永続する品質改善へ向けて

松原 友夫
松原コンサルティング
IEEE Software諮問委員会委員
Cutter IT Journal編集委員会委員

なぜ「永続」させねばならないか

- 後戻りは最大のリワーク
  - 途中の努力はすべてムダになる
- 永続してこそ人が育つ
  - 前向きに回転する品質システムの中から優れた人が育つ
  - 人は組織文化醸成の基盤であり結果でもある
- 永続する品質基盤はプラスの波及効果を及ぼす
  - 最終的にはビジネスや企業倫理に及ぶ
- 永続が組織文化醸成の条件
  - 品質や安全は文化になってこそよい結果をもたらす
  - 底力のある組織には確固とした組織文化がある

私が言う文化とは

- 組織が共有する身に付いた思考と行動
  - 緊急事態に際してのとっさ判断に現れる
  - 問題が起きた直後のトップの反応でわかる
- 「当たり前のことがきちんと出来ていることが最も恐ろしい」
  - かつて米国の視察団が日本の自動車産業視察を視察したときの報告（HBRで読んだ）
    - 高い生産性と品質の秘密は銀の弾丸ではなかった
    - 工場内の整頓、清潔といった当たり前のことがきちんと行われていた
    - これは米国の自動車産業にとって大きな脅威だ
- 火事は家事の躾と密接に関わる
  - 消防士は「火事を起こす家の多くは乱雑だ」と言う
    - 整理整頓は心がけの問題で忙しさとは関係ない

組織文化の基盤がない品質改善は付け焼き刃

- 上からの指示に従うだけでは品質や安全性は維持できない
  - 一人一人の自覚と意識にも続く行動が支える
- モデルや規格に従うだけでは品質や安全性改善されない
  - HACCPの承認を受けていながら中毒事故を起こした食品会社
    - 規格に従ってプロセスを几帳面に記録していれば汚染範囲が特定できたはず。それをやっていなかったので判断が迷走した
  - ISO 14000の認証を受けて毒液を垂れ流していた企業
    - 規格がなくても最終廃液槽の水質チェックは当然やるべきだ
  - CMMIの取り組み組織は2つに分かれる
    - モデルをガイドラインとして使う＝組織文化醸成に向かう
    - モデルをチェックリストとして使う＝実質よりもレベル達成優先
      - Bill Curtis, Which Comes First, the Organization or It's Processes? IEEE Software, Nov/Dec 1998
- 文化基盤がなければ人や環境が変わればすぐに元の木阿弥になる

優れた組織文化の共通要素

- オープンネス
  - 職位に関係なく自由闊達な議論ができる風通しの良さ
  - 情報が上にも横にも伝わる
  - 権威風を吹かせる人がいない
- 決断と行動が速い
- 自発的に行動する気風がある
  - 進んで引き受け、自発的に提案し、自発的に学ぶ
- 閥がない、学歴差別、性差別などがない
  - 本来能力には関係ない先入観を与えるものを無視する
- 明確な責任と役割
  - これが明確なほど一旦緩急あれば協力しあう
- 旺盛な挑戦精神
  - やってみて、修正しながらものにしていく
- 一人一人を大事にする

専門技術分野にもある文化の違い

- 専門家の固有文化
  - 機械屋：現物主義、見て触って測って決断
    - 機械工場から移ってカルチャーショックを受けた
  - 電気屋：捨てることを気にしない
    - 水や空気の質に鈍感
  - 化学屋：回収再利用を常に考える
    - 水や空気の質に敏感
  - ソフト屋：立体感覚が薄い、自慢話をしない
- 多層プリント基板の初期不良多発原因は文化の違いによるものだった
  - 初期の歩留まりはわずか15%だった
  - 電気屋は空気中の埃に鈍感、洗浄水の性質にも無関心、金が多量に含まれたハンダ滓を捨てていた
  - 化学屋、物理屋を交えた対策で、短期間に歩留まりは飛躍的に向上した

ソフトにはハードと大きく異なる特性がある

- ハードウェア
  - 物理的実体が存在
  - 可視性に頼れる部分多い
  - 修正ループが短い
  - 実験により検証
  - 技術的改善に重点
  - 計測とチャートによる可視化
  - 支援環境は必須
  - 効果は数値で明確に把握
  - 機械に任せられる部分が多い
- ソフトウェア
  - 物理的実体が存在しない
  - 可視性に頼れない
  - 修正ループが長い
  - 実験はほとんど行われない
  - プロセス改善に重点
  - 計測しにくい
  - 支援環境への投資を軽視
  - 効果の正確な把握は困難
  - 人の知識スキルやる気が決め手

ソフトはハードよりやるべきことの領域が広い

- 物理原則に頼れないから
  - 原因仮説探索の範囲が広い
  - 「正しい」解決策がない
    - 有効な範囲で「正しい」
- 無形性から問題の摘出に特別な努力が要る
  - 計測値に誤差が大きい
    - 例えばプロセスの工数
  - 分類が主観的になりやすい
- 人の知識、スキル、やる気が品質に影響を与える
  - 社会学、心理学など、人間的要因の分析が重要になる
  - 方法論、プロセスなどの教育・訓練の影響が大きい
  - ドメイン知識が必要

ハードウェア生産から学べるものも多い

- 問題の攻め方
  - エンジニアリング・アプローチ
    - 現象を観察する
    - 原因仮説を立てる
    - 原因調査のための実験をする
    - 仮説に関わる要因を計測し解析する
    - 原理に基づいた解決策を積み上げる
  - 特に学ぶべきは計測値の利用法
    - 目的があって計測する
    - 現象の理解に計測値の図表現を用いる
- 立体的な問題把握
  - 重要度を識別し区分する文化がある
- 現物と現場が真実を語る、という考え方
  - 理論だけでは決断しない
    - 見て、触って、計って決断
    - そのために実験や試作を行う

現場主義、実物主義、実証主義はソフトウェアでも有効

- ソフトウェアでは現場を回る管理者は少ないが…
  - ソフトウェア開発でも現場でしか得られない情報は多い
  - 私の経験
    - 担当者の一言で「不可能な仕様」がわかり早めに対処できた
      - それはCODASILが決めた新仕様の追加だったがそれに誤りがあるのに気づかなかった
  - 問題の存在と優先順位の適切な判断、士気レベルの観察などには欠かせない
- システムが稼働する現場を観察することも重要
  - 例えば、銀行システムを設計する前に最も混雑する日と時刻に支店に行き今システムが停止したらどうなるかを想像する
- 私の信条:
  - 真理は顧客の言葉にあるのではなくシステムそのものの中にある
    - 顧客が誤ったことを言ったら訂正するのも大事な行為
    - 顧客の言いなりになることが大事にすることではない

ハードウェア生産的思考には弊害もある

- ツールの即効性を期待する傾向がある
  - 生産工場で新鋭工作機械を導入すれば比較的短期間で生産性、品質が向上するためか
  - ソフトウェアでは効果は習熟曲線に沿って緩やかに上がる
- エンジニアリング教育を軽視する傾向がある
  - 生産工場ではエンジニアリングは教えられているのが前提なためか
  - ソフトウェアでは大学でも教えられていない
- 改善目標で生産性を最重視する傾向がある
  - 有形の製品は正確に測れるためか
  - ソフトウェアの生産性はゴムのスケールで測るようなもの
    - 分子(生産量)も分母(工数)も誤差が大きいし測り方は組織毎に異なる
  - 組込みソフトではサイクルタイムの改善の方が重要
- 欠陥の作り込みを悪と見なす傾向がある
  - オシャカを作ることもあるが再製伝票を書けるくらいの比率
  - ソフトウェアの欠陥除去は選果作業に似ている

ソフトウェアで特に重視すべきこと

- 人を動かすメカニズム
  - 自発的行動を促す開発環境
  - 人をやる気にさせる制度
  - 人を巻き込む機会を作る
- 教育・訓練
  - エンジニアリング知識と経験を組み合わせた教育
  - 議論重視の教育
  - 抱えている問題に対応した解決策を助言する教育
- 見えないものを見えるようにする
  - 問題の早期摘出と修正ループ短縮のために
  - 問題の原因分析のために
- システム的思考
  - ソフトな要因を含む数多くの要因が複雑に相互に作用するため

教育は大事だと口では言うが実態は…

- 典型的な教育費予算の配分
  - ½は新人教育へ
  - ¼は管理者教育へ
  - ¼はWindows, Oracleなどの製品教育へ
  - 肝心のソフトウェア・エンジニアリングの教育は”0”に等しい
- 横行するOJTという名の放任
- 多くは座学
  - ワークショップでの議論、実習を軽視
- 経営が厳しくなれば真っ先に削られる
- 産業と大学のベクトルが合っていない
- Laszlo Beladyの提言
  - 技術移転が困難な理由の一つは「問題」と「解決策」の不一致
  - これを解決するには座学でなく直面している問題にアドバイスを与えるConsultative Trainingが有効である

広い視野で品質を捉える (1)

- 欠陥密度は狭義の品質
  - 欠陥の摘出と除去だけに注目するのは対処療法
- ISO/JISのソフトウェア品質定義
  - 機能性:Functionality 必要な機能が実装されているか
  - 信頼性:Reliability 機能が正常に動作しているか
  - 使用性:Usability 分かりやすいか、使いやすいか
  - 効率性:Efficiency 資源利用の効率
  - 保守性:Maintainablity 保守しやすいか
  - 移植性:Portability 別環境に移しやすいか
- さらに広く捉えることもできる
  - 我々は、納期通りに納めるといった、ユーザ視点の品質を含む他の品質特性について考えようとしないで、欠陥という観点から*だけ*で品質を定義する傾向がある
    - Ed Yourdon, Death March 2nd Edition
- 組込みソフトウェアではハードやドメイン固有の観点も必要

広い視野で品質を捉える (2)

- 狭義の品質特性にこだわると折角の改善が意味を失うことがある
  - e.g.欠陥はないがニーズに合わないシステム、使い勝手が悪くて売れない製品
- 顧客視点、マーケット視点の品質特性は広義の品質
  - 機能性、使用性、効率性、保守性、移植性、安全性
  - 適時性
- 各特性は相互に依存しネットワークになっている
  - 原因・結果の連鎖を遡る
    - 源流は日常の何気ない行為に遡ることができる
  - 適切なレベルのRoot Causesを狙って改善
    - ある問題を解決すればどの問題が自然に解決するか？
    - 針灸治療、ツボ刺激の原理
  - 清潔、整理整頓も源流指標の一つ
    - 東芝の土光さんが指でほこりをすくって注意したのは理にかなっている

文化を醸成するメカニズム

- 望ましい組織文化は説教からは生まれない
  - 命令や説教は人を受け身にするだけ
  - 企業文化の改革をトップの決断で成功した例はある
    - 新パラダイム賛同者の比率が閾値を超えていた
- 常に刺激を与えるメカニズムを作る
  - 改善は性善説を前提とする
    - 人は自らの位置を知り改善のやりかたを知れば自発的に改善に向けて動き出すもの
    - 位置と経過を示すメカニズムの中に人を置く
  - メカニズムを作った人がいなくてもそれは回り続ける
  - 強制されるのでなく自発的に動くので永続する
    - 最近は受け身人間が多いので参画意識の醸成から始める必要があるだろう

システム的に考える (1)

- システムを開発するシステムは極めて複雑
  - バグを追うだけでは品質レベルは上がらない
  - 要因のネットワーク内での相互作用を理解する
- 異質な要素を数多く含むシステムの開発では必須
  - 予想外のトラブルの解決
    - システム的思考のよい訓練になる
  - 例えば処理能力不足は古くて新しいシステム問題
    - 最近の例では東証システム
  - かつてはシステム的問題を扱う部門があった
    - 私はその一つである方式設計部に所属して銀行システムを担当
    - そこは大規模で難しいシステムの設計と開発を担当した
    - この部門はかつてはシステム屋の養成所であった
    - 現在は消滅

システム的に考える (2)

- システム的思考ができる人をどう育てるかが問題
  - 最近の教育では技術間口の狭い専門家の養成に力を入れる傾向が強い
  - システム設計を学ぶ場はリストラで減る一方
  - 座学ではだめ、複数技術分野での経験が必要
  - 計画的キャリアパスで育てるのがよいのだが実施例は稀
  - システム屋の需要は増える一方
    - ミッションクリティカルシステムの増加
    - システム事故の増加
    - システムLSIの開発、など

システム的に考える (3)

- 盲点を突かれた実例
  - 私が経験したあわや迷宮入りのトラブル
    - 銀行の預金システムで口座の記録が存在するのに時々「口座が存在しない」というエラーメッセージが出た
    - 次々と違う負荷テストを実施したが原因がわからず迷宮入りかと思われた
    - あるとき、それが朝早い時間に起こることに気づき、原因がディスクのアームを動かす油圧ノズルの泡であることを突き止めた
    - 負荷テストは油の温度を上げたので油の温度が低いときに出る現象には無効であった
    - この問題の解決には、OS、アプリケーション、ディスクなど、多くの技術者が関わった
- 難しい問題は技術分野の境界領域で起きることが多い

ソフトウェアは設備産業

- 人、環境、ツールへの適切な投資が品質を高める
  - しかし日本では
    - 未だに多い人海戦術
    - 増え続ける派遣依存
  - 投資のコミットメントを引き出すには
    - ソフトウェアはトップと現場の認識乖離が大きい
    - 改善のための投資を提案する
      - コミットメントが得られない理由の多くは提案していないからだ
      - なにをすればよいかが分かっているのは責方だけだ
        - トップの理解程度のリトマス試験紙になる
        - 理解が得られたら一気呵成に進める
    - 機会損失見積もり資料の蓄積が不可欠
      - 実際の損失額を基礎とする
      - 大損害を起こしても再発防止の投資をする組織は少ない

ソフトウェア安全性

- ソフトウェアも安全にかかわる
  - 殺人放射線治療器Therac-25
  - Ariane-5の打ち上げ失敗---爆破
  - ニューヨークでの電話網9時間停止
  - 原子炉設計計算で安全係数のサイン誤り
    - 1ビットの誤りでも重大事故につながりかねない
- 信頼性とは異なる
  - 一部ではあるが全体ではない
  - コンポーネントに欠陥があっても安全を損なわない
- 組織文化とシステム的思考は必須
  - 尼崎の事故対応で安全文化が確立していないことが露呈した
- 組込み製品の設計者の多くが誤解しているソフトウェア安全性の神話を次に紹介

Nancy Levesonのソフトウェア7つの神話

神話1:コンピュータのコストはアナログまたは電気機械装置よりも安い
信頼性の高いソフトを書き保守するコストは装置よりはるかに高い
神話2:ソフトウェアは変更しやすい
エラーを導入せずに変更するのは難しい
神話3:コンピュータは電気機械的装置よりずっと信頼性が高い
時間とコストのプレッシャーは完全な設計を妨げる – Therac-25の例
神話4:信頼性が高まれば安全性も高まる
信頼性は要件にたいするものだが安全性の問題はしばしば要件の誤りによる
これを誤解している人は多い
神話5:テストまたは正しさの「証明」ですべての誤りを除去できる
そんな技法はまだ存在しない
神話6:ソフトウェア再利用は安全性を高める
利用環境が変われば問題を起こす
神話7:コンピュータは機械的システムよりリスクが少ない
リスクを減らす可能性はあるが保証できない

Safeware: System Safety and Computers, Nancy Leveson, Addison Wesley

最後に再び文化について

- 文化を担うのは組織にいるすべての人
  - 個人個人も役割を持つ
- 安全問題は組織文化を思い起こすよい機会
  - 事故が起こったときの個人のとっさの行動
  - 各階層の人の行動から文化が透けて見える
  - 最初に手を付けなければならないのは安全文化の確立なのだが
    - それに気づいているか？
    - 下からの盛り上がりがあるか？
    - 風通しはよくなりつつあるか？
- 「日常身を置く場が人格を形成する」