戻る 目次

# 3.3. ソースコードレビューとユニットテスト

2.4で説明したとおり、ソースコードレビューとユニットテストは検証する対象もＩＮＰＵＴも同じです。
すると、ここで一つの疑問が浮かびます。

「テスト対象が同じなのに、ソースコードレビューとユニットテストの両方を実施する必要はあるのか？」

ユニットテストのように、プログラムやシステムなどを動かしてテストする”動的テスト”に対して、ソースコードレビューは、動かさずにテストするという意味で”静的テスト”とも言われます。
ソースコードレビューとテストとの違いは、プログラムを動かして確認するかしないかという点にあり、ソースコードレビューとユニットテストも、コーディング工程において作成されたプログラムを、”動かさずに”テストするのか、”動かして”テストするのか、という違いにすぎません。

## どちらが優秀か？

ソースコードレビューはユニットテストに比べて検出可能な欠陥が多いため（実行されないコメントの指摘や実行では確認がとりづらい冗長コードの指摘、ユニットテストでは実施できないタイミングに関する欠陥の検出）、ソースコードレビューの方が優秀です。
ユニットテストはソースコードレビューの代わりにはなりません。

**ソースコードレビューの方が優秀**

**ソースコードレビューは…**

- **ユニットテストで検出できる欠陥は全て検出できる**
- **冗長・コメントの欠陥を検出できる（ユニットテストは検出できない）**
- **タイミングの欠陥を検出できる（ユニットテストは検出できない）**

**ユニットテストはソースコードレビューの代わりにならない**

## ユニットテストの必要性

これだけを考えると、確かに、ユニットテストを実施せず、ソースコードレビューだけを行っていればいいように感じられます。
しかし、ソースコードレビューには、大きなデメリットがあるのです。
実は、ソースコードレビューは、人間が目視で行うために安定せず、欠陥を見落とす可能性が高いのです。
また実際にプログラムを実行しないため、コンパイラ依存の問題もあります。
下に、ソースコードレビューのメリット・デメリットについてまとめました。

| | |
|---|---|
| **メリット** | 冗長・コメントのような、機械的には判定できないような欠陥を検出できる<br>タイミングに関する欠陥についても検出が可能である（他テスト工程では検出が困難） |
| **デメリット** | レビューアのスキルレベルによってソースコードレビューの品質が左右されてしまう<br>レビューアのスキルが十分であっても、先入観等により、欠陥を見落とす可能性がある |

このようなメリット・デメリットがあるため、ソースコードレビューだけでは、テストは十分とは言えません。

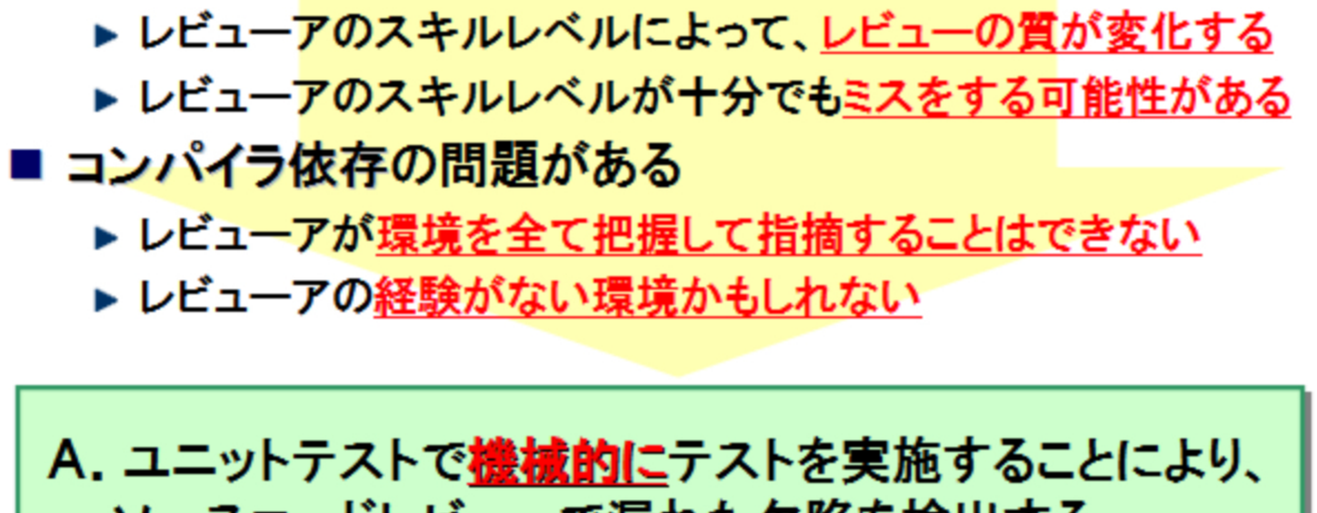
**Q. ソースコードレビューが優秀でも ユニットテストを実施する意義は・・・？**

- ソースコードレビューは目視で行うため…
  - レビューアのスキルレベルによって、レビューの質が変化する
  - レビューアのスキルレベルが十分でもミスをする可能性がある
- コンパイラ依存の問題がある
  - レビューアが環境を全て把握して指摘することはできない
  - レビューアの経験がない環境かもしれない

**A. ユニットテストで機械的にテストを実施することにより、ソースコードレビューで漏れた欠陥を検出する**

## ソースコードレビューとユニットテストの必要性

コーディング直後のソースコードを確認するのに、確かにソースコードレビューも大切です。
しかし、結果がレビューア（ソースコードレビューを実行する人）によって左右されやすく、どのような観点で行われたのか曖昧になる可能性が高いことから、信頼性の面ではユニットテストより劣ります。
そのため、ソースコードレビューは、機械的には判断しづらい箇所に観点を絞って行い、確認の大部分はユニットテストに委ねるのが一番良い方法だと考えます。




戻る 目次