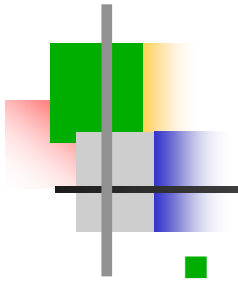
間違いだらけのSPI

– 開発組織の観察を通して –

松原 友夫

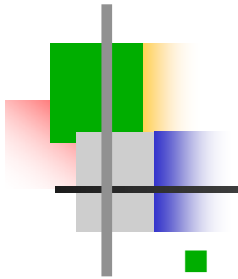
Matsubara Consulting

tmatsu@xa2.so-net.ne.jp



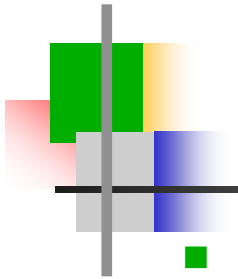
何を得たいのか分からない

- 欲しいのはレッテル
  - 表向きには改善と言うが、本心は看板
    - いまどこにいてどこに行きたいかを示さない
    - とにかくCMM(I)レベルXをとれ！
    - いくら「獲得は誤り」と言っても治らない
      - ISO 9000認証の後遺症
      - アセスメント/アプレーザル範囲は小さくアナウンス範囲は大きく
  - 現時点の実力を知らない/がわからない
    - 組織の計画達成分布を掴んでいない
      - コスト以外は把握していない組織が多い
    - 世の中での位置についてはさらに無知
  - どこに向かって走ればいいのか？
- ビジネス上の効果が得られてこそ改善



モデルをチェックリストとして使う

- プロセス・エリアの升目にチェックするだけ
  - チェックリストとして使う組織は点取り主義
    - 広く浅い網羅主義
    - コンテキストに適合したプロセスの重点化という意識が乏しい
    - ソフトウェア・エンジニアリングの基礎ができていない
      - ソフトウェア・エンジニアリングの主要な技術を知らない
    - エンジニアリングセンスさえ乏しい
      - 問題の攻め方がわかっていない
    - モデルが示しているのはプロセスの枠組みだけ
      - 一つ一つの升目には膨大なソフトウェアエンジニアリングの蓄積がある
- **よいプロセスはプロの規律を備えた組織文化が支えてこそ維持できる**
  - CMMの著者の一人、Bill Curtisが書いた次のエッセイは必読
    - http://www.magdesign.com/files/WhichComesFirst.pdf



測定しても使われない数字

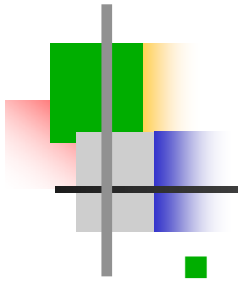
- 数字は集めるが…
  - 使い方を知らない
  - 使い方を後で考える
  - 定義がいい加減で分析に使えない
  - 分類が不適切
    - 直交性が不適切
      - どこに分類すればよいか迷うような分類
    - 粒度が不適切
    - 細かすぎる分類
  - 問題の理解を助ける工夫が乏しい
    - 生きたグラフが使われていない
- 数字は使うためにある
  - エンジニアリングでは問題が先にあってそれを理解し解決するために実験し計測して数字を集める
  - そのために生きたグラフを作る

なぜか速さを競う

- わずか7ヶ月でレベルx達成!?
  - この見出しに意図が見え見え
  - 基礎が出来ていたからならよいが、大部分は拙速
  - 基礎ができていない付け焼き刃のSPIはすぐに後戻り
  - 後戻りは最大のリワーク
  - 7ヶ月がよくて7年が悪いのか?
  - 速さを求めれば基礎固めがおろそかになる
  - SPIを支える企業文化の醸成には時間がかかる
- 基礎固めに重点を置けば時間がかかるのは当然
  - 実質的な改善を目指す組織にはアセスメント結果を公表しないところも多い

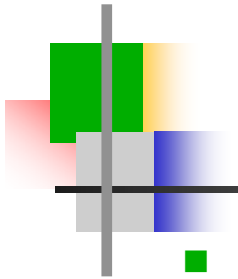
笛吹けど踊らず

- 冷めた現場
  - ISO 9000の騒ぎの次はまたCMMか!
    - 特に問題がないのになぜ改善が必要なのか？
      - 問題が分かっていない
      - 外部からの刺激がない
    - トップのせいにする
- 理解の足りないトップ
  - 改善リソースを与えようとしない
    - 改善せよと発破をかけるが
      - 乏しい予算
      - 乏しい専任者
      - 貧弱な開発環境
      - 足りない教育･訓練への熱意
    - 現場のせいにする
- トップと現場が噛み合ってこそ改善ができる



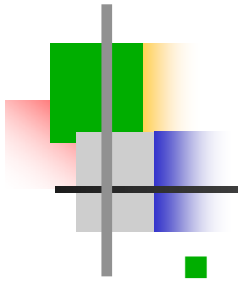
モデルに合わせて規格を作り直す

- 実施されない実施できない規格を作る
  - 無理に実施すれば仕事が増えてコスト高
  - これもISO 9000後遺症か
  - 魂が入っていない
- アセスメントのために規格を作る
  - アセスメントを考慮してダイナミックに更新されるべきプロセスを反映させない
- 規格はベストプラクティスをみんなで実施するためのもの
  - 問題は実質
  - 開発プロセスの中で生かされていることが重要
  - あるがままのスタイル、構成でよい



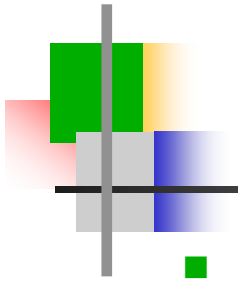
「作ってから直す」開発方式

- ソフトウェアエンジニアリングを教えていない
  - 典型的なソフトウェア組織の教育予算
    - ½は新人教育
    - ¼は管理者入門教育
    - ¼はオラクルなどのプロダクト教育
    - ソフトウェアエンジニアリング教育予算はほとんどゼロ
- プログラミングができれば一人前
  - コーディングだけでシステムができる?
  - 日本の労働市場では知っている言語で人の価値が決まる
- それが当たり前と思っているコミュニティーがある
  - オープンコミュニティーからの刺激を受けないとそうなる
- 「作ってから直す」プラクティスは高リスク低リターン
  - 「ソフトウェア開発プロフェッショナル」Steve McConnell著参照
  - コーディングは開発作業のごく一部



投資などしなくても

- 技術者がしっかりやればよい
  - 環境投資には消極的
    - 一時的なコスト増加を嫌う
    - 表面的なコスト低減のために
      - 外注の多用
      - 海外アウトソーシング
    - 結果としてはコストは下がっていない
- ソフトウェアは基本的に設備産業
  - プロセスを支援する仕組みでもっとも重要なものは開発環境



コスト削減のため安い海外へ

- プロジェクトコストだけに着目
  - 経験とスキルのある人的資源、投資してきた環境資源を遊ばせても
  - 狙ったコスト低減が必ずしも成功していない
- ソフトウェアの空洞化は早い
  - 開発作業をやらなくなれば急速に見積もり評価能力を失う
  - 遠からず開発力が相手国に奪われる
  - そうなってからあわてても遅い
- 海外アウトソースは戦略的に
  - 技術的主体性を失わない戦略
  - 資源を最大限活用して組織全体の利益増大を狙う
  - etc.