

計測 と 見積り

～ ソフトウェア開発のよりよい見積りのために ～

biac

わんくま同盟 名古屋勉強会#07

2009/04/11



わんくま同盟 名古屋勉強会 #07

「見積り」とは

- 過去の経験に基づいて、未来を予測する技術
※ 工数・工期の見積りまでは技術。見積書に記載する金額は「政治」
- すなわち、過去のデータを未来にあてはめると



計測

見積り

見積りの基本 = サイズ ÷ 速度

- 例) 遠足

- 出発地点から目的地までの距離 (サイズ) : 8km
- 歩く速さ (速度) : 4km/h
- 必要な時間は? $\Rightarrow 8 \div 4 = 2$ 時間

- 注意

- ここでは、サイズは正確な値。ぶれる事はない。
- 速度 (4km/h) は、経験値。実際にやってみると、たぶん違う値になる。

見積りの基本 = サイズ ÷ 速度

- 例) 遠足2 … サイズも経験値

- 出発地点から目的地までの直線距離 : 8km
実距離の推定 : 折れ曲がり係数(仮称) = 1.5
⇒ 実距離(推定) = $8 \times 1.5 = 12$
- 歩く速さ (速度) : 4km/h
- 必要な時間は? ⇒ $12 \div 4 = 3$ 時間

見積りの精度 ～ サイズ と 速度

- ソフトウェア開発の見積り精度を良くするには
 - プロジェクトのサイズをより正確に見積もること
 - チームの速度（生産性）をより正確に見積もること
- **×** 見積り結果だけを精度アップさせよう
 - 生産性の向上分が計測できない
 - ⇒ 開発者のスキルアップを評価しない
 - サイズの見積り外れ(仕様膨張)を見ない
 - ⇒ 遅延の原因はすべて開発者の責任にされる

閑話休題 (1) : 遅れを取り戻せ!

- 予定工期の半分で 遅延が発覚
 - トータル 8km、1時間で 3km だった
⇒ $(4\text{km}(\text{見積り}) - 3\text{km}(\text{実績})) \div 4\text{km} = 25\%$ (遅れ)
 - 残り 1時間で 5km だから、後半のスピードを…
⇒ $5\text{km/h}(\text{目標}) \div 4\text{km/h}(\text{見積り}) = 125\%$
… 25% アップすればよい (?)
 - これは実績ベースで考えないといけない!
⇒ $5\text{km/h} \div 3\text{km/h}(\text{実績}) = 167\% !!$

ソフトウェアのサイズ (1)

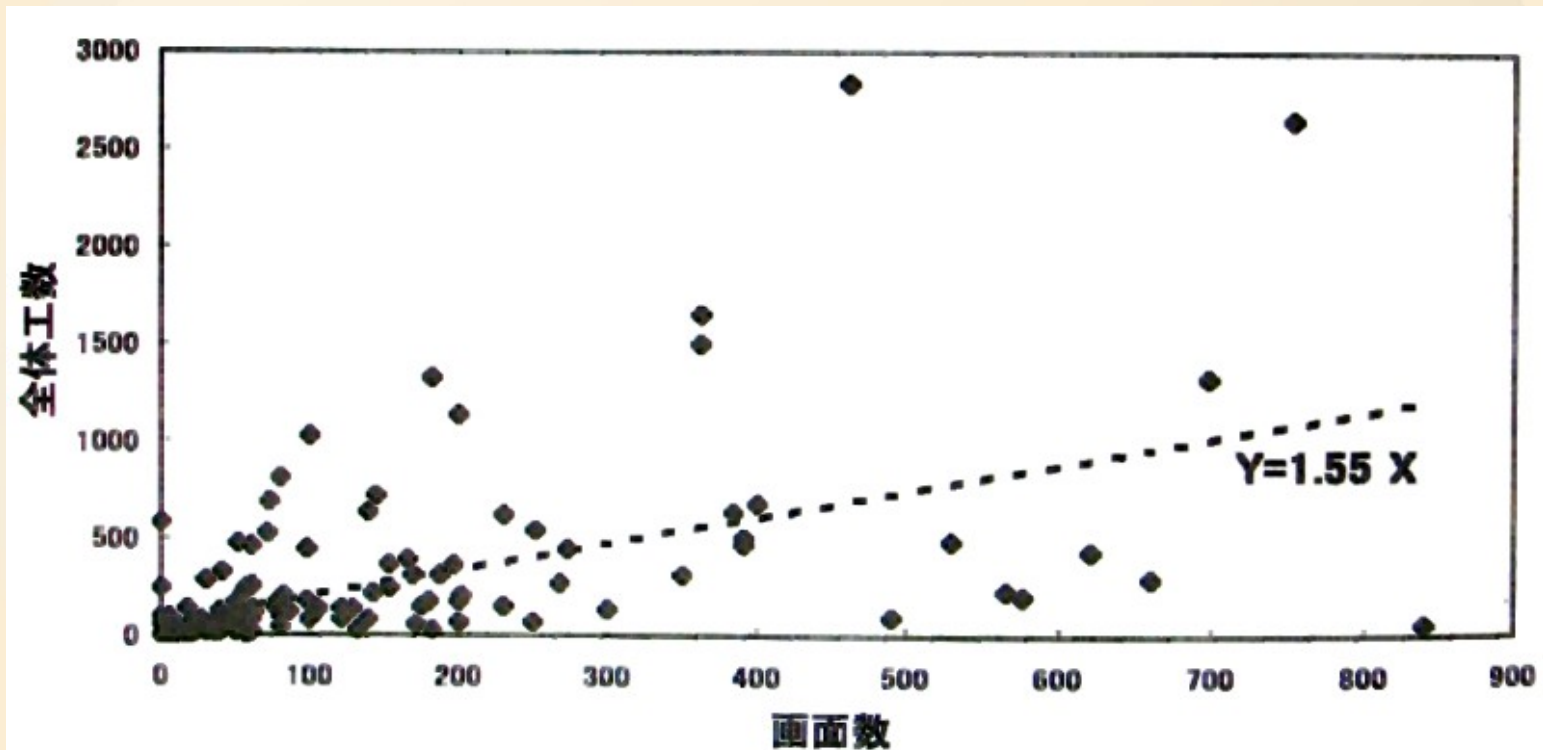
- **コード行数**
 - 一定の条件を満たせば、サイズを測るのに使える。
- **同一言語、同一ライブラリ**
- **同一の品質**
 - コード品質計測ツール
 - リファクタリングに主眼を置いたコードインスペクションの実施

ソフトウェアのサイズ(2)

- 画面数と帳票数

- 日本の業務アプリでは、けっこう使える

- ※ 人月 = 画面数 × 1.55 (画面数 ~ FP データもあり)



出典: 「JUAS ソフトウェアメトリックス調査 2007」 p.148

ソフトウェアのサイズ (3)

- **ファンクションポイント (FP)**

- 現在、世界的に一番権威がある
- FP算出法にも何通りかあるので、どの手法で計測したか明らかにすること

IFPUG 法, NESMA法, (画面)要素見積法 (日本)

<http://www.nesma.nl/sectjap/home/earlyfpa.htm>

<http://sel.ist.osaka-u.ac.jp/~lab-db/betuzuri/archive/582/582.pdf>

ソフトウェアのサイズ (4)

- その他

- オブジェクトポイント法、ユースケースポイント法など
- 粒度を一定にするのが難しそう

- 計測手法の選定には

- 安定して計測できること
- 他と（時間軸, 空間軸）比較可能なこと

成果物の見積り

- 生産性を、個々の成果物 (work) の量に対して計測/見積りしている場合は、ソフトウェアのサイズから成果物の量を見積もる
 - 設計書のページ数
 - コードの行数
 - テスト報告書のページ数
 - マニュアルのページ数

MSN Encarta より

work

8. **something manufactured**: something that has been or is in the process of being worked on or manufactured.

生産性 (1)

- ※ 生産性を見積もる ← 生産性を計測すること
- 同質の作業ごとに計測する。少なくとも 3つに分けて計測する
 - 要件開発 (なにを作るか) … いわゆる要件定義から外部設計まで
 - コンストラクション (どう作るか) … いわゆる内部設計から実装完了まで
 - 検証 (どこか間違っていないか) … いわゆる結合テスト以降

生産性 (2)

- 細かく見る場合には、前述の成果物 (work) 単位にまで分けて計測すること。
 - ※ 米国産の見積り支援ソフトには、そこまで入っている。
 - 中間成果物まで細かく規定されている「重たい」開発プロセスなら、可能。

FP - 成果物量 の例

文書	ページ数 / FP
プロジェクト計画文書	0.30
要求定義文書	0.53
機能仕様文書	0.93
ロジック仕様文書	0.89
テスト計画文書	0.29
ユーザーチュートリアル文書	0.45
ユーザー用参考文書	0.60
オンラインヘルプ	0.34
合計	4.31

出典: 「ソフトウェア見積りのすべて」 p.231 表13.11 より抜粋

生産性データの適用範囲

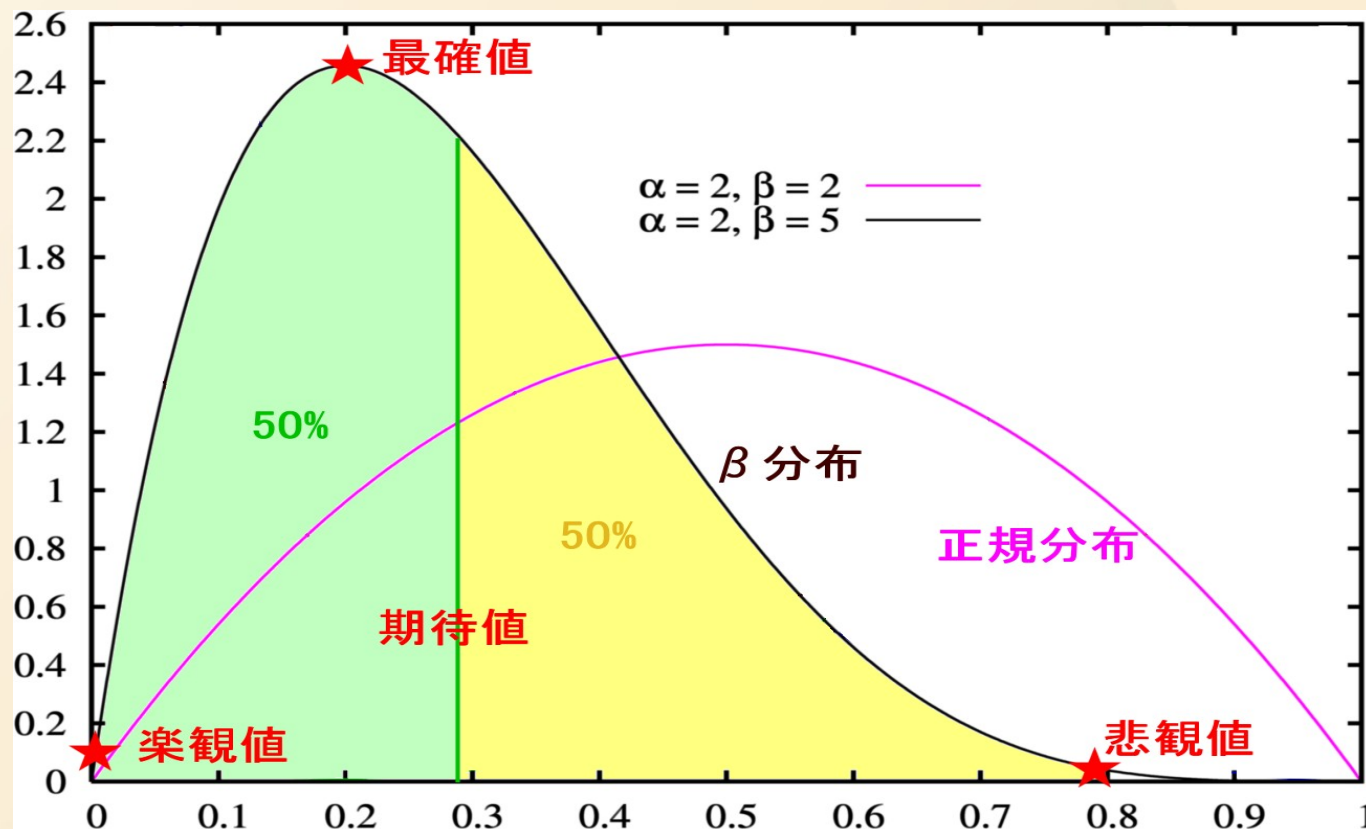
- ソフトウェア開発の生産性はバラつきが大きい
 - 人 (属人的なスキル)
 - チーム (属人的、組織的なスキル)
 - 開発プロセス
 - ※ 求められる品質と生産性の関係は、おそらく定量化可能
- 条件が変わったら、適用できない！
 - 推定する + 不確実性を大きく見る必要がある

閑話休題 (2) : テストの工数

- 「テスト工程」の工数はバラつく
 - テストケース開発 ~ テスト実施 ← サイズに比例
 - バグ票の起票 ~ トリアージ ~ 不具合修正 ~ 再テスト ← バグ数に比例
 - 起票以降の工数は、バグを作りこんだ工程 (設計/製造) に振替えるべきかも。
 - ※ そうすれば、設計/製造での品質向上へのインセンティブになる。

不確実性は非対称 ～ β 分布

- 短縮は限度があるけど、遅延はどこまでも
- 確率分布が最大の点（最確値）より、期待値は右側



出典: Wikipedia

http://en.wikipedia.org/wiki/File:Beta_distribution_pdf.png の図より抜粋の上、追記

3点見積り ～ β 分布の応用

- 最良と最悪も加えた 3点から算出する

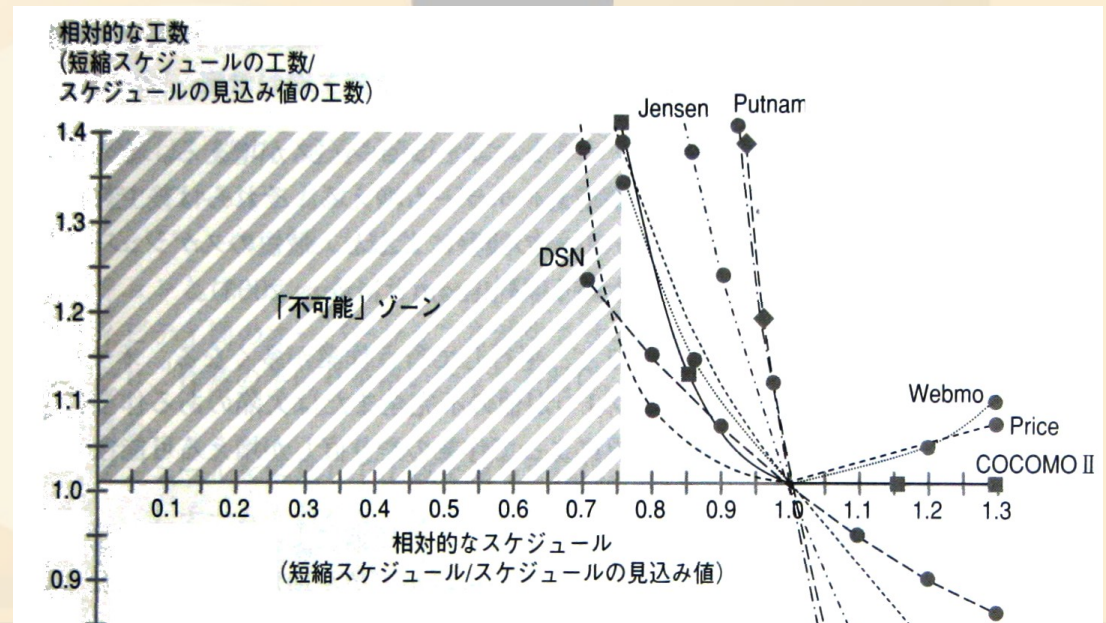
- ベータ分布の標準偏差 = (楽観値 - 悲観値) $\div$ 6
と仮定する。すると、期待値は次式で表される。
(PERT, 1958)

$$\text{期待値} = (\text{楽観値} + \text{最確値} \times 4 + \text{悲観値}) \div 6$$

工期 ← 工数 (ただし…)

- 工数見積りから工期を見積もることができる。
 - 最適工期 = $2.5 \times \text{工数}^{0.35}$ [COCOMO]
 - 最適工期 = $2.4 \times \text{工数}^{0.33}$ [JUAS 2007]
- ただし。最適工期から 25% 以上短縮することは不可能 !!

出典: 「ソフトウェア見積り
— 人月の暗黙知を解き明か
す」 p.243 より



閑話休題 (3) : レビューの工数

- 文書のバグ出しレビュー = インспекション
 - レビューアーによる査読 : $0.06 \sim 0.1 \text{H/ページ} \times 3 \text{人}$
 - レビューミーティング : $0.05 \sim 0.2 \text{H/ページ} \times 5 \text{人}$
 - 文書修正 : $0.1 \sim 0.5 \text{H/ページ} \times 1 \text{人}$
 - レポート・フォロー・進行 : $0.01 \sim 0.05 \text{H/ページ} \times 1 \text{人}$
 - 合計 : $0.5 \sim 2.0 \text{人} \cdot \text{H/ページ}$
⇒ 400ページ (約100FP のプロジェクト) には、
200～800人H (1.25～5人月)

出典: 「ソフトウェア見積りのすべて」 第20章のデータを基に試算した

プロジェクトとの関わり

開発プロセス	・ 計測と見積りの区切りかた
プロジェクト実施	・ 計測 ・ 進捗のフィードバック
プロジェクトの評価	・ 生産性のフィードバック
案件の引き合い	・ ラフな見積り
プロジェクト計画	・ 見積り ・ 計画へのフィードバック

まとめ(1)：計測なくして 見積りなし

- ・ 精確な計測から、精確な見積りが生まれる
- ・ 精確な見積りができると…
 - 赤字プロジェクト・デスマーチが減らせる
 - 生産性の向上が計れるようになる
 - ※ 見積り精度が $\pm 10\%$ で、実績が20%良かったら、10%位は生産性が上がったと言える。
 - ⇒ 開発プロセスの改善が進む

まとめ (2) : 工数見積りは技術

- サイズ・工数を見積もるところまでは、技術
 - 「政治」 嫌いの開発者も知ってほしい
- 過去の計測値から未来を予測する
 - 計測していなければ、良い見積りは出せない
- 見積りは確率
 - 未来予測に 100% 正確はありえない
 - 幅のある未来予測から決断するのは経営の責任 (技術サイドだけで判断してはいけない)

参考文献

- 「ソフトウェア見積りのすべて ― 規模・品質・工数・工期の予測法」
Capers Jones, ISBN-10: 4320097319
- 「ソフトウェア見積り ― 人月の暗黙知を解き明かす」 Steve McConnell,
ISBN-10: 489100522X
- 「本当に使える見積もり技術 ― ソフトウェア開発を成功に導く」 初田 賢
司, ISBN-10: 4822229777
- 「JUAS ソフトウェアメトリックス調査2007」 (社)日本情報システム・
ユーザー協会, ISBN-10: 4903477060
- URL
 - ・ IPA/SEC <http://sec.ipa.go.jp/index.html>
 - ・ 初田賢次 ちょっと待った！ その見積もり
<http://itpro.nikkeibp.co.jp/watcher/hatsuda/>
 - ・ プロジェクトマネジメント O S 本舗 ― 手法 7
<http://pmstyle.jp/honpo/method/PERT.htm>