

ソフトウェアテストシンポジウム 2006



.NET開発でカバレッジ100%を実現

日本コンピュータ株式会社
営業技術本部

目次

- ・カバレッジ100%の難しさ
- ・カバレッジを100%に出来ないと
未テストのコードが残る(1)
未テストのコードが残る(2)
チェックと修正にかかる工数が増大
- ・コンピュウェア製品によるカバレッジ100%の実現
- ・まとめ

カバレッジ100%の難しさ

カバレッジ分析を使用しても、未テスト箇所を完全に無くするのは困難

- ・テスト漏れ箇所
- ・デッドコード
- ・例外処理コード

→カバレッジ100%という目標値を設定できないために、テストを徹底出来ない。

カバレッジを100%に出来ないと

- ・ソースコード中に未テストの箇所を残したまま
次工程に進んでしまい、品質が低下
- ・後工程のチェックでバグが発見され、手戻りによって
工数が増加

品質の低下と工数増加による、プロジェクトの
コスト増・納期遅れ

未テストのコードが残る(1)

- ・例外処理のコード部分が、テストできずに残ってしまう。

```
try {
    if (val == 0) {
        totalValue += 0;
    }
    try {
        StreamReader strm2 = new
        StreamReader(@"c:\West.txt", Default);
        catch (System.IO.IOException iex2)
        { result.Items.Add(iex2.Message.ToString());
          result.Items.Add("strm2 IOException catched.");
        }
    }
    finally {
        result.Items.Add("strm2 finally done.");
    }
}
```

例外処理コード

■ テスト済みコード
■ 未テストコード

- ・例外処理のテストには、ハードウェアによる環境の用意が必要
- ・テスト環境の準備に十分な工数、予算はとられていない
- ・例外処理のテストには、高度な技術力も必要

未テストのコードが残る(2)

- ・カバレッジの目標値を80~90%等にせざるを得ないため、例外処理以外の
未テスト箇所を見逃してしまう。

カバレッジの目標値は超えている

```
try {
    myDataAdapter1.Fill(dsSal, "SALTOTAL");
    dsSal.TotalRow = new DataRowView(dsSal.Tables["SALTOTAL"]);
    dsSal.TotalView.Sort = "QUST_CD";
    if (dsSal.TotalView.Count > 0) {
        result.Items.Add("1st Message ToString()");
        result.Items.Add("2nd Message ToString()");
    }
    finally {
        result.Items.Add("finally done.");
    }
}
switch (dsSal.TotalView.Count) {
    case "MSDE":
        connStr = "workstation-id='TO-8824',packet size=4096,server id=ds,Integrated security=SSPI,provider=
        'Microsoft SQL Server',persist security info=false,initial catalog=master";
        myDataAdapter2 = new System.Data.SqlClient.SqlDataAdapter("select security info from sys.sysconfigurations");
        mySqlConnection = new SqlConnection(connStr);
        mySqlDataAdapter2.Fill(dsSal, "SECURITY");
        result.Items.Add("GROUP BY QUST_CD, COLUMNNAME");
        mySqlConnection.Open();
        myDataAdapter2.Fill(dsSal, "CUSTOMER");
        break;
    case "DWDB":
        result.Items.Add("when database work done?");
        break;
}
class "ORACLEDB" {
    SqlConn = "jdbc:oracle:thin:@//localhost:1521/orcl";
    CustomerClassSet = new System.Data.OracleClient.OracleConnection(connStr);
    OracleDataAdapter = new OracleDataAdapter(connStr);
    mySqlConnection = new OracleConnection(connStr);
    mySqlDataAdapter.Open();
    OracleDataAdapter.Fill(dsSal, "CUSTOMER");
    myOracleAdapter.Fill(customerDataSet, "CUSTOMER");
}
```

■ テスト済みコード
■ 未テストコード

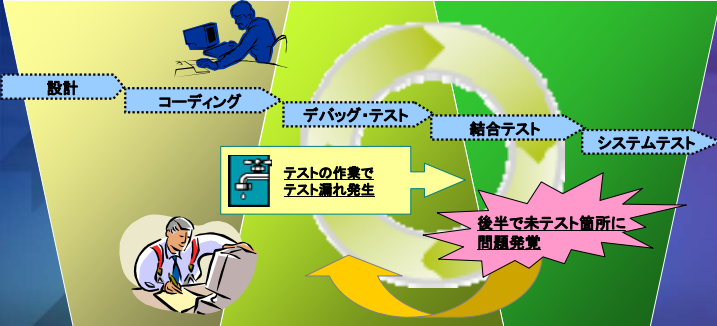
例外処理コード

※) 環境が無いためにテスト出来ない

※) 全体のカバレッジが目標値を超えているので、テストが漏れてしまう

※) 本来テストすべきコード部分

チェックと修正にかかる工数が増大



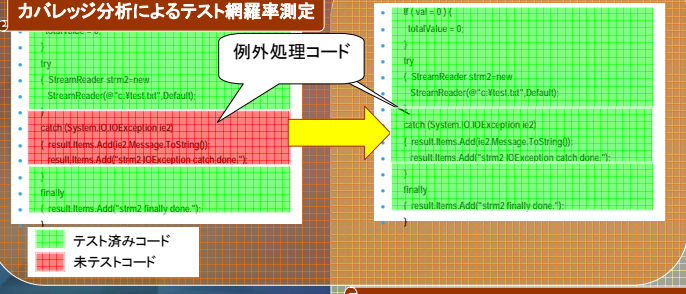
- ・手戻り作業によって、作業工数が増大してしまう
- ・運用段階で例外処理に問題が発生すると、システムダウンの怖れ
- システム修復に莫大な工数・コストがかかる

解決策の御提案

コンピュウェア製品の活用による
カバレッジ100%の実現

カバレッジ100%を実現するDevPartner

DevPartner Studio :
カバレッジ分析によるテスト網羅率測定



DevPartner Fault Simulator :
例外をシミュレートし、例外処理部分のテストを実現

まとめ

- ・例外部分のテストによって、カバレッジを徹底
- 後工程への持ち越しを防ぎ、手戻りによる工数増加を防ぐ
- 例外処理のテストを徹底することで、それ以外(通常処理)の未テスト箇所も明確に
- 100%という明確な目標を持つことで、テストを徹底出来る

システム品質の向上および、
作業工数の増加防止を実現

COMPUWARE®
www.compuware.co.jp