10.4. 欠陥分析

欠陥分析の目的

欠陥分析はその結果を以下のように利用するために行います。

- ユニットテストを止めて前工程に戻らなくてはいけないかを判断する材料
- 次工程に進んでもよいか、対策をとるかを判断する材料
- 次プロジェクトの改善

欠陥分析の観点

欠陥分析は、以下の３つの観点で行います。

欠陥の種類

欠陥の種類(論理、インターフェイス、単純、冗長、コメント、その他)について分析を行います。
欠陥の種類の種類には次のものがありました。([参照]3.1. 欠陥の種類)

種類	意味	対応
論理	処理ロジックに誤りがある欠陥(設計の欠陥)	修正
インターフェイス	外部へのアクセス、インターフェイスの間違いなどの欠陥(設計の欠陥)	修正
単純	単純ミスによる欠陥(プログラミングミス)	修正
冗長	正しい結果が得られるが、冗長性のある処理	修正
コメント	コメントに不備がある欠陥	修正
その他	上記以外の問題がある欠陥	検討

欠陥を作り込んだ工程

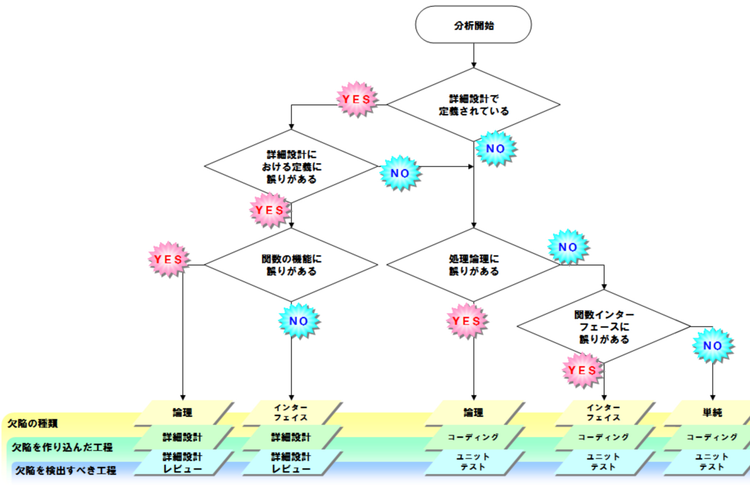
どの工程(基本設計、詳細設計、コーディング、ソースコードレビュー)にて作り込んだ欠陥かを特定します。

欠陥を検出すべき工程

どの工程(基本設計、詳細設計、コーディング、ソースコードレビュー、ユニットテスト)にて検出すべき欠陥であったかを特定します。

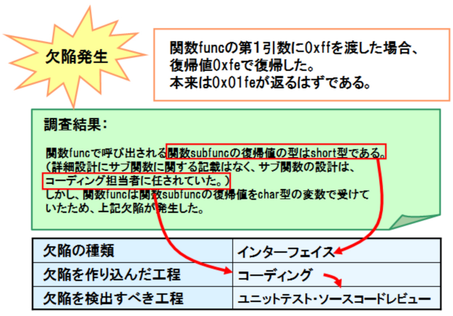
欠陥分析の手法

欠陥分析の流れを図でまとめると以下の通りです。



分析した結果は「インシデントレポート」に記入します。

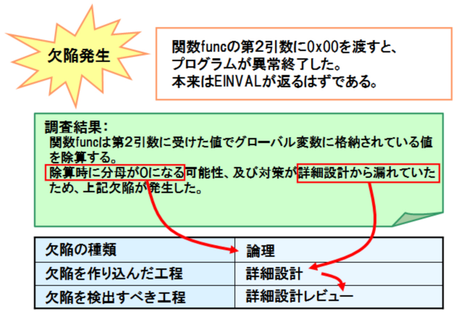
欠陥分析の例①



本カリキュラムでは、テスト対象関数と他関数の係わりで発生した欠陥を「インターフェイスの欠陥」と分析しています。
テスト対象関数の復帰値が異なっていても、関数内の要因であれば「単純ミスによる欠陥」や「論理的な欠陥」と分析しています。

- 欠陥の種類
  関数subfuncの復帰値の型とそれを受ける変数の型が合っていなかった。 →インターフェイスの欠陥
- 欠陥を作りこんだ工程
  詳細設計に呼び出し関数の設計は書かれておらず、コーディング担当者に任されていた。 →コーディング
- 欠陥を検出すべき工程
  V字モデルより、コーディングミスはユニットテスト・ソースコードレビューで検出。

欠陥分析の例②



- 欠陥の種類
  除算時のゼロ割という処理論理が抜けていた。 →論理的な欠陥
- 欠陥を作り込んだ工程
  ゼロ割の観点を詳細設計時に考慮するべきだったが漏れていた。 →詳細設計
- 欠陥を検出すべき工程
  詳細設計のミスは詳細設計レビューで検出。