

ハードウェア ODC 分析の試み Applying ODC for Hardware Development

キャノン株式会社 品質統括センター
Canon Inc. Global Quality Management Center
○瀬能 芳幸¹⁾ 仁科 秀一²⁾
Yoshiyuki Senoh Hideichi Nishina

Abstract This is an attempt to apply Orthogonal Defect Classification(ODC), this is a defect analysis method mainly used for software, for applying hardware development. Requirements for specification changes by hardware teams that happen in a later phase of development, will have huge risks for software development. We intend to reduce later changes from hardware teams using ODC. This paper describes how to apply ODC for hardware development, what the signature looks like and how effective ODC on hardware development is.

1. はじめに

既存の欠陥分析方法には信頼度成長モデルと根本原因分析がある。信頼度成長モデルは欠陥数の積算をグラフ化し収束状況を観察する。欠陥はすべて平等にカウントされるが、開発者は重要な欠陥とそうでない欠陥があることを知っている。一方、根本原因分析は一件々々、欠陥の原因の原因を突き止めるので分析に膨大な手間がかかる。また、同様の欠陥が他にないことを示すことは非常に困難である。

そこで、Ram Chillarege 氏は欠陥の開発プロセスに対する意味合いを分析できる ODC を IBM 社において 1992 年発表した。彼が示した方法論は、ソフトウェア開発に適用できる方法論である。日本の電機メーカーの製品多くは、ハードウェアとソフトウェアから成っている。ソフトウェア開発において、開発終盤ハードウェア開発チームからの設計変更依頼は、ソフトウェア開発に多大な影響を与えるが、ソフトウェア開発チームは、依頼を受けざるを得ない状況が多々ある。ソフトウェア開発で活用している ODC 分析をハードウェア開発に水平展開し、開発終盤の設計変更を少しでも減らせないか、というアイデアがこの論文のテーマである。

2. ODC 分析方法論をハードウェア開発に適用

2.1 ODC 分析とは

欠陥をさまざまな観点(重要度、修正モジュール別など)で分類し、品質状態やその開発のどこに問題があったかフィードバックを試みる取り組みは、ODC 分析が紹介される前から多くあった。

キャノン株式会社 Canon Inc.

神奈川県川崎市高津区下野毛 3-16-1 Tel: 03-3758-2111 e-mail:seno.yoshiyuki@mail.canon
1-2-1, Koenji-Minami, Suginami, Tokyo Japan

1)キャノン株式会社 品質統括センター 製品品質保証部 主席

Principal Engineer, Products Quality Assurance Div., Global Quality Management Center

2)キャノン株式会社 品質統括センター 製品品質保証部 製品品質保証課 主幹

Senior Engineer, Products Quality Assurance Dept., Products Quality Assurance Div., Global Quality Management Center

【キーワード：】 ODC 分析、品質管理

しかし、当時、確立された方法論はなかった。そこで Ram 氏は ODC を発表した。ODC は Orthogonal Defect Classification の略で、Orthogonal とは、直交ということだが、それぞれの属性値が重なり合わないということを示している。

ODC の主な属性は以下に示す通り。

- 開発プロセス上の意味合いを表す「欠陥タイプ」属性
- 検証プロセスが意図通りか測定できる「欠陥トリガー」属性

欠陥タイプは、それぞれ開発がどうであったか、欠陥トリガーは検証がどんな具合なのかを示している。それらの測定結果は時系列の比率として表現することができる。これらの比率の変化をシグナチャーという。あるべき姿と実際のプロジェクトの差を取ると、我々は開発プロセスの弱い部分や病巣を知ることができ、具体的な改善活動につなげることができる。

欠陥トリガーは検証の状態が分かる重要な属性である。タイプとトリガーを使用した組合せ分析を行うとより詳細な分析が実施できるが、議論を簡単にするため本論文では欠陥タイプを対象とする。

ソフトウェア開発の欠陥タイプ属性は図 1 のように定義されている。

欠陥タイプ属性	解説
代入/初期化(Assignment)	変数/定数値の代入あるいは初期化の欠陥
チェック/条件(Checking)	条件文での条件値、判定式の欠陥
アルゴリズム(Algorithm)	処理手順、アルゴリズムの欠陥
タイミング/順序(Timing)	順番制御/リソースアクセスへの排他制御/スレッドの同期処理の欠陥
インターフェイス(Interface)	ブロック/コンポーネント/プロダクト間、ハードウェアとソフトウェア間のコミュニケーションに関する欠陥
機能/クラス(Function)	要求仕様との不整合、機能性/グローバルデータ構造の欠陥
ビルド(Build)	ビルドプロセス、ライブラリ又はバージョン管理に関連する問題
ドキュメント(Document)	マニュアル、技術文書、表示されるメッセージの欠陥

注：上記のタイプ属性に対して「誤り」あるいは「漏れ」という限定詞が付与される。

図 1:ソフトウェアの欠陥タイプ属性

欠陥タイプの理想的なシグナチャーは図 2 の様になる。これは基本設計と詳細設計では設計レビュー指摘を、実装・単体テストでは実装レビュー指摘を分類し、結合テストとシステムテストは欠陥票を分類した比率をグラフ化したものである。

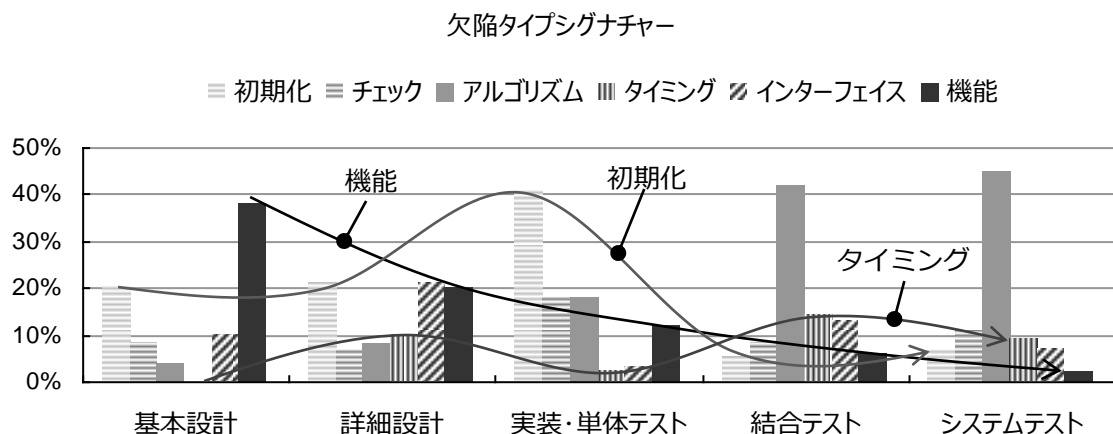


図 2:欠陥タイプシグナチャー

欠陥タイプ属性には重要度がある。「機能」は最も重要度が高い属性で、基本設計フェーズで多く指摘され、後になると減少すべきである。次に重要度が高い「インターフェイス」「タイミング」は詳細設計フェーズあるいは、結合テストとシステムテストで検出され、最終的には収束する。軽微な欠陥「代入・初期化」や「チェック」は、実装・単体テストフェーズで最も多く検出される。

理想のシグナチャーと実際を比較するとギャップが得られる。一般的傾向や過去のギャップデータから何が問題で、どう対策すればよいかを導き出すことができる。ただし、実際にはプロジェクトや製品の性質・プロセス種類によりあるべき姿のシグナチャーパターンは調整が必要なことは言うまでもない。

ODC 分析とは、正しく分類、分析できれば、開発や検証の問題の状況を捉えることができる方法論である。

2.2 適用において重要なこと

ODC 分析の欠陥タイプ属性では、開発プロセスの各段階で何がどの程度しっかり行われているかという事に立脚している。ODC 分析のコンセプトをしっかり押さえた上でソフトウェア開発とは別の開発に適用することが重要である。それを整理し以下の3点に配慮した。

1. 属性値が重なり合わない
2. 欠陥タイプの属性値は開発プロセスのタスクと関連し一貫している
3. 製品間で共通である

1つ目の「属性値が重なり合わない。」とは、ある欠陥タイプは、開発プロセスのある作業に関連しているので、属性値は重ならない、すなわち直交していることが重要である。

2つ目の「開発プロセスのタスクと関連し一貫性があること」とは、例えば、「タイミング」は、同時に実行される異なるスレッドの同期問題である。この問題が検出される機会は、スレッドを設計する詳細設計と、両方のスレッドを動作させることができる結合テストあるいはシステムテストにおいてである。従ってその欠陥タイプ属性はプロセスにまたがって一貫性がある必要がある。

3つ目の「製品間で共通である」は、他の製品に適用できなければ、方法論として展開できないので失格となる。

3. ハードウェア ODC タイプ属性の設計

ハードウェア開発に適応できる欠陥タイプを定義するため、ハードウェア開発プロセスを手掛かりに欠陥タイプ属性案を策定した。欠陥タイプ属性のセットに有効性があるか、2つ目の開発プロセス間で一貫性があること、3つ目の製品間で共通であることを検証するために、過去の実際の数種類のハードウェア開発における多くの欠陥票を用いて分類し、シグナチャーを検証した。その結果、以下の定義となった。

欠陥タイプ属性	解説
図面(Drawing)	部品図面化の際の配慮不足、ミス
部品(Parts)	部品の強度・材料・形状の問題
制御(Control)	ユニット、コンポーネントで発生する機能不良でパラメータ変更・条件等の制御などハードウェア変更なくファームで修正した問題
機構(Mechanism)	ユニット、コンポーネントで発生する機構不良で強度・剛性などハードウェア修正を要する問題
インターフェイス(Interface)	ユニット、コンポーネント間で発生する機能不良で、複数ユニットにまたがるコミュニケーションやインターフェイス仕様変更を要する問題
基本構成(Fundamentals)	設計思想・製品基本構成に影響する問題、仕様の漏れ

図 3：ハードウェアの欠陥タイプ属性

設計観点からハードウェアの欠陥タイプにもソフトウェアと同様重要度がある。「図面」は最も低く、下にリストされる属性ほど重要度が高くなり、基本構成が最重要となる。

4. ハードウェア欠陥タイプ属性のシグナチャー

ハードウェア開発の欠陥タイプ属性がこれらの定義で、適切に働くかは実際の欠陥票に当てはめそれらのシグナチャー(比率変化)によって開発状況と照らし合わせ読み取れるか検証が必要である。本活動ではハードウェア開発プロセスの製品検証フェーズの欠陥票を対象とした。

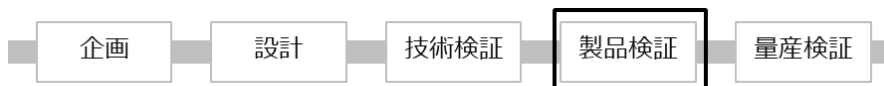


図 4：製品開発プロセス

数種類のハードウェア開発における多くの欠陥票を分類・分析し、あるべき姿を導き出し、それとのギャップ分析を試みた。

4.1 あるべきシグナチャー

製品検証フェーズに移行する際、設計レベルでの機能検証が十分であるべきである。それができた製品では、欠陥タイプは以下の傾向を示した。

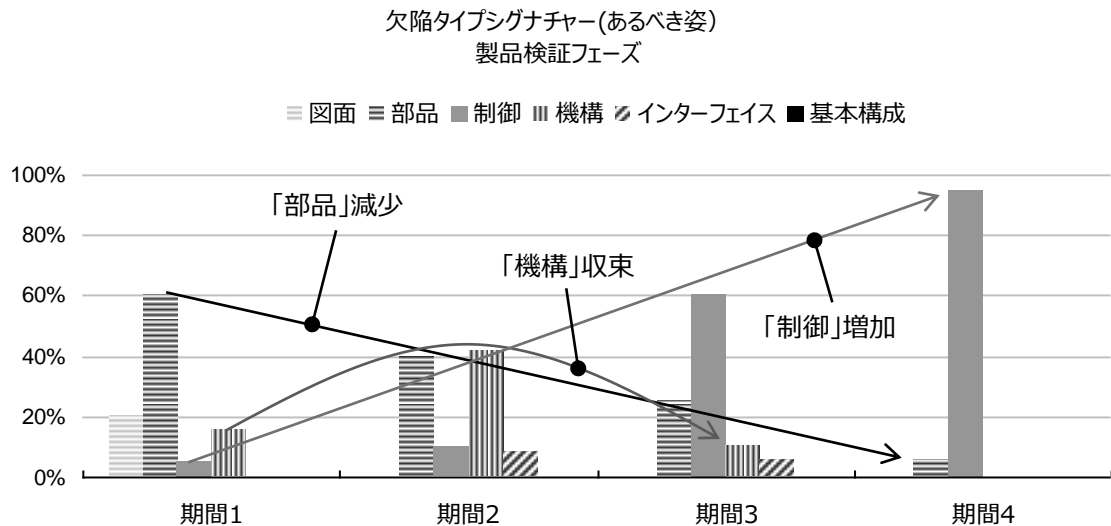


図 5: ハードウェア欠陥タイプシグナチャー(あるべき姿)

「部品」が高い割合を占め、徐々に減少し 製品検証フェーズ終盤には収束する。

「機構」は検証が進むと割合が増加し、後半には収束する。

「制御」は終盤に向け増加する。製品検証フェーズの終盤では「制御」における設定値や条件の修正が多くを占める。

製品開発において生じる問題はさまざまなものがある。それらの代表的なものの一つは開発後半に重大な設計変更が生じてしまったもの、またもう一つは新規性・難易度が高い開発である。こうした製品開発における問題についてこの方法を使うことで、あるべき姿とのギャップを取り実際の状況を的確につかみ、それらの対策につなげるように品質管理にいかに関与できるかを次節以降に示す。

4.2 重大な設計変更

製品検証フェーズに重大な設計変更が入ることは設計者にとってあってはならない事態だが、この例は、そうしたことが起きた場合の欠陥タイプのシグナチャーを示している。

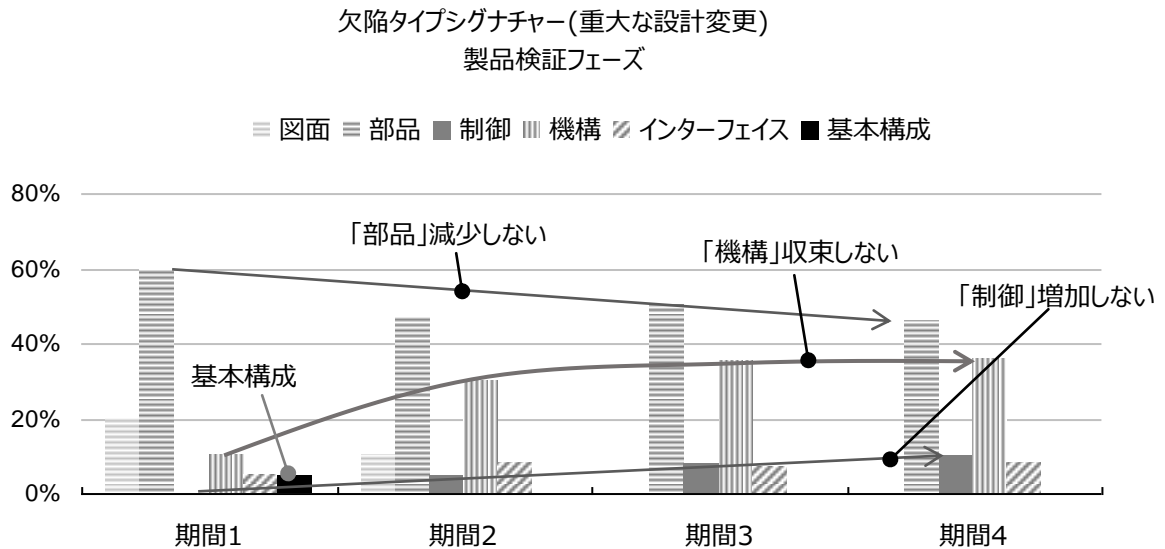


図 6：ハードウェア欠陥タイプシグナチャー(重大な設計変更)

「基本構成」とは設計思想・製品の基本に関わる重大問題である。これが製品検証フェーズで検出されることは望ましくない。「基本構成」の変更は多くの「部品」に影響があったと考えられる。

「部品」は、減少すべきだがこの例では減少しない。

「機構」は、中盤に向けて増加するのはよいが、その後収束しない。設計変更により製品機能が未成熟なためか、多くの「部品」、「機構」の欠陥が検出されている。この未成熟さが結果的に「制御」の欠陥検出まで達せず比率は増加しない。

「基本構成」の変更が「機構」や「部品」に影響していることが想定できる。

4.3 難易度が高い

新規性・難易度が高い製品は、欠陥数が多く製品検証フェーズに移る際も課題が残る場合が多い。開発部隊は慎重に進めるので、欠陥タイプシグナチャーは検証フェーズ前半にあるべき姿に近い形を取る。しかし、検証フェーズ後半にあるべき姿と乖離する。

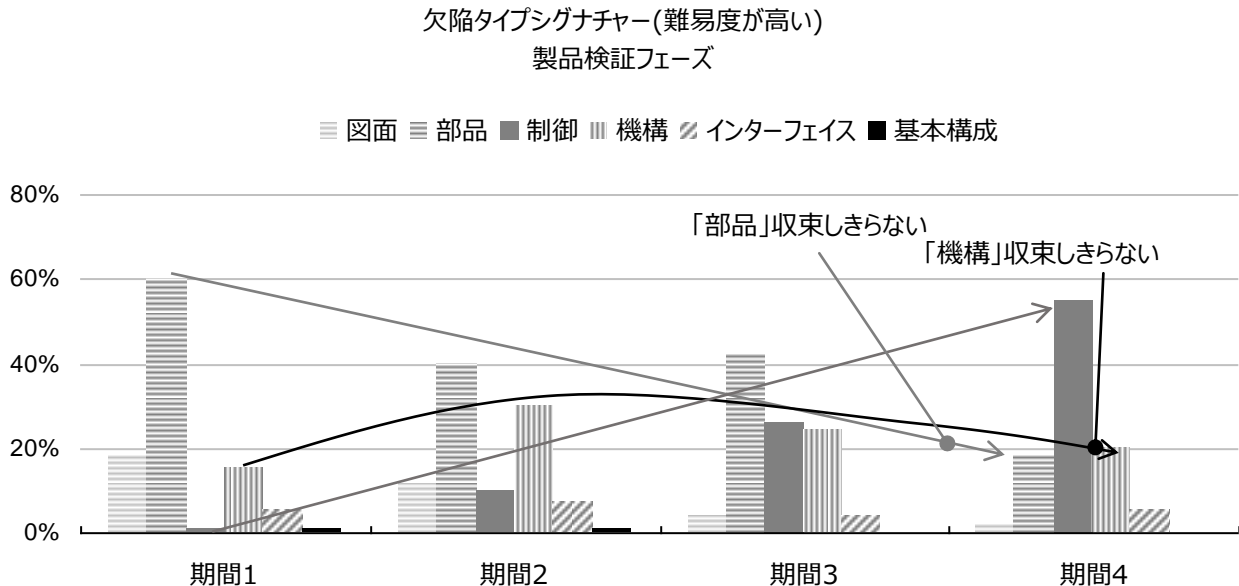


図 7: ハードウェア欠陥タイプシグナチャー(難易度が高い)

「部品」は減少するが、その後収束しきらない。

「機構」は一旦増加し、その後収束しきらない。

従って、製品検証フェーズ終盤「制御」が増加するが、難易度の高さからハードウェア修正が入り、終盤においても「部品」「機構」の欠陥タイプが残ってしまう。

5 考察

ここで示した欠陥タイプ定義におけるハードウェア開発欠陥タイプのシグナチャーは、あるべき姿に対してギャップを明確に示し、特徴ある動きを読み取ることができるので、ハードウェア開発における欠陥タイプを正しく定義できたと考えられる。また、これを活用すれば品質管理上で有効である。

重要度を考慮すると質的な把握ができる。従って、従来の信頼度成長曲線、根本原因分析に加え、ODC 分析を行えば質的考察が可能となるので、対策や判定に有効に利用できる。

5.1 おわりに

ここで示した欠陥タイプ定義に至るために実際の多くの欠陥票で分類し属性定義の検証とチューニングを数回繰り返す必要があった。

ソフトウェア技術者にとってハードウェア欠陥票の記述は難解であり、製品によって記載レベルがさまざま異なることが分かった。その上解決までの期間が長く、修正も様々な変遷があり最

最終的な修正でどれが有効であったか判定が難しく分類に手間取った。ハードウェア技術者の協力が無ければ、ハードウェア用の欠陥タイプを定義することは難しかった。

今後の課題として、設計、技術検証そして、量産検証フェーズについての欠陥に対する考察が残っており、早いフェーズから ODC 分析を適応した場合の検証を行うことが必要である。

この活動で定義したハードウェア開発用の欠陥タイプ属性定義は、引き続きより広い製品を対象として調整し精度を上げるべきである。

ハードウェア開発プロセスの質的な改善ができれば、開発の終盤でのソフトウェアに対する仕様変更が質的に改善できることを検証すべきである。

謝辞

本研究においてキャノン株式会社 福井隆之氏には多大な助言をいただき感謝の意を表します。また 分類属性の定義および欠陥分析に協力いただいた 石井誠氏、穂積俊春氏、福原亮祐氏、久保田隆幸氏、豊貴絵氏に感謝の意を表します。

参考文献

- [1] Ram Chillarege “Orthogonal defect classification – a concept for in-process measurements”, IEEE Transactions on Software Engineering, vol. 18, Nov 1992
- [2] Kathryn A. Bassin, Theresa Kratschmer, P Santhanam “Evaluating Software Development Objectively”, IEEE Software, vol 15, Nov/Dec 1998
- [3] IBM “Orthogonal Defect Classification v5.2 for Software Design and Code, Sep 2013
- [4] 杉崎 眞弘, “ODC 分析の基礎 ソフトウェア開発の見える化 – 不具合分析から見えるプロジェクトの健全さ –”, 日科技連 ODC 研究会第 3 期第 1 回, Jun 2019