NUnitを使った ハイクオリティプログラミングのススメ

第2回

テスティングフレームワークによる.NET開発

テスト駆動開発の実際

浅井 斉 *ASAI, Hitoshi*
株式会社テクノロジックアート
テクニカルデプトシステム
デベロップメントグループ

High Quality Programming

Technology Tools

- ☑ Visual Basic .NET
- ❑ Visual C# .NET
- ❑ SQL Server 2000
- ❑ Oracle 9i
- ❑ Access 2002
- ❑ ASP.NET
- ❑ Internet Information Services
- ☑ Other:
  NUnit

Level

★★☆☆☆

Samples

・この記事で取り上げたソースコードおよびサンプルプログラムは、付録CD-ROMの\DOTNET\HIQUALITYディレクトリに収録しています。

\BATTINGAVERAGECALCULATOR
テスト駆動開発で作成した打率計算アプリケーション

はじめに

振り返りたくない過去、ありますか？

「あの時、ああしていれば……」。そんな思いは、誰だって経験があるはずです。「もう少し早く気が付いていれば……」。発覚した後、誰もが後悔するんです。でも、どうしてもう少し早く気が付かなかったんでしょう？　そう。それは、テストが無かったから。

前回はテスティングフレームワークを導入することにより、どれだけテストという作業を楽にすることができるかについて解説しました。連載2回目となる今回は、前回紹介したテスティングフレームワークを使用して、“テスト駆動開発”という新しい開発手法を試してみます。

開発手法のあれこれ

本稿で紹介する“テスト駆動開発”ですが、これはいったいどのような開発手法なのでしょう。簡単に説明すると、テスト駆動開発とはソフトウェア開発における「テスト」の存在を最重要視し、「テスト」を中心にして進めてゆく開発手法です。

というわけで、今回はテスト駆動開発を理解するために、まずテストの存在について振り返り、テストの重要性を再認識してゆきます。その後、テストを重要視した開発手法の必要性を考え、テスト駆動開発の解説をしてゆきます。

◆真実のテスト

それでは、ソフトウェア開発におけるテストとは何だったのでしょうか？

ソフトウェア工学によると、

「テストとは、作成したソフトウェアが正しく動くかどうかを確認する作業である」

とあります。では、この作業を行なう意味は何でしょう？

ソフトウェアというものは、いつでも正しい動きをしなくてはいけないものです。たとえば、原子力発電所の監視システムが核融合炉の温度を間違って報告することは決して許されませんし、コンビニの売り上げ管理システムが当日に売れた鮭おにぎりの数を間違って報告することも許されません。

このように私たち開発者は、いつでも正しく動くソフトウェアを作らなくてはいけないのです。ですから、私たちはテストという作業を行ないます。

つまり、決して間違いを許されることのないソフトウェア開発という世界において、正しさを証明するためにテストが必要となるのです。これが、テストという作業を行なう意味であり、テストの役割でもあります。テストとは「正しさを証明するための道具」なのです。

◆いつテストしよう

では、前回紹介したVB的テスト手法や、それを発展させたテスティングフレームワークによるテスト手法は、ここで紹介しているテストの意味を成し、テストの定義を満たしていたでしょうか？　そして、ソフトウェアの本当の正しさを証明することができていたのでしょうか？　残念ながら、答えはNOです。

なぜなら、前回紹介したテスト手法は、いずれも作成した機能に対してのみテストを行なっているからです。当然、この手法を用い、作成した機能のテストを行なうことで、その機能が正しく動作するかどうかはテストすることができます。でも、作成し忘れた機能がある場合はどうでしょう。機能を作成する時点で仕様を勘違いしていた場合はどうなのでしょう。どちらの場合も、テストはテストの意味を果たすことができません。前回紹介したテスト手法では、あくまでも作成した機能が作成したとおりに動作することをテストするだけで、その機能がそれで本当に正しいのかどうかを証明することはできないのです。

それでは、具体的な例を元に、このようなテストを後回しにした開発が起こしてしまう問題を見てゆきましょう。次のような場合を考えてみてください。

システム
- Web上の国内線航空機予約システム（図1）

機能と仕様
- 出発地、目的地、出発日を指定して、便情報と空席情報を座席の種類ごとに一覧表示する
- 一覧から空席を選択すると、予約を行なうことができる
- 予約は2か月前から前日まで可能

手順
- 仕様を元に機能を洗い出す
- 各機能を実装
- 各機能が正しく動くかテスト

このような状況でテストを行おうとすると、2つの問題が起こります。

ひとつ目の問題は、テストの精度が下がってしまうことです。仮に、実装の際に「予約は2か月前から前日まで」という仕様を誤解して、いつでも予約可能である、

図1：国内線航空機予約システム