小さなプロジェクトの小さな改善
～大仰なキーワードの身近な適用～

SEPG Japan 2005

2005年10月13日

SRA
開発サービスカンパニー
産業システム第1部
杉山　万利子
mariko@sra.co.jp

SRA先端技術研究所
林　好一
yosikazu@sra.co.jp

本日の献立

1. 何をやったか
2. どんな効果があったか
3. まとめ

1. 何をやったか

A. 新しい開発環境の構築
B. 納品物の編集･格納の管理
C. 環境構築方法の文書化
D. 作成した文書の管理

A. 新しい開発環境の構築

元：単体テストを、他社と共有の開発用ＤＢサーバ･アプリケーションサーバを使用して行なっていた

- **定数の変更やテストデータの追加が気軽にできない**
- **テストコードの動作確認をするにもアプリケーションサーバへのデプロイが必要で、その度に再起動の連絡をし、他のメンバは再起動中動作確認ができない**

改：SRAチーム用ＤＢサーバを自前（SRA）で用意し、Eclipseベースの開発環境を構築した

- **メンバ各々用の専用アカウントを作成し、個別に作業が可能**
- **ＤＢ管理担当を決め、テーブル定義の更新などは担当が一括して実施**
- **Eclipseのプラグインが利用可能**
  - **Junit、DBUnitでのテストプログラムのひな型を作成**
  - **Tomcatを使用しての画面テスト環境を構築**

B. 納品物の編集・格納の管理

元：ソース・設計書はファイルサーバ上の、日付名のフォルダ等に管理されていた

- 常に最新版が置かれているわけではなかった（運用ルールが決まっていなかった）
- 最新版と思われるソースも、きちんとコンパイルできないことがあった

改：版管理ツールであるWinCVSを導入し、統一管理を行なった

- 二重管理の必要が生じた際は、ブランチ機能を活用した
- 残念ながら他社も含めた作業者全体ではなく、SRAチーム内のみ

C. 環境構築方法の文書化

- 技術担当メンバが、前述したような幾つかの便利ツールを導入し、その使用説明書 (あくまで使用者が使用できるレベルの記述) も作ってくれていたが、プロジェクトを抜けてしまった
- その後ベース環境のバージョンアップという話が出て、我々の環境もそれに合わせたバージョンアップの必要が出てきた
- 技術担当がいなくなってしまった以上、自分で一から設定できるようにならないと、単体テスト方法、果てはファイル管理方法までが元の状態に戻ってしまうと思い、調べ、その内容を文書化した
  - その資料を参考にして、他メンバがさらに環境バージョンアップを行なった実績あり

D. 作成した文書の管理

元：情報伝達は口頭主体、もしくは個人が個別の環境で作成したメモ書き資料を配布するなど

- ファイルサーバ上に置いたとしても、
  - おき場所が分散してしまい、
  - どこに何があるかすぐにはわからず、
  - ファイル名も内容を示していない場合があり、

  その都度開いて確認していた

改：個人のメモツールとして使用していたwiki（注）による内部ウェブサイトを立ち上げた

- 一箇所から同じ方法で文書が探せるようになり、
  - 開発環境の構築方法や、
  - 解決したトラブルのメモ、
  - 参考ページのURLや参考資料置き場のパス等を記述し、

  情報がかなり一元管理できるようになった

注：wikiとは追加・変更が簡単で自由なウェブページ管理ツール

2. どんな効果があったか

A. **新しい開発環境の構築**
B. **納品物の編集･格納の管理**
C. **環境構築方法の文書化**
D. **作成した文書の管理**

A. 新しい開発環境の構築

- メンバ各々が単体テストを個別に行える環境を構築したことにより、
  - 定数データ変更等でも他人が影響を受けないため、
    - 単体テストを自動化することができ、
    - 作業効率が上がり、
    - コード修正によるデグレードも防ぐことができた
  - アプリケーションサーバにデプロイしなくても画面の動作確認を行うことができたため、
    - 他人の作業を止めなくても済むようになった

B. 納品物の編集・格納の管理

- WinCVSでの版管理により、
  - 自分達の作業対象（ソース・設計書）がどれであるのかが明確になった
  - 最新版がどこにあるか悩んだり、調べたりすることがなくなった
  - 他人に最新ファイルを送付してもらうなどの余計な手間がなくなり、またファイルを待っている間の無駄な時間がなくなった
  - 過去の版もすぐに取り出せるので、修正を取り消す場合の危険が減り、またリリース前の変更箇所確認も容易になった

C. 環境構築方法の文書化 と D. 自分達のための文書の管理

- 情報の文書化と一元管理により、
  - 同じ事を何度も確認したり思い出したりしないで済むようになった
  - 同じことを複数人に伝えたい場合（既存メンバに伝えた後、新規メンバが入ってきた場合なども）、そして自分がプロジェクトを抜ける場合に、何度も口頭で説明しなくて済むようになった
  - 必要な参考ファイルの場所がわからなくなったり、どれが最新版かわからなくなったりしなくなった
  - 必要な情報を検索しやすくなった

3. まとめ

- 「プロセス改善に照らすと、、、」
  - 大仰なキーワードとの対応
- 分析
- 今後に向けて
- 結び

「プロセス改善」に照らすと、、、

A. 新しい開発環境の構築
⇒**自動化**：Eclipseとプラグインの活用
⇒**標準化**：開発環境の統一

B. 納品物の編集・格納の管理
⇒**技術管理**：納品物の統一的な変更管理
⇒**リスク管理**：誤ったファイルの使用の回避

C. 環境構築方法の文書化　と

D. 作成した文書の管理
⇒**技術管理**：内部文書の統一的な変更管理
⇒**標準化**：検索方法の統一、異なる人・時に対しても同じ情報
⇒**リスク管理**：容易な検索と情報共有による勘違いの回避

分析

- 小規模プロジェクトでは目の前の技術作業に目と時間を奪われ、技術管理やプロセス管理をないがしろにしがちだが、大規模プロジェクトに比べ改善案を提案・実施しやすい環境とも言える
- 少しの努力で少しだけ改善作業を行なって作業効率が上がり、後の憂い（リスク）も軽減できた

今後に向けて

- お客様や上司に受け入れられる改善案の提案や、提案の重要性を理解してもらうことがうまくできず、プロジェクト全体的なプロセス改善はあまり行えなかった
- チーム内だけでなく、他社も含めプロジェクト作業者全体で、開発・管理作業の運用を考えていけるような雰囲気を作るにはどうしたらよいだろうか？

結び

- 報告した改善は、例えば「標準化」などのキーワードを適用しようとして始めたものではなく、ニーズから起きた事例
- 結果として、大仰に見えたプロセス改善キーワードも、身近なところで実践できて、成果も実感できるものだとわかった

- ごく小さな改善例を報告させていただきましたが、皆さんの参考としていただければ大変嬉しく思います。
- 質問やコメントも是非お聞かせください。