# 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

玉田 春昭　　松村 知子　　森崎 修司　　松本 健一

## 1　開発背景

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[2]では，現場の開発者やプロジェクト管理者に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．

上記の研究の一環として，同プロジェクトではデータを半自動的に収集・分析するツール EPM(Empirical Project Monitor)[1]を開発した．このツールは，構成管理ツール，障害管理ツール，メーリングリスト管理ツールなどから，半自動的にデータを収集し，収集データからプロジェクト管理に役立つ分析を行い，可視化する機能をもつ．

さらに，2004 年度，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づく 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 モデルを作成した[3]．このモデルでは，EPM 収集データから計測されるメトリクスを用いている．

2005～2006 年度は，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 EPM ツールと 4 つの GQM モデルを適用し，その有効性の評価や改善を目的に活動してきた，しかし，分析を進めるに当たり，EPM が持つデータ分析機能以外の分析方法もプロジェクト管理者やプロジェクトリーダから求められるようになった．具体的には障害の修正対応の進捗状況を分析する**障害対応遅延分析**機能と，プロジェクトの進

[2] http://empirical.jp/

捗度合いを分析する**プロジェクト遅延リスク検出**機能である．ただし，これらの分析機能は EPM と密接に結合するものではないため，EPM の Analyzer としてではなく，個別のツールとして実装することができる．そこで，我々は障害対応遅延分析，プロジェクト遅延リスク検出分析を行う分析補助ツールを開発した．

## 2　概要

本レポートで紹介するエンピリカルデータ分析補助ツール ProStar は，EPM のトランスレータによって作成された XML ファイルを入力として，様々なメトリクスの分析・可視化が可能であるとともに，分析単位，グラフ上の表示メトリクス組み合わせなどが比較的自由に設定できる．CSV 形式ファイルの入力や図表データの CSV 出力など，外部とのデータの入出力にも対応できる機能を持つ． 図 1 に ProStar の構成を示す．ProStar は標準エンピリカルデータ形式[1]に依存するため，構成管理ツール CVS[3]や障害追跡ツール Gnats[4]のデータ形式から EPM に含まれるトランスレータ機能を用いて形式を変換する必要がある．もちろん，標準エンピリカルデータの形式に対応しているツールであれば EPM のトランスレータを用いる必要はない．エンピリカルデータ分析補助ツールは，障害追跡ツールで管理されるバグ管理データを扱う ProQAV（Project Quality Assessment Viewer）と，構成管理ツールで管理されるソースコードデータを扱う ProPFV（Project Process Flow Viewer）に分かれる．また，ProPFV では，ソースコードデータとともにバグ管理データも使って分析を進めることができる．

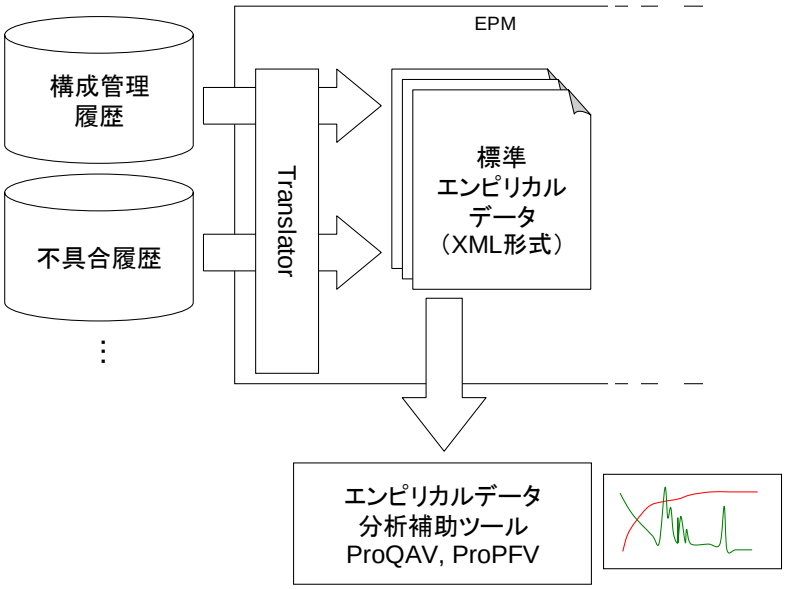


図 1　エンピリカルデータ分析補助ツールの構成

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

## 3　ProQAV による分析

ProQAV は，バグ対応の進捗を管理するためのツールである．ProQAV では，EPM による XML 形式のファイルに加えて，CSV 形式のファイルも扱うことが可能である．ProQAV では，図 2 の障害対応遅延分析データ作成画面と，図 3 に示す障害対応遅延分析グラフ作成画面の 2 つの画面を使って分析を進める．

図 2 では，分析対象とする XML データや CSV データを指定して，グラフ作成に使用するデータを CSV 形式のファイル上に作成する．このとき，画面上の「BUG データ絞込み」に表示されている SQL 文を必要に応じて修正することで，特定の条件を満たすデータのみを表示対象にすることもできる．

作成した CSV データは，ProQAV の図 3 に示される障害対応遅延分析グラフ作成画面を経てグラフにすることができる．このグラフ化での大きな特徴の 1 つは，バグの累積件数や未解決件数などの表示内容を重要度と優先度に従って選別できることである．

次に，ProQAV でのグラフ表示例を示す．


**図 2　障害対応遅延分析データ作成画面**

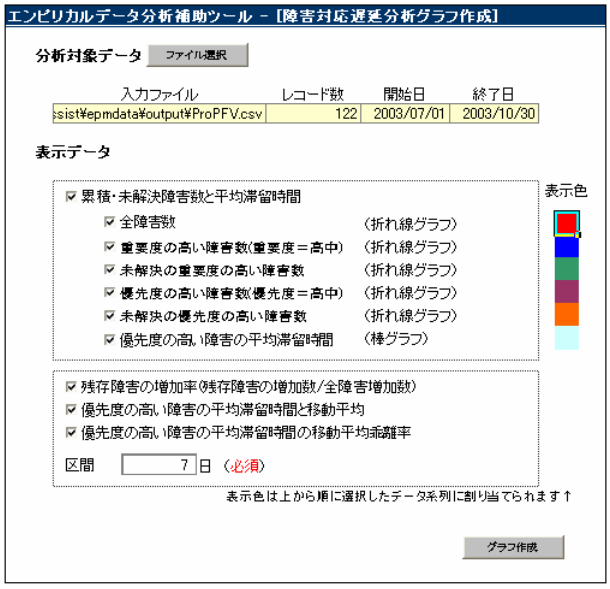


図 3　障害対応遅延分析グラフ作成画面

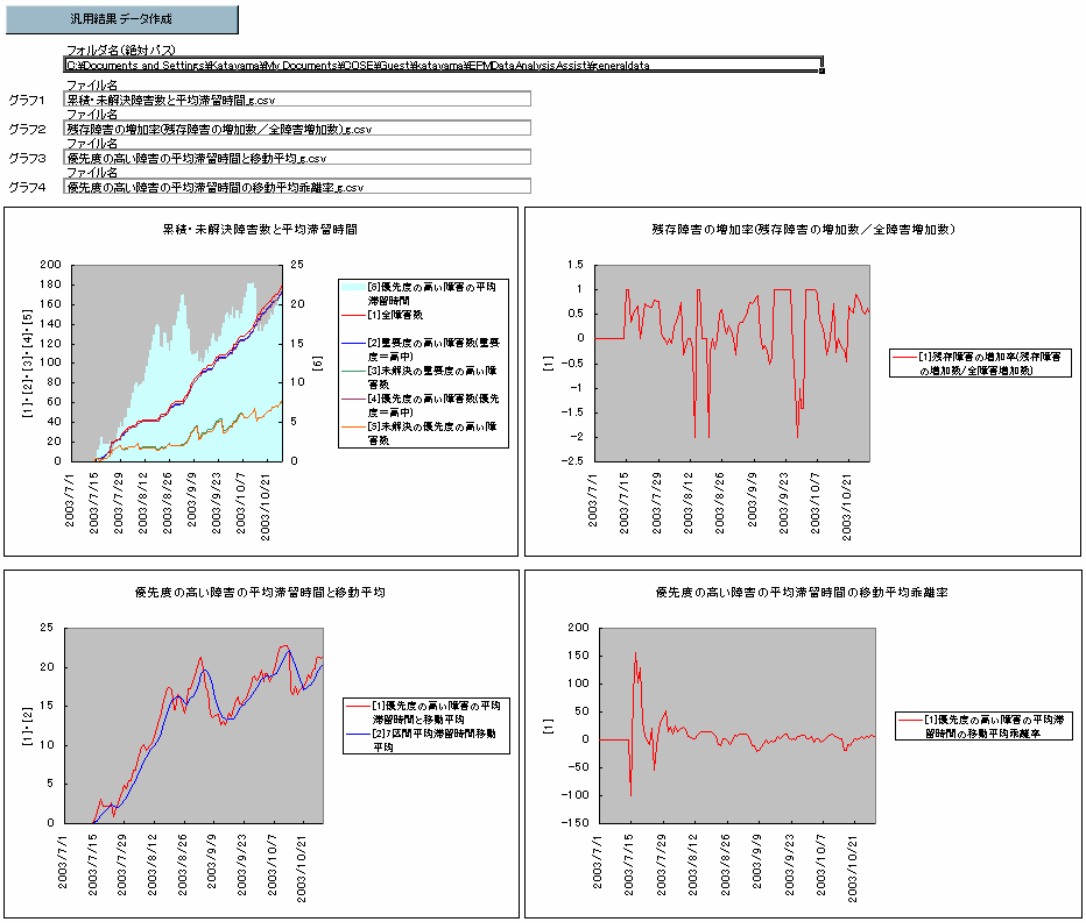


図 4　ProQAV でのグラフ表示例

図 4 は ProQAV でのグラフの表示例を示しており，EPM と同様のバグ累積件数，未解決件数，平均滞留時間の推移を示すグラフが左上に表示されている．また，このグラフでは，重要度や優先度の高い障害に関するデータも表示されている．

さらに，ProQAV では，バグの発生数と残存数からその残存率を求め，その増減をグラフとして表示することができる．このバグ残存率の推移は図 4 の右上に表示されている．このグラフによって，発生したバグのうち未解決のものの増減がどのように推移しているかを，視覚的に把握することが可能である．たとえば，発生した 10 件のバグのうち 5 件が発生日中に解決されなかったときには，バグの残存率は(10−5)÷10=0.5 となる．このようにして求める残存率が 0 より大きい状態にあり続けるときには，何らかの原因によって障害への対応が追いついていないと判断できる．

図 4 に示している ProQAV によるグラフのうち左下には，バグの滞留時間の推移が示されている．平均滞留時間は，新たなバグが発生すると一時的に減少する．このような変動を平滑化するために，指定区間の平均滞留時間の移動平均を表示させることができ，左下のグラフは区間を 7 日間としている．そして，同様に右下には，この移動平均とそれに続く実測値の乖離率が示されている．たとえば，新たにバグが発見されたときや，バグが解決されたときには，一時的に乖離率が大きく変動する．ただし，このような乖離率の変化は問題ではない．これに対して，乖離率が右上がりの上昇傾向を示したままであったときには，バグの滞留時間が延び，バグ対応の遅延が拡大していると考えられるため，その原因を解決する必要がある．

## 4　ProPFV による分析

ProPFV は，プロジェクトの進捗を管理するためのツールである．ProQAV と同様に，ProPFV では，CVS などの構成管理ツールのデータから作成した XML ファイルの内容を元にして，グラフ化のためのデータを作成する．また，ProPFV では，Gnats などの障害追跡ツールのデータも合わせて利用することができる．

ProPFV では，ProQAV と同様に**図 5** に示すプロジェクト遅延リスク検出データ作成画面から，処理対象とする XML データなどを指定し，グラフ作成のために使用する CSV データを作成する．

図 5 では，ProQAV の障害対応分析データ作成画面（図 2）と同様に，表示対象とするデータを SQL 文によって絞り込むことができる．また，画面上の「コンポーネント選択条件」での指定によって，CVS が管理するリポジトリの特定のディレクトリ上にあるコンポーネントのみを表示対象にすることも可能である．

作成した CSV データは，図 6 に示すプロジェクト遅延リスク検出全体グラフ作成画面を使ってグラフにして表示することができる．ここで表示可能なデータの種類は，EPM に比べて多くなっている．たとえば，ファイル更新回数の累積値，10 行以上削除された更新の

回数，追加行数，削除行数や，それぞれの比率などを表示することができる．また，障害追跡ツールのデータも利用しているときには，仕様変更，設計変更，コーディングミスのそれぞれによるバグの発生件数や密度も表示することができる．さらに，複数のプロジェクトを表示対象としたときには，それぞれのプロジェクトの開始点や終了点をそろえてグラフ上に表示することもできる．これによって，たとえば，昨年のプロジェクトと現在のプロジェクトの進捗状況を比較するといったことが容易に行うことが可能になる．

図 6 のプロジェクト遅延リスク検出全体グラフ作成画面での指定によって，図 7 のようなグラフを表示することができる．図 6 の例では，各プロジェクトでのプログラム行数の推移を示している．

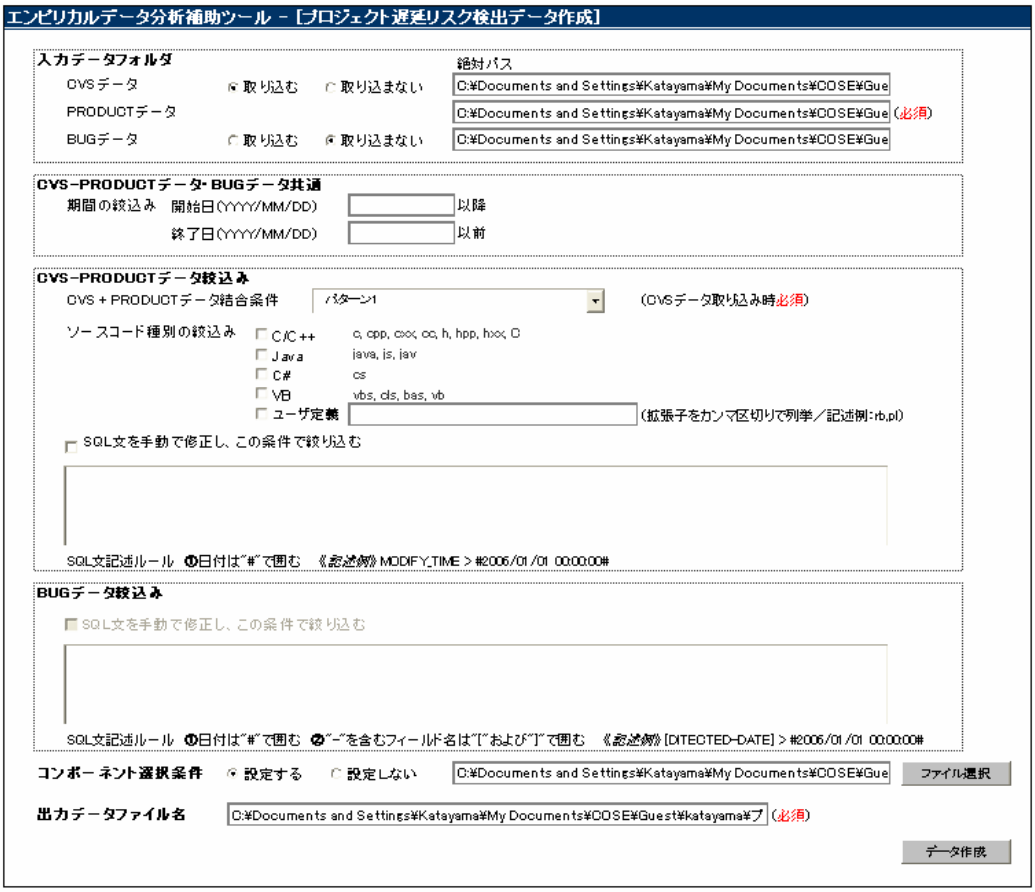


図 5　プロジェクト遅延リスク検出データ作成画面

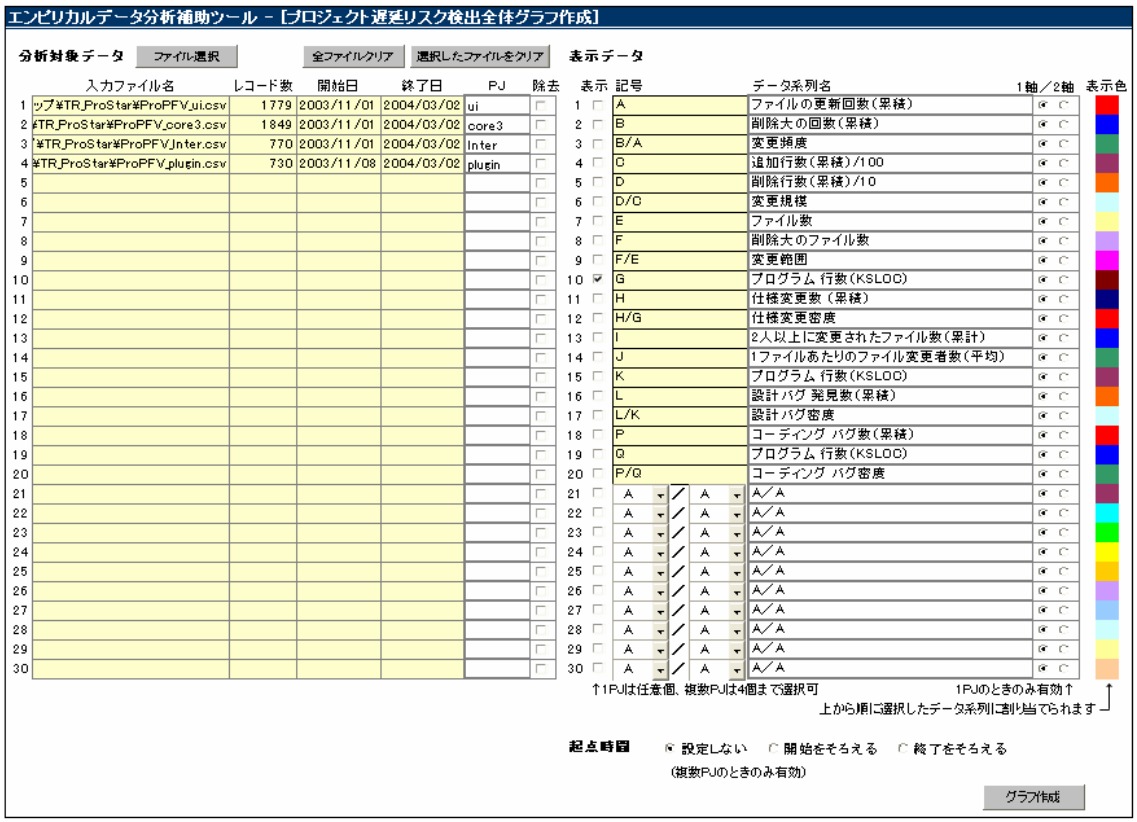


図 6　プロジェクト遅延リスク検出全体グラフ作成画面

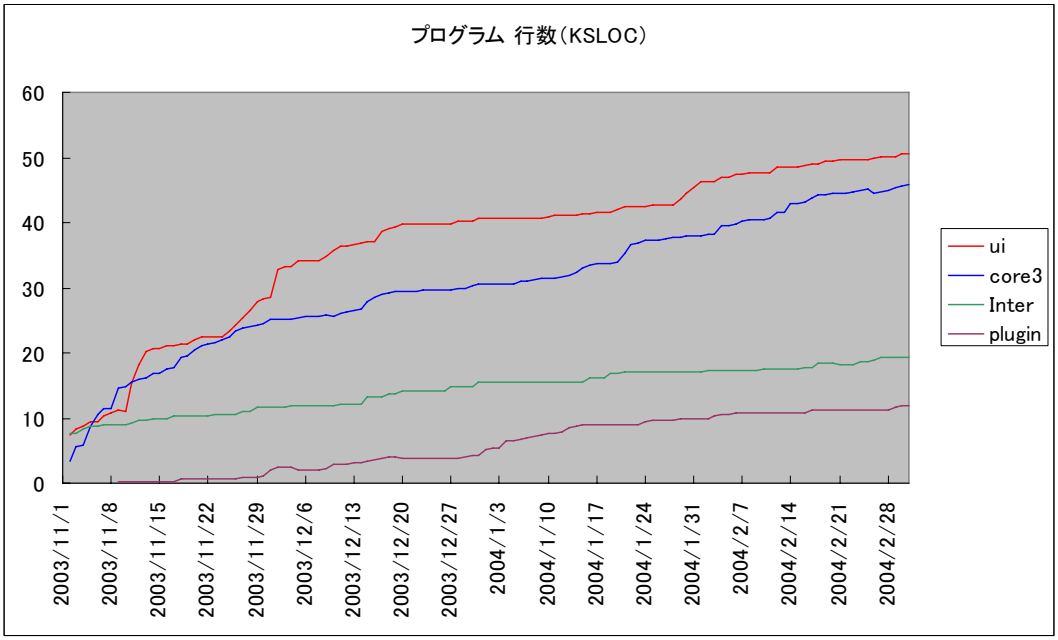


図 7　ProPFV でのグラフ表示例


**図 8　プロジェクト遅延リスク検出差分グラフ作成画面**

また，プロジェクト遅延リスク検出データ作成画面（図 5）で作成した CSV データを，図 8 のプロジェクト遅延リスク検出差分グラフ作成画面を使ってグラフ化して表示することが可能である．図 8 では差分期間を指定することができる．たとえば，差分期間として 7 日間を指定することで，1 週間での変化をグラフにして見ることが可能になる．このように差分期間を指定することで，全体の累積を表示していたときには小さくて見えづらかった変動が視覚的にわかりやすくなる．実際にプロジェクトの後工程になるにしたがって，この差分期間による表示の有用性は高くなる．

図 6 と同様に，プロジェクト遅延リスク検出差分グラフ作成画面（図 8）での指定によって，図 9 のようなグラフが表示できる．図 9 の例では，各プロジェクトでのファイル更新回数の推移がその差分で示されている．

さらに，ProPFV では，ファイルの更新回数や行削除回数といった回数，追加行数や削除行数といった変化量に加えて，作業者に関するデータも処理することができる．具体的には，ファイルの更新回数（図 11）やファイルへの追加行数などの回数や行数の変化（図 12）を見ることができ，作業の進捗を確認することができる．また，2 名以上の作業者によって変更が加えられたファイル数と変更を加えた作業者の 1 ファイルあたりの人数の確認（図 10）など作業者に関するデータを見ることで，要員の管理や配置が適切に行われているかどうかが確認できる．たとえば，2 名以上の作業者によって変更が加えられたファイル数が

増えているときには，要員の交代が行われたと考えられるため，要員のスキルや知識の差による影響を考慮する必要があると思われる．このように，作業者に関するデータの変化を見ることで，現場の動きを捉え，その影響を早い段階で処置することが可能になる．

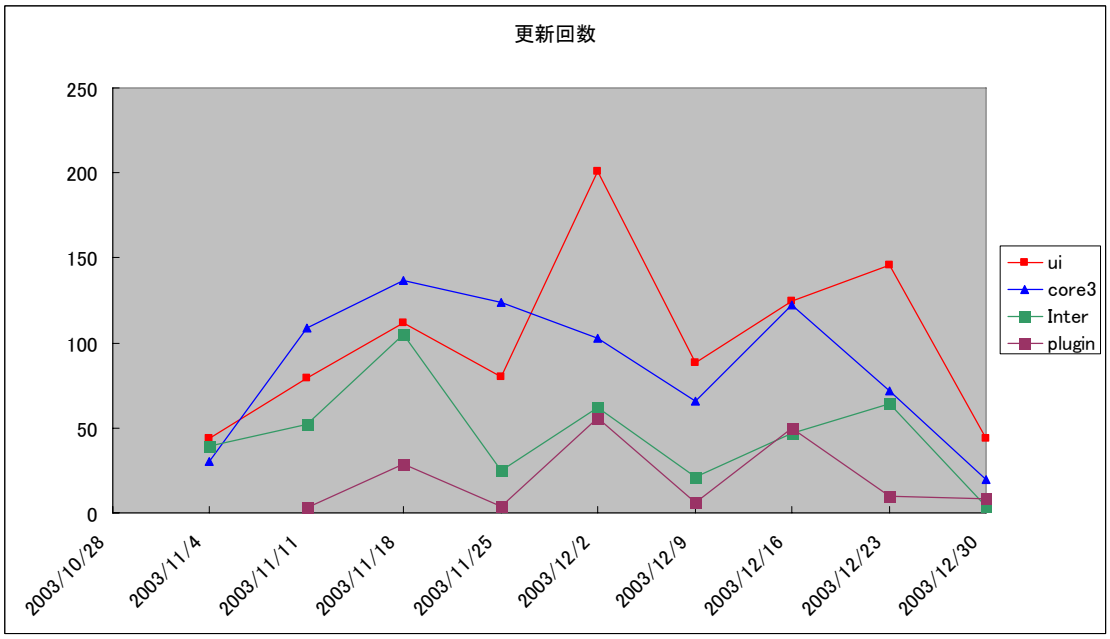


図 9　ProPFV でのグラフ表示例

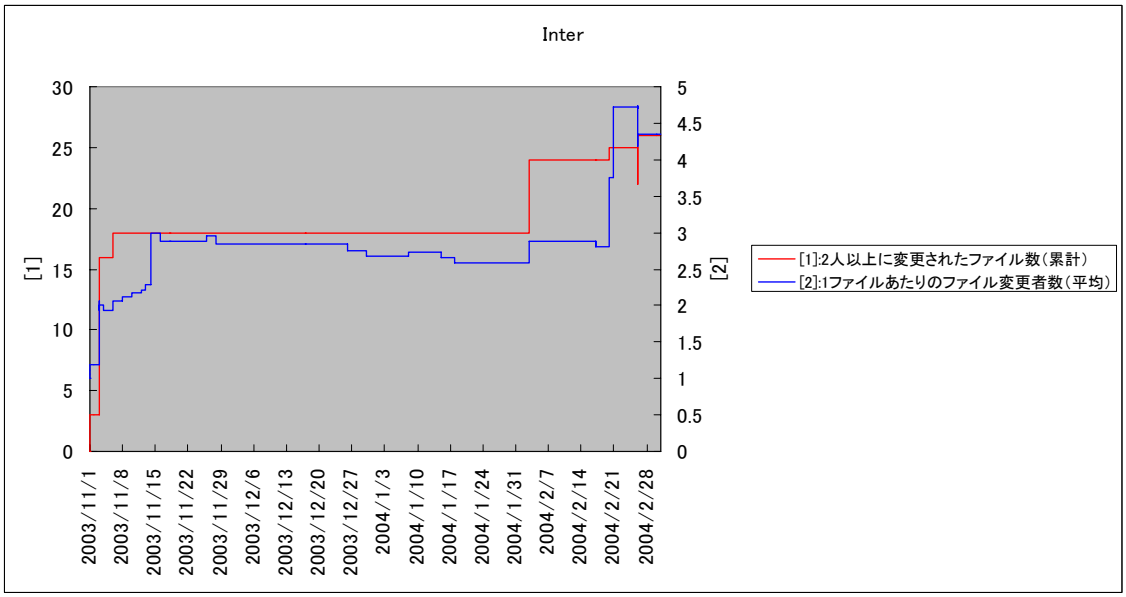


図 10　2 人以上に変更されたファイル数（累計）

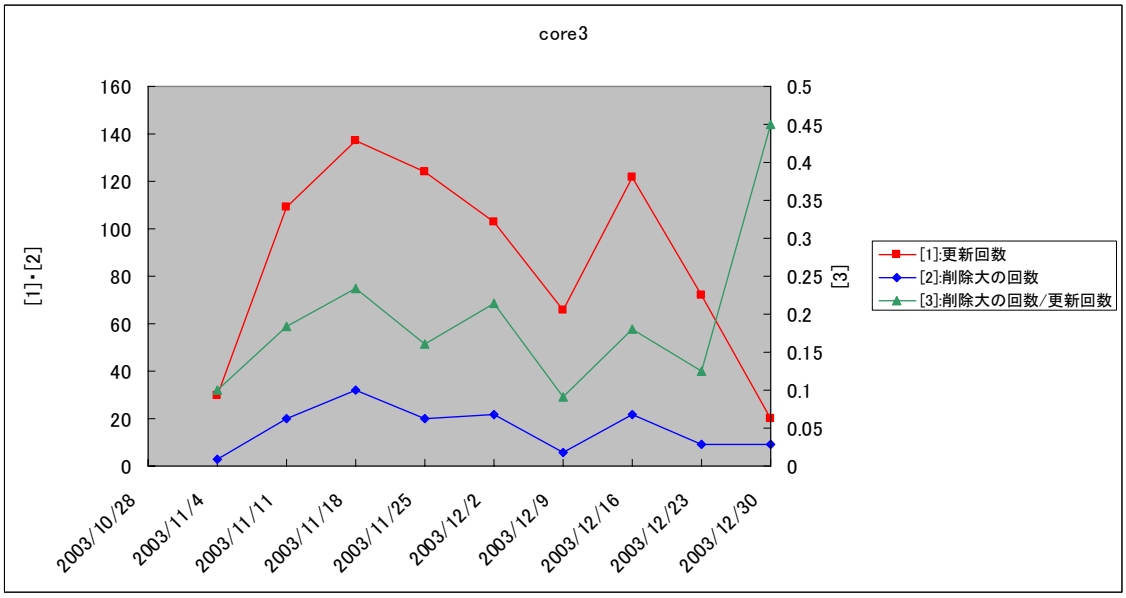


図 11　ファイルの更新回数

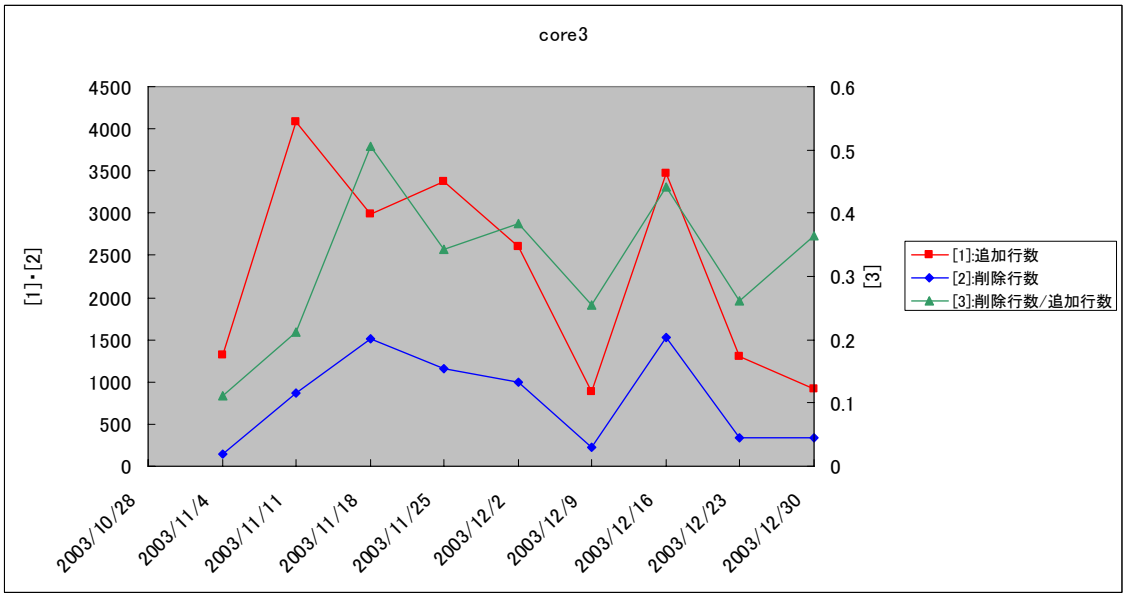


図 12　追加・削除された行数

また，ProPFV による処理を予実管理に利用することができる．ProPFV のプロジェクト遅延リスク検出全体グラフ作成画面（図 6）では，複数のプロジェクトを表示対象に指定できる．このうちの 1 つに計画内容を表す CSV データを指定することで，予定と実績との差を比較し進捗を管理することが可能となる．

## 5 まとめと今後の課題

本ツールは，2006 年度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，その有効性が確認され，運用上の改善点などを検討することができた[4].

今後は，運用プロセスの確立を目指して研究する予定である.


## 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.


## 参考文献


[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] 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.

[4] 松村 知子，勝又敏次，森崎 修司，玉田 春昭，大杉 直樹，門田 暁人，楠本 真二，松本健一，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，奈良先端科学技術大学院大学テクニカルレポート, NAIST-IS-2007001, February 2007.