活動報告：技術文書WG

ODC分析の教科書（概説）

報告内容の主旨：

ODC研究会　技術文書WGでは、本年度活動テーマ「ODC分析の教科書を作る」として、
ODC分析について、考案された背景、分析技術コンセプトに始まり、分析手法、評価手法を
事例を交えて解説する技術文書を作成してきました。
その内容について概説します。

この技術文書は、これまでODC分析に関するまとまった書籍がない中、Webにある断片的な
情報をもとに試行を志す方々への教科書として、ODC分析の基礎知識、実践手法が学習できるよう、
まとめたものです。

2019/3/20

ODC分析研究会　運営委員
杉崎 眞弘

ODC分析の教科書　概要

II.　本書について

■ 本書の目的

本書は、Orthogonal Defect Classification（以下、ODC分析）についての基礎編として、そのコンセプト、分析・評価手法、効能について解説したもので、一応にODC分析についての基礎知識が得られることを目的とする。

■ 本書の想定する読者

ソフトウェア開発に携わる開発/試験技術者、品質評価者、プロジェクト管理の方々、またプロセス改善に関わる方々とする。

■ 本書の範囲

下記参考文献にある、IBM社におけるODC分析着想から手法化にいたる論文をもとに、まずはODC分析のオリジナルを理解してもらえるように努めた。

III.　参考文献

1）Orthogonal Defect Classification – A concept for In-process Measurement
Ram Chillarege, Inderpal S. Bhandari, Michael J. Halliday, etc.,
IBM Thomas J. Watson Research Center
IEEE Transactions on Software Engineering Vol .18, No. 11, Nov. 1992©IEEE

2）Orthogonal Defect Classification v 5.2 for Software Design and Code 2013
(This document is made available for general interest and information purpose only.
IBM assumes no responsibility for its usage.) ©IBM

3）Experiences with Defect Prevention
R. Mays, C. Jones, G. Holloway, and D. Studinski
IBM Systems Journal, vol. 29, no. 1, 1990

目次

PART 1: ODC分析のコンセプト

まず、ODC分析についてのコンセプトは、次の言葉から始まる。

Defect Control is:

A measurement and analysis methodology based on Orthogonal Defect Classification to gain insight and direction for improving software quality.

ソフトウェア開発において、その進捗を妨げる主たる要因は不具合である。
その妨げを低減するには、「不具合を制御する (Defect Control)」ことが重要であると考える。

ここで、「不具合を制御する」とは、
現状このまま推移すると、この先で起こることが予測される不具合に対して、適切にアクションを取ることで、未然に防いでいく(controlする)ことである。
と、解釈する。

すると、上記の言葉は、次のように読める。

「不具合を制御する (Defect Control)」には、

ソフトウェア品質改善への洞察と方向性を得るための計測方法と分析方法論が必要で、その元となるのが ODC (Orthogonal Defect Classification)**での不具合の捉え方である。**

膨大な不具合についての研究から気づき、探り当てたこと

■ **ODC** (Orthogonal Defect Classification) **の意味**　(原文訳)

“Orthogonal（直交）”という言葉は、数学関係者でない人には奇異に聞こえるかもしれない。幾人かの人たちは、直交する線を頭の中でイメージするかもしれない。
この言葉を使う訳は、
１件の不具合の位置付けを説明することは、直角に交わる壁と床で囲われた部屋の中で、ある物の位置を示すのに壁や床からの位置を使って説明するのに似ているからである。
（下図　図-1.3参照）

“Defect（不具合）”という言葉は、以前から使われていて、「瑕」または「欠陥」のことである。

“Classification（分類）”という言葉こそ、この議論の核である。
何かを分類(classify)するとは、種類ごとに振り分けることである。
これまで不具合は研究されてきているが、それらはほぼ市場での信頼性の予測ぐらいのことでしかない。

我々の研究の成果として、意味論的に不具合を分類(classify)し、分析する事によって、不具合を制御(Defect Control)し、適用する**開発プロセスの改善への提言を生み出す**可能性があることを、探り当てることができた。

ODC分析の目的は、開発プロセスの改善にある。

PART 1: ODC分析のコンセプト

- **ODC** (Orthogonal Defect Classification)**での不具合の捉え方**
  (イメージ)

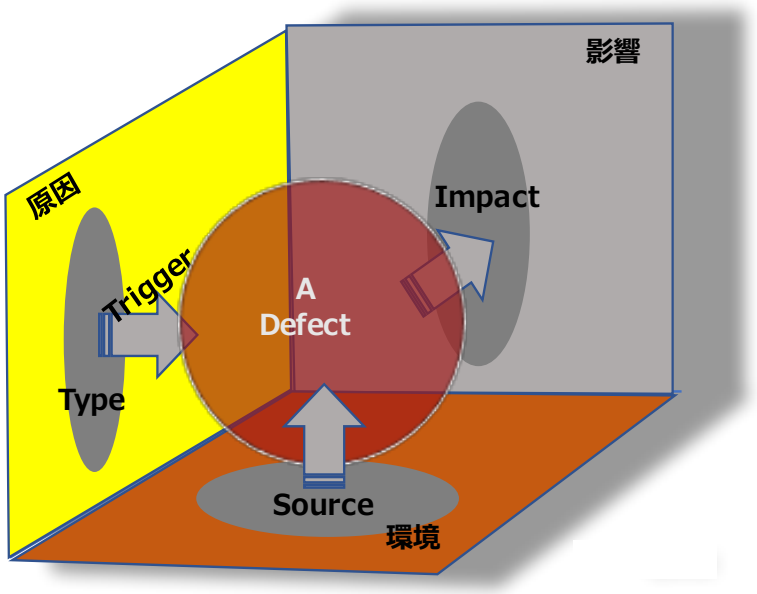
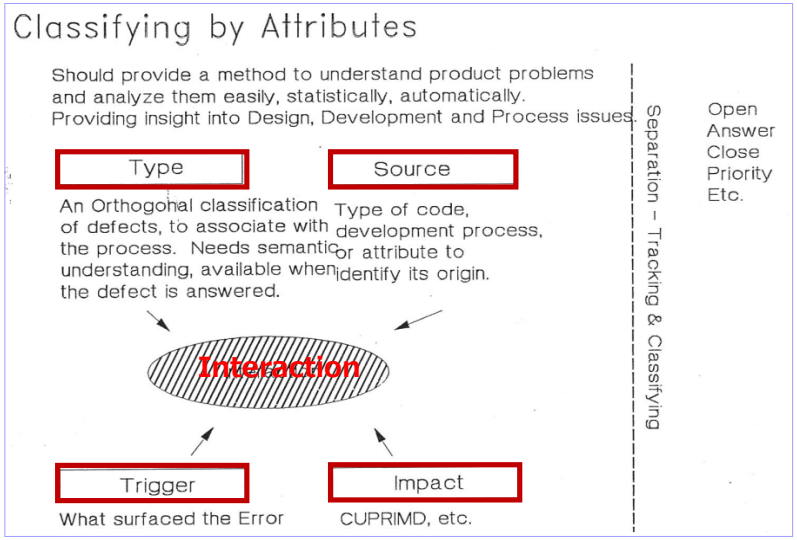


図-1.3 １件の不具合とその側面との位置関係

- **不具合について見えてきたこと― 不具合の属性**



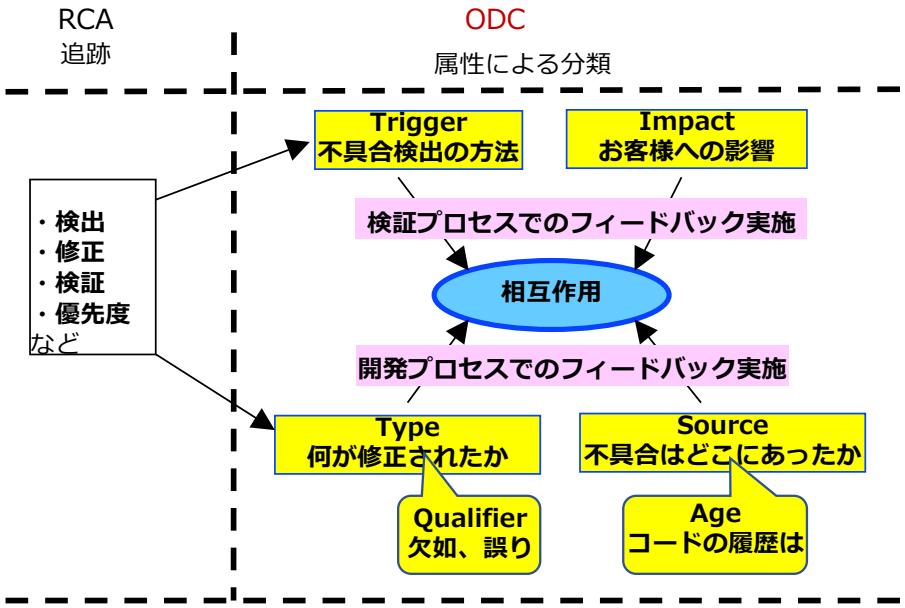
参照：IBM Orthogonal Defect Classification

個々の不具合には、それぞれ素性があり、
それらが相互作用して１件の不具合を形成している。
その素性は、４つの属性で表わせる。

ODC分析で定義する４つの不具合属性：
- **タイプ：**プロセスに関わる不具合を示唆する属性
- **トリガー：**不具合を表面化した動機
- **ソース：**不具合を含んでいたコードあるいはプロセスの箇所
- **インパクト：**ユーザ・顧客への影響（品質特性）

PART 2: ODC分析手法 (ODC不具合属性の説明)

■ ODC分析：４つの不具合属性による分類



参照：IBM Orthogonal Defect Classification

- 多次元の因果関係データをもとに分類・分析する。
- 開発を通じて同一の分析方法でおこなうようにすることによって、共通の認識・理解を図ります。
- 同一分析方法で開発対象、組織間を通じての整合性を保つようにします。

■ ODC分析の手法と不具合属性

Classifying by Attributes (不具合属性による分類）を手法とする。

不具合の追跡(tracking)は行わない。追跡には意味論に欠けるからである。
分類(classifying)には設計、テスト、プロセスの問題点を得ることができる。
そのための分類種別を、**属性(attribute)**と呼ぶ。

ODC分析では、以下の**４つの属性**が定義されている。

タイプ (Defect Type)
不具合の意味論理解に影響力のある属性はタイプ(defect type)である。
意味論にもとづいて分類され、プロセスに関する示唆を含んでいる。
この属性の選択肢をバリュー(value)と呼び、**お互いが直交する** 。
この属性の適切なバリューの選択は、通常はその不具合が開発者によって解決（修正）された時に明らかになる。

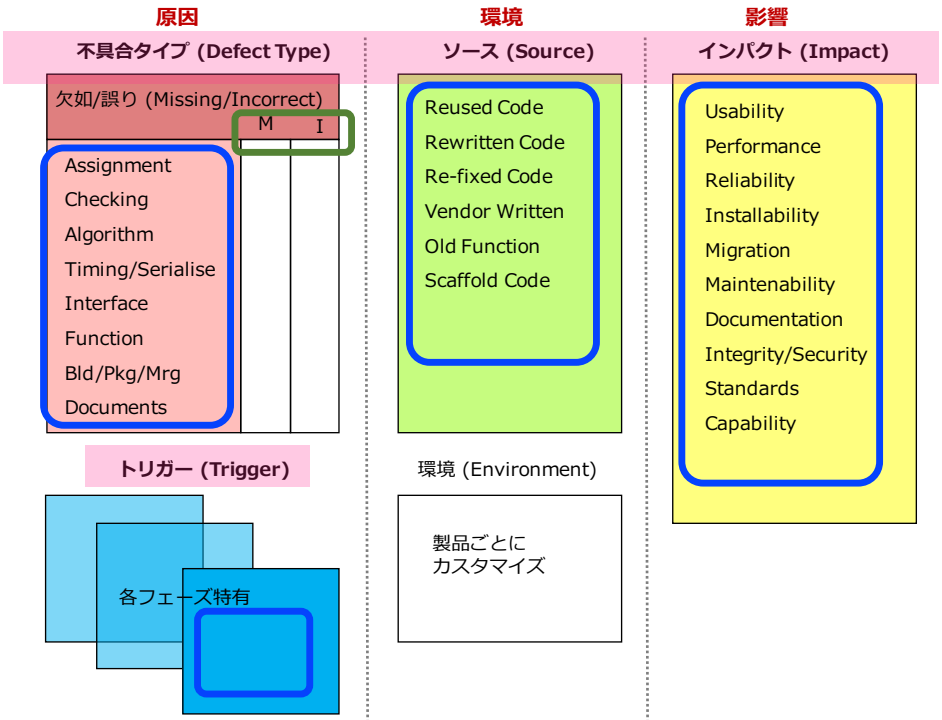
トリガー(Defect Trigger)
不具合が表面化した原因（状況）をトリガー(Defect trigger) と呼ぶ。
その適切なバリューの選択は、テスターによって行われる。この属性は**工程固有**となる。

ソース(Source)
不具合が摘出されたコード・プロセスの分類の属性 をソース(source)と呼ぶ。
このバリューの選択は、この不具合を修正した開発者による。

インパクト(Impact)
お客様への影響は、インパクト(impact)属性によって分類される。
品質特性により分類される。 そのバリューの選択は、テスターがふさわしい。

PART 2: ODC分析手法について（ODC不具合属性の説明）

■ **ODC不具合属性群** (属性と選択肢 Attribute/Value/Dominant)



参照：IBM Orthogonal Defect Classification

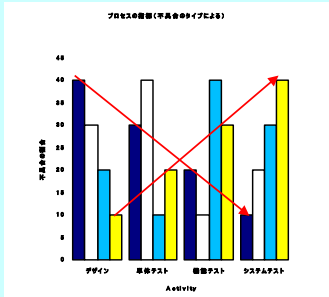
PART 3: ODC分析の適用に際して（理解しておくべきこと）

■ **開発プロセスと不具合の関係**

ODC分析の適用手順

1. 不具合を、不具合属性毎に分類し、その分類結果・分布を集計する。
2. 集計結果とプロセスの期待とを比較・分析する。
3. 発見された”ずれ“の原因から示唆される必要改善策を導き出します。

期待される不具合の分布

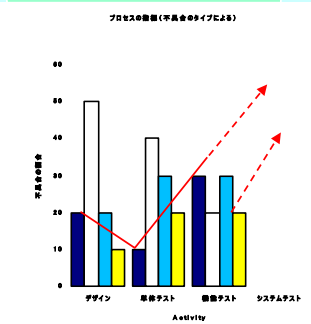


開発プロセスの期待
これをプロセス・シグネチャーと呼ぶ

“ずれ”の原因を分析

工程改善に必要な改善策を導き出す。

現状の不具合分布



なぜ”ずれ”を問題にするのか？

PART 3: ODC分析の適用に際して（理解しておくべきこと）

■ 開発プロセスの特性（シグネチャー）とODC属性分布との関係

開発プロセスの各工程ではそれぞれ目的・主眼、期待が定義されており、
その通り実行することで、最終的に全開発工程を通じて、開発対象を
あらゆる観点から設計・検証できるように設計されているはずである。

したがって、各工程で検出される不具合も、検証される事項に沿った
属性の不具合であるはずである。これをプロセス・シグネチャーと呼び、
プロセスの期待である。
例えば、不具合タイプと検出工程の関係は下図のような関係になる。

Phase / Defect Type	HLD	LLD	Code	Unit Test	Func Test	Sys Test
Assignment			X	X		
Checking			X	X		
Algorithm			X	X	X	
Timing/Serialize		X				X
Interface		X	X	X	X	X
Function	X				X	

参照：IBM Orthogonal Defect Classification

- シグネチャーは、開発プロセスに依存した固有の期待値（こうあるはず）である。
- シグネチャーを定規として、現実の状況に差異がある場合、「何かおかしい？」とする。
- その「何かおかしい？」を分析することで、そこから示唆される改善策「何をすべきか」を策定して、実行することが、プロセス改善につながる。

■ ODC分析の効果的な実施タイミング

ご清聴　ありがとうございました。

ODC分析を定期的、継続的に実施することで、
- 開発プロセス改善の継続
- 市場対応の早期化
- 次期製品の顧客満足度向上

を、図る。