73

ODC 分析による設計品質改善の取り組み[1]

～先進的な設計・検証技術の適用事例報告書（2013 年度）事例を活用した取組み～

1. 承前

本件では、ODC 分析に取り組むにあたり、先進的な設計・検証技術の適用事例報告書 2013 年度版で公開されたオリンパスソフトウェアテクノロジー株式会社 山崎 隆氏の「Orthogonal Defect Classification 分析による欠陥除去と品質の成熟度可視化」を参考にさせていただいた。

ここで技術的に重複するものは、IPA/SEC で公開済みの該当資料を参考にすることを前提とする。[1]

2. 適用プロジェクトの概要

当社の某組み込みソフトウェアの派生開発において、テスト期間が超過し、リリースが間に合わないケースが発生していた。この問題を解決するため、開発プロセスの見直しの一環として、設計品質の向上を図る改善活動を始めた。

その改善策の一つとして、質と量の分析に対応できる ODC による欠陥分析を導入した事例を紹介する。

これまでの欠陥分析は、検査リーダーのスキルに依存しており、標準的な分析方法も確立していなかった。また対象となるプロダクトが増え、さらに複雑化する環境においてテスト結果報告も客観的な分析が求められてきている状況にあった。

これらに対応する施策の一つとして、ソフトウェアの欠陥分析を標準化する取組みを始めた。ソフトウェアの欠陥分析方法は数多く存在するが、社内標準化するために利用可能な技術を調査していたところ、ODC（Orthogonal Defect Classification,以下、ODC と省略する）分析の存在を知り、IPA/SEC から先進事例として報告されていたものが活用できそうだと考えた。

本事例では、特定のプロジェクトの欠陥分析を試行として、そこから標準的な欠陥分析手法として確立していくための取り組みを紹介する。

今回の改善活動に取り組んだのは、対象プロジェクトの主管開発部門と筆者が所属する品質管理部である。対象プロジェクトは組み込みの派生開発をここ数年に渡って実施しており、リリースを数カ月単位で行なっている。しかし納期に間に合わず、リリース日を再設定することも生じている。こうしたことは、顧客との信頼関係や当社の利益にも大きな影響を与えてしまう問題である。

---

[1] 事例提供: エプソンアヴァシス株式会社 小島 義也 氏

この問題に対し、スケジュール厳守と利益確保、つまりはプロジェクトを正常化させる必要があった。

これらを実現するために、現場ではいくつかの開発プロセスの改善を計画し、上流工程での品質の作り込みの実現を目標のひとつに掲げた。この目標に対し、品質管理部では ODC を用いて分析した結果から設計欠陥の真の原因を突きとめ、リリースを遅らせている品質問題を発生させないようにするため、改善活動の実施を関係部署や客先と調整し、ODC による欠陥分析の取り組みを開始した。

3. 解決すべき課題の整理

今回の解決すべき課題への対策は、「テスト工程での欠陥を少なくするためにはどうすればよいか？」との議論から始まり、上流工程での欠陥混入を防止する対策を立てることが抜本的であり、かつ長期的にみれば賢明な解決策であるとの意見に集約された。

そのためには上流工程の真の原因を突き止め、しかるべき対処を取れば、テスト工程の欠陥数が減るはずだと判断した。その根拠としては、テスト工程で発見された欠陥がどこの工程で作りこまれたか、その混入工程を調査した。

すると、設計工程での混入欠陥数の割合が全体の 80%を超える数値を示した。

これらの調査から設計工程の欠陥混入の削減が重要だと判断し、全体および設計工程での欠陥密度の低減を目指した。そして次に、混入原因を特定するところへ焦点を移していった。

ソフトウェアテストで検出した欠陥の原因を分析することは、プロダクトの品質の成熟度を示す重要な活動であるが、個人のスキルに依存した分析ではプロダクトの品質を客観的に検査することが難しい。また個人のスキルに依存した分析では、プロジェクト毎に“ばらつき”が発生するリスクも高くなる。

欠陥の分析は、テストを担当するチームのみではなく、開発を担当するチームとも共有する必要があるため、公開されている実用実績のある技術が適切であると判断した。そこで有識者などとの情報交換をすすめ、IPA/SEC が公開している事例を参考にした。

今回の適用範囲としては、要件定義工程を外した設計・開発、テストの全工程とした。
要件定義工程を外した理由としては、要件定義工程で検出される不具合に対しては、まだ我々では ODC を使用して明確に分類することが難しく、判断作業に対して大幅な時間を割かなければいけないと考えたからである。

また事例でも設計・開発工程からリリースまでの ODC の使用例が示されていたため、これを十分に参考にできると考えたからでもある。

4. 欠陥分析作業の課題と ODC を使用するに至った事情について

欠陥の分析作業は、設計・製造後の検査工程において、検査リーダーがその内容と品質状況を

顧客あるいは社内開発部門へフィードバックしている。その品質状況の内容は、不具合を質的・量的な観点で収集した情報の分析結果である。

これまでは不具合検出状況管理図を作成し、その量的な分析を実施してきた。しかしながら、それだけでは品質判断ができず、さらに質的な分析が要望されるに至り、不具合の内容をみて、原因分析を行うようになってきている。

しかし、検査者によっては、バグの内容解釈がそれぞれ異なり、果たして正しい分析ができているのか、設計側と問題になることが多いことが課題である。

さらに深刻なことは、不具合件数が大量になると工数の圧迫が必至であり、すべての不具合原因を調べていくのは不可能であることだ。

こうした正確性やデータ量の問題を限られた工数のなかで、どのように欠陥分析を行えば効果的であるのかが、大きな課題となっていたのである。

このような状況から、まずは分析能力が高い検査者を育てることが必須と考えたが、一方で誰が分析しても同じ結果になるような方法がないかを模索する中で、欠陥分析手法のひとつである、ODC 分析を導入するに至った。

5. 問題に対する対策を試行、実施

今回の改善作業で採用した ODC 分析とは、欠陥を多角的な属性（意味）に従って分類し、分析することであり、日本語では「直交欠陥分類」や「交差欠陥分類」と訳されている。

分類には８つの項目属性があり、それぞれが重ならないように定義されており、欠陥の性質と量から次のアクションを考えるための技法である。

米国 IBM 研究所で開発され、現在、V5.2 が最新である。[2]

今回の ODC を使用した取り組みは前述のとおり、設計品質向上を目的としており、大きく設計工程と検査工程に分けて、検証を実施した。

そして、欠陥情報を用いて品質向上を図るため、まずは必要な欠陥データの精度向上から検証していき、製品リリース時の判断にも使用できることを目指した。

実施の内容については、大きく検査者と設計者に分け、以下にその主要な観点を記載し、この流れでそれぞれの改善活動の詳細を記載する。

■欠陥データ信頼性の向上

・検査者による ODC 項目属性の設定

・バグ票の記載データを均一化し、信頼性を向上する

■欠陥分析の精度向上

・設計者による ODC 項目属性の設定

・分析の方法を均一化し、課題を視覚化する

また、この取り組み自体の達成状況は、現在も継続して改善中であり、以下、定性的な効果確認を中心に事例記載する。

6. 欠陥データ信頼性の向上

ここでは、検査者が ODC を使用した取り組みの実施、および実施上の問題、対策と工夫を記載する。

まずは、ODC 項目での分類ができるように、バグ票の見直しとともに、ODC 項目属性の追加設定を検討した。

ここでは検査者が分類設定できることを主眼とした項目を追加し、ODC 分析が行えるバグ票の設計を実施した。これらは検査者テンプレートとして、分析軸の項目属性をリスト設定したエクセル表を用意した上で、不具合の分類をはじめた。

表 73-1　検査者テンプレートの主要項目

分類種別	項目	items
ODC	検出工程	Defect　Removal　Activities
ODC	トリガー	Triggers
ODC	画面レビュー	Triggers--　GUI　Review
一般	品質特性	ISO/IEC　9126-1
一般	機器分類	Instruments
一般	機能分類	Functional module
一般	重要度	Weightings

表 73-1 のような分類項目を使って、テストで検出された欠陥について検査実施者が ODC に基づいた項目属性の設定を実施した。その結果、これまでのように設定項目に空白が多かったバグ票から、これまでより情報量があり、かつ検査実施者間でバラツキが割合に少ない欠陥情報に変換することができた。

しかし、原因系である「トリガー」は、検査者によって入力した属性にバラツキが出てしまう課題は簡単に解決できず、設計側で必要と思われるアウトプットとしては精度が低いのではないかという疑念が残ってしまった。

そこで検査者によって異なってくるデータの分類作業のバラツキをなるべく小さくする工夫を考えることとした。欠陥の項目属性設定の正確度を上げるためには、分析の手順を論理的に組み立て、属人的にならない方法を検討した。

属人的な項目属性設定にしないためには、検査者にとってわかりやすい分類を作成し、どの項目を選択すれば良いか、先進事例などを基に、対象とした開発案件に適用できるようにカスタマイズした項目属性と検出した欠陥を選別するフローチャート的な手順書を作成し、検査者への説明も適宜実施した。

この「トリガー」によって、検出された欠陥の発生要因を推測できるのだが、何度か試行した結果、どうしても検査者によって、項目属性設定にバラツキが出てしまっていることがわかってきた。やはり、開発者でないと、欠陥の程度を推測するための項目属性を設定するのは難しいのではないか、と考えた。

検討の末、「トリガー」の代わりに ODC 分類の「市場影響」を使用することを試行した。

この「市場影響」は、検出された欠陥が市場に流出してしまった場合に、ユーザが受ける影響を推測する項目属性となっており、検査者にとってはこちらの方が理解しやすい利点があった。

そして、その結果を確認しているうちに、「市場影響」は品質特性とよく似た属性を持っていることに気がついた。

「品質特性」は以前からテスト設計で使われており、検査者にとってはわかりやすい項目属性である。この品質特性および品質副特性を分析軸に追加し、それによって品質の網羅性を確認するための属性値として使用することとした。

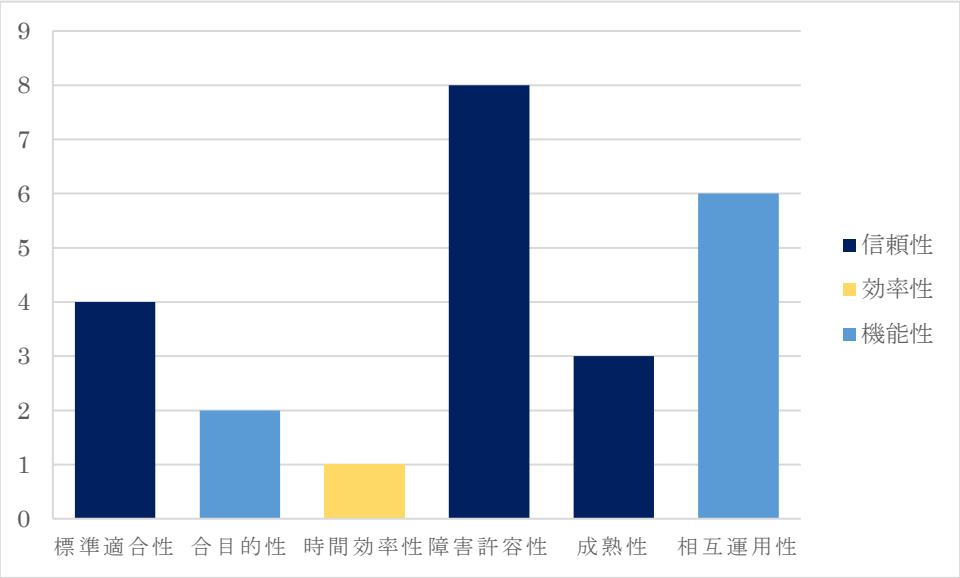

図 73-1　品質特性の分布

上記図 73-1 のグラフでは、品質特性とその下位の品質副特性を確認することで、今回の品質傾向を見ることができる。

傾向としては、「信頼性」→「障害許容性」が第一に挙げられている。

これは、障害が起きてもソフトウェアが機能を提供し続ける観点のテストで多く不具合が検出されていることになる。したがって、検査としては品質を担保するために、この観点のテスト項目は減らしてはいけないことがわかる。

この分布からは、何の要因で間違いが起きたのかはわからないが、品質特性、品質副特性の項目値の分布から、ユーザが直接受けるインパクト、リスクなどといった分析が行え、品質の確保を判断することを目的として利用することができる。[3]

7. 欠陥分析の精度向上

ここでは、設計者が ODC を使用した取り組みの実施、および実施上の問題、対策と工夫を記載する。

欠陥データ信頼性の向上では、検査者によりテスト検出された欠陥について ODC 分析を適用

し、一定の検査と改善を得ることができたため、次に設計者による ODC 項目属性の設定を開始した。

設計者では、設計工程において、レビュー時に欠陥を分類することを始めた。ODC 手法に則し、欠陥データはレビュー議事録を基に、設計に特化した分類を各欠陥に対して項目属性の設定を行った上で、その分類毎の集計をグラフ化した。この分類集計することによる量と質の傾向から、プロジェクトの陥りやすい欠陥の傾向を見ることが可能になってきた。

これによって、改善案もその結果を検証した上で、順次進めていく道筋をもてるようになった。見える化に際しては、欠陥データに対する効果的な分析を行うため、交差する分析（DeepDive）の考え方を導入した。

DeepDive 分析は、ODC 項目で得られたデータをさまざまな組み合わせで交差させることにより、設計や検査の問題点を分析するための手法である。

ここではどの分類項目とどの分類項目を組み合わせればよいのかが、はじめは課題となったが、バグデータの分類項目単位に量の集計を行った後、分類項目の関係性に注目することにした。欠陥データの分類項目の組み合せにおいては、項目が空白ではなく、属性が設定されているデータ件数を集計し、件数が多い分類項目同士との関係を優先することで、事実性の高い傾向分析になると考えた。

この多角的な視点が提供できる分析手法を使い、交差させるための分類項目群の組み合わせを検証した。その結果として有効であったものを標準的な交差分析とすることにより、見える化ツールとして活用している。

以下に、いくつかその事例を示す。

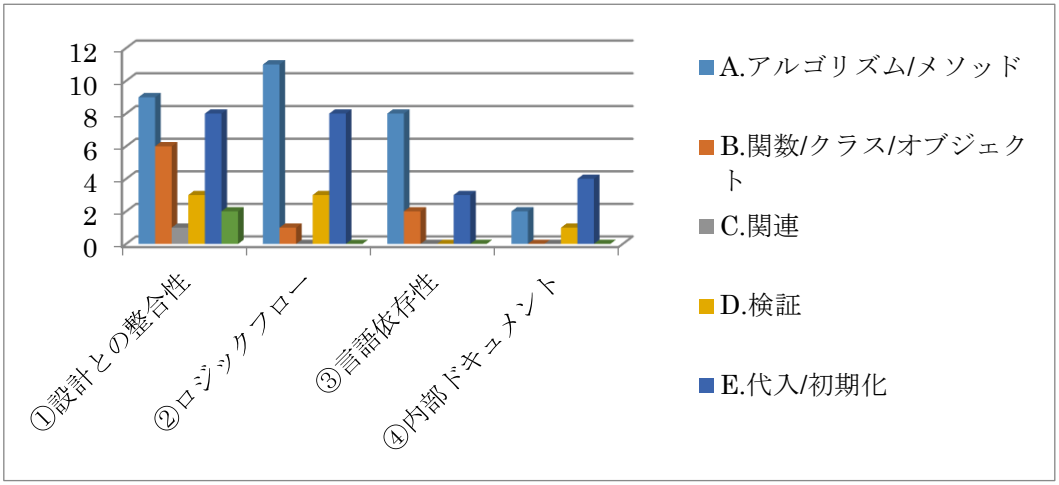

図 73-2　欠陥原因の分布

上記図 73-2 のグラフは、レビューのトリガーと欠陥タイプを交差分析したものである。これは設計に関わる当該プロジェクトの技術的な弱点の傾向を確認したものである。

ふたつ以上の分類項目を交差させたグラフで表すことにより、欠陥の分布から偏りがわかり、レビューでの指摘観点や欠陥タイプとの関係から、どこに問題があるのか特定し、必要に応じて是正措置をとることができる。

次に、ユーザ操作画面における欠陥分析の工夫を示す。

あるプロジェクトでは、ユーザーインターフェイスの不具合が多く、これらの欠陥原因を突き止め、是正策を出していく必要があった。これまでは欠陥分析をした設計リーダーに原因追求の方法は依存しており、その分析のまとめ方がプロジェクトによってバラバラであったため、その傾向や対策の方法については共有されないものとなってしまっていた。

そこで、GUI（画面）の欠陥を扱うことができる ODC の拡張トリガー（Extentionts）を利用することとした。この拡張トリガーでは、「設計との整合性」「画面部品の外観」「文字列」「入力デバイス」「ナビゲーション」「画面部品の動き」といった項目属性が用意されている。

拡張トリガーの詳細については、森龍二氏の「ODC（直交欠陥分類）概説」を参考とさせていただいた。[4]

この項目属性を利用して、ユーザ操作画面のテストで検出されたバグを分類集計した。

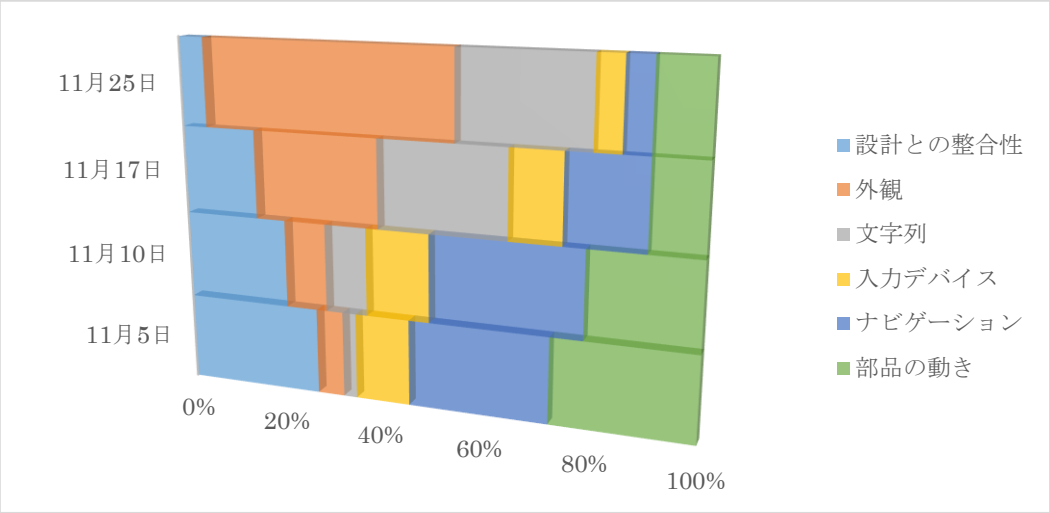

図 73-3　画面欠陥の要因の時系列推移

この項目属性によって、時系列で画面の欠陥要因を見ていくと、「設計との整合性」や「ナビゲーション」といった重たいバグが解消されていっているのがわかるようになった。

さらに、これらの不具合は人為的な間違いに起因するものが多いと考えたため、ODC 属性の「起因」を分析軸に追加することとした。

しかし、検出した欠陥に属性を当てはめてみると、その他になってしまうものが多く、「間違い」「漏れ」以外の選択肢が必要であることがわかってきた。

そこで、Diane Kelly and Terry Shepard が考案した、ODC を詳細化する「ODC-CC」を参考にして、「起因」の項目属性を追加することとした。[5]

表 73-2　ODC-CC の「起因」項目属性

項目属性	items	説明
漏れ	Missing	抜けや漏れに起因する欠陥。
誤り	Wrong	間違いや簡単なミスに起因する欠陥。
曖昧	obscure	不明瞭で曖昧な設計に起因する欠陥。
矛盾	inconsistent	一貫性がなく、矛盾した設計、実装に起因する欠陥。
余分	Superfluous	余計な設計、実装に起因する欠陥。

この項目属性で欠陥を分類することにより、人為的ミスがどのような要因から起きているのかが確認できる。また、これまで単純に「ミス」としてきた不具合を一段階掘り下げることができ、対策としても属性毎に検討が進められようになった。

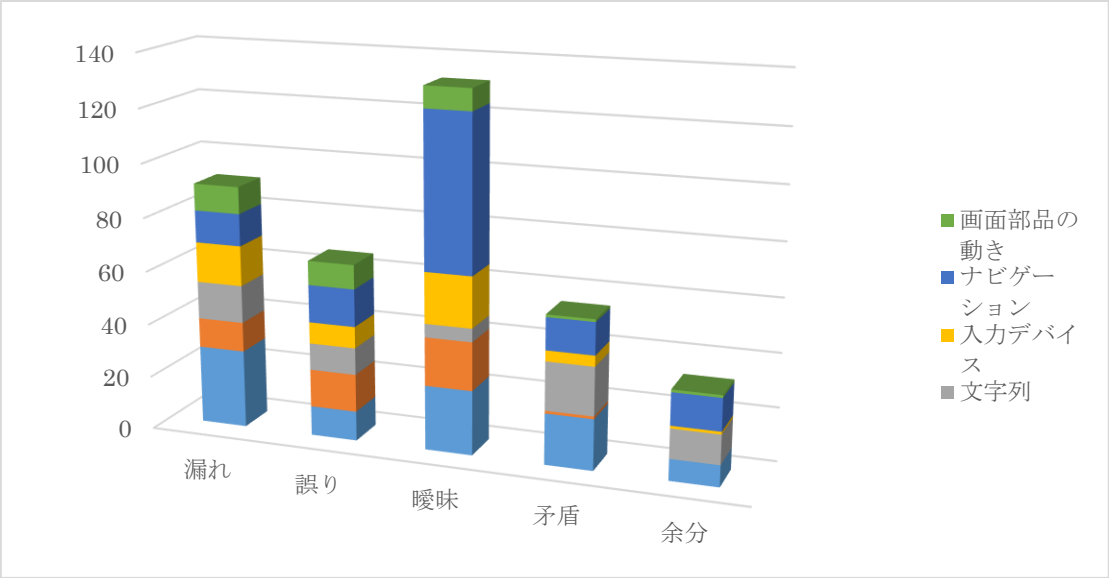

図 73-4　画面欠陥の起因分類

上記図 73-4 のグラフは、GUI のトリガーと起因を交差分析したものである。これは設計に関わる当該プロジェクトの人為的なミスの傾向を確認したものである。

画面設計に「曖昧」な個所が多かったことがわかる。曖昧さは設計記述において、解釈する人によって解釈の仕方が異なるような表現で記載されていることが多い欠陥である。そしてさらに、UI としての「ナビゲーション」に弱さがあることがわかる。この弱点の改善策を検討していくことで、当該プロジェクトの開発者のスキルが上がっていくことが期待できる。

上記のように、交差分析の結果から、設計者の弱点やプロジェクトの傾向、品質の作りこみ状況を確認することができるようになった。[6]

8. 達成度の検査、取り組みの結果

当初の目的である、リリース日の変更が必要なプロジェクトはまだ無くなったわけではないが、改善傾向にあることがプロジェクトサマリから見ることができるようになってきた。

また、検査では重度の欠陥検出は少なくなり、欠陥数そのものが減少したことにより、欠陥の平均除去コストが改善されているプロジェクトも出てきた。

欠陥除去コストは、ドキュメント修正＋モジュール改修＋回帰テストの工数を計測したものだが、加えて検査者からの修正依頼・返却期間が短縮され、テスト期間の正常化につながったと思われるケースも見られている。

もちろん ODC 分析の成果だけではなく、開発部門やプロジェクトチームの改善意識によるものでもあるが、改善成果の一助になっているとの現場からの声も聞いている。

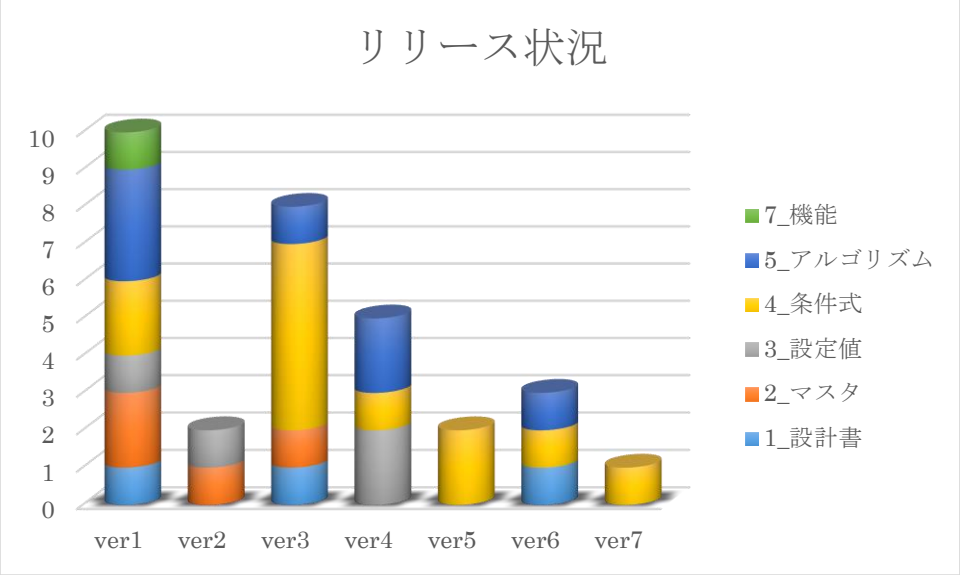

図 73-5　欠陥タイプでの品質判断

上記図 73-5 のグラフはリリースバージョン毎に欠陥タイプの検出を見える化したものである。バージョンが上がるにつれて、機能やアルゴリズムといった影響範囲の大きい欠陥が減少しているのが確認できる。

このように、品質管理部としても ODC 分析の結果については、リリース時の判断データとしての利用も始めている。

リリース前に欠陥の原因を分析することにより、リリースできる品質になっているかを推測するためのメトリクスの一つとして機能しはじめている。

9. 今後の取り組みについて

今回は、問題のあった組み込みソフトウェアの派生開発プロジェクトに合わせて改善活動を実施してきた結果を報告している。現在はこの改善成果をまったく別なプロダクト開発に適用していけるのか、検証しているところである。

今後、ODC 分析自体の活用に関しては、工程毎のトリガーの活用や分析可能性を探ることである。また、時系列データの蓄積による傾向や分析も今後は可能になってくると考えている。そして、最終的にはソフトウェアの品質を担保する説明根拠とすることを目標としている。

しかし、課題として大きいのは、こうした欠陥の見える化された後の対策や再発防止策の検討そして提案である。これに関しては、分析者の能力に依存してしまうところが大きい。設計改善の提案は、設計部門に任せるというやり方もあるが、その場合、自主的に動いてもらわないとその先に進まず、単なる分析提示で終わってしまう。

したがって、設計改善をある程度、提示できる分析資料の作成が行える人材の育成が必須であり、その過程でもっとわかりやすい分析のプロセスが生まれてくるものと考えている。

参考文献

[1] 山崎 隆：ODC 分析による欠陥除去と品質の成熟度可視化～医療機器の安全性・高品質性を担保するために～、独立行政法人情報処理推進機構（IPA）、2014、https://www.ipa.go.jp/files/000040829.pdf

[2] IBM：Orthogonal Defect Classification v 5.2 for Software Design and Code、IBM、2013、http://researcher.watson.ibm.com/researcher/files/us-pasanth/ODC-5-2.pdf

[3] 小島 義也：評価者による ODC を使用した不具合分析の現場展開～属人化を排除していく試み～、JaSST ソフトウェアテストシンポジウム 2015、2015、http://jasst.jp/symposium/jasst15tokyo/pdf/A5-2.pdf

[4] 森 龍二：ODC（直交欠陥分類）概説。JaSST ソフトウェアテストシンポジウム 2015、2015、http://jasst.jp/symposium/jasst15tokyo/pdf/A5-1.pdf

[5] Diane Kelly and Terry Shepard：A Case Study in the Use of Defect Classification in Inspections、Royal Military College Of Canada、2001、http://dl.acm.org/citation.cfm?id=782103&dl=ACM&coll=DL

[6] 小島 義也：設計プロセスの課題を ODC 分析で改善してみた、SPI Japan ソフトウェアプロセス改善カンファレンス 2015、2015、http://www.jaspic.org/event/2015/SPIJapan/session1C/1C-2_ID004.pdf

独立行政法人情報処理推進機構 技術本部 ソフトウェア高信頼化センター（IPA/SEC）