ソフトウェア工学 120－19
（1998．7．24）

S/W 部品庫 EvoMan における品質管理機能

柳生 理子[†] 田村 直樹 佐々木 幹郎[†,†]

ソフトウェアシステムの高度化と顧客要求の多様化に効率良く対応する目的で、我々はソフトウェア部品庫 EvoMan の開発を進めている。ソフトウェア部品庫 EvoMan は、発展型ソフトウェア開発のアプローチを支援する目的で、ソフトウェア部品の再利用を支援するための複数の視点からのソフトウェア部品の検索機能、ソフトウェア部品に対する変更を管理するための構成管理機能、そしてソフトウェア部品の品質管理と変更の理由付けを記録する目的で品質情報管理の機能を提供している。本稿では、このうちとくに品質情報管理機能を中心にソフトウェア部品庫 EvoMan の機能を紹介する。

Quality Managemnet Supports in A Software Repository EvoMan

YAGIU RIKO,[†] TAMURA NAOKI[†] and SASAKI MIKIO[†]

Software reuse and configuration management are the key technologies to succeed evolutional software development approach. We are developing a software repository named EvoMan, to support this development approach. The major facilities of EvoMan is querying reusable software components form various perspectives, managing configuration of existing software components, and sharing quality information among developers.

In this paper, we are introducing quality management facilities of EvoMan, and how the quality information support systematic software reuse and evolutional software developments.

1. はじめに

ソフトウェア（S/W）システム開発では、近年、システムの複雑化・高度化が進むと同時に、顧客要求の多様化への対応が難しくなってきている。一方で、開発組織は、つねに Time-To-Market の短縮、製品の高品質化を求められている状況にある。

これらの課題に対応する方法のひとつに、「発展型 S/W 開発」のアプローチ[1)2)] がある。これは、例えば S/W 製品のファミリ展開や、既存システムの漸進的な拡張・改良によるシステム開発においてよく見られる形態である。基本となる S/W システムを用意しておき、これに様々な変更を加えながら、各顧客の多様な要求に答えていくアプローチと言える。

この「発展型 S/W 開発」を実現するためには、次の2つの技術的な課題を解決する必要がある。ひとつは、S/W の組織的な再利用の実現であり、もうひとつは S/W に対する「変更」を管理することである。我々は、こうした作業を支援することを狙って、S/W 部品庫 EvoMan☆の開発を進めている。S/W 部品庫 EvoMan では、蓄積された S/W 部品の検索を支援する為の機構、変更理由や日付などの情報とともに部品を管理するための構成管理の機構、さらに変更などの品質情報を管理し、流通するための手段として品質情報管理の機能を提供している。

本稿では、このうち特に品質情報管理の機能について、紹介する。以下2章では、発展型 S/W 開発のアプローチについて述べる。3章では、我々が開発を進めている EvoMan の狙いと主要な機能について概説する。4章では、特に EvoMan で提供している品質情報管理機能について述べる。

2. 発展型 S/W システム開発

2.1 S/W システムの変化

一般的に、S/W 開発は図1の様なライフサイクルを辿り、S/W システムに対する要求の変化と変更は、開発開始から、運用に至るまでのあらゆる側面で発生する。

† 三菱電機情報技術総合研究所
Mitshbishi Electric Information Technology R&D Center

☆ Evolution-Manager

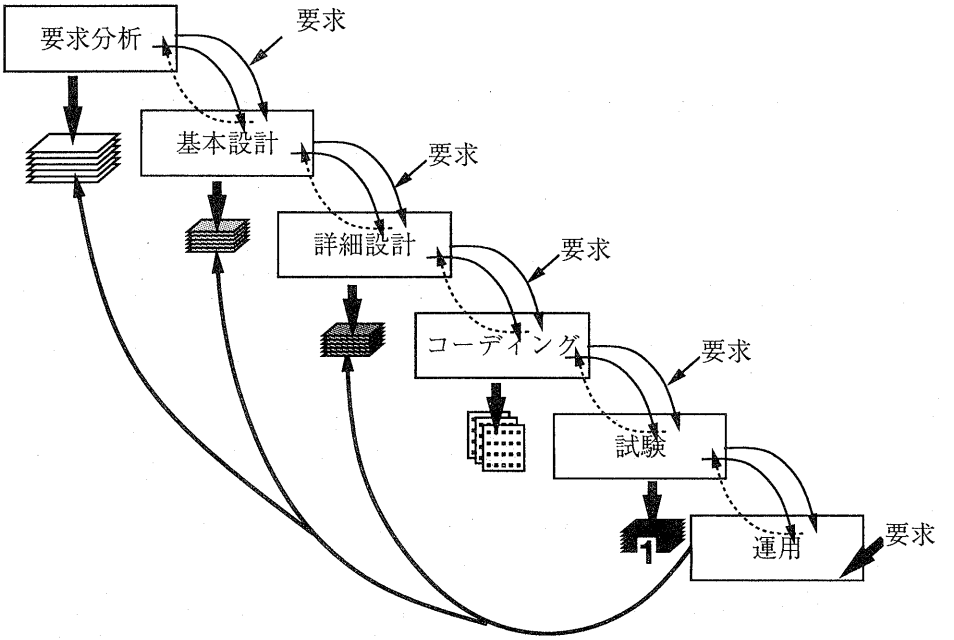

図1　S/W 開発の流れと変更

図1において四角で囲ったものは各工程を表し、各工程から出ている矢印の先のものはその工程の生産物を示す。生産物は順に、要求仕様書、基本設計書、詳細設計書、コード、試験成績書（品質情報）を表している。これらのS/W システムに対する顧客要求の変更は、様々な理由で発生[9)10)]する。例えばBoehm は、こうした要求変更のうち、平均して25% は避けることが出来ないものと述べている[11)]。

さらに、システムの運用後には、さらに頻繁にシステムに対する要求の変化が発生する[12)]。こうした開発においては不具合および要求は、その度合によって各工程へとフィードバックされる。従来、運用後の作業はS/W 保守として扱われてきた。しかし、ソフトウェア保守の55% が機能拡張等の顧客要求変化への対応、25% が実装環境の変化への対応を目的として行われるとされている[13)]。こららのことから、S/W 保守作業も、その多くが顧客要求の変化への対応と捉えることが出来る。

このように、S/W システム開発では、こうした変化に対応してソフトウェア・システムそのものを継続的に修正・拡張していくことが求められる。

2.2　S/W 再利用と問題点

継続的な修正・拡張への対応として、従来からS/W 再利用の適用に対する期待は大きい。S/W システムの継続的な修正、拡張に対応する方策として、従来からS/W 再利用の適用への期待は大きい。特に、近年のS/W システムの規模の複雑化、高度化は、ゼロからのS/W システム開発をより難しいものにしている。こうした中、S/W 再利用は、高品質のS/W システムを短期間に市場へ投入するための手段として、大きな注目を集めている[3)]。

S/W 再利用技術では、これまでに、様々な提案がなされている。これらは、次の3種類に分類することが出来る。

プログラムコードの再利用　S/W モジュールの単純な再利用からはじまり、再利用コードの適用範囲拡大を狙ったアプリケーションフレームワーク開発[4)]が幅広い適用分野で行なわれている。さらにアプリケーションフレームワーク開発の設計手法としてのデザインパターン[5)]の適用行なわれてきている。

S/W システムの仕様記述からの再利用　S/W 再利用部品を準備しても、利用者が適用対象範囲を明確に出来ないという問題がある。このことから、適用対象ドメインを明示しよう、とするドメイン分析[6)]のアプローチが提案され、幅広い適用もなされてきている。

再利用型のS/W 開発プロセス　再利用を前提としたS/W 開発のアプローチ[7)]が幾つか提案されている。S/W 部品の開発チームをS/W 部品の利用者

（アプリケーション開発チーム）と異なる組織であることを前提として、S/W システム開発の進め方を示している。

しかし、これらの技術の幅広い適用がなされている一方で、未だにソフトウェア再利用は必ずしも広く普及している状況には無いという問題提起も多い[14]。

2.3 発展型 S/W 開発アプローチ

特に、大規模システム開発では、ユーザ自身がそのシステムで何をしたいのか良く判っていないことも多く、たとえユーザが彼ら自身の要求を判っていたとしても長期に渡る開発の中で要求を変更せざるおえなくなることは多い[8]。

この状況に対処するライフサイクルモデルの一つとして、発展型ライフサイクルモデルが提案されている。図2は、発展型 S/W 開発のプロセスモデルを示す。

発展型 S/W 開発においては、始めに標準となる S/W を開発し、それ以降の開発では、ここで開発されたシステム（標準 S/W）を部品として用いて開発を行なう。部品として用いられる S/W は、カスタマイズがなされ、また様々な組み合わせで用いられる。こうして作られた製品自身もまた、最初の部品から派生された部品として、他の S/W 開発に用いられる。このように、発展型 S/W 開発では、単純に標準部品を再利用するのではなく、部品自身も「変更」され、また開発された S/W 自身も部品となり得る。

この発展型 S/W 開発のアプローチの利点として、次のものがある。

- 基本となる S/W システム製品をベースとして、複数の版の S/W システムが段階的に開発される。各々の版の開発は、基本となる製品をベースに同時並行で行なうことも可能である。
- 基本となる版（特に標準部品）の多くの部分を流用することにより、S/W システムの高品質化を期待出来る。

こうした利点がある一方、後のプロジェクトで発見された不具合等は、以前の版にも反映していかなければならず、こうした型の開発をしない場合にはない手間がかかる可能性が出てくる。そのため、S/W 開発作業の中で、プロジェクト間での情報の共有化を図ることが重要となる。

3. 発展型 S/W 開発の課題と Evoman の狙い

3.1 発展型 S/W 開発の実現上の課題

発展型の S/W 開発を円滑に進めるための技術的課題として以下のことがある。

S/W 部品の適用ドメイン異存性

S/W 開発を繰り返しながら再利用部品を蓄積して行く方法は、S/W 部品を蓄積する手段として、しばしば用いられている。しかし、出来上がった S/W 部品の適用範囲が明確化されないことが多い。

部品調査や使用のための情報が不十分

部品を利用する為には、部品の仕様書以外の例なども含めた使用法が必要であるがドキュメント化されていない場合が多い。一方、複雑に絡んだコードからこれらの情報を作るには、ドメイン知識を前提とする。しかし、これには人手もかかり、調査に費用がかかる。

断続的な変更要求

変更は「試験」「保守」工程においても発生し、運営に入ってからの「要求のバグ」等への対応など、上流からだけではなく、下流からも上がって来る。

ノウハウ共有が難しい

S/W 部品のメリットは、長期に渡って（複数のプロジェクトで）利用されることで生まれる。しかし、現実にはプロジェクト単位で異なる要求をどう反映させるか、が大きな課題となる。

一般に、S/W 開発は、数名から数百名に及ぶチームで行なわれる。チームの編成は、部所を越えるだけでなく協力会社など、社外に及ぶ場合も頻繁に発生する。また、複数のプロジェクト☆が並行して開発されている状態も頻繁に発生する。従来は、「人」をメディアとした方法が主流だったが、上述の状況に加え長期に渡って用いられる S/W の場合は、開発当初の技術者に直接コンタクトをとることが難しいのが現状である。従って、情報の共有化を図ることが大きな課題となってくる。

これらの技術的課題の解決を図るため、開発手法、管理技法、そしてツール支援を整備していく必要がある。従って、「発展型の S/W 開発」においては次の前提条件が必要である。

☆ 同一製品の異なるバージョンを作る為のプロジェクト。

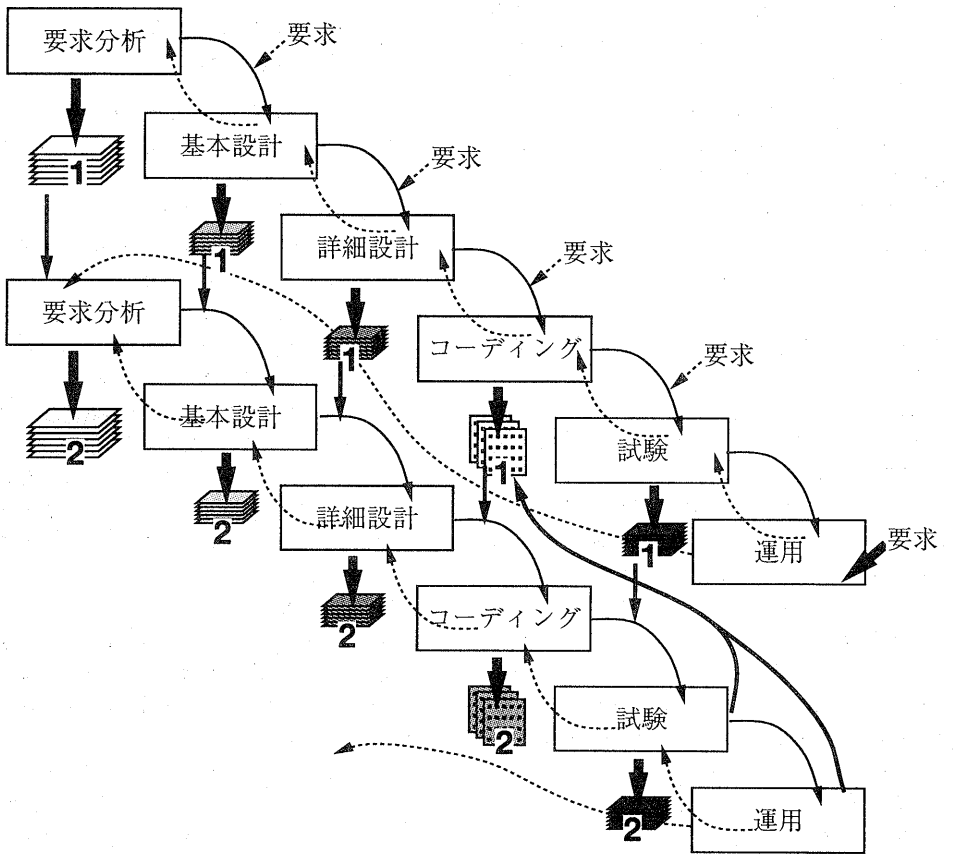

図 2　発展型 S/W 開発の流れ

- 初めから再利用されることを前提として、S/W 部品が作られていること。
  このためには、再利用の対象ドメインが明確になっていること、ある程度、再利用を意識した構造となっていることが必要である。
- 変更要求を素早く伝えるための機構があること。
  断続的な要求変更は避けられないものであるとの前提に立ち、これをなんらかの形でサポートを与えるものが必要である。
- 人以外のメディアによる情報伝達の機構があること。
  このためには、導入当初にある程度の情報があること。部品を用いた開発者の経験や知識を蓄積し、それを共有化するための仕組みがあることが必要である。

従って、部品導入（使う為の準備）、部品の保守・管理（特に構成管理）、品質の情報の管理・流通の 3 つのコストを下げることが成功の鍵を握ると言える。

3.2　S/W 部品庫 EvoMan の概要

前述の問題点に対し、我々は発展型 S/W 開発を成功させる上での課題として、次の三つのものを設定し、これらの課題の解決の為に部品発展型再利用の開発支援環境である「S/W 部品庫-EvoMan」の開発を行なった。

- 部品を用いる際に、部品の意味、制約、使い方、など部品に関する様々な情報を、必要に応じて適切に取り出し、S/W 部品の理解を支援する。
- 部品の構成・枝分かれを的確に管理し、関連する部品をスムースに検索し取り出すための、構成管理を行なう。
- 部品を用いて開発し部品の不具合が発生した際に、関連する部品を用いている開発者や部品の管理者へ、関係する部品の情報を、素早く、的確に不具合とその解消の情報を伝える。

3.3 EvoMan の機能

EvoMan では、これらの課題の為の 3 つの「部品および付随する情報検索」、「部品の版および版の構成管理」、「品質情報の管理」機能を持っている。それぞれの機能は以下を目的とする。

(1) 部品および付随する情報検索（部品情報検索）
開発者の意図や S/W 部品の使い方など、S/W 部品の使用の際に必要となる様々なノウハウや周辺知識を、容易に的確に獲得する。

(2) 部品の版および版の構成管理（版管理）
どの版を部品として用いることが適当かの判断を、容易に的確に行なう。また、それに適した部品管理を行なう。新しくできた版を新たな部品として登録する際に、容易に適切な（使用時の配慮をなされた）部品管理を行なう。

(3) 品質情報の管理
部品の不具合が発見された、或はカスタマイズの際に不具合が発見された場合、状況を直ちに、正確に、関係者全員に伝える。
部品の不具合が修正された、或はカスタマイズの際の不具合が修正された場合、直ちに、正確に、関係者全員に修正箇所を伝え、また修正された部品を配布する。

3.4 EvoMan に蓄積する情報

先に述べたように、EvoMan は管理する S/W 部品を用いた開発のサポートを目的としている。そのためには、S/W 部品そのもの（コード）だけでなく、その仕様や仕様方法などの情報、品質やカスタマイズなどの変更に関する情報など、周辺の情報を含めた利用のサポートを行なう必要がある。また、そのため、検索、蓄積、保守・管理に必要なデータも併せて蓄積・管理する必要がある。以下の図 3は、EvoMan で扱うデータを表したものである。図 3中の付随情報とは、仕様書、モデル図、ソースそれぞれの版毎の作成時の情報、例えば作成理由や作成者や作成日時など、である。この付随情報は版情報を識別するラベルとしても使用される。

このように、EvoMan では部品のソース、仕様書、モデル図を、版の情報と品質情報とを結びつけ、部品の管理を行なっている。

4. EvoMan における品質情報管理

S/W 開発時における品質に関する情報は通常、試験時を中心に発生する。具体的なデータとしては、不具合発見時および修正時に作成される不具合の内容や発生状況、発見者などを中心とした情報と、それらの情報から解析されるプロジェクト全体の情報がある。特に、発展的 S/W 開発においては、一つの S/W 利用する開発者が、通常では情報交換する必要のない複数の開発プロジェクトに渡る場合が多々ある。現在、開発が進行中のプロジェクトだけではなく、既に S/W のリリースが行なわれたプロジェクトとの情報交換もありうる。そのため、これらの品質情報の正確な伝達は、S/W の品質の向上と開発工程の短縮の上で非常に重要となってくる。

4.1 S/W 開発現場における品質管理の現状

しかし、これらの情報の作成、管理は一般に、S/W 開発現場での紙ベースなど手作業のみによるトラブルの管理を行なっているのが現状である。この為、生じる問題としては以下のものがある。

- 不具合発生の申告洩れが生じる。このため S/W のバグが埋もれる。
- 不具合処理がどこまで進行しているか把握できない。
- 品質情報の検索、及びデータ解析が難しい。
- 過去のデータ紙で探しながら、書類を作成するのに手間がかかる。

特に発展型 S/W 開発を行なっている場合に顕著になる問題としては以下のものがある。

(1) 連絡のタイミングがずれる。

(2) 関連する部品利用者など連絡しなければいけない開発者が複雑化し、連絡洩れが生じる。

これらのことから、結果として、試験が二度手間になる、同じ問題に対する修正を複数人が行なうなど修正結果のマージが難しくなるという問題が生じる。

4.2 EvoMan における品質管理機能

これらの問題を受け、EvoMan においては、部品を利用した開発時、その中でも試験・デバッグ工程におけるトラブルの発生とその修正時のサポートを中心としている。中でも我々は特に大きな障害の原因を、品質情報が電子化されていないことから生じる以下の問題に絞った。

- データの作成作業が簡略化できない。
  関連するデータの検索、過去のデータも利用できず、また入力の手間が省けない。
- 情報の正確かつリアルタイムな伝達が出来ない。
  不具合の発生や修正に関する情報作成、あるいは不具合の発見および修正作業そのものと、それらの情報流通との同期がとれない。
- 単純な紛失、処理のし忘れが頻繁に起こる。
  処理が滞り、またプロジェクトの流れ・進捗を追うのが難しい。

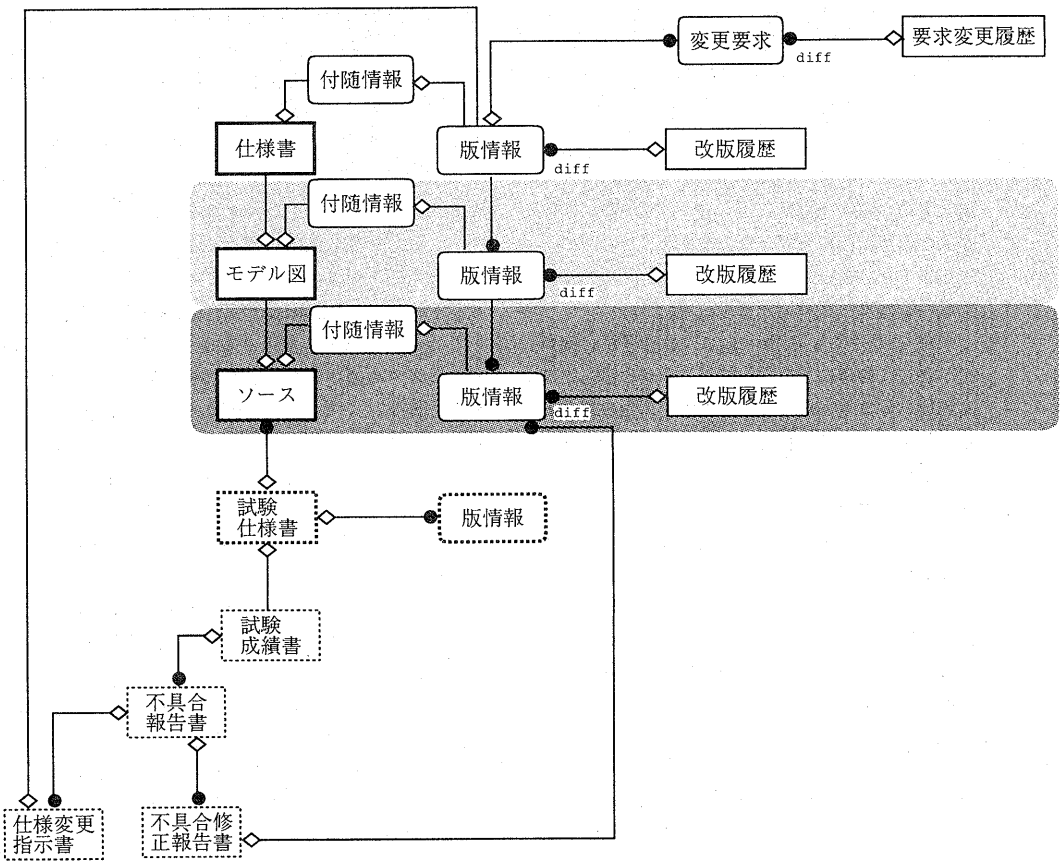

図 3　EvoMan で扱うデータ構造

- データの集計および解析に非常に手間がかかる。
  リアルタイムなプロジェクトの予測や計画、および品質の測定が出来ない。よって、プロジェクト進行上の重大な問題を見逃し、結果として製品の品質を低下させるおそれがある。

これらのことを考慮し、以下の機能を用意した。

(1)　品質情報の伝達。
以下の機能により、品質情報とその流通の電子化を行なう。

- 不具合処理の書類の電子データ作成補助。
  - 過去のデータ、関連するデータの検索の補助。
  - 入力の補助。選択項目の表示、日付や書類の一連番号などの自動入力。
  - ラフなデータ入力。最低限の入力☆を行ない、その他は後から補う。
- ネットワーク上での情報公開、および情報の送受信。
  - 不具合発生の報告のリスティング。
  - 不具合発生の関係者への伝達。
  - 処理のし忘れを確認補助。

(2)　部品の変更管理。
構成管理ツールを利用する他、1 の機能との連係により変更の情報をネットワーク上で流通・管理する。

(3)　プロジェクト管理データの自動取得。
構成管理ツールなどを利用し、ツールのはきだす品質情報を蓄積。

4.3　EvoMan における不具合処理と品質情報の蓄積

不具合に関する情報は、以下の作業と共に発生する。

1. 部品のコード或は仕様の不具合の発生（発見）
2. 不具合内容の詳細化

☆ 検索し得る為のデータ、忘れてはならないデータ、すぐに必要なデータ

3. 不具合の箇所の特定
4. 解決策の決定
5. 修正対象への作業
6. 修正内容の伝達
7. 不具合に関する検討・解析

不具合の発生する段階は、コードの不具合はテスト時が最も多いが、部品の調査時、カスタマイズ時、運用時などS/Wのライフサイクル全般に渡る。また、仕様の不具合は要求の変化も含む為、発生時期はさらに散らばった分布となる。

不具合内容の詳細化の作業では、まず仕様の不具合か、プログラムの不具合かに分類する。次に不明確であったり抽象的であったりする内容を、より具体的な要求へと分析、不具合項目の分割、などを行なう。この作業の後、プログラム上の問題がある場合、不具合の原因を機能、カテゴリー、モジュールと原因を狭め、プログラム上での箇所を特定を行なう。

変更対象となるモジュールが明らかになると、モジュールおよび対応する改訂対象となるドキュメント類について変更方法、担当者、変更スケジュールなどを決定し、決定された対象の変更や追加などの作業を行なう。

修正作業と並行して（或は前後して）、変更された部品を継承或は組み合わせて作られている別な部品、変更された部品と同じ種類で版の異なる部品を特定し、関係者へ伝達する。

これらの作業の後、不具合に関する検討・解析の作業を行なう。そこで、次にこれらについて述べる。

4.4 不具合原因の検討・解析

EvoManを用い、上述の作業を進めた結果、蓄積された情報を以下の項目にまとめ、不具合原因の解析を行なう。

(1) 不具合提出状況
- 誰が提出した、プログラム不具合あるいは要求の変更か？
- 開発のどの段階で提出された、プログラム不具合あるいは要求の変更か？

(2) 不具合箇所特定に関する作業量
- 時間的側面から測定した作業量はどれくらいか？
- 調査したコードおよびドキュメントはどのようなものか？
- コードおよびドキュメントの調査方法はどのようなものか？
  （用いたツールはあるか？相談や議論を行なったか？など）

(3) 不具合に関連する修正作業量
- コードの物理的な修正量はどれくらいか？（ライン数）
- ドキュメントの物理的な修正量はどれくらいか？（ページ数）
- コードの修正に要した時間はどれくらいか？
- ドキュメントの修正に要した時間はどれくらいか？

(4) 不具合の深刻度☆

(5) 不具合箇所の特性

(6) 不具合が出てきた段階、引き起こした要因
以下の側面から分析を行なう。
- 仕様の変更、客先による要求変更、他の部品の変更に追随した仕様変更
- 仕様書の誤りの場合の原因は？
- ソースコードの問題の場合の原因
- 人為的な側面による分類☆☆
- 関連するS/W部品の理解不足
- ケアレスミス
- 試験側の問題
- その他

EvoManで集積されたデータは、プロジェクト管理に用いることを念頭に置いている。これらのデータの中でも、特に数値化できるもの、例えば、各作業にかかった工数、工数と修正量との掛け合わせを各開発毎に集積し、数値の上で、プロジェクトの傾向を掴むことが出来る。

また、（6）で上がってきた、要因による場合分けとの掛け合わせで、どの様な要因からどの程度の作業量の不具合が発生するかと言う傾向を知ることも出来る。

この様な情報の分析の結果、以下のデータが収集出来る様になる。
- プロジェクト毎の問題の発生の数の減少具合
- 開発者毎の問題の発生の数の減少具合
- プロジェクト毎の問題の原因・特徴の割だし
- 開発者毎の問題の原因・特徴の割だし
- プロジェクト毎の問題の解決法の割だし
- （開発者毎の）問題の解決法の蓄積

このようなデータを蓄積して行くことで同種の問題に対する解決のヒント集の作成も可能となる。

☆ システムダウン、誤動作、性能上の問題などに分類。
☆☆ 知識の不足および誤解部分、情報流通、スケジュールなど

5. ま と め

本稿では、S/W 部品を用いたシステムの発展的な開発を支援の課題と我々のアプローチである EvoMan について、その品質管理機能を中心に述べた。EvoMan の一つの特徴は品質管理とその他の支援機能を融合することにある。例えば、品質情報を、S/W 部品の版のラベルの一つとしてつけておくことで、版が作られた時、どの様な事情（不具合の発生、要求の変更など）で、どの様な変更経緯を辿ったのかを追うことが出来る。さらに、それらの情報自体のみならず、情報を追う作業自体で部品の作成意図が見えてくる為、理解の助けになる。さらに、品質管理情報は、プロジェクトの解析・予測に用いることで、プロジェクトの改善、要求変更の予測にも非常に有効である。

今後、運用を通じてデータの収集を行ない、プロジェクト管理機能を充実化させて行く予定である。

参 考 文 献

1) Mayer, A.V., and Hirsh, B., "Productivity Improvement With Evolutionary Development," Proceedings of The Fourteenth Annual International Computer Software & Applications Conference (COMPSAC90), pp.323-329, IEEE Computer Society Press (1990-10).
2) Dorfman, M., "Requirements Engineering," Software Requirements Engineering, Second Edition, pp.7-21, IEEE Computer Society Press (1997).
3) Griss, M.L., and Wosser,M., "Making Reuse Work at Hewlett-PAckard," IEEE Software, Vol.12, No.1, pp.105-107, IEEE Computer Society Press (1995-1).
4) Calder,P., and Linton, M., "The Object-Oriented Implementation of a Document Editor," OOPSLA 92 Conference Proceedings, pp.154-165, ACM Press (1992-10).
5) Gamma, E., et.al.,"Design Patterns - Elements of Reusable Object-Oriented Software," Addison-Wesley Longman, Inc.,(1995)
6) Prietro-Diaz, R., "Domain Analysis: An Introduction," ACM SIGSOFT Software Engineering Notes, Vol.15, No. 2, pp.47-54, ACM Press, New York, NY (1990-4).
7) Karlsson, E., Software Reuse: A Holistic Approach, John Wiley & Sons (1995).
8) Brooks,F.P.,Jr., Chairman, "Report of the Defense Science Board Task Force on Military Software," Office of the Under Secretary of Defense for Acquisition, U.S. Department of Defense, Washington D.C., (1987).
9) Harker, S.D.P., and Eason, K.D., "The Change and Evolution of Requirements as a Challenge to the Practice of Software Engineering," Proceedings of the IEEE International Symposium on Requirements Engineering (RE 93), pp.266-272, IEEE Computer Society Press, Los Alamitos, CA (1993).
10) Lubars, M., et.al., "A Review of the State of the Practice in Requirements Modeling," Proceedings of IEEE International Symposium on Requirements Engineering (RE93), pp.2-14, IEEE Computer Society Press (1993-1).
11) Boehm, B.W., "Software Engineering Economics," Prentice-Hall (1981).
12) Gomaa, H., "Reusable Software Requirements and Architectures for Families of Systems," Journal of Systems Software, Vol.28, pp-189-202 (1995).
13) Pigoski, T.M., Practical Software Maintenance, Best Practices for Managing Your Software Investment, John Wiley & Sons (1996).
14) Meyer, B., and Effelsoft, "The Next Software Breakthrough" IEEE Compter, Vol. 30, No. 7, pp.113-114 IEEE Computer Society Press (1997).