生産性向上ツールを有効活用したチーム開発事例の紹介

Introduction of Team Development Case where Productivity Improvement Tool is Effectively Used

村上　裕美*　山本　一二三*
Hiromi Murakami, Hifumi Yamamoto

ソフトウェアの生産性は個人のプログラミングスキルに因るところが大きいと言われている。
しかし、会社という組織で受注した業務で個人のスキルのみに依存することは大きなリスクを伴う。工程及び予算に応じたソフトウェア開発を行うためにも、組織としての技術力蓄積・向上が必要になる。
当部署では、ソフトウェアのチーム開発スキルの向上・標準化及び技術・ノウハウの共有を組織目標とし2004年度から実践している。すなわち、個人が得た技術・ノウハウを他のチーム員に共有することで、チーム全体の技術力の底上げを行っている。
本稿では、実際にチーム開発した経験をもとにその活動の一端を紹介する。

It is said that the productivity of software owes a lot to individual programming skills.
Although it is a company that makes and has responsibility to a contract, relying too much on each individual skills might cause a big risk. To develop software appropriately in terms of process and budget, it is necessary to enhance and improve the technical capabilities as a whole company.
In our group, we set a goal to improve, standardize and share the software development skills with the team. Toward the goal, we started our activities last year(2004). That is the effort to establish a system and process to share the knowledge or know-how obtained by a member of the team with others and enhance the level of the whole team skills.
In this report, we introduce a part of our activities applied to software development.

1. まえがき

当部署では、グループ・プログラム設計の標準化推進活動の一貫として、生産性向上ツール(ソフトウェアの開発をサポートするツール)を導入すると共に、その利用促進と情報共有の実現に取り組んできた。
生産性向上ツールは多数出回っており、中には高性能な無償ツールも少なくない。
これらのツールをうまく取り入れることで、低コストで手間のかからない開発環境を得ることが可能となる。
本稿では、当部署が開発標準として取り入れている生産性向上ツールの紹介とチーム開発の事例を紹介する。

2. 当部署の組織目標

当部署では、『グループ・プログラム設計の標準化推進』を組織目標に掲げている。
プログラム設計の標準化とは、開発プロセスに生産性向上ツールを取り入れて作業の効率化を図ると共に、ツールそのものと、ツールを取り入れた開発手法を一般化しようという試みである。
これを実現すべく、生産性向上ツールをいくつか導入し、日々の業務に取り入れると共に、それらツールの情報やノウハウを公開する活動を行っている。
また、ツール関連の情報だけでなく、様々な分野(技術、ノウハウ、現場環境情報など)において情報共有し、組織全体の生産性向上の足がかりになるよう、情報共有化の推進にも力を入れている。

3. 情報共有の仕組み

当部署が取り入れている生産性向上ツールを⑴に、各ツールの情報共有の仕組みを⑵に紹介する。

⑴　生産性向上ツール紹介

ⓐ　EA(Enterprise Architect)(参考1)
- スパークスシステムズ　ジャパンが提供する製品
- Windowsで動作するUMLモデリングツール(UML2.0に対応)
- UMLを利用した開発ライフサイクル全体をサポートするために、要求の収集から分析・モデル

＊つくば事業部 第一技術部

の設計・ソフトウェアの実装・テスト・保守までを広くサポートする
－直感的で分かりやすい操作体系
－複数ユーザによる開発も可能
－現実的な価格設定（１ライセンス3万円前後）

(b) Eclipse(参考2)
－統合開発環境
－CVSとの優れた親和性を持つ
－プラグイン・アーキテクチャであるため、機能を拡張しやすい
－柔軟なライセンス体系
－無償

(c) CVS(参考3)
－ネットワークに対応したバージョン管理システム
－プラットフォームはWindows、UNIX系双方に対応
－ファイルの変遷をすべて記録し、任意のバージョンを取り出すことができる
－複数人で編集するコンテンツ（プロジェクト等）の管理に真価を発揮する
－無償

(d) PukiWiki(参考4)
－Wiki（WikiWikiWeb(参考5)）クローン(参考6)の一つ
－誰もがwebブラウザを通してコンテンツ（webページ）を編集（追加、削除含む）することが可能なwebアプリケーションサーバ（PHPで作成された国産プログラム）
－複数人が共同でwebサイトを構築していく利用方法を想定している
－サイトに制限（編集など）をかける機能を有する
－HTMLの知識がなくても容易にコンテンツを作成できる
－プラグインによる拡張が可能
－無償

※開発ライフサイクルと利用可能なツールの対応を図１に示す。

(2) ツールを使用した情報共有の仕組み

(a) EA
－１つのプロジェクト・ファイルを複数人で同時に編集することが可能

共有フォルダにEAのプロジェクト・ファイルを置いて各メンバがそれを直接編集する。メンバ同士で修正の衝突が起こらないよう、ロックする機構が用意されている。（プロジェクト・ファイルを丸ごとロックするのではなく、図やパッケージ単位でロックできる。）

本手法の概略図を図２に示す。

－プロジェクト・ファイルの複製が可能

プロジェクト・ファイルの原版を複製したものを各メンバが修正し、定期的に同期をとる。

普段はローカルで作業可能であるため、「分散環境での開発」に適している。

－共有のデータベースを利用可能

プロジェクト・ファイルを共有データベース（SQL Server等）に登録し、それを直接編集する。

機能的には１項目と同じだが、データベースを利用することによって、修正中にPCやネットワークに不具合が生じた場合などにも、ファイルが破損しない。

－プロジェクトを作業範囲毎に分割可能

前述したいずれの方法でも、同じ図を複数のメンバで修正した場合には衝突が検知されるが、修正内容を細やかに選択してマージするような機能はない（どちらのメンバの修正を反映させるかを選択できる程度）。

これを回避する方法として、メンバの担当部分毎にプロジェクト・ファイルを分割し、これらのファイルを結合してプロジェクト・ファイルを構成する方法がある。

分割されたファイルはXML形式（テキスト）であるため、CVSなどでバージョン管理しやすいという利点もある。

(b) Eclipse、CVS

ここでは、EclipseをCVSのインターフェースとして使用する場合のチーム開発におけるバージョン管理手法を紹介する。

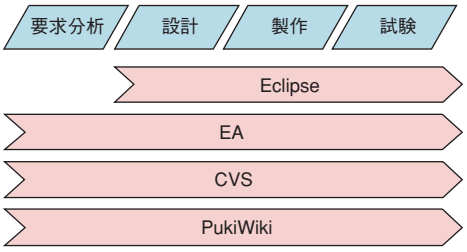

図１　開発ライフサイクルと生産性向上ツールの対応

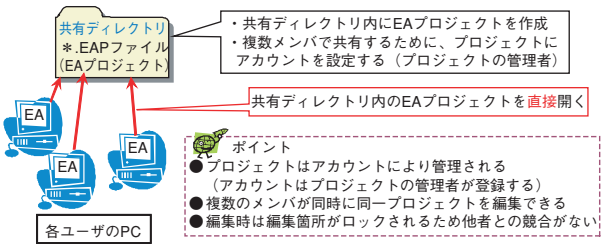

図２　EAでの情報共有の仕組み（一例）

まず、管理対象のコンテンツ(製造環境など)をネットワーク上のPC(ホストPCとする)のCVS管理下(リポジトリ)に登録し、各メンバはローカルPC上のEclipseを通して、ホストPC上で管理されているコンテンツをローカルPCに複製する。

作業(追加、編集、削除等)は複製したコンテンツに対して実施し、変更があった場合はEclipseを通してホストPC上のコンテンツに反映させる。また、この時、他者と競合(同じファイルを編集していた場合)が生じてもEclipseの機能でマージすることができる。

本手法の概略図を図3に示す。

(c) PukiWiki

webサーバにPukiWikiをインストールし、公開する。

情報発信者はwebブラウザを通してPukiWiki上にページを追加または編集する。

本手法の概略図を図4に示す。

4. 生産性向上ツールの有効活用

EAは、オブジェクト指向の開発フェーズをサポートする機能が充実しており、UMLからソースコード(Java、C++、C#、VB、VB.NET、Delphi、PHPに対応)を生成したり(その逆も可能)、ドキュメントを生成したりすることもできる。また、開発管理のサポート機能も有する。

このため、オブジェクト指向の設計開発を行う場合に有効で、全工程で利用することも可能である。

本ツールは有償であるが、比較的安価に設定されており、費用対効果は高い。

Eclipseは統合開発環境であり、標準ではJavaの開発環境が提供される。また、プラグインを追加することによって、他の言語(C、C++等)の開発を行うこともできる。

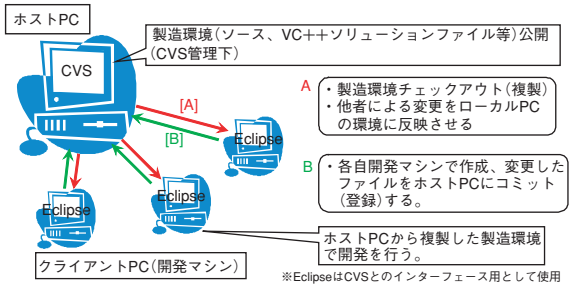

図3 Eclipseでの情報共有の仕組み(一例)

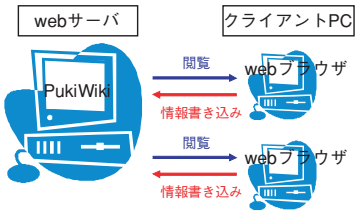

図4 PukiWikiでの情報共有の仕組み

本体、プラグイン共に無償で提供されており、ライセンスもCPL(参考7)であるため商用製品の開発に使用することも可能である。

本ツールは拡張性に富み、他のツール(エディタ、CVSなど)をプラグインとして組み込み、連携することもできる。

当部署ではCVSとのインターフェースにEclipseを使用している。機能が豊富で操作性も良いため、効率的にCVSを利用できる。

CVSは、チーム開発に限らずソフトウェア開発には非常に有効なツールで、ソースコードを始めとした作成資料のバージョン管理をサポートする。

基本的にバージョン管理できるファイルに制限がないため、WORDやEXCELで作成したファイル(バイナリ形式)も管理できる。

PukiWikiは、クライアント側に特別なツールをインストールする必要がない(webブラウザがあればよい)ため、不特定多数の人間が参加できる情報共有の場を提供することができる。管理者負担も少なくて済むため、導入しやすいツールの一つである。

当部署では、PukiWikiを早い段階で取り入れ、様々な情報・ノウハウの共有の場として利用してきた。具体的には、工事情報、スケジュール、現場環境に関する設定事項等(ネットワーク設定、プリンタの使い方、etc)の情報共有や、各種業務内容のまとめ、及び個々人の技術メモ代わりとしてこれを活用している。

現在は範囲を広げ、社内イントラネット上(参考8)で公開している。

公開しているPukiWiki の画面イメージを図5に示す。

メールや口頭でも情報共有は可能であるが、それらのまとめ先としてPukiWikiを介してネットワーク上で公開するのも効果的である。

図5 PukiWiki画面イメージ

5. チーム開発事例の紹介

実際に当部署で実施したチーム開発の事例を紹介する。

本事例の業務概要を表１に示す。

本事例は、MS-DOS上で動作しているプログラム（以下、DOS版とする）をWindows上で動作するプログラムにコンバージョンし、かつ、幾つか機能を追加するという業務である。

本事例では、技術提案・設計にEAを利用し、製造（コーディング）フェーズのバージョン管理にCVS（EclipseをCVSのインターフェースとして使用）を利用した。また、開発フェーズを通して、ノウハウ、トラブルシュートなどの情報を随時PukiWiki上に公開し、情報共有を図った。

本事例における、開発ライフサイクルと使用ツールの対応を図６に示す。

本事例は入札方式の案件であったため、技術提案資料を作成する必要があった。技術提案資料では、機能、アルゴリズムに対する記述書を始め、処理のフローチャート、データフロー、画面設計（画面のレイアウト、遷移など）資料を作成する必要があったが、都合上約１週間で仕上げなければならなかった。そこで、作業効率を上げるために、フローチャート、データフロー、画面設計などの作図を伴う作業には、サポート力のあるツールを取り入れることにした。そのために導入したのがEAである。その選択理由は、UMLで上記のいずれの図も作成することが可能であり、作成支援機能が充実しているためである。また、価格が安価であることも大きな要因であった。

表１　チーム開発事例の業務概要

業務概要	MS-DOSプログラムのWindows版制作	
言　語	C、Visual C++	
業務規模	期間	約半年
	開発メンバ数	４人
	コード行数（コメント行、空行含む）	約237KL
技術分野	官公庁システム	
主要技術	Windows GUI（MFC、MDI）、TCP/IP通信、シリアル通信	

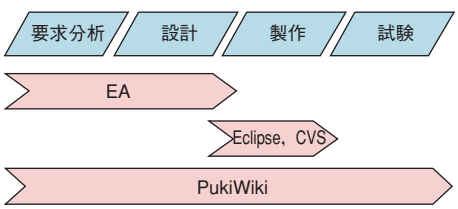

図６　チーム開発事例における開発ライフサイクルと生産性向上ツールの対応

こうして、短期間で技術提案資料を仕上げ、受注に至ることができた。

契約後の設計フェーズでは、技術提案資料作成時のEAデータ（UML）を流用した。画面設計については、客先からの要望によりできるだけDOS版の画面に近いレイアウトや操作性となるように実装することになったため、提案時の設計は使用できなかったが、その他の大部分の設計（フローチャート、データフロー）に関してはおおよそ流用することができた。

製造フェーズでは、CVSを用いてソースコードを管理した（バージョン管理）。

CVSを利用したバージョン管理は、他メンバの改修を上書きしてしまう等の心配もなく、改修を元に戻したい場合も容易に戻せる（任意のバージョンの状態に戻せる）ので、気兼ねなくコーディングを行うことができた。

本事例は当部署にとって経験の少ない分野であったため、個々人が同じような問題に直面することが多く、PukiWiki上での情報共有（問題に遭遇したメンバのトラブルシュート方法など）は、プログラミングを効率的に行う上で非常に有効であった。また、PukiWikiを通して、本稿で紹介した各ツールの利用方法、設定情報等に関する情報交換を行うことで、ツールに不慣れな時期も操作の理解や設定に各個人が費やす時間を短縮することができた。

6. むすび

本稿で紹介したようなツールをチーム開発に取り入れることで、管理負担を軽減でき、作業効率も向上させることができる。

情報共有は、組織の技術力向上に貢献するものであるが、実現手法が煩雑になるとなかなか定着しにくいものであるため、容易に実現できるツールを取り入れるのも有効な手段となる。

今後の課題としては、まだ導入したツールの利便性を十分に発揮できていないため、今後も積極的に活用し、そのノウハウを共有していく活動を推し進めたい。

PukiWikiにおいては、事業部内に公開してはいるものの、まだ一部の社員しか情報発信していないのが実情である。閲覧はしても、敷居が高く感じる、何を書けばいいのか分からないなどの理由から情報を書き込むことに躊躇する社員がまだまだ多いため、定期的に趣旨をアナウンスしていく努力が必要である。

本稿で紹介した生産性向上ツールは数あるツールの一端にすぎず、手法についてもこの限りではない。

今後も、プログラム設計の標準化や情報共有を推進すべく活動していく予定である。

参考文献等

(1) 参考サイト；http://www.sparxsystems.jp/ea.htm
(2) 参考サイト；http://www.eclipse.org/
(3) 参考サイト；http://www.nongnu.org/cvs/
http://www.wincvs.org/index.html
http://www.linux.or.jp/column/20000308.html
(4) 参考サイト；http://pukiwiki.org/
(5) Cunningham & Cunningham, Inc.が作成したシステム。
参考サイト；http://c2.com/cgi/wiki?WikiWikiWeb
(6) WikiClone(別名：WikiEngin)
WikiWikiWebと同じ機能を持つプログラムを指す。
(7) CPL：Common Public License
参考サイト；http://opensource.org/licenses/cpl.php