Visual Studio 2005 で実現する高品質なソフトウェア開発

マイクロソフト株式会社
デベロッパー&プラットフォーム統括本部
岩出 智行

Djhqgd

- システムのライフサイクルと品質
- プロジェクト成功率向上の為に
- Team System が提供する品質向上の為の機能
- まとめ

システムのライフサイクル

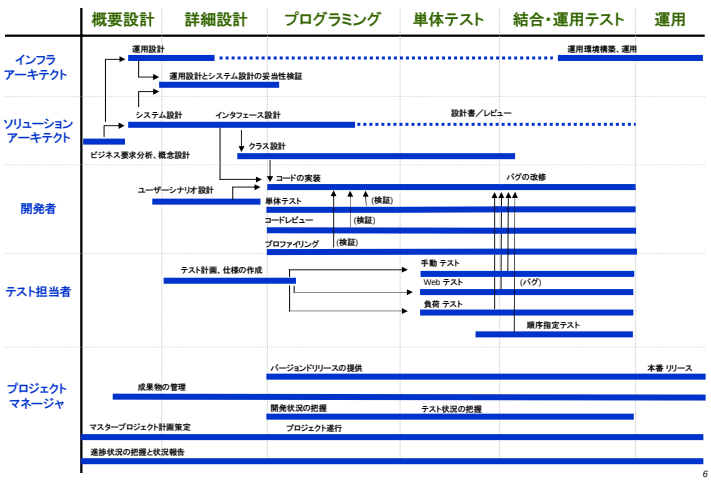
システムの品質

- バグの少なさ≠システムの品質
  - 信頼性、堅牢性の高さも品質
  - 運用のしやすさ、利用のしやすさも品質
  - 運用コストの低さ、利用者のニーズに応えることも品質
  - 運用者や利用者のフィードバックを受け、システムを容易に変更できる、ということも品質

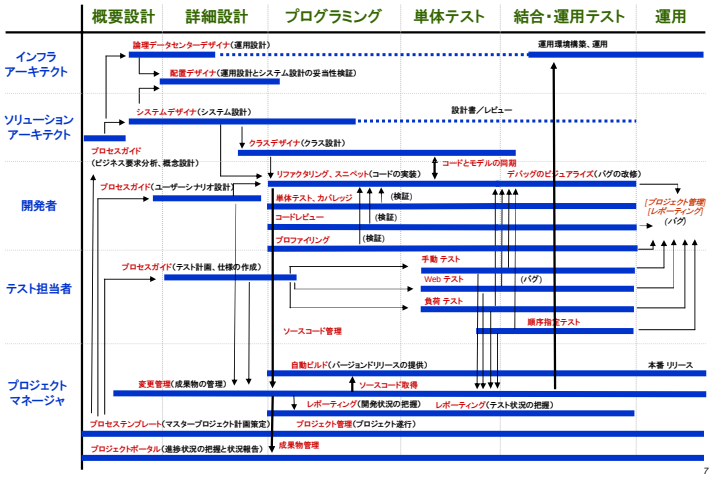
品質はテスト担当者だけの問題ではない
システム開発に携わる全てのメンバーが意識すべき問題

開発における役割の明確化

開発における各役割の担当作業

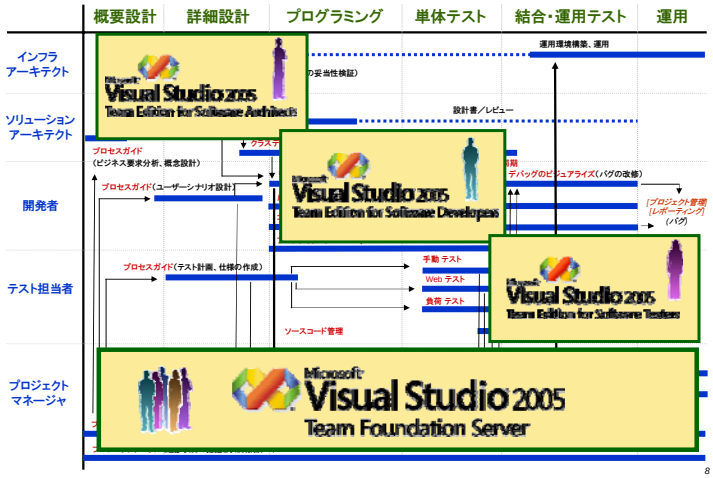


Team System が提供する機能



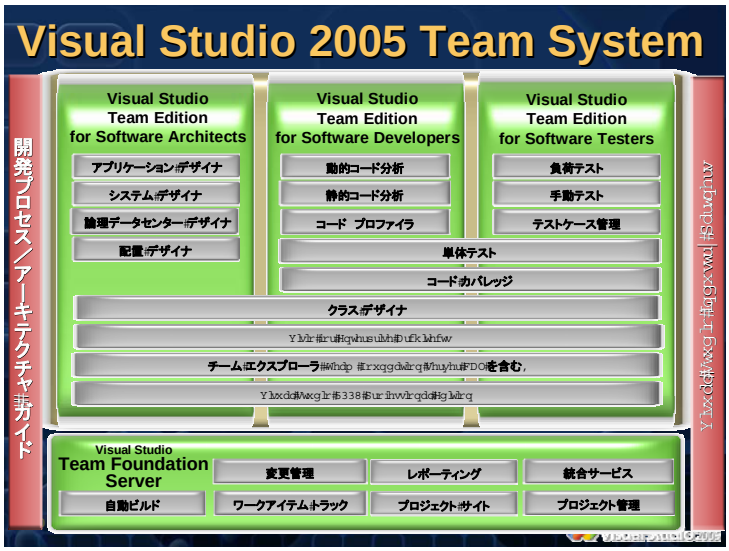
7

Team System のエディション構成



8

9



Djhqgd

- システムのライフサイクルと品質
- プロジェクト成功率向上の為に
- Team System が提供する品質向上の為の機能
- まとめ

11

プロジェクト成功率向上の為に

- @IT 調査より、今後実施予定、検討中の項目 Top 5
  - 機能テストや回帰テストの自動化
  - 単体テストツールによる品質の向上
  - 適切な開発プロセスの選択/最適化
  - 要求管理/要求変化の影響分析
  - 実稼動前の性能テスト/負荷テスト

出典：@IT読者調査（2005年 N=287）
http://www.atmarkit.co.jp/news/survey/2005/05dev/dev.html

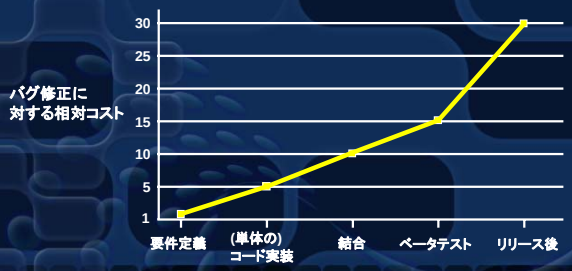
- 手戻りの原因としてテスト関係の不備が16%近くに上る

出典：経済産業省　2005年版組込みソフトウェア産業実態調査　報告書
http://www.ipa.go.jp/software/sec/download/200506es.php

12

プロジェクトにおける修正コスト

- 一般に、後工程の見直しはコストが高い
- 後工程の見直しを最小限にする必要



National Institute of Standards and Technology (国立標準技術研究所: 米国政府機関の一つ、工業規格の標準化を行う)のレポートより。(参照: http://www.nist.gov/director/prog-ofc/report02-3.pdf)

テストの強化～多層防御と自動化

- 多層防御戦略(Defence in depth)の開発現場への応用
  - 早期から、頻繁に
    - テストフェーズではなく、設計フェーズから品質向上の取り組みを開始
    - テストの効率化、自動化を進め、繰り返しテストを実施する
  - 多面的に
    - 静的テストと動的テスト
    - ブラックボックステスト・ホワイトボックステスト・グレーボックステスト
    - 単体テスト・統合テスト・システムテスト
    - 機能要件・非機能要件それぞれに対するテスト
    - ユーザー指向・不具合指向・要件指向・設計指向

※ 全て実施するのではなく、テスト戦略の策定において強弱を設定する

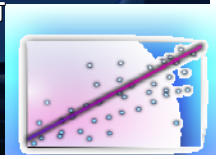
- テスト、およびテスト報告の効率化・自動化

Team System の提供機能①

- 様々なテストサポート機能の提供
  - 静的コード分析、および動的コード分析
  - プロファイリング
  - 単体テスト支援、およびコードカバレッジ測定
  - Web アプリケーションの機能テスト、およびロードテスト
  - 順序指定テスト
  - 手動テスト、など
- テストの効率化・自動化のための機能の提供
  - 自動ビルド機能(Team Foundation Build)
  - テスト管理ツール(テスト マネージャ)
  - バグ トラッキング機能(作業項目のトラッキング)

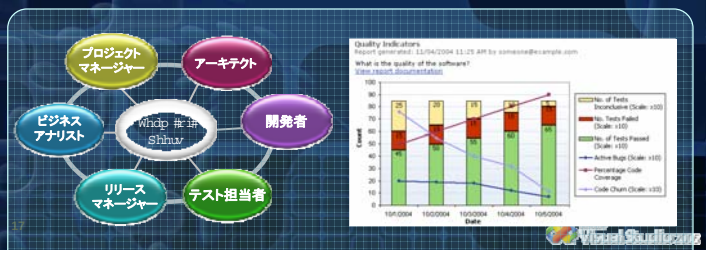
品質向上作業をプロセスに組込む

- 開発メンバーの品質に対する役割と責務を明確化
  - 役割のあいまいさをなくす
    - Microsoft Solutions Framework (MSF) の例
      - 開発者の役割とテスト担当者の役割兼任は非推奨
      - ビジネス アナリストやリリース マネージャの役割とテスト担当者の役割兼任は可能
  - それぞれの役割が果たすべき責務を明確にする
    - MSF の例
      - テスト担当者は品質基準の設定、テストアプローチと計画の作成、テスト仕様の決定、テストの実施、テストの報告、等の責務を負う
- 開発プロセスと同様にテストプロセスを用意する
  - 開発プロセスとテストプロセスの協調(Wモデル)
- 品質に関する情報を開発メンバー全員で共有



Team System の提供機能②

- 開発プロセスの提供とツールへの統合機能の提供
  - MSF の最新バージョン (Ver. 4.0) となる MSF for Agile Software Development、MSF for CMMI Process Improvement の提供
  - 開発ガイドをツールに統合
- 品質情報共有のための機能の提供
  - テスト結果、バグ情報等共有のためのレポーティング機能

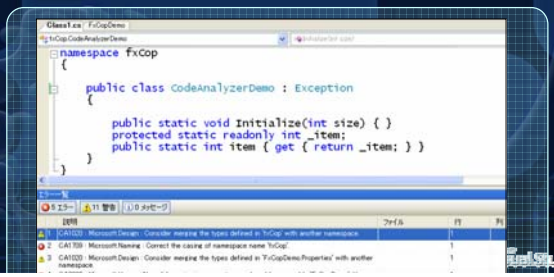


Djhqgd

- システムのライフサイクルと品質
- プロジェクト成功率向上の為に
- Team System が提供する品質向上の為の機能
- まとめ

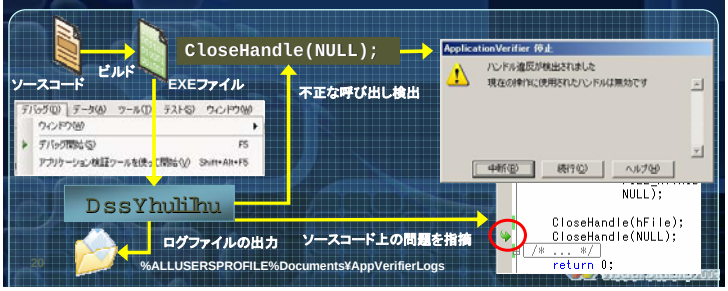
静的コード分析

- ビルドを行う際に、自動的にコードを分析
- 設計ガイドに違反した場合や、ありがちなバグに違反した場合に警告、またはエラーとして報告する
  - 国際化や相互運用性、メンテナンス性等に関するルールの確認
  - バッファー オーバーラン、メモリの初期化漏れ、null ポインタ参照
  - SQL 文におけるコードインジェクションの危険性



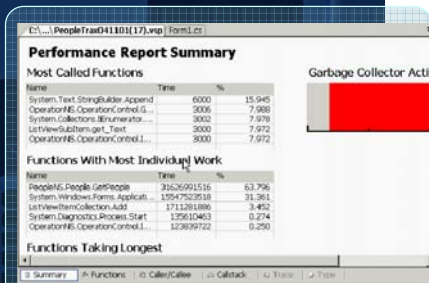
動的コード分析

- ネイティブコードアプリケーションのセキュリティや安定性に関する潜在的な問題を検出
  - 机上や単体テストでは見つけにくいバグの検出
  - ヒープメモリ破壊、スタック・ハンドル違反等



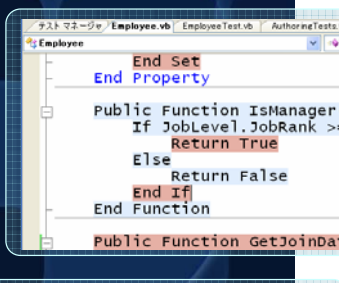
プロファイリング

- パフォーマンス上のボトルネック発見を助け、開発フェーズにおけるパフォーマンス確認と向上が容易になる
- パフォーマンスが重要な要件であれば、早期から頻繁に実施
- プロファイリングにおける二つの測定方法
  - サンプリング
    - ポーリング方式
    - プログラムに立ち入らない
    - 呼び出されたメソッドを見落とす可能性がある
  - インストルメンテーション
    - 測定用のコードを追加する
    - 正確な結果
    - 大きなオーバーヘッド



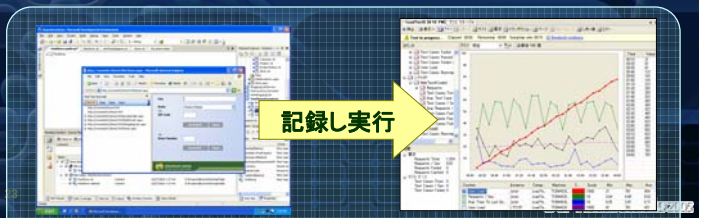
単体テストとコードカバレッジ

- 単体テスト
  - 実装と共に行う基本的なコード チェック
  - レグレッション テストとして有益
- 単体テスト支援機能の利用
  - テスト クラスの自動生成
    - Private メソッドもテスト可能
  - テストケースの管理と実行支援機能
- コードカバレッジ機能の利用
  - 対象コードがどれだけテスト実行されたかデータを取得、表示
    - パーセント表示、グラフィカル表示
    - 単体テストの精度向上に利用



Web テストとロードテスト

- Z he#テスト
  - Z he#操作の記録機能により、簡単にテストを作成可能
  - テストのカスタマイズには#F&#や#E#を利用
  - IQ HW#のシステムに限らず任意の#Z he#ページのテストが可能
  - 便利な機能(フォーム認証、ビュー#ステートの追跡、クッキーの追跡)
  - G E#のデータを利用したデータ駆動テスト(Gdwd#Duhq#Whvw)の実施
- ロードテスト
  - Z he#テスト等に対し、任意の負荷をシミュレート
  - 分散負荷テストを行う場合は#Whdp #Whvw#Ordg#Dj hqw#を利用

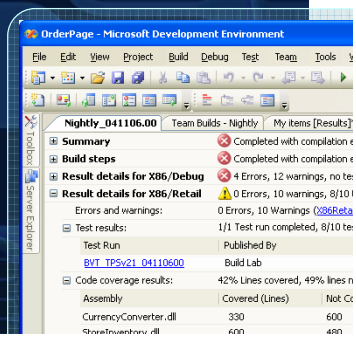


多彩なテスト機能

テストの種類	説明
単体テスト (Unit Test)	メソッドを実行するテストコードを自動的に生成し、かつテストを自動的に実行。さらに生成されたテストコードを、クラス ライブラリの API を用いて自由にカスタマイズすることも可能
手動テスト (Manual Test)	自動テストでカバーできないテスト ケースにおいて、テスト担当者が手順ごとに手動で実行するマニュアル テストの作成と実施
Web テスト (Web Test)	Web サイトの機能テストを実現。ブラウザで操作した手順を記録し再現したり、操作にかかるさまざまな設定(操作の待ち時間)を詳細に定義することが可能。さらに、テストコードを生成してカスタマイズすることで、より複雑な手順も実現可能
ロードテスト (Load Test)	負荷テストを実現。(特に Web アプリケーションに限定されない)同時実行ユーザー数や実行環境を詳細に設定することが可能
汎用テスト (Generic Test)	Visual Studio Team System 以外のテスト ツールを使用したテストの実行や管理を実現
順序指定テスト (Ordered Test)	既存のテストを組み合わせて、指定した順序でテストを実行し、全体としてのテスト結果を判定

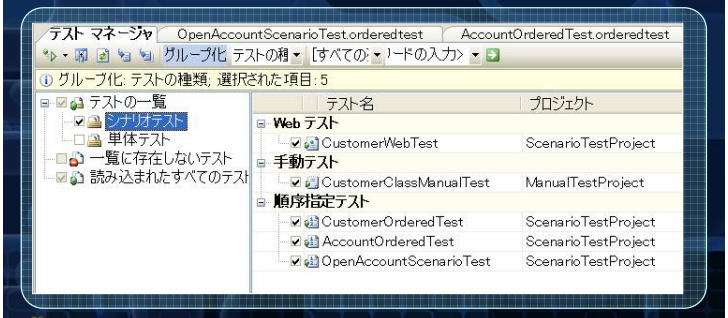
自動ビルド機能

- Team Foundation Build
  - Team Foundation Server の付加機能
  - 新しいビルドエンジンである MSBuild を利用
  - 手動 または スケジューリングによる起動
- Team System の他機能と連動
  - 単体テストの実施
    - バグの自動登録
  - 静的コード分析
  - ソースコード管理との関連付け
- ビルド結果の表示と分析



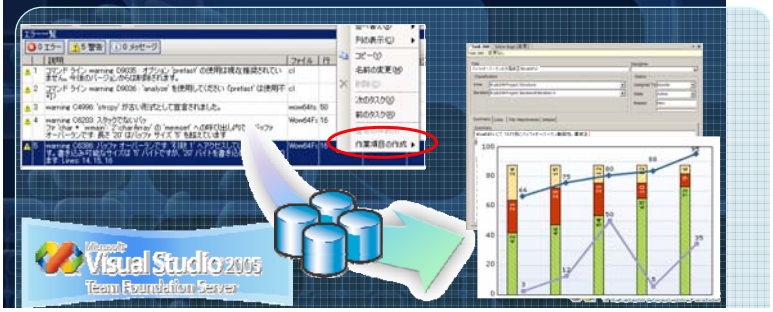
テスト実施の効率化と自動化

- テスト マネージャによるテスト管理
  - テストの一覧表示
  - テストのグループ化とグループ単位での一括実施



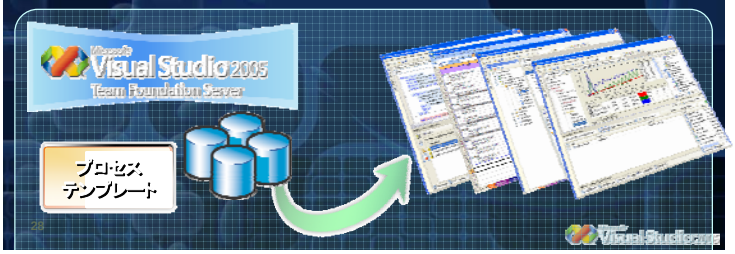
テスト報告の効率化と自動化

- 「作業項目」トラッキング機能の利用
  - エラーリストや、テスト結果のウィンドウから作業項目（例えばバグやタスク）を作成し、チームで共有
- レポーティング機能によるテスト実施状況の確認



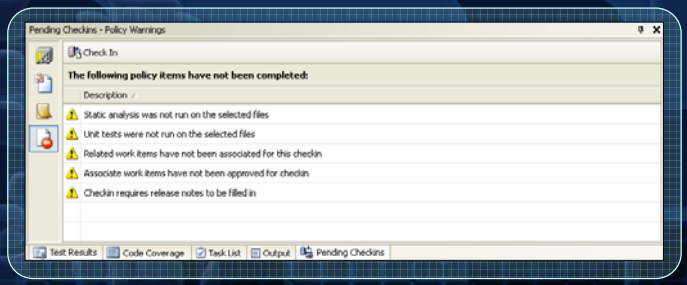
開発プロセスの提供と統合

- Microsoft Solutions Framework (MSF)
  - 1994年バージョン1.0 提供開始
  - Visual Studio 2005 のリリースにあわせバージョン 4.0登場
- MSF 4.0 はプロセステンプレートとして Team Foundation Server に統合
  - プロセステンプレートを元に、個々の開発プロジェクト専用の作業項目リストやポータルサイトを作成



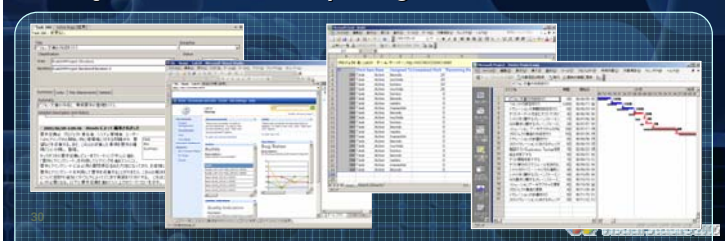
チェックインポリシーの利用

- 「チェックインポリシー」はチェックインの際のルール
  - コードのチェックイン作業における品質向上作業の実施確認
  - 「静的分析を実施しているか」、「単体テストを実施しているか」、「どのタスクに基づき行った作業かを明示しているか」等、ルールを設定可能
  - 作業の実施忘れを防ぐ（ポカよけ）



情報の共有

- SQL Server 2005 によるデータリポジトリ
  - 作業項目（バグ、タスク、顧客要求等）
  - テスト結果
  - プロジェクトの進捗状況
  - ソースコード管理、など
- Visual Studio、Web ブラウザ、Microsoft Excel、Microsoft Project 等使い慣れたツールを使用して各種データへアクセス
- SQL Server 2005 の Reporting Services による柔軟な出力



Djhqgd

- システムのライフサイクルと品質
- プロジェクト成功率向上の為に
- Team System が提供する品質向上の為の機能
- まとめ

まとめ

- 情報システムの価値を高める上で品質向上は必ずクリアすべき課題である
- 品質向上の取り組みは、開発に関わる全てのメンバーが意識し、取り組まなければならない
- 品質向上のアプローチとして多層防御戦略を応用
- 効率化と自動化を進める
- 開発プロセスとテストプロセスの融合を行う
- 情報共有によりメンバー全員で品質向上に取り組む

Appendix：参考情報

- Visual Studio 2005 Team System
  - www.microsoft.com/japan/msdn/vstudio/teamsystem/
- Visual Studio 2005 購入ガイド
  - www.microsoft.com/japan/msdn/howtobuy/vs2005/
- Microsoft Solutions Framework
  - www.microsoft.com/japan/msdn/vstudio/productinfo/enterprise/msf/
- Visual Studio 2005 Team System パートナー ソリューション
  - www.microsoft.com/japan/msdn/vstudio/teamsystem/partner/
- Visual Studio 2005 オンライン セミナー
  - www.microsoft.com/japan/msdn/vstudio/training/
- Visual Studio 2005 コミュニティ
  - www.microsoft.com/japan/msdn/vstudio/community/

Appendix：ライセンス モデル

- 開発者ライセンス
  - Team Edition (Architects, Developers, Testers)
  - Team Suite
- サーバー/CAL ライセンス
  - Team Foundation Server
  - CAL：Team Foundation CAL (Team Explorer)
    - Microsoft Office Excel、Microsoft Project Professional を利用し、Team Foundation Server との連携を行う場合に必要
- プロセッサ ライセンス
  - Team Test Load Agent
    - Team Edition for Software Testers の補完製品
    - 仮想ユーザーあたりではなく、プロセッサ毎の課金

参考：www.microsoft.com/japan/msdn/howtobuy/vs2005/editions/team/license/

Appendix：ライセンス構成

Microsoft

Your potential. Our passion.™