自動データ収集･可視化ツールを用いた リアルタイムフィードバックシステムの構築と試行

松村知子[i]，勝又敏次[ii]，森崎修司[i]，玉田春昭[i]，大杉直樹[i]，
門田暁人[i]，楠本真二[iii]，松本健一[i]

概要

プロジェクト管理のためのデータ収集と分析，それに基づくプロセス改善などの活動は，ソフトウェアの品質向上や開発の効率化のための重要な活動である．文部科学省の「e-Society 基盤ソフトウェアの総合開発」の委託に基づいて行われている EASE(Empirical Approach to Software Engineering)プロジェクトでは，現場の開発者やプロジェクト管理者の負担を最小限にして，プロセス改善に有効なデータを収集・分析する手法の研究を行っている．データを半自動的に収集・分析するツールの開発と，そのツールに基づくプロジェクト管理およびプロセス改善のフレームワークの確立を目的として，2005 年度から 2 年間にわたり経済産業省の支援の受けて進められた情報システム開発プロジェクトにおいて，実践的な産学連携の活動を行ってきた．

本レポートは，EASE プロジェクトで開発したツール群を用いたプロジェクト管理およびプロセス改善のフレームワークについて紹介し，それに基づくプロジェクト管理の実践事例，および対象プロジェクトの開発関係者からの評価について述べる．

[i] 奈良先端科学技術大学院大学　情報科学研究科
[ii] 株式会社 NTT データ
[iii] 大阪大学　大学院基礎工学研究科

1　はじめに

1.1　背景（2004，2005 年度の活動）

奈良先端科学技術大学院大学および大阪大学が進めている EASE(Empirical Approach to Software Engineering)プロジェクト[1]では，ソフトウェア開発プロジェクトをリアルタイムにモニタすることで，「プロジェクト遅延」や「障害対応遅延」に繋がる問題発生を早期に検出し，問題への対処を行うことで，納期遅延，コスト超過，品質低下を防止する研究を進めている．

2004 年度，EASE プロジェクトでは GQM（Goal/Question/Metric）パラダイム[2]を用いて，「プロジェクトの遅延の主な要因」を検出するモデルを作成した．GQM パラダイムとは，分析目的を明確にした上でトップダウンに計測を行うフレームワークで，メリーランド大学の Basili 教授らによって提唱された． EASE プロジェクトでは，2004 年 11 月 Basili 教授を招いて，構成管理ツールと障害管理ツールからの収集データに基く GQM モデル構築ワークショップを行った．多くの現場関係者にアンケートを行った結果，「プロジェクトの遅延の主な要因」として抽出された「要求の不安定」「設計の不完全」「劣悪な品質」に応じて，これらを目的（Goal） に設定した．Basili 教授と協同で作成したモデルを EASE 独自にカスタマイズし，４つの GQM モデル[4]を作成した．このモデルでは，EASE プロジェクトで開発した EPM(Empirical Project Monitor)というツール[1]の導入を前提とし，EPM 収集データから計測されるメトリクスを用いている．

2005 年度，著者らは上記の 4 つのモデルのうち 2 つを経済産業省の支援を受けて進められているプローブ情報システム開発プロジェクトに適用し，各メトリクスの有用性を確認した．

1.2　適用フレームワーク(先行研究)

GQM モデルは，計測のためのフレームワークであるが，さらに計測結果を Goal を達成するための活動，すなわちプロセス改善活動にフィードバックするための QIP（Quality Improvement Process）フレームワークを Basili 教授らは提唱している[3]．

このフレームワークの構造を図 1 に示すが，図中の活動概要は以下のとおりである．

- C1:プロジェクトの環境の現状把握
- C2：改善のためのベースを提供するための計画
  - C2.1：GQM パラダイムの Goal の設定
  - C2.2：C2.1 の結果を出すための手法計画

[1] http://www.empirical.jp/top.html

- ✧ C2.2.1：モデル，メソッド，ツールなどの選択
- ✧ C2.2.2：Question，Metrics の設定

- C3：C2 の計画の実行
- C4/C5：C1-C3 の経験の蓄積と実施プロセスの普及改善

2005 年度の GQM モデルの構築は，上記の C2.1，C2.2.2，C3 に当てはまる．C3 に関しては，実際に計測を行ったが，計測結果はプロジェクトの中間と完了時に試験的に開発者（プロジェクトマネージャ，プロジェクトリーダなど）にフィードバックしただけで，フィードバックやプロセス改善に関しては計画的に実施しなかった．

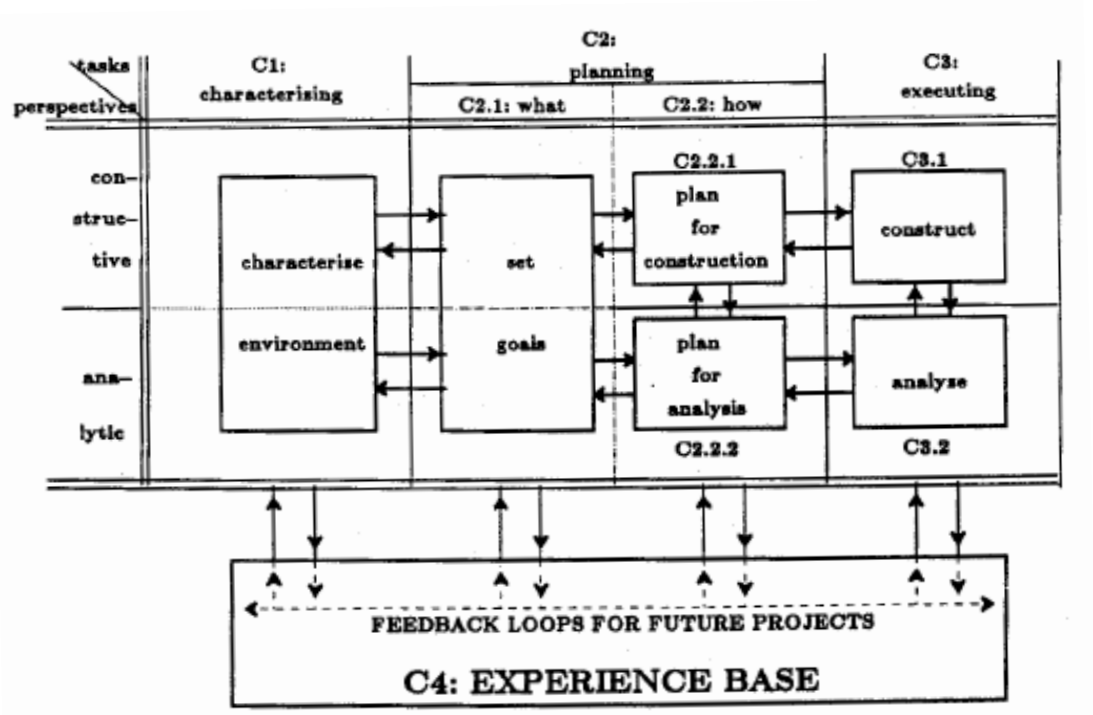

Fig. 1. The improvement-oriented TAME software process model.

図 1　プロセス改善志向のTAMEソフトウェアプロセスモデル[3]

1.3　本研究のアプローチ

2006 年度は，2005 年度構築した GQM モデルを含めたプロセス改善を目的とした QIP (Quality Improvement Process) フレームワーク[3]を用いて，2005 年度適用対象プロジェクトのフェーズ 2 開発でのプロセス改善計画を立案・実行する．

この研究によって，著者らは QIP フレームワークを実プロジェクトに適用するに当たって，考慮しなければいけない観点を明確にする．さらに，『C4/C5：

C1-C3 の経験の蓄積と実施プロセスの普及改善』に向けて，実行結果の評価方法等についても，実験的な作業とその結果について報告する．

まず,『C1：プロジェクトの環境の現状把握』としては，2005 年度に行われた対象プロジェクトの第 1 フェーズ開発の状況について，各社に対して行ったアンケート結果を用いる．アンケートでは，2005 年度対象プロジェクトで発生した問題点を，様々な立場の開発関係者から具体的に挙げてもらった．アンケートの結果から,2006 年度対象プロジェクトで問題となりそうな観点で Goal を再設定し，2005 年度構築した GQM モデルを再構築する（『C2.1：GQM パラダイムの Goal の設定』, および『C2.2.2：Question, Metrics の設定』）．さらに,『C2.2.1：モデル，メソッド，ツールなどの選択』として，データの収集方法や収集データの分析間隔，分析するシステムの分割単位，レポート形式，フィードバック方法などを詳細に計画し，必要なツールや作業体制を整えた．

1.4　対象プロジェクト

本論文で適用対象としたプロジェクトは，今日の日本における情報システム開発の典型とも言える「マルチベンダ開発」である．具体的には，経済産業省の支援を受けて進められた中規模の情報システム開発プロジェクト「プローブ情報システム開発プロジェクト」を対象とした．本プロジェクトは，2 年にわたって新規開発(一年目)，処理の共通化と機能追加（二年目）が行われたが，2006 年度はその二年目にあたる．

このプロジェクトでは，EASE プロジェクトとソフトウェアエンジニアリング技術研究組合(以降，COSE と呼ぶ)[2]，情報処理推進機構（IPA）下に設置されたソフトウェアエンジニアリングセンター（以降，SEC と呼ぶ）[3]の 3 者が連携して，データ収集体制の構築とデータ分析を行った．COSE はユーザ的立場の企業とプロジェクト管理担当企業，開発担当５社から成り，ユーザ的立場の企業が要求を出し，開発担当５社が分担してサブシステムの開発を行い，プロジェクト管理担当会社が 5 社を統括した全体のプロジェクト管理を行う．開発はウォーターフォールプロセスにより進められ，具体的には，要件定義を受けて基本設計を全社の協働で行い，以降各社で詳細設計，プログラム設計，製造，単体試験，社内結合試験が行われた．その後，各社のプログラムを集め，組合全体として社間結合試験，総合試験が行われた．各社で分担して作成した基本設計書については，全社の担当者によるレビューが行われたが，以降社内結合試験までは週次の進捗報告(進捗率)と各工程完了時の品質評価報告のみがプロジェ

---

[2] ソフトウェアエンジニアリング技術研究組合（COSE）：COnsortium for Software Engineering

[3] http://sec.ipa.go.jp/index.php

クト管理担当企業に提示される．このような限定されたプロジェクト管理方法となった要因の１つには,開発対象が協調（公開）領域と競争（非公開）領域が混在したものになっており，社間のみならずプロジェクト管理担当会社に対してもプロダクトや社内情報に関する機密保持が求められたからである．

本研究で用いた分析手法および適用フレームワークは，対象プロジェクトのような研究組合型マルチベンダ開発の特徴である限定された情報によるプロジェクト管理に対して，有効な支援を行う方法として導入された．

2　実行計画

2.1　プロジェクトの環境の現状把握

著者らは，2005 年度中のプロジェクトの定期的なミーティングや開発関係者との GQM モデルによる計測結果フィードバックミーティングを通して，プロジェクトにおけるいくつかの問題点を認識していたが，2005 年度のプロジェクト終了時に以下のようなアンケートを行って，重大な問題を絞り込んだ．

以下にアンケート内容を示す．回答は,「社内」,「社間」,「その他」の 3 つの観点に分類された選択肢からの重複選択によって得た．回答者は 26 名で，プロジェクトマネージャ，プロジェクトリーダ，品質保証担当者など，様々な立場の開発関係者から回答を得ている．表 1 は，回答の多い順に並べた発生問題のリストである．

- ［質問］当プロジェクトでご自身がどのような問題に遭遇されましたか？
  - ✧ 社内において
    - a.進捗が遅れた
    - b.コストが超過した
    - c.品質が悪かった
    - d.業務知識の獲得に苦労した
    - e.開発手法やツール等の変更があり，なれるのに苦労した
    - f.人員の変更が発生した　　g.残業時間が異常に多かった
    - h.開発環境の整備が遅れた
    - i.要件定義（基本設計）の決定が遅れた（変更された）
    - j.社内におけるプロジェクトへの理解を得るのが困難だった（要員確保・予算確保・ソフトウェアの調達[流用]において）
  - ✧ 各社間において
    - k.要件定義（基本設計）の決定に時間がかかった
    - l.社間の調整作業などに時間がかかった
    - m.コミュニケーション方法の確立に時間がかかった

n.用語や標準の違いで誤解が発生した
o.相手側の体制や窓口（担当者）が不明確だったり変更されたりして調整が滞った
p.他社の進捗情報・品質情報などに不安があった（気になった）
q.開発環境の整備が遅れた

✧ 個人的・対 SEC/EASE・その他

r.会議やミーティングの日程調整などがスムーズに行われなかった
s.プロジェクトや会議の進行方法が異なるため戸惑った
t.本プロジェクトにおける業務(作業)内容がすべて把握できていなかったため，後から発生する様々な作業に振り回された
z.その他

表 1　2005 年度発生問題に関するアンケートの回答

回答数	問題内容
11	進捗が遅れた
7	社間の調整作業などに時間がかかった
5	業務知識の獲得に苦労した
4	開発手法やツール等の変更があり，なれるのに苦労した
3	人員の変更が発生した
3	開発環境の整備が遅れた
3	その他
2	品質が悪かった
1	要件定義（基本設計）の決定が遅れた（変更された）
1	社内におけるプロジェクトへの理解を得るのが困難だった（要員確保・予算確保・ソフトウェアの調達[流用]において）
1	用語や標準の違いで誤解が発生した
1	相手側の体制や窓口（担当者）が不明確だったり変更されたりして調整が滞った
1	他社の進捗情報・品質情報などに不安があった（気になった）
1	プロジェクトや会議の進行方法が異なるため戸惑った
1	本プロジェクトにおける業務(作業)内容がすべて把握できていなかったため，後から発生する様々な作業に振り回された

2.2　GQM パラダイムの Goal の設定

表 1 のリストから，プロセス改善の対象として，2006 年度特に発生する可能

性があり，重大なプロジェクト遅延の原因になりうる具体的な問題を２つ選択し，Goal を設定する（赤字の項目）．リストの中には，対象プロジェクトの初年度であるための発生した問題（例えば，「業務知識の獲得に苦労した」や「開発環境の整備が遅れた」など）もあり，2 年目に発生する可能性が低い項目は除いた．また，2005 年度の開発に対し，2006 年度は機能の追加，モジュールの共通化などの改造開発がメインであることを鑑みて，特に改造時に発生する問題から２つの Goal を追加した．

- 要員に基づくリスクを管理する
- 組織内外のコミュニケーションを円滑に行う
- 既存コードの効率的・効果的活用を行う
- 変更（要求，設計，プログラム）の安全性を確保する
- （高品質なソフトを効率的に開発する）

最後の「高品質なソフトを効率的に開発する」は，非常に抽象的な目的であるため，GQM モデルとして具体化することができなかった．しかし，実際には本活動中に，他の 4Goal に結び付けられない問題が付随的に検出されるケースが 2005 年度も多く見られたため，Goal の１つとして挙げておく．

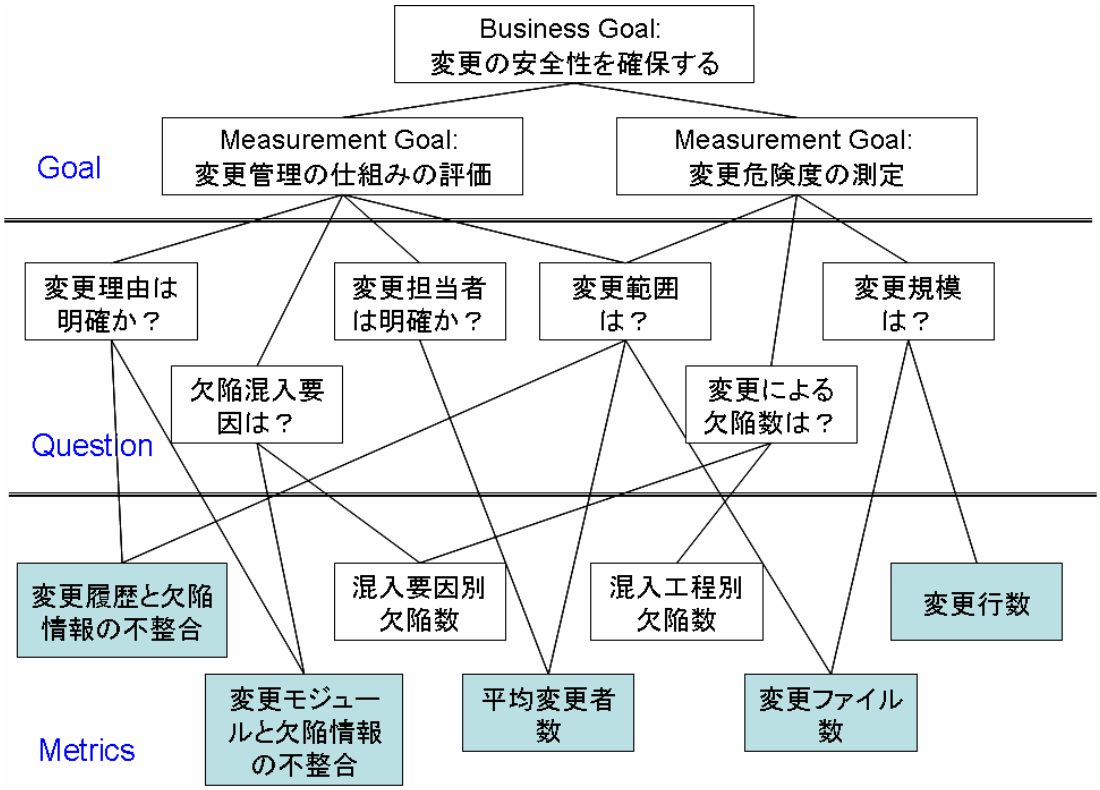

図２　新 GQM モデル例

2.3　Question，Metrics の設定

上記の Goal について，2005 年度の実績のある Question/Metrics から適切なものを選択・修正し，新たに GQM モデルを作成した．図 2 に，「変更（要求，設計，プログラム）の安全性を確保する」を Goal とした GQM モデルを示す．

2.4　モデル，メソッド，ツールなどの選択

図 1 の C2.2.1 すなわち「モデル，メソッド，ツールなどの選択」を計画し，C3 すなわち実運用を行った．運用のサイクルは，図 3 のとおりである．

それぞれの作業手順については，以下の方法で行った．

- ✧ データ提出：WebDAV[4]を用いて，ネットワーク経由での提出を可能にした．2005 年度は郵送などによりデータを提出していたため，媒体の読込みエラーなどが発生した場合に，再提出までに多くの時間がかかっていた．WebDAV によって，データ提出における機密性が向上し，コストが低減し，提出遅れのフォローも適宜行えるようになり，定期的にデータを提出することを定着できる．また，WebDAV から取り出したデータは暗号化された後，入場者が管理されていて機構内外のネットワークから切り離された機密室に移され，専用のコンピュータに格納され，分析される．データの提出から分析まで，機密性を保持した運用を行う．
- ✧ データ分析
  - ➢ 分析時期：2.3 の図 2 に示した GQM モデルで，「変更履歴と欠陥情報の不整合」など青く塗りつぶしたメトリクスは，できるだけリアルタイムに計測し問題の指摘を行うことが効果的と思われるものある．理由としては，　それぞれが 1 つ 1 つの変更理由に依存するもので，開発者本人しか内容を理解していないため，時間がたつと忘れてしまう可能性が高いからである．一方，塗りつぶしていない「混入要因別欠陥数」などは，一定量のデータ集合を分析することにより傾向を把握するもので，工程区切りなどで分析するべきメトリクスである．
  - ➢ 分析方法：データ分析は，まず，各メトリクスを計測し，可視化したレポートを作成する．レポートの作成は，メトリクスの性質ごとに異なる．週次で分析を行うものに関しては，今回は分析単位を社別からコンポーネント/モジュール別と粒度を小さくしたことから，分析用のツール（ProStar：MS-Excel，Access の VBA･マクロによるツール群）[5]を開発し，専門的な知識のない作業者がある程度機械的に分析結

---

[4] WWW でファイルの転送に使われる HTTP を拡張子，クライアント（Web ブラウザ）から Web サーバ上のファイルやフォルダを管理できるようにした仕様．
http://e-words.jp/w/WebDAV.html

果をレポートにすることを可能にした．図 2 の青く塗りつぶしたメトリクスについては，これを用いて分析する．分析の手順は，(1) EPM による必要なデータ収集（Linux 上），(2) Windows マシンに収集データ（XML 形式）を移動，(3) ProStar によるグラフ化→グラフのレポート化（パワーポイントファイルに整理して貼り付け），となる．一方，工程区切りなどでデータ分析を行うメトリクスについては，一般的な表計算ソフトウェアやデータベースソフトウェア，統計解析ツールを用いて行う．

- ✧ 各社へのフィードバック：分析結果のグラフを元に，EASE 研究員による考察を行い，2.2 の Goal に関連すると思われる現象をチェックする．その結果抽出された現象について，各社に背景の確認や調査を依頼し回答を得る．このやり取りは，週次分析についてはリアルタイム性を重視するため，すべて社別のメーリングリスト（EASE 研究員＋SEC 研究員＋COSE プロジェクトマネージャ＋各社担当者を含む）を通して行われた．レポートはすべて社別のパスワードで暗号化し，誤送やネットワーク上のセキュリティにも配慮する．工程区切りでの分析については，社別の会議形式で，分析結果の説明・インタビューを行うとともに，同じく確認や調査依頼・回答はメーリングリストを通して行った．会議形式の方が，回答率がよく，また深い背景の聞き取りなどができるという長所があるが，時間的制約がありコストがかかるという問題もある．
- ✧ 各社からの回答・対策：今回は，問題指摘によるプロセス改善や問題への対処は，各社内でそれぞれ取り組むこととし，プロジェクトマネージャや分析者から特に問題への対処を指示した依頼したりしない．本来は，対処の依頼，さらにその結果の確認（問題状況の解消）をすることによって，この運用サイクルは完成するが，短期開発でありプロジェクトマネージャと各社間の機密保持上の理由から，指摘された問題への各社内の対応については，開発完了時のインタビューで聴取した．ただし，データの入力ミス，データ収集ツール（障害管理ツール：GNATS[5]や構成管理ツール：CVS[6]など）の運用に関する指摘など分析対象データの精度にかかわる問題については，即時の対応を依頼した．

---

[5] GNU　Bug Tracking System　http://www.gnu.org/software/gnats/
[6] Concurrent Versions System　http://www.nongnu.org/cvs/

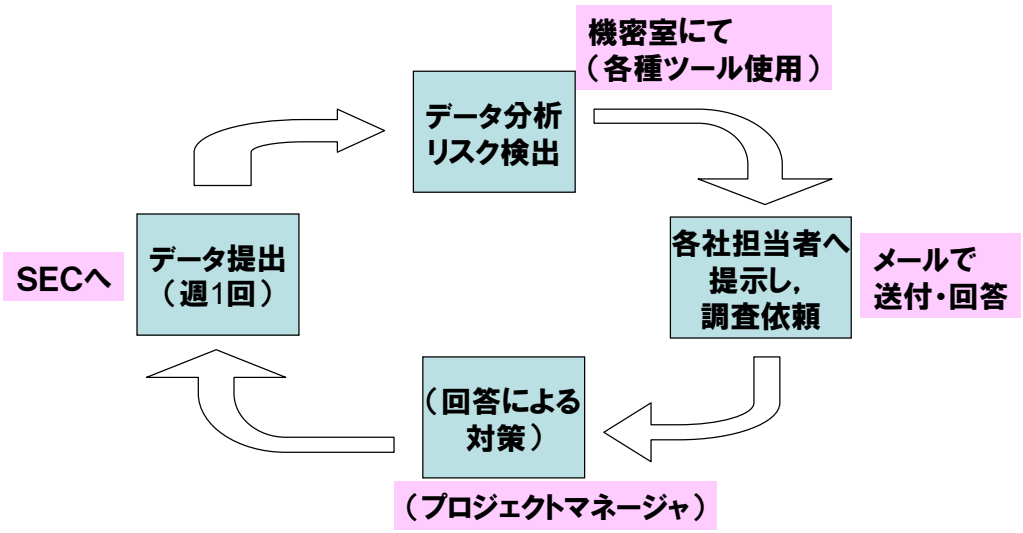

図３　QIP モデルの運用のサイクル

2.5　経験の蓄積と実施プロセスの普及改善

本研究では，特に 1.2 の『C4/C5：C1-C3 の経験の蓄積と実施プロセスの普及改善』に関する項目については，事後評価を行うこととし，開発途中での経験の活用については対象としなかった．本来，この部分は複数の開発プロジェクト間での経験の共有や次のプロジェクトへの適用時の改善を意図しており，本研究の対象プロジェクトは単一プロジェクトであることから，この項目については有益な成果が得られないと判断した．しかし，適用事例の評価のため，本手法の適用後，各社に対してインタビュー・アンケートを実施し，指摘事項の有用性・社内プロセスへの改善効果・適用した手法の有用性や運用方法への改善提案などを調査した（4.2 参照）．この調査を通して得られた情報の蓄積が，本手法の普及・改善に繋がる，と考えているが，実際にプロセスに計画的に埋め込むには，さらに検討が必要と考えている．

3　　実行事例

3.1　分析対象データの収集

5 社 6 拠点（１社のみ，二拠点［２つの異なる事業部］で開発）から毎週提出される成果物の構成管理履歴（CVS 履歴）データおよび障害管理データ（GNATS 等）に対して分析を行った．ただし，各拠点内がさらに物理的・体制的に分割されている場合があり，１拠点から複数の CVS や GNATS のリポジトリが提出されるケースもある．

2005 年度は，主に拠点別の分析を行い，一部の特徴的なコンポーネントに対しては，対象コンポーネントのデータのみを抽出して分析を行った．2006 年度は，改造開発と新規開発が混在し，モジュールの共通化なども行われたため，

拠点単位の分析では状況把握が困難であるため，開発管理単位（コンポーネント，もしくはモジュール）で分析を行った．ただし，障害進捗に関しては，1 つ 1 つの障害をコンポーネント別に分類するのが困難であることと，開発管理体制や要員体制に関わる問題の検出が目的であることから，拠点別の分析が妥当と判断し，拠点別にグラフ化・分析を行った．

3.2　データ分析結果例

以下，分析結果の一部を添付する．

図 4 は障害対応の進捗を示している．総発生障害数だけでなく，重要度・優先度の高い障害数，未解決の障害数，未解決の障害の平均滞留時間を可視化している．このグラフから，深刻な障害発生による進捗遅延やさらに詳細な調査から障害対応に関するリソース不足などを検出することができる．図 5，図 6 は拠点ごとのコンポーネント/モジュール別の規模推移と更新回数（週別の集計数）を示している．図 5 からは，各コンポーネント/モジュールの進捗状況を把握することができる．一方で，全体の規模変化が収束しているように見える場合にも，頻繁に変更が行われていることがある．特に，仕様変更や設計変更などにより，小さな更新が頻発し，全体規模の変化が認められなくても開発が完了状態にない，といった問題を検出するために，更新回数をチェックする(図 6 参照)．このプロジェクトでは，データ提出が週次だったので週ごとの更新回数を示しているが，可能ならば毎日計測することも有益であると考えられる．

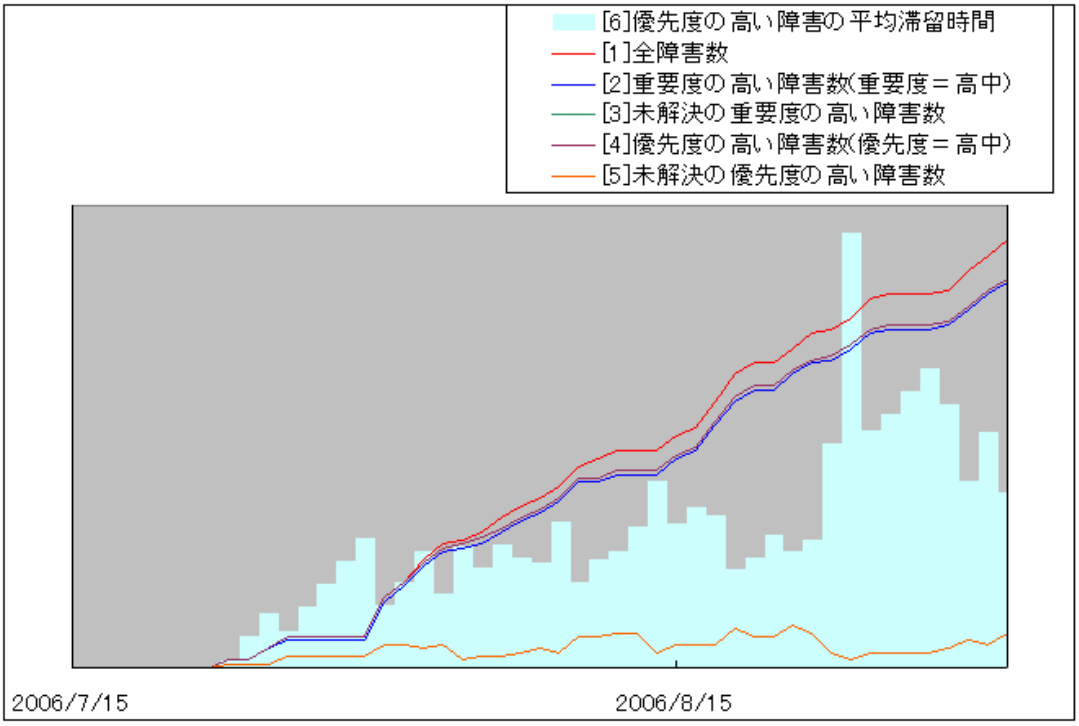

図 4　障害対応進捗グラフ(社別)

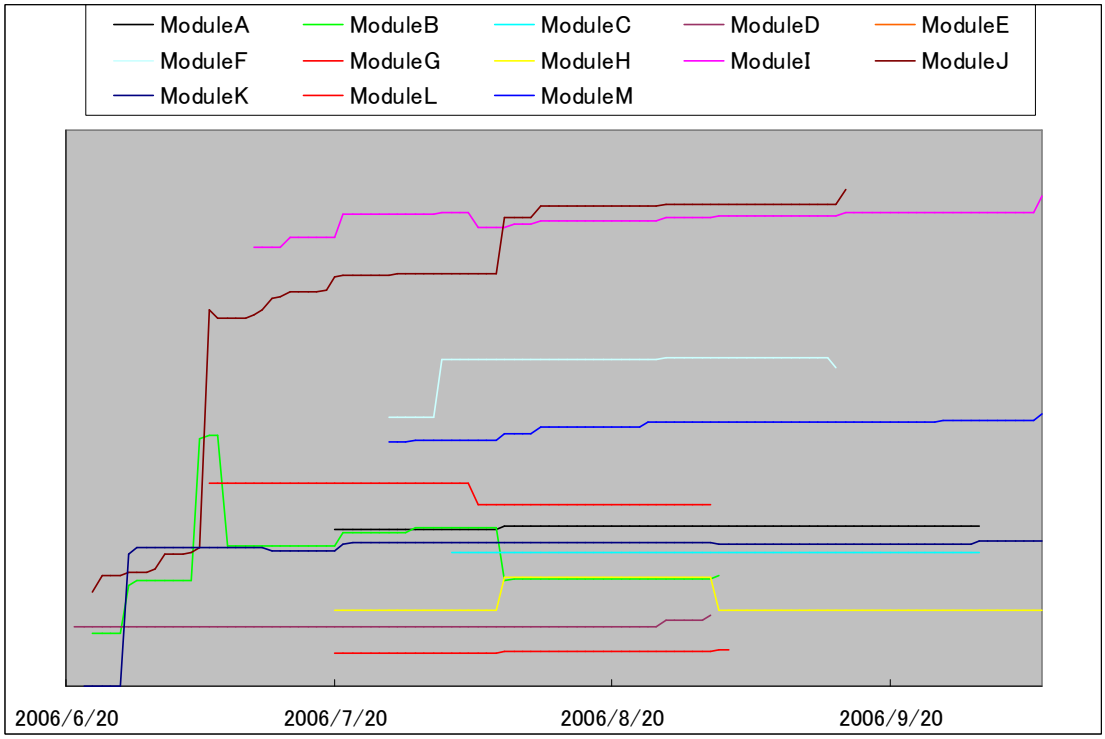

図５　コンポーネント/モジュール別規模遷移グラフ

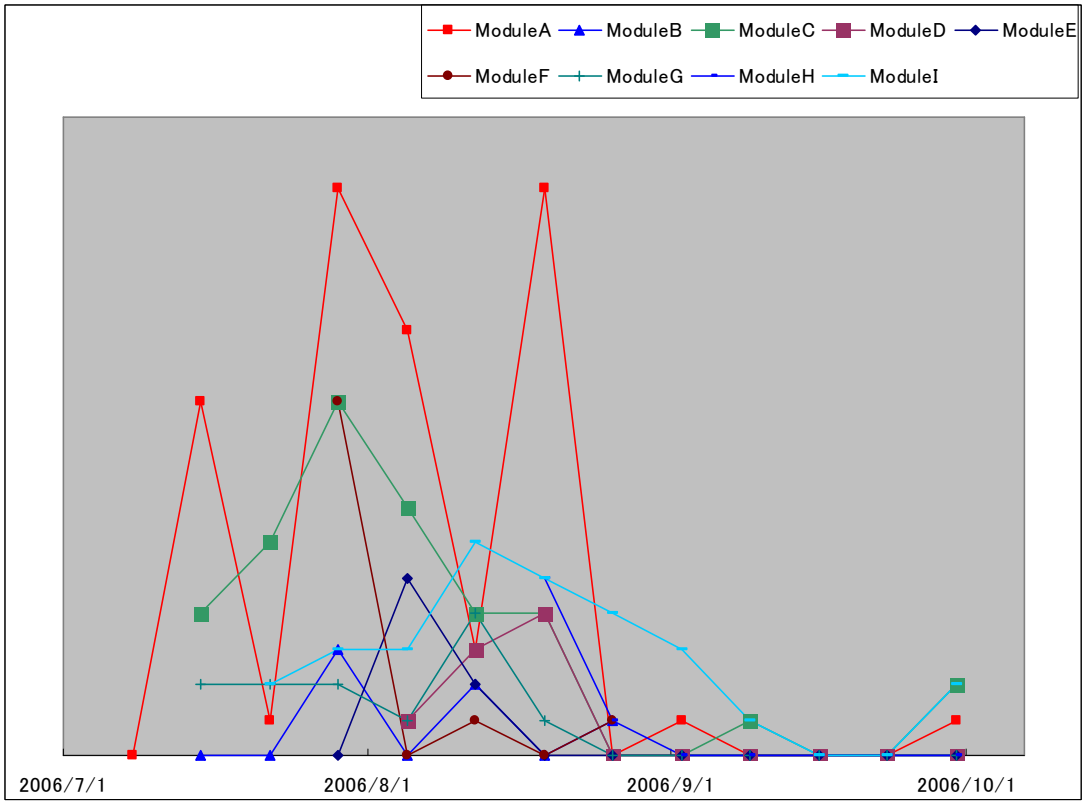

図６　コンポーネント/モジュール別更新回数グラフ

図 7～10 は単一コンポーネント/モジュールのメトリクスの計測結果例である．図 7 のファイル数，KSLOC は図 5 と同じく進捗の把握に用いることができる．図 8 は，今回作成した計測モデル（2.3 参照）に対応するメトリクス群の時間推移に伴う変化を示している．これらは，複数のメトリクスを用いてより正確に状況を把握するのに用いられる．例えば，追加行数と削除行数がともに大きく増加するときは大規模な変更が行われ，追加行数のみが増加するときは新規開発が行われている，などである．図 9 図 10 は，週別の集計した変化量として，変更頻度と変更規模を示している．変更頻度とは，全体の更新回数と共に削除量の大きい更新回数とその全更新回数に対する比率を示し，変更規模は追加行数，削除行数と削除行数の追加行数に対する割合を示している．削除量の大きい更新の比率が高い場合や削除行数の割合が高い場合，既存コードへの変更が頻繁だったり大量であることを示し，変更の危険性や既存コードの修正の複雑さなどを推測することができる．

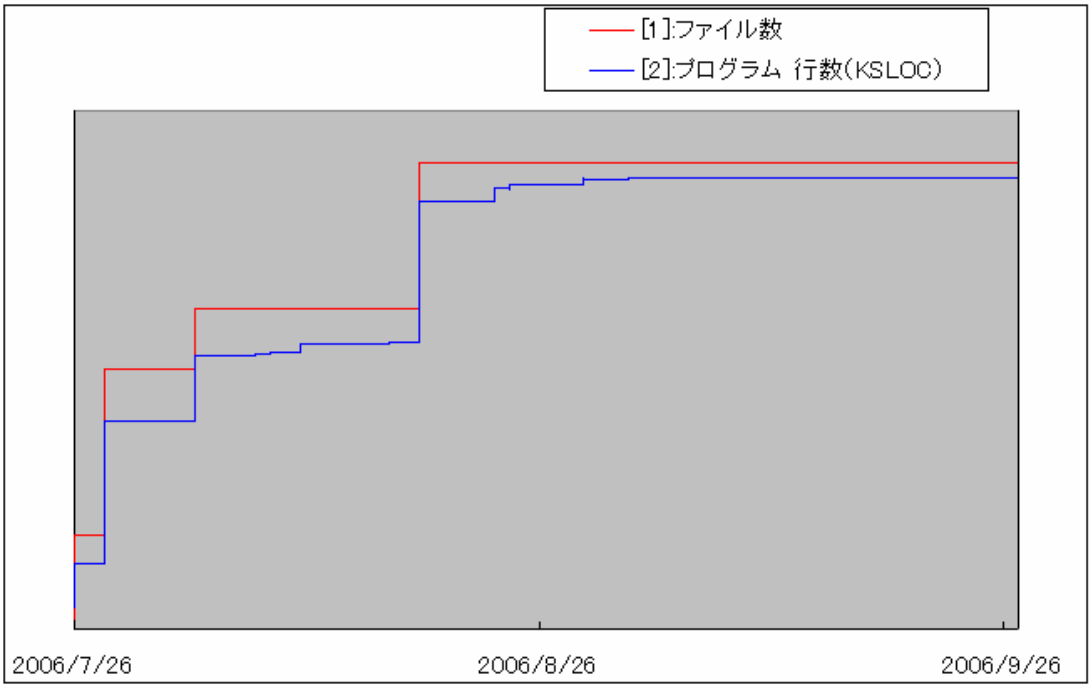

図 7　単一コンポーネント/モジュール規模遷移グラフ

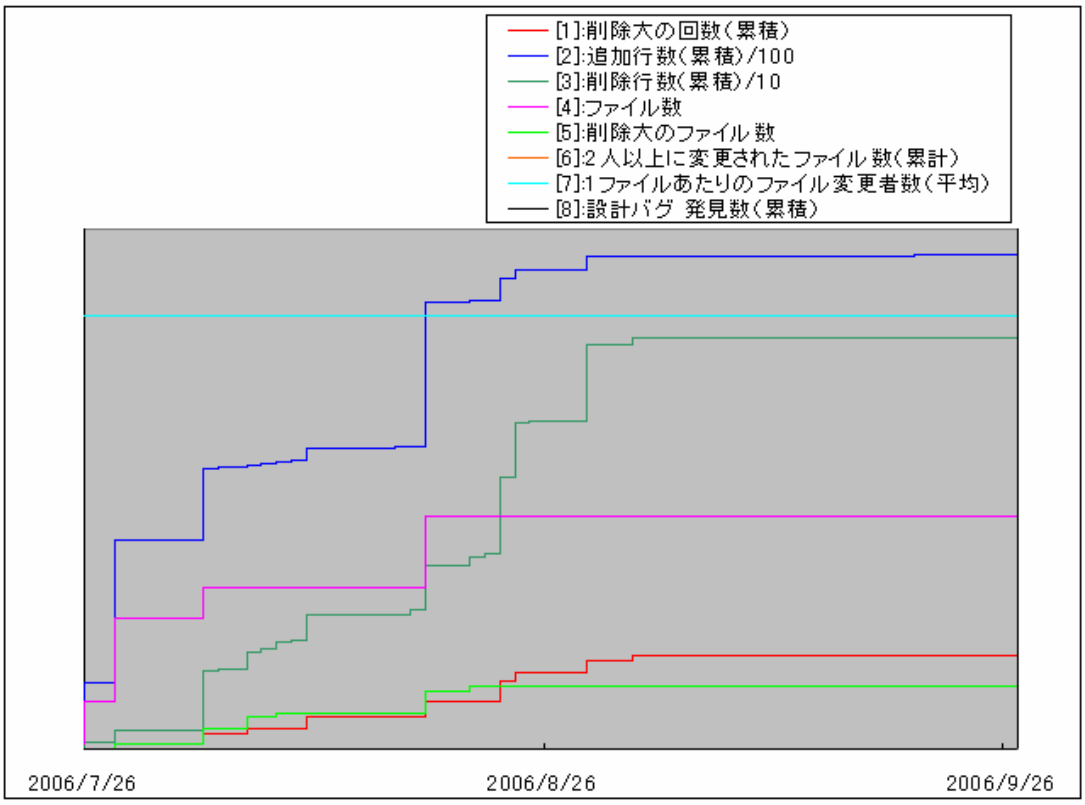

図８　単一コンポーネント/モジュール各メトリクス累積グラフ

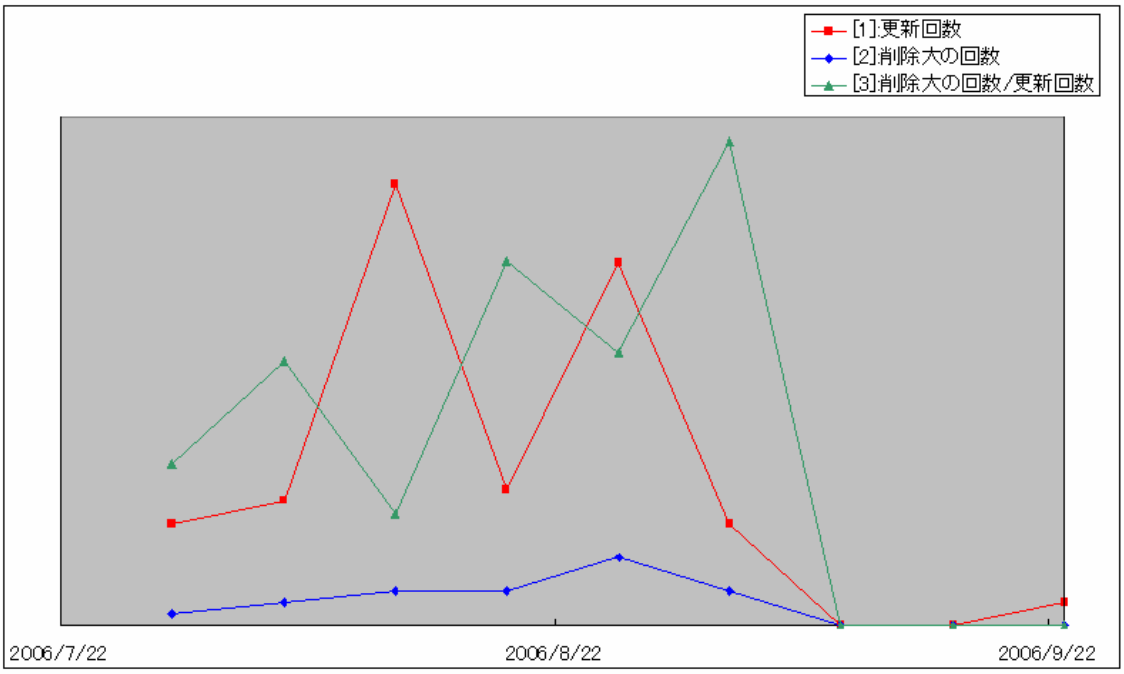

図９　単一コンポーネント/モジュール変更頻度グラフ

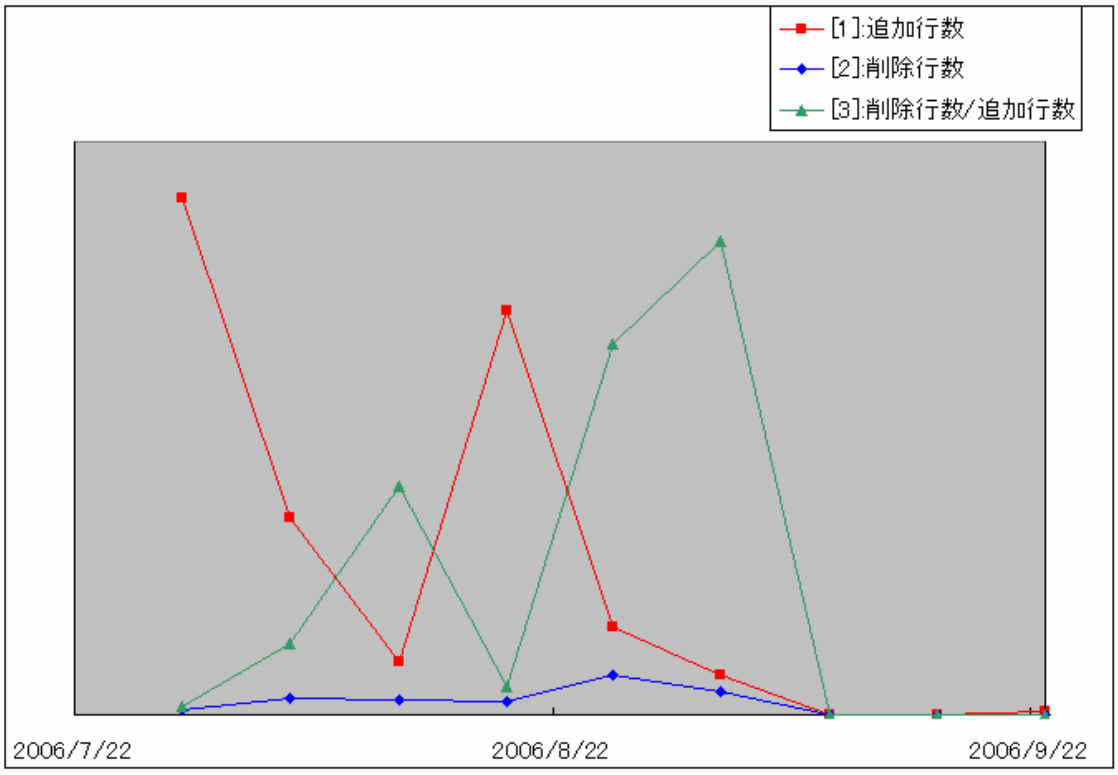

図１０　単一コンポーネント/モジュール変更規模グラフ

図 11 は，ある拠点での社内結合試験までに発生した障害の故障発生原因の種類別の障害件数の棒グラフで，「問題を発見しなければならなかった設計もしくはテスト工程で摘出されなかった要因」（以降，混入要因とする）別の割合を示している．この図から，既存コードの流用（『再利用時のチェック漏れ』）や変更･バグ修正時のデグレード（『修正時のチェック漏れ』）の危険性や難易度を検出することができる．また，その「混入要因」を精査することにより，プロセス改善の指針（レビューの強化など）を導出することができる．

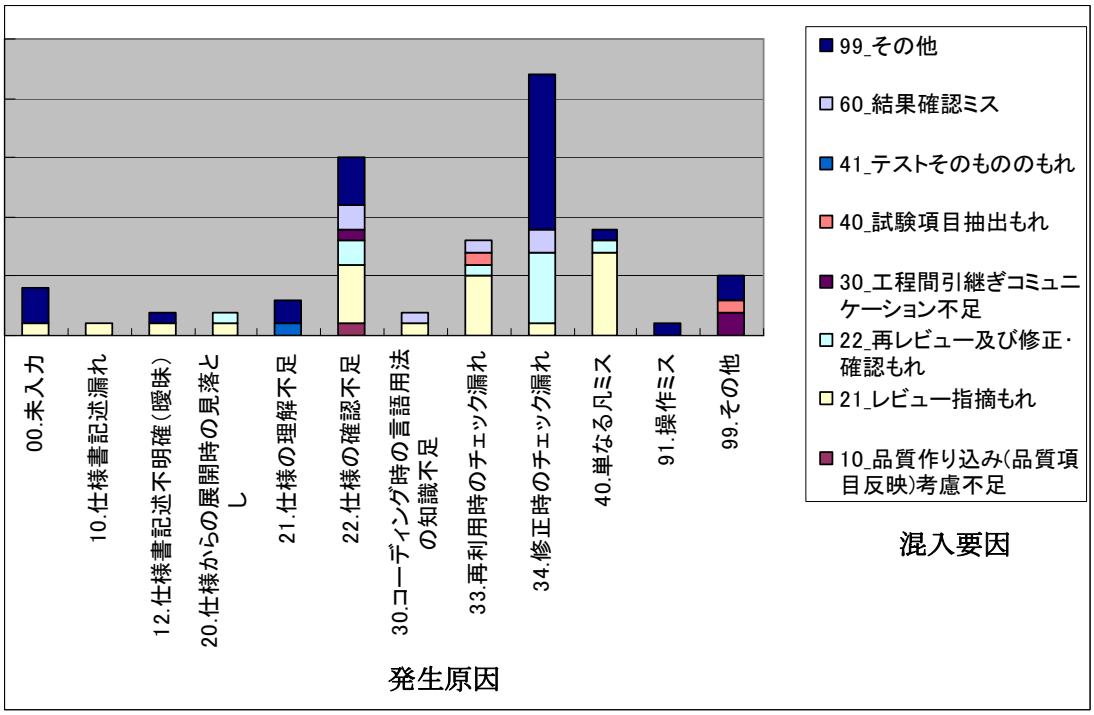

図１１　故障発生原因－混入要因による ODC 分析

3.3　フィードバックと回答

3.2 のデータ分析結果について，プロジェクトの進行状況を把握している EASE 研究員から，各社に対して以下のような質問を分析レポートと共に提示・送付し，回答を求めた．以下は，質問と回答の例である．

例1）　［質問］7/15～7/22 の週で，更新回数，変更規模が落ち込んでいますが，この週の作業に何か問題（出戻り等)が発生していなかったでしょうか？

［回答］7/8 の週までに初期投入した流用元ソースの中で，間違って投入したものがあり，7/8～7/15 の間にそのソースを削除して，正しいソースを投入しなおしたため，変更規模が大きくなっております．7/15～7/22 は通常の開発です．7/22～7/29 の週は，未投入だった流用元ソースをすべて投入したため，変更規模が大きくなっております．

例2）　［質問］xxx コンポーネントで，８/９あたりに大規模な変更が行われています．GNATS の 20，22，23 あたりですが，GNATS 上の工数は特に大きくありません．変更量に対して，修正工数が低い理由が何か考えられますでしょうか？

［回答］上記修正番号で修正が大きかったのは 23 です．C++の string クラスに関して使い方が間違っていたことが分かり使用している部分

を全て見直しため，変更量が大きくなりました．修正工数が少ないのは，関連見直しのため複雑な修正ではなかったためです．

例3） ［質問］8/30 あたりから，CVS 上新しいアカウントのメンバーによる変更が入っています． 開発メンバーが交代，もしくは新規参加されたと考えてよろしいでしょうか？もしくは，何らかの理由で一時的にファイルの更新を一括で担当されただけでしょうか？
［回答］一時的にメンバーが追加となりました．

4 実行事例の評価

4.1 フィードバック・回答の分類

3.3 のような質問（指摘）と回答の結果は，以下の観点で分類した．

- 異常指摘：プロジェクト管理の観点から，問題のありそうな状況の指摘
  例）異常な欠陥密度，開発完了(進捗 100%)でのコード更新など
- 入力データ品質確認：入力データの不整合・ミスの指摘
  例）単体試験工程で発見された障害の混入工程が結合試験など
- データ収集プロセス確認：ツール運用上のミス・データ提出漏れ
  例）障害の登録漏れ(遅延)，完了(更新)漏れなど

さらに，指摘時期は 3 種類に分類できる．詳細設計完了時に会議形式で行った「設計工程品質評価会議」，製造以降社間総合試験まで週次でメーリングリスト経由で行った「週次フィードバック」，そして社内結合試験完了時に会議形式で行った「社内試験工程品質評価会議」である．

質問（指摘）内容と指摘時期によって分類した件数を表 2 に示す．

表 2 質問（指摘）－回答内容の分類表

	異常指摘	入力データ品質確認	データ収集プロセス確認	未回答/不明	合計
設計工程品質評価会議	34	7	3	1	45
週次フィードバック	35	11	3	10	59
社内試験工程品質評価会議	14	11	3	4	32
合計	83	29	9	15	136

4.2　フィードバック・回答についてのインタビュー

4.1 の指摘のうち，異常指摘の 28 件について，各社にインタビューを行った内容とその結果について表 3 にまとめる. インタビュー時間に限りがあるため，各社 4〜9 件を抽出した（C 社と D 社は一括で行われた）. また, A 社の 1 件は，実質 4 件の類似指摘を含む．従って，インタビューを行った指摘の割合は，80 件中 28 件で 35%であった.

表 3　各指摘事項に関するインタビュー内容とその結果

		対象社	A	B	C	D	E	F	合計
		アンケート対象指摘件数	6	9	1	3	4	5	28
1.　事実の認識		A)　指摘され，内部調査してわかった	1		1				2
		B)　指摘され，状況がすぐに思いついた	2			2		1	5
		C)　指摘される前から状況を把握していた	3	9		1	4	4	21
	→「指摘される前から把握していた」場合	D)　障害票や構成管理などのデータチェック・分析から知った	2				3	2	7
		E)　日次・週次などの定期的な進捗会議における担当者からの報告で知った	1	9		1	1	2	14
		F)　現場の状況を観察している中で知った							0
		G)　その他							0
	→「障害票や構成管理などから知った」場合	H)　GNATS 項目や CVS の運用ルールなど，COSE 方式（もともと自社内のシステムにはなく，昨年度から新たに導入した方式）によって知ることができた〔GNATS 項目記入時や品質評価会議向けの GNATS 項目の自社内分析など〕	1				1		2
		I)　もともとある自社内の管理方法（障害項目・品質管理等）でも知ることができた					2	2	4
		J)　COSE 方式では知ることができず，自社内のシステムもしくはチーム内の別の管理方法で知ることができた							0
		K)　その他	1						1
2.　問題（＝何らかの対策が必要な状況）としての認識		A)　当該事実を指摘され，初めて問題と認識した	4		1	1		1	7
		B)　当該事実を指摘されたが，問題と認識していない（いなかった）				1			1

		C) 指摘される前から知っていて，問題と認識していた	2	1		1	2	4	10
		D) 指摘される前から知っていたが，問題と認識していなかった		8			2		10
3. 問題の波及範囲		A) すべてのプロジェクトで，頻繁に発生する可能性がある問題である．	3		1	1	2		7
		B) 一般的なプロジェクトでも，時々発生する可能性がある問題である．	2	3		1	2	2	10
		C) 当プロジェクトに特有なものであるが，一般的なプロジェクトにも〔このような条件・状況下で〕当てはまる場合がある		4		1		3	8
		D) 当プロジェクトに特有なもので，一般的なプロジェクトでは発生しない問題である．		2					2
		E) **回答不可**	1						1
4. 問題への対処		A) 〔このような〕プロセス改善を実行した	2	1				1	4
		B) とりあえず，〔このような〕対症療法を行った	2			2	3	2	9
		C) 問題と認識していたが，何もしなかった	2		1				3
		D) 問題でないと認識していたので，何もしなかった		8		1		2	11
	→「対症療法やプロセス改善を行った」場合，効果は？	E) 同様の問題の再発が見られなかった（問題発生を予防した）	2			2	2	3	9
		F) 同様の問題を検出したが，すぐに対処することで，問題の拡大を防止した							0
		G) 同様の問題が再発した（効果がなかった）							0
		H) 同様の問題の再発が見られなかった（そもそも同じような状況が再現しなかった）					1		1
		I) その他	2	1					3
	→「問題と認識	J) 時間的人的余裕がなかった	1		1				2

	していたが, 何もしなかった」場合, なぜか？	K) 改善・対処の手段を思いつかなかった							0
		L) 再発の可能性が低いと思った	1						1
		M) その他							0
5. 今後への改善対策	5.1. 問題の検出方法について	A) 当該問題を検出する方法は，有効である	2	8		2	2	3	17
		B) 当該問題を検出する方法は，有効であるが，導入は困難である	4		1				5
		C) 当該問題を検出する方法は，あまり効果的とは思わない				1			1
		D) 当該問題を検出する方法として，〔このような〕方法がより有効である							0
		E) 当該問題を検出する必要はない						1	1
	5.2. 問題の指摘時期について	A) リアルタイム（日次・週次）の指摘が効果的である	5	8	1	2	2	3	21
		B) 工程区切りでの指摘が効果的である	1		1				2
		C) プロジェクト後の事後検証時の指摘が効果的である							0
		D) 個々の事例の指摘は不要だが，プロジェクト横断的な分析時に有効である							0
	5.3. 問題への対処方法について	A) 〔このような〕プロセス改善・対処方法によって，〔このような〕効果があった						3	3
		B) 〔このような〕プロセス改善・対処方法によって，〔このような〕効果が期待される	6		1		1		8
		C) プロセス改善や対処を行っても，あまり効果が期待できない（投資対効果が薄い）							0
		D) プロセス改善や対処を行う必要はない							0

4.3　インタビュー結果の考察

上記のアンケート結果から，以下のような考察を得ることができた．

- 指摘内容のほとんどが，「問題」の顕在化に寄与している．
- 問題を事前に把握した手段は，日常的な進捗会議や障害票のチェックによるもので，日常的に現場とコンタクトが取れなかったり，障害データなどをリアルタイムに入手できなかったりする環境（外注・オフショア）などでは，本手法は有効であると考えられる．
- 当手法による指摘は，一般的なプロジェクトでも活用可能である．
- 大方の問題は，既知（過去に経験がある）のもので，検出できれば，対処は可能である．非常にまれなケースやプロジェクトに特有な問題ではない．
- 分析内容のほとんどが，日次（リアルタイム）の運用が有効である，ということから，コストの問題とも絡め，自動化による普及が非常に効果的である，と思われる．

5　まとめ

2005 年度の研究成果に基づき，2006 年度は実際のプロジェクト管理においてプロセス改善を目的として，当手法の適用を実施し，評価を行った．

その結果，当該手法は，EASE プロジェクトで目的とする「プロジェクト遅延リスク」とおよび「障害対応進捗遅延」の検出に有効であることが確認できた．また，ツールや要員・コストにかかわる問題点を対象プロジェクトの開発関係各者（プロジェクトマネージャ，各開発チームリーダ，EASE プロジェクト研究員）において検証し，次のステップとして以下のような点を明確にしていく必要がある，と考えている．

- ツールや環境，体制を整えるのに必要なコスト（時間，人員，機材，技術等）
- 本手法運用のベストプラクティス
- データの蓄積と普及改善プロセスの構築（QIP の C4,C5）

謝辞

本研究の一部は文部科学省「e-Society 基盤ソフトウェアの総合開発」の委託に基づいて行われた．本研究にあたり多くのご協力を頂いたソフトウェアエンジニアリング技術研究組合参加企業，ソフトウェアエンジニアリングセンター，EASE プロジェクトの関係諸氏に感謝します．

引用文献

[1] 大平 雅雄，横森 励士，阪井 誠，岩村 聡，小野 英治，新海 平，横川 智教，“ソフトウェア開発プロジェクトのリアルタイム管理を目的とした支援システム，”電子情報通信学会論文誌 D-I，VolJ88-D-I，No.2，pp228-239，2005.

[2] 井上克郎，松本健一，飯田元著，「ソフトウェアプロセス」pp.112-117 共立出版，2000.

[3] V. Basili and H. D. Rombach，“The TAME Project: Toward Improvement-Oriented Software Environments”，*IEEE Trans. on Software Engineering*，Vol.14，No.6，pp.758-773，1988.

[4] Akito Monden，Tomoko Matsumura，Mike Barker，Koji Torii and Victor Basili，”Customizing GQM Models for the EASE (Empirical Approach to Software Engineering) Project,″ Nara Institute of Science and Technology Technical Report，March 2007.

[5] 玉田 春昭，松村 知子，森崎 修司，松本 健一，”プロジェクト遅延リスク検出を目的とするソフトウェア開発プロセス可視化ツール ：ProStar，“ 奈良先端科学技術大学院大学テクニカルレポート，NAIST-IS-TR2007002，February 2007.