ソフトウェア工学 124－12
（1999．10．22）

ソフトウェア開発環境EvoManにおける構成管理機能の評価

佐々木 幹郎　田村 直樹
三菱電機株式会社 情報技術総合研究所

ベースラインとなる標準システムに対して変更を加え、複数の製品を開発する「発展型開発」を支援するソフトウェア開発支援環境EvoManの開発を行っている。EvoManは、発展型開発におけるソフトウェアの変更の管理、プロジェクト状況把握やソフトウェアの評価を目的としたソフトウェアの計測、そして開発者間の情報共有を実現する。現在、ソフトウェアの開発過程で生成されるソフトウェア生産物を管理する構成管理機能を実開発プロジェクトに適用しており、本稿では適用とその結果について記述する。

Evaluation of Configuration Management Function in Software Development Environment: EvoMan

Mikio Sasaki　Naoki Tamura
Information Technology R&D Center, Mitsubishi Electric Corp.

We are developing a software development environment named "EvoMan", which supports "Evolutional Software Development" for developing family-lined software products. The environment provides following facilities: 1) Managing software system changes in Evolutional software development, 2) Measuring software products for grasping the project and evaluating the software, and 3) Sharing engineers' knowledge and information about the software in the project.
We have applied EvoMan's software configuration management function to a practical software development project. In this paper, we summarize our experiences and evaluate the environment through the result.

1　はじめに

ソフトウェアの大規模化とそのための開発作業の複雑化が急速に進んでいる。顧客ニーズの多様化から様々な機能の組み合せを持った製品ファミリの開発が必要となっている。更に、短期間で品質の良いソフトウェア開発が求められる一方で、出荷した製品の保守も継続して行う事が求められる。

ソフトウェア構成管理(Software Configuration Management)は、開発および保守工程を通してソフトウェア生産物を識別・管理し、プロジェクトを効率よく運営するための素材となる情報を蓄積する。

我々は現在ソフトウェア構成管理を基盤として、開発プロジェクト管理の効率化、開発者間の情報共有を実現するためのソフトウェア開発支援環境EvoManの開発を行っている[1),2)]。本稿ではソフトウェア開発支援環境EvoManの構成管理機能の実開発プロジェクトへの適用とその評価について記述する。

2　発展型開発とその課題

複数の製品をラインナップとして持つ製品ファミリの開発や、顧客ごとにカスタマイズを加える必要のある製品の開発を行う場合、ある一つの製品をベースラインとし、そこから複数の派生製品を開発する開発形態が取られる。このような開発の形態を「発展型開発(Evolutionary Development)」と呼んでいる[3)]。

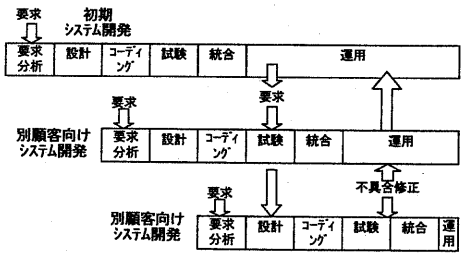

図 1 発展型開発モデル

発展型開発では、図 1に示すように、ソフトウェア開発のライフサイクルを、最初のソフトウェア（初期システム）の開発からその標準ソフト

ウェアを変化させながら繰り返し開発や保守を行い最終的にその役割を終えるまで、と捉える。一度開発したソフトウェアを顧客に提示することで顧客要求を効率よく獲得したり、ソフトウェア生産物の再利用による開発期間の短縮と品質の改善が期待できる。

しかし、その一方で次のような問題も発生する。

・版管理の複雑化
　派生した複数製品の開発・保守が並行して行われることにより、不具合情報に代表される設計情報や製品に対する情報管理が複雑化する
・プロジェクト管理の複雑化
　同様の理由から、ソフトウェアに対する機能追加や不具合修正など変更の影響の見積りが曖昧化し、プロジェクト管理が困難になる
・初期設計情報の消失
　一つの製品に対して長期に渡り開発が繰返される中で、当初の設計意図や制約などの設計情報が徐々に失われ、生産効率が低下する

EvoManはこれらの問題を解決し、発展型開発を支援する事を目的とした開発ツールである。

3　課題解決の方針

EvoManでは、上記の問題点を解決するために、構成管理機能を基盤とし、ソフトウェアの計測やプロジェクト管理の機能を開発者およびプロジェクト管理者に提供する。簡単に説明する。

3.1　ソフトウェア構成管理による版の記録

ソフトウェア構成管理活動[4) 5)]では、ソフトウェアの開発工程および出荷後の保守工程において、設計データやプログラムなどソフトウェア生産物に関する情報や、それら生産物に対する変更についての情報を記録する。変更記録の詳細は次章で説明する。

また、蓄積された情報を、計測やプロジェクト管理のための素材として提供する。

3.2　ソフトウェアの計測

構成管理によって得られるソフトウェアの変更情報から、開発対象のソフトウェアの状態やプロジェクトの状況を評価するための計測を行う。例えば、ソフトウェア変更量や、変更点の分布、変更頻度などの値を計測する。

3.3　開発の評価とプロジェクト管理

生産物に関する情報と、開発に投入されたリソースに関する情報との関りを定義し、その進捗状況に関する情報を明らかにする。例えば、発生した不具合に対する修正による作業量や、他の版に与える影響をすばやく見積るための指標を提供する。

本稿では、これらの機能の中で、構成管理機能と計測の一部について実装し、実プロジェクトに対して適用した結果について報告する。

4　ソフトウェア構成管理

EvoManの機能の実現にあたって、次の事項に注意した。

4.1　変更の記録と共有

発展型開発の過程では、過去に出荷した版の保守作業と新規機能の開発が同時に行われる。このような複数の開発が並行して行われている場合は、変更情報を記録し、開発グループ間での情報共有を支援する機能が必要である。例えば、あるソフトウェアに対して行った不具合修正の結果を共有する事で、同じ不具合に対して複数の開発グループで別々に対処するといった無駄な作業を低減させることが考えられる。

また、ソフトウェアの開発・保守期間の長期化や修正作業の繰返しによって、設計意図や背景情報など、初期設計の情報がだんだんと失われてしまう。こうした事態を避けるために、変更が発生した場合には、変更の目的と作業者の情報を記録・蓄積する機能を用意した。

4.2 ソフトウェアの状態およびプロジェクト状況把握のための計測

構成管理により蓄積された、ソフトウェア開発成果物の変更の推移に関する情報を、変更量、変更目的、作業者などを指標として計測する。

また、これらの計測結果は、投入されたリソースなどの情報や、上流の設計情報とあわせてプロジェクトの開発能力などが算出でき、プロジェクト管理に用いることができる。

4.3 日常開発作業とリリースへの対応

開発者がソフトウェア開発作業を行う場合に、動作が安定した動作安定版と作業中の版を意識的に分けて作業を行いたい場合がある。版管理機能として、動作が安定した版や、リリースされた版を作業中の版と区別して管理できるようにする事で開発者の作業を支援する。

また、複数の開発拠点での並行開発が行われている場合、ネットワークを通して、版に関する情報や開発状況が得られるようにする。

4.4 操作の簡易化・利便性の向上

これまで、構成管理ツールをあまり使っていなかった開発プロジェクトに対して、できるだけ学習などの導入コストをかけることなく利用可能にする。そのためのWebインタフェースによる簡単な操作環境を提供している。

5 適用

5.1 開発プロジェクトの概要

適用対象の開発環境の概要を表 1に示す。

表 1 適用プロジェクトの概要

適用ﾌﾟﾛｼﾞｪｸﾄ	組込みソフトウェア開発
開発者数	20人程度
開発規模	全体で1.5MLOC程度。 （ソースファイル数4000ファイル程度）

開発対象となるシステムは、基盤となる版をベースとして、順次機能追加のための開発が行われる。次のような特徴がある。

- 機能追加の開発と、不具合への対処を中心とした保守作業のための開発が並行して行われる
- 不具合対処のための版が数ヶ月おきにリリースされる

なお、本ツールの導入は、ファミリ製品群の開発プロジェクトの途中から（既にベースラインとなる版が出来上っており、機能追加の開発が行われる）となった。そのため、まず詳細設計の仕様およびソースコードの版管理を主眼としてツールの適用を行った。

5.2 開発環境およびスタイル

ツール導入前は、ファイルサーバ(UNIX系OS)上にソースが格納されており、Windows PC上からファイル共有を行っていた。版管理は、ディレクトリを版ごとに用意して作業を行うというスタイルをとっていた。

また、専用コンパイラを利用しており、このための専用のホストが用意されている。ファイルサーバとコンパイルホストとはファイル共有されており、ビルドの際には、各作業者が各PCからターミナルソフトを通してコンパイルホストにログインし、 コンパイル処理を起動していた。

5.3 システム構成

システム構成を図 2に示す。EvoManはファイルサーバ上にWebベースのアプリケーションとして構築されている。

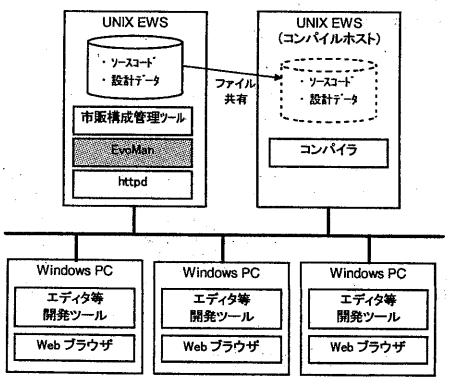

図 2 システム構成

開発者がPC上で作業を行う形態は以前と同様である。EWSホスト上の構成管理ツールに登録された設計データやソースコードをWebインタフェースを通して自分の作業領域に取出す(チェックアウト)。編集などの作業が終了すると、新しいバージョンとして登録する(チェックイン)。

登録の際には、図 3に示すように、変更の目的や内容を記入するようになっている。これは、本プロジェクト向けに必要な項目を定め、カスタマイズを行った。

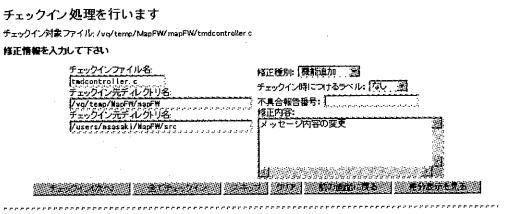

図 3 変更情報入力

また、編集作業以外は、ほぼ全てWebインタフェース上で作業可能な形態を採用している。

5.4 作業バージョンとリリース管理

図4に今回の適用における版管理の概要を示す。サンプルとしてabc.cというファイルに対する版番号およびリリース管理のための識別子(ここでは「ラベル」と呼ぶ)の付け方を示している。

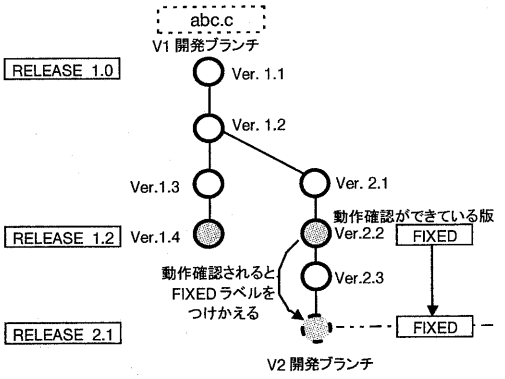

図4版管理とリリース

各構成管理要素に対して、“FIXED”という動作確認済を意味するラベルと、各リリースに対応して作成されるリリースラベルの二種類のラベルをつけることで、作業中の版と、動作確認された版、そして、製品としてリリースされた版を管理している。何もラベルがついていない版が作業中の版である。

6 適用結果

ツールを実プロジェクトに適用し、ソフトウェア開発を行った。その過程で一回のリリース作業を行った。結果について記述する。

6.1 変更情報の記録

開発者にはチェックイン時に、修正の目的、修正内容、修正IDを入力してもらう。作業者名、時間などは自動的に記録される。これらの情報は各版のファイルに対し属性として記録する。これにより、変更の追跡が可能になり、作業者の責任範囲が明確になった。

また、修正IDと修正仕様(別管理)との組み合せで、修正に対する影響範囲の検出が可能になった。

6.2 作業状況の共有

Webインタフェースを通して、チェックアウト状況の表示が行えるようになり、他の作業者の状況や、ソースファイルに対する修正に関する情報を作業者間で共有できるようになった。

6.3 操作の利便性

開発者側はユーザ登録し、Webブラウザを用意すれば本ツールを利用可能になる。また、チェックアウトやチェックインに必要とされる入力項目はガイドされるので、直感的に利用できる。あまり学習時間を必要とせずツールを簡単に使えるようになった。

6.4 測定の方法とその結果

同一製品に対して複数のリリースがある。リリースされた製品とその変更量を調査した。

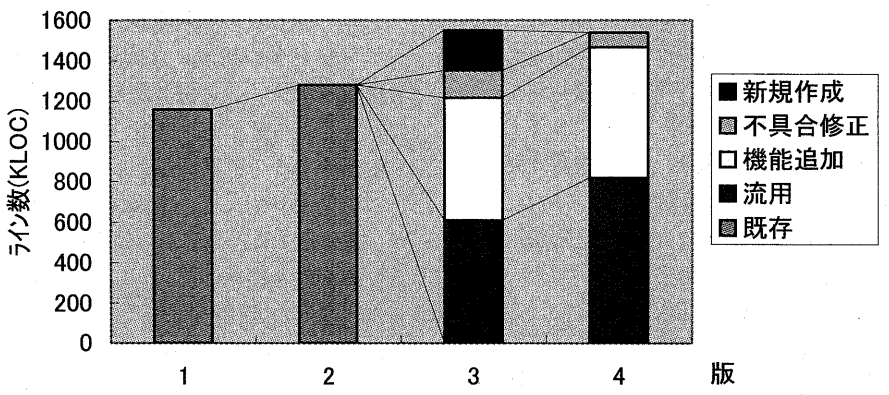

図5 変更量の版による変化

ここでは計測結果の一例を図5に示す。版1,2,3はツール適用前に開発された製品である。第3版をベースラインとして構成管理を開始し、第4版の開発を行った。版が進むにつれ、変更が収束していくことがわかる。また、第4版開発の変更量の詳細な変遷を図6に示す。3.1,3.2,3.3は第4版開発の過程で発生した安定版である。

このように、蓄積される変更情報から、変更とその目的の分類が可能になった。

さらに、詳細な変更内容として、変更の回数や修正コメント、修正IDの確認、変更が集中するファイルの特定なども行うことができるようになった。

7 考察

ツール導入の効果としては、これまで行えなかったソースコードの変更管理、リリースごとの版の管理が、ツールの導入によって可能になった事が最も重要な点である。

また、ツール利用に伴い、蓄積された情報を自動的に測定できるようになった事を、計測の結果例として示した。現在は、ソースプログラムの流用量や変更率が得られるようになっており、開発が進んでいくにつれ変更が収束しているといった傾向の読取りが可能になることが確認できた。今後、単位時間や投入したリソースなどの情報と連携してソフトウェアの生産性の評価を行えるようにする必要がある。

また、現在はプログラムの行数(LOC)による計

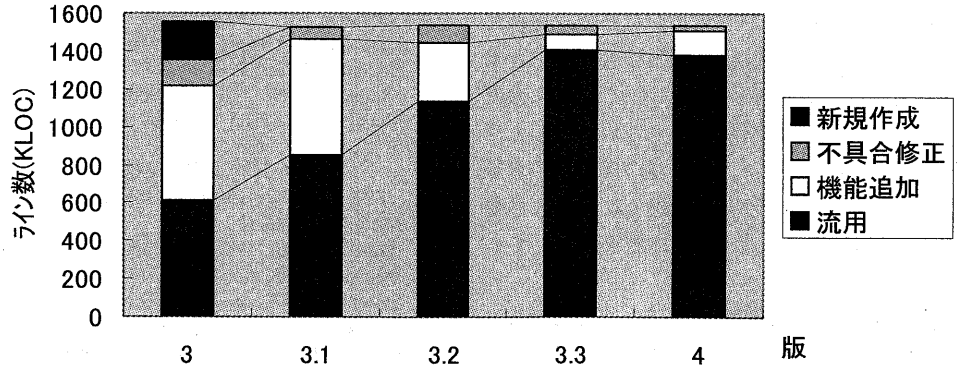

図 6 変更量の版による変化（詳細）

測を行っているが、開発量に対して行数だけでは見積れない機能の重み付けを考慮する必要がある。これらの問題については最近ではFunction-Point法[6]などが利用されてきており、より上流の工程への適用とあわせ、見積りに対する評価の形で計測可能になるように検討を行っている。

今回適用したプロジェクトは、開発者ほぼ全員が最初の設計段階から開発に関っており、設計意図など共通の知識が保たれている。このため、ツールによる変更情報の蓄積・共有の効果により再利用率が向上したかという点について、その効果は不明確である。しかし、ソースコードに手を入れるWhite-box型の再利用であるので、今後開発が長期に渡り継続されたり、開発メンバが変更になる場合に、蓄積された変更情報が利用され、その効果について明らかになるのではないかと考える。

8 問題点・課題

現在、EvoManの適用については、一回のリリースのための開発が経過した段階であり、今後いくつかの問題点を解消して、より効果的な利用が行えるようにしていかなければならない[7) 8)]。現在、以下に述べるような問題点が指摘されている。今後、これらの問題点を解消し、適用・検証を継続する。

8.1 適用工程の拡大とプロジェクト管理

今回は、詳細設計とコーディングという比較的下流の工程に対する適用のため、設計意図や背景となる情報の整合性確保のための上流成果物とのフィードバックなど行えなかった。今後、上流へ適用範囲を拡大して、より上流の成果物の管理、トレーサービリティの確保、見積りに対する評価モデルの確立など行っていくことを検討している。

また、蓄積された成果物の完了の度合による進捗管理機能などを備えたプロジェクト管理ツールや、不具合報告と変更情報を組み合せた品質情報管理ツールとの連携を実現する。

8.2 ツールの動作速度

ヒアリングの結果、反応速度について、利用者は不満が強いことがわかった。特に、構成管理の対象となるディレクトリ・ファイルをブラウズする部分でその不満が強い。

この問題に対して、細かな改良を加えているが、根本的な問題の解消には至っていない。ネットワークの状態や、呼出されるバージョン管理ツールの速度など、速度に与える要因はさまざまであり、今後、処理速度調査を行い、対策を行う予定である。

9 まとめ

以上のように、ソフトウェア開発環境EvoManの構成管理機能を実プロジェクトに適用し、結果として得られる情報の簡単な評価を行った。構成管理を行う事で、開発時や出荷時の製品の管理が容易になる。また、変更に関する情報を蓄積し、プロジェクト管理や生産性向上に役立てることができる。課題も多く残されているが、今後は課題を解決しつつ、より使やすい環境を目指して開発を続けて行く予定である。

参考文献

1) 柳生、田村、佐々木: "S/W部品庫EvoManにおける品質管理機能," 情報処理学会研究会報告(98-SE-120-19), pp.133-140 (1998)
2) 佐々木、柳生、田村: "ソフトウェア再利用環境EvoManにおける構成管理機能," 情報処理学会研究会報告(98-SE-122-12), pp.87-92 (1999)
3) Dorfman, M.: "Requirement Engineering,"*Software Requirement Engineering Second Edition*, IEEE Computer Society Press (1997)
4) Berlack, H. R.: "Software Configuration Management," John Wiley & Sons, Inc.(1992), 水田浩監訳, "ソフトウェアコンフィグレーションマネージメント," 日刊工業新聞社(1997)
5) IEEE Std828-1990: "Software Configuration Management Plans"
6) Albrecht, A. J.:"A/DM Productivity Measurement and Estimate Validation", IBM Corporate Information Systems, IBM Corp., Purchase, N. Y., May 1984
7) Schamp, A., and Owens, H.: "Successfully Implementing Configuration Management," IEEE Software, Vol.30, No.1, Jan. 1997
8) Dart, S.: "Achieving the Best Possible Configuration Management Solution", CrossTalk -The Journal of Defence Software Engineering, (Sep. 1996)