2013年度　第１回特別講義　レポート

日時	2013 年 5 月 10 日（金）　10:10～12:00
会場	日本科学技術連盟・東高円寺ビル　地下 1 階講堂
テーマ	「ソフトウェア品質向上活動の重要性～現場に喜ばれる活動を研究論文へ～」
講師名・所属	飯泉　紀子氏（日立ハイテクノロジーズ）
司会	SQiP 研究会運営委員長　秋山　浩一氏　（富士ゼロックス株式会社）
アジェンダ	1. ソフトウェア開発における制約と品質 2. ソフトウェア品質向上の道筋 3. ソフトウェア品質技術者の心得
アブストラクト	ソフトウェア品質向上活動の重要性は、今更語らずとも重々承知と言われそうです。 しかし、それでもなお語らずに居られないのにはいくつかの理由があります。 その一つとして、品質は製品やサービスを提供する対象や組織によって異なる、ということを理解した活動になっていないことが挙げられます。 また別の理由としては、品質向上活動が重要だと認識しているのにも関わらず、実践していない現状があるからです。 この講義では、ソフトウェア開発における種々の制約から品質を考えます。品質も制約の一つと考えるわけです。 そして、組織的な戦略に基づくソフトウェア品質向上の道筋について、事例やマネジメントフレームワークを引いて説明します。 また、研究会活動で習得・発揮していただきたいスキルについてもお話します。

＜講義の要約＞

◆1 章　ソフトウェア開発に

■ソフトウェア開発の課題

SQuBOK おける制約と品質にあるように『品質は経営層から現場に至るすべてのレイヤの体系化されたアクティビティにより追及すべきもの（経営層と現場が一緒になってやらなければうまくいかないもの）』である。しかしながら、昨今のソフトウェア開発事情は、経営層は品質を必須の条件と考えているが、現場は破ると見えやすい条件（納期）を強く意識している。要因は、コスト・納期は達成しなければならない数値が明示されていて、調整ができるが、品質は達成しなければならない数値が明示されず、調整もきかないことにある。品質を守りながら、コスト・納期も守るという制

約条件で開発を進めるのであれば、品質を明確にならない限り、コスト・納期と同様にトレードオフすることはできない。したがって、品質向上策に取り組むためには、達成する“品質”は何か、単位を何にするのか、目標値をいくつにするかということが課題となる。

■達成する“品質”は何か、単位を何にするのか、目標値をいくつにするのか

SQuBOK の Weinberg の言葉にあるように、『品質は誰かにとっての価値である』例えば、ソフトウェアを 1 日 8 時間利用するユーザにとっては使い勝手のよさが高品質であるし、故障のたびに批判されるシステム管理者にとっては、ゼロ故障が高品質である。厳しい予算の制約下にあるプロジェクト管理者にとっては開発費用が少ないことが高品質である。したがって、『品質とは障害（機能単位の能力の縮退または喪失を引き起こす異常な状態）がないこと』と一意に定義できるものではない。

『品質とは障害がないこと』とした場合、障害の少なさを単位当たりの量で把握する。≪単位≫はコード行数やファンクションポイント、≪量≫は開発の工程（プロセス）ごとに検出された障害数を利用する。

世の中の平均、自社の実力、顧客の要求から品質の目標値を決める。例えば、「1 ヶ月後までに 300 万円で要求事項を実現してほしい」という顧客の要求に対し、「割り当てられるリソースは 4～5 人なので、品質レベルは“B”で請け負う（品質レベル“B”とは・・・）」と定義して納期とコストを守る。

SQuBOK の Martin の言葉にあるように、品質の達成度は『システムが本稼働するとき、どこまで真のビジネス（ユーザ）ニーズにあっているかということ』である。そのためには、進化し続けるシステムを構築すること、時間とともに変化する顧客の要求を的確に把握し、その変化に可能な限り応えることが重要である。また、開発のスピードを上げながら、ユーザのニーズとの不一致を極力少なくする努力が必要である。したがって、品質は、何を達成しようとしているのか、状況に合わせて定義するものである。

◆2 章 ソフトウェア品質向上の道筋

高品質とは“障害が少ないこと”と定義する。高品質を達成するための道筋は、ゴール設定、アプローチ検討、施策実施と結果確認である。単純ではあるが、どのように当てはめるのかはポイントがある。

■ゴール設定

ゴール設定には三つのポイントがある。
一つ目は、課題を設定し目的にあったゴールを設定することである。
二つ目は、「品質活動に関係者を巻き込む」ことである。そのためには、価値観を共有すること、もしくは、危機感を高めることが挙げられる。価値観を共有することは難しいが、危機感を高めることはできる。危機感を持った人が半数を超えていれば、品質活動は進めやすい。
三つ目は、みなが合意する問題を定義することである。ひとつの方法としてソフトウェア製品の不

具合分析がある。
ソフトウェア製品の不具合分析を活用するためには、不具合情報が記録として残されていることが前提条件となる。その上で、不具合が発生している要因の仮説を立て、不具合情報を分析すれば、みなが合意する問題を見つけることができる。
例えば、弱い工程を探すのであれば、作り込んだ工程と発見すべき工程の統計を取る。スキル不足を探すのであれば、スキルのカテゴリ（個人、ベテラン、新人など）を設定して分類する。リスクの高い機能を探すのであれば、機能ごとに統計値を比較する。

■アプローチの検討
組織の特徴を SWOT に当てはめ、マネージメントコントロールフレームワークを活用して、期待されるように振舞いたくない原因を明らかにし、適合するアプローチでコントロールを決める。

〇SWOT とは

強み（Strength）、弱み（Weakness）、機会（Opportunity）、脅威（Threat）の４つの要因を軸に事業の評価や目標達成のための戦略を練るツールである。

〇マネージメントコントロールフレームワークとは

マネージメントコントロールフレームワークとは期待されるように振舞いたくない原因を明らかにし、適合するアプローチでコントロールする手法で、原因とアプローチには以下のものがある。
≪原因≫
(a)Lack of Direction：期待されることがわからない
(b)Personal Limitation：期待されていることを知っていてもそれに応えるスキルがない
(c)Lack of Motivation：期待されていることを知っていてスキルもあるのに、取り組まない
≪アプローチ≫
(1)Action Control：具体的な行動を示し、行動を促す
(2)Result Control：望ましい成果には報酬を与え、望ましくない成果には罰を与える
(3)People Control：人の性格や特性に働きかけ、行動を促す

■施策実施と結果確認
施策実施のために、施策推進体制を整え、関係者にオリエンテーションを実施する。施策実施の過程は（フィードバックできるように）記録する。また施策実施の効果を何にするか、どう測るかを決め、計測を実施する。結果確認は、関係者間で振り返りを行い、結果を共有する。

ソフトウェア品質向上策：事例 1
＜機能仕様とテスト仕様の同時設計による見逃し防止＞　≪Lack of Motivation – Action Control≫

〇動機
テスト工程が終わらず開発遅延する傾向が顕著になっていた。工程ごとのメトリクス（リソース／期間／不具合数など）を分析すると、手戻り作業が多発しテスト工程が終わらないことがわかった。

〇要因
品質計画書（設計者が行う不良再発防止策とその実施工程を宣言した文書）による振り返りから、設計者の意識はテスト工程を見直すこと（テスト項目を追加すること）であった。
テスト項目が不足してしまった現状・背景は、仕様書に記述があるのに、対応するテスト項目がない。仕様書の記述が不備なため、テスト項目が抜けた。仕様変更による影響範囲を把握しきれず、テスト項目が不十分ということであった。
テスト品質に影響を与える要因は、開発リソース（ソフト規模増大によるアウトソーシング：仕様の誤解釈、翻訳誤り）、ソフトウェア開発プロセス（V モデルに基づく設計：機能設計とテスト設計が時間的・人的に分離）、異なるドキュメントフォーマット（機能仕様とテスト仕様が別：レビュー時間がかかる／十分性・網羅性を確認しきれない）、ソフトウェア機能の複雑化（テスト項目が増大、トレーサビリティマトリクスの保守が困難）であった。

〇解決策
機能設計とテスト設計が分離した開発スタイルを見直す。
設計者がテスト設計に責任を持ち、必要十分なテスト項目の設計／テスト数を考慮した設計／テスト可能な設計を実施する。
そのために、既存の開発プロセス（コーディング後のテスト工程でテスト設計を実施）するやり方を設計者がテスト設計にも責任を持つプロセス（上流設計段階でテスト設計も並行して実施）するやり方に変更する。また、上流工程でのテスト設計を支援するツール（機能仕様を書きながらテスト仕様を書くツール）を準備する。

〇効果
QA への持ち越し不良が軽減した。同時に、どのようにテストするか／テスト可能性を設計時から考慮するようになり、正確なテスト設計には明確な機能仕様が必須であると気付き、機能仕様書の品質が向上した。また、機能とテストの対応づけが不要となり、設計レビューで QA 担当者からの指摘が増加し、レビューの品質も向上した。

〇成功の秘訣
問題の要因を動的要因まで掘り下げたこと。機会を活用したこと（オフショア開発に適用）。組織的なアプローチを実施したこと。

ソフトウェア品質向上策：事例 2
＜ドメイン知識の共有による組織的開発力強化＞　≪Personal Limitation – People Control≫

〇動機

短期開発、安全性の欲求、他者との競合など外部要因の変化により、ソフトウェア開発環境は小数精鋭チームから他人数多様化したチームへと変化し、未経験領域のソフトウェア開発が実施されるようになった。
このような状況の中、あるシリーズ製品の過去の不具合を分類したところ、ドメイン知識に関する欠陥が約半数を占めていた。

〇課題
ドメイン知識は個人の経験を通して個人の中に蓄積されていく。そのため、ドメイン知識を必要とする人は何を必要としているのかをうまく示せない。また、ドメイン知識を所有している人は、自らそれを表出できない。
ソフトウェアエンジニアに必要なドメイン知識は、対象分野を理解するための知識（顧客のビジネス、装置を導入する目的、顧客の業務フロー、法や規約改正などの社会的背景）、品質の良いソフトウェアの開発のための知識（要求や仕様の変遷、ハードウェア技術に関する背景、ソフトウェア技術に関する背景）である。今までは、これらの情報を OJT によって引き継いできたが、外部要因の変化により OJT で引き継いでいくことが難しくなった。

〇解決策
□ドメイン知識の共有
ソフトウェアエンジニア自身が実践できるように設計、チーム体制をとって、既存のドキュメントを利用した知識のフレームワークを構築、質問票を利用したインタビュー形式で知識を抽出する。
抽出したドメイン知識は、独学できるように解説書（文書）とドメインモデル図（関連図）で表現する。
□ドメイン知識の抽出手順
抽出する知識を定義→情報収集→収集した情報を整理し確認事項のリストを作成→専門家から質問票を用いて知識を抽出
□ドメイン知識の表現方法
抽出されたドメイン知識はソフトウェアエンジニアの限定された知識を補うために利用する。新規参入者がいつでも一人で学習できること、ドメイン知識の全体像とサブドメイン間の関係を把握できることを実現するため、解説書はユーザの視点で説明する（ソフトウェアの目的を説明する。設計根拠となる背景を述べる）。

〇ドメイン知識の抽出と表現の実施
部長によるトップダウン、チームがドメイン知識の抽出・記述を実施、サポートチームがドメイン知識抽出のプロセスを支援した。

〇ドメイン知識の共有（活用）・効果
ソフトウェアエンジニアリングの導入教育に活用したり、アクセスし易い場所に解説書を格納し各自の自習に活用したりしている。
また、既存の欠陥を使用した机上シミュレーション（今回抽出したドメイン知識を習得していれば、

どれだけの欠陥を防げたかをシミュレーション）した結果、18 事例のうち 16 事例を防止できることがわかった。

〇成功の秘訣
機会（担当者がいなくなった危機間、知りたいという欲求）を活用したこと。過去の失敗の経験から、ノウハウをチームの成果物としたこと。組織的なアプローチを実施したこと。

◆3 章 ソフトウェア品質技術の心得

■ソフトウェア品質技術者とは
以下のような人をソフトウェア品質技術者と呼んでいる。
・ソフトウェア品質向上に関する知識を、包括的かつ体系的に身に付けている人
・ソフトウェアの品質向上に対する効果的な策を、継続的に実施している人
・該当するキャリアが以下の人
品質保証スタッフ、開発者、テスト技術者、
プロジェクトリーダ・マネージャー、保守運用技術者、教育担当者、経営者
品質重視の組織文化を作るためにはソフトウェア品質技術者ひとりひとりの認識が必要で、ソフトウェア品質技術者は、品質重視の組織文化を作る種の役割がある。

■研究会は練習の場
研究会はソフトウェア品質技術者になるための練習の場である。「練習」は自らが繰り返したり、工夫したりして技術の向上をはかることである。研究会は調べる／考える／議論する／整理することを練習する場である。特別講義／指導陣の経験／研究員から世の中を知り、研究員同士のケーススタディから問題を把握する力がつき、それらを論文にすることで、成果を外へ発信できるようになる。

■論文とは
論文は論理的かつ分かりやすく整理した文書、外向けの文書である。論文の評価の観点には、有用性／信頼性／新規性／構成力がある。これらのうち、新規性は、全ての知恵と知識は先哲の業績の上に成り立っている。自分達の問題領域で、千哲たちのやってきたことに乗りながら、その上の新しい試みを見つければよい。

◆質疑応答
Q:成功にとらわれずに、みなに認められたことを壊して次に進められる（例えば、機能仕様とテスト仕様の同時設計による見逃し防止では、オフショア先にやらせるように変えていく）のは、何か秘訣があるのか？
A:壊したのではなく、壊れてしまった。続けていくうちに、テスト仕様は書けるようになったが、モチベーションを保つことが難しくなった。また、テスト仕様まで考えた機能仕様書をオフショア先に渡した場合に相手が同じように理解しているかということが不安になってきた。そこで、それをどのよう

に変化させればよいかを考え、相手にやらせれば一石二鳥になるという結論になった。このように、継続要否は柔軟に考えることが重要であると考える。

Q:オリエンテーションに感銘した。このステップは飛ばし、知識レベルが異なるまま進めているが、担当者によって違う反応が返ってくることになり困っている。何かよい方法はないか？
A:特別なことはないが、オリエンテーションは、メンバーに意図を伝え、やってほしいことを理解してもらう場であると考える（なぜそれをやらなければならないかをメンバーで共有することが重要である）。

Q:ドメイン知識の共有による組織的開発力強化は、だれかが責任を持ってメンテナンスしていくことが必要であると思うが、それをどのようにしているか教えてほしい。
A:メンテナンスをしていくことは重要である。メンテナンスは決まった担当者に任せるとモチベーションが下がるので、前回の受講者を次回の講師とする事で自身の知識も振り返る事ができるようにしている。結果としてうまくいっている。

＜講義の感想＞
ソフトウェア品質向上の道筋を立てるコツとして、みなが納得する事実（過去不具合）を分析し、その結果を活用して組織長と現場に危機感を煽ることで、みなが同じ目的意識を持って行動できるようにしていく手順をご教授頂き、そのエレガントさに感銘しました。自社においては、組織長と現場の認識が異なることで、品質活動が思ったように進まないことに悩んでいましたが、組織長と現場の認識を一致させることが何よりの解決策であると認識できました。まずは自社の過去不具合分析から、みなが納得する事実を見つけ出し、組織長と現場の認識を合わせていきたいと思います。