クオリティ ワン Quality One

Vol.15　2011年8月号

Software Quality Profession

財団法人 日本科学技術連盟

1. 品質

USDM と DRBFM を羅針盤にして派生開発を成功に導く

NEC ソフト株式会社
酒井　賢

１．はじめに

2011 年 2 月にソフトウェアテストシンポジウム 新潟で「派生開発で USDM（Universal Specification Description Manner）と DRBFM（Design Review based on Failure Mode）をミックスして一気通貫で品質確保する」という事例発表をさせていただきました。発表後、資料を見た何人かの方から、進めるに当たっての課題や現場にどのように導入したらよいかなどのご質問をいただきました。このような機会をいただきましたので、この場をお借りして詳しく説明します。

２．導入前の状況

２．１　プロジェクトの概要

次のようなプロジェクトです。2003 年から 2009 年までの約 6 年間担当しました。

- オフィス機器（以降「装置」と書きます）に添付される Windows 上で動作するユーティリティソフトウェア
- ソフトウェア単体では用をなさず、装置と通信を行うことによって利用出来る
- 設計、製造、テストまでの工程を担当
- 開発プロセスはウォーターフォール開発

装置開発の多くは、1 つのマスターに対して、自社向け、OEM 向けなど複数の製品が作られ、さらに、自社向けには廉価モデル、フラグシップモデルなどいくつかのモデルが製品化されます。

数年のサイクルで、その後継機が開発されてゆき、後継機開発までの谷間では、機能強化やバグフィックスなどの目的でメンテナンスリリースが行われたり、特定用途向けの改造開発が行われたりします。

私たちは、お客様(発注者)が何回か派生開発を経た後に引き継ぎました。

２．２　状況の分析

引継ぎ当初は、規模も小さく、改造も限られた範囲だったので、大きな品質問題が発生することはありませんでした。

しかし、開発を重ねるごとに、機能、対応機種、サポート OS が増加していき、規模と複雑さが急激に高まっていきました。メンバーの増員はあったものの、モジュール別に担当者を分けていたため、意思の疎通がうまく行っていませんでした。そうしたことが重なり、品質問題が頻発するようになりました。

	引継ぎ時	3年後
サポートOS	5エディション	12エディション
規模	50KL	120KL
モジュール数	4モジュール	21モジュール
対応機種	4機種	12機種
メンバー	2名	4名

表 1：引き継ぎ時と 3 年後の状況比較

メンバーへのヒアリングや過去に発生した問題を分析した結果、次のような問題があることが分かりました。

１）仕様化があまい

要求から仕様への落とし込みがあまいため、コーディング時になって仕様化がされていないことに気がつく後戻りが発生しました。また、テストフェーズで、動作途中でネットワークが切れた場合などの異常系を仕様化していないことがわかり、バグ修正と仕様化を同時に行うことがありました。

２）変更の経緯が分からない

仕様化する前段階で改造点の洗い出しを行っていると、どうして現在のような仕様になっているのか分からない部分が現れました。引継ぎ以前の経緯が分からないため、その部分には手を加えず、処理を分岐する形での設計を行わなくてはなりませんでした。

３）変更の影響範囲がわからない

変更に対するソースコードの修正箇所の特定が不十分で、修正漏れが多く見られました。また、変更を加えたことで、本来は影響のないはずの機能の振る舞が変わってしまうデグレードが発生しました。

４)”思いもよらない”問題が発生

テスト終盤になって、OS メーカーからサービスパックがリリースされ、それを適用するとセキュリティの関係で、これまで正常に動作していた機能が動作しなくなるという問題が発生しました。納期間際の想定外の問題の発生で現場が混乱し、仕様調整や対応で大きな作業負荷となりました。事後にふり返ってみると、セキュリティ強化のアナウンスはニュースサイトで読んで知っていた事を思い出しました。

３．課題

問題点を整理して、次の 4 つを課題としました。

１）要求の仕様化技術：どうしたら要求を正しく仕様化できるのか？

「仕様化があまい」という問題の解消

２）変更点の追跡：どうしたら変更点を追跡できるのか？

「変更の経緯が分からない」という問題の解消

３）抜け漏れ防止：どうしたら変更の影響範囲を特定できるのか？

「変更の影響範囲が分からない」という問題の解消

４）未然防止：どうしたら問題発生を未然に防げるのか？

「”思いもよらない”問題が発生」の解消

４．施策

メンバーの持ってきた本を元にして「USDM」を導入し、お客様から紹介された資料をきっかけに「DRBFM」を導入することとしました。どちらも、レビューを重視する点が共通している手法です。

導入にあたっては、私自身の”つかんだ”という感覚と、お客様やメンバーへの定着度合いをみながら順をおって進めました。

No.	課題	手法	引継ぎ時	3年後
1	要求の仕様化技術	USDM TM	変更要求仕様書	2003～
2	変更点の追跡			
3	抜け漏れ防止			
4	未然防止	DRBFM	心配点シート	2008～

表 2：課題と施策の対応

４．１　USDM の導入

施策の第一段階として、USDM（Universal Specification Description Manner）とトレーサビリティ・マトリクス（TM）を取り入れました。

USDM は株式会社システムクリエイツの清水吉男氏が考案された、要求を仕様化するための表記方法です。

USDM には次のような特徴があります。

- 1 つの表に全ての要求・仕様を記述する
- 要求を 2〜3 段の階層構造で定義する
- 各要求に対して、その要求が必要な理由を併記する
- 最下位の要求に対して仕様（その要求を実現するためのシステムの振る舞い)を記述する
- 要求・仕様にはユニークな ID を付与する

このような記法によって変更要件の意味を理解して、変更しなければならない仕様を漏れなく見つけ出して一覧化します。

トレーサビリティ・マトリクス（TM）は、USDM に付随するもので、USDM によって明確にした変更仕様が、どのモジュール（またはソースコード）に影響するかを改造規模で示すものです。これによって、ある要求を満たすには、どこを変更する必要があるのかが俯瞰でき、どのくらいの変更なのかを定量的に把握できます。

USDM と TM をセットにしたドキュメントが「**変更要求仕様書**」です。（図 1）

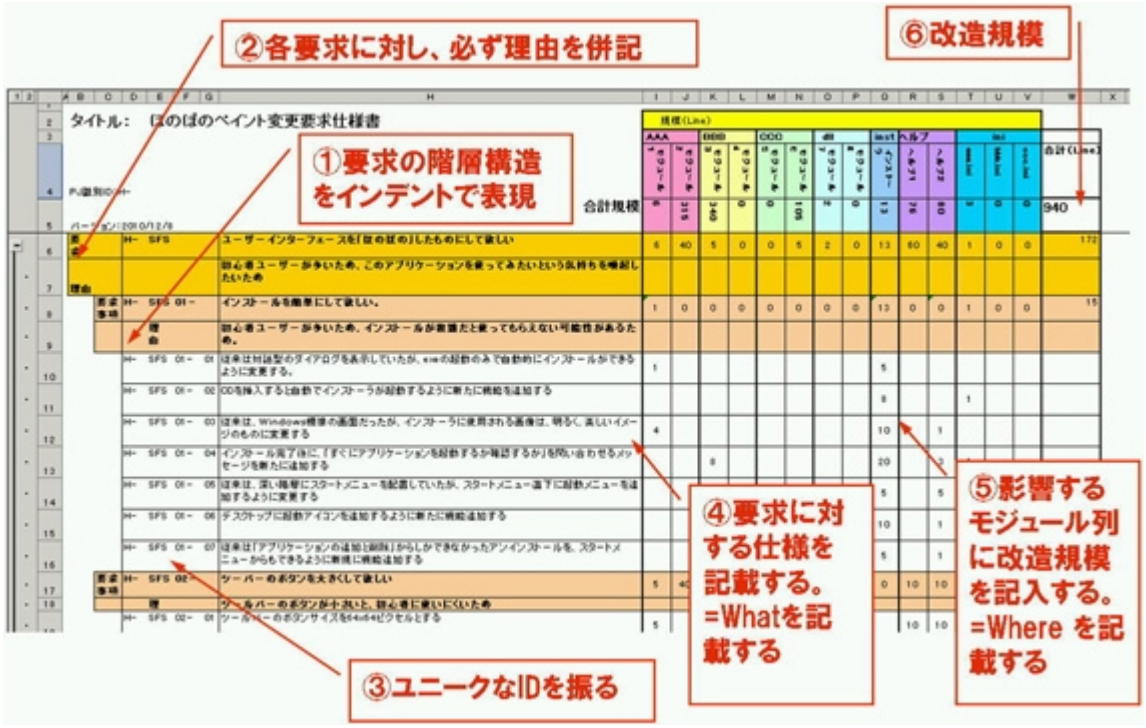

図 1：変更要求仕様書

表の左側部分は USDM 記法で記載した仕様です。これは「何を変更するのか」という“What”の視点で記入します。

右側部分は「トレーサビリティ・マトリクス（TM）」です。モジュール単位の列を作成して、仕様に対して変更が必要なモジュールとの交点に改造規模を記入します。TM は「どこを変更するのか」という“Where”の視点で記入します。

表が完成したら、要求が正しく理解できているか、仕様に抜け、漏れがないかを確認するためにメンバー全員で集合形式でレビューを行います。同時に、自担当部分の変更が他メンバーのモジュールに与える影響などの検討も行います。

図には簡単に記入方法を記載しましたが、詳しくは次の書籍をご覧いただくとよいかと思います。

- 「要求を仕様化する技術・表現する技術 - 入門＋実践 仕様が書けていますか?」

  清水 吉男（著）

- Software People Vol.8 「特集：コンサルタントが教える戦慄の仕事術 失敗しない派生開発」

  Software People 編集部（編集）

４．２　DRBFM の導入

USDM がメンバーに完全に定着した段階で、「DRBFM（Design Review Based on Failure Mode）」を導入しました。

DRBFM はトヨタの GD3（ジー・ディー・キューブ：Good Discussion, Good Design Review, Good Design）の一翼をなすもので、吉村達彦氏の造語です。「未然防止」の手法で「まだ起きていない問題の発生を予測し、それが起きないように未然に防止すること」を目的に考案されました。

- 変更点・変化点と機能に着目
- レビューによって心配点の発掘・探索
- 叡智を集め対策を作り込む

という考え方の下に、次のような表を使って、レビューを行います。

部品名／変更点	機能	変更に関わる心配点		心配点はどんな場合に生じるか		お客様への影響	心配点を除くためにどんな設計をしたか	推奨する対応（DRBFMの結果）						対応の結果実施した活動
		変更がもたらす機能の喪失、商品性の欠如	他に心配点はないか（DRBFM）	原因・要因	他に考えるべき要因はないか（DRBFM）			DRBFMで示された設計へ反映すべき項目	担当期限	DRBFMで示された評価へ反映すべき項目	担当期限	DRBFMで示された製造へ反映すべき項目	担当期限	

図 1：変更要求仕様書

※「トヨタ式未然防止手法 GD3 いかに問題を未然に防ぐか」吉村達彦著 （日科技連）を参考に作成

DRBFM は次のように行います。

1．設計者が水色のセル（「部品名／変更点」～「心配点を除くためにどんな設計をしたか」）を記入する。

2．関係者を集めて、黄色のセルの「他に心配はないか」「他に考えるべき要因はないか」を議論する。

3．心配点が見つかったら、それを除くための対応を議論し、「推奨する対応」に記入する。

4．４．「心配点を除くためにどんな設計をしたか」の妥当性を議論する。新たな心配点が見つかった場合は表に追記し、対応を議論して、「推奨する対応」に記入する。

5．最終的に心配点が見つからなくなったら終了とする。

6．後に対応が完了したら、黄緑のセル（「対応の結果実施した活動」）を記入する。

私たちは、導入当初の何度かは、図 2 の表をそのまま使ってやってみましたが、記入やレビューに時間がかかる割に効果が現れませんでした。

メンバーに変更要求仕様書とその背景となる考え方がすでに浸透していたため、変更要求仕様書を完成させた段階で、DRBFM 表の項目のような想定される問題はすべて考慮済みとなっていたのだと思います。

”思いもよらない”問題の発生を防ぐには、あまり形式ばらずに、「なんとなく怪しい感じがする」とか「ここは、あとあと何かが起きそうだ」など、理屈では説明できないけれど、経験上なんとなく焦げ臭いところを吸い上げて記録しておき、全員レビューのたびに気づきを誘発する、そんなものが必要なのではないかと思いました。「変更要求仕様書」が論理的な思考をあつかう「左脳」だとすると、予想や直感などをあつかう「右脳」に相当するものです。

そのような発想で出来上がったのが「**心配点シート**」です。

４．３　心配点シート

心配点シートは DRBFM を使ってみてやりにくかった点をスリム化して、変更要求仕様書とセットで使えるようにしたものです。基本的な考え方は、DRBFM と同じです。

次のような工夫を施しました。

- 変更要求仕様書と一体化できるように、要求事項と対になるようにした
- 詳細化しないように、わざと最上位要求という大くくりの単位で作成するようにした
- 気づきを誘発させるために DRBFM の表よりもゆるいフォーマットとした
- 各工程のレビューフェーズでつねに記入できるような欄をあらかじめ設けた

試行錯誤の結果、最終的には、次のようなフォーマットに落ち着きました。

要求事項	理由
メールの検索機能を追加して欲しい	多くのメールの中から、求めているメールを探し出したいため

変更するに当たっての懸案、心配点	心配点を取り除くためにどんな設計をしたか
メールが多くなると、検索に時間がかかるのではないか	【変更要求レビュー】 ・検索スピードを高めるロジックを調査する ・何件のメールを基準とするか検討する ・推奨するPCのスペックを販売推進部に確認する（お客様） 【設計レビュー】 ・xxx方法にて検索するのが最適と判断 ・1万件を基準値とする ・PCの推奨スペックはxxxとする ・10秒以内に検索が完了することとする 設上記設計書および変更要求仕様書に記載した。 【テスト項目レビュー】 テスト項目に記載があることを確認した

図３：心配点シート

心配点シートは次のように使います。

変更要求仕様書と同じファイルに納める

心配点シートは変更要求仕様書と対を成すものなので、変更要求仕様書と同一のファイルに別シートとして納めます。（図 4 参照）

シートの単位は、変更要求仕様書の最上位要求単位として、シート名も最上位要求の ID を記入します。

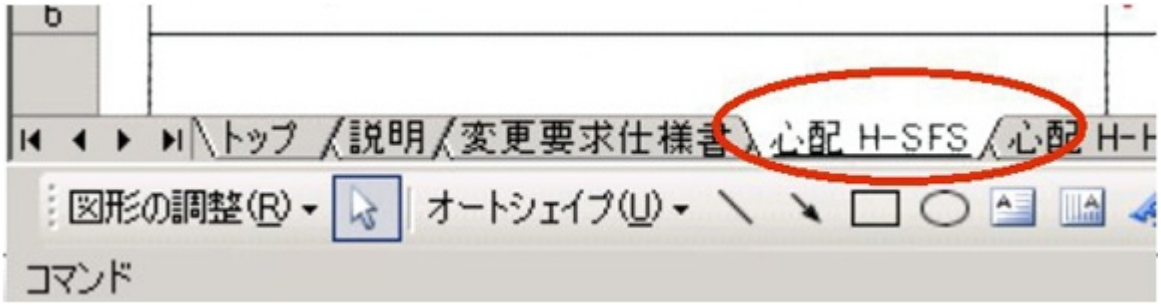

図４：心配点シート

シートの上段

上段左側に、変更要求仕様書の大要求の「要求事項」を、左側には「理由」を、変更要求仕様書どおりに記入します。

シートの下段：左側

左側に「変更するにあたっての懸案・心配点」をなるべく短く記入します。レビューで気づきを誘発するのが目的なので、詳細に書く必要はありません。今はまだ発生していないけれど、将来発生しそうな問題､なんとなくぼんやりした不安点などを､論理的かどうかは気にせず 1 行につき 1 つ記載します。

シートの下段：右側

右側には各工程のレビュー名を記載します。おそらく一般なウォーターフォール開発では、基本設計、機能設計、詳細設計、単体テスト…など細かなフェーズに分かれると思います。それらのフェーズをあらかじめすべて記入しておきます。（図 3 では工程を簡略化しています）

変更要求仕様書を作成しながら記入する

変更要求仕様書を作成している最中はいろんなことを気にしているはずです。多くは仕様化することで解消されるでしょうが、それでも変更要求仕様書作成段階では払拭できない心配点は少なからず残るはずです。そうした心配点を漏れなく記入しておきます。

変更要求仕様書レビュー時にレビューする

変更要求仕様書のレビューのたびに、レビューメンバー全員で、心配点が顕在化するかどうかを考えます。特に思いつかなければ「対処の必要なし」と記載して済ませます。何か気づきがあれば、「その心配を取り除くためにどんな対処をしたか（するか）」を記入します（図 3 の赤字の部分）。検討している最中に新たな心配点が浮かんだら追記して、さらに検討を行います。

５．成果

変更要求仕様書と心配点シートを導入することで次のような効果が得られました。

- 時間をかけて全員でレビューすることで、いままでよりも詳細な仕様化ができ、仕様化漏れがなくなった。
- 理由を記載することで、要求の真の目的を理解でき正確な仕様化ができるようになった。
- ユニークな ID を成果物に記入するようにしたので、どの開発でどのような変更が発生したのかを追跡することができるようになった。
- TM によって、変更に対する各モジュールへの影響が横断的にわかるようになり、レビュー段階で変更箇所の漏れに気づくことができるようになった。
- 変更が一覧化されて 1 つの表に集約されているため、影響範囲がひとめでわかり、変更量と複雑さがイメージでつかめるようになった。
- メンバー全員がレビューに参加することによって、自分の担当モジュールだけでなく、他のメンバーの担当モジュールへの影響も意識するようになった。
- 心配点シートで些細な心配でも吸い上げるので、メンバー全員が「プロジェクト全体」に対して、責任感と参加意識が強くなった。さらに、バグの発生は、メンバー全員の敗北という意識が芽生えた。
- 成果物レビュー時に心配点シートのレビューを行うことで、緊張の時間と弛緩の時間を作り出し、レビューにメリハリが出るようになった。
- ベテランになればなるほど、経験に裏付けられた心配点が記入される傾向があり、なぜそのような心配点が出るのかが若手に伝わるチャンスとなった。

変更要求仕様書と心配点シートを導入することで、課題としていた 4 つは解消されました。大げさな話ではなく、本当に変更仕様に関する問題は発生しなくなりました。

少なくとも、ソフトウェアという不可視なものを扱う場合、「そこに変更のすべてがあり、常に最新の状態に保たれていて、その変更をメンバー全員が知っている」という安心感は、品質の向上にも役に立つものと思います。

６．導入にあたってのポイント

導入と進め方のポイントを経験をふまえて記載します。

a. 必要性と表記方法のスタディ

変更要求仕様書と心配点シートを完成させるには、メンバー全員の協力が必要となります。そのためには、なぜ変更要求仕様書と心配点シートが必要なのか、どのようなルールで記入するのかを理解しておかなければなりません。事前に集まって勉強会を開きましょう。

b. 記入ルールの明確化

変更要求仕様書の要求事項、理由、仕様はプロジェクトリーダーが記入、TM は各モジュール担当が記入、心配点シートは全員が記入する事としました。なぜなら、USDM は、「何を変更するのか」の観点で記載するため、細かな実装を知らなくても書けるからです。これが書けないようではお客様との仕様調整は難しいということになります。TM は実装を知っているモジュール担当者が記入します。TM を記入する際には、必ず仕様を参照するので、自然とクロスレビューになります。

c. お客様をまきこむ

いきなりお客様に「変更要求仕様書を導入します」と言っても、なかなか OK が出ないかもしれません。私たちは、導入当初はお客様と実施するデザインレビューの際に改造仕様書のサブ資料のようなかたちで提出していました。開発のたびにだんだん露出度を上げていき、最終的には、改造仕様書を作成する前段階の仕様書として認知されるようになり、お客様もまじえてレビューを行うことができるようになりました。

d. 事前配布・事前レビュー

変更要求仕様書と心配点シートは事前配布・事前レビューを徹底します。配布時にレビュー記録表を添付して、レビューアはレビュー結果と追加の心配点を記入して、集合レビューの前に作成者に戻します。

これには 2 つの効果があります。１つは、仕様書がレビューに値する完成度になっているかの確認です。あまりにも要求を取り違えているとレビューに時間がかかるばかりで先に進みません。逆に、「て・に・を・は」などの些細な間違いは、レビュー時に気になって指摘し始めると、意外と時間がかかってしまうものです。そのような些細なものは、集合前の段階で指摘してレビュー前に取り去ってしまいます。

2 つめは、事前に目を通すことで、無意識下に働きかける効果です。技術者は意外と仕事の時間以外でも、なにがしか担当モジュールについて考えているものです。変更仕様を脳内で熟成する時間を確保することで、心配点シートに載るような、ひらめきや気づきに期待します。

e. 仕様変更の度に全員レビュー

要件の追加や変更が入った場合や仕様変更が発生した場合には、必ず変更要求仕様書を改版して、常に最新の状態をキープします。仕様が変更になれば、影響範囲や心配点も変わるはずなので、どの工程であったとしても、再度全員でレビューを行います。時間はかかりますが、より安全で確実な変更を行うための投資と考えます。

f. 各工程の成果物のレビュー毎にその機能が取り込まれているか照らし合わせる

変更要求仕様書と心配点シートは、次工程のインプットであるとともに、実際は、すべての工程のインプット文書となります。変更要求仕様書に書かれた変更点は、改造仕様書にも反映されますし、テストも行われるはずです。ですので、各工程の成果物をレビューするときには、変更仕様が確実に反映されているか、他に心配点はないかを確認します。

g. 以前の変更要求仕様書と心配点シートを参照する

新たな変更要求仕様書を作成するときは、今まで作成した変更要求仕様書を見直して、類似の変更をしたことがないか、顕在化しなかった心配点が今回は顕在化する可能性はないかを考えながら作成します。変更要求仕様書はメンバーの叡智の固まりなので、何らかの気づきにつながるはずです。

h. 理由や粒度にこだわり過ぎない

変更要求仕様書導入当初は、要求の理由を書くのが難しく感じるはずです。要件書に理由が書かれていることは少ないですし、今まで要求に対する理由を考えたことがあまりないからです。仕様の粒度にも悩むと思います。細かくすると膨大な数になる可能性もありますし、荒くすると要求事項と同じになる気がします。でも、大丈夫です。要求事項によって荒い粒度になったり細かい粒度になったりするのは、複雑さや難易度などが自然と反映されているからです。レビューしてくれる人を信じて、理由や粒度にはあまりこだわらずに記載します。

i. レビューは緊張と弛緩を交互に

成果物のレビューは常に緊張感が伴います。そうした緊張感の合間に、頭をほぐす意味も含めて心配点シートのレビューを行います。変更要求仕様書の記載で想定される問題の検出はされているはずなので、心配点シートでは問題が発見されなくて当たりと思って、やや発散系になってもよしとしてやるのが良いと思います。リラックスした場の方が気づきが生まれる可能性が高まります。

７．まとめ

変更要求仕様書と心配点シートを使った開発のイメージを図にしてみました。（図 5）

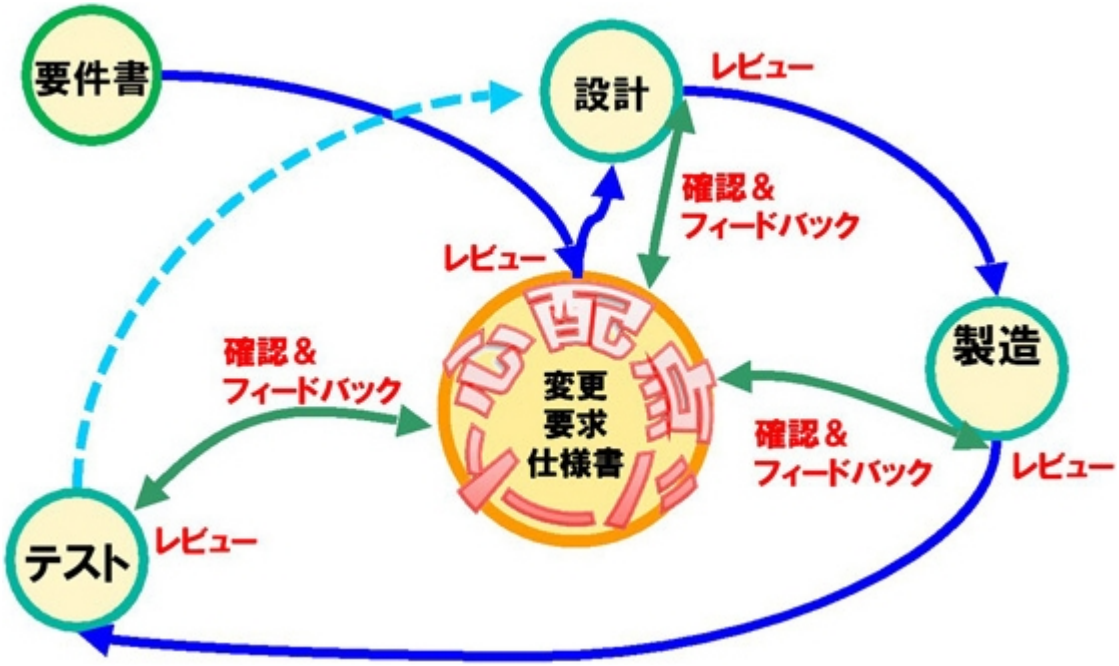

図 5：変更要求仕様書と心配点シートを中心とした開発のイメージ

変更要求仕様書は、プロジェクトの羅針盤のようなものだと考えています。

ウォーターフォール開発のサイクルでは、それぞれの工程の成果物は、前工程の成果を受け継ぎながら進んでいきます。進むにあたって、変更要求仕様書という羅針盤を常に確認し、「その成果物には変更が反映されているのか」を必ずチェックし、心配点シートで「心配していたことが顕在化していないか、さらに心配すべきことはないか」を検討しつづけます。

航海に例えるなら、今までの航路とこれから進もうとする航路が正しいのかを羅針盤で確認し、行く手に危険が待ち受けていないかを推測する作業を怠らないということです。

変更要求仕様書と心配点シートを中心に据えて開発サイクルを回すことで、いつ難破してもおかしくない派生開発という困難を、正しいゴールに向かって安全にかつ穏やかに進めることができます。

プロジェクト管理システムやツールはたくさんありますが、品質を確保する最後の砦は、コミュニケーション、チームワーク、経験、助け合いなど、「人」が主役だと思います。

変更要求仕様書と心配点シートを導入してからの約 3 年の間に 14 ファイルができました。つまり、14 回の派生開発を繰り返したことになります。

変更が入るたびに何度も書き直され、全員でレビューしたファイルは知識が凝縮されたプロジェクトの宝でありチームの誇りです。

プロフィール
酒井 賢　（さかい けん）
ＮＥＣソフト株式会社 新潟支社 リーダー
プリンターやスキャナーのユーティリティソフトウェアの開発リーダーを経て、現在は、Android アプリケーション開発プロジェクトの品質保証職に従事。
新潟ソフトウェア開発勉強会（SWANii）会員
日本産業カウンセラー協会会員
日本ファシリテーション協会（FAJ）会員
「チームワークやコミュニケーションを活性化するための場づくりや見える化など、現場に根ざした「ソフトウェアファシリテーション（PF)」を実践しています。メンタル面やモチベーションがソフトウェアの品質にどのような影響を与えるかについて考えています。」

2. 人材育成

■ 論文を書くことの大切さ

株式会社システムクリエイツ
代表取締役　清水　吉男

・はじめに・

今回は、私にとって「論文を書くこと」あるいは「論文的思考」が、どれほど役に立ったかということについてお話しします。この論文的思考によって今の私が存在しているといっても過言ではありません。その有効性、そして重要性を分かっていただくために、まずは私の最初のつまずきと、そこで出会った「論語」と「大学」の文章からお話しします。

・失敗から始まった私の SE 人生・

弁解がましくなりますが、当時は数人の仲間が集まって仕事を融通しあっていました。オフコンのようなものは一人でも対応できますが、コンピュータのメモリーが大きくなると、稼働日までに必要なシステムを開発するには何人かで手分けすることになります。仲間の中では「失敗」の状態は程度の差であって、私だけが失敗していたのではありません。それでも私の場合は、最初の「憧れ」が強すぎたのでしょうか。私としては「失敗」の状態を解決できないまま仕事を続けることができませんでした。私の「ソフトウェアの世界での第 1 歩」はこうして惨憺たるものでした。

・復帰のきっかけは「論語」と「大学」・

1 年間まったく別の世界にいて、そこから戻るきっかけになったのは、半年程経った時に「論語」を読んで下記の記述を見つけたことでした。

> 弟子の冉求（ぜんきゅう）が
> 「子の道は説（よろこ）ばざるに非ざるなり。力足らざればなり」
> （先生のお話しは大変ありがたいのですが、私にはとてもそこまでできません）
> といったのに対して、孔子は、
> 「力足らざる者は、道中（みちなかば）にして廃す。汝は画（かぎ）れり」
> （能力の無い者は、途中で倒れるもの。お前はやる前から「かぎって」いるだけだ）

もともと漢文は好きな領域でしたし、私たちの世代は「論語」は人生に迷ったときに読む本という認識を持っている人が多かったと思います（もちろん「儒教」というレッテルを貼って嫌う人も少なくないですが)。論語の中にこれを見つけたことで、自分は本当に「力足らざる者」なのか、を確かめるためにソフトウェアの世界に戻ることにしたのです。

そう、私は道半ばで倒れたわけではありません。まだ生きているのです。あのとき「自分にはこの仕事は向いていない」と啖呵を切って抜けただけだったのです。私の中に 3 年前の「憧れ」は消えていなかったのでしょう。

戻るにあたって日々の行動のガイドラインとしたのは「大学」の以下の節です。

格物　致知　誠意　正心　修身　斉家　治国　平天下

一般には「天下を平らげるにはまず自分の国を治め・・・」と後ろから解釈するのですが、ここでは簡単に前からの解釈を紹介しておきます。

物のあるべき姿をただし、知るべきことを極める。そうすれば誠意を貫けるし、心も正しく維持でき、自分の身を修めることができる。そこまでくれば、家がととのうし、国を治め、さらに天下を平らかにすることができる。（ちなみに、斉家から平天下までが同時に達成するという解釈もあります）

私は、この最初の「格物 致知」を自分の置かれている状況に合わせて以下のように読み替えました。

「格物」すなわちソフトウェアの世界の有るべき姿を知って、「致知」すなわち、それに必要な知識や技術をすべて手に入れる。そうすれば、「誠意　正心」すなわち行動は誠に背かないし、お客さんに心正しく対応できる（ようになる)、と。

一度撤退した者としては、「大学」のこの 1 節を信じるしかない状態でした。当然ですが保証はありません。そのためにも、とにかく「格物　致知」ということで、すべての時間をそこに投入しました。

今の若い人たちの中には「大学」という本があることすら知らないかもしれませんね。でもここで出会った「格物　致知　誠意　正心」は、この後の私の人生に決定的な役割を果たすことになります。人生とはそういうものかも知れません。

・私と論文の接点・

復帰後、「格物　致知」ということで、手に入る文献は片っ端から入手して、毎晩仕事が終わってからアパートの部屋で夜中の 2 時頃まで自習しました。この状態が約３年続きます。しかしながら組織に属していないため新しい技術などで単行本になる前のものが入手できません。雑誌といっても、内容的にしっかりしたものは「bit」という雑誌ぐらいでしたので、それだけでは不足しているという不安を埋めることができません。

また、大手の企業には組織の中に先輩たちが残した知識や技術がありますし、直接話しを聞くこともできます。でもフリーで活動している者にはそうした知識や技術を手に入れることができません。時々、客先で目にしたものをお願いして借りて帰るぐらいです。今日のようなインターネットのない状況では、このハンディはとてつもなく大きなものでしたし、私自身「このままではマズイ」と「危機」を感じていました。「格物 致知」にならないのです。

この不利な状態をどうカバーするかということで、あるとき情報処理学会の論文誌を見つけ、読み始めたのです。当時は情報処理学会の会員ではなかったので、近くの書店にお願いしてオーム社から取り寄せてもらいました。その他、客先によっては電気通信学会の論文誌が置いてありました。この雑誌にもソフトウェアに関する論文が書かれていることがあり、それを見つけて借りて帰りました。これが私と論文との最初の接点です。

当時、私の回りの人でそこまでやっている人はいなかったのですが、私自身、一度躓いてこの世界から撤退しており、二度は失敗できないという状況にあったことと、復帰する際に「格物 致知」を腹に決めていたことによって、「致知」の一つの実現手段として、論文を読むというところに繋がったのでしょう。論文を手に入れて読むということには何の迷いもありませんでした。

当時、手に入る単行本は片っ端から入手して読んでいましたので、それで十分と考えることもできたかもしれませんが、復帰前に「大学」を読んでいたことで、それだけでは「致知」にならないと考えたのです。実際、まだ「そこ」に知るべき「場（領域）」があるのですから。もし事前に「大学」を読んでいなかったら、この時点で論文を読むことはなかったでしょうし、そうなると今の私はなかったと思います。

いや、自転車操業でボロボロになっていたかもしれません。実際、フリーで仕事を続けるのは決して楽なことではありません。そんな中で、「営業しない」という方針を貫くことができ、しかも一度も仕事が途切れなかったのは、やはり「格物 致知」の効果であり、そこから発した「論文を読む」という行動のお陰だったと言えるでしょう。

・論文の構成が共通していることに気がついた・

こうして論文を読んで、その構成に沿って理解した範囲で要約をまとめていました。そうして論文を読んで要約をまとめていると、論文の構成が

「abstract － まえがき － 現状分析 － 課題抽出・・・」

と共通していることに気がついたのです。

（現状分析）

- まず、現状を分析して問題を整理します。この時、データが収集されます。
- 集めたデータを原因などの特徴で分類して選択しやすいように整えます。

（課題抽出）

- 整理された問題から、解決の優先度が高いものや効果の大きいものを選択します。
- 選択された問題を、取り組みやすいように「課題」として形を整えます。
- このとき、問題に繋がったと思われる原因の分析が行われます。

（対策の選定）

- 推定されている原因に有効がありそうな技術を探したり、既存の技術を組み合わせたりします。
- この時、関連した課題を扱っている論文や文献を調べることになります。関連しているものがあればそこでの対応方法が参考になります。
- 今回の課題に適したものがなければ、それまで経験してきた中から新しい対応策を考えることになります。

（対策の選択と実施）

- そうした対応策の中から良さそうな方法を選んで、実際に取り組むためにプロセスなどの環境の整備を行い、その上で実際に取り組みます。

（検証と考察）

- 取り組んだ結果を検証し、当初の課題がどこまで解消したか検証します。
- 結果のデータ計測は、事前の現状分析のときにやった方法と基本的に同じです。
- 解決しなかった課題や、部分的に解決しなかった事象を確認して、継続取り組みとします。
- この時、今回の取り組みをベースに、さらに工夫して対応することになります。

さらに、この論文の構成が、まるで物事が解決していく階段のようで、読み手の私にとって心地良さ、リズムのようなもの感じたものです。同時に、この構成が問題を解決する思考のステップであり実践のステップになっていることに気付いたのです。

（このステップは、いわゆる「QC ストーリー」とほぼ同じですが、当時の私にはそのような知識は持ち合わせていません。直感的にこのステップが問題を解決する、と感じたのです）

・技術の習得にも使える？・

この種の論文は、基本的にそこにある問題を解消する、あるいは軽減するための取り組みが行われ、その経緯が記述されています。そこから、自分の回りにある問題もこれで対応すれば良いのではないか、と考えるようになりました。

当時、私の回りにあった問題としては、

- 顧客要望を掴みきれていない
- 見積もりの精度が悪い
- 開発途中の仕様変更が多い
- バグが思った以上に出ている

といったものでした。

実際問題として、ソフトウェアの開発現場にはいろいろな問題が溢れています。これらの問題の解決が遅れれば遅れる程、問題が新たな問題の原因になったり、問題が絡んだりして複雑になり、解決の糸口が見えにくくなります。

これとは別に、新しい設計技術などが提案され、文献の形になって出てきます。ソフトウェア開発に関する技術には、モジュールの分割尺度のように、ほとんど知っているだけの状態でも役に立つものもありますが、設計手法のように「習熟」が必要な技術も少なくありません。そして「習熟」が必要な手法は簡単に習得できるものではありません。

そこで考えたのは、このような身の回りにある問題の解決や新しい文献に書かれている技術の習得にこの論文の構成が使えるのではないかということです。

・論文を書き始める・

最初は、不足しがちな知識や技術を補充するために「論文を読む」ことを始めたのですが、ここにきて自分の問題を解決するために「論文を書く」、あるいは新しい技術をしっかり習得するために「論文を書く」という段階に進むことになります。

論文を書くといっても学術論文を書くわけではありません。自分の回りにある問題を課題として捉え、それを一つずつ解決していくために論文を書くわけで、いわゆる「経験論文」のようなものです。「ようなもの」というのは、私自身はこうして書いた論文は正式にはどこにも提出していませんので、第３者の査読を受けていません。もしかすると「論文」と言えるものになっていないのかも知れません。真っ赤に訂正されて送り返されたかもしれません。

それでも、私の回りにあったバグは大幅に減少します。また見積もりの精度が上がったことなどの効果からスケジュールの遅延も直ぐに解消します。仕様の書き方が安定することで、途中で発生する仕様変更などの問題も片っ端から片付いていきます。

こうして、レビュー技法や、見積もり技法、スケジューリングなどの他、リスク管理や構造化手法もこの方法で入手してきました、また「一人レビュー」やクリーンルーム手法（のマネごと）に取り組んだときも、そして最後のプロジェクトの一つ（このとき 2 つのプロジェクトを並行）で取り入れた「Review less による開発」のときも、論文の形にしながら考えた方法を検討し、取り組み方をシミュレーションしながら実施してきました。

私の場合、問題を整理して課題として抽出する際に「原因」を仮定します。「問題」は症状として見えていますが、それが起きるのは何が原因だろうか、と考えるのです。何かができていない、何かを間違えているからこの問題が発生しているのではないかというように「因」と「果」で捉えます。こうすることで、取り組みの方法を選択する時点で、この「因」に対して効果がありそうな方法を選択しますので、結果は 70～80%改善するのです。

この「因」と「果」で捉える方法は、後にリスク管理でも活用する方法を考えました。リスク管理では、この考え方の他に「USDM(Universal Specification Describing Manner の略。 仕様がモレない表現方法として筆者が開発したものです)」の表記法を応用して対応する方法を、やはり論文の形で実施し、効果を確認できています。

こうして課題に取り組みながら論文の形に整理することで、そこで取り組んだ方法が確実に身につきます。また論文の「形」に残していることで、その上に「未解決事項」への取り組みの工夫がやりやすくなります。さらに、数年後にこの方法が使えるという時に、引っ張り出すこともできます。

こうして結果的に 20～30 歳代の間に 30 数本の論文（らしきもの）を書いてきました。もちろん、40 代にも数本書いています。その効果は、

- バグの発生率は「0.1／KLOC 以下」（単体テストを含む）
- 22 年間納期遅れなし、仕様トラブルなし
- 組み込みシステムの時代は週 35 時間しかメインの顧客に投入していない
- 健康に何の問題もなし

という結果に現れていると思っています。

1996 年から提供を開始した「硬派のホームページ」（URL: http://homepage3.nifty.com/koha_hp/）は、当時「文字の青木ヶ原」と呼ばれていました。そこに掲載したコンテンツのネタは、こうして書き溜めた論文（らしきもの）や、いろいろな場面で書き溜めた「対策メモ」が元になっており、それを文章に作り直したものです。

そして今、SQiP の研究部会でこうしてお手伝いするというところにも繋がっているのです。

お知らせ：次回の Quality One（#16）では、続編として SQiP での研究会の活動の意味や、そこで論文を書くことの効果について書く予定ですので、ご期待ください。

プロフィール
清水 吉男　（しみず　よしお）
株式会社システムクリエイツ　代表取締役
派生開発推進協議会　代表
「私のほうで提案している「USDM」「XDDP」という方法を中心にして派生開発における開発プロセスの改善活動をしています。中でも、「XDDP」はいろいろな場面に応用できますので、派生開発推進協議会や SQiP の研究会などを通じてその可能性を広げる活動を行っています。また「USDM」による要求仕様の記述の精度を向上させる研究にも取り組んでいます。」

3. SQuBOK®

■ 「SQuBOK の利用法：参照スタイルから進化スタイルへの提案（その２）」

株式会社 NTT データ MSE
ソリューションサービス事業部
コンサルティンググループ　堀　明広

「SQuBOK」とは「Guide to the Software Quality Body of Knowledge」の略で、正式名称は「ソフトウェア品質知識体系ガイド」です。「SQuBOK ユーザー会」は，以下を目的に，2009 年に設立されました。

対して「SQuBOK ユーザー会」は、以下を目的に、2009 年に設立されました。

- 先人が切り開き、培ってきた品質技術を伝承し、それらを有効利用して、更には発展させるため、SQuBOK 活用の事例を共有する。また、SQuBOK をより密に活用する方策・方法を模索し、共有する。
- ソフトウェア開発に関係する者にとって、SQuBOK がより価値ある知識体系に進化し続けることに、SQuBOK のユーザー自身も参画し、これを実現する。

従来の「SQuBOK ユーザー会」の活動のメインは、メーリングリストを通じての議論・情報交換を主としていました。

前回の記事では、ソフトウェア品質に関する知見は非常に広くて深淵であることを、私の実体験を交えて紹介しました。また、SQuBOK をより活用してソフトウェア品質向上活動に繋げる一つの手段として、「SQuBOK ユーザー会」で今後取り組んでいきたいことの概要に触れました。今回は、それらをもう少し詳しく記します。

【SQuBOK ユーザー会 新活動の概要】

SQuBOK には、今まで先人が切り開いてきた知見が体系的にまとめられています。

SQuBOK ユーザー会を発足させた時には、これらの知見を蓄えた SQuBOK の活用の仕方を共有する、ということを目指していました。

つまり、今までの SQuBOK ユーザー会は、色々な知見が書かれている SQuBOK をどう利用するか、という観点で捉えていました。

ソフトウェア品質に関する知見、事例は常に進化しています。

これらを、SQuBOK ユーザー会で積極的に扱っていきたいと考えています。

- 新たに出された知見を整理し、体系付けるのに、SQuBOK を軸として利用する。
- 勉強会等で出された意見等を、その場限りでなく、一個の体系に記録し、蓄えていく。
- それらを使って、更に議論を深めていき、新たな知見を産み出していく。

提案したいことは、SQuBOK で整理されている樹形図を利用して関連事項を集約・充実させ、過程の中で議論をしていこうよ、というものです。

これらを行うには、internet 上で何らかのインフラが必要と考えています。

どんなインフラか、その運用方法はどうか。　これらを SQuBOK ユーザー会で議論していきたいと思っています。

【SQuBOK の樹形図の構成、参考文献】

SQuBOK では「樹形図」を軸にして、ソフトウェア品質に関する知見を整理しています。

この樹形図は、SQuBOK では目次として機能しています。

例えば、「第 1 章 ソフトウェア品質の基本概念」は、以下のように構成されています。

第1章　ソフトウェア品質の基本概念

1.1 KA：品質の概念

1.1.1 S－KA：品質の定義（品質の考え方の変遷）

1.1.1.1 T：品質の定義（Gerald M. Weinberg）

1.1.1.2 T：品質の定義（James Martin）

1.1.1.3 T：品質の定義（Joseph M. Juran）

1.1.1.4 T：品質の定義（Philip B. Crosby）

1.1.1.5 T：品質の定義（Roger S. Pressman）

1.1.1.6 T：品質の定義（Robert L. Glass）

1.1.1.7 T：品質の定義

1.1.1.8 T：品質の定義（IEEE Std 610）

1.1.1.9 T：品質の定義（ISO 9000）

1.1.1.10 T：品質の定義（ISO/IEC 25000）

このように、SQuBOK の樹形図は、最大４階層で構成されています。上記のそれぞれには「本文」「関連トピックス／知識領域」「参考文献」「関連文献」が記載されていて、ソフトウェア品質に関する知見が要約されています。

SQuBOK でソフトウェア品質に関する調べ物をする際には、樹形図で関連する箇所を探し、そこの「本文」に書かれている内容を読んで概要を理解し、更に詳細を知りたい時には「参考文献」に記載されている文献にあたる、といったことがされていると思います。

【ソフトウェア品質の知見を整理し、蓄積し、共有し、そして議論する】

現在の SQuBOK の「参考文献」で紹介されているものは、「書籍」や、ISO・IEC・IEEE 等の「規格」が中心です。

SQuBOK ユーザー会では、これら SQuBOK からポイントされている「書籍」「規格」を更に充実化させると共に、更に加えて、ソフトウェア品質に関わる「記事」「論文」「トピック」といったものをも扱っていくことを考えています。

以下、詳細を記します。

●書籍

繰り返しになりますが、SQuBOK はその名は「ソフトウェア品質知識体系ガイド」です。

SQuBOK の本文中に記載されているのは要約であるため、その中身をより理解を深めるためには、「参考文献」をあたる必要があります。

SQuBOK の「参考文献」には、その記載量の制限から厳選されていますが、「参考文献」でポイントされていないものにも、良書はたくさんあります。また、SQuBOK が出版されて以降にもソフトウェア品質に関わる多数の書籍が出版されています。更に視野を広げ、ソフトウェア品質に直接的に関連しないものにも参考にすべき書籍は非常に多く、これらを紹介し合うことには、大きな意義があると思います。

書籍を紹介する際には、単に書籍名・著者・出版年月等のデータだけではなく、紹介者による感想や内容に関する意見も含めておくと、議論もしやすくなると思います。

●規格

SQuBOK の出版後にも、ISO・IEC・IEEE・JIS 等の規格は新規策定・改訂・廃止がなされています。現状に合わせて状況を俯瞰・整理することを考えています。

また、ソフトウェア品質の観点で、その規格で記載されている内容の意図、複数の規格の関連・相違点などが議論できると良いと思います。

●記事

ここで言う「記事」とは、技術情報誌やビジネス誌や、新聞・ニュースの記事と、これらに関連する internet サイト、web マガジン、blog 等を指しています。

ソフトウェア品質に関する調査は、書籍や論文だけでなく、上述の「記事」を検索サイトで調べることも頻繁に行いますが、ここで言うまでもなく、各種メディアや internet 上での情報は爆発的に増加してきているため、目的の情報を得るのが難しくなってきています。

こういった情報を抽出・整理・共有すると、知見の発掘・整理に大いに役立つと考えています。

●論文

今回提案している中で、一番重視したいのは、この「論文」です。

論文と言うと、何か堅苦しい、敷居が高いイメージもあるためか、論文を読んだことが無いエンジニアが多いように思います。

まったく、もったいない話です。どうか食わず嫌いをしないで、一度、論文を読んでみることを、強くお勧めします。

例えば、日本科学技術連盟主催の「ソフトウェア品質シンポジウム（通称：SQiP シンポジウム）、ASTER 主催の「ソフトウェアテストシンポジウム（通称：JaSST）」では、実務者が執筆した事例報告が数多く報告されています。

これら論文では、その組織やプロジェクトで抱えている問題・課題を整理し、それを解決するための方法を考察してそれを実施し、その結果がどうだったのか検証するまで、分かりやすく整理されており、書籍等だけは得られにくい実践事例を知ることができます。

為すべきことは分かっているが、具体的にはどう手をつけていけば良いのか分からないといったケースや、取り組みはしているが、壁に当たって行き詰まり・手詰まりし、悩んでいる方も多いかもしれません。こういった時には、論文でヒントになる事柄が見つかることが多いものです。

また、論文を読むことで、各組織ではどんな取り組みがなされているか、トレンドも把握でき、視野を広げることにも役立ちます。

しかし、どこで、どんな論文が発表されているか、その入手方法が分からない方も多いことでしょう。

これらを SQuBOK の樹形図を軸にして整理し、その情報を共有したいと考えています。

●トピック

私自身の感触ですが、ソフトウェア品質に関わるエンジニアは、組織の垣根を越えた交流が盛んで、横の繋がりが強いように思います。

例えば、SQiP コミュニティやソフトウェアテスト技術者交流会等のコミュニティを母体に、各地域で自主的に、活発に勉強会等が催されています。

私自身、このような勉強会や研究会活動をこれまで一生懸命にやってきましたが、その過程で多くの方々に育てていただいたと思っています。

こういった活動の存在そのものを紹介し合うこと、また、そこでディスカッションで交わされた意見や得られた知見、整理された事項等を持ち寄って蓄積・共有することが出来れば、互いに大きなメリットになると思っています。

また、これらの情報共有をする過程で、互いに持ち寄った情報に対して意見・補足情報等を加えるようにして、いったら、別の議論が生まれ、それが新たな知見の創出に繋がるものと思っています。

【準備はほぼ整いました】

「書籍」「規格」「記事」「論文」「トピック」と、SQuBOK ユーザー会の新活動で扱う事項についてご説明してきました。

この活動を行うには、internet 上で情報を蓄え、共有するための器（インフラ）が必要ですが、クラウドサービスを活用するよう、現在 準備を進めています。

この原稿を執筆しているのは 8 月 1 日ですが、この時点で internet 上のインフラ準備はほぼ整っており、まもなく SQiP コミュニティに案内を出せるところまで来ています。

次回のこの記事では、本取り組みをどのように実践しているか、報告出来ると思います。

なお、SQuBOK ユーザー会にはどなたでもご参加いただけます。
SQuBOK ユーザー会の詳細は以下に情報がありますので、興味をお持ちの方は是非ご参加ください。
お待ちしております。（SQuBOK ユーザー会の web サイトも、近々に公開する予定です。）
http://www.juse.or.jp/software/142/

プロフィール
堀　明広（ほり　あきひろ）
株式会社 NTT データ MSE
ソリューションサービス事業部　コンサルティンググループ
日科技連 SQiP シンポジウム委員会　副委員長
日科技連 SQuBOK ユーザー会　世話人
ソフトウェアエンジニアリングで好きな分野は、メトリクス。中でも見積りは特に重要であると確信している。もうひとつ好きなのはテスト。テストは破壊的であるのと同時に、創造的な魅力に溢れていると思っている。
得意技はバグ分析。バグ情報には，品質を占う情報と，改善の道へ繋がるヒントがたくさん散りばめられている。

4. トピックス

（１）ソフトウェア品質技術者資格認定　第２回中級試験 11 月 26 日に開催決定！

財団法人　日本科学技術連盟　SQiP 事務局

１．第１回中級試験合格者は 14 名

すべてのソフトウェア技術者が品質技術を身につけ、実践していくことにより、ソフトウェア品質の向上を実現することを目的として昨年にスタートした「ソフトウェア品質技術者資格認定試験」の第 1 回中級資格試験は 2010 年 11 月 6 日に実施しました。

71 名が受験し、そのうち 14 名の方が見事合格を果たしました。合格率は 19.7%という難関でした。

中級は、ソフトウェア開発プロジェクトを品質面で成功に導くことができる技術レベルを目標としており、試験対象者は、品質保証技術者はもちろん、プロジェクトリーダー、設計技術者、実装技術者、テスト技術者、保守運用技術者、教育訓練職、企画職、営業職、マネージャ層、経営層といったソフトウェア開発に関わるすべての方々です。

２．第 2 回中級試験の開催

第 2 回目である今年は、11 月 26 日（土）に開催することを決定いたしました。当日は、初級資格試験も午前中に予定されているため、午後に実施されます。

試験の概要を以下に記しますが、受験申込開始時期は、9 月中旬～下旬を予定しています。

（1） 試験時間：2 時間（休憩なし）

（2） 試験問題　※サンプル問題は Web サイトへ掲載しています。

①選択式　25 問

②記述式　3 種類の問題形式（下記）よりあわせて 15～20 問程度

- 文章中の用語の穴埋め：10 問程度
- 用語についての定義や活用方法の説明（50 字程度）：5 問程度
- あるテーマについてその理由や留意点などの考察を記述（50 字×5 項目）：２問程度

（3） 合格ライン：70%程度（選択式、記述式とも）

（4） シラバス（http://www.juse.or.jp/software/35/#02）に準拠して出題

３.「中級資格者の会」（仮称）設置について

中級資格試験のさらなる充実と資格取得者のネットワーク構築と情報交換の場として「中級資格者の会」（仮称）の設置を検討しています。

そのキックオフとして SQiP シンポジウム 2011←web ページにリンクを張る（9/7～9：早稲田大学・西早稲田キャンパス）にて、「中級資格者の会 SIG」を設ける予定です。

中級資格者だけのクローズな場ですが、オンライン上の討論をはじめ、SQiP 委員との交流会、情報交換会など様々な企画を検討していく予定です。

さあ、みなさん、中級資格にチャレンジしてみませんか？受験資格は特にありません。

多くの方々の受験をお待ちしています。

4. トピックス

■ （２）第５回世界ソフトウェア品質会議

（5th World Congress for Software Quality：5WCSQ）

参加募集のご案内

～ ひとつの旗の下に団結しよう　―卓越した最高品質を目指して―

財団法人　日本科学技術連盟　SQiP 事務局

世界ソフトウェア品質会議（WCSQ） は、全世界のソフトウェア品質専門家が集まって幅広く議論を行う、唯一のソフトウェア品質についての国際会議です。ASQ (The American Society for Quality)、EOQ (The European Organization for Quality)、JUSE （財団法人日本科学技術連盟）の 3 団体がホスト団体として WCSQ を運営しています。

第 1 回会議を 1995 年にアメリカで開催したのを皮切りに、これまで、第 2 回(2000 年)を横浜、第 3 回(2005 年)をドイツ・ミュンヘン、第 4 回 (2008 年)をアメリカ)にて開催しました。第 5 回目となる 2011 年 10 月 31 日からの会議は、中国・上海にて開催いたします。

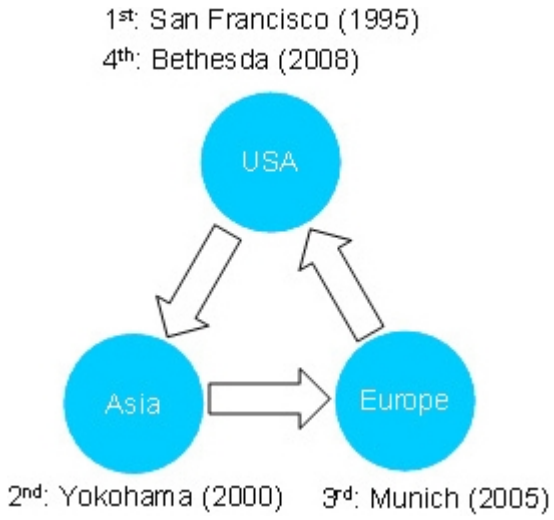

図 1：WCSQ ホスト団体のローテーション

来る第 5 回では、テーマに、斉心協力 追求卓越（United Under One Banner: Best of Best Quality ／ひとつの旗の下に団結しよう―卓越した最高品質を目指して―）を掲げて、ソフトウェア業界の変化が激しい状況下、常に世界のソフトウェア業界の情報をインプットすべく開催いたします。

著名な経営者や、第一線で活躍するエキスパートらによる基調講演のほか、様々なチュートリアル、ソフトウェア品質について議論が交わされるパネルディスカッションなど、盛りだくさんの内容を予定しています。

論文発表セッションでは、一般論文の発表に加え、各国のシンポジウムなどで優秀な評価を得た「ベスト・オブ・ザ・ベスト」論文の発表も企画しております。

3 年に一度､世界のソフトウェア品質専門家が一堂に会する WCSQ へのご参加をお待ちしています！

表 1：5WCSQ スケジュール

5WCSQ Schedual Tutorials (10/31 MON)			
8:30	Tutorial Registration		
9:00	**Tutorials1** (China) **Dehua Ju** "From Product- to Service-Oriented Quality"	**Tutorials2** (EOQ) **Geoff Thompson** "TMMi – the model in action"	**Tutorials3** (Japan) **Naomi Honda** "Software Quality Accounting System™- Quality Management Method to Realize High Quality beyond CMMI Level 5 – "
12:00	Lunch Break		
13:00	**Tutorials4** (China) **Yang Genxing** (The title is undecided)	**Tutorials5** (EOQ) **Karol Frühauf** "Requirements – The Foundation for Quality"	**Tutorials6** (Japan) **Makoto Nonaka** "Measuring and Analyzing Software Defects and their Data"
16:00	Session Chairperson & Speaker Meeting		
17:00	Until 18:30 **Welcome Coktail**		

5WCSQ Schedual First day (11/1 TUE)				
8:30	Registration			
9:00	Opening Plenary Session, Welcome Address			
9:30	**Keynote speaker 1**			
	何积丰（中国科学院院士、華東師範大学軟件院院長、博士教授） (The title is undecided)			
10:30	Refreshment Break			
10:45	**Keynote speaker 2**			
	Mr. Masahiro Sakane, Chairman of the Board, Komatsu Ltd., Japan “DANTOTSU Management” – a company that builds on strengths from generation to generation			
11:45	**Keynote speaker 3**			
	Prof. Dr. Bernd Hindel, Method Park Software AG, Germany “Are we ready for Engineering in the Cloud?”			
12:40	Lunch Break			
13:40	**A1** Oral Session	**B1** Oral Session	**C1** Oral Session	**D1** Oral Session
14:10	**A2** Oral Session	**B2** Oral Session	**C2** Oral Session	**D2** Oral Session
14:40	**A3** Oral Session	**B3** Oral Session	**C3** Oral Session	**D3** Oral Session
15:10	Refreshment Break			
15:30	**SIG** (Special Interest Groups)	**SIG** (Special Interest Groups)	**SIG** (Special Interest Groups)	**SIG** (Special Interest Groups)
17:30	Until 19:30 **Banquet**			

5WCSQ Schedual Second day (11/2 WED)				
9:00	**A4** Oral Session	**B4** Oral Session	**C4** Oral Session	**D4** Oral Session
9:30	**A5** Oral Session	**B5** Oral Session	**C5** Oral Session	**D5** Oral Session
10:00	Refreshment Break			
10:15	**A6** Oral Session	**B6** Oral Session	**C6** Oral Session	**D6** Oral Session
10:45	**A7** Oral Session	**B7** Oral Session	**C7** Oral Session	**D7** Oral Session
11:15	**A8** Oral Session	**B8** Oral Session	**C8** Oral Session	**D8** Oral Session
11:45	Lunch Break			
12:45	**A9** Oral Session	**B9** Oral Session	**C9** Oral Session	
13:15	**A10** Oral Session	**B10** Oral Session	**C10** Oral Session	
13:45	Refreshment Break			
14:00	**A11** Oral Session	**B11** Oral Session	**C11** Oral Session	
14:30	**A12** Oral Session	**B12** Oral Session	**C12** Oral Session	
15:00	Refreshment Break			
15:45	Until 17:15 **Panel Discussion**			
	Discuss about Verification Technology (tentative)			

5WCSQ Schedual Third day (11/3 THU)				
9:15	**A13** Oral Session	**B13** Oral Session	**C13** Oral Session	
9:45	**A14** Oral Session	**B14** Oral Session	**C14** Oral Session	
10:15	Refreshment Break			
10:30	**A15** Oral Session	**B15** Oral Session	**C15** Oral Session	
11:00	**A16** Oral Session	**B16** Oral Session	**C16** Oral Session	
11:30	Lunch Break			
12:30	**Keynote speaker 4**			
	Dr. Patricia McQuaid, The ASQ Software Division, "U.S.ASoftware Disasters – Understanding the Past, to Improve the Future"			
13:30	Refreshment Break			
13:45	**Keynote speaker 5**			
	Mr. Yaron Tsubery, Smartest technologies Ltd., Israel "The challenges followed by the recent trends in Software QA and Testing"			
16:45	Until 16:00 **Closing Plenary Session**			
	Awarding Ceremony Closing Remark Announcement of the next WCSQ			

5WCSQ Schedual Forth day (11/4 FRI)	
9:00	**Technical Visit**
	Shanghai Pudong Software Park, China 863 Software Incubator (Shanghai) Base
16:00	End

詳しくは、Official web site から！

http://www.5wcsq.org/en/home.html

5. 健康

■ 『部下の不調を見逃すな！マネージャーのためのメンタルヘルスケア』

株式会社プラネット･コンサルティング
代表取締役　根岸　勢津子

部下を持つ皆さんの、一番大切な仕事の目的とは何でしょうか。多分多くの方は、チームの生産性を向上して、よい成績・成果を上げること、とお答えになるでしょう。その通りですね。会社は小さなチームの集合体によるピラミッドあるいは文ちん型を形成し、成績・成果を集約し、企業としての利潤をあげて行くものだからです。では、チームの生産性はどうしたら向上するのでしょうか。それはリーダーシップとフォロワーシップが見事にかみ合った時、マネジメント機能が発揮され、そのチーム最大のパフォーマンスを見せることができると言われています。しかし、それもすべて、メンバー全員が健康であるという前提において語られることです。今回は、自分のチームメンバーの健康管理にスポットを当ててお話しましょう。

【メンバーの不調がもたらすリスク】

皆さんの部下がメンタル不調に陥った場合、チーム全体に与える影響にはどのようなものがあるでしょう。まず、他のメンバーが不調者に気を使います。メンタル不調というのは勤怠に乱れが出たり（来たり来なかったり、または遅刻など）仕事の能率が下がったりしますので、本当に周りは困ってしまいます。病状が進んだ状態になれば、職場の士気に影響が出るでしょう。休職となれば、その人の代わりに入れるメンバーの確保、教育などにお金と時間がかかり、原価を増やす原因となります。

次に、顧客に与える影響です。仕事が滞れば納期も遅れ、お客様と約束した日までに、約束したクオリティの製品を納入することが難しくなります。また、メンタル不調者は仕事のミスも増えることが多いので、トラブルの原因にもなりかねません。

【部下のメンタルヘルスに気を配るのはリーダーの義務】

なぜリーダーは、部下の健康にまで気を配らなくてはならないのでしょうか。ひとつには、冒頭に書いたとおり生産性を向上してチームの成績・成果を上げるのがリーダーのミッションだからです。もうひとつは、企業の安全配慮義務という側面があります。これは労働契約法第５条に明記されており、『企業は労働者が安心して安全に仕事に従事できるよう注意を払わなくてはならない』という意味のことが書かれています。では、実際にその義務を遂行するのは誰でしょうか。そうです、リーダーである皆さんの義務なのです。製造業・建設業などにおいては、現場での安全確保がそれにあたりますが、パソコンの前で開発業務などに当たる方たちの安全とは何か、それを考慮して日々のマネジメントに活かすのが、皆さんに課せられた法的な義務なのです。今やメンタルヘルスケアは、企業における安全配慮義務の最重要課題と言えましょう。

【企業を取り巻くリスク】

安全配慮義務を怠ると、企業はどのようなリスクを被るのでしょうか。ここ数年、企業と労働者の間でのトラブルや訴訟が激増しています。未払い残業代、名ばかり管理職の問題と並んで、メンタル不調による解雇や労災認定の問題がクローズアップされています。東芝のうつ解雇無効訴訟や、電通の若手社員の過労自殺訴訟などは、いずれも企業が不利な立場に立たされ、多額の賠償金を支払う結果となっています。労働基準監督署の指導内容も年々厳しくなっており、企業は対応を迫られています。

リーダーの皆さんは、残業を命令する立場であるならば（管理監督者）、労災事故の際に責任が発生する可能性もありますので、どうか仕事の一環として部下の健康管理に取り組んで頂きたいと思います。

【部下の不調に気づくポイント】

体の病気も同じですが、メンタル不調も、早期発見・早期治療が何よりも大切です。あなたの部下に下記のような人はいませんか？

1. け・・・欠勤
2. ち・・・遅刻
3. な・・・泣きごとめいたことを言う
4. の・・・能率が下がった
5. み・・・ミスが増えた、身だしなみが乱れた
6. や・・・辞めたいという

1～6 の頭文字をとって、『けちなのみや』と覚えましょう。あくまでも、『以前と比べて』という条件付きです。あんなに毎朝きちんと出勤してきた人が・・・とか、あんなに明るかった人が・・・という変化に気づいて下さい。ただし、変化というのは A から B に変わることを言うのであって、A を知らなければＢになったことを変化と捉えられません。何が言いたいかというと、普段の部下の様子に気を配っていなければ、些細な変化に気づくことはできないということです。

これらのポイントのうち、2、3 つ程度現れたら、迷わず声をかけましょう。ここであなたがためらっていては、手遅れになる可能性があります。

【上手な声のかけ方】

いきなり、『君、なんかウツっぽいから病院行けよ！』と言われたら、誰だって気を悪くするでしょう。ここは下記のような声のかけ方がいいと思います。

『ねえ〇〇君、ちょっと顔色悪いみたいだけど、ちゃんと睡眠取れてる？』

『ねえ△△君、ここ最近元気ないみたいだけど、ちゃんと食事摂れてる？』

このように、体調の心配から入るのがいいでしょう。こういうふうに声を掛けられて気を悪くする人は、めったにいません。また、睡眠障害と食欲不振は、うつの始まりの二大兆候と言われておりますから、ちょっとした問診にもなるのです。

しかし、相手は『いえ、大丈夫です。』と答えることが多いものです。ここで、それを鵜呑みにしてはいけません。あなたはいったん引きさがりますが、その後決して目を離してはいけません。かれの状態が、そのままか、悪くなるか、回復するか、少なくとも 1 週間～10 日間の観察が必要です。そして、1 週間経っても様子が変わらない、あるいは悪くなっていくようでしたら、次の行動に出ます。

【専門家への受診を勧める】

次にあなたの取るべき行動は、部下を専門家に受診させることですが、そんな時、あなたの会社の人事部門は助けてくれますか？早速会社に確かめておきましょう。万が一、部下にそういう事象がおこった場合、会社が用意している医療体制があるのか、あれば、どのように利用するのか、等々。

- 会社に産業医などの体制があり、相談に乗ってくれる場合・・・人事部を通して部下を産業医に受診させる。
- 会社に体制が無い場合・・・部下に自分で医療機関に行くように勧める。

いずれの場合にも、話し方にはテクニックが必要です。そして少しの勇気も。ここは『アイ・コミュニケーション』を利用します。これは、『私は～』から始まる話し方で、自分がどう感じているか、に重きを置いて考えを伝えるやり方です。

例えば、部屋が散らかっている子供に対して・・・『さっさと片付けなさいよ！』というのが、『ユー・コミュニケーション』です。あなたは片付けるべきだ、と『あなた』が主語になっていますね。では、これはどうでしょう。『あなたの部屋が散らかってると、お母さん悲しくなっちゃうわ。片付けてくれたら嬉しいんだけどな。』これは、自分の気持ちを伝えている『アイ・コミュニケーション』です。この手法を、部下にも応用してみましょう。

『ねえ、〇〇君。最近の君を見てると、どうしても心配なんだよ。

一度専門家のところに行ってくれれば僕も安心できるんだけどね。』

こんな調子です。ひとつも『病院に行け』とは行ってませんね。これを、あなたなりにアレンジして、実際に部下の顔を思い浮かべて、ロールプレイングをしてみて下さい。いざという時、あわてないで済みます。

【安心して働ける環境づくりと】

一方、職場づくりもリーダーの大切な仕事ですが、具体的には何に注意したらいいでしょう。それは大きく分けて 2 つあります。1 つはハラスメントに気をつけることです。ハラスメントには、セクシュアル・ハラスメント、パワー・ハラスメント、アルコール・ハラスメントなど様々な種類がありますが、自分がされて嫌なことは人にもしない、という基本的なこが理解できないメンバーもいるかもしれません。一人ひとりの言動に気を配り、行きすぎた行為、発言があればリーダーとして毅然とした態度で注意しましょう。あなたがそれを放置することで職場の士気が下がります。自分で解決できそうにないと感じたら、すぐに会社の人事部門に相談しましょう。

2 つめは、部下への仕事の与え方です。つい優秀な人にたくさんの仕事を任せがちになりますが、これは過重労働の原因となりますので、できるだけメンバー全員に均等な業務量となるよう配慮しましょう。それから、メンバーごとに以下の 3 つの観点から適正な状態かどうかチェックします。

1. 仕事の量
2. 裁量度
3. 周囲からの支援

この 3 つを合わせて『仕事の質』といいます。仕事の量が余りにも多すぎたり少なすぎたりしては、人は実力を発揮できませんね。また、言われたことばかりやっていては創造性が発揮されず、結果として生産性が低下します。権限の移譲を上手に行って、部下に適正なる裁量度を与えましょう。そして大切なのが周囲からの支援です。あなたがもし、新規プロジェクトを任せられ、わからないことがあるのに頼れる先輩も無く、愚痴を言う仲間もいなかったらどうでしょう。健康な人でも落ち込んでしまいそうですね。3 つの観点から、部下一人ひとりの『仕事の質』を常にチェックし、偏りが無いかどうか観察しましょう。

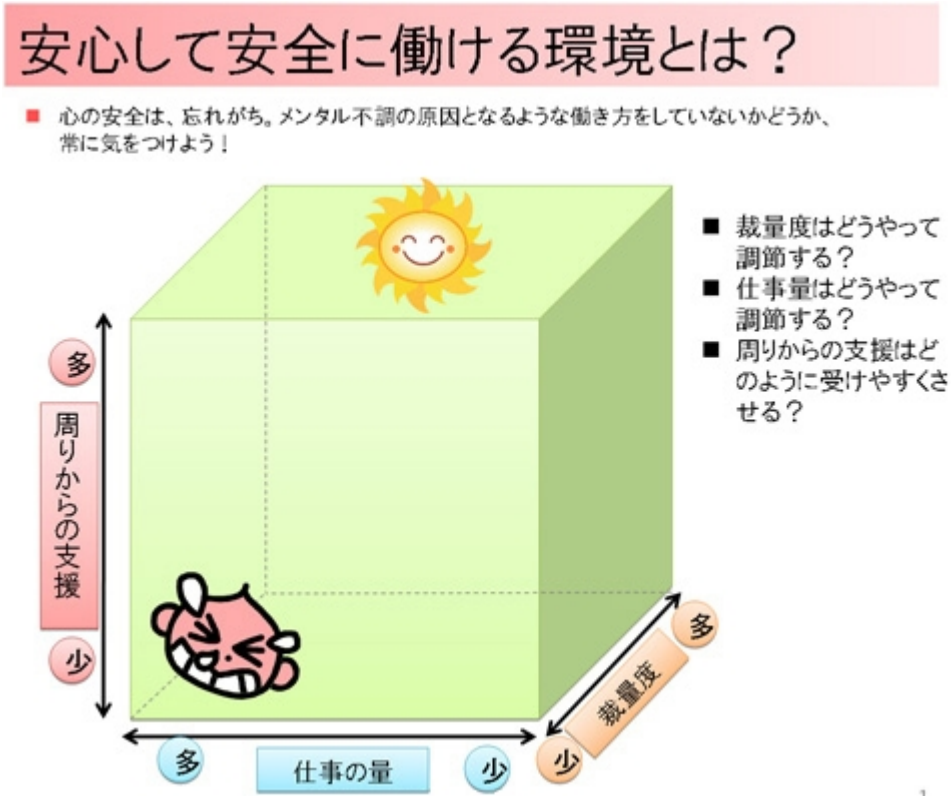

図 1：変更要求仕様書

ハラスメントを決して許さない職場、仕事の質がよい職場は、メンタルヘルスも向上します。部下の健康管理はリーダーの義務と心得て、日々のマネジメント、頑張って下さいね。応援しています。

プロフィール
根岸 勢津子　（ねぎし　せつこ）
株式会社プラネット・コンサルティング 代表取締役
企業や団体の中で『人の役に立つためには』を常に考え、外資系、海運会社、IT 企業などで秘書の経験を積む。その後、大手損保代理店に転職したが、法人・団体を守るためには保険販売のみならず、リスクマネジメントの考え方が必要と感じ、アドバイザーの経験を積む。近年、産業界にヒューマンエラーによる不祥事が続発したことを受け、企業に対するメンタルヘルスケアの体制構築に関するアドバイスに注力して事業を進め、2006 年に法人化。メンタルヘルス対策に取り組む企業からの様々な相談に応じ、コンサルティングを行う。