.NET開発において、ソースコードから未テスト箇所を完全に無くす

～コンピュウェアのテストソリューション～

2006年 5月 11日
日本コンピュウェア株式会社
営業技術本部

会社案内

米コンピュウェア

会社名：	コンピュウェア・コーポレーション (NASDAQ: CPWR)
設立：	1973 年
本社：	米国ミシガン州デトロイト
売上高：	12.3 億米ドル（2005年3月末）
従業員：	約 7,700 名(2005年6月末)
拠点：	60ヵ国 84拠点
顧客数	全世界 23,000社

日本コンピュウェア

会社名：	日本コンピュウェア株式会社
設立：	1992 年
本社：	東京都港区虎ノ門
売上高：	約33.6 億円（2005年3月末）
従業員：	約145 名(2005年3月末)
拠点：	東京、大阪、名古屋

	1973	米コンピュウェア創立
	1977	Abend-AID販売
日本で販売開始	1984	
日本コンピュウェア㈱設立	1992	IPO（株式公開）
大阪営業所開設	1994	
	1998	売上高10億ドル達成
大阪営業所を西日本支社に改組	1999	
名古屋営業所開設	2000	COVISINT 社を買収
	2004	Changepoint 社を買収
CJCパートナープログラム開始	2005	Adlex社を買収

---

1. システム開発の課題

1-1. システム開発の課題

- 品質の底上げが必要
  - 成功案件：全体の20～40%
    - 米Standish Group社調査による
  - 大きなトラブルも続発
- 他業種との比較
  - 製造業、建設業との違い
    - 大量生産しにくい
      - パッケージ化が困難
      - モジュールの再利用が難しい
    - 原価に対して人件費割合が高い
      - ソフトの制作は10年前から手作業が中心
    - ***製造工程が見えにくい（管理しにくい）***
      - 短納期のためチェックしきれないこともある

2005年11月16日：朝日新聞

ハード面の進歩に比べ、ソフトの制作方法は10年前とほとんど変わらない。手作業で制作するうえ、ソフトの不備の検証ツールなどの普及が進まず、使える技術者も少ない。

技術面のリーダーの育成が急務だ。

1-2. 製造工程の可視化の必要性

- ***製造工程が見えにくい（管理しにくい）***
  - 製造工程 ⇒ 担当者の技術、能力に依存しがち
    - 属人性 ⇒ 管理者は「漏れ」をチェックできない
  - 「漏れ」のチェック：テスト過程で十分なチェックを
    - ＜現実は＞
      - 短納期のため十分なテストができない
      - テスト漏れが把握できない

短納期⇒テストへの影響		テスト漏れ
ブラックボックステストに依存 最低限のテストを実施 テスト漏れを把握できない	⇒	前工程でのテスト漏れによる手戻り 運用後トラブルにつながる

1-3. 製造工程でのテストの重要性

- 多くのテスト手法が存在
  - テストの手法は多い
  - ただし、品質向上への特効薬・万能薬はない
  - そのためにもテストにより、誤りを見つけ除去するアプローチが重要
- 短納期 ⇒ ***ブラックボックステストに依存***
  - 短納期などのため、ブラックボックステストに依存せざるを得ないプロジェクトが多い
  - 本来は単体テストでホワイトボックステストを行い、結合テスト以降でブラックボックステストを行うのが原則
  - ブラックボックステストだけでは万能ではない

1-4. 製造工程の「可視化」 イメージ

よくあるプロセス

設計 → 製造 → 単体テスト → 結合テスト → 総合テスト → 運用

ブラックボックステスト

テスト漏れが検証できない

ご提案のプロセス

設計 → 製造 → 単体テスト → 結合テスト → 総合テスト → 運用

ブラックボックステスト

ホワイトボックステスト

クロスチェックで高品質を実現

1-5. ブラックボックステストとホワイトボックステスト

	ホワイトボックステスト	ブラックボックステスト
長所	プログラム全て ⇒ プログラマの意図通りかチェック 網羅的なテスト ⇒ 珍しい事態（例外など）にも対応 テストが数値化できる ⇒ 人依存をなくせる	仕様書どおりの実装か ⇒ 低コストで確認 最小限の工数、最低限のテストができる 万能ではないが、単体テストから統合テストまで幅広く適用可能
短所	プログラマの誤解 ⇒ チェックできない → 利用者の視点でチェックする必要性 計画、記録に手間がかかる ⇒ ログの埋め込みなど実施	システムの内部処理 ⇒ 一切問題としない テスト結果は担当者が判断 ⇒ 管理しにくい テストケースの漏れ ⇒ 発見できない 特殊なケースのテスト ⇒ 漏れがち
	定量的な指標に使える ツール利用 ⇒ 簡単に高品質なテストを実現	*管理しにくいのが最大の問題！*

2. ご紹介

2-1. Compuware .NET 対応製品ラインナップ

設計 | コーディング | 単体テスト | 結合・機能テスト | 負荷テスト | 運用

DevPartner Studio
DevPartner Fault Simulator
TestPartner
QALoad
Vantage

この工程での品質向上についてご紹介します

2-2. ご紹介のプロセスイメージ

設計 → 製造 → 単体テスト → 結合テスト → 総合テスト → 運用

ブラックボックステスト

ホワイトボックステスト

例外対応コード支援　例外処理動作検証　自動機能テスト

実施手順	製品名	機能名称
ホワイトボックステスト	DevPartner Studio	カバレッジ分析
例外対応コード支援	DevPartner Fault Simulator	ソースハイライト
例外処理動作検証	DevPartner Fault Simulator	例外シミュレーション
自動機能テスト	TestPartner	

2-3. ご紹介の概要

① *単体テスト ＋ コードカバレッジ*
テストを管理 ⇒ テストケース漏れなどを見逃さない

② *例外シミュレーション ＋ コードカバレッジ*
例外処理もツールで検証 ⇒ 全てのコードをテスト

③ *自動機能テスト*
回帰テストはツールで省力化

ツール利用 ⇒ 高品質なシステム構築

2-3-1. ①単体テスト＋コードカバレッジ

① *単体テスト＋ コードカバレッジ*

② *例外シミュレーション＋ コードカバレッジ*

③ *自動機能テスト*

単体テスト＋コードカバレッジの必要性

単体テスト：短納期などのためテストが漏れることも

コードカバレッジ ⇒ ホワイトボックステストが容易になる

低コストでテスト漏れをなくし、テストの定量化ができる

DevPartner Studio

2-3-2. カバレッジ分析

ブラックボックステストの問題点

テスト状況がわかりにくい ⇒ リーダーが評価できない

テスト漏れが把握できない

テスト漏れは運用後にしか見つけられない

DevPartnerの解決策

ブラックボックステストをしながら、ホワイトボックステストへと発展できる

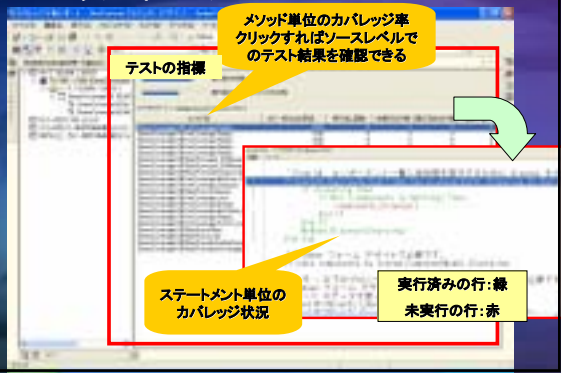
テスト状況が把握できる

テスト品質を均一化できる

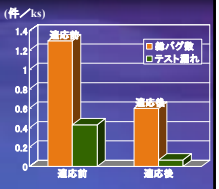
DevPartner Studio⇒ テスト漏れが把握できる

DevPartner Studio

カバレッジ イメージ



カバレッジ分析導入事例



導入事例

- コスト削減を実現するため、DevPartnerを導入
- 単体テスト受け入れ時の検査項目

導入効果

- 受け入れ検査時に体裁面だけでなく、品質面からもチェックできた
- 結合試験以降のバグ発生件数、テスト漏れ件数を抑制できた
  （結合試験以降：総バグ発生率約6割減、テスト漏れ発生率約8割減）
- 手戻り工数が削減でき、総工数の5%が削減できた
- 網羅率でテスト仕様書のケース漏れも把握でき、テスト仕様書の精度が向上した

2-4-1. ② 例外シミュレーション＋コードカバレッジ

① *単体テスト＋ コードカバレッジ*

② *例外シミュレーション＋ コードカバレッジ*

③ *自動機能テスト*

例外シミュレーション＋コードカバレッジの必要性

単体テスト ⇒ 例外処理はテストしにくい

例外処理をテストせずにリリースの危険あり

例外処理が抜けていると大きなトラブルの可能性も

DevPartner Fault Simulator

2-4-2. 例外シミュレーション＋カバレッジ

例外処理検証の問題点

（コーディング時）発生箇所がわかりにくい

（テスト時）例外が発生させられない ⇒ 未テストのままリリース

（運用時）運用環境では発生 ⇒ 大きな問題の可能性

FaultSimulatorの解決策

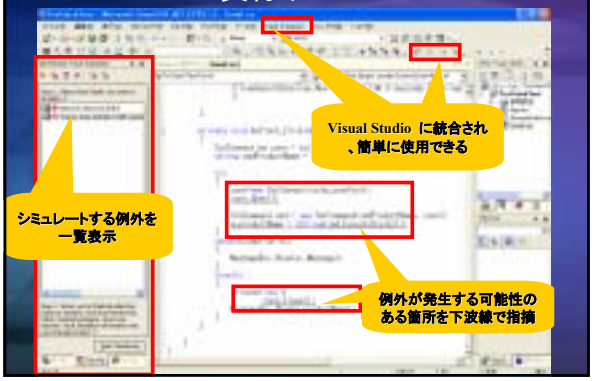
（コーディング時）ソースハイライト ⇒ 例外発生位置を指摘

（テスト時）実際に例外を発生 ⇒ 簡単にシミュレーション

コードカバレッジ100%を実現

Fault Simulator 実行イメージ

DevPartner Fault Simulator



2-5-1. ③ 自動機能テスト

① 単体テスト + コードカバレッジ

② 例外シミュレーション + コードカバレッジ

③ 自動機能テスト

自動機能テストの必要性

画面系のアプリケーションはテストしにくい

同じテストの繰り返しが多い ⇒ コスト増

ツールにより繰り返しテスト工数を削減

TestPartner

2-5-2. 自動機能テストツール

回帰テストの問題点

テスト工数が多大

納期などの関係でテストが不十分になるケースも

デグレードをチェックしきれない

TestPartnerによる解決策

テスト手順をスクリプトに記録

2回目以降はスクリプトの再生によりテストできる

⇒ 回帰テスト工数の大幅省力化を実現

3.まとめ

- 製造工程を*いかに管理するか*が成功の鍵
- *カバレッジ分析*で *テストを管理できる*
- *例外シミュレーション*で堅牢なシステムを構築
- *開発を管理すること*により*問題を早期発見*
- *問題の早期発見*で*コストを抑制*
- プロジェクトの利益向上

参考：機能一覧

製品名	機能一覧
DevPartner Studio	カバレッジ分析 静的ソースコード解析 パフォーマンス分析 メモリ分析 エラー検出
DevPartner Fault Simulator	例外シミュレーション
TestPartner	自動機能テスト

COMPUWARE

www.compuware.co.jp

お問合せ先
devpartnerj_west@compuware.com