ソフトウェア品質保証の肝

構想２年、製作４年の超大作を公開！

2015年 11月 9日

SQiPソフトウェア品質保証部長の会
グループ３

メンバー　(五十音順)
池上　直之　AJS株式会社
稲冨　秀人　株式会社FAITEC
鎌倉　洋一　株式会社富士通ミッションクリティカルシステムズ
桑原　秀昌　TISシステムサービス株式会社
佐藤　孝司　日本電気株式会社
早崎　伸二　株式会社リンクレア
藤川　昌彦　アズビル株式会社

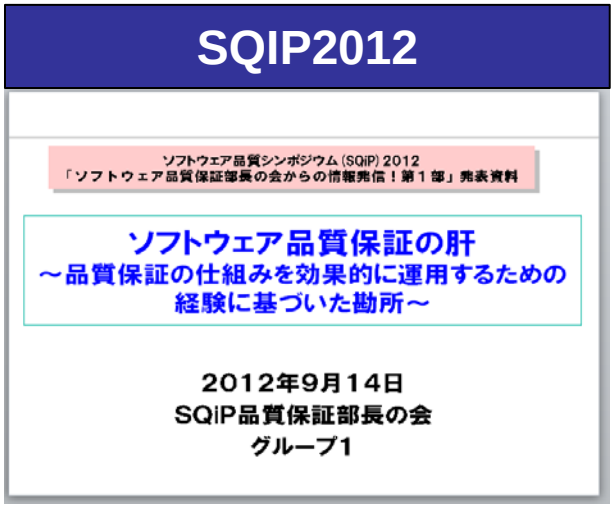
目　次

- これまでの活動の振り返り
- 今年度の活動計画と実施内容
- 成果物（一部）のご紹介
- 残された課題
- 最後に（まとめ）

これまでの 「先ずは、肝を集めてみよう」

【第３期】 品質保証の肝 Part Ⅰ

～品質保証の仕組みを効果的に運用するための経験に基づいた勘所～



- **品質保証に関する悩みを抽出**
  →広範囲に、たくさん出てきた
  →“この際、全部、整理してみよう”
- **１件毎に悩みの内容と、各メンバの経験をもとに肝は何かを検討**
- **グループ１以外のメンバからも、幅広く肝を収集**
- **「ソフトウェア品質保証の肝」として検討結果をまとめた**

“ソフトウェア品質保証の肝”とは・・・

- **“品質保証プロセス/仕組みはわかっているが、現場でうまく運用できない”**
  “わかっちゃいるけど、うまくいかない”!
- **品質保証部門長の“悩みが尽きない”!**
- **一方、これらの悩みを豊富な経験から解決してきた事例もたくさん持っている**
  ⇒これらの悩みと解決ノウハウを収集し“肝”として整理した

これまでの活動の振り返り（２）

初期の肝の紹介

3.見積もり①

【肝】見積もりの評価はベテランやデルファイ法を使う

【肝の説明】

- **見積もりが適切かを判断するのは非常に難しい**
  - **見積もりの経験豊富なベテランに審査してもらう**
  - **複数人で見積もられた合意（デルファイ法）を使う**
- **見積りは、金額を見積もる前に工数を見積もる。その工数に単価をかけて金額とする**
- **見積りは、工期も見積りが必要である。工数と工期には関係がある。（COCOMOⅡ、JUASによる工期モデルを参考にする。ただし、プロジェクトが大きい場合には合わない）**
- **新規開発と改造では、見積もりの基準値は異なる**

これまでの「品質保証部門のミッション再確認」

【第４期】 品質保証の肝 Part Ⅱ

～もう品質保証業務で悩むのは止めよう～

SQIP2013

第4回 品質保証部長の会 成果発表会

ソフトウェア品質保証の肝Part II

～もう品質保証業務で悩むのは止めよう～

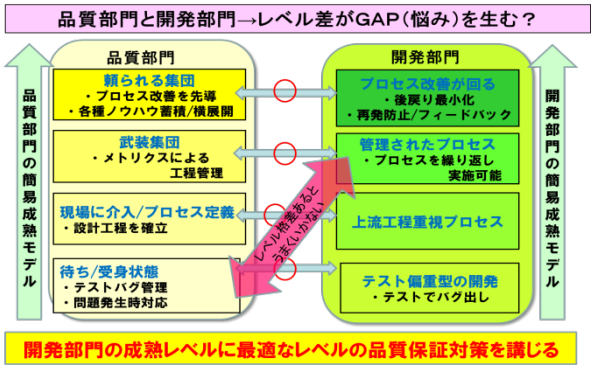
2013年11月29日

SQiP品質保証部長の会

第4期 第1グループ

グループメンバー
- 日本電気株式会社 佐藤 孝司
- アズビル株式会社 藤川 昌彦
- 株式会社日立ハイテクソリューションズ 岡本 卓
- 株式会社FAITEC 稲富 秀人
- 株式会社富士通ミッションクリティカルシステムズ 鎌倉 洋一

5．品質部門はどうあるべきか（１）



悩みの構造解析
（なぜ、悩みが生まれる）
→お客さまに対する視点が、開発部門と品質部門で、ずれることがある

- **開発部門は、短期的なCとDの成果達成で精一杯**
- **品質部門は、常にお客様の視点であるべき、品質は長期的視点では最重要、品質積上げがお客さまとの信頼関係の基盤を築く。（１件のバグが簡単に信頼を無くすと考える！！）**

①開発部門と品質部門のゴールの共有、双方の立場の相互理解が必須
②肝が必要になる理由
- **開発部門と品質部門が相互に期待することが異なっている**
- **部分最適と全体最適のサイクルにずれが生じる**

第４期からの新たな試み
日本文化の知恵を活用して一文での本質表現に挑戦！

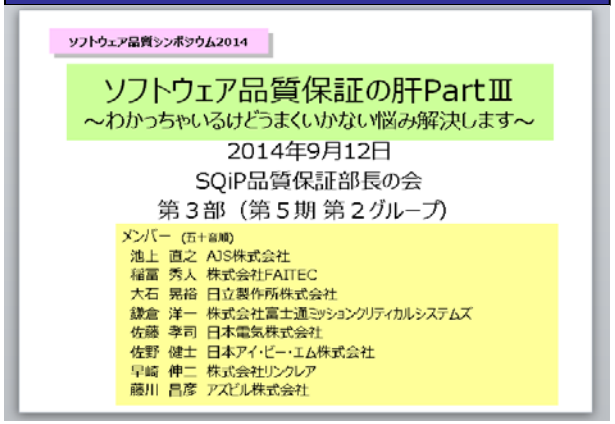
・「あ」
　－ありえない、上から目線のアドバイス
・「い」
　－いけません、後だしじゃんけんと倍返し
・「う」
　－うれしいな、今日もトラブルコールなし

これまでの「肝の体系化」

【第５期】 品質保証の肝 PartⅢ

～わかっちゃいるけどうまくいかない悩み解決します～

SQIP2014



第１章 「ソフトウェア品質保証の肝」とは

- “品質保証プロセス/仕組みはわかっているが、現場でうまく運用できない”
  - 品質保証部門長の“悩みが尽きない”!
  - “わかっちゃいるけど、うまくいかない”!

◆一方、これらの悩みを、豊富な経験から解決してきた事例も、たくさん持っている

⇒これらの悩みと解決ノウハウを収集し、“肝”として整理した

- 品質保証活動の肝となる考え方を掘り起し、整理し、分かり易く伝える資料としてまとめた。
- 品質保証業務経験者の生の声であることにこだわった。
- 品質保証活動の悩みを解決する為の肝を伝えることで品質活動の初級者の育成に役立つ内容を目指す。
- 品質保証部門に限らず、プロジェクトリーダーの立場から品質保証を考える人にも役立つ内容とする。

- 理屈を表面的に現場に持ち込んでも形骸化するだけ。
- 肝は、理屈ではなく、現場感覚で矛盾する事象の本質を捉えることで導き出せる。肝の意見が異なるということはそれぞれの事象の中に潜む肝を分けて考えていくことが必要となる。
- **85個の肝を143ページにまとめた。**

これまでの活動の振り返り（６）

2.4 監査のマネジメント①

P D C A PM QA

【肝009】監査を技術伝承の場にする

【背景】

- 「監査」は、監査員も、監査される側のイメージも、上から目線になりがちである。

【解決のヒント】

- 監査体制の提案
  - ①嫌われ役の監査員、②監査される側の助け舟を出す監査員で構成する
  - 嫌われ役の監査員は、あえて「嫌われ」役として振る舞う。監査される側の助け舟を出す監査員は、嫌われ役の指摘を緩和することでネガティブな場の雰囲気をポジティブに変える役割をする。
  - ただし、嫌われ役は本人の適性もあることも配慮する。
- 監査される側と監査する側が長い年月をかけて、仕事を続けていく同士であるから、お互いに緊張感を維持しつつ、良い関係を築いていければ良い、と思う。

これまでの活動の振り返り（７）

品証カルタ①

い	• いけません　後出しじゃんけん　倍返し
ろ	• ロジックの　シンプル故に　バグ無しに
は	• パレート図　眺めて対策　優先度 • はずれ値に　真の課題が　隠れてる
に	• 日本式　体力勝負じゃ　もう限界 • 日本式　こだわり品質　ガラパゴス
ほ	• 本当の　品質良くする　現場の声 • 本当の　品質良くする　トップの背
へ	• 変更に　良かれと応え　悪化させ
と	• 取り返せ　なぜあの品質で　出荷した

これまでの「肝のブラッシュアップ」

【第６期】 品質保証の肝 （完結編）

第６期の活動方針

- **肝１件ずつを深く議論して全員が納得出来る内容に仕上げる**
- **肝の表現、解説、書式など全面的に見直す**
- **成果物は知識体系（※BOK）として完成させる**
  - **※KIMOBOK（Guide to the Kilo-Important Body Of Knowledge）**
  - **副題：“品証部の煩悩を取り払う108つの肝”**
- **SQuBOK Ver2の樹形図で分類を見直す**
- **１肝２頁で構成し、１肝に対して１枚の詳細解説･具体例の追加を試みる**
  - **第５期迄の成果発表会の発表事例を引用する**

成果物（一部）のご紹介

「品証部の　煩悩を取り払う　108つの肝」

SQuBOK 体系 2. ソフトウェア品質マネジメント　目次	肝の個数
2.1 ソフトウェア品質マネジメントシステムの構築と運用	9
2.2 ライフサイクルプロセスのマネジメント	5
2.3 ソフトウェアプロセス改善のマネジメント	4
2.4 検査のマネジメント	2
2.5 監査のマネジメント	2
2.6 教育・育成のマネジメント	8
2.7 法的権利・法的責任のマネジメント	1
2.8 意思決定のマネジメント	1
2.9 調達のマネジメント	2
2.10 リスクマネジメント	3
2.11 構成管理	3
2.12 プロジェクトマネジメント	7

SQuBOK 体系 2. ソフトウェア品質マネジメント　目次	肝の個数
2.13 品質計画のマネジメント	4
2.14 要求分析のマネジメント	6
2.15 設計のマネジメント	5
2.16 実装のマネジメント	2
2.17 レビューのマネジメント	9
2.18 テストのマネジメント	3
2.19 品質分析・評価のマネジメント	8
2.20 リリース可否判定	2
2.21 運用のマネジメント	8
2.22 保守のマネジメント	1
その他	0
総　　計	９５

参考

SQuBOK®樹形図 （Guide to the Software Quality Body Of Knowledge V2）

1. ソフトウェア品質の基本概念

1.1 品質の概念
- 1.1.1 品質の定義
- 1.1.2 ソフトウェア品質モデル
- 1.1.3 ディペンダビリティ
- 1.1.4 使用性
- 1.1.5 セーフティ
- 1.1.6 セキュリティ

1.2 品質マネジメントの概念
- 1.2.1 品質保証の考え方
- 1.2.2 改善の考え方

1.3 SWの品質マネジメントの特徴
- 1.3.1 プロダクト品質とプロセス品質
- 1.3.2 品質作り込み技術の考え方
- 1.3.3 システム及びSW測定の考え方
- 1.3.4 システム及びSW評価の考え方
- 1.3.5 V&V(Verification & Validation)
- 1.3.6 日本におけるSW品質保証

2. ソフトウェア品質マネジメント

組織レベル
- 2.1 SW品質マネジメントシステムの構築と運用
- 2.2 ライフサイクルプロセスのマネジメント
- 2.3 SWプロセス改善のマネジメント
- 2.4 検査のマネジメント
- 2.5 監査のマネジメント
- 2.6 教育・育成のマネジメント
- 2.7 法的権利・法的責任のマネジメント

プロジェクト共通レベル
- 2.8 意思決定のマネジメント
- 2.9 調達マネジメント
- 2.10 リスクマネジメント
- 2.11 構成管理
- 2.12 プロジェクトマネジメント

プロジェクト個別レベル
- 2.13 品質計画のマネジメント
- 2.14 要求分析のマネジメント
- 2.15 設計のマネジメント
- 2.16 実装のマネジメント
- 2.17 レビューのマネジメント
- 2.18 テストのマネジメント
- 2.19 品質分析・評価のマネジメント
- 2.20 リリース可否判定
- 2.21 運用のマネジメント
- 2.22 保守のマネジメント

3. ソフトウェア品質技術

工程に共通なSW品質技術
- 3.1 メトリクス
- 3.2 モデル化の技法
- 3.3 形式手法

工程に個別なSW品質技術
- 3.4 品質計画の技法
- 3.5 要求分析の技法
- 3.6 設計の技法
- 3.7 実装の技法
- 3.8 レビューの技法
- 3.9 テストの技法
- 3.10 品質分析・評価の技法
- 3.11 運用の技法
- 3.12 保守の技法

専門的品質特性のSW品質技術
- 3.13 使用性の技法
- 3.14 セーフティの技法
- 3.15 セキュリティの技法

2章の分類に沿って整理

参考

2. ソフトウェア品質マネジメント

組織レベル

2.1 SW品質マネジメントシステムの構築と運用

1 品質マネジメントシステム
- ①ISO 9000シリーズ
- ②TQC（統合的品質管理）
- ③TQM（統合的品質マネジメント）
- ④QMS持続的成功の秘訣

2 セキュリティのマネジメント
- ①コモンクライテリア
- ②脆弱性及び脆弱性管理
- ③ISMS（情報セキュリティMS）

3 ソフトウェア品質推進活動
- ①シックスシグマ
- ②QCサークル
- ③SWQC【NEC】
- ④Qfinity【富士通】
- ⑤品質会計【NEC】

2.2 ライフサイクルプロセスのマネジメント

1 ライフサイクルモデル
- ①ISO/IEC 12207
- ②ISO/IEC 15288

2 セーフティ・クリティカル・ライフサイクルモデル
- ①機能安全
- ②自動車電子制御の機能安全
- ③医療機器SWライフサイクルプロセス

3 プロセスモデル
- ①ウォーターフォールモデル
- ②反復型開発プロセス
- ③プロトタイピング
- ④スパイラルモデル
- ⑤アジャイル開発
- ⑥プロダクトライン開発
- ⑦派生開発（XDDP）

2.3 SWプロセス改善のマネジメント

1 SWプロセス能力改善のためのプロセスモデル
- ①CMMI（能力成熟度モデル統合）
- ②PSP（パーソナルSWプロセス）
- ③TSP（チームSWプロセス）
- ④TPI（テストプロセス改善）
- ⑤TMMI（テスト成熟度モデル統合）

2 SWプロセス能力改善のためのマネジメント技法
- ①ISO/IEC 15504
- ②IDEAL
- ③ポストモーテム
- ④落穂拾い【日立】
- ⑤なぜなぜ分析
- ⑥三階層SEPG【東芝】

2.4 検査のマネジメント

2.5 監査のマネジメント

1 購買先プロセス監査
2 SW開発における監査

2.6 教育・育成のマネジメント

1 スキル標準
- ①ITSS（ITスキル標準）
- ②ETSS、③UISS、④CCSF

2 教育・育成のマネジメント技法
- ①キャリア開発計画
- ②動機づけ、③PS（パートナー満足）

2.7 法的権利・法的責任のマネジメント

1 知的財産権の法的権利・法的責任のマネジメント
- ①特許法、②著作権法、③OSSライセンス

2 知的財産権以外の法的権利・法的責任のマネジメント
- ①不正アクセス禁止法、②個人情報保護法
- ③PL法（製造物責任法）

参考

2. ソフトウェア品質マネジメント

プロジェクト共通レベル

2.8 意思決定のマネジメント
1 IPD(統合製品開発)【IBM】

2.9 調達マネジメント
1 請負契約による外部委託
2 オフシェア開発
3 ブリッジSE

2.10 リスクマネジメント
1 リスクマネジメントに関する規格
2 システム及びSW保証に関する規格
3 リスク識別
4 リスク分析

2.11 構成管理
1 変更管理
2 バージョン管理
3 不具合管理
4 トレーサビリティ管理

2.12 プロジェクトネジメント
1 プロジェクトマネジメント体系
①PMBOK
(プロジェクトマネジメント知識体系)
②P2M
(プロジェクト＆プログラムマネジメント)
③プロジェクトにおける
品質マネジメントの指針に関する規格
④プロジェクト計画に関する規格
2 プロセス設計におけるテーラリング

プロジェクト個別レベル

2.13 品質計画のマネジメント
1 ベンチマーキング

2.14 要求分析のマネジメント
1 要求分析の計画
2 要求の妥当性確認と評価

2.15 設計のマネジメント
1 設計の計画
2 設計方針の決定
3 設計の評価

2.16 実装のマネジメント
1 実装の計画
2 実装方針の決定
3 実装の評価

2.17 レビューのマネジメント

2.18 テストのマネジメント
1 テストドキュメントに関する規格
2 テストの組織
3 テストレベル
4 V字モデル
5 W字モデル
6 テスト計画
7 テストリスクマネジメント
8 テスト進捗マネジメント
9 テスト環境マネジメント
10 テストに関する規格

2.19 品質分析・評価のマネジメント
1 プロダクト品質の分析・評価
2 プロセス品質の分析・評価

2.20 リリース可否判定

2.21 運用のマネジメント
1 ITIL
2 サービスマネジメントに関する規格
3 SLM(サービスレベルマネジメント)
4 SLA(サービスレベルアグリーメント)
5 サービスの継続性マネジメント
6 サービスの可用性マネジメント
7 インシデントマネジメント
8 問題マネジメント
9 リリースマネジメント
10 キャパシティマネジメント

2.22 保守のマネジメント
1 保守に関する規約

KIMOBOK（Guide to the Kilo-Important Body Of Knowledge）

SQuBOK Guide V2
樹形図　知識領域

役立つステージ
（P･D･C･A）

役立つ立場
（PM･QA）

2.1 ソフトウェア品質マネジメントシステムの構築と運用①

P D C A PM QA

【肝001】QMS構築は組織の責任分担を明確にする

【背景／悩み】

- 新しく品質保証機能（QMS：Quality Management System）を立ち上げるためにはどのように考えればよいのか。

【肝の説明／解決のヒント】

- SEPG（Software Engineering Process Group）、SQAG（Software Quality Assurance Group）、PMO（Project Management Office）などの組織と実際のプロジェクト間でフィードバックし合える役割分担を設定する。

格言　プロジェクト　組織と連携　機能する　6

【肝002】品質保証部門のミッションを明確にして「ブレない」こと

【背景／悩み】

- 品証部門はISO9001、トラブル対応、品質データ分析など、トップから矢継ぎ早に業務を命じられる。
- やるべきことが多すぎることもあり、何から手を付けて良いのかが分からなくなる。

【肝の説明／解決のヒント】

- 品証部門の業務は多岐に渡り、組織の中の位置づけも各々のケースで決める必要があるため、トップを含めた組織的な合意が必須。
- ミッションは、「製品が品質要求事項を満たして必要十分な情報を提供できる状態であることを、事実として証明できる体系的な活動を継続すること」。
- 基本的には組織の品質目標を策定し目標達成をミッションとすべき。
- 業務の方向性は以下の２種類がある。（障害対応系は別として）
  ① 品質管理や出荷判定を主導し“出荷品質”を確保すること
  ② 品質の良い開発をする為のプロセス改善を主導し“強い開発組織”を作ること

【肝013】現場の開発プロセス把握が品質分析の前提

【背景／悩み】

- プロジェクトの品質データを横並びで見たり、過去の実績と比較したりすると、データのブレ幅が大きいことがある。なぜだろう？

【肝の説明／解決のヒント】

- あらかじめ、プロジェクトが採用している開発プロセスを知ったうえで品質分析をしないと、ミスリードしてしまう。
  - ① プログラム設計書が妙に少ない。
    ツールを使った開発が大半で、プログラム設計書を書かない。
  - ② ソースレビューの指摘件数が妙に少ない。
    ソースレビューは単体テスト後にやるリファクタリングが中心。（テスト駆動開発のため）
  - ③ テスト件数がやたら多い。
    テストが自動化されていて、過去から累積した全項目を実施している。

【肝015】本番障害やトラブルでの失敗を改善に繋げる仕組みを作る

【背景／悩み】

- 本番障害報告書は作成されて報告されているが、プロジェクト関係者にしか理解できない内容になっており、組織内で共有できていない。
- トラブル報告書は始末書的な内容となっており、プロセス改善のために活用されていない。

【肝の説明／解決のヒント】

- 障害発生時には、直接原因の是正(バグFIX、設計書修正他）だけではなく、なぜその欠陥を作り込んだかの根本原因を究明し、プロセス改善（未然防止）に繋げる。
- トラブルが発生した場合には個人の問題として扱うのではなく、組織・仕組みに問題がないかを確認し改善する。
- 100ページあるチェックリストは、単に増やすだけでは何の役にもたたない。
  関係するチェックポイントを簡単に抽出できる仕組みを作る。

【肝019】工程検査で品質の実態を現場に示す

【背景／悩み】

- レビューで指摘をしても、顕在化した障害ではないので開発部門に受け入れてもらえないことがある。

【肝の説明／解決のヒント】

- レビュー記録票や故障票の分析で品質の課題を指摘しても状況証拠なので、品質の良否について説得力に欠ける場合がある。出荷検査でバグや仕様書の誤りを見つけてこそ、開発者に対する影響力が大きい。
  ① 上流工程のレビュー指摘に対し、開発部門は「レビューでの指摘は障害ではない」という認識が強いので、上流での指摘に対して危機感が薄い。
  放置するとどのような事態に陥るかの認識を共有させることが有効である。
  ② よって、工程検査で「要求」や「仕様」の不適合を開発部門に具体的に示すことで、どれほど手戻りが多いかを認識してもらうことが有効。
  ③ レビューの意識が薄くても、検査で検出された不適合に対しては非常に関心を示す。

【肝022】ISO9001は「気づきを得るツール」として使う

【背景／悩み】

- 開発プロセスに関する規程を制定しているが、制定以来改訂されていない。
- 監査のために相当の時間をかけて準備しているが、時間の無駄ではないか？

【肝の説明／解決のヒント】

- ネガティブではなく品質向上プロセスととらえ、プロセスに息を吹き込む。
  - ① 形骸化しているプロセスは大胆に除外することを検討する。
    QCDに対して意味のないプロセスならば不要。
  - ② 不適合の指摘はプロセスの弱点を考える良い機会と考える。
  - ③ 品質目標未達成の原因、バグ分析、バグ傾向分析など、改善の機会を組み入れてみる。
  - ④ 自ら積極的に探し出した改善活動こそQMS（Quality Management System）活動。
  - ⑤ ISO9001の不適合を積極的に受け入れること。指摘を受けると作業が増えるので執拗に抵抗するのは本末転倒。

【肝027】品質改善推進者は実践の中で育成する

【背景／悩み】

- CMMI（Capability Maturity Model Integration）、ISOを勉強しているだけでは人財が育たない。
- 過去の経験や技術の伝承がうまくできていない。

【肝の説明／解決のヒント】

- 改善活動は、活動する毎に振り返りを行う。何が良かったのか？悪かったのか？
- レビューの場は育成の機会と捉える。
  マネジメントレビューの場は、経験を伝承する絶好の場である。
- 組織の支援体制も重要である。
  ① 有識者は、原理原則の理解と実践の場（OJT：On-the-Job Training）の繰り返しによって育つ。
  ② 外部（日科技連）に出て刺激を受けることも重要。

【肝031】現場が認識を持たないといけない法規制を明示しておく

【背景／悩み】

- 開発中に権利侵害やOSS（Open Source Software）ライセンス違反が発覚すると、手戻りが大きい。
- 法律に引っかかることを知らずに不法行為を行ってしまう。

【肝の説明／解決のヒント】

- 会社のルールだけに頼るのではなく、プロジェクトの計画段階で、チェックするルールを明文化しておく。
  品証部門の担当範囲でない場合は、担当部署と連携してルールを作る。
- 受託開発の場合は、OSSの使用を認めないお客様もいるので、要件定義の時点で、お客様に確認しておく。

【肝032】意思決定に正解はない。決定しないことによるロス予防を優先する

【背景／悩み】

- 現在のプロセスで完了させるべき未決事項が放置されたままプロジェクトが進んでいる。
  結果として、大きな手戻りが発生してしまう。
- 下記のような現象を見ても指摘をしない決裁者も問題。
  工程会議で進捗報告しない、マイルストンの記載なし、自社のスケジュールのみ

【肝の説明／解決のヒント】

- 今決めるべきことか先送りしても問題ないかを判断して決める。
- 意思決定の判断材料が全て揃っていなくても、決定しないことによるマイナス面を天秤にかけて、自分の判断を信じてやり通す。
  判断材料が全部揃ってから意思決定をしたほうが楽であり、正解に近づくかもしれないが、それでは間に合わない場合が多い。
- 要フォローアップ事項を随時更新する形で管理する。
  この時、期限を日付で設定できなければ、例えば「単体テストの完了まで」といった形で定義させる。

【肝034】オフショア開発は、成熟度や文化が国内と異なることを考慮する

【背景／悩み】

- 国内の協力会社と同じような内容（契約書・発注書）でオフショア開発を実施したが、受け入れ時に不適合が多発してスケジュールが大幅に遅延した。
- 発注元の意図が正確に伝わらず、また、受入れ時のレビュー・テストで指摘・バグが多く発生する。

【肝の説明／解決のヒント】

- 発注する際のドキュメントは、曖昧さがなく、行間を読まなくても、伝わるような記載であるか確認する。（日本語チェックツールを活用）
  ① 以上、以下ではなく「≦」「≧」で表現する。
  ② 文章ではなく「○」「×」で表記してみる。
  ③ 複雑なロジックはディシジョンテーブルを使う。
- 国内の開発に比べ、指標値を1.5～2倍に設定し、２回目の発注であっても、慣れたからといって指標値を変えない。（人員流動で習熟効果が期待できない）
- コストが安い国ほど成熟度は一般に低いので発注元工数を想定しておく。
- 実際にオフショア先の開発現場に出向き、実態を確認をする。

【肝035】リスクは特定することで対策を容易に導くことが可能となる

【背景／悩み】

- リスクを洗い出したが、対策が立案できない。
- リスクとして挙げたものに課題とすべきことが混じっている。
- プロジェクトを進めている中で、様々な事象が発生して対処に苦労する。
  いくつかの事象は顕在化した時点では手遅れとなってしまう。

【肝の説明／解決のヒント】

- リスクを洗い出す際には、リスクが顕在化する恐れのある時期（工程）と、顕在化した際の影響（インパクト）を明確にして初めてリスクを特定したことになる。
  同じリスクでも顕在化する工程によって影響が異なる場合がある。
- 影響（インパクト）を明確にすると、おのずと対策が見えてくる。
- リスクの洗い出しの十分性についてはPJメンバだけではなく、有識者を入れて検証することが大切。

【肝038】トレーサビリティの確保はＶ字の両端から始める

【背景／悩み】

- 開発規模が大きくなると、トレーサビリティ負荷は爆発的に増える。
- 有限な資源で、どこまで実現できるかは、いつも悩ましい。

【肝の説明／解決のヒント】

- 各工程の入力ドキュメントと出力ドキュメントの全てを紐付けすることが出来れば、トレーサビリティが確保できて品質と保守性は向上するが、工数は増加する。
- 最低限必要なトレーサビリティは、要件 →（最終工程の）テスト項目である。なので、リソースが十分でない場合は、Ｖ字の両端のみを管理して、途中のトレーサビリティは割愛するやり方で良いと考える。
- 但し、非機能要件は許容値を文書化し、機能要件へ紐付することが必須である。
- 無理してトレーサビリティを追求して、途中で力尽きてしまっては意味がない。「頭と尻尾を繋げる」最初はこれだけで良い。これが出来てから、次の段階として途中のドキュメントを紐付けていけば良い。

【肝046】まずい進捗報告は課題抽出のネタになる！

【背景／悩み】

- プロジェクトの進捗報告が毎月同じような内容となっている。
- プロジェクトの管理項目が3か月以上の管理単位となっている。
  （実装：08/01~11/30、テスト：12/01~03/31）

【肝の説明／解決のヒント】

- WBS（Work Breakdown Structure）から導く「課題管理項目」
  ① WBSの最小単位が長すぎて、適切な進捗管理ができない。
  ② 進捗の基準がなく、報告が個人の主観になっている。
  ③ WBSの管理単位（ワークパッケージ）で予定工数が記載されていないので、WBS自体の妥当性が分からない。
- 報告内容から導く「課題管理項目」
  ① 遅れの報告時に対策を提示してこないので、進捗会議の時間が長引く。
  ② 予定が頻繁に書き直されるので、当初予定に対する遅れが分からない。

【肝049】品質目標を決めると問題が見えてくる

【背景／悩み】

- 品質目標をどのように設定して良いのかわからない。
  （何を基準に設定して良いのかわからない）

【肝の説明／解決のヒント】

- 問題とは、目標と現状とのギャップである。（目標と現状との違いを分析、現物を確認して次の目標を具体的に設定することが大切）
- 目標値は、当該プロジェクトや類似プロジェクトの過去の実績値をもとに設定するとよい。
  ① 実績値が無い場合には、当該部門や会社の実績値を参考にする。
  ② それも無い場合には、世間一般（IPA：情報処理推進機構のサイト等）から引用した値を参考にする。
- 目標値は、開発部門が自ら決めることが望ましい。自力で決められない場合も多いが、上記を参考に手助けすれば大抵は決められる。

【肝053】非機能要件の許容値は確定させる時期を合意する

【背景／悩み】

- 性能や信頼性の要求がはっきりしないまま基本設計以降の開発を進めた結果、検収時点でクレームとなり大幅な手戻りが発生した。
- 顧客から提示された非機能要件（性能・信頼性等）を要件定義の時点で、あいまいにしたまま進め、性能改善コストが大幅に掛かった。

【肝の説明／解決のヒント】

- 要件定義時点では実現可否を確認していない約束をしてはいけない。
  必要なタスク（プロトタイピング等）を計画に盛り込み、妥当な許容値が明確に出来る時期を約束し、課題管理の対象として未完であることを明示する。
- 非機能要件に対して要求を出すステークホルダの特定（限定）を要件定義時点で行い、他のステークホルダからの要求はスコープ外であることを合意しておく。
  （ユーザテストや受入れテストに入ってから新しい要求が出ないようにする）
- 外部（インターネット）に接続される場合のセキュリティは、明示されなくても堅牢性は要求され、責任を追及される。
  最新情報（IPA：情報処理推進機構提供）を元に実装する。
- Fault（システムダウン等）は製造責任の範疇となる。信頼性設計は必須！

【肝060】設計では、障害時の対策や運用を漏らさないこと

【背景／悩み】

- プロジェクトによって作成される仕様書／設計書の記載レベルがまちまち。
- 仕様書／設計書のレビューの観点が人によって大きく異なる場合がある。

【肝の説明／解決のヒント】

- 設計書のテンプレートにあらかじめ記載必要な項目を設定しておき、必ずテンプレートを使わせる。（目次を決めておくだけも効果的）
- 設計審査のチェックリストに運用設計を考慮した項目を記載し、設計審査時に使用してもらう。
- 品質特性等を利用してチェックすると効果的。チェックリストを作らず仕様／設計のテンプレート自体に品質特性を盛り込んだ項目を追加しておくと、書き忘れを防止する効果がある。（障害耐性は機能性や信頼性、運用は使用性で設計する）
- テスト仕様書フォームは事前に準備しておき、設計段階で気づいた点はテスト仕様書に書き込んでいく。

【肝064】コード変更による影響範囲を見誤らないコツ

【背景／悩み】

- コードの変更による影響範囲を見極めることができない。
  そのため、変更するたびに、すべてのコードの回帰試験が必要になってしまう。

【肝の説明／解決のヒント】

- 構造解析ツールを利用してシンプルな構造にしておけば、比較的コード変更による影響範囲を正確に見極めることができる。
- 機能毎にテスト項目のブロック化を行なっておく。
- 最初のうちは、全てのブロックを実施するが、何回か実施して修正と影響のあるブロックの関係が見えてきたら、修正に対応した実施すべきブロックが見えてくる。
  そうすれば、テストすべきブロックの強弱がつけられるようになる。
  （ある場合は影響のありそうなブロックのみ実施。ある場合は全ブロックを実施）

格言 **構造を　しっかり押さえて　変更を**

【肝073】レビュアーには心得が必要

【背景／悩み】

- 発言が「質問」ではなく「詰問」という雰囲気になってしまう。
- レビューの場での発言者が限られた人になってしまう。

【肝の説明／解決のヒント】

- レビューは、「活発な議論がしやすい雰囲気」を築くことにより、「リスクを検知しやすい環境」の形成に繋がる。
- レビュアーは単に質問を投げかけるだけでなく、「隠されたリスク」を顕在化させる問い掛けやファシリテーションの工夫を図ることが求められる。
  - 心得の七柱 –

    「サービス業」の意識、「レビュー」≠「業務監査」、「指摘」＜「質問」、
    「問題」＜「課題」、「完璧ではない」という自覚、「根拠」と「合意」、
    これから先への「How」

【肝074】テストの見積もりのコツ

【背景／悩み】

- どれくらいのテスト項目数が必要かわからない。
- どれくらいのテスト工数が必要かわからない。
- 従って、テスト工期の判断ができない。

【肝の説明／解決のヒント】

- 過去プロジェクトやIPA（情報処理推進機構）の資料を参考に、規模あたりのテスト項目数の基準値を設定する。工数も同様にテスト生産性から算出する。
- テスト工期もテスト生産性と規模と人数をもとに算出する。
- テストの１項目の考え方を定義しなければ比較できない。
  せめて画面系・操作系と制御系を分けるとよい
- 結合テスト以降は、プログラムやシステムの特性によりテスト項目の粒度が異なるため、一律な基準を設定することが難しい。
  従って、対象物の過去の開発時のデータをもとに見積もる。
- テストで検出されるバグの、修正工数・確認工数も見積もる。

【肝078】データを見誤らないための、分析のコツ

【背景／悩み】

- メトリクスで見ると一見問題ないように見えたプロジェクトで開発の終盤に炎上する事件が発生した。

【肝の説明／解決のヒント】

- データを平均値で見ない。→ 個別の異常な“外れ値”を探す。
- データ上何の問題もない。→ 逆に怪しい。成果物から実態を探る。
- データを見る視点を広げる。→ 同じ組織や似た案件との比較で乖離を把握する。
- データの分析方法を段階的に進化させる。
  - ① 測った結果を「理解」できること。（理解＝分析）
  - ② 理解した結果を使って、現状の説明ができること。（説明＝対策）
  - ③ 対策を繰り返していける仕組みを作れること。（仕組み＝管理）
  - ④ 管理した状態を標準化できること。（標準＝定着化）
  - ⑤ 標準を改善できること。
    （改善＝CMMI：Capability Maturity Model Integrationのレベル５）

【肝085】テスト終了判断はテスト計画立案までに決める

【背景／悩み】

- テストをいつ終了すべきか分からない。
- テストが十分かどうかも分からない。
- 不安なので、期限ぎりぎりまでテストをする。
- 期限になったら、「やるだけのことはやった」とテストを終了する。

【肝の説明／解決のヒント】

- テスト終了判断の内容は、テスト計画で以下のような判断基準を決める。
  ① V字開発プロセスにより各設計仕様書を入力情報として抽出した、網羅的なテスト項目を全て消化したこと。（テスト未消化がないこと）
  ② テスト進捗とバグ摘出状況の信頼度成長曲線から品質を判断する。
  但し、信頼度成長曲線は、総合テストなどで使って初めて意味がある。
  機能テストでは、テスト項目の実行順序に影響されるため余り意味がない。
  ③ テストカバレッジの基準を設ける。

【肝091】ＳＬＡの意味をお客様に十分に理解いただく

【背景／悩み】

- お客様が納得するSLA（Service Level Agreement）を締結できない。
- SLAで要求される内容が現実離れしている。
- SLAにどのような項目を設定すればよいか分からない。
- SLAに設定する妥当な許容値が分からない。

【肝の説明／解決のヒント】

- SLAはお客様との“契約”として合意し納得いただくべきであるが、欧米と異なり日本のIT業界では、“緻密な契約”が不得手であったり、お客様が契約を無視して追加要求してくることがある。（日本のお客様は品質に対して厳しい）
- サービス提供側として以下を注意するとよい。
  ① お客様はSLAを十分に理解できていない場合が多い。お客様の対応者（情報システム部門、経営層等）によってSLAに対する意見が異なる場合もある。
  ② SLAの値（稼働率99.99%など）の意味、本当に必要なことか、コストとの関連等をお客様に説明して理解いただくことが重要である。
  ③ お客様から“どういう状態が困るか”、“最悪の状態は何か”を具体的に聞き出すと良い。

【肝095】デグレード防止にはテストの自動化が欠かせない

【背景／悩み】

- 十分に注意して作業をしているつもりなのに、リリース後の修正で、だびたびデグレードを起こしてしまう。

【肝の説明／解決のヒント】

- リリース後の修正作業で、影響範囲を完全に見極めるのは難しい。
- 修正箇所の確認以外に、周辺の機能について、デグレードが無いことの確認が必要になる。
- 既存のテスト資産を蓄積しておき、デグレードがないことの確認に利用すると良い。
- この場合に、テストの実施と、結果の確認を自動化しておくことで、大量のテストを効率よく実施することが可能となる。
  テスト資産を蓄積する際は、機能毎に分類しておき、実施する部分を選択できるようにすると、さらに効率化が図れる。

残された課題（やり残し）

- **今期はSQIPシンポジウムでの発表を見送り、また、4回の臨時検討会を開催し精査（肝の表現，書式統一，解説追加など全面的な見直しと肝の追加）に集中したが、目標とした肝の数108に届かなかった。（実績は95）**
- **メンバーの経験をもとに肝を作成しているため、“若い”組織に対しては気付きとなっても、“成熟した”組織には当たり前の内容が多いものとなった。**
- **肝は生き物であり、今後も継続的に見直し・追加が必要になると認識しているが、このための仕組みについての検討が全くできなかった。**
- **活動の成果を広く世間で役立てて欲しいと思っているが、この点についても同様に検討が十分できなかった。**

我々の最終ゴールには到達していない！！

最後に（まとめというか決意表明）

- 4年に渡り活動を続けてきた『ソフトウェア品質保証の肝』は、今期をもって本当に、本当に活動を終了します。

  →来期以降も活動を継続することは、ソフトウェア品質保証部長の会のミッションやビジョンからみて好ましくない。

- しかし、このまま終わらせてしまうのは．．．
- 責任感が強い私たちは後ろ髪をひかれる思い．．．

そこで

来期以降は水面下に潜り『仮想グループ』として活動します。

（仮称）“佐藤君とゆかいな仲間たち”

- ライフワークとして継続的に見直し・追加を行うことで、肝の陳腐化を可能な限り防ぎます。
- 新規メンバ大歓迎！！

  特典：悩みを抱えている方、解決のヒントを入手できます。

END