見積もり再考

～世界の手法から～

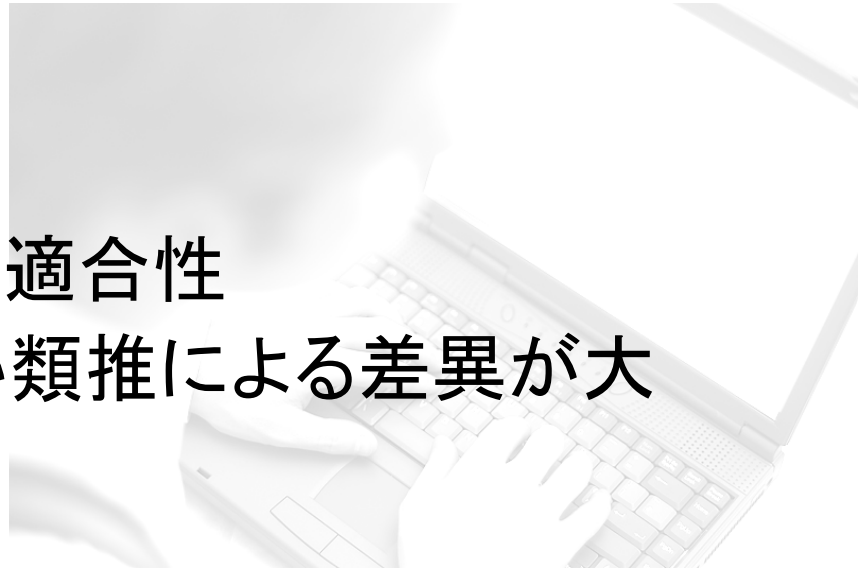
2007年5月8日

OPEN TONE

イントロダクション

- 見積もりのもっとも容易な方法
  - 経験的概算
  - 類似プロジェクトの実績値

- 蓄積がなされない
  - 精度の向上
  - 未経験技術分野への非適合性
  - 規模が大きくなるに従い類推による差異が大きくなる



経験的概算の実験

- 「(感覚で)見積もってみよう」

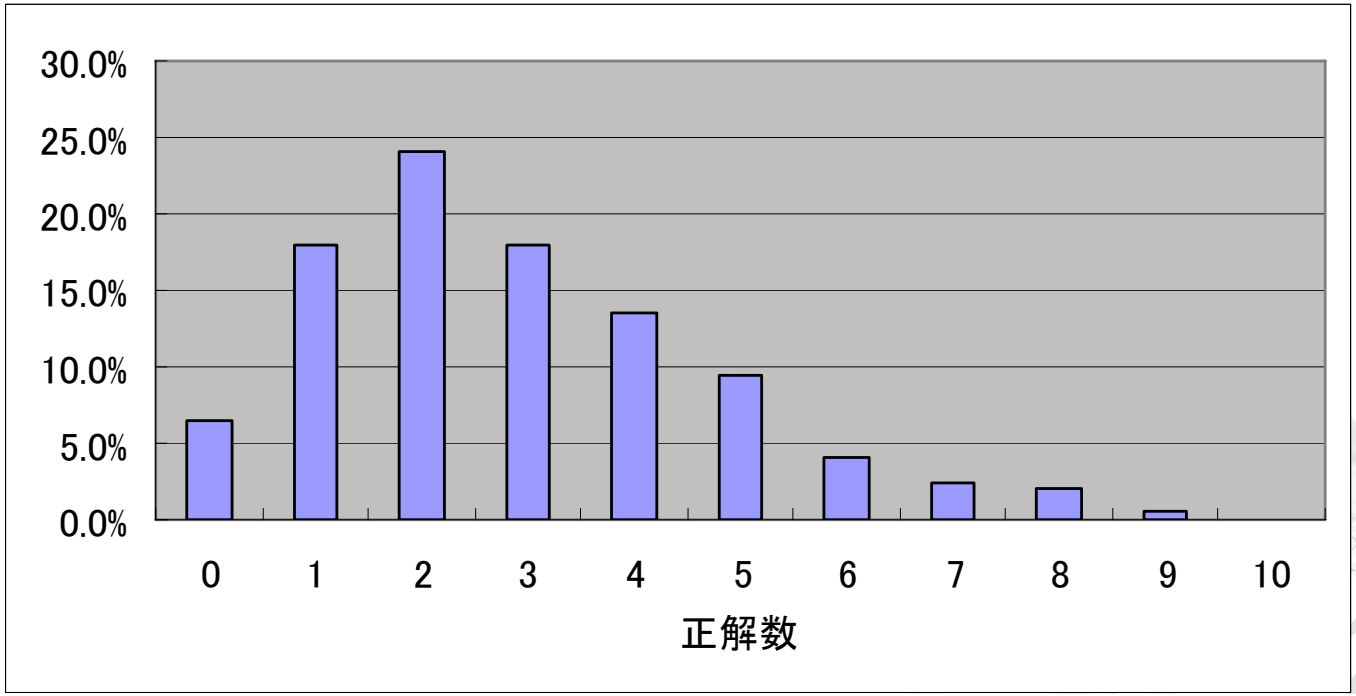
90%以内の確度あなたが思う数値を含む範囲を答えなさい

	下限	上限
太陽の表面温度		
上海の緯度		
アジア大陸の面積		
アレクサンダー大王の生まれた年		
2004年の米ドルの通貨流量		
1776年以降に米国で出版された書籍		
タイタニックの総興行収入		
日本の海岸線の総延長		
東京都の上水道の使用量(1日)		
シロナガスクジラの体重の世界記録		

実験の結果

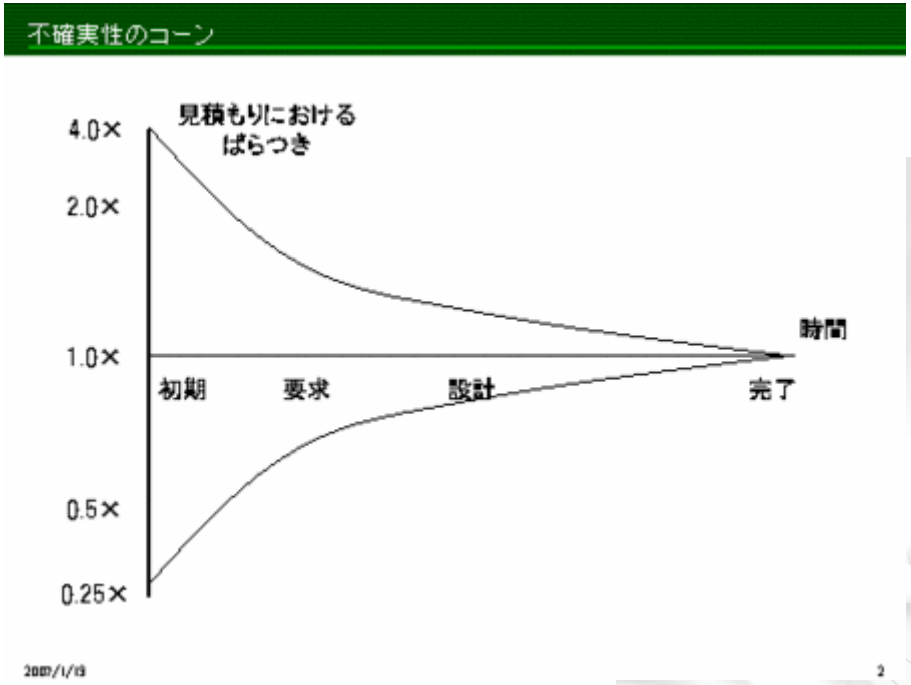
- 「(感覚で)見積もってみよう」

本来は90%になるはず



見積りの不確実性

- 見積りはだんだん増えていくのか？減っていくのか？



見積もりとは「数える」ことにより始まる

- プロジェクト初期に数えられるもの
  - ビジネス的要求
  - 機能
  - ユースケース（粒度：粗）
  - ストーリー
- 基本設計後に数えられるもの
  - 画面（あるいはWebページ）
  - データベース
  - クラス
  - コード行
  - テストケース
  - ユースケース（粒度：密）

見積もり可能な程度の基本設計

- 基本設計の初期段階に数えられるもの
  - 画面（あるいはWebページ）
    - 想定バッファ内に収まる程度の数の推定
  - データベース
    - テーブルレベルの数の推定
  - クラス
    - 設計進行時に数の増減が大→初期には難しい
  - コード行
    - 設計進行時に数の増減が大→初期には難しい
  - テストケース
    - 主要なテストケースの数の推定
  - ユースケース
    - 主要なユースケースの詳細

様々な手法での計測

- 手法により向き不向きがある
  - FPでは画面なしのシステムなどが正確に出ない
  - 1手法のみでは計測が不正確
- 複数の手法を用いるためには「共通言語が必要」
  - 通貨のような使い方
  - プログラム行数は事後計測は容易である
  - LOC ← →FPの変換により容易に2つの指標を用いることが出来る

解決策例

- コード行数を予測する
  - FPからの予測

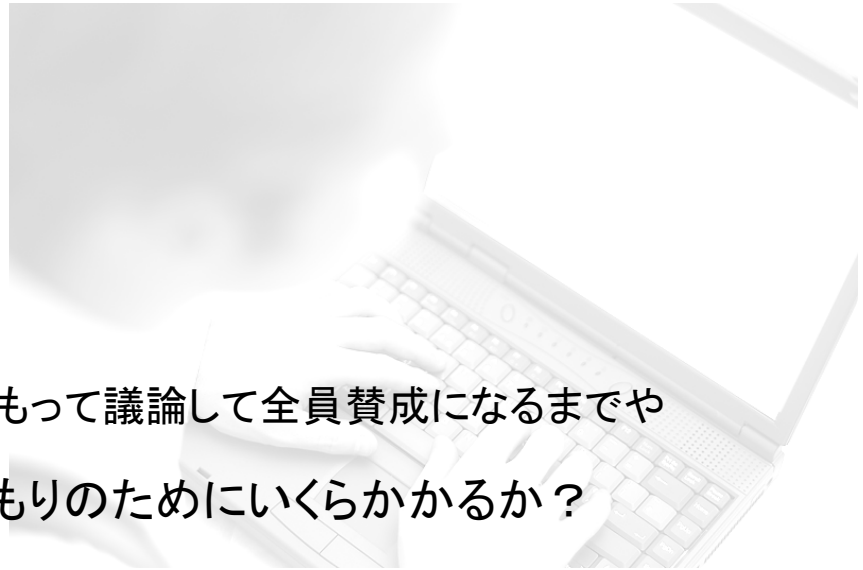
プログラミング言語	最小値 （-1シグマ）	モード （最頻度）	最大値 （+1シグマ）
C	60	128	170
C#	40	55	80
C++	40	55	140
COBOL	65	107	150
Java	40	55	80
Perl	10	20	30
SQL	7	13	15
VB	15	32	41

どうやって数えるか？

- コード行数
  - 過去の推計
    - ツール（コロ助）などで容易に計測できるため、情報の蓄積は容易
  - 実プロジェクト時
    - マイグレーション時には現行のPGから類推が可能
    - 類似プロジェクト時の類推が可能
  - 問題点
    - 新規の新しい技術でのプロジェクトや、類推の手がかりとなる過去や類似のプロジェクトがないと数値が出せない

様々な手法

- 手法の列挙
  - 見積もりの計測手法
    - COCOMO Ⅱ
    - オブジェクトポイント法
    - ファンクションポイント法
    - ISBSG
    - ユースケースポイント法
    - Price-S（現在はツール製品化）
    - WebMo
    - Putnam
    - 類推
  - 見積もりの検討手法
    - デルファイ法
      - アプローチの手法。数人で見積もって議論して全員賛成になるまでやる方法
    - DSN（Deep Space Network）見積もりのためにいくらかかるか？



COCOMO Ⅱ

- Constructive Cost Model
- プログラム行数から見積もりを行う
- 計算式に拠る見積もり
  - 粒度を定義
    - 基本・中間・詳細
  - プロジェクトのタイプを定義
    - 組織モード
    - 組み込みモード
    - 半組み込みモード
- 係数を初期状態で提示
  - 係数要素
  - 間接係数要素

COCOMO Ⅱ

- Constructive Cost Model
- 計算式に拠る見積もり
  - 粒度を定義

	粒度	係数	説明
基本	システム	なし	LOCから開発工数をシステムレベルで計算する
中間	コンポーネント	15	規模だけでなく、複雑さなとの開発特性を考慮
詳細	モジュール	15	基本設計・詳細設計・実装・テストまでのフェーズごとの考慮

  - プロジェクトのタイプを定義
    - 組織モード・・・少人数での業務アプリケーションの構築（50KL以下）
    - 組み込みモード・・・ハードウェアまで含んだ大規模開発（300KL以下）
    - 半組み込みモード・・・その中間（それ以上）
- 係数を初期状態で提示
  - 係数要素
  - 間接係数要素

COCOMO Ⅱ

- 基本式

  人月（MM）＝ コード行数(KDSI)^{基準係数}

  開発期間（M） ＝ 人月（MM）^{開発期間係数}

	コード行数（KDSI）	基準係数	人月（MM）	開発期間係数	期間（M）
組織モード	2	1.05	2.07053	0.38	1.318594
半組込モード	200	1.12	377.7057	0.35	7.980016
組込モード	500	1.2	1732.862	0.32	10.87438

注：KDSIはプログラム行数（単位1000）

COCOMO Ⅱ

- 因子

因子	非常に低い	低い	見積もり値	高い	非常に高い	特別に高い	影響度
アプリケーション(業務領域)の経験	1.22	1.10	1.00	0.88	0.81		1.51
データベースの規模		0.90	1.00	1.14	1.28		1.42
再利用を考慮した設計		0.95	1.00	1.07	1.15	1.24	1.31
文書化要求の程度	0.81	0.91	1.00	1.11	1.23		1.52
使用言語とツールの経験	1.20	1.09	1.00	0.91	0.84		1.43
複数サイトでの開発	1.22	1.09	1.00	0.93	0.86	0.78	1.56
要員の継続性(離職率)	1.29	1.12	1.00	0.90	0.81		1.59
プラットフォームの経験	1.19	1.09	1.00	0.91	0.85		1.40
プラットフォームの変わりやすさ		0.87	1.00	1.15	1.30		1.49
プロダクトの複雑さ	0.73	0.87	1.00	1.17	1.34	1.74	2.38
プログラマの(一般的)スキル	1.34	1.15	1.00	0.88	0.76		1.76
ソフトウェアの信頼性に関する要求	0.82	0.92	1.00	1.10	1.26		1.54
要求アナリストのスキル	1.42	1.19	1.00	0.85	0.71		2.00
ストレージに関する制約条件			1.00	1.05	1.17	1.46	1.46
時間的な制約条件			1.00	1.11	1.29	1.63	1.63
ソフトウェアツールの活用	1.17	1.09	1.00	0.90	0.78		1.50

FP

- Function Point method
  - IFPUG法
  - フューチャーポイント法
  - COSMIC-FFP法
- 定量的な計測手法

種類	例	複雑さ		
		低	中	高
外部インプット	入力画面	3	4	6
外部アウトプット	集計や分析、フォーマッティングを行った出力画面、帳票	4	5	7
外部クエリ	インプットとアウトプットがそのまま同じような単純な入力・出力機能。単票の入力画面など	3	4	6
内部論理ファイル	データベース	4	10	15
外部インターフェイスファイル	オンライン処理など	5	7	10

IFPUG

- IFPUG法
  - もっとも一般的なFP法
  - 画面やファイルなど「処理」を計測する手法

種類	例	複雑さ		
		低	中	高
外部インプット	入力画面	3	4	6
外部アウトプット	集計や分析、フォーマッティングを行った出力画面、帳票	4	5	7
外部クエリ	インプットとアウトプットがそのまま同じような単純な入力・出力機能。単票の入力画面など	3	4	6
内部論理ファイル	データベース	4	10	15
外部インターフェイスファイル	オンライン処理など	5	7	10

フューチャーポイント

- フューチャーポイント法
  - 画面が無いが複雑なシステムはIFPUG法で計測できない
  - 「アルゴリズム」を要素に加えている

種類	例	重み
アルゴリズム		3
外部入力	入力画面	4
内部出力	集計や分析、フォーマッティングを行った出力画面、帳票	5
論理的内部ファイル	データベース	7
外部インターフェイスファイル	オンライン処理など	7
外部照会	一覧検索など	4

ISBSG方式

- International Software Benchmarking Standards Groups
- FPやCOCOMOより複合的な要素を用いている
  - プロジェクトの規模
  - 種類
  - チーム人数
- 600ほどのデータを組織的に収集して公式を作成している

ISBSG方式

- プロジェクトの種類と係数

人月（MM）＝　補正係数＊FP^{FP係数}＊最大チーム人数^{人数係数}

公式集	補正係数	FP係数	人数係数
一般	0.512	0.392	0.791
汎用機	0.685	0.507	0.464
サーバー	0.472	0.375	0.882
PC	0.157	0.591	0.810
第3世代言語	0.425	0.488	0.697
第4世代言語	0.317	0.472	0.784
機能強化	0.669	0.338	0.758
新規開発	0.520	0.385	0.866

オブジェクトポイント

- 非常に初期の段階で使用する見積もり法
  - 画面と帳票の要素だけで見積もりが可能
  - 人月　＝　（画面・帳票ポイント）　/　PROD

画面・帳票項目数	データソース（テーブル）		
	1～3	4～7	8以上
1～2	S	S	M
3～7	S	M	D
8以上	M	D	D

	S	N	D
画面	1	2	3
帳票	2	5	8

ICASE maturity and capability	Very Low	Low	Nominal	High	Very High
PROD	4	7	13	25	50

WebMO

- Web独特の要素を考慮してポイントを生成
- 指標は細かい
- 設計が進む必要がある

Web Object Predictors	Low	Medium	High
Other	2	4	6
# of scripts	1	2	3
# of graphics files	2	4	6
# of xml, sgml, html & query lines	3	5	8
# of web components	2	4	6
# of application or object points	*	*	*
# of multi-media files	1	2	4
# of components	2	4	6
# of building blocks	1	2	4

	A	B	P1	P2
Web based	2.1	2	1	*
B-to-B applications	2	1.5	1	*
Financial/trading application	2.7	2.2	1.05	*
Web based electroniccommerce	2.3	2	1.05	*

Putnam モデル

- 基本的にはLOCからの算出構造
- Construx Estimate の基礎理論

$$ProcessProductivity = \frac{Size}{\left[\frac{Effort}{\beta}\right]^{1/3} \cdot Time^{4/3}}$$

$$Effort = \left[\frac{Size}{Productivity}\right]^{3} \frac{1}{Time^{4}}$$

$$\frac{\beta \cdot Size}{Productivity} = Effort^{1/3} \cdot Time^{4/3}$$

ユースケースポイント

- ユースケースポイント法
  - ユースケース図ないしハイレベルの記述を書く
  - アクターに重みをつける

意味	タイプ	係数
プログラムによるインターフェイス	単純	1
対話型あるいはプロトコルでのインターフェイス	平均	2
GUI でのインターフェイス	複雑	3

  - ユースケースに重みをつける

意味	タイプ	係数
3つ以下のトランザクション/4以下の分析クラス	単純	5
4から7のトランザクション/5から10の分析クラス	平均	10
8以上のトランザクション/11以上の分析クラス	複雑	15

ユースケースポイント

– 技術要因の重み付け（TCF）

要因番号	意味	重み係数値	重みレベル
T1	分散システム	2	
T2	レンポンスやスループットのパフォーマンスが重要	1	
T3	オンラインでエンドユーザが効率的に使える	1	
T4	内部処理が複雑	1	
T5	コードが再利用可能である	1	
T6	インストールしやすい	0.5	
T7	使いやすい	0.5	
T8	移植性が高い	2	
T9	変更しやすい	1	
T10	並行処理を行う	1	
T11	セキュリティについての特別な配慮が必要	1	
T12	サードパーティから直接アクセスできる	1	
T13	ユーザの特別なトレーニングが必要	1	

– TFactor = Σ重み×係数

– TCF = 0.6 + (0.01×TFactor)

0-5で重みレベルを振る

(0=無関係，**5**=本質的, **3**=平均的)

ユースケースポイント

- 技術要因の重み付け(EF)

意味理由	要因番号	重み係数値	重みレベル
採用するプロセス(工程)に慣れている(*1)	E1	1.5	
アプリケーション開発の経験がある	E2	0.5	
採用する方法論に慣れている(*2)	E3	1	
リーダの能力	E4	0.5	
モチベーション	E5	1	
仕様の安定性	E6	2	
メンバに外注，アルバイトが含まれる(*3)	E7	-1	
プログラミング言語が難しい(*4)	E8	-1	

- EFactor = Σ 重み × 係数
- EF = 1.4 + (-0.03 × EFactor)

0-5で重みレベルを振る

(0=無関係，**5**=本質的, **3**=平均的**)**

ユースケースポイント

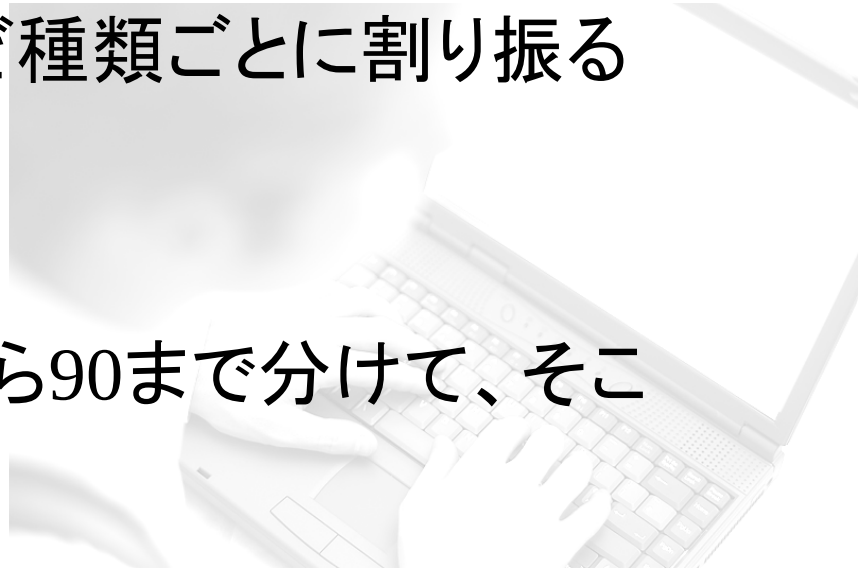
- ユースケースポイントの算出

  UCP = UUCP * TCF * EF

  – 1UCP = 20人時間　ただし環境要因の危険指数が高い場合は28時間などに直す

個別技法

- 難易度などをどうやって分けるか？
- ファジー理論
  - 小さいから非常に大きいまで分類して平均値で割り振る
  - 応用して、DB、画面など種類ごとに割り振る
- パーセンタイル
  - 自社のPJのLOCを10から90まで分けて、そこに5段階を振る



個別技法

- PERT法の採用(Putnum and Myers 1992,Stutzke 2005)
  - （Program Evaluation and Review Technique）
  - 数種類の手法や、担当者で見積もる
  - 複数の手法で見積もる
- どれを採用すべきか？平均？！

  期待ケース＝（最良ケース＋（4＊最有力ケース）＋最悪ケース）／6

  期待ケース＝（最良ケース＋（3＊最有力ケース）＋最悪ケース＊2）／6

スケジュール

- スケジュールの公式(1981 Barry Boehm)
  月 = 3.0 * MM^1/3

- スケジュールの公式(1988 William Roetzhei)
  スケジュール　= 過去のスケジュール * (MM/過去のMM) ^1/3
    注:規模の小さいものは^1/2

- いずれにせよ不可能領域がある(Putnuam & Myers 2003)
  - スケジュールの短縮は工数の増大を招く
  - 25%程度から不可能領域に入る

DSN

- 見積もり作業の見積もり

CE = K*CP^ 0.35

CPはPJ規模（見積もり規模100万ドル）

CEは見積もりの価格（100ドル）

Kの値

概算	24
予算策定	60
決定可能な	115

- プロジェクト予算が1億円で概算

CE = 24*1 ^0.35

¥2,400,000円が概算見積もりに必要であり

CE = 115*1 ^0.35

¥11,500,000円が決定的な見積もりには必要