解 説

実証的ソフトウェア工学環境への取り組み

近年，ソフトウェア開発に関するプロセスやプロダクトのデータを収集，分析し，改善を行うという実証的ソフトウェア工学の研究や実践が盛んになってきている．本稿では，実証的ソフトウェア工学の意味や動向を紹介する．また，文部科学省のリーディングプロジェクトの１つとして，実証的ソフトウェア工学の実践を支援する環境の構築を行っているEASEプロジェクトについて概説する．EASEプロジェクトでは，開発時のデータを収集し分析して開発者や管理者に提示するシステムEPMを構築している．EPMを用いることにより，対象となる組織のプロセス改善を少ない手間で効果的に行える．

井上克郎[*1]　松本健一[*2]　鶴保征城[*3]　鳥居宏次[*4]

[*1] 大阪大学　inoue@ist.osaka-u.ac.jp
[*2, 4] 奈良先端科学技術大学院大学　[*2] matumoto@is.naist.jp [*4] torii@is.naist.jp
[*3] 独立行政法人 情報処理振興機構／高知工科大学　tsuruho@ipa.go.jp

■ソフトウェア開発の現状と工学■

現在，システム構築の遅延や予算オーバーの問題に悩まされていないSI会社やソフト会社は皆無だと言っても過言ではなく，場合によっては経営の屋台骨を揺るがしかねない状況になってきている．顧客の要求が明確でない，カットオーバーまでの期日が短い，技術の進展に対して人材の育成が追いつかない，等々の問題がボディブローのようにシステム構築に大きな影響を与えている．これらの条件は，今後ますます厳しく，難しくなることはあっても，緩和される見込みはない．

エンドユーザにとっても他人事ではない．システム構築の失敗が経営を直撃するのは，銀行システムの例を持ち出すまでもなく明らかである．ベンダ，ユーザにおけるこのような背景が，実証的ソフトウェア工学への大きな期待になっている．

そもそも，どのような産業にも基盤となる工学があり，当然のこととして実証的・実験的（エンピリカル）アプローチがなされている．たとえば，構造物の材料の質・量をどの程度落とせば，強度がどの程度落ちるのか，実証的に明らかにされており，このような知見をベースとすることによって，企業は安心して限界ぎりぎりまで，生産性向上やコスト削減に邁進できる．

しかし，工学が確立されてなくても，構造物は作られてきている．たとえば，橋が比較的低速の風による振動で崩落するという事象は，1940年の米国Tacoma Narrowsの事故以降解明されたが，それ以前のものは，破壊の危険に常に曝されていた．工学に立脚せずに構築された物は，非常に不安定であり，事故と隣り合わせである．

では，ソフトウェアの分野はどのような状況なのだろうか．ソフトウェアの信頼性，生産性は，いわゆるソフトウェア工学という分野で研究され議論され続けているが，従来のソフトウェア工学は，方法論やツールを個別に提示していたに過ぎない．社会や生活の隅々にまで入り込んだ巨大システムの構築を支えるには，あまりにも脆弱な状況で，これが，現在多くの会社を直撃しているソフトウェアシステムのトラブル多発の要因と言えよう．

本稿では，実証的ソフトウェア工学の研究動向を紹介し，それを推進するEASEプロジェクトについて述べる．

■実証的ソフトウェア工学■

ソフトウェア工学

ソフトウェア工学は，1968年にNATOの科学委員会で初めて用いられた言葉で，ソフトウェアの生産や品質に関する技術の総称として用いられている．今までに数多くのソフトウェア工学にかかわる技術（手法，ツール，方法論等）が提案されてきている．しかし，それらの中には，提案されるにはされたが，しばらく時間がたつと誰も使わずに，忘れさられてしまうものも多い．工学と呼ぶにはふさわしい積み上げもあまりない．

ソフトウェア工学の分野では質，量ともに最大であるソフトウェア工学国際会議（International Conference on Software Engineering：ICSE）では，このようなソフトウェアに関する技術を評価する１つの試みとして，発表される論文の評価をその時点で行うのではなく，10年後に，

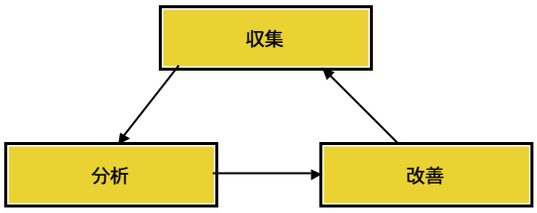

図-1　工学におけるデータ収集・分析・改善モデル

最も影響を与えた論文（Most Influential Paper）として表彰する制度を設けている．たとえば，2003年の第25回会議では，1993年に開催された第15回大会の発表論文の中から，要求仕様における複数視点に関する論文が選ばれている[1]．ソフトウェア技術は，直接の評価が困難なため，長期間に渡る社会に対する普及度，影響度を調べて間接的に評価しているのである．しかしこのような悠長な手法では，現在のソフトウェアシステムに対する社会的な要望に応えられないのは明白である．

実証的ソフトウェア工学

通常，工学の多くの分野では，図-1のように，何か新しい技術の提案が行われれば，まず，それが優れているものかどうかを評価するために，データの収集が行われ，定量的や定性的に分析される．さらにその分析結果に基づいて，技術の改善が行われる．

ソフトウェア工学の技術に関しても，他の工学分野と同様に，短期間かつ直接的な評価が行われることが望まれる．ある新しい技術が提案されても，それが本当にソフトウェアの品質や生産性の向上につながるのか，きちんとした科学的な評価がなければ，現場に適用することは躊躇される．逆に説得力ある評価結果が示されれば，積極的に導入しやすい．

このような考えに基づいて，ソフトウェア開発に関するデータの収集を行い，その定量的（または時には定性的な）分析を行って，ソフトウェアの品質や生産性の改善を行うことを，実証的ソフトウェア工学（Empirical Software Engineering）と呼ぶ．

実証的ソフトウェア工学に関する動向

実証的ソフトウェア工学は，今までも通常のソフトウェア工学に関する雑誌や国際会議の中でも断片的に議論されてきているが，近年，専門の雑誌や国際会議ができつつある．

Empirical Software Engineering Journal（ESE）は，Kluwer社から年4回刊行されている国際学術誌で，厳密なデータ分析，評価に基づいた数多くの実証的ソフトウェア工学の論文が掲載されている．

実証的ソフトウェア工学国際会議（International Symposium on Empirical Software Engineering：ISESE）は，3年目を迎える新しい実証的ソフトウェア工学に関する国際会議で，毎年100～200名の参加者を企業や大学から集めている．

また近年，いろいろな所で，実証的ソフトウェア工学を専門に研究する組織が設立されてきている．

Institute for Experimental Software Engineering（IESE）はドイツのFraunhofer財団が作る実証的ソフトウェア工学の研究機関で，ドイツの東部のKaiserslautern にある．現在100名余りの研究者がおり，大学と企業との研究連携，人材育成の場となっている．また，米国のMaryland州にも分室があり，20名程度の研究員が在籍している．

日本では，経済産業省のもとで，ソフトウェア工学センター（SEC）が間もなく設立され，実証的ソフトウェア工学がその活動テーマの重要な1つとなる予定である[2]．

■EASEプロジェクト■

e-Society産官学連携プロジェクト

平成15年度より始まった「e-Society基盤ソフトウェアの総合開発」は，文部科学省が行うリーディングプロジェクトで，世界最高水準の高度情報通信システム形成のための鍵となるソフトウェア開発を実現させ，いつでもどこでも誰でも安心して参加できるIT社会を構築することを目的としている[3]．このe-Societyのもと，10のプロジェクトが5年計画で行われている．これらのプロジェクトでは，既存の大学的な研究ではなく，日本の産業育成や活性化のキーになる成果を求められている．そのためには，産官学の連携を推進し，企業や大学が単独では行いにくいソフトウェア開発を行い，作成したソフトウェア製品を社会に広く流布させることが必要である．

e-Societyの1つとして我々は「データ収集に基づくソフトウェア開発支援システム」，通称EASE（Empirical Approach to Software Engineering）プロジェクトを推進している[4]．

EASEプロジェクトの目的

EASEプロジェクトでは，以下のようなことを目指している．

1. エンピリカルソフトウェア工学の研究と実践を行うための基盤として，ソフトウェア開発にかかわる種々のデータの収集・分析・改善を行う開発管理支援システム「実証的ソフトウェア工学環境EPM（Empirical Project Monitor）」を構築する．
2. EPMを広く配布し，多くの開発現場で利用され得るように種々の支援を行う．

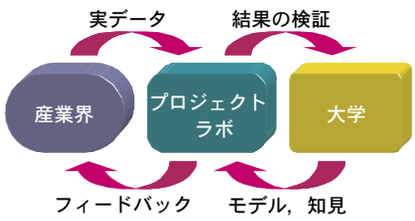

図-2　産学連携モデル

3. EPMによって集められたデータを可能な限り共有し，ソフトウェア開発における標準的なデータの蓄積を図る．
4. EPMを適用した組織が，これを利用したことによって生産性や品質の向上が推進され，利益を得られる．

これらのことを推進するために，図-2のような大学と企業との連携モデルに基づいて，大阪府の千里中央にラボを設け，奈良先端科学技術大学院大学と大阪大学の2大学，そしてNTTソフトウェア，日立製作所，日立公共システム，SRA先端技術研究所の4社からそれぞれ研究者や開発者が集まり，研究開発活動を行っている．

■実証的ソフトウェア工学環境EPMの概念■

ソフトウェア工学の研究として，ソフトウェア開発者個人の作業データを集め，それを分析し，改善につなげる試みは，今までも多数ある．また，1つのプロジェクトの開発データを収集，分析し，改善に導く手法や技術も多数あり，実際，そのような提案に基づいて，数多くのツールが開発されてきている

しかし，図-3に示すように，何百というプロジェクトが同時に走っているようなソフトウェア開発会社全体のデータ収集を行い，それらの比較や分類，現状把握をし，会社全体の改善につなげていけるようなアプローチは，あまり試みられていない．

本プロジェクトでは，個人から会社全体まですべてを統一的に扱えるような，実証的ソフトウェア工学環境EPMの開発を目指している[5), 6)]．

図-4にEPMをある会社とその関連会社に適用した場合の概念を示す．会社内のすべてのプロジェクトデータは，ネットワークを通じてリアルタイムに集める．また，外注先会社や請負先会社のデータも収集する．さらに，パブリックドメインソフトウェアやその関連データも必要に応じて収集する．そして，これらのデータを分析，加工し，改善に必要な情報を開発者や管理者にフィードバックする．

■EPMの概要■

EPMの設計方針

EPMを実際のソフトウェア開発環境の1つとして実現するために以下のような方針を立てた．

1. データ収集に関しては，開発者の負担を最小にし，既存の開発環境の改変はしないで，できる限りリアルタイムに収集するようにする．
2. 分析は，簡単な単一プロジェクトに対するメトリクスの計算機能の実装から行い，プロジェクト間のメトリクス情報の計算，プロジェクトの分類，変遷の解析，再利用部品の抽出，知見の再利用などの分析を，順次追加していく．
3. 分析結果をどう改善に利用するかについては，その目的によって必要な支援が大きく異なる．ここでは，分析データを視覚化し，管理者や開発者が，容易にプロジェクト状況を理解できるようにする．

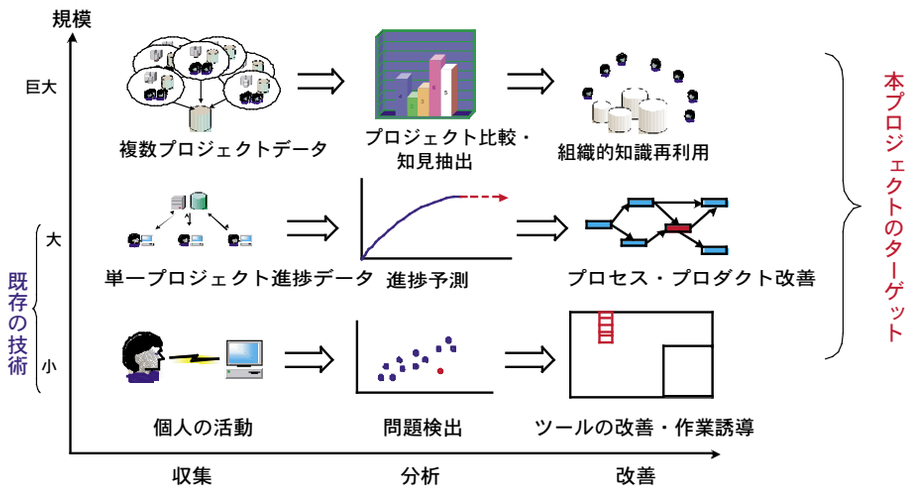

図-3　実証SEの対象の規模による分類

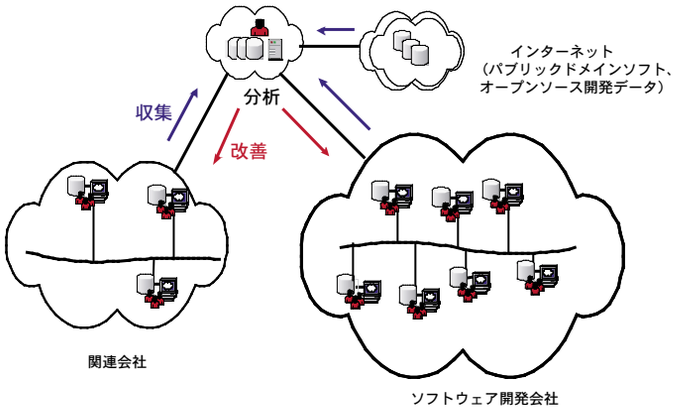

図-4　EPM適用の概念

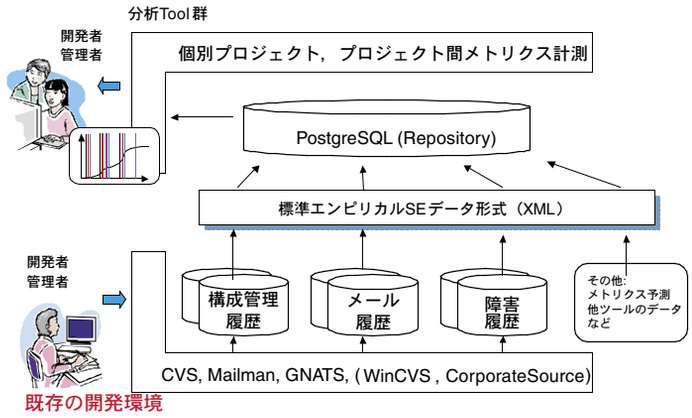

図-5　EPMのアーキテクチャ

EPMのアーキテクチャ

図-5に現在開発中のEPMのアーキテクチャを示す．EPMでは，開発者が，構成管理ツールCVS，メール管理ツールMailman，障害管理ツールGNATSを使用することを前提としている．それぞれのツールのデータベースには，開発プロダクトの履歴，開発者の間でやりとりしたメール，そして障害管理情報などのデータが独自の形式で蓄積されている．EPMでは，それらを，誰が，いつ，どのようなイベントを行った，かのタグ付けを行ったXML形式のデータに変換する．このデータを標準エンピリカルSEデータと呼んでおり，PostGreSQLデータベースに格納している．

現在のEPMでは，これらの特定の開発ツールを使うことを前提としたが，これらは，近年，UNIX系システムのみならず，PC系などいろいろな開発形態で頻繁に用いられているものである．また，これらのツール以外のツールを使う開発環境であっても，標準エンピリカルSEデータ形式に変換するトランスレータを作成すれば，以降の処理を同様に行うことができる．

収集されたデータに対して，SQLのコマンド実行により種々のメトリクス情報を計算する．得られたメトリクスは，ユーザのWebブラウザ上にグラフ化して表示する．

EPMの機能

現在，EPMには，対象とするプロジェクトを選択し，プログラムサイズ，バグ個数，メール数などのグラフを表示する機能を標準として用意している．また個別の開発形態に合わせて，プログラムサイズの予定成長曲線やバグの収束曲線を追加することもできる．

実際のEPMの出力を図-6，7，8に示す．これらは，EPMの開発自体を計測の対象とした例で，図-6は，EPMの総行数の変遷を示している．縦線は，CVSに対するチェックイン操作を行った時期を示す．このグラフに

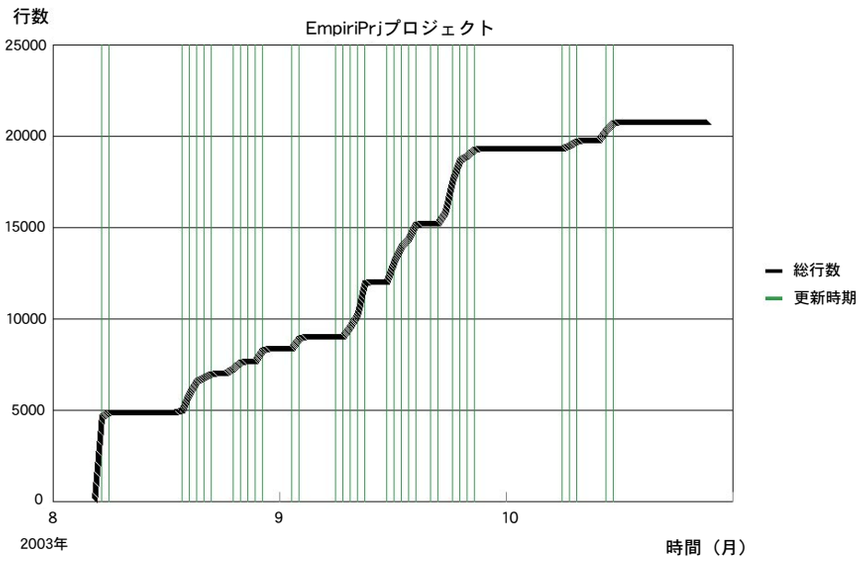

図-6　総行数の変遷

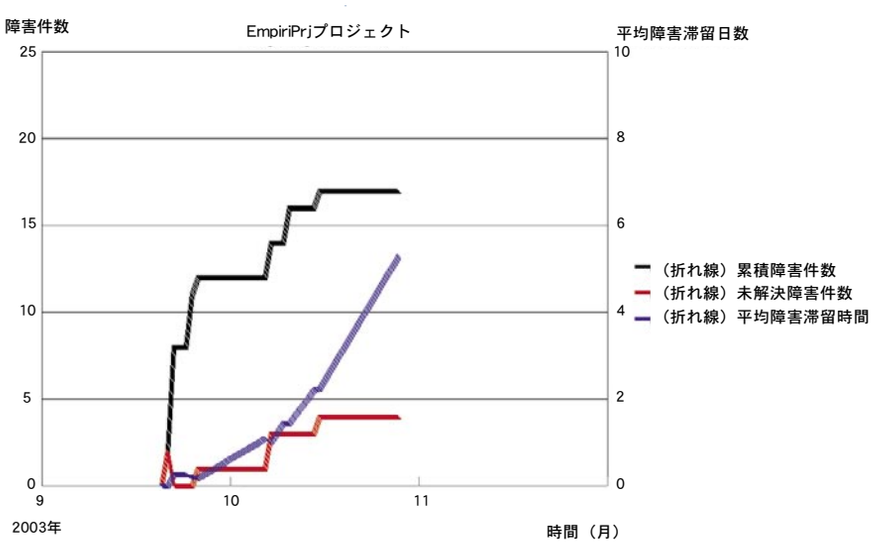

図-7　総障害数，残存障害数，平均障害除去時間の変遷

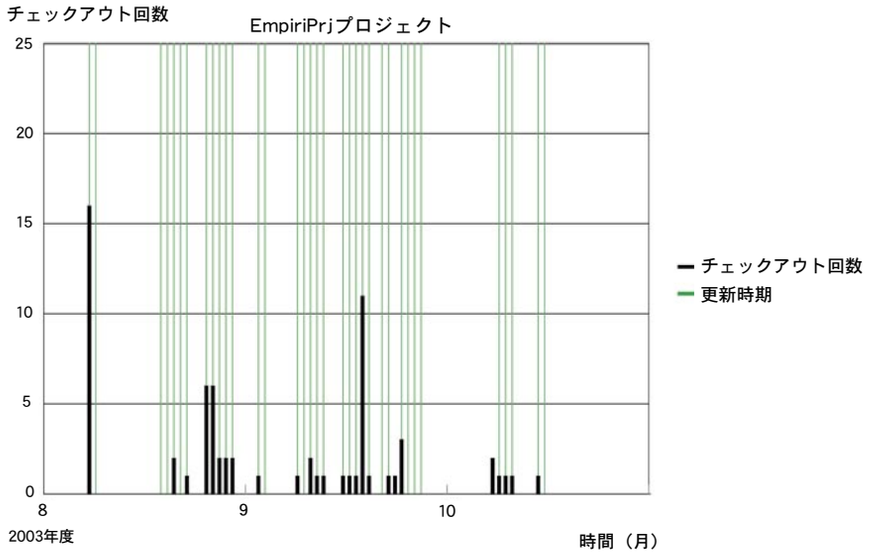

図-8　チェックイン時刻とチェックアウト回数

より，たとえばプロジェクトがどのような状況にあるのか，すなわち，コーディングの初期の立ち上がり時期なのか，中盤の活発な開発時期なのか，終盤の収束期なのかを容易に把握できる．また，過去の同種のプロジェクトや予定との比較により，進捗の遅れや進み具合を知ることができる．図-7は，生じた障害の累積数や未解決の障害数，そして，障害を解消するのに要した平均時間を示している．これにより，発生している障害の状況の推移を直感的に把握でき，開発者や管理者への障害解消への動機付けになろう．図-8は，CVSにチェックインした時刻と，チェックインしたものをチェックアウトした(取り出した)回数を示している．これを見ることにより，更新したものが適切に参照されているかが分かる．

これらのグラフを見ることにより，たとえば，プロジェクトの進捗が9月下旬から10月上旬に停滞していた，障害の滞留が増え解決時間が徐々に大きくなりつつある，などが容易に分かる．

EPMの運用

EPMを有効に使うためには以下のような簡単な制約のもとに各作業者が開発作業を行う必要がある．

- 各作業者は，日常作業の結果を定期的に決められた場所のCVSレポジトリに保存する．たとえば，1日の作業の終了時に，必ずその日の開発物を決められた場所のレポジトリにチェックインする，という慣行を実行する．このチェックインの頻度が少なければ，得られるデータの実時間性が低下する．
- プロジェクトに関する情報交換は，もっぱら決められたメーリングリストで行い記録を残す．そのアドレスは，開発に関連することのみに用い，他の用途には別

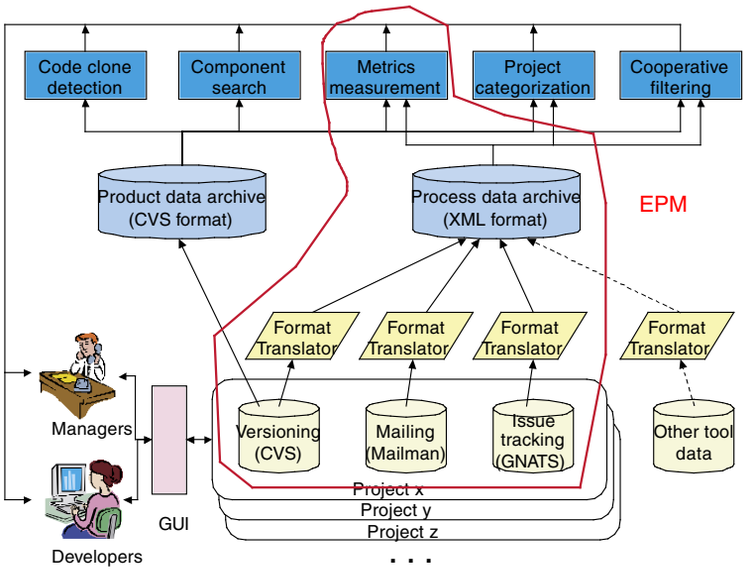

図-9　EPMの発展

のアドレスを使用する．

- 障害の発生やその解決等の管理は，必ず障害管理ツールで行い，そこに障害の分類とその内容の記録を残す．

このような制約は，既存のプロジェクト慣行とは異なる場合もあろうが，大規模なデータ収集を行うものとしては，比較的簡便で，適用しやすいものと考えている．また，このデータ収集方式は，主にプロダクトデータを集め，そこから必要なプロセスデータを復元する，という手法である．この方式では，詳細なプロセスデータを得ることは困難であるが，既存の作業環境に特殊なデータ収集ツールを組み込む，などの変更が不要で，導入しやすいものと思われる．

この方式では，開発データの下流工程のデータは直接収集できるが，仕様や設計などの上流工程のデータを収集するのには，工夫が必要である．これらの上流工程のデータに関しては，仕様書や設計書を何らかのツールで記述して電子化し，そのデータをレポジトリに保存するようにすれば，分析して，メトリクス表示することが可能になる．

EPMで用いている既存のツールは，基本的に分散開発にも適用可能である．複数の拠点で行われる個々の開発作業の情報も，1つのレポジトリに集めることで複数の開発拠点の情報も一元管理することができる．また，近年，普及し始めている開発プロセスの成熟度モデルCMMI（Capability Maturity Model Integration）のプロセス領域KPA（Key Process Area）との対応を考えることにより，EPMをCMMIのレベル向上の支援に用いることもできる．

■EPMの発展■

EPMは現状では，プロジェクトに関する種々のメトリクス情報をグラフ化して表示する機能を有しているが，今後，以下のような機能の追加を目指している．

- プロジェクトの開始から現状までの推移データに基づいて今後の推移を予測し，グラフ上にその予測曲線を表示する機能．
- 過去のプロジェクトや同時進行している多数のプロジェクトのデータを一覧表示する機能．

さらに，図-9のように，今後新たなレポジトリや分析ツールを追加していき，以下のようなより深い分析や解析が行えるようにする．

- プロジェクトの自動分類

  すべてのソフトウェア開発プロジェクトのデータを一元管理するレポジトリを用いて，プロジェクト間の類似度や距離を計測し，プロジェクトの分類の解析を行う[7]．分類の手がかりとしては，ソースプログラムやドキュメントに含まれる語彙分布などのメトリクス情報，文字列としての対応度，構造としての近さなどが考えられる．これらを目的に応じて使い分けるとともに，分類結果が直感的に把握できるような表示を行う．
- 協調フィルタリング

  プロジェクトの分類の一種とも考えられるが，与えられたプロジェクトに最も近いプロジェクトを，複数のメトリクス情報から得る．この際，すべてのメトリクス情報を全プロジェクトに渡って揃える必要はなく，一部のメトリクスに抜けがある場合でも，結果を得ることができる[8]．
- コードクローン分析

  ソースコードの断片で，別の場所に同型の断片が存在するものをコードクローンと呼び，その高速解析を行うツールが開発されている[9]．コードクローンを調べることによって，ソフトウェアシステムの品質やその変遷を知ることができる．

- ソフトウェア部品検索
  すべてのプロジェクトのソースコードを分析し，同一部分や類似部分として繰り返し利用されているコード断片を，再利用部品として取り出し，部品の利用率に基づいて，検索部品を提供するシステムが開発されている[10]．本システムをEPMに組み込むことで，一度作成したプログラムからソフトウェア部品の自動抽出が行え，部品としての再利用が促進される．

これらのツールを追加することによって，たとえば以下のシナリオのような改善を期待することができる．

1. 進行中のプロジェクトXをEPMでモニタしていると，計画に対して実際の進捗が遅れていることに気づく．
2. 何が問題かを知るために，Xと似た過去のプロジェクトをレポジトリから探す．これには，プロジェクトの分類や協調フィルタの技術を用いる．
3. 検索の結果，過去のプロジェクトXとYが類似していることが分かった．XとYのいろいろなメトリクス情報を比較し，Yに比べてXのプログラム再利用率が極端に悪いことが判明した．
4. ソフトウェア部品検索システムの利用をプロジェクトXの中で促進し，再利用の向上に努め，生産性向上を目指す．

■現状と今後の展望■

実証的ソフトウェア工学とそれを実践しているEASEプロジェクトについて概説した．

現在，EPMは，α版の開発がほぼ完了し，実際の企業のプロジェクトに適用しようとしている．今後，適用する企業を増やすとともに，意見を集約してEPMの改良を行う予定である．また，海外の研究者や研究所との連携を強化していく予定である．

ソフトウェア工学という用語が使われ始めて30年を越える年月が過ぎているが，工学と呼ぶにはふさわしい積み上げもあまりなく，高度情報化社会という魅惑的な言葉の影に隠れて農産物にも負けないくらいの目に見えない庇護を受け，それ以降の進化をとどめてしまって現状の悲惨な技術レベルになってしまった，と言っても言い過ぎではないであろう．

ソフトウェア工学が機械工学や土木工学の水準に達し，見通しの良い方法論のもとでシステム構築が整然と進められるようになるには，まだかなりの時間を要すると思われる．しかし，ソフトウェアベンダもユーザも手をこまねいて，方法論の確立を待っているわけにはいかない．

今現場では，優秀なエンジニアが，後手になったトラブルプロジェクトの修復に走り回る光景がよく見られる．トラブルの症状を，実証的データに基づいてより早いタイミングで把握することだけでも相当な効果が期待できるのではないか．地道にプロセスの改善を積み重ねる以外に現状を一挙に打破する銀の弾丸はないであろう．

比較的成熟した工学的な裏づけを有する自動車産業においては，毎年膨大な数の改善を測定されたデータに基づいて実施し，膨大な金額の合理化を達成しているという．これが巨大な利益の源泉になっている．逆説的に言うと，自動車産業の足元にも及ばないソフトウェア産業は，合理化余地の大きい有望な産業だとも言える．

遅まきながら，まず着実にデータを集め，それを分析し，その分析結果に基づいて，プロセス改善やプロジェクトマネジメントを実証的に行うことが必須である．実証的ソフトウェア工学の研究がより進み，その実践が広く普及して，そのようなデータを活用しながらソフトウェア開発を行うことが，ごく当たり前になろう．EASEプロジェクトがこのような動きを加速させることを期待する．

謝辞　EASEプロジェクトを支援していただいている文部科学省，協力いただいている各社に深謝する．EPMの設計や開発，展開に協力いただいている方々に感謝する．本研究は，文部科学省「e-Society基盤ソフトウェアの総合開発」の受託業務として，奈良先端科学技術大学院大学と大阪大学が行っている．

参考文献

1) Nuseibeh, B., Kramer, J. and Finkelstein, A. : Expressing the Relationships between Multiple Views in Requirements Specification, 15th International Conference on Software Engineering, pp.187-196, Baltimore, Maryland (1993).
2) 嶋田　隆，祝谷和宏：ソフトウェアエンジニアリングセンター構想について，情報処理，Vol.45, No. 4, pp.367-371 (Apr. 2004).
3) http://cif.iis.u-tokyo.ac.jp/e-society/index.html
4) EASE (Empirical Approach to Software Engineering) Project, http://www.empirical.jp/
5) 大平雅雄，横森励士，阪井　誠，松本健一，井上克郎，鳥居宏次：Empirical Project Monitor：プロセス改善支援を目的とした定量的開発データの自動収集・分析システムの試作，電子情報通信学会技術報告 SIGSS, Vol.103, No.708, SS2003-48, pp.13-18 (Mar. 2004).
6) Ohira, M., Yokomori, R., Sakai, M., Matsumoto, K., Inoue, K. and Torii, K.：Empirical Project Monitor: Automatic Data Collection and Analysis toward Software Process Improvement, 日本ソフトウェア科学会研究会資料シリーズ，No.28, 第1回ディペンダブルソフトウェアワークショップ (DSW2004)論文集，pp. 141-150 (Feb. 2004).
7) 川口真司，松下　誠，井上克郎，潜在的意味解析法LSAを利用したソフトウェア分類システムの試作，情報処理学会研究報告，2003-SE-140, Vol.2003, No.22, pp.55-62 (Mar. 2003).
8) 大杉直樹，門田暁人，森崎修司，松本健一：協調フィルタリングに基づくソフトウェア機能推薦システム，情報処理学会論文誌，Vol.45, No.1, pp.267-278 (Jan. 2004).
9) Kamiya, T., Kusumoto, S. and Inoue, K. : CCFinder: A Multilinguistic Token-Based Code Clone Detection System for Large Scale Source Code, IEEE Transaction on Software Engineering, Vol. 28, No. 7, pp. 654-670 (2002).
10) Inoue, K., Yokomori, R., Fujiwara, H., Yamamoto, T., Matsushita, M. and Kusumoto, S.: Component Rank: Relative Significance Rank for Software Component Search, Proceedings of the 25th International Conference on Software Engineering (ICSE2003), pp.14-24, Portland, Oregon (2003).

(平成16年6月2日受付)