，見積り支援専任者をプロジェクト関係者以外から設置し，見積り支援をするのが効率的である。

支援内容としては，FP計測結果のレビューと規模変動係数，生産性変動係数の設定に関するアドバイスがある。特に規模変動計数と生産性変動係数に関しては，個々人の内観の差による誤差を極力抑えるために，評価者と詳細な討議を行う。

１回あたりの支援は,1～２時間程度。支援を行う時期は，各プロジェクトでの見積り検証時期２回と，完了時期の１回で計３回であり,1プロジェクトあたりの支援所要時間は半日～１日程度である。よって，少数の支援者でも，かなり多くの範囲のプロジェクト支援を受け持つことができる。

3.4　見積りモデルの有効性

比較検証することにより，従来法の精度は飛躍的に向上する。特に，FP法見積りにおいて，FP見積りと積み上げ見積りの間に乖離がある場合，両者の比較検証を通して，見積り精度は格段に向上する。ここでいう「精度」とは,「数値として予定と実績が近い」という意味ではなく,「考慮に入れるリスクの範囲が広がり」，それに伴い「見積り工数の実現性が向上する」という意味である。

当社では,1997年にデータ収集を開始。2001年までに完了プロジェクト約120件のデータを収集し，約90件の見積り検証を実施。うち約４割のプロジェクトで従来法見積りとFP法見積りの乖離を検出した。

3.5　当該見積り方法の優位点と課題

（１）　当該見積り手法の優位点

当手法は，実践的なプロジェクトコントロールを主眼においている。そのため，使用する局面を限定し，見積りモデルは可能な限りシンプルにしている。

使用する局面を限定するとは，つまり，①社内の請負開発，②比較的新しい技術分野(利用当初であればCS系，現在ではWeb系)，③新規開発(またはある程度の規模のある機能のまとまりの追加)，④受注済みプロジェクトの要件定義～外部設計局面での利用，と利用の前提を具体的に絞り込むことである。これにより，a)生産性変動要因を当該条件に合ったものに絞込み，見積りモデルを単純化できるし，また，b)見積りの目的が明確になり，特異な値が出た場合でも，単なる見積の技術論ではなく，より実践的な対応策をとることができる。特に，生産性変動要因を５つに絞り込み，積算ではなく加算での計算式を用いることで，見積りに関する専門知識をもたない開発要員にも，理解しやすい平易なモデルとなっている。

一方，規模の計測には，国際標準であるIFPUG法をカスタマイズせずに使用することで，得られた数値を他の手法で得られた数値と比較することが容易にできるようになっている。

（２）　不得意な分野など，利用するに当たって留意すべき点

使用する局面を限定している分，条件に合わないケースへの適用は注意を要する。条件が異なれば，標準生産性と生産性変動要因は異なるので，条件に合わないプロジェクトに対して，同じ生産性と変動要因を使用してはいけない。

生産性変動要因の特定からやり直すべきである。

規模の計測にIFPUG法を利用していることから，IFPUG法で計測できない（または妥当な数字の得られない）分野のアプリケーション，すなわち通信系や制御系への適用は困難である。

200FP以下の小さな案件では，個人差による生産性変動のほうが大きく出てしまうため，この見積りモデルの適用には適さない。保守開発は，開発するアプリケーションの大きさと単純に相関しないため，同じくこの見積りモデルでは適用できない。

要件定義以前の工程での適用は，FPによる規模見積りができれば可能であるが，FPを見積るための材料が極端に少ないため，FPによる規模見積りに熟練した者でないと適用はむずかしい。

第４章　日立システムアンドサービス手法

4.1　取り組みの背景

一般に，ソフトウェア開発では，見積りから仕様が固まるまで，損益の大半が決まるといわれている。そのため，要求仕様の明確化や見積り精度向上が重要課題である。従来，ソフトウェアの規模見積り手法として，ステップ数見積り法が適用されてきたが，開発環境や言語の多様化，ソフトウェア部品化技術の進展などにより，ステップ数を規模見積りの指標にするのが困難なプロジェクトが増加している。また，同じ機能のソフトウェアでも，開発方法によりステップ数が変動し，既存部品の有効活用などにより開発ステップ数を小さくした場合，生産性を低く評価され設計努力が報われないなどの課題もある。このような状況から，ステップ数見積り法に代わる規模見積り法が求められてきた。

当社も上記と同様な状況があり，この解決策の一つとして，1996年６月にJFPUG（日本ファンクションポイントユーザ会）に加入し，IFPUG（International Function Point Users Group）法を導入した。その後，FP法適用ガイド作成，FP計測技術者育成（現在，約1 000名以上教育済），生産性データの蓄積と分析，FP法有効性分析，FP計測効率分析などを推進してきた。1998年には，当社の生産効率管理にFP法を採用した。さらに，各種のプロジェクトへのFP法適用経験を踏まえ，当社独自の「機能要素見積法」を開発し，それを反映した機能定義支援ツールである「機能整理FP算出ツール」を開発・運用し，FP計測精度・計測効率向上に努めてきた。また，2000年からFP法の顧客教育を開始するとともに，社外の各種団体でFP法関連の講演・セミナなども実施している。

現在，FP法を規模見積りの中心手法として位置づけ，新規開発プロジェクトへは原則適用必須として推進している。さらに，規模管理のPDCAサイクル，すなわち，見積り精度向上（PLAN）－変更管理（DO）－実績・ノウハウ把握（CHECK）－見積りへのフィードバック（ACTION）で活用している。

4.2　見積り方法

（１）　概　要

ソフトウェア開発における見積りには，規模見積り，工数見積り，期間見積り，金額(コスト，価格)見積りなど，各種の見積りが含まれる(**図4.1**)。以下，当社で適用している，FP法による規模見積り手法およびそれを踏まえた工数見積り手法を中心に述べる。なお，当社では，規模見積りにはFP法のほかに，ステップ数を用いた方法なども採用しているが，FP法の比率を高めている。

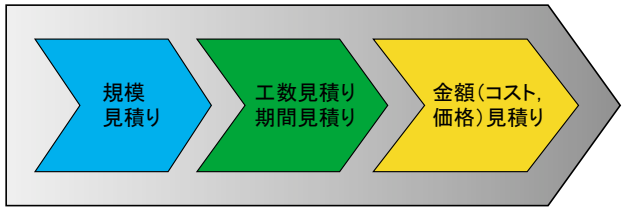


図4.1　ソフトウェア開発見積り作業

（２）　FP法による規模見積り手法

(a)　FP規模見積り手法概要　　プロジェクトの各段階で見積りの前提とな

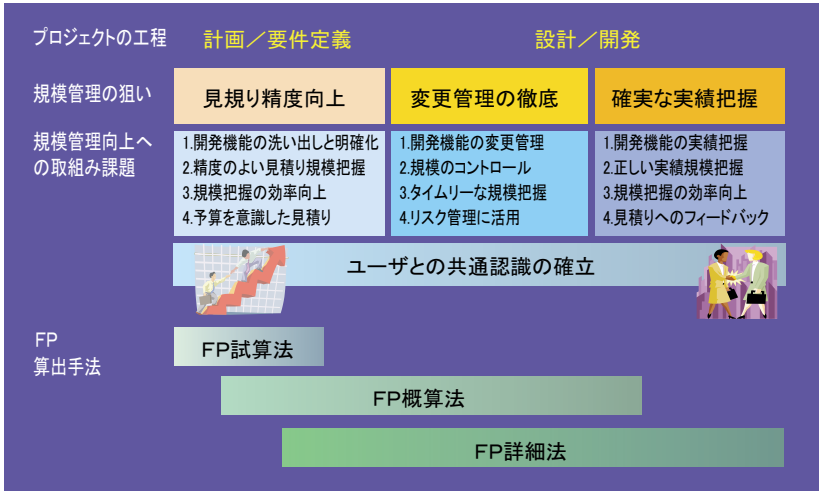


図4.2　開発工程と規模(FP)見積り手法

る情報は変化する。それに応じた規模見積りを実施する必要がある。そのため当社では，FP法の適用において，FP試算法，FP概算法，FP詳細法(IFPUG法)を採用している。なお，一定規模以上の新規開発プロジェクトは，原則として，FP法(FP概算法またはFP詳細法)適用を必須にしている。**図4.2**および**表4.1**に，開発工程と規模(FP)見積り手法の関係およびFP算出法の概要を示す。

表4.1　FP算出法の概要

項番	開発工程	見積り段階	規模見積り手法	規模見積りパラメータ	FP算出法	備考
1	システム化計画	機能が把握できない段階	FP試算法(当社手法)	画面数，帳票数，ファイル数，類似システム規模，予算，その他(上記の概算でも可)	下記パラメータで把握できる範囲で，試算FP１～４などを算出し，試算FP１～３およびチェック用としてFP４の値などを総合的に判断し，試算FPを決定。 ・画面数＋帳票数→試算FP１ ・ファイル数→試算FP２ ・類似システムステップ数→試算FP３ ・予算→試算FP４	・試算FPは規模の概略把握に使用し，正式見積りには使用しない。
2	要件定義	機能一覧レベルが把握できる段階	FP概算法(当社独自の機能要素見積法をベースにした手法)	・機能一覧，機能要素など	規模(FP) ＝ILF数×重み１ ＋EIF数×重み２ ＋EI数×重み３ ＋EO数×重み４ ＋EQ数×重み５ (注１)重みは当社実績より設定 (注２)ILF,EIF,EI,EO,EQは4.2節(２)の(c)④参照	・機能一覧，要素機能を前提とするので比較的高い精度の規模見積り可能 ・IFPUG法に比べ見積り工数がかからない(数分の１)
3	設計	関連ファイル数や入出力項目数がほぼ把握できる段階	FP詳細法(IFPUG法)	IFPUG法パラメータ	IFPUG法による	・精度の高い規模見積りが可能 ・FP概算法に比べて見積り工数がかかる

(b)　FP試算法

①　概要：開発対象システムの機能把握が困難なときに適用し，開発対象システムの想定画面・帳票数やファイル数などからFPを試算したり，類似システムの規模からもFPを試算する。また，想定開発予算からもFPを試算し，開発範囲と予算との妥当性チェックなどに使用する。そのようにして，おおよその規模把握および見積りクロスチェックに試算FPを使用し，正式見積りには，試算法ではなく，後述する，FP概算法(機能要素見積り法)またはFP詳細法(IFPUG法)を使用する。

②　入力：画面数，帳票数，ファイル数，類似システム規模，予算など(概算でも可)。

③　出力：FP

④　基本的なアルゴリズム･方式：下記の試算FP１～試算FP４を可能な限り算出し，試算FP１～試算FP３を総合的に判断し，試算FP４をチェック用に用い，試算FPを求める。

試算FP１＝(画面数＋帳票数)×a

試算FP２＝更新系ファイル数×35＋参照系ファイル数
×15(NESMA法)

試算FP３＝類似システムステップ数×b

試算FP４＝予算×c

（注１）　a～c＝FP換算値

（注２）　NESMA：Netherlands Software Metrics Users Association

⑤　見積り方法の前提条件：上記入力パラメータが把握できること。

⑥　見積り時期：システム化計画時など，機能把握が困難な段階に適用。また，他の見積り手法のクロスチェックなどにも使用。

⑦　見積り対象：ソフトウェア開発全般。

⑧　見積り活動：本FP試算法は正式見積りには使用せず，正式見積りにはFP概算法(機能要素見積法)またはFP詳細法(IFPUG法)を使用する。

(c)　FP概算法(機能要素見積法)

①　概要：規模見積りのキーワードは「見える化」であり「精度向上」である。そのためには，前提となる要件，特に，機能の明確化がカギである。当社では，規模見積りにFP法を採用するとともに，粒度をそろえた機能要素を選択指定するだけで機能定義ができる「機能要素見積法」を開発

し，機能の明確化に取り組んできた。

FP法による規模見積りでは，現在，ここで述べるFP概算法(機能要素見積法)が当社で最もよく使用されている。

②　入力：機能一覧およびそれを構成する機能要素，仕様の明確度など。

③　出力：FP，各種分析情報。

④　基本的なアルゴリズム方式：機能要素からIFPUG法のファンクションタイプ(ILF，EIF，EI，EO，EQ)数を決定し，次の式から規模(FP)を求める。

規模(FP)＝ILF数×重み1＋EIF数×重み２＋EI数×重み３ ＋EO数×重み4＋EQ数×重み５

（注１）ILF：内部論理ファイル，EIF：外部インタフェースファイル
EI：外部入力，EO：外部出力，EQ：外部照会

（注２）重み１～重み５は当社プロジェクトの実績より設定。

⑤　見積り方法の前提条件：機能一覧，機能要素が把握できること。

⑥　見積り時期：主として，要件定義段階。プロジェクト完了時にはFP詳細法(IFPUG法)を適用。

⑦　見積り対象：ソフトウェア開発全般。

⑧　見積り活動：規模見積りをベースに，工数見積り，期間見積り，金額見積りなどを実施。

⑨　併用見積り手法：規模見積りとしては，FP試算法やステップ数法などを併用する場合もある。

(d)　FP詳細法(IFPUG法)

①　概要：FP詳細法としてはIFPUG法を採用。

②　入力：機能一覧，ファンクションタイプ，関連ファイル数，データ項目数など。

③　出力：FP，各種分析情報

④　基本的なアルゴリズム方式：IFPUG法のルールに従って機能ごとに，ファンクションポイントを求め，集計する。

詳細はIFPUG法に関する各種出版物を参照されたい。

⑤　見積り方法の前提条件：上記入力パラメータが把握できること。

⑥　見積り時期：要件定義段階から適用する場合もあるが，主として，設

計段階およびプロジェクト完了時に適用。

⑦　見積り対象：ソフトウェア開発全般。

⑧　見積り活動：規模見積りをベースに，工数見積り，期間見積り，金額見積りなどを実施。

⑨　併用見積り手法：規模見積りとしてはFP試算法やステップ数法などを併用する場合もある。

（３）　工数見積り手法

機能規模が決まっても，工数は該当プロジェクトの特性(技術要件，品質要件，発注者要因，供給者要因など)に左右される。工数見積りはプロジェクトの作業量の「目標設定」であり，「効率化」が求められるとともに，「実現可能性」が求められる。そのために「プロジェクトリーダの意志」を反映したものでなければならない。そのようなことを考慮しつつ，概略，以下のような考え方で工数を見積る。なお，必要に応じて，積み上げ法など，他の手法も併用する。

①　プロジェクト全体工数＝ソフトウェア開発工数＋その他工数

②　ソフトウェア開発工数＝開発規模(FP)／プロジェクト生産性

③　プロジェクト生産性＝開発パターン別基準生産性
×該当プロジェクトの特性

なお，開発パターンは，システム形態などを考慮し設定している。

4.3　見積り方法の前提条件

（１）　手法別の前提条件

4.2節の各手法の⑤～⑨に記述。

（２）　体制・役割分担・企業文化

プロジェクト全体に対しては，プロジェクトマネジメント部，生産技術部，品質保証部，購買部など，支援・管理部門が支援しており，見積りに関しては，生産技術部内に専門グループを設置し，見積り支援を実施している。

4.4　精度向上のための活動

（１）　差異分析，フィードバックの方法

実績データの収集と基準生産性へのフィードバックを推進中。

（２）　精度向上のための具体的な方法

規模見積りの精度向上のために，見積り機能の洗い出し・明確化・定義・分析を支援し，FPを自動算出する「機能整理FP算出ツール」を開発し，適用推進している。

機能整理FP算出ツールは機能一覧を入力するとともに，それぞれの機能を構成する機能要素を選択指定することにより機能を定義し，FPを求めるツールであり，**図4.3**に示す機能を備えているのが特徴である。なお，機能整理FP算出ツールはFP法適用時に使用必須としている。

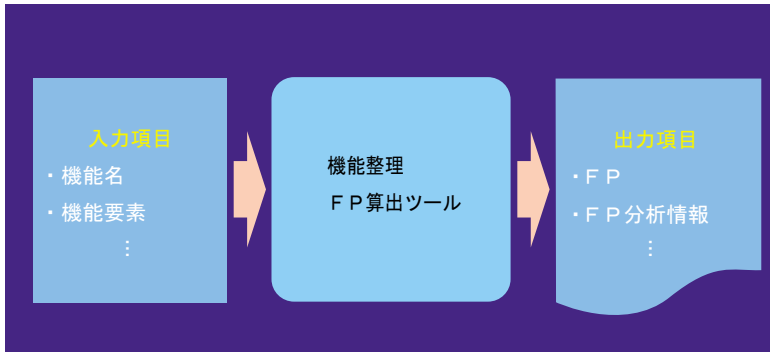


図4.3　機能整理FP算出ツールの機能

①　機能の洗い出し･明確化･定義機能：本ツールの特徴は，粒度をそろえた機能要素が事前に準備されていることであり，それらの機能要素を選択指定するだけで機能定義ができることである。これにより，機能の洗い出し，定義が簡単にでき，機能の明確化が図れる。また，機能を構成する機能要素をチェックすることにより，見積りから漏れた機能要素の洗い出しができる。

②　機能一覧および開発情報一覧表示機能：各機能とそれを構成する機能要素を一覧表示することにより，ユーザ側と開発側で開発機能に関して共通認識に立つことができ，総合判断およびトラブル防止に有効である。

③　FP比率分析機能：機能タイプ別FP比率，定義妥当性総合評価などにより，機能洗い出し妥当性検証や重点追加洗い出し分野の明確化ができ，漏れ機能の重点的洗い出しができる。

④　仕様明確度分析機能：個別機能ごとに指定した仕様の明確度に基づき，仕様明確度分析ができ，追加洗い出し分野の明確化やリスク管理に有効である。

⑤　FP自動算出機能：定義した機能から，自動的にFPが求まり，タイムリーな規模把握ができる。

⑥　多様なプロジェクトタイプや見積り局面に対応する機能：新規開発プロジェクトおよび機能改良プロジェクトに適用でき，かつ，見積りおよび実績把握に対応している。

本ツールを用いて定義した機能をマクロ分析(例えば，参照系機能，更新系機能などの機能グループの比率分析)およびミクロ分析(例えば，機能要素の関連チェック)することにより，機能定義漏れなどを防ぎ，規模見積りの精度向上を図っている。

また，工数見積りの精度向上のために，実績データの収集，レビュー，分析を実施している。さらに，見積り支援のための専門グループによるプロジェクトのレビューなども実施している。

⑦　その他関連機能

4.5　実施実績

過去の実績については,4.1節の取り組みの背景でも触れているとおり,1996年からFP法を適用してきた。

（１）　適用分野(業種，システムプロジェクトの特徴)

業種にかかわらず，一定条件以上の新規ソフトウェア開発プロジェクトへはFP法適用を必須としている。また，機能改良プロジェクトにもFP法を適用している。そのため,1 000名以上に対してFP計測法を教育済。

（２）　見積り時期とその精度

見積り時期により見積り精度は当然異なるので，段階的見積りや再見積りを推進している。

4.6　当該見積り方法の優位点と課題

（１）　当該規模見積り手法のウリ

当該規模見積り手法，特にFP概算法，FP詳細法は，次のような優位点がある。

①　プロダクトスコープが明確となる：見積り前提機能の内容と範囲の明確化により，プロダクトスコープが明確となる。

②　ユーザにわかりやすい：ユーザにわかりやすい表現のため，ユーザが優先順位を考慮し機能の取捨選択ができ，ユーザと開発者が共通認識に立てる規模見積り手法である。

③　規模見積り精度の向上：機能定義が機能要素の選択指定により行え，かつ，上記マクロ分析およびミクロ分析により機能定義の妥当性検証，漏れ機能の追加などができ，見積り精度の向上が図れる。

④　規模見積り効率の向上：機能定義によりファンクションポイントを自動的に得ることができ，規模見積り効率がよい。

⑤　設計効率の向上：開発者が早期に機能範囲を把握でき，設計効率が向上する。

⑥　プロジェクトコントロールに有効：機能・規模のタイムリーな把握と変更管理により，プロジェクトコントロールがしやすい。

（２）　当該規模見積り手法適用上の留意すべき点

規模見積りは，機能の抽出がポイントであり，IFPUG法の概要を知っていると適用しやすい。ただし，知らなくても適用できるようツールに工夫をこらしている。社内教育も継続実施している。

工数見積りは，そのためのベースとなる地道な生産性データ収集活動が不可欠である。

第５章　日立製作所手法

5.1　取り組みの背景

昨今のビジネスアプリケーション開発では，見積りに関連するプロジェクトのトラブルが多くなっており，その解決のため客観的で精度の高い見積り手法が，重要となっている。見積りを客観的に行う方法としては，FPを用いた見積りが普及してきているが，画面やファイルなどのレイアウトの設計が完了しないと見積りを行うことができないため，基本設計完了などの時点にならないと適用できない。また，短時間で見積りを行うことがむずかしいといった問題もある。このため，設計前の要求定義段階などでもほぼ正確な見積りが，すばやく行えるように工夫を行ったのが，本見積り手法である。

5.2　見積り方法（モデル）

本見積り手法の流れを**図5.1**に示す。

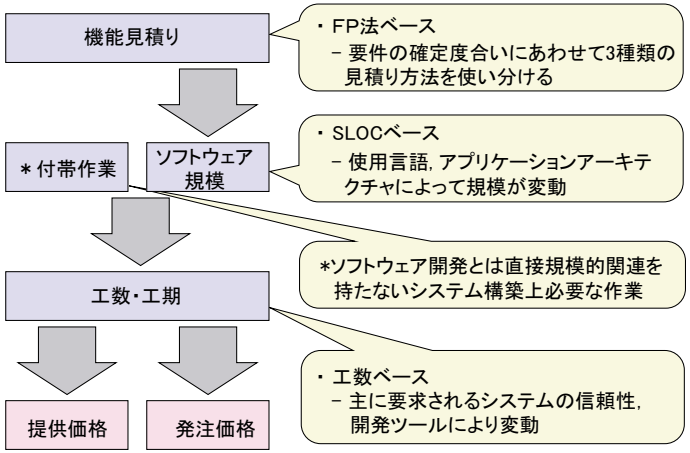


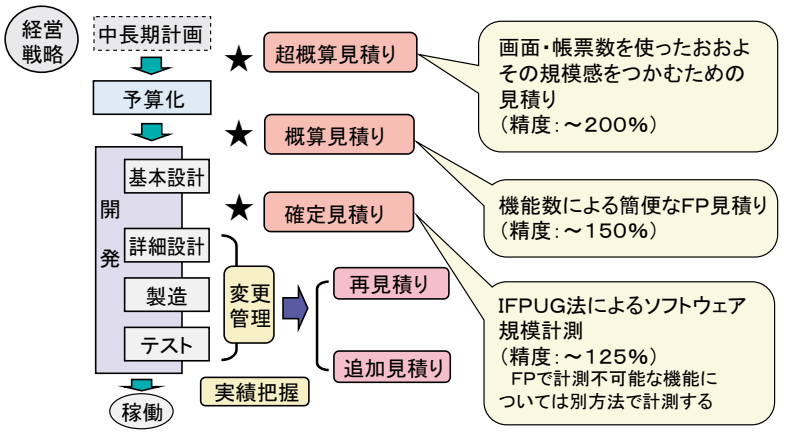
図5.1　本見積り手法の流れ

本見積り手法では，最初に機能見積りを行い，FPを求める。次に，FPとSLOCの換算テーブルに基づき，SLOCベースの見積りを算出する。いったんSLOCベースの見積りを算出するのは，品質管理などではSLOCベースで管理するのが適しているからである。また，FPが適用しにくいケースも考慮し，SLOCの新規入力や修正も可能として，FPとSLOCを併用できる形をとっている。

SLOCからは，システムに要求される信頼性や開発ツール，プロジェクト特性などのパラメータを用いて，最終的に開発工数，工期を算出する。

本見積り手法の適用時期と見積り方法の対応関係を**図5.2**に示す。

機能見積りは，適用時期や要件の確定度合いに応じて図5.2のように予算化前の超概算見積り，開発開始前の概算見積り，基本設計完了後の確定見積りの３種類の見積り方法があり，これらを使い分ける。最も適用ケースが多いのは開発開始前（引合い/受注段階）の概算見積りであり，その段階で把握が可能なマスタや機能の個数（DBテーブル数，画面数，要素機能（例えば，新規登録機能など）の個数など）の実測値または推測値を入力データとして，これらの値とFPとの換算テーブルに基づきFPを算出する。



（注）予算化の中に要件定義を含む。
基本設計はシステム設計とほぼ同じ。
詳細設計はソフトウェア設計とほぼ同じ。

図5.2　本見積り手法の適用時期と見積り方法の対応関係

5.3　見積り方法の前提条件

（１）　見積り対象

企業システムなどのビジネスアプリケーションソフト。

（２）　見積り活動・体制・役割分担・企業文化

見積り技法の整備，見積り精度向上の活動，企業内への普及活動は専任グループが行う。

実際のプロジェクトの見積りは，各プロジェクトのSEが行い，PMOメンバが内容をチェックする。見積り作業時に，必要に応じてPMOメンバが，見積り技法の活用法や，細かいノウハウについてSEのサポートを行う。

また，見積りレビュー会議を行う。リスクが高い案件は，レビュー前に専任グループが入って内容確認を実施する。

なお，本見積り手法の普及に当たっては，習得時のコストを低減するため，マニュアルやガイドラインを提供し，必要なときに学習できるようにしたり，管理者層，担当者層に，それぞれに向けた教育（座学，eラーニング）を常時開設するなどの工夫を行っている。

概算見積りは，簡易な方法で実施することができるため，IFPUG法で見積る場合の１／３～１／５程度の時間で実施することができる。

最近は，Webシステムなど，SLOCでは，規模を表しにくいケースも多く，本見積り手法のように，FPをベースとした見積り手法が，ユーザ側にもベンダ側経営層にも理解が得られるようになってきている。このため，全社的な展開がしやすくなってきている。また，本見積り手法を適用することにより，従来と比べて見積り根拠が，明確となるメリットもあるため，経営層や現場の理解も得やすくなっている。

（３）　併用見積り手法

基本的に本見積り手法は，単独で適用するものであるが，工数積み上げや類推法などの他の見積り手法と組み合わせて，見積り精度を向上させる場合もある。例えば，類似するプロジェクトの実績情報（規模，工数，プロジェクト特性，プロジェクト運営上の工夫点など）や見積り情報（規模，工数，プロジェクト特性，プロジェクトリスクなど）を活用する場合もある。

5.4　精度向上のための活動

専任グループは，見積り精度を向上させるため，半年に１回，実績データに基づき，FP，SLOC，開発工数などを算出するときの換算テーブルのパラメータの見直しを行っている。

5.5　実施実績

（１）　適用分野（業種，システムプロジェクトの特徴）

FPを用いた見積りは，1995年より試行を開始。大規模なプロジェクトや顧客からFPでの提示を求められるケースではFPを使用し始めた。全面的に本見積り手法を採用したのは2003年からで，制度化したのは2004年からである。現在，大規模新規プロジェクトを中心に適用している。

FPによる見積りは，100件以上のプロジェクトで実施済みである。ただし，FPによる実績値の計測は，サンプリングしたもののみであり，数は多くない。

（２）　見積り時期とその精度

超概算見積りや概算見積りは早い段階で行う見積りであるため，IFPUG法で行う確定見積りと比べて精度は低くなるが，概算見積りの場合，見積ったFPの精度は，最大150％（実績値/見積り値）程度である。

5.6　当該見積り方法の優位点と課題

（１）　当該見積り手法のウリ

本見積り手法のメリットは次の点である。

① 引合い/受注段階（仕様があいまいな段階）などの早い段階で見積りが行えること。

② FPという一般的な尺度を用いながら，IFPUG法などの厳密な方法で見積りを行う場合と比べて，1/３～１／５程度の時間で見積りが行えること。

③ SLOCで見積りを行っている場合との認識を合わせられること。

（２）　不得意な分野など，利用するに当たって留意すべき点

SLOCや開発工数などを算出する場合のパラメータは，前提となる開発支援ツールによって変化してくる。昨今の開発支援ツールの進化は早いため，それに対応したパラメータの見直しを継続して行っていく必要がある。

第６章　ファンクションスケール法（富士通）

6.1　取り組みの背景

（１）　見積りの範囲

ITベンダにとって，見積りには次のような２つの側面がある。

① ユーザ/ベンダ間でITシステム開発にかかわる契約を締結するための見積り。

② ベンダ内のプロジェクト実行計画/推進のベースラインを策定するための見積り。

いずれも，基本的に「量」を見積り，それをもとに「価格もしくは費用」ならびに「納期もしくは期間」を見積る点は同じである。

ITシステムの構築にかかわる一連の見積りは，おおむね**表6.1**のようなものである。

表6.1　見積りの対象範囲

<table>
<tr><th>企画/分析フェーズ</th><th colspan="2">設計/開発/試験フェーズ</th><th>運用/評価フェーズ</th></tr>
<tr><td rowspan="6">調査/企画</td><td colspan="2">リスク</td><td></td></tr>
<tr><td>アプリケーション/パッケージカスタマイズ</td><td rowspan="5">移行関連</td><td rowspan="5">運用/保守</td></tr>
<tr><td>付帯（開発支援整備など）</td></tr>
<tr><td>システム構築</td></tr>
<tr><td>ミドルウェア/PP/PKG</td></tr>
<tr><td>ハードウェア</td></tr>
</table>

（２）　概　況

前述の見積り対象のうち主要な部分である，ハードウェア/ミドルウェアなど，システム構築，アプリケーションに関して，以前と現在では状況が変わってきている。かつての，メインフレームによるシステム構築では，以下のように経験的な見積りで対応してきた。

① ハードウェア/ミドルウェアの見積りは，各システムごとにトランザクション量やデータ量からハードウェア構成を見積り，機能構成からミドルウェアを見積り，それらから価格を見積った。

② システムの方式･運用などの設計･構築作業は，有識者の経験から作業項目，作業工数，価格を見積ってきた。

③ アプリケーションに関しては，やはり有識者の経験から規模（SLOC）を見積り，これをもとに工数，価格を見積ってきた。

その後，オープン製品主流のクライアント/サーバシステム，Webシステムの浸透に伴って，製品が次々に開発されるとともにシステム方式や開発方法も種々発展し，組み合わせの種類が飛躍的に拡大し続けて，経験的ノウハウによる見積りでは，健全なプロジェクト運営がむずかしくなっている。

当社ではこうした状況を踏まえ，見積り技術の整備に取り組んでいる。

- ハードウェア/ミドルウェアは，システム規模や業務のカテゴリから基本的な構成をテンプレート（事例をもとに事前検証したシステムパターン）化し，相当するテンプレートとその可変部分の見積りから構成を決め，価格を見積るように変えつつある。

  これは，個々の製品がいかに良くても相性などで必ずしも最良とは限らないため，個々に検証するとなると，時間も費用もかかることになる。そこで，あらかじめ頻度の高いケースを検証して，テンプレート化することで，時間，費用，品質を確保するための手段である。この技術が見積り方法も変えてきている。
- システムの方式・運用などの設計・構築作業は，WBSテンプレートに従って必要な作業項目を選択し，作業項目ごとに工数，価格を見積る。
- アプリケーションに関しては，Webシステムを中心に，従来の出来高ベースの見積り（SLOC）から機能規模の見積りにシフトし，それに基づいて，工数や価格を見積る方向に変えつつある。

このうち，特にアプリケーションの「量」，すなわち規模の見積りに関してはユーザとベンダ間の共有，ならびにプロジェクト計画/推進にとって，ポイントであると考えてきた。

機能規模の見積りとしては，IFPUG（International Function Point Users Group）が，推進するファンクションポイント（FP：Function Point）がよく知られている。しかし，これに従った見積りは，見積りに訓練が必要であることと，

かなりの時間と稼働を要する。また，プロジェクトの詳細計画策定や推進(コントロール)には，種々のベンチマーク指標(品質指標や工数産出指標など)が必要であるものの，FPの限界もわかってきた。

そこで，FP法の簡易版として，新たに，ファンクションスケール(Function Scale。以下，FSと略す)を設定し，これによる機能規模ならびに工数/価格や期間の見積り法を整備した。

6.2　見積り方法(モデル)

(1)　コンセプト

見積りの現状を考えると，見積り時期により，見積りに使える情報や求める精度のレベルが変わる。手間や精度を考慮し，見積りのタイミングに合わせて複数の方法を使い分けできるが，一つの枠組みの中で活用できる方法として，整備することをコンセプトとした。

また，ユーザに提示する単位(たとえば，システム単位や提供機能単位など)だけでなく，プロジェクトコントロールの単位(ライフサイクルの各フェーズで，スケジュール管理，品質管理などを行う単位)で規模などが求められることも考慮した。

本節では，当社における，クライアント/サーバあるいはWebベースのオンラインシステムにおける見積りの取り組みについて紹介する。

全体のおおまかな見積りの流れは，**図6.1**に示すとおりである。

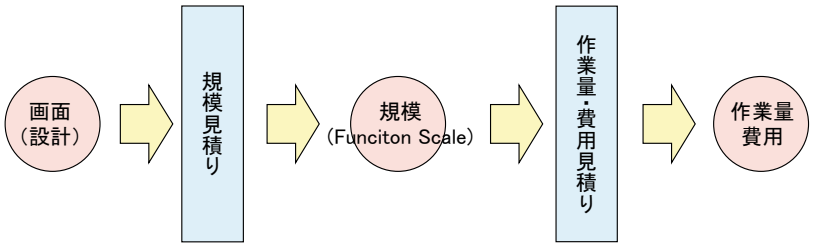


図6.1　見積りの流れ

(2)　規模見積り方法の概要

一般的なクライアント/サーバ，Webシステムでは，機能をフロント系(画面処理)とバック系(業務ロジック処理)とに分割できる。Webシステムのイメージを**図6.2**に示す。

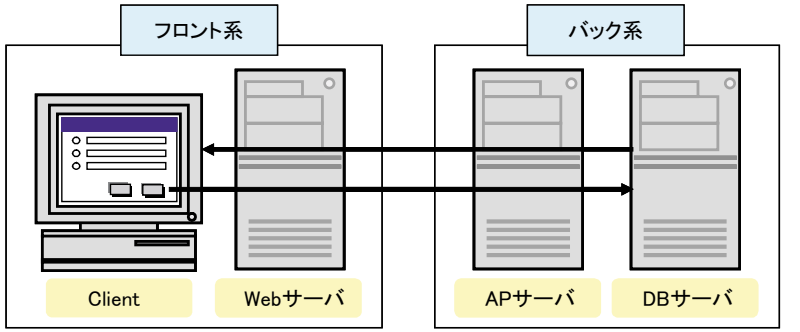


図6.2　Webシステムのイメージ

FSの見積りは，画面単位に，画面の構成(レイアウト)から，バック系の業務ロジック処理まで含めた規模を計測する。そのために，規模を表すモノサシとしてFSを設定し，規模を画面構成から計測するための基準値を設定した。見積りのイメージを**図6.3**に示す。

ただし，画面設計が完了するまでは，画面設計情報がないため，画面設計前の段階では，業務処理(仮想的な画面)の難易度別にFS基準値を設け，各難易

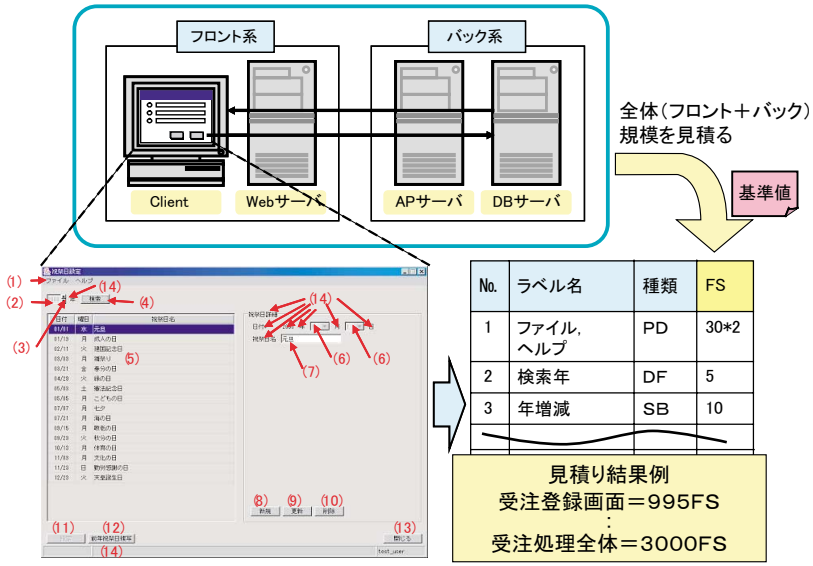


図6.3　ファンクションスケール(FS)での見積りのイメージ

度に振り分けた処理数と基準値から全体の規模見積りを行う。このとき，基準値テーブルだけでは，難易度ランクの選択がむずかしいため，ランク条件や処理イメージ(画面イメージなど)をカタログとして整備した。

以上の見積り方法の概要を**表6. 2**に示す。

表6. 2　FS規模見積り方式概要

見積り方法	タイミング	方法の概要	出力	入力	見積り式
概算見積り	基本設計前	要件レベルの段階で設計書がなくても，カタログベースにユーザとイメージを合わせて見積る方法	FS（規模）	機能の実装難易度ごとの概算FS基準値，業務処理数(注)	概算FS＝Σ(当該ランクの概算FS基準値×当該ランクの業務処理数) ランク：機能の実装難易度
詳細見積り	基本設計完了以降	画面の設計情報(レイアウト)をもとに見積る	FS（規模）	画面項目種類ごとの簡易FS基準値，種類ごとの画面内項目数	画面ごとのFS＝Σ(画面項目ごとの簡易FS基準値) 全体FS=Σ(画面ごとのFS)

(注) クライアント画面からの要求によりデータベースに対する登録・更新・削除・検索を行い，その結果を返す一連の流れを１業務処理とカウントする。

当カタログは，当初は一般的なサンプル画面集であった。しかし，現在業種別カタログとして順次整備中である。こうすることによって，カタログからの該当ランクの選択が容易になり，見積りの手間と精度を向上させることができる。

見積りは画面単位に行い，その結果を合計することにより，見積り対象範囲全体の規模を算出する。見積り単位が明確で，担当者間の認識の差がなくなる。また，見積りにあたってはプロジェクト特性(変動要因)を加味する場合もある。

変動要因としては，以下のものを考慮し，基準値に基づく規模算出後の調整計数として使用する。

- 業務の難易度
- 品質要求のレベル(性能要件，操作性要件など)
- エラーチェックの種類
- ビジネスロジックの複雑度など

調整係数範囲としては，おおむね0.9～1.3程度となっている。

（３）　工数・費用見積り方法の概要

COCOMO IIをベースに，当社用パラメータにカスタマイズした見積り式により費用を見積る。

6. 3　見積り方法の前提条件

（１）　見積り時期

見積りの時期に合わせて規模見積り法を使い分ける。見積り時期と見積り法の対応を**図6. 4**に示す。

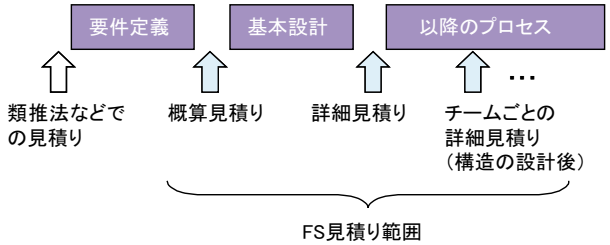


図6. 4　見積り時期

（２）　見積り対象

当見積り手法はアプリケーションを対象としている。また，見積り基準値は適用する開発技法，フレームワークにより変わる。

現在の当社基準値は，SDAS(Systems Development Architecture & Support facilities：当社のアプリケーション開発体系)を適用しているシステムが中心で，特に，Web系のアプリケーションでの適用を図っている。基準値を設定すれば，他の形式のシステムにも応用できる。

（３）　見積り活動

おおむね下記の手順で見積る。

① 見積りの手順，基準値，同様開発技法やフレームワークを適用した過去のプロジェクト実績を確認する。
② 対象とするアプリケーションの要件/設計情報をもとに，規模(FS)を算出する。
③ プロジェクトの特性(変動要因)を考慮し，上記で算出した規模を調整する(必要な場合)。
④ 要員構成などのプロジェクト条件を特定し，工数，工期を算出する。

⑤　過去プロジェクト情報などをもとに，見積り結果を検証する。

（４）　併用見積り手法

見積り妥当性の評価を行うために，類推法による見積りを併用している。

（５）　体制・役割分担・企業文化

当社では，手法のまとめ・推進を推進部門が実施し，見積りは，各プロジェクトマネージャが実施する。ただし，FS法の社内普及にあたり，専門部隊による，見積り支援を実施している。

当見積り手法適用にあたり，実績プロジェクトデータを分析し基準値を設定すれば，どこでも適用可能である。実績データ収集～フィードバックを回せられればさらに精度は高まっていく。基準値は適用する開発技法，フレームワークにより変わるため，その推進部門などがとりまとめると実施はしやすい。

6.4　精度向上のための活動

（１）　差異分析，フィードバックの方法

プロジェクト実績データを収集し，計測したFS値とプロジェクトの実績値を比較分析する(SLOCとFS，FSあたりの時間など)。その際，実装方式，開発技術などのパターンを整理し，パターンごとに実績データを分析する。

（２）　精度向上のための具体的な方法

専門部隊により，実績収集から基準値のブラッシュアップを実施している。また，実績データの収集には，当社プロフェッショナル人材制度に基づき，プロジェクトマネージャが実績収集する仕組みを導入する。

6.5　実施実績

2004年度から，社内向けに見積りのガイドラインを作成し，適用を開始した。順次適用プロジェクトを拡大中である。

（１）　適用分野(業種，システムプロジェクトの特徴)

SDAS適用の事務処理系アプリケーションに適用。業種，業務の限定はない。

（２）　見積り時期とその精度

実施プロジェクトでは，ほとんど基本設計後の見積りに適用している。

図6.5は，当社のある組織で適用した際のFS見積り規模と実績規模(SLOC)の関係を示したものである。

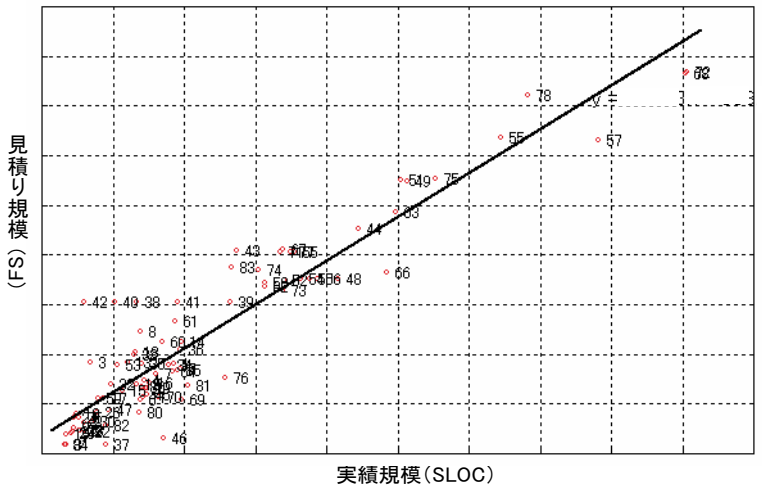


図6.5　見積り規模(FS)と実績規模(SLOC)間の関係例

6.6　当該見積り方法の優位点と課題

（１）　当該見積り手法の優位点

- 30～40分程度の説明で手法を習得可能。
- 手法習得後,1画面５分程度で規模見積り可能。
- 誰が見積っても結果は同じ(単純で計測者のスキルに依存しない)。
- 見積り内容がユーザ・ベンダともに分かりやすい。
- 概算から詳細見積りまで一気通貫の基準で見積りが可能。
- 部分部分の見積りの総和が全体の規模に等しい。開発管理単位での見積り/管理に適用できる。

（２）　不得意な分野等，利用するに当たって留意すべき点

改造見積り方法は継続検討中である。

基準値は適用する開発技法，フレームワークにより変わる。また，ビジネスロジックの複雑度や品質要求レベルなどのプロジェクト特性(変動要因)を考慮する。

第７章　日本ユニシス手法

7.1　取り組みの背景

メインフレームをもとにした従来の開発において，見積りは個々の技術者が独自の経験に基づいて行っていた。このような俗人的な見積りはオープンフレームの世界では通用しないため，当社では，開発規模見積りの基準としてきたファンクションポイント法(IFPUG法)に加えて，工数・期間見積りの客観的基準としてCOCOMO IIを使用して，見積りの妥当性と精度向上を推進している。

○ 複数方式での見積り実施による妥当性向上
　（トップダウン方式(FP・COCOMO II)と従来のボトムアップ方式）
○ 規模見積と工数・期間見積りの分離

7.2　見積り方法(モデル)

見積りは，規模見積りと工数見積りに分け実施している。見積り方法の概要を**図7.1**に示す。

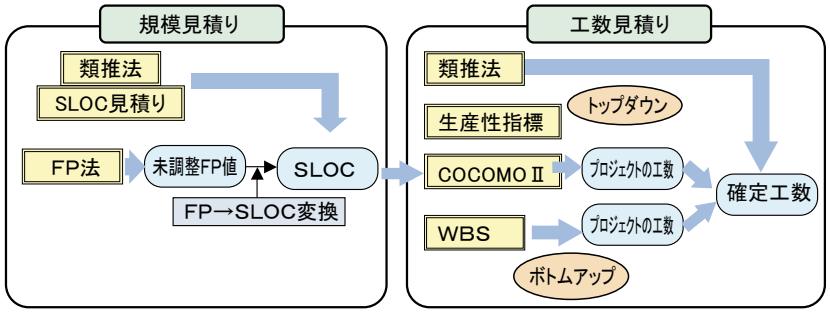


図7.1　見積り方法の概要

（1）　規模見積りについて

顧客要求がファンクションポイント(以下，FPという)法による計測が可能な場合は，FP法により未調整FP値を計測する。FP法による計測が可能でない場合や，バッチ処理，科学技術計算処理，制御系処理などの妥当性のあるFP値が出にくいアプリケーションの場合は，FP値ではなく，SLOCをソフトウェアの規模としている。

未調整FP値あるいは類推したFP値を，FP→SLOC変換表によりSLOCを求める。類推法での規模がSLOCで求まる場合は，その値を使用する。

（2）　工数見積りについて

複数の方法で工数を算出し，比較を行い，最も妥当と判断した値を決定する。

トップダウン見積り
　－FPをSLOCに換算し，COCOMO IIにより工数・期間を算術式で求める。
　－プロジェクト実施組織ごとの生産性指標から工数を求める。
　－類似システムの開発事例を参考に工数を見積る。

ボトムアップ見積り
　－標準WBSまたは類似システムのWBSに基づいて工数を見積る。

7.3　見積り方法の前提条件

（1）　見積り時期と見積り方法

次の時点で見積りを実施している。

- RFP受領時の“概算見積り”。
  見積り範囲は，全工程である。
- 要件が確定した段階(要件定義・論理設計[注1]終了時)の確定見積り
  見積り範囲は，物理設計[注2]以降の全工程である。

（注１）論理設計：「開発プロセス共有化」における「システム設計」工程の「システム設計書(内部)」作成までの工程に相当する。

（注２）物理設計：「開発プロセス共有化」における「システム設計」工程の「システム設計書(内部)」作成以降の工程。

見積り時期と見積り方法の対応関係を**表7.1**に示す。

表7.1　見積り時期と見積り方法の対応

見積り工程/実施時期 / 見積り方法	概算見積り時（RFP入手時）	確定見積り時（要件確定時）
FP法による規模見積り	○	◎
類推法による規模見積り	○	○
COCOMO IIで工数見積り	○	◎
FP/人月で工数見積り	◇	◇
類推法による工数見積り	◎	
WBSボトムアップで工数見積り		◎
類推法による期間見積り	◎	
WBSによる期間見積り	○	◎

◎：必ず実施　　○：実施することが望ましい　　◇：実施は任意

7.4　精度向上のための活動

当社では，工数･期間見積り時にCOCOMO IIを併用している。使用に先立ち2000年度に，3回にわたり見積りワークショップを開催した。ワークショップでは，実際の開発案件を参加者に持ち寄ってもらい，COCOMOIIによる見積りを実施した。それらの結果を解析した結果，規模とパラメータの評価さえ適切であれば，精度の高い見積りができ，実用に十分耐え得るものであることが確認できた。逆に，パラメータの評価が不適切であれば，異常な結果が出てしまうということもわった。

（1）　COCOMO IIモデル式とパラメータ

スケールファクタ(SF：Scale Factors)とは，開発案件に対して編成するプロジェクトの工数が，開発規模に応じて非線型に(この場合は指数的に増減する)影響するファクタである。

工数乗数(EM：Effort　Multiplier)とは，規模に対して線型に影響するファクタである。

おのおののファクタの生産性への影響度合いに応じて，特定の値がそれぞれのレベルごとに設定されている。式の計算においては，評価したレベルに対応する設定値が使われる。

図7.2は，式の基本形を示しているが，定数と非線型と線型の要素から成っていることがわかる。

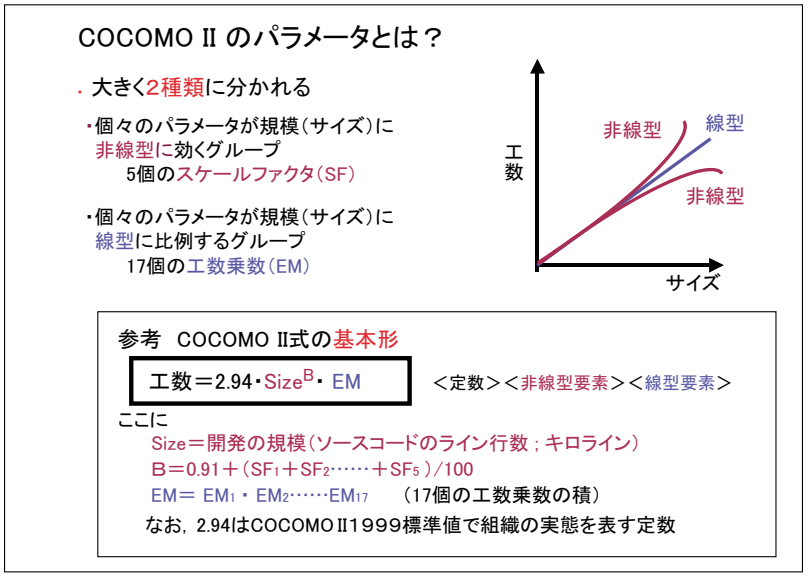


図7.2　COCOMO IIの基本形

（2）　精度向上のための具体的な方策

COCOMO IIのスケールファクタや工数乗数のレベルの判定が，測定者の主観によるところが多く，また，選択したレベルの違いによって，見積り工数がかなり大きく変化する。

例えば，スケールファクタはCOCOMO II標準のスケールファクタ値を採用すると，その取り得る値の範囲は0.91～1.23である。これはSizeの値が100(KSLOC)だとすると66.7～283.4(約4.2倍)と，工数にかなりの違いが出てくる可能性がある。

工数乗数(17個の工数乗数の積)についても同じであり，0.06～115.58とスケールファクタ以上に工数の開きが出る。

５個のスケールファクタ，17個の工数乗数は，それぞれ“Very　Low”から“Extra　High”までの評価レベルの範囲を持ち，それぞれの評価レベルはそのレベルの重みを表す値を持っている。共通の基準でレベル判定を行うために，当社の環境に合わせた表現によるレベル判定早見表を用意している。

（３） 差異分析，フィードバックの方法

当社ビジネスプロセスではプロジェクト完了時に，プロジェクトマネージャは，「プロジェクト完了報告書」を作成し，プロジェクトマネージャの上司によるプロジェクト完了レビューを受けることを義務づけている。プロジェクト完了報告書には，プロジェクトの総括事項を記述するが，その一つとして”見積り工数と実績工数の差異”も記述する。

具体的には，大きな差異がある場合，次の観点から問題点を分析し，フィードバックすべき事項を記述する。

- 活用した過去のデータ
- 見積り手法
- 生産性

より効率的にかつ精度の高い見積り作業を達成するために，先行プロジェクトからの経験のフィードバックが不可欠であり，どのようなアプローチでプロジェクトの見積を行ったかを記述し，さらに工夫した点や提言が重要である。そのために，プロジェクトマネージャにプロジェクト実績を報告してもらっている。

7.5　実施実績

当社では,2000年より開発規模と工数・期間を分離して見積る方法を見積り標準プロセスとして制定しており，大規模プロジェクトにおいては，標準に準拠した見積りが定着してきた段階にある。工数・期間を見積る方法として勘と経験から脱却するために，COCOMO IIのパラメータと実績結果を収集し，分析した結果をパラメータ設定基準表として現場にフィードバックしている。

7.6　COCOMO II使用の優位点と課題

（１） 優位と思われる点

工数見積り手法は，類推法，積み上げ法，パラメトリック法と各種あるが，標準見積り手法として，パラメトリック法の一種であるCOCOMO IIを導入した理由は，次のような優位点を評価したためである。

（ｉ） 工数算出根拠の明確化を図り，納得性を高める。

（ii） 見積りの基礎となる生産性基準を明確化し，生産性向上の施策立案を容易にするために，スケールファクタや工数乗数を活用する。また，パラメータ決定の過程で，プロジェクトの強み/弱みが可視化され，生産性を向上させるべき要素が議論できる。

（iii） 見積り作業のじん速化が図れる。

生産性指標の定まっていない開発言語を使用する場合，ひとつの工数案をじん速に算出できるという効果がある。

（２） 留意すべき点

COCOMO IIでは数多くの入力パラメータを設定する必要がある。

入力パラメータの設定のむずかしさが最大の弱点といえる。パラメータの設定により工数が大きく変わってくる。見積りを行う人による工数のぶれを回避するために，当社の標準的なパラメータ値を示し，各パラメータの設定に際しての社内基準を明確にしている。

例えば，開発の先進性といった場合，「社内で初」，「自部門で初」などと具体的な状況を示してパラメータ設定ができるようにしている。

第 8 章　日本IBM手法

8.1　取り組みの背景

ソフトウェア開発負荷見積りのための「アルゴリズム的見積りモデル」は，1970年代後半から多くが提案され，各種が実用化されてきた。過去のモデルは，負荷がサイズの累乗に比例する形式であるが，用いる指数（累乗，Exponent）の値または範囲が，各モデルにより異なる。

このためサイズの領域によっては，各モデルの間で見積り結果に大きな差が生じる。この乖離は，サイズの累乗に比例する式を用いる限り，どのようにしても避けられない。そのため，広いサイズ領域を取り扱う実ビジネスでの見積りにおいては，個別に色々な工夫をしないと過小見積りや過大見積りの原因となる。

当社も1970年代からソフトウェア製品の開発などにおいて多くの見積り手法を用い，試行錯誤してきたが，特に日本においてシステムインテグレーション事業を1980年代末に開始し，適切な見積りの実施がビジネス上不可欠な状況になってきた。

そこで先に述べた不都合を回避するために，どのような形の式が適切かを研究・考案し，企業内で多数の顧客向けプロジェクトで実際に使用してきた。

8.2　見積り方法

アルゴリズム的見積り式の既存モデルは，実データから導き出されたが，上述のようにサイズの領域により相互に矛盾が生じる。そのため，サイズの開きが 3 桁以上に及ぶ実ビジネスでは，そのままでは適用しにくい。サイズと負荷の関係を両対数をとって表示しても，**図8.1**に示すように比例関係にはできないことがわかる。

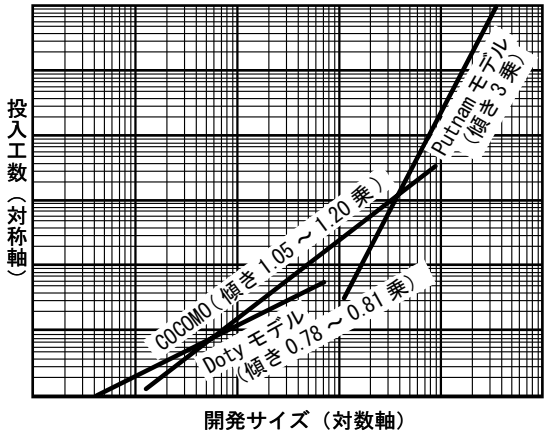


図8.1　Doty/COCOMO/Putnam 3 つのモデルにおける相互の乖離

そこで，当社では，比例関係にないこの関係をさらに両対数にした。その結果，実データの適用においても，完全とはいえないまでも，見積りやすい比例に近い形を取り出し，これを見積りのベースにしようしている。

$$E = a\{10 \wedge (\log S)^b\} \wedge c$$

サイズをS，負荷をEとし，それぞれ 2 度自然対数をとった値をX，Yとして，b, cは定数。

aは調整のための係数（後述），記号∧は累乗を表す。これをグラフで表したのが**図8.2**である。

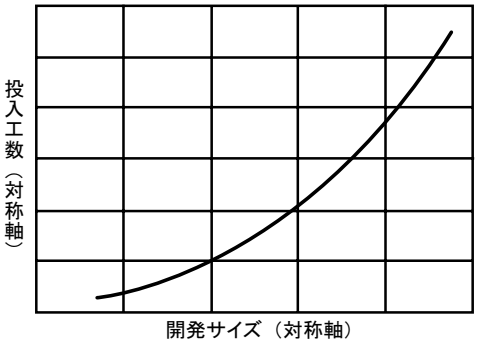


図8.2　汎用見積り式の形状（調整のための係数が入っていない状態）

8.3　見積りツール

当社は，この汎用式を見積りツール内にロジックとして実装している。開発環境や言語，ツールが多様化してきた今日，精度の高い見積りが，常に可能であるのか，あるいは必須なのかという疑問がある。一方，企業のビジネスプロセスやビジネスモデルなどに密着したソフトウェア開発プロジェクトでは，投資対効果が重要である。見積ったコストあるいは開発負荷となるように，作業やアウトプットをコントロールしなければならない側面もある。

そこで，見積りツールの機能としては，なんらかの単位でのサイズを入力し，見積り負荷を得る機能および投入可能な負荷を入力し，制限すべきサイズのパラメータを得るという２つの機能が必要である（**図8.3**）。ただし，後者の適用は困難なこともある。困難なケースの典型は，

- 他システムとの結合の多い場合
- 同一業務の再構築の場合
- 属人的あるいは機能中心の設計の場合
- 概要から詳細という工程の場合

などである。

見積りツールを提供する利点は，見積り値の確かさが増すことだけではない。対象の仕事をその範囲でコントロールしようという動機付けが強く働き，結果としてマネジメントがうまくいきやすくなる点が見逃がせない。

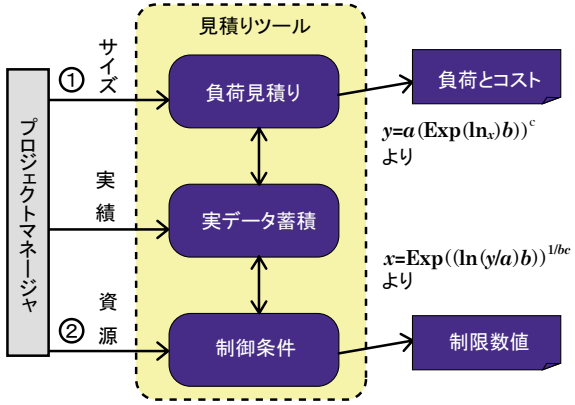


図8.3　見積りツールの機能例

見積りツール（＝汎用式およびその補助ロジック）への入力（図8.3の①）は，当社では，通常データ項目を用いている。これは，システム開発プロジェクトの要件定義という早い段階で分析を行い，ある程度精緻な情報を得られることを重視してのことである。

8.4　体制・役割分担

当社では，各プロジェクトを支える組織として，プロジェクトマネジメントオフィスを組織しているが，この中にはQA（サービス案件の品質保証活動），QI（サービス案件の品質点検活動）がある。また，SEPGはプロジェクトマネジメントのみならず全社的観点から，プロセス改善を推進している（**図8.4**）。

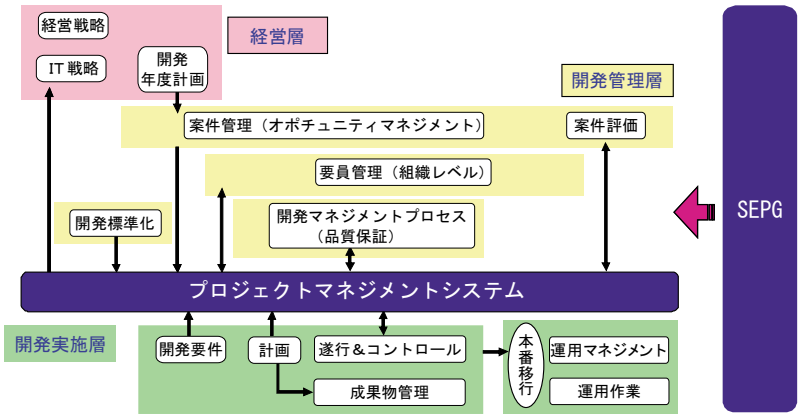


図8.4　日本IBMにおけるプロジェクトマネジメントオフィス

見積りは，サービス案件のQA手順にのっとって精査され，プロジェクト開始の承認のために試用される。なお，当社では見積りを１種類のみではなく，数種（通常，当汎用式による方法や，WBSによる積み上げ，FP，比例配分法などが）使われている。

見積り式を適用するためのデータ取得はSEPGによって行われており，ツール自体にも蓄積データによって補正する機能が備わっており，恒常的な改善が行われている。

8.5　実施実績

当汎用見積り式による見積りは,1991年から過去12年間に少なくとも１万件を超える大小の開発負荷見積りに用いられ，適合してきた。適用分野は業種を問わず，エンタープライズ系の開発全般にわたっている。

8.6　当該見積り方法の優位点と課題

当汎用見積り式の利点は，開発ライフサイクルの早い時期からきちんとした見積りを行える点にある。特にデータ項目と開発負荷の関係は，過去の膨大な実績からもその相関関係が十分に実証済みであり，さまざまな入力項目を必要とするアルゴリズム的見積り式とは一線を画していると考えている。

なお，日本IBMで使用している開発プロセスの１つ，ADSG(アプリケーション開発標準化ガイド)では，要件定義の段階で現行システムのモデル化を行ってから新システムのモデルへと変換を行う。これは，現行物理モデル→現行論理モデル→新論理モデル→新物理モデルという手順で実施されるが，現行モデルの作成時にデータ項目を棚卸し，データディクショナリを作成するため，このような基礎数値を入手することが可能になる。

一方，今後の課題は以下のとおり複数あり，徐々にではあるがこれに対応しているところである。

（１）　開発環境の反映

開発環境，言語，ツールが多様化したため，開発負荷の見積り値を得るには，そのような環境の反映が欠かせない。そのためには，見積りの計算過程(入力)にそれらを反映する方法と，計算結果(出力)に反映する方法がある。ここで示したものは，入力を一定の単位へ換算し，計算結果に対して，環境の影響を係数aで反映する方法である。

普遍化された入力データにするほうが，統計的データを多数得やすい。つまり，係数aを除いた計算結果に対して，事後に環境を反映するほうが，見積り式の安定性が得やすいと思われる。しかし，今後の大きな課題は，開発環境の種類が多すぎる状態をどのようにとり扱うかということだろう。

（２）　データ収集のむずかしさの克服

見積り精度を向上するには，実データの正確な把握・収集が必須である。多数のソフトウェア開発プロジェクトをビジネスとして実施している場合は，これが容易そうにみえる。

しかし，分析に必要なデータが共通の尺度で全てそろうことはまれで，よほど標準化が行き届き，一定のプロセスで開発が進められない限りは，必ずしも容易なことではない。ソフトウェア開発技術が常に進歩していること，手法の自由度が高いこと，人による生産性の差が大きいことなどが，さらにそれをむずかしくしている。

１つの組織体でも困難が伴いがちなデータ収集を，複数企業をまたがって行おうとすると，プロセスの多様性の点でなおさらむずかしい。例えば，サイズの単位，負荷の単位について明確な分類や定義が必要である。負荷の単位でいえば，EVM(アーンドバリューマネジメント)導入や調達・契約方法の変化も今後に向けて進行中であることから，複雑性が増している。

そこで，少なくとも両軸(サイズと負荷)の各物差しの整備と，汎用的な見積り式の共有は必須と考える。企業間で負荷データ自体を共有することは，競合しあう実ビジネス環境では困難を伴うだろう。しかし，例えば，単位を明確にしたサイズの実データを各モデルに入力した結果で得られる負荷と，実際との乖離(分散値など)をつかむアプローチなどは，実用的な見積りモデルを進化させる上で大変意義のあることと考える。

第９章　ジャステック手法

9.1　取り組みの背景

当社は，創業以来，役務提供を前提とした人月契約から脱却し，一括請負型契約を拡大すること，および生産性原理に立脚した能力主義を実践するために，独自の生産管理システムを構築した。

生産管理システムの中心となるのがソフトウェア開発の見積りモデルである。見積りモデルは，開発計画の精度を高め，開発計画達成へのコントロールを可能にし，さらには開発プロセス改善の機会を創出するために必要である。

受託ソフトウェア開発の価格は量と正の相関関係にあることを前提にしている。当社の見積りモデルは，生産物量と生産性を変数とした基本アルゴリズムを適用している。一般に，ソフトウェアは，可視化しにくいといわれるが，決して可視化できないわけではない。当社ではソフトウェア開発工程ごとの生産物を開発量として可視化している。生産物量とは，ソフトウェア開発の各工程で作成する生産物，つまり設計書，プログラムソース，テストデータなどの量である。また，生産性とは生産物量単位，つまり文字数(KC)，ソースコード数(KLOC)，テスト項目数などを作成するために要する時間数，あるいはコストである。生産物を開発量としてとらえるには，アクティビティごとの記述項目，記述水準およびカウント方法などの標準化を図る必要がある。アクティビティとは，工程ごとの生産物をさらに分別し，作業の割り振りを容易にするための作業単位である。記述水準とは，分別した生産物の記述の深さを規定する。基本アルゴリズムは，生産物量見積り方式および生産性見積り方式から成り立っている。また，それぞれの方式において，開発環境の違いや品質要求の多寡による変動を吸収する「環境変数」と呼ぶパラメータを導入している。

9.2　独自の生産管理に基づく見積りモデル

(1)　見積りモデルの基本アルゴリズム

(a)　コスト見積りモデル：当社が考案した生産物量と生産性を変数とする

新規開発の基本アルゴリズムを示す。本アルゴリズムはソフトウェア製品を新規に開発する場合に適用するものである。したがって，運用，移行，教育などのサービスにかかわるコスト見積りは含まない。ある工程iの生産物量をV_i，生産性をP_i，標準生産物量を$V^B{}_i$，生産性のベースラインをP_i^Bで表現すると，コストC_iは次式で求まる。

$$C_i = V_i \times P_i = V_i^B(1+a_i) \times P_i^B(1+b_i)$$

$$a_i : (\Sigma\alpha_{ij}) \quad b_i : (\Sigma\beta_{ij})$$

a_i，b_iは，それぞれV_i^BおよびP_i^Bに対して開発環境の違いや品質要求の多寡による変動を吸収する「環境変数」と呼ぶパラメータである(a_i：生産物量環境変数，b_i：生産性環境変数)。**図9.1**に示すとおり，a_i，b_iは品質特性と環境特性から影響される独立した変動要素(α_{ij}，β_{ij})から構成されている。品質特性は，V_i^BおよびP_i^Bの双方に影響し，環境特性はP_i^Bのみに影響する。本モデルは，ソフトウェア開発で作成する生産物ごとに適用し，プロジェクトのコストは複数の生産物の総和となる。なお，V_iを求めるアルゴリズムを生産物量見積り方式，P_iを求めるアルゴリズムを生産性見積り方式と呼ぶ。

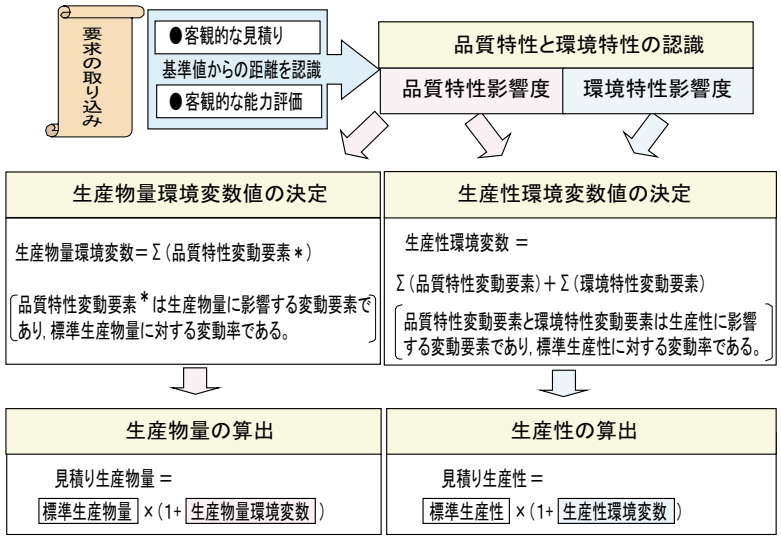


図9.1　生産物量と生産性の算出式

(b)　生産物量見積り方式：ある工程iの生産物量V_iの求め方を**図9. 2**に示す。図において，機能要求書，基本設計書およびパッケージ設計書は，それぞれ要件定義工程，システム設計工程およびソフトウェア設計工程の出力である。よく規格化された(記述項目，水準，書式，表現方法などが定められていること)基本設計書の量は，ソフトウェア要求の量を測る尺度としてよく用いられる「ファンクションポイント」との相関が高いことが経験的にわかっている。また，各工程の生産物量の間にも高い相関関係があることに着目し，上位工程で作成する生産物量から下位工程の生産物の量を算出するアルゴリズムを考案した。

$$V_i^B = V_i^B - 1 \times H_i$$

H_i：生産物量変換係数

生産物量の変動パラメータである生産物量の環境変数をa_iとして表現すると，生産物量は次式で求まる。

$$V_i = V_i^B \times (1 + a_i)$$

a_i：$(\Sigma\alpha_{ij})$

任意の生産物は，複数の記述項目から構成され，その中には要求の量，または実装結果であるソースコードなどの量との相関が高いもの，低いも

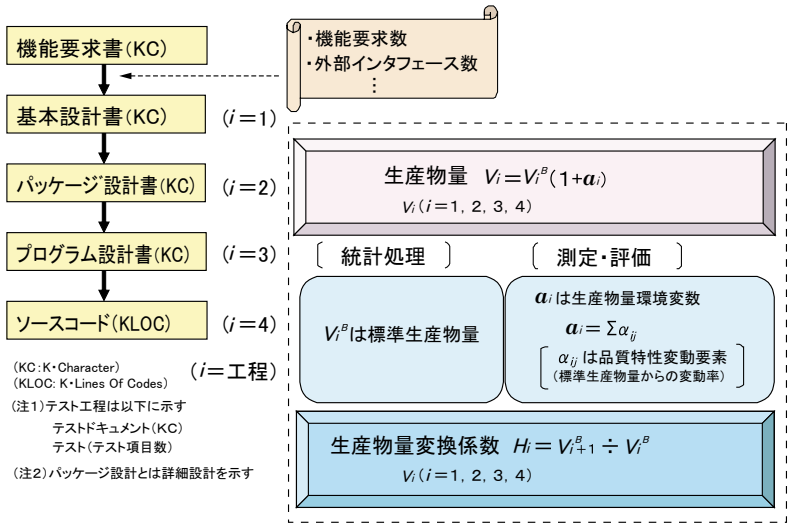


図9. 2　生産物量見積り方式のアルゴリズム

のが混在している。本方式では，前者を標準生産物量(V_i^B)と呼ぶ。後者は，**表9. 1**に示す品質特性変動要素評価表に基づき生産物量の環境変数a_iを算出して，V_i^Bを調整する。品質特性は日本工業規格(JIS X 0129)に基づき品質特性および副品質特性を識別し，尺度および数量化方法を定めており，顧客の要求水準を確認して決定する。

生産物量に影響する品質特性は４個あり，さらに品質特性を細分化した副品質特性が13個ある。

表9. 1の機能性(正確性)を例にとると，変動要素は「正確性(検証)にかかわる標準テスト密度を基準にしたテスト項目への要求水準」をあげている。「テスト密度」とは，ソースコード単位量当たりのテスト項目数である。この例は，一般に要求される「テスト密度」の平均との乖離によって，機能性のうちの正確性に対する要求水準を測り，テスト工程の生産物量(テスト項目数)を調整するものである。

(c)　生産性見積り方式：モデルを単純にするために個人の能力差による生産性の相違を吸収して，生産性を生産物の単位量当たりのコストとした。生産性の変動パラメータである生産性の環境変数をb_iとして表現すると，生産性は次式で求まる。

$$P_i = P_i^B \times (1 + b_i)$$

b_i：$(\Sigma\beta_{ij})$

生産性のベースラインP_i^Bは，生産性の環境変数b_iによる変動を除去した後の実績生産性に基づく平均値を基準にしている。

次に，開発ベンダの立場から，環境変数の外部環境変数および内部環境変数について述べる。

外部環境変数は「顧客の特性による変動」および「プロジェクトの特性による変動」があり，ソフトウェアの価格に反映する。

内部環境変数は「開発ベンダの能力による変動」であり，ソフトウェアの価格に反映しないが，プロジェクトの実行計画には反映しなければならない。

生産性の環境変数の具体例を，**表9. 2**(品質特性変動要素評価表)および**表9. 3**(環境特性変動要素評価表)に示す。生産性に影響する環境特性は７個あり，「業務特性」，「ハードウェア特性」，「ソフトウェア特性」，「コミュニケーション特性」，「環境変数」，「工程入力特性」および「改造・再構築特性」である。さらに，環境特性を細分化した副環境特性は16個あり，それぞれに外部環境変数および

表9.1　品質特性変動要素評価表(規模「生産物」に影響する外部環境変数)

品質特性	副品質特性	変動要素	ベースラインからの変動率(%)			
			要件定義	設計	製作	テスト
機能性	合目的性	利用者/利害関係者の広がり，コンテンジェンシー対応，不正移行データ対応などの該当事象数	0〜50	0〜50	0〜50	0〜50
	正確性	正確性(検証)にかかわる標準テスト密度を基準にしたテスト項目量への要求水準	―	―	0〜20	0〜50
	接続性	他システムとの接続によるコード変換，フォーマット変換数	0〜5	0〜20	0〜20	0〜20
	セキュリティ	対応が必要なセキュリティ実現機能数，ただし，機能要件に定義されている部分は除く	0〜20	0〜20	0〜20	0〜20
信頼性	成熟性	故障低減に必要な実現機能数	0〜 5	0〜10	0〜10	0〜10
	障害許容性	異常検知に必要な機能数	0〜5	0〜10	0〜10	0〜10
	回復性	再開処理に必要な実現機能数	0〜5	0〜10	0〜10	0〜10
使用性	理解性	理解性向上(機能など)のためのプレゼンツールなどの作成対象数	―	0〜10	―	―
	習得性	習得性向上(使い方など)のためのマニュアルなどの作成対象数	―	0〜10	―	―
	操作性	操作性向上(心理的/肉体的配慮，運用やインストール容易性など)のための実現機能数	0〜10	0〜20	0〜20	0〜20
保守性	解析性	解析に必要な実現機能数	―	0〜10	0〜10	0〜10
	変更作業性	作成する保守用ドキュメントの数	―	0〜10	―	―
	試験性	試験に必要な機能数	―	0〜10	0〜10	0〜10

(注) 移行，教育，保守，運用作業は適用対象から除いている。

表9.2　品質特性変動要素評価表(生産性に影響する外部環境変数)

品質特性	副品質特性	変動要素	ベースラインからの変動率(%)			
			要件定義	設計	製作	テスト
機能性	合目的性(要求仕様の網羅性)	要求の記述水準および網羅性。要件定義については新規性，方針明確性，ステークフォルダの多様性などを考慮	0〜100	0 〜30	―	0 〜10
	正確性	正確性(検証)にかかわる標準レビュー工数(各工程10%)を基準にした要求水準	0 〜 5	0 〜 5	0 〜 3	0 〜 5
	接続性	基準単位(100kS)に対する社内/社外システムとのインタフェース先の数	-10 〜 10	-5 〜 10	―	-5 〜 5
	整合性	整合をとる社内/社外の規格・基準の数，全体適合性やグローバル化対応も含む	-10 〜 10	-5 〜 10	-3 〜 5	-3 〜 5
効率性	実行効率性	実行効率に対する一般的要求水準(既知)の最適事例を基準にした要求水準	0 〜 5	0 〜10	0 〜 5	0 〜10
	資源効率性	資源効率に対する一般的要求水準(既知)の最適事例を基準にした要求水準	0 〜 5	0 〜10	0 〜 5	0 〜10
保守性	解析性	ソースコードの解析性をコード化規約に定めるコメントに対する要求水準により評価	―	―	0 〜 5	―
	安定性	ソフトウェア変更に対しシステム品質維持可能とする水準をライフサイクル目標年数の長さにより評価	0 〜 5	-3 〜 7	-3 〜 5	―
移植性	環境適用性	多様なハード，ソフト，運用環境に適用させる要求の水準	0 〜 7	0 〜 7	0 〜 3	0 〜 5
	移植作業性	環境を移す際に必要な労力を低減させる要求の水準	0 〜 7	0 〜 7	0 〜 3	0 〜 5
	規格準拠性	移植性に関する国際/国内規格または規約を遵守する要求の水準	0 〜 7	0 〜 7	0 〜 3	0 〜 5
	置換性	使用環境/条件を変更せずに他のソフトウェア製品と置き換えて使用可能とする要求の水準	0 〜 7	0 〜 7	0 〜 3	0 〜 5

(注) 移行，教育，保守，運用作業は適用対象から除いている。

表9.３　環境特性変動要素評価表（生産性に影響する外部環境変数）

環境特性	副環境特性	変動要素	ベースラインからの変動率（%）			
			要件定義	設計	製作	テスト
業務特性	業務ナレッジ	顧客の開発対象業務に対する業務ナレッジが生産性に及ぼす影響	-10～50	-10～10	―	-10～10
ハードウェア特性	安定度/信頼度/使用度	システムもしくは製品となるハードウェアの安定度・信頼度	―	-5～5	―	-5～5
ソフトウェア特性	安定度/信頼度/使用度	システム/製品となる他社作成ソフトウェアもしくはCOTSの安定度・信頼度	―	-5～5	―	-5～5
コミニュケーション特性	顧客窓口特性	意思決定能力（期限遵守，決定事項のくつがえる度合）	-10～20	-10～10	―	-7～7
	工期の厳しさ	基準工期（月）＝2.7×（人月）$^{1/3}$に対し▲30%限度とした短期化度合	0～10	0～10	0～5	0～7
	コミニュケーション基盤	開発拠点分散，資料等情報共有，電子媒体・システム具備など物理的基盤充実度	-10～10	-5～5	-3～3	-3～3
	レビュー体制	むだなレビュー（重複多段階など）の排除およびレビュー効率向上への工夫度合	-5～5	-5～5	-3～5	-5～5
開発環境特性	開発手法/開発環境	開発手法・環境（ソフト/ハード/ツール）の信頼性，占有率などを考慮した使用実績	-3～3	-3～3	-5～5	-5～5
	テスト手順書水準	テスト手順の具体化度（操作手順＆入出力の具体化の要求水準）	―	―	-3～3	-3～3
工程入力情報特性	業務関連資料	必要資料の具備状況（正確性，信頼性を含む）および使いやすさ（検索性，理解性）	-10～10	-7～7	-3～3	-3～3
	他システム関連資料	必要資料の具備状況（正確性，信頼性を含む）および使いやすさ（検索性，理解性）	-10～10			
	規約・標準化関連資料	必要資料の具備状況（正確性，信頼性を含む）および使いやすさ（検索性，理解性）	-7～7			
改造・再構築特性	既存システムの練度	改造対象母体または再構築する元のシステムに関する顧客の熟練度	-50～50	-40～40	-10～10	-40～40
	既存テスト環境流用水準	既存のテストリソースの流用度合	―	―	-20～0	-30～0
	母体調査ツールの水準	調査ツール機能（絞込み，モニタリング，リバースなど）	―	-20～15	-10～5	-20～10
	既存母体の品質	正確性（潜在バグなど），解析性，環境適用性などの事象数	0～50	0～50	0～50	0～50

（注）　COTS：Commercial Off The Shelf
移行，教育，保守，運用作業は適用対象から除いている。
改造・再構築特性は既存ソフトウェアに対して改修（追加，削除）を施す開発形態を指す。

内部環境変数に区分している。「コミュニケーション特性」を例にすると，その配下には「顧客窓口特性」，「工期の厳しさ」，「コミュニケーション基盤」および「レビュー体制」の４個の副環境特性を定めている。

次に，生産性に影響する品質特性は４個あり，さらに品質特性を細分化した副品質特性が12個ある。

生産性の環境変数は，品質特性と同様に尺度および数量化方法を定めており，顧客の特性水準およびプロジェクトの特性水準を確認して設定する。この過程で，見積り対象のプロジェクトが抱えるリスクが明らかになり，顧客とリスクを共有し，相互協力による生産性改善対策を講じることで効果を挙げている。生産物は，複数の記述項目から構成されているので，生産性も記述項目に対応させ保持しており，生産性の環境変数は各記述項目の生産性に対して作用させている。

なお，各工程での外部環境変数とベースラインからの変動幅が，±30%以内になることを経験値から得ている。ただし，内部環境変数の混入除去，組織標準の精錬および運用の徹底を図ったうえでの値である。

（２）　改造型開発の見積りモデル

本モデルは，新規開発の見積りモデルに比べ，成熟度の点から検討すべき課題を含む。当社は，今後とも研究と実証を通して本モデルの精錬を図って行く。ここでは，現時点での改造型開発の見積りモデルについて述べる。

改造型開発と新規開発との主要な相違点は三つある。

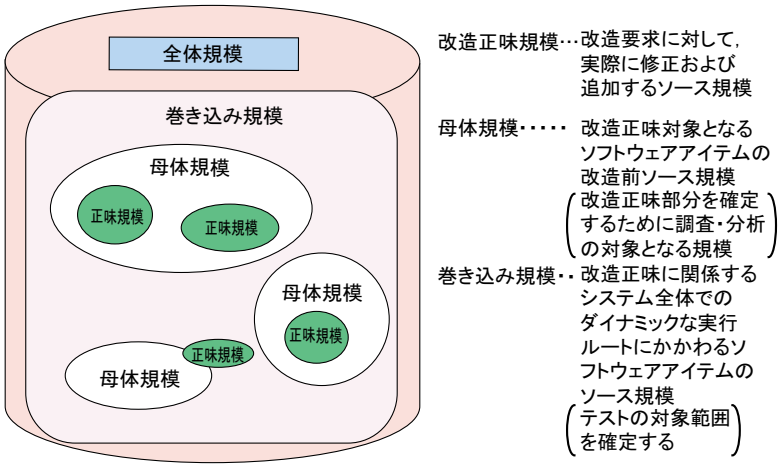
一つ目は，改造型開発では要求を実現する手段が，改造対象システムの内部構造に制約されるため，**図9.３**に示すように改造要求による影響範囲を識別し「改造正味規模」，「母体規模」および「巻き込み規模」を特定しなければならない。

二つ目は，新規開発では，次工程への指示書と製品の設計書（製品仕様書）とが同一であるが，改造型開発では改造した結果の製品仕様書とは別に，改造事項を特定する指示書（改造設計書）を作成する。改造設計書の生産物量は，「改造正味規模」，「母体規模」および「巻き込み規模」をパラメータとして見積ることとした。設計作業の負荷と相関が高いのは，改造設計書なので，改造設計書を設計の生産物として，その量を見積る。なお，製品仕様書は別の生産物として扱う。

三つ目は，改造対象システムの調査が必要であり，場合によっては改造対象システムの設計書をソースコードなどから復元する場合がある。「改造対象シ

ステムの調査に対応する生産物量には，調査対象の生産物量を割当て，改造対象システムの設計書の復元作業に対応しては，復元仕様書を生産物として追加した。

改造型開発における見積りモデルの環境変数は，新規開発の見積りモデルの環境変数を併用するとともに，表9.3(環境特性変動要素評価表「改造・再構築特性」)に示すとおり，改造母体に対する練度など，改造型特有の環境変数を追加している。



（注）ソフトウェアアイテムはプログラム(コンパイル単位)を指す

図9.3　改造型開発の規模構造

（３）　仕様変更見積りモデル

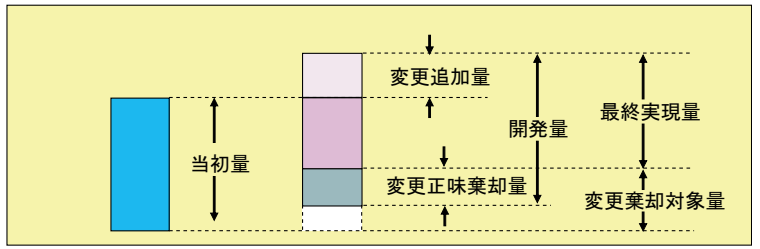
当社では，契約にあたり，発注者(顧客)の決断を必要とする項目，および発注者との相互合意を必要とする項目を提示している。特に，仕様変更見積りにかかわる発注者の決断項目には，仕様変更に伴う量(追加量，棄却対象量など)と変更タイミングがある。相互合意の項目には，仕様変更の認定基準と仕様変更に伴う量(正味棄却量)がある。

次に，「仕様変更量」とは何か，について述べる。**図9.4**に「仕様変更量」の構造と算出式を示す。

ここで，「当初量」とは，仕様変更がまったくなかった場合の見積り量である。「変更棄却対象量」とは，仕様変更によって開発不要となった部分の見積り量で，

「変更正味棄却量」とは，仕様変更により途中まで開発したが開発不要となり，実際に捨てるであろう量である。また，「変更追加量」とは，仕様変更により追加となるであろう量である。開発完了時に最終的に納品される量が，「最終実現量」であり，開発コストに比例するのが，「開発量」である。

注目すべきは，「変更正味棄却量」である。この量は，仕様変更の発生するタイミングによって，同じ変更でも大きく変動する。つまり，システム設計時点で発生すれば，変更により不要となり棄却するのは，基本設計書の該当部分だけであるが，システムテスト時点で発生すれば，システム設計からシステムテストまでの全ての工程の生産物を棄却することとなる。



・開発量＝当初量－変更棄却対象量＋変更追加量＋変更正味棄却量
（最終実現規模）

・変更棄却対象量＝当初量×変更棄却対象率
・変更追加量＝当初量×変更追加率
・変更正味棄却量＝当初量×変更棄却対象率×完成率

図9.4　仕様変更量の構造と算出式

9.3　見積り方法の前提条件

（１）　開発形態による見積りモデルの適用方法

当社の見積りモデルは，ウォータフォールライフサイクルモデルの適用を前提としている。スパイラルモデルには，開発サイクルごとに本モデルを適用し，インクリメンタルモデルには，インクリメンタルの段階ごとに本モデルを適用する。また，既存システムの保守作業は，個々の保守案件ごとに本モデルを適用している。規模の小さい案件では，各工程で作成する生産物量をソースコードの量に代替することで，簡易化を図っている。

（２）　見積り時点による適用方法

本モデルでは，プロジェクトの途中で順次完成する生産物量に基づき，簡単に生産物量の再見積りができる。また，ソースコード量と基本設計書の量との関係式も導出でき，ソースコード量から基本設計書の量を推定する場合に利用できる。システム設計が完了していない場合には，次の方法のいずれかを選択して，まず基本設計書の生産物量を見積る。

①　類似のシステムや再構築対象の既存のシステムが存在する場合など，確からしいソースコード量を計測できる場合は，ソースコード量から基本設計書の量を見積る。

②　ユーザの要求を確認して，システムへの入力，出力，機能を洗い出し，これを根拠に基本設計書の量を求める。この場合の精度は，洗い出しに投下する作業に依存する。

③　通常は①および②を併用して，見積りの精度を高めている。

ただし，システム設計が完了するまでは，見積り精度の低下を防ぐことはできないので，「システム化計画段階」，「要件定義中」，「要件定義完了時点」，「システム設計完了時点」と数段階の見積りを定めている。実務上では，顧客との合意に基づき段階を決定し，契約を取り交わしている。

（３）　体制・役割分担・企業文化

当社は創業以来ソフトウェア開発部門を製造部（製造本部）という組織名にしている。すでに，製造業をお手本にしていることを述べた。

独自の生産管理システムの実施体制ついては，**図9.5**に示す。

ここでは見積りモデルに関係する各部門の体制と役割を中心に述べる。

当社の見積りモデルには，顧客向けの外部見積り（標準見積り）と社内向けの内部見積り（標準開発計画）がある。見積りの中心となるのが標準開発計画である。

標準開発計画は，顧客のRFP（提案依頼書）に基づき，営業部門と製造部門とが共同でサーベイ作業を行い，作成している。標準開発計画は環境変数を配慮した量，生産性などを基準に見積っている。標準開発計画は，製造部門の達成責任になる。

標準見積りは，顧客と当社の共通語とするために，標準開発計画の量，生産性を継承することが前提にある。それゆえ，営業部門の責任は，価格折衝責任を除くと，継承責任が重要になる。継承責任とは量，生産性とともに，変動パラメータの環境変数を顧客に公開し，相互共有と協力関係を構築することである。

標準開発計画とその実績はプロジェクトのプロセス資産としてデータベースに蓄積している。製造部門では，プロジェクトチームのほかに，SEPG（ソフトウェアプロセス改善グループ）を設置し，製造部門に共通するプロセスの改善およびプロセス資産の活用促進などを図っている。

次に，製造部門をけん制し支援する品質保証関連の組織について述べる。一つは検査課で，成果物の検査を行う専門組織である。検査課は，成果物の検査以外に，プロジェクトの計画と実績との乖離を統計分析し，製造部門での是正処置の動機づけを促している。二つ目の組織は品質保証委員会である。品質保証委員会は，品質マニュアルなどの組織標準およびプロセス資産を管理推進すること，ならびにプロセスの監査を行う専門組織である。特に，見積りの基礎となる生産性のベースラインは本組織で統計分析し，製造部門にフィードバックしている。

最後に，各部門の見積りに関連するプロセス改善目標は，経営者のマネジメントレビューに基づき予算化し，達成度合いを監視している。

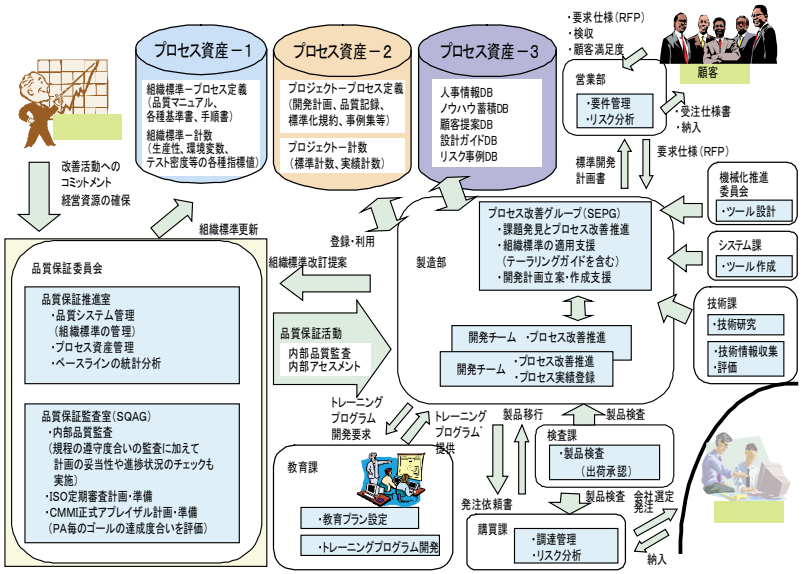


図9.5　生産管理システムの実施体制

9.4　プロジェクト実績の蓄積と活用

（１）　プロジェクトマネジメントでの活用

ソフトウェア開発を成功させるためには，開発途中の生産物量および生産性を日々監視し，それが見積りと乖離しないようコントロールする必要がある。当社には，個々の技術者が日々の生産物量と，それに要した工数を入力し，リアルタイムに出来高を把握する仕組みがある。

図9.6に見積りと生産管理関連図を示す。当社では，生産実績の分析手法を定め，具体的なプロセス改善を図っている。事例として，開発量の予実差異分析は，生産物量環境変数と仕様変更量の変動量に分解し，生産性の予実差異分析は生産性環境変数の変動値に分解する。この分析によって，予実差異の根本原因を可視化している。さらに，仕様変更量，環境変数の中で，変動が懸念される項目を開発リスク項目として管理し，その顕在化の監視手順，防止方法および軽減化手順をリスク対策事項として定め，開発計画に組み込んでいる。

（２）　技術者個人の生産性と品質の把握

技術者個人の生産性と品質の把握は，能力主義を実践するうえで必要である

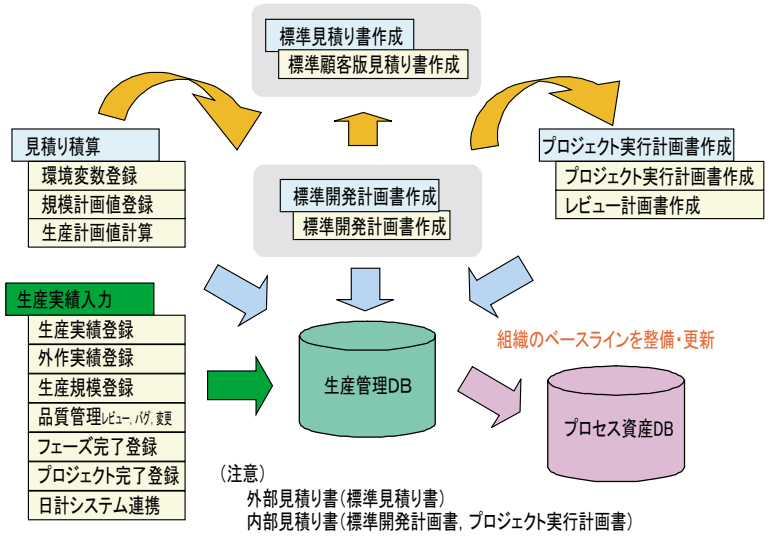


図9.6　見積りと生産管理の関連図

ことをすでに述べた。個人の生産性と品質を把握するには，生産実績の入力機能が必要である。さらに，当社の生産管理システムは，プロジェクト固有の環境変数を可視化するとともに，被指示者，指示者といった開発体制面の相互作用を吸収するように配慮している。

9.5　実施実績

当社では全てのプロジェクトについて本見積りモデルを適用している。ただし，すでに記載したが開発形態により見積りモデルの適用方法を決めている。

平成15年度の開発形態（プロジェクト類型）ごとの適用システム案件数を**表9.4**に示す。

表9.4　開発形態ごとの適用システム案件数

プロジェクト類型	システム案件数	適用率	備　考
開発（通常）	132	100％	ウォータフォール型
開発（小規模）	18	100％	期間３ヶ月以内＆工数1 000時間以下
保守（通常）	61	100％	工数1 000時間を超す
保守（小規模）	34	100％	工数1 000時間以下（ソースコード代替）
付帯サービス	35	０％	教育，障害対応，移行など
計	280		

9.6　当該見積りモデルの優位点と課題

（１）　優位点

見積りモデルの二つの優位性について述べる。

一つは，生産管理システムの導入効果を高めるための見積りの優位性である。一般には，見積りモデルと生産管理システムとは連携していない場合が多い。当社の見積りは開発計画の量，生産性，品質，納期などと整合する。見積りは，プロジェクトチームおよび個人の目標を明示し，生産物量による出来高などの監視指標に基づき，プロジェクトをコントロールし，目標の達成を可能にしている。さらに，生産管理システムは，生産性や品質などの見積りのベースラインを設定するのに，プロセス資産の蓄積・分析を通して，その一翼を担っている。

二つ目は，生産性向上に関する見積りの優位性である。生産性向上とは，顧客との相互協力による生産性向上および技術者個人の生産性向上を示す。見積

りは顧客と当社との共通語である。例えば，見積り段階から量および生産性の変動パラメータ(外部環境変数)を開示して，プロセスの改善点を客観的に共有する。プロセス改善効果は，生産性向上度合いを定量化することで，相互に享受可能とした。次に，技術者個人の生産性向上である。当社は能力主義を実践している。そのために，評価尺度のひとつに個人の生産性と品質がある。見積りモデルは，プロジェクト固有の変動パラメータを可視化し，客観的な個人の生産性と品質を捉えることを可能としている。

(2)　現在の課題と今後の取り組み

(a)　システム化計画工程・要件定義工程での見積り精度の向上　当社の見積りモデルも，他のモデル(ファンクションポイント法など)でも，システム化計画工程・要件定義工程と以降の開発工程の見積り根拠を可視化し，精度を向上するという課題がある。当社の課題への取り組みは，本稿の生産管理に基づく見積りモデルを基軸に行う。現状の見積りモデルは，システム化計画工程・要件定義工程でのアクティビティ，生産物(記述項目，水準，書式，表現方法)，環境変数などが，十分とはいえない。今後，顧客の協力を得て，継続的に取り組んでいきたい。

(b)　簡易版見積りモデルの開発および普及　本稿での見積りモデルは，詳細に積算するため,一般企業で初期導入する場合の障壁が高いと認識している。また，顧客視点からの意見(外部環境変数項目の追加など)もある。そのようなことを配慮して，顧客企業や開発ベンダなどで広く利用可能な，簡易版の見積りモデルの開発を検討している。

SECでの見積り手法に関する実証実験

第2部では，先導的な取り組みを行っている各社の事例について示しましたが，ここでは，先進的な手法に関してSECが共同研究の一環として実証実験した手法について解説を行います。

ーCoBRA法（ドイツフラウンホーファ協会IESE(注)）
　Cost Estimation,Benchmarking,and Risk Assessment法。経験豊富なプロジェクトマネージャや見積り専門家の知識の形式知化手法
ーEASE協調フィルタリング法（奈良先端科学技術大学院大学）
　過去の実績データから類似システムを探索し，類似システムのデータ集合から見積りを行う手法

（注）http：//www.iese.fhg.de/

第 1 章　CoBRA法

1.1　手法の背景

組織での見積りモデルの構築は，過去のプロジェクトデータの分析に基づいて，モデルを構築することが一般的である。しかし，ソフトウェア開発の組織において過去のプロジェクトデータをほとんど集約していない例も多く，見積りモデルを構築するためには，データの集約から始めなくてはならない。データがなかったり，分析されていない場合は，結果として個別のプロジェクトごとに，プロジェクトマネージャが，過去の経験や勘を駆使して見積ることになり，組織的に統一のとれていない，再現性のない見積りが，繰り返されることになりがちである。

組織的に統一された見積りモデルを確立したい一方，プロジェクトデータを統計的な分析に耐えるほど用意できないジレンマの中で，根拠のあるなんらかの見積り方法を確立できないかというニーズには高いものがある。

CoBRA法（Cost Estimation，Benchmarking，and Risk Assessment）は，この問題に対して，ひとつの解決策を与えるものである。この方法の特徴は，少数の過去プロジェクトデータと経験豊富なプロジェクトマネージャの知識を組み合わせて，見積りモデルを作る点にある。この考え方の背景には，特定の組織内で経験豊富な人間の感覚が正確であるとの経験則があり，その感覚や考え方をモデル化して共有しようとするものである。

なお，CoBRA法は，見積り手法というよりは，個別の組織で見積り手法（モデル）を構築するための手法である。

1.2　見積りモデル構築方法

（1）　見積りモデル構築にあたっての考え方

例えば，工数（Effort）を見積る場合，CoBRA法は，「理想」のプロジェクトでは，工数は規模（Size）に正比例するが，「現実」ではさまざまな工数増加要因により，「理想」に比べ大幅な工数増加（コストオーバヘッド：CO）が発生すると

考えられる。式では次のとおり表される。

Effort＝α×Size×（１＋CO）

この考え方のイメージを**図1.1**に示す。図の自動車で目的地に行く場合の時間予測は，実際には行くまでに，さまざまな要因（渋滞，事故など）により速度が増減し，結果的に到着するまでの時間に，ぶれが生じることを示している。CoBRA法では，プロジェクトで理想な場合（最もスムーズにプロジェクトが進む場合）に対して，さまざまな要因により，工数が増加（上記の式のCO）していくとして，その増加部分について，経験豊富なプロジェクトマネージャなどの専門家の知識を利用して定量化する。増加分は，パーセンテージで求めるので，使用する規模の単位に制約はない（FPでも行数でもよい）。

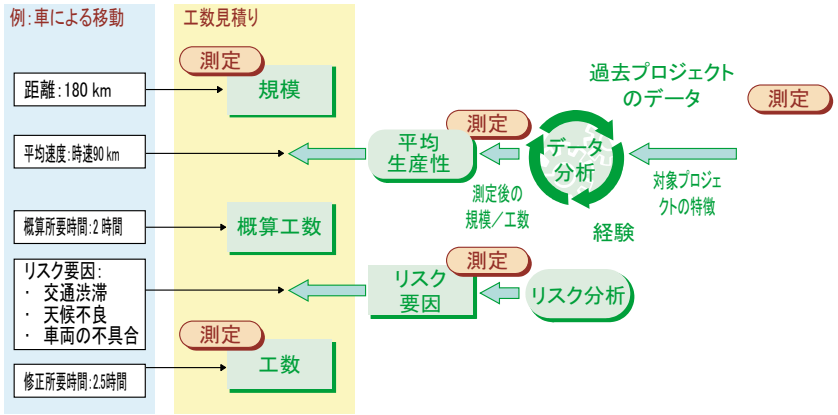


図1.1　CoBRA法の考え方

（２）　要因関係図の作成

最初に工数増加の要因となるものをブレーンストーミングにより決定する。何が工数に影響するかを洗い出すために，複数名のプロジェクトマネージャが集まり，何が重要な工数要因かを議論して決定する。具体的には，**図1.2**のような要因関係図を作成する。通常，10～20程度の要因が決定される。議論に先立って，コーディネータ（1.3節を参照）があらかじめ可能性のある要因についてアンケートで集約しておくと効率がよい。なお，設定にあたっては，似た要因を複数設定することは，避けなければならない。同じような要因を設定すると，増加分をダブルカウントすることになってしまうためである。

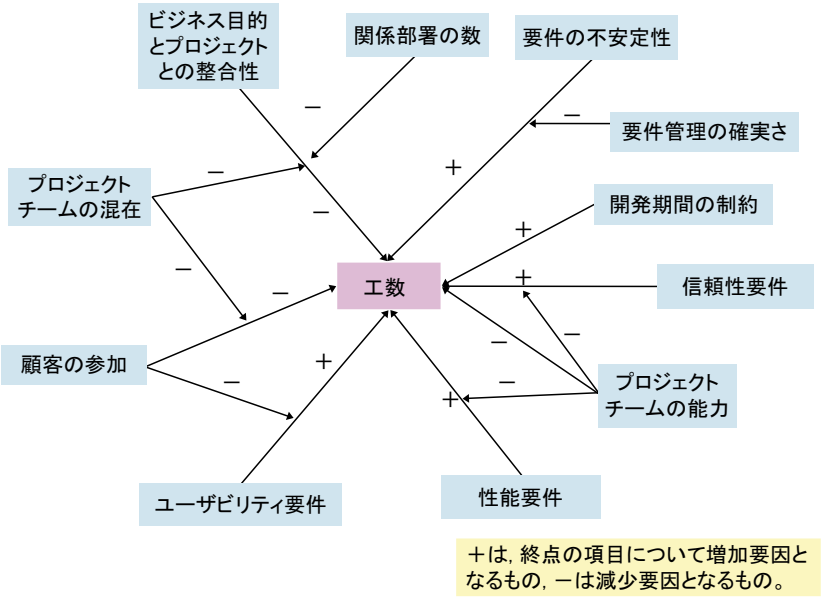


図1.2　要因関係図（例）

（３）　要因ごとの影響度の設定

続いて，各要因が，どの程度の影響を及ぼすのかを決定するために，プロジェクトマネージャに一人ずつインタビューを行い，「通常」「最善」「最悪」の３つの場合に分けて，各要因の影響度合いを確認する。３つの値を聞くのは，感覚にぶれがあることを反映している。

設定の仕方は，次のとおりである。例えば，工数増加要因として「要求の安定性」があったとき，プロジェクトマネージャに対して「最も」不安定な場合（最悪の場合）に，工数増加が，どの程度あるかをたずねる。あるプロジェクトマネージャが，「通常だと30%増し，最善だと20%，最悪だと70%」と回答した場合は，**図1.3**に示すような三角分布が得られることになる。同様に，他の要因についても３つの値を確認し，要因ごとの三角分布を得る。さらに，複数のプロジェクトマネージャ（10名程度）に対して，これを繰り返す。

これら影響度は，感覚に基づいた回答なので，プロジェクトマネージャによって違う三角分布が得られる。基本的には，すべての分布を採用するが，明らか

に大きな違いがある場合は，当該分布を回答したプロジェクトマネージャに再確認する。

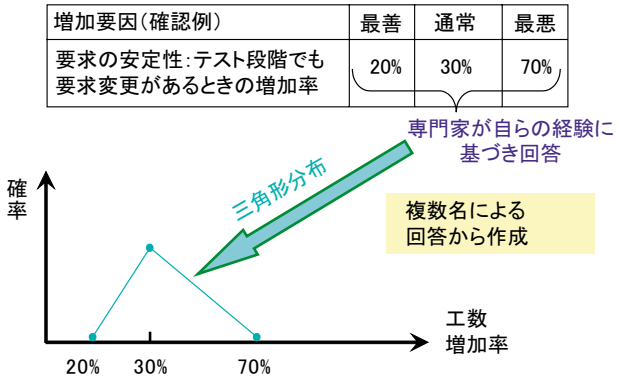


図1.3　要因の影響度の確認例

（4）　個別のプロジェクトでのコストオーバヘッド(CO)の求め方

上記で設定したものは，各要因が最悪の場合(最も増加が多い場合)の値である。図1.3では「要求の安定性」という工数増加要因の値が「20%」「30%」「70%」となっているが，要因そのものの影響度がそれほど大きくない場合，これら3つの値も，それに応じて小さくなるはずである。そこで個別のプロジェクトでは，各要因がどの程度のものかを4段階(0～3)で評価し，段階に応じて値を割り当てる。具体的には,0の場合は影響度なし(増加要因とならない)として，3の場合に最悪の値とし,1，2の場合は比例配分する(2)。**図1.4**に，影響度合いの評価例と，その影響度の計算方法を示す。この場合，「顧客の参加度」は，「あまりない」と予想されるので，その増加率は，最悪46.7%との計算になる(つまり，「13.3%」「20%」「46.7%」の三角分布になる。仮に「ほぼいつも」と予想される場合は，増加率は0,すなわち増加要因とならない。

このようにして，すべての要因について影響度(三角分布)を設定したのち，モンテカルロ法により，工数の増加分の確率分布を求める(**図1.5**参照)。この分布から，増加分の最低値と最大値，平均値がわかる。

(2)　このあたりは，CoBRA法ではプロジェクトマネージャらの感覚に基づいて見積りモデルを作ることから，複雑な計算に陥ることないよう，シンプルさを追求している。

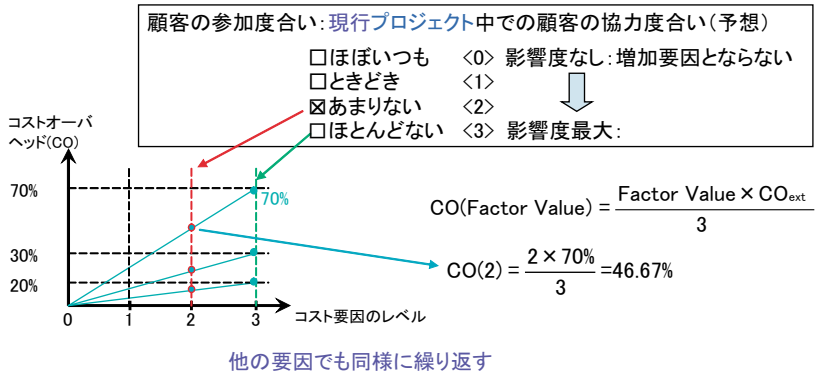


図1.4　個別の要因の評価例

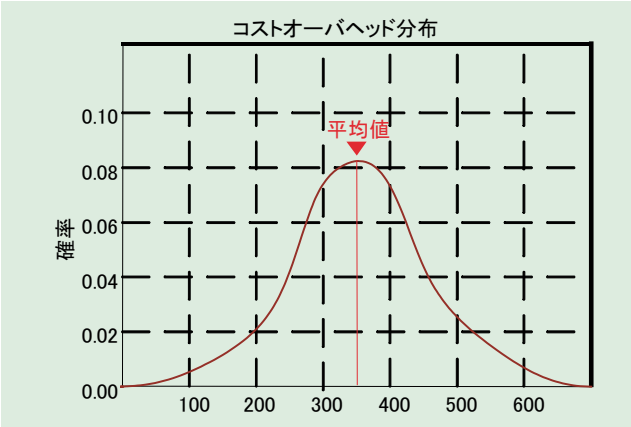


図1.5　モンテカルロ法による増加分の分布

（5）　理想の場合の係数の設定

最初の(1)で示した式の中で，αは，理想の場合の比例係数(規模と工数の場合は，生産性)であり，CoBRA法における見積りのためのベースラインである。αは,10件程度の過去プロジェクトデータの規模，工数とコストオーバヘッド(CO)から求める(**図1.6**参照)。COは，(3)で設定した要因すべてについて各プロジェクトでの段階(0～3)を評価し求める。

（6）　個別のプロジェクトでの見積り

以上に示したとおり，見積りにあたって，対象プロジェクトの規模を求め，

	規模	工数	平均値	CO
過去プロジェクト2	200	2000	46.25	292.5
過去プロジェクト3	150	3000	42.14	213.21
過去プロジェクト4	210	2300	39.04	291.984
過去プロジェクト5	190	1900	71.24	325.356
過去プロジェクト6	310	2700	30.85	405.635
過去プロジェクト7	270	2650	59.01	429.327
過去プロジェクト8	255	2500	53.46	391.323
過去プロジェクト9	190	2800	58.21	300.599
過去プロジェクト10	220	2100	86.11	409.442

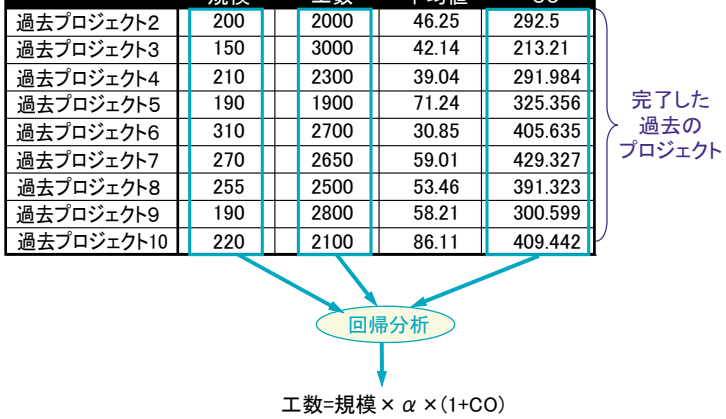
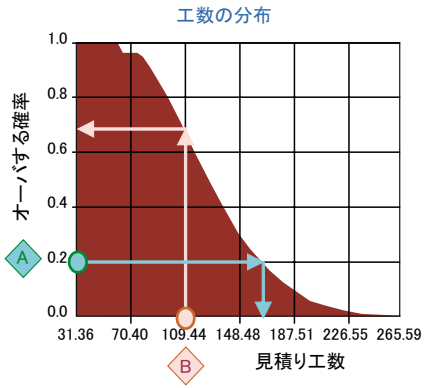


図1.6　ベースラインαの求め方

さらに，影響要因の段階を予想し，COの分布を求め，見積りの予想分布を求める。予想分布の利用方法には，2通りあるが，分布として利用するのは，図1.5から作成した**図1.7**である。これは，見積り工数をオーバする確率の分布をグラフとしたものである。

ひとつは，分布から見積り値を設定する方法で，あらかじめ組織の統一基準として実績が見積りをオーバする確率(リスク)を設定しておき，それにあった見積り値を設定するもの(図1.7のA)である。例えば，図1.7において，実績がオーバする確率を20%しか認めない(Aの場合)とすると，見積り値はおおよそ160くらいである。この確率を50%でもマネジメント可能と判断すると，見積り値は130くらいまで抑えられる。

一方，もうひとつの利用方法は，逆に顧客の事情などで予算が決まっている場合に，予算オーバしてしまう確率(リスク)を見積るもの(図1.7のB)である。まず顧客の予算に関係なく，見積りの分布を作成する。そして実は予算上避けられる工数が110だったとすると，実績がそれを超えてしまう確率は，約70%であると求められる。このリスクが高いと判断した場合は，顧客と交渉して再度予算交渉するか，規模を抑えるか，増加要因を顧客と調整して抑えるか，という対策が考えられる。実際，**図1.8**に示すとおり，どの要因が最も大きな影響を与えているかを感度分析として得ることができるので，何をコントロールすればよいかについての参考にできる。例えば，図の例では，チームの能力の



A:オーバする確率を20%とした場合の見積り工数を求める例。組織ごとにリスクは決定。

B:逆に予算上工数が決まっているときにオーバする確率を求める例(約70%の確率でオーバ)。

図1.7　見積工数の分布と見積シナリオ

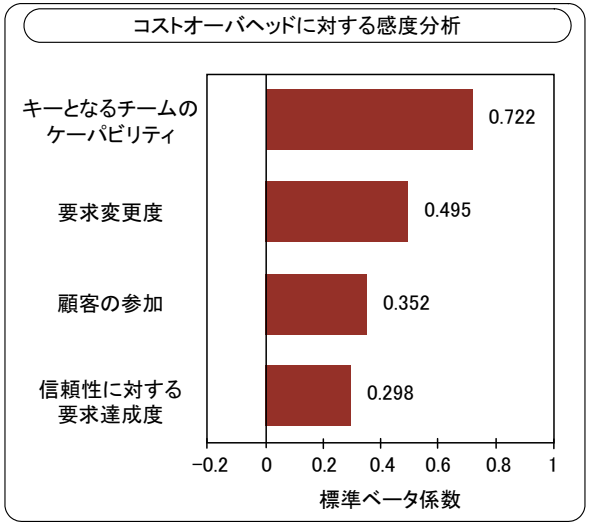


図1.8　感度分析の例

要因が大きいので，その能力を高めることで，リスクを減らすことが考えられる。このとき，どの程度のリスク軽減が可能かは，モデルの再計算により得ることができる。

図1.9にCoBRA法における見積りモデル構築の手順を示す。

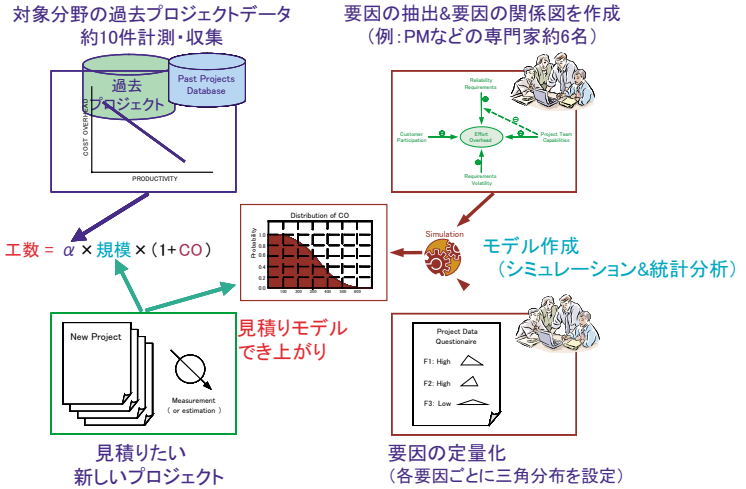


図1.9　見積りモデル構築の手順

1.3　見積り方法の前提条件

モデルの構築に際して，次のものが必要となる。

（1）　過去のプロジェクトデータ

- 10数個の過去のプロジェクトデータ。数は少なくてもよいが，工数，規模については特に精度の高いものが望まれる。
- プロジェクトデータに関して，要因がどのような状況であったかを確認可能であること。そのために，担当プロジェクトマネージャなどのプロジェクト関係者の協力が必要である。
- 経験豊富なプロジェクトマネージャの協力
- 10名程度のプロジェクトマネージャによる入力が必要である。
- 特に，組織内における経験豊富なプロジェクトマネージャであることが望ましい。

（2）　組織におけるコーディネータ

- モデル作りにあたって，組織のコーディネータの役割は大きい。要因モデルを作成するに当たってのコンセンサスづくりで，妥当な結果を得るためのバランスをとりつつ，効率よく実施したりするなど調整能力が買われる。

1.4　精度向上のための活動

CoBRA法は上記のとおり，経験豊富なプロジェクトマネージャの知識を活用することから，プロジェクトマネージャの間でのコンセンサスづくりが重要となる。また，見積りにあたって，要因のレベルの判断基準を明確にする必要がある。実施実績に示す企業での適用時には，最初のモデルを作成し，2回目のモデルで改善することにより，精度向上を実現した。

具体的には，次の事項を実施することになる。

- プロジェクトデータの精査
- 要因の見直し（人によって解釈・判断がぶれないように要因の定義の見直しを含む）
- 要因のレベル設定の確認・見直し

1.5　実施実績

ドイツにおいて4社で実績があり，国内では，2005年度にIESEとSECが共同で1社に対して試行し，良好な結果が得られている。国内の試行においては，モデル作成を2回行い，見積り精度の向上を実現することができた。具体的には，最初のモデルでは，平均見積り誤差30%であったが，要因モデルの見直しや過去プロジェクトデータの検証などを行った結果，同じ評価方法で，平均誤差14%を実現した。**図1.10**に改善の結果の様子を示す。

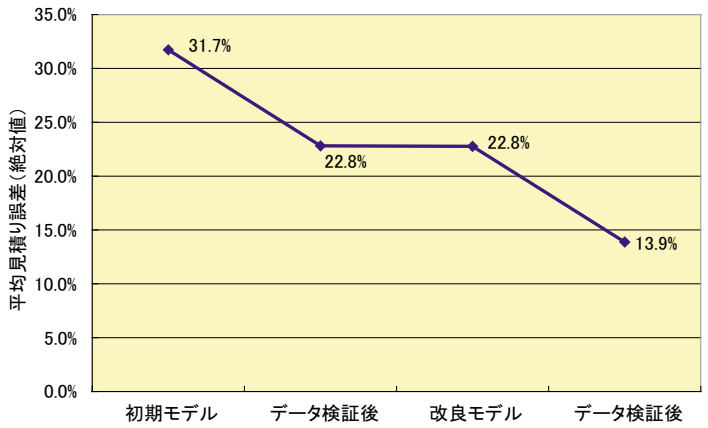


図1.10　見積りモデルの精度向上の結果

ちなみに，国内の試行は,15の過去プロジェクトデータを利用し,10名の経験豊富なプロジェクトマネージャらの参加を得て行った。要因モデルの構築には,1回あたり半日ほど全員で要因抽出とその関係を検討した。また，要因の影響の定量化は，各プロジェクトマネージャに１時間程度のヒアリングを実施して設定した。

1.6　当該方法の優位点と課題

（1）　優位点

この方法における優位点は，次のとおりである。

- 比較的容易に見積りモデルを構築できる。少数のプロジェクトデータから，見積り基準（ベースライン）と変動要因の影響度を定量化することができ，組織に適した見積り手法を構築できる。これは，現場への定量的見積りモデル作成の導入を容易にする点で非常に有効である。
- 見積り値とともに，実績がオーバーするリスクを評価できる。
- 要因項目の定量化方法として現場の理解が得られやすい。
- いわゆる「見積りのプロ」といわれる人でなくても，見積りプロセスのベストプラクティスとして，見積りモデルを利用できるようになる。つまり，組織内での見積りに関する経験知識をナレッジシェアできる。見積りで何に注目しなければならないのかが示されるので，見積りの経験が少ない人に対する教育的な効果も期待できる。
- 経験のあるマネージャが，コスト見積り作業（プロセス）に対し，必要以上に関わらなくてもすむようになる。
- 見積りプロセスの信頼性が向上する。
- コスト関連のリスク査定とリスク削減のための道具としての役割を果たす。
- 見積りにより出た数字に対し，プロジェクトのトレースができ，値にずれが生じた場合には，早期に発見することが可能であり，かつ，その原因も説明することができる。
- 測定プログラムの改善を始めることができる。
- プロジェクト間に共通の要因を把握することができるので，ソフトウェア開発プロセスの改善を始める足がかりとなる。

（2）　課　題

この方法における課題は，次のとおりである。

- プロジェクトマネージャの知識（感覚）に依存している。
- 変動要因の解釈（レベルの解釈）のぶれが生じやすい。ミーティングの時点で参加者の理解をほぼ共通のレベルにしておくことが必要である。

なお，参考文献として脚注のものがあるが[3]，CoBRA法についての詳細な利用ガイドを今後SECからも公開する予定である。

（３）　Lionel C Briand, et.al ; CoBRA：A Hybrid Method for Cost Estimation, Benchmarking and Risk Assessment, ISERN-97-24

第２章　EASE協調フィルタリング法

2.1　手法の背景

従来の見積りモデルの多くでは，見積りのもととなるプロジェクト特性変数（システム種別，開発言語，開発者数など，プロジェクトの特性を表す変数）の種類が多いほど，また，数多くのプロジェクトのデータが利用できるほど，見積り精度は高くなる。このことは，プロジェクトごとに収集される特性変数やその計測方法が異なると，利用できる特性変数が減り，見積り精度が大幅に低下することを意味する。

しかし，首尾一貫した方法で大量のデータを収集することは容易ではない。一般に，企業全体で一貫したデータ収集の体制を整えるためには長い年月を要する[4]。アメリカのSoftware Engineering Institute（SEI）は，準備を整えるまでに平均43ヶ月を要すると報告している[5]。長期にわたるデータ収集活動は，多くの工数を要する一方，データの蓄積が進まないうちは見積り精度が低く，得られる利益も小さい。大きな利益をすぐには生まない活動に大きな工数を捻出することは，多くの会社にとって容易ではない。

蓄積データが少ない組織では，独立行政法人情報処理推進機構ソフトウェア・エンジニアリング・センターが収集した「ソフトウェア開発プロジェクトデータ（SECデータベース）[6]」のように，他の（複数の）組織から収集されたデータセットを見積りに利用することも考えられる。ただし，そうしたデータセットの多くは，それぞれの組織が，それぞれの必要に応じて，ばらばらに収集し

（４）　Paulk, M., Curtis, B., Chrissis, M., and Weber, C.：Capability Maturity Model for Software（Version1.1），CMU/SEI-93-TR-024（1993）

（５）　Carnegie Mellon University, and Software Engineering Institute：Process Maturity Profile：Software CMM　2004　Year End Update（2005）.

（６）　独立行政法人　情報処理推進機構　ソフトウェア・エンジニアリング・センター：ソフトウェア開発データ白書2005～IT企業1000プロジェクトの定量データを徹底分析～，日経BP社（2005）.

たデータ（特性変数）の集積であり，結果として，欠損値（収集されていない値）が多数含まれることになる。SECデータベースにおいても，欠損値は，全体の87.3％にもなる。一般に，欠損率が30％を超えると，見積り精度が著しく低下するといわれている[7]。

2.2　方　法

協調フィルタリングは，情報検索の分野で研究されてきた技術である[8]。これを利用すると，

- 過去に手がけた類似のプロジェクトのデータをもとに，新規プロジェクトの工数を見積る，いわゆる「類推見積り」を系統的に行うことできる[9]。
- 重回帰分析やニューラルネットでは，ただ１つの見積り式（モデル）が作成されるのに対し，協調フィルタリングでは，対象プロジェクトごとに見積り式（モデル）が構築されるため，プロジェクトの個別性をより強く反映した見積りを行える。
- 欠損値の量や分布（欠損値の偏り）が変化しても精度は低下しにくい。

という従来にない見積りが可能となる[10]。

協調フィルタリングによる見積りでは，（１）特性値の正規化，（２）プロジェクト間の類似度計算，（３）類似度に基づく見積り値計算，という３つの手順で実行される。

（７）　Kromrey, J., and Hines, C.：Nonrandomly Missing Data in Multiple Regression：An Empirical Comparison of Common Missing－Data Treatments, Educational and Psychological Measurement, Vol.54, No.3, pp.573～593（1994）

（８）　Goldberg, D., Nichols, D., Oki, B.M., and Terry, D.：Using Collaborative Filtering to Weave an Information Tapestry, Comm. of the ACM, Vol.35, No.12, pp.61～70（1992）.
Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J.：GroupLens：An Open Architecture for Collaborative Filtering of Netnews, In Proc. of the ACM Conf. on Computer Supported Cooperative Work, pp.175～186（1994）
Sarwar, B.M., Karypis, G., Konstan, J.A., and Riedl, J.　：Item－Based Collaborative Filtering Recommendation Algorithms, In Proc. of the　10th Int'l World Wide Web Conf., pp.285～295（2001）

（９）　EASEプロジェクト，http：//www.empirical.jp/

（10）　柿元健，角田雅照，大杉直樹，門田暁人，松本健一：協調フィルタリングに基づく工数見積もりのロバスト性評価，第11回ソフトウェア工学の基礎ワークショップ（2004）.

（1） 特性値の正規化

各特性変数が見積りに対して与える影響を均一にするため，次式により，特性変数の値（特性値）$v_{i,j}$の値域を$[0,1]$に正規化した値$v_{i,j}'$を求める。

$$v_{i,j}' = \frac{v_{i,j} - \min(m_j)}{\max(m_j) - \min(m_j)} \qquad (1)$$

ただし，ここでは**図2.1**のような表形式のデータを用いて見積りを行うこととする。$v_{i,j}$はプロジェクトp_iについての特性変数m_jの値，$\max(m_j)$は変数m_jの最大値，$\min(m_j)$は変数m_jの最小値を表す。

特性変数

プロジェクト	m_1	m_2	…	m_j	…	m_b	…	m_n
p_1	$v_{1,1}$	$v_{1,2}$	…	$v_{1,j}$	…	$v_{1,b}$	…	$v_{1,n}$
p_2	$v_{2,1}$	$v_{2,2}$	…	$v_{2,j}$	…	$v_{2,b}$	…	$v_{2,n}$
…	…	…		…		…		…
p_i	$v_{i,1}$	$v_{i,2}$	…	$v_{i,j}$	…	$v_{i,b}$	…	$v_{i,n}$
…	…	…		…		…		…
p_a	$v_{a,1}$	$v_{a,2}$	…	$v_{a,j}$	…	$v_{a,b}$	…	$v_{a,n}$
	…	…		…		…		…
p_m	$v_{m,1}$	$v_{m,2}$	…	$v_{m,j}$	…	$v_{m,b}$	…	$v_{m,n}$

図2.1　見積りに用いる特性値行列

（2） プロジェクト間の類似度計算

見積り対象のプロジェクトp_aと見積りの根拠として用いる他の各プロジェクトp_iとの類似度$sim(p_a, p_i)$を求める。代表的なアルゴリズムである*Adjusted Cosine Similarity*と*Correlation Coefficient*における計算式は次のとおりである。

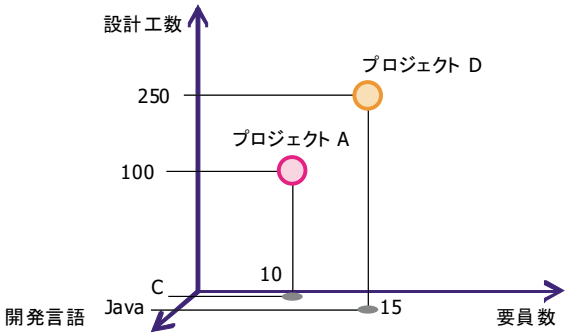
Adjusted Cosine Similarity

$$sim(p_a, p_i) = \frac{\sum_{j \in M_a \cap M_i} \left\{ \left(v_{a,j}' - \overline{m_j'}\right) \times \left(v_{i,j}' - \overline{m_j'}\right) \right\}}{\sqrt{\sum_{j \in M_a \cap M_i} \left(v_{a,j}' - \overline{m_j'}\right)^2} \sqrt{\sum_{j \in M_a \cap M_i} \left(v_{i,j}' - \overline{m_j'}\right)^2}} \qquad (2)$$

Correlation Coefficient

$$sim(p_a, p_i) = \frac{\sum_{j \in M_a \cap M_i} \left\{ \left(v_{a,j}' - \overline{p_a'}\right) \times \left(v_{i,j}' - \overline{p_i'}\right) \right\}}{\sqrt{\sum_{j \in M_a \cap M_i} \left(v_{a,j}' - \overline{p_a'}\right)^2} \sqrt{\sum_{j \in M_a \cap M_i} \left(v_{i,j}' - \overline{p_i'}\right)^2}} \qquad (3)$$

ただし，M_aとM_iは，それぞれプロジェクトp_aとp_iについて記録された特性変数の集合を表す。また，$\overline{m_j'}$，ならびに，$\overline{p_i'}$は特性変数m_j'，ならびに，プロジェクトp_i'の正規化された特性値の平均値を表す。類似度$sim(p_a, p_i)$の値域は$[-1.0, 1.0]$である。式（2），（3）では欠損していない変数のみを用いるため，欠損値処理法を用いる必要はない（**図2.2**参照）。



	要員数	開発言語	設計工数	製造工数
プロジェクト A	10	C	100 人月	（欠損値）
プロジェクト B	5	Java	（欠損値）	80 人月
プロジェクト C	25	（欠損値）	260 人月	280 人月
プロジェクト D	15	Java	250 人月	100 人月

図2.2　類似度計算

（3） 類似度に基づく見積り値計算

手順（2）で求めた類似度$sim(p_a, p_i)$を用いて，プロジェクトp_aの特性変数m_bの見積り値$\hat{v}_{a,b}$を計算する。代表的なアルゴリズム*Amplified Weighted Sum*における計算式は次のとおりである。

$$\hat{v}_{a,b} = \frac{\sum_{i \in k\text{-}nearestProjects} \left(v_{i,b} \times amplifier(p_a, p_i) \times sim(p_a, p_i)\right)}{\sum_{i \in k\text{-}nearestProjects} sim(p_a, p_i)} \qquad (4)$$

$$amplifier(p_a, p_i) = \frac{\sum_{j \in M_a \cap M_i} \left(\frac{v_{a,j}}{v_{i,j}} \times correl(m_b, m_j) \right)}{\sum_{j \in M_a \cap M_i} correl(m_b, m_j)} \qquad (5)$$

ただし，*k-nearestProjects*は，特性変数m_bの値が収集されており，かつ，プロジェクトp_aと類似度が高いk個のプロジェクトの集合を表す。kは見積り精度に影響を与えるが，図2.2で示す実験では，$k=15$で精度が最も高くなった。また，*correl*（m_b，m_j）は特性変数m_bとm_jの相関係数を表す。ただし，相関係数が負の値になる場合は0として計算した。

2.3 適用範囲（前提条件）

EASE協調フィルタリング法は，工数だけでなく，システム規模や開発リスクの見積り（予測）に利用することができる。以下では，**表2.1**に示すこれまでの適用事例を参考に，EASE協調フィルタリング法の適用範囲（前提条件）を説明する。なお，表2.1にあげた事例のほかにも，EASE（Empirical Approach to Software Engineering）プロジェクトにおいて，連携企業を中心にさまざまな適用を行っている[9]。

開発総工数の見積りを行った事例Ⅰでは，単一組織の1 081個のプロジェクトで収集されたデータを利用している。これだけ多数のプロジェクトのデータが利用できる場合は，利用できる特性変数が，13種類と比較的少なくデータ欠損率が60％と比較的高い状況であっても，EASE協調フィルタリング法では，従来法よりも高い精度での見積りが可能である[11]。

同じく開発総工数の見積りを行った事例Ⅱでは，事例Ⅰとは異なり，複数の企業で独自に収集されたデータ（組織横断的に収集されたデータ）を利用している。その結果，利用できる特性変数が，97種類と非常に多くなっているが，統一的なデータ収集となっていないため，データ欠損率も67％と非常に高い。従来法では，利用できる特性変数がいくら多くても，これだけ欠損率が高くなると，精度の高い見積りはむずかしく，組織横断的なデータ収集が行われない一因となっていた。EASE協調フィルタリング法では，統一的なデータ収集が徹底しているとはいえない状況でも，また，データの蓄積は進んでいないが他組織のデータは利用可能といった状況でも，従来法よりも高い精度での見積りが可能である[12]。

(11) 角田雅照，大杉直樹，門田暁人，松本健一，佐藤慎一：協調フィルタリングを用いたソフトウェア開発工数予測方法，情報処理学会論文誌，Vol.46, No.5, pp.1155～1164（2005）

表2.1　EASE協調フィルタリング法による見積り事例

事例番号	適用目的	利用データ	アルゴリズム	適用結果	適用組織	参考文献
Ⅰ	開発総工数見積り	特性変数：13種類（システム種別，開発種別，各工程の工数，レビューで発見された欠陥数など） プロジェクト数：1081 データ欠損率：60％	類似度計算：Cosine Similarity 見積り値計算：Amplified Weighted Sum（Type 1）	MRE：0.79 Pred 25：37％	（株）NTTデータ	(11)
Ⅱ	企業横断的データを用いた開発総工数見積り	特性変数：97種類（開発種別，FP,SLOC，各工程の期間や工数，非機能要求レベルなど） プロジェクト数：378（複数企業から収集） データ欠損率：67％	類似度計算：Adjusted Cosine Similarity 見積り値計算：Amplified Weighted Sum（Type 2）	MRE：0.64 Pred25：30％	（独）情報処理推進機構ソフトウェア・エンジニアリング・センター（SEC）	(12)
Ⅲ	プロジェクト早期のシステム規模見積り	特性変数：20種類（業種，開発言語，画面数，帳票数，ファイル数，IFPUG一般システム特性14種） プロジェクト数：85 データ欠損率：７％	類似度計算：Cosine Similarity + Euclid Similarity 見積り値計算：Amplified Weighted Sum（Type 1）	MRE：0.28 Pred25：56％	（株）日立システムアンドサービス	(13)
Ⅳ	コスト超過リスクの予測（判別予測）	特性変数：199種類（コスト超過に影響を与える可能性があるリスク項目） プロジェクト数：45 データ欠損率：42％	類似度計算：Adjusted Cosine Similarity 見積り値計算：Weighted Sum	適合率：73％ 再現率：100％ F１値：0.84	（株）日立製作所	(14)

（注）MRE（相対誤差平均）は，平均してどの程度のずれが発生するかを示す。値が小さいほど精度が高いことを示す。
Pred25は，相対誤差が0.25以下となった見積りの割合を示す。値が大きいほど精度が高いことを示す。
適合率は，コスト超過発生だと予測したプロジェクトの中で実際にコスト超過が発生した割合を示す。
再現率は，実際にコスト超過が発生したプロジェクトの中で予測が的中した割合を示す。
F１値は，適合率と再現率のバランスの良さを示す。
適合率，再現率，F１値は，いずれも値が大きいほど，精度が高いことを示す。

(12) 大杉直樹，角田雅照，門田暁人，松村知子，松本健一，菊地奈穂美：企業横断的収集データに基づくソフトウェア開発プロジェクトの工数見積もり，SEC　journal No.5（2006）

事例Ⅲでは，システム規模の見積りを行っている。単一組織で収集されたデータを利用しているが，他の事例とは異なり，対象プロジェクトの実情をよく知る技術者によって，不完全なデータや例外的なプロジェクトのデータを除外したうえで，データを見積りに利用している。その結果，85個のプロジェクトで収集されたデータしか利用できなかったが，欠損率は７％と非常に低くなっている。見積りに適さないデータの除外は，手間がかかり，系統的に行いにくい作業であるが，高い見積り精度が要求される場合には，不可欠な作業である。欠損率が低い状況では，従来法でも高い見積り精度が期待されるが，EASE協調フィルタリング法でも，MRE（相対誤差平均）が0.28，Pred 25が56％と，非常に高い見積り精度が達成される(13)。

最後に，事例Ⅳでは，他の事例とは異なり，「プロジェクトにコスト超過が発生する/しない」といった将来の状態（カテゴリ）を予測（判別予測）している。一般に，状態（カテゴリ）を判別予測する場合は，判別分析法を用いるなど，通常の見積り法との使い分けが必要となるが，EASE協調フィルタリング法では，数値を見積る場合と同様の方法で判別予測を行うことができる。しかも，再現率100％という条件のもとで，適合率73％，Ｆ１値0.84と非常に高い精度を実現している(14)。

以上のように，EASE協調フィルタリング法は，多様な目的や状況において比較的高い精度を実現する見積り法である。類似度計算や見積り値計算のアルゴリズムの改良も進んでおり，その適用範囲は広がりつつある。

2.4 実　績

表2.1に示した適用実験のうち，ここでは，SECの適用事例である開発総工数見積りの実験（事例Ⅱ）について述べる。実験では，詳細設計完了時に開発総工数を見積ることを想定し，SECデータに含まれるプロジェクト全1 009件から，「開発総工数（実績工数（総計人時）__プロジェクト全体）」が収集されているプロジェクト378件のデータを抽出し，実験データとして使用した。実験データには，数社のプロジェクトが，混在していた。全ての企業で収集された特性

(13)　大杉直樹，松本健一，津田道夫，中屋広樹，十九川博幸：協調フィルタリング技術によるソフトウェア規模の予測，日立システムジャーナル（2006），（to appear）

(14)　本村拓也，柿元健，角田雅照，大杉直樹，門田暁人，松本健一：協調フィルタリングを用いたプロジェクトコスト超過の予測，信学術報，SS2005－39，pp.35～40（2005）

変数もあるが，いくつかの企業では収集されなかった変数もある。このため，データには，多くの欠損値が含まれていた。

表2.2に実験データとして使用した特性変数と，それぞれの欠損率（対象378件における欠損値の割合）を示す。ここで，特性変数m_0が見積り対象となる開発総工数であり，m_1～m_{97}が見積りに用いる特性変数である。表2.2からも分かるとおり，実験データの欠損率は，全体で約67％と非常に大きく，欠損率が20％以下の変数は全体の約９％，逆に，欠損率が80％以上の変数は全体の約42％にも上る。分布にも大きな偏りがあり，変数m_4，m_{30}などは欠損値を全く含まないが，変数m_2，m_3などは90％を超える欠損率となっている。

相互検証法（ホールドアウト法）による10回の見積りの結果を**表2.3**に示す。表2.3では，三つの評価指標，絶対誤差平均（MAE），相対誤差平均（MRE），Pred 25によって，EASE協調フィルタリング法と従来の見積り法（対数重回帰分析，ニューラルネット，重回帰分析）との見積り精度の差を示している。各セルに記した値が，直下のセルに記した値よりも有意水準５％で有意に精度が高い場合，そのセルの値を斜体と下線によって強調している。また，精度の有意差によってセルを色分けしてある。EASE協調フィルタリング法の見積り精度が，従来法よりも高いことがわかる。特に，Adjusted Cosine Similarityを類似度計算アルゴリズムとして用いた場合のほうがCorrelation Coefficientを用いた場合よりも，高い精度であることがわかる。

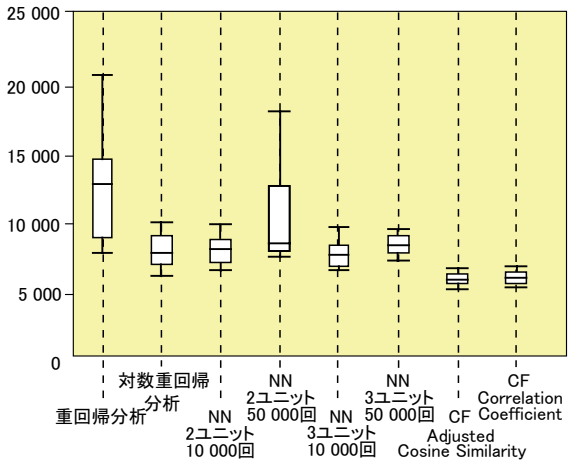
各評価指標値のばらつきを**図2.3**にボックスプロットとして示す。図2.3では，ニューラルネットをNN，協調フィルタリングをCFと略記した。EASE協調フィルタリング法，特に類似度計算アルゴリズムとしてAdjusted Cosine Similarityを用いた場合，従来法と比べて，見積り精度が高いだけでなく，ばらつきも非常に小さいことがわかる。

表2.2　評価実験に用いたデータに含まれる特性値ならびに各特性の欠損率

	特性値の種類	欠損率		特性値の種類	欠損率
m_0	実績工数(総計人時)_プロジェクト全体	0.00%	m_{49}	アーキテクチャ	0.00%
m_1	月数(実績)プロジェクト全体	42.86%	m_{50}	開発対象プラットフォーム	33.60%
m_2	利用者数	97.88%	m_{51}	Web技術の利用	44.97%
m_3	利用拠点数	97.62%	m_{52}	オンライントランザクション処理	96.83%
m_4	FP実測値_調整前	0.00%	m_{53}	主開発言語	21.16%
m_5	改修FP値_母体FP	89.15%	m_{54}	DBMSの利用	47.88%
m_6	ILF実績値_FP	60.32%	m_{55}	開発ライフサイクルモデル	0.00%
m_7	EIF実績_FP	61.90%	m_{56}	運用ツール利用	91.27%
m_8	トランザクションファンクション実績値_機能数	74.07%	m_{57}	類似プロジェクト_有無	91.53%
m_9	トランザクションファンクション実績値_FP	65.34%	m_{58}	プロジェクト管理ツール_利用	65.34%
m_{10}	データファンクション実績値_機能数	73.28%	m_{59}	構成管理ツール利用	69.05%
m_{11}	データファンクション実績値_FP	63.23%	m_{60}	設計支援ツール利用	69.05%
m_{12}	設計書文書量要件定義書	81.22%	m_{61}	ドキュメント作成ツール利用	69.05%
m_{13}	設計書文書量基本設計書	81.22%	m_{62}	デバッグ_テストツール利用	69.05%
m_{14}	設計書文書量詳細設計書	80.69%	m_{63}	上流CASEツール利用	97.09%
m_{15}	他規模指標_DBテーブル数	82.80%	m_{64}	統合CASEツール利用	97.09%
m_{16}	他規模指標_画面数	81.22%	m_{65}	コードジェネレータ利用	95.77%
m_{17}	他規模指標_帳票数	83.07%	m_{66}	開発方法論利用	95.24%
m_{18}	月数(実績)要件定義	76.72%	m_{67}	要求仕様_明確度合	52.12%
m_{19}	月数(実績)基本設計	77.25%	m_{68}	ユーザ担当者_要求仕様関与	52.65%
m_{20}	月数(実績)詳細設計	66.93%	m_{69}	ユーザ担当者_システム経験	69.84%
m_{21}	実績工数(開発)システム化計画	83.86%	m_{70}	要求仕様_ユーザ承認有無	97.35%
m_{22}	実績工数(開発)要件定義	37.83%	m_{71}	ユーザ担当者_設計内容理解度	97.35%
m_{23}	実績工数(開発)基本設計	21.43%	m_{72}	設計_ユーザ承認有無	97.35%
m_{24}	実績工数(開発)詳細設計	26.19%	m_{73}	ユーザ担当者_受け入れ試験関与	69.05%
m_{25}	外注実績(工数)要件定義	92.06%	m_{74}	要求レベル_信頼性	67.72%
m_{26}	外注実績(工数)基本設計	83.07%	m_{75}	要求レベル_使用性	97.09%
m_{27}	外注実績(工数)詳細設計	87.30%	m_{76}	要求レベル_性能・効率性	61.64%
m_{28}	レビュー指摘件数基本設計	98.94%	m_{77}	要求レベル_保守性	97.09%
m_{29}	レビュー指摘件数詳細設計	99.47%	m_{78}	要求レベル_移植性	97.35%
m_{30}	開発プロジェクト種別	0.00%	m_{79}	要求レベル_ランニングコスト要求	97.35%
m_{31}	母体システム安定度	70.11%	m_{80}	要求レベル_セキュリティ	67.99%
m_{32}	開発プロジェクト形態	0.00%	m_{81}	法的規制有無	69.58%
m_{33}	受託開発作業場所	64.02%	m_{82}	PMスキル	67.72%
m_{34}	新規顧客	86.77%	m_{83}	要員スキル_業務分野経験	61.90%
m_{35}	新規業種・業務	86.77%	m_{84}	要員スキル_分析・設計経験	67.99%
m_{36}	新規協力会社	88.62%	m_{85}	要員スキル_言語・ツール利用経験	61.90%
m_{37}	新技術利用	87.83%	m_{86}	要員スキル_開発プラットフォーム使用経験	61.90%
m_{38}	役割分担_責任所在	67.46%	m_{87}	主なFP計測手法	4.76%
m_{39}	達成目標_優先度_明確度合	67.99%	m_{88}	FP計測手法実績値の純度	5.03%
m_{40}	作業スペース	67.99%	m_{89}	FP計測_支援技術	47.62%
m_{41}	プロジェクト環境_騒音	67.99%	m_{90}	テスト体制	71.16%
m_{42}	業種	29.63%	m_{91}	定量的出荷品質基準_有無	93.92%
m_{43}	業務種類	57.41%	m_{92}	SLOC実測値_コメント行取り扱い	85.19%
m_{44}	システム用途	95.24%	m_{93}	SLOC実測値_空行取り扱い	85.19%
m_{45}	利用形態	0.00%	m_{94}	品質保証体制_基本設計	91.53%
m_{46}	システム種別	0.00%	m_{95}	品質保証体制_詳細設計	90.74%
m_{47}	業務パッケージ_利用有無	29.89%	m_{96}	品質保証体制_結合テスト	89.95%
m_{48}	処理形態	61.64%	m_{97}	品質保証体制_総合テスト(ベンダ確認)	91.01%

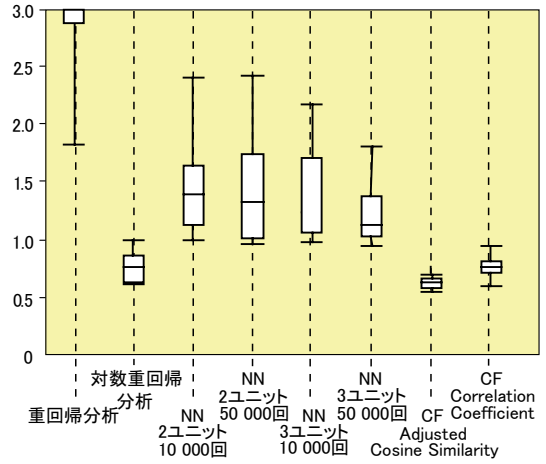
表2.3　評価結果

	絶対誤差平均 MAE	相対誤差平均 MRE	Pred25
EASE協調フィルタリング法 (Adjusted Cosine Similarity)	*5637.185* (0.017)	*0.642* (0.000)	0.301 (0.069)
EASE協調フィルタリング法 (Correlation Coefficient)	*5863.709* (0.002)	0.771 (0.417)	*0.286* (0.001)
対数重回帰分析	8102.984 (0.439)	*0.855* (0.014)	0.180 (0.129)
ニューラルネット (中間層3ユニット，学習回数50,000回)	9096.571 (0.151)	1.281 (0.222)	0.221 (0.261)
ニューラルネット (中間層2ユニット，学習回数50,000回)	*8020.968* (0.010)	1.428 (0.339)	0.218 (0.369)
ニューラルネット (中間層2ユニット，学習回数10,000回)	8135.367 (0.640)	1.473 (0.595)	0.209 (0.426)
ニューラルネット (中間層3ユニット，学習回数10,000回)	*10279.519* (0.014)	*1.584* (0.002)	0.201 (0.000)
重回帰分析	12315.063	6.247	0.101

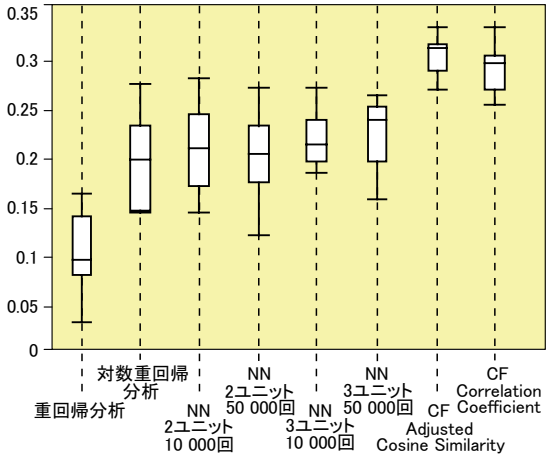


(a)絶対誤差平均(MAE)

図2.3　評価指標値のばらつき(1)



(b)相対誤差平均(MRE)



(c)Pred25

図2.3　評価指標値のばらつき(2)

2.5　当該見積り方法の優位点と課題

協調フィルタリングによる見積りは，特性変数値の欠損に非常に強い。これは，プロジェクト間の類似度を，対象となる2つのプロジェクトで共に収集されている特性変数値を利用して計算するためである。すなわち，対象となる2つのプロジェクトで共通に収集された特性変数値が一つもないという，極端な欠損がない限り，類似度計算およびそれに続く見積り値計算が可能となっている。従来の見積り法の多くでは，欠損部分に，その特性変数の平均値を挿入するなどの処理法が，開発されているが，わずかな欠損でも見積り精度は大きく低下することが，知られている。

欠損に強いということは，企業間にまたがって収集されたデータを用いた見積りにおいても優位である。2.4節で示した実験では，複数の企業から収集された欠損率約67%のデータであるにもかかわらず，工数見積りのMRE(相対誤差平均)は0.642，Pred 25は0.301であった。この精度は，単一組織で収集されたデータに基づく工数見積り事例と比べても，遜色のない精度である。例えば，文献[11]では，ある企業内データを用いて，試験工程開始までに得られるプロジェクト特性値に基づいて，試験工数の見積りを行っているが，その精度は，MREが0.79，Pred25が0.37である。

ただし，MREが0.642であることは，工数の見積り値が，平均して64.2%の誤差を含むことであり，従来法に比べて精度が高いとはいえ，改良の余地は残されている。例えば，相対誤差の大きなプロジェクトは，小規模のプロジェクトである場合が多い。ある程度以上の規模のプロジェクトに限定して利用するなど，利用方法を工夫することで，見積り精度の更なる向上が期待される。

謝　辞

本研究の一部は，独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センターとの共同研究の成果，ならびに，文部科学省「e-Society基盤ソフトウェアの総合開発」の委託に基づいて行われた研究成果に基づく。

用語解説

アイドリング

仕様決定の遅延や仕事の段取りが悪いなど，前後関係がある作業間において，前の仕事などにより特定の仕事が作業待ちになる状態をいう。

アジャイル開発プロセス

ソフトウェアをすばやく，臨機応変にむだのないように開発することを目的とした開発プロセスの総称。

開発工程

本ガイドブックでは，次のとおり設定している。

- 新規開発：新規開発の場合は，一般的なウォータフォールモデルに準拠した，システム化の方向性，システム化計画，要件定義，設計，製作，システムテストを設定。
- 保守：要件定義(システム変更計画，現行システムの理解を含む)，設計，製作，テストを設定。

機能規模

機能要件を定量化して得られるソフトウェアの規模。(「JIS X 0135-1：1999」より)

銀の弾(Silver Bullet)

「人月の神話(“The Mythical Man-Month”)」というソフトウェア工学(または，ソフトウェア開発におけるプロジェクトマネジメント)の古典を書いたBrooks教授が「銀の弾はない」と言ったことに由来する。この意味するところは，ソフトウェア開発のすべての課題に効き目のある手法・技術はないと言ったものである。「これさえ実施すれば，大丈夫」といったものを求めがちな傾向に対する警告として出された。

組込みソフトウェア

電子機器などに組み込まれて，製品の動作を制御するシステム。携帯電話や家電製品，カメラ，自動車など，コンピュータ制御を行う製品に搭載されている。

コントロール

計画と実績の比較，差異分析，プロセスが改善する傾向の評価，代替案の評価，および必要に応じた適切な是正処置の提言などを行うこと。(「PMBOK」の定義より)

ビジネスアプリケーション

財務，会計，人事，給与などの事務処理に特化したソフトウェア。

品質特性

ソフトウェアの品質を定義し，評価するために用いられるソフトウェアの属性の集合をいう。品質特性は，さらにいくつかのレベルの品質副特性(sub characteristics)に詳細化される(「ソフトウェア品質評価ガイドブック」より)。JIS X 0129-1:2003に示されている品質特性と品質副特性を**図1**に示す。

プロジェクトライフサイクル

一般に順序どおり実施される各プロジェクトフェーズを集めたもの。プロジェクトフェーズの名称や数は，プロジェクトにかかわる組織におけるコントロールの必要性により決まる。ライフサイクルは方法論によって文書化する。(「PMBOK」の定義より)

プロセス領域(プロセスエリア：Process Area)

ある領域における関連するプラクティスのひとまとまりのこと(「CMMI標準教本」より)。

ベースライン

- 見積りの基準値。プロジェクトや組織における過去のプロジェクトデータ

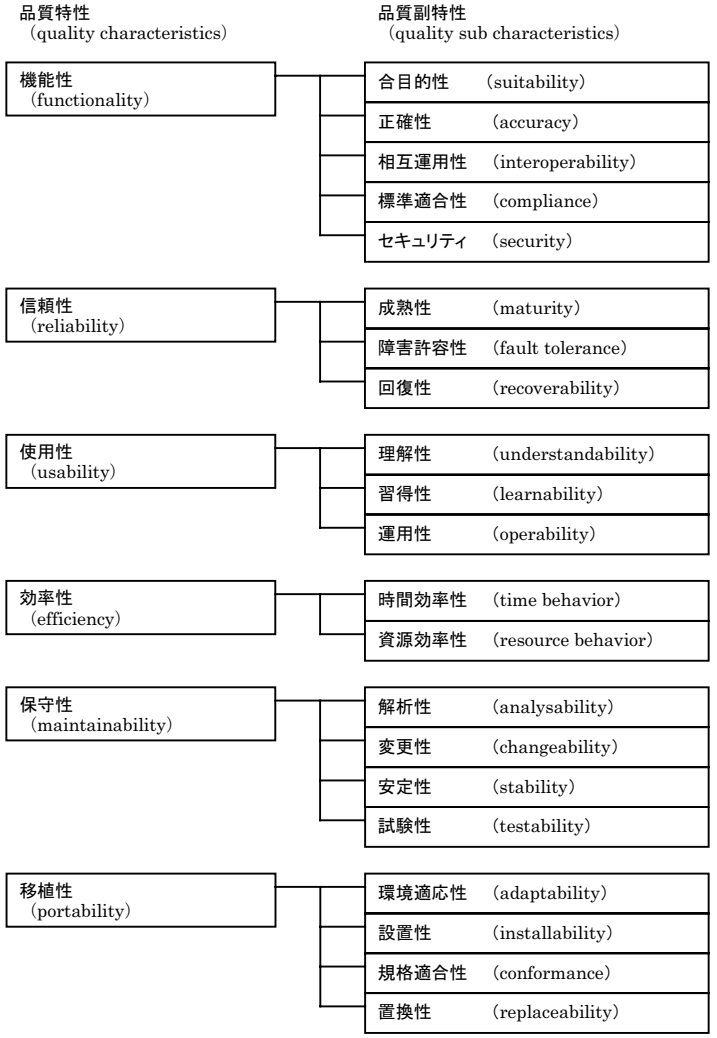


図1　品質特性の構成

などに基づいて設定される。実際の見積りでは，この基準値にさまざまな変動要因を考慮して現実的な値を使う。

- 本ガイドブックでは，基準となる値などの上記の意味で使用している。構成管理などで使われる「正式にレビューされ合意された，一連の仕様または一連の作業成果物」という意味合いではない。

ベンダ制御要因

（生産性の）ベースラインからのベンダがコントロールできる変動要因。（本ガイドブックで新出の用語）

マイルストーン

プロジェクトにおいて重要な意味をもつ時点やイベント。（「PMBOK」の定義より）

見積り成熟度

組織において，見積り活動の向上を図るうえで，次のステップに進むためのガイドを示す概念。（本ガイドブックで新出の用語）

見積り時期

- 新規開発の開発工程で設定した工程に対応させて設定した見積りタイミング。

見積り①：「システム化の方向性」工程での内容による見積り。
見積り②：「システム化計画」工程での内容による見積り。
見積り③：「要件定義」工程での内容による見積り。
見積り④：「設計」工程での内容による見積り。ユーザの観点による外部仕様が明確になる工程で，この時点で規模見積りをすると確度の高い見積りが期待できる。

見積りモデル

データ収集，ベースラインの設定，変動要因の設定，見積りのためのアルゴリズムを数式化したもの。

モニタリング（監視）

計画にかかわるプロジェクトのパフォーマンスデータを収集し，パフォーマンス測定を行い，パフォーマンス情報を報告し配布すること。（「PMBOK」の定義より）

モンテカルロ分析

プロジェクトの総コストや完了日について，その分布を計算するために，コストや所要時間の確率分布からランダムに選んだ数値をインプットとして，プロジェクトコストやプロジェクトスケジュールを多数回繰り返し計算する技法。（「PMBOK」の定義より）

ユーザ制御要因

（生産性の）ベースラインからのユーザがコントロールできる変動要因（品質などのプロダクト要件，開発制約条件，プロジェクト要因，プロセス要因，人的要因）。ユーザの努力によりコスト低減が可能なもの。（本ガイドブックで新出の用語）

要　件

- システムに対する要件

本ガイドブックでは，機能要件とそれ以外の要件（非機能要件）に分類する。

- 機能要件

利用者の要求を満足するためにソフトウェアが実現しなければならない利用者の業務および手順をあらわす。（「JIS X 0135-1：1999[1]」の「利用者機能要件」より）

- 非機能要件

本ガイドブックで，機能要件以外の要件をまとめて呼ぶ用語。

- 品質要件

JIS X 0129-1：2003で定義されたソフトウェア品質に関連した要件のすべて。（「JIS X 0135-1：1999」より）

（1） JIS X 0135-1：1999（ISO/IEC 14143-1：1998）ソフトウェア測定－機能規模測定－第1部：概念の定義。

●技術要件

ソフトウェアの開発，維持管理，支援および実行のための技術・環境に関連した要件。技術要件の例としては，プログラム言語，試験ツール，オペレーティングシステム，データベース技術，利用者インタフェース技術などがある。(「JIS X 0135-1：1999」より)

ライフサイクル

プロジェクトライフサイクルを参照。(「PMBOK」の定義より)

リスク

見積りにおいて最終的な規模やコストに対して起こりうる誤差。

「仮に発生すると，見積り(プロジェクト目標)にプラス又はマイナスの影響を及ぼす不確実な事象または状態」(「PMBOK」の定義より)

ワークブレークダウンストラクチャ(WBS)

プロジェクト目標を達成し必要な要素成果物を生み出すためにプロジェクトチームが実行する作業を要素成果物をもとにして階層的に要素分解したもの。WBSはプロジェクトのスコープの全部を系統立ててまとめて規定したものである。一段レベルが下がるごとにプロジェクトの作業はさらに詳細に定義される。WBSはワークパッケージまで要素分解される。要素成果物をもとにした階層には，組織内の要素成果物と組織外の要素成果物の両方を含める。(「PMBOK」の定義より)

ワークパッケージ

ワークブレークダウンストラクチャの各枝の最下位レベルにある要素成果物またはプロジェクト作業構成要素。ワークパッケージには，ワークパッケージの成果物またはプロジェクト作業の構成要素を完了するのに必要なスケジュールアクティビティとスケジュールマイルストーンが含まれる。(「PMBOK」の定義より)

CMMI(Copability Maturity Model Integration)

米国カーネギーメロン大学ソフトウェア工学研究所によって開発された，ソフトウェアを開発する組織の能力を示す統合された能力成熟度モデル。成果物とサービスの開発と保守に関するベストプラクティスからなる。ソフト

ウェアエンジニアリング，システムエンジニアリング，調達の分野で個別に扱われていた知識体系を統合したもの。

COTS(Commercial Off The Shelf)

商用ベンダから購入できる品目のこと。商用市販の成果物(「CMMI標準教本」より)。

DFD(Data Flow Diagram)

システム化や業務効率の改善を図る際に，データの流れに注目して作成し現状の分析を行うためのモデル図。データの流れを構造化し，必要な処理を明確化して必要な事項や問題点を整理し発見するのに活用する。DFDは，

① データの流れを示す「データフロー」
② 業務内容を示す「処理」
③ データを記録する「ファイル」
④ データの発生源と出力先を示す「データ源泉/データ先」

の4つの要素からなる。

E-Rダイヤグラム(Entity-Relationship diagram)

リレーショナルデータベースの論理設計に広く使われるデータモデル図。「従業員」「部署」などのエンティティ(Entity)を箱で表し，それらを結び付ける「～に属する」などの関係(Relationship)を表す線で結ぶ。1976年にピーター・チェン(Peter Chen)に提唱されてから研究・開発が進められ，情報システムにおけるデータモデルとして基本的なダイヤグラムとして活用されている。

PDCAサイクル(Plan Do Check Act cycle)

QC活動で広く実践されている，「Plan：計画，Do：実行，Check：チェック，Act：対策」を繰り返し行うことで，自組織のプロセスの弱点を発見，補強し，成熟度を向上させる活動。

SLOC(Source Lines Of Codes)

ソースプログラムの行数。

参考文献

［1］SEC見積り手法部会：「ITユーザとベンダのための定量的見積りの勧め」，オーム社，2005年

［2］SEC開発プロセス共有化部会：経営者が参画する要求品質の確保，オーム社，2005年

［3］Mary B. Chrissis, Mike Konrad, Sandy Shrum：「CMMI：Guidelines for Process Integration and Product Improvement」，Addison Wesley，2003年

［4］Capers Johnes：ソフトウェア見積りのすべて，共立出版，2001年

［5］児玉公信：実践ファンクションポイント法，日本能率協会マネジメントセンター，1999年

［6］佐藤正美：データベース設計論－T字型ER，ソフト・リサーチ・センター社，2005年

［7］JAPIC CMMI V1.1翻訳研究会訳：CMMI標準教本，日経BP社，2005年（［3］の翻訳）

［8］高橋宗雄：クライアント/サーバシステム開発の工数見積り技法～工数見積りモデルの適用法～，ソフト・リサーチ・センター，1998年

［9］椿正明：名人椿正明が教えるデータモデリングの技，翔泳社，2005年

［10］David Garmusら：ファンクションポイントの計測と分析，ピアソン・エデュケーション，2004年

［11］Tom Demarco：構造化分析とシステム仕様（新装版），日経BP出版センター，1994年

［12］東基衛編集：ソフトウェア品質評価ガイドブック，日本規格協会，1994年

［13］L.H.Putnam, W.Myers：プロジェクトの見積りと管理のポイント，共立出版，1995年

［14］Frederick Brooks：人月の神話―狼人間を撃つ銀の弾はない（新装版），ピアソンエデュケーション，2002年

［15］Project Management Institute：プロジェクトマネジメント知識体系ガイド第3版，2004年（PMBOKと呼ばれる）

［16］Barry Boehm：Software Engineering Economics，Prentice-Hall，1980年

［17］Barry Boehm, A.W. Brown, S. Chulani, B.K. Clark, E. Horowitz, R. Madachy, D. Reifer, and B. Steece：Software Cost Estimation with COCOMO II，Prentice-Hall，2000年

［18］前川徹：ソフトウェア最前線，アスペクト，2004年

［19］松本吉弘訳：ソフトウェアエンジニアリング基礎知識体系－SWEBOK 2004」，IEEE（オーム社），2005年

［20］John McGarryら：実践的ソフトウェア測定，共立出版，2004年

［21］真野俊樹，誉田直美：見積りの方法，日科技連出版社，1993年

［22］JIS X 0135-1：1999「ソフトウェア測定－機能規模測定－第1部：概念の定義（ISO/IEC 14143-1：1998）」，日本規格協会

［23］JIS X 0129-1：2003「ソフトウェア製品の評価－品質特性及びその利用要領」，日本規格協会

［24］ISO/IEC 24570：2005「Software engineering－NESMA functional size measurement method version 2.1－Definitions and counting guidelines for the application of Function Point Analysis」

［25］FAR（Federal Acquisition Regulation） http://www.arnet.gov/far

＜見積り時の参考データ＞

［26］ソフトウェア・エンジニアリング・センター：ソフトウェア定量データ分析白書2005，日経BP社，2005年

［27］（財）経済調査会：ソフトウェア開発費積算に関する調査研究

［28］ISBSG（International Software Benchmarkiry Standards Group），「ISBSG BENCHMARK」

［29］Capers Jones：ソフトウェア開発の定量化手法，構造計画研究所，1998年

索　引

さ　行

た　行

な　行

は　行

ま　行

や　行

ら，わ　行

アルファベット

執 筆 者（敬称略，五十音順）

主　　査：	榊原　　彰	日本アイ・ビー・エム株式会社
副 主 査：	太田　忠雄	株式会社ジャステック
委　　員：	粟野　憲一	富士通株式会社
	石谷　　靖	ソフトウェア・エンジニアリング・センター（株式会社三菱総合研究所）
	井上　智史	TIS株式会社
	大槻　　繁	株式会社一（いち）
	尾股　達也	社団法人情報サービス産業協会
	加藤　允基	株式会社日立システムアンドサービス
	菊地奈穂美	ソフトウェア・エンジニアリング・センター（沖電気工業株式会社）
	楠本　真二	大阪大学
	合田　治彦	富士通株式会社
	高橋　宗雄	桐蔭横浜大学
	角田　千晴	社団法人日本情報システム・ユーザー協会
	谷田　耕救	株式会社日立製作所
	中元　秀明	株式会社野村総合研究所
	庭野　幸夫	株式会社ジャステック
	服部　克己	日本ユニシス株式会社
	細川　宣啓	日本アイ・ビー・エム株式会社
	堀　　明広	ソフトウェア技術者ネットワーク
	向井　　清	住商情報システム株式会社
	安田　　守	ソフトウェア・エンジニアリング・センター（株式会社野村総合研究所）
	横山　健次	ソフトウェア・エンジニアリング・センター（株式会社野村総合研究所）
奈良先端科学技術大学院大学：		
	大杉　直樹	情報科学研究科（EASE協調フィルタリング法）
	松本　健一	情報科学研究科（EASE協調フィルタリング法）
経済産業省：	山田　圭吾	商務情報政策局情報処理振興課
	上原　　智	商務情報政策局情報処理振興課
	坂本　教晃	商務情報政策局情報処理振興課
執 筆 支 援：	豊嶋　大輔	株式会社三菱総合研究所

編 者 紹 介

独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター

2004年10月に独立行政法人 情報処理推進機構（IPA）内に設立されたソフトウェア・エンジニアリング・センター（SEC）は，エンタプライズ系ソフトウェアと組込みソフトウェアの開発力強化に取り組むとともに，その成果を実践・検証するための実践ソフトウェア開発プロジェクトを産学官の枠組みを越えて展開している。

［所在地］ 〒113-6591 東京都文京区本駒込2-28-8
文京グリーンコート センターオフィス
電話 03-5978-7543，FAX 03-5978-7517
http://sec.ipa.go.jp/index.php

SEC BOOKS
ソフトウェア開発見積りガイドブック
～IT ユーザとベンダにおける定量的見積りの実現～

平成 18 年 4 月 25 日　　第 1 版第 1 刷発行

編　　者　独立行政法人 情報処理推進機構
　　ソフトウェア・エンジニアリング・センター
発 行 者　佐 藤 政 次
発 行 所　株式会社 **オーム** 社
　　郵便番号　101-8460
　　東京都千代田区神田錦町 3-1
　　電 話　03 (3233) 0641(代表)
　　URL　http : //www.ohmsha.co.jp/

印刷・製本　　報 光 社
ISBN 4-274-50068-3　Printed in Japan