様式 C-19

科学研究費補助金研究成果報告書

平成２１年６月２３日現在

研究種目：基盤研究（C）
研究期間：2006～2008
課題番号：18500032
研究課題名（和文）　実証データに基づく知識還元型ソフトウェア開発環境の構築

研究課題名（英文）　An empirical data based software development environment for knowledge feedback.

研究代表者
花川　典子（HANAKAWA NORIKO）
阪南大学・経営情報学部・教授
研究者番号：60351673

研究成果の概要：ソフトウェア開発において過去の過ちを繰り返さないために，過去のプロジェクトを再現するツールである“リプレイヤ”を開発した．ソフトウェアが開発によって複雑に進化する様子が過去のデータに従って可視化できる．また，再現だけでなく，人員配置の変化による進捗のシミュレーションもできる．同じ過ちを何度も繰り返さないための過去で得た知識を将来のプロジェクトに有効に活かせるソフトウェア開発環境である．

交付額

（金額単位：円）

	直接経費	間接経費	合　計
２００６年度	1,100,000	0	1,100,000
２００７年度	1,400,000	420,000	1,820,000
２００８年度	900,000	270,000	1,170,000
年度			
年度			
総　計	3,400,000	690,000	4,090,000

研究分野：ソフトウェア工学
科研費の分科・細目：情報学・ソフトウェア
キーワード：ソフトウェア進化，プロジェクトの再現，複雑さの増加，シミュレーション，過ちの繰り返し，ソフトウェア開発

１．研究開始当初の背景

近年の社会基盤である銀行のオンラインシステムや東証システム，航空機の予約システム等のシステムダウンは日本のみならず，世界中の様々な活動に影響を及ぼす．たとえば，2003 年の全日空のルーター用通信制御プログラムの設計ミスにより，全日空のすべてのシステムダウンした．さらに，全日空システムダウンの影響で，羽田空港のチェックインシステム全体がダウンし，他の航空会社のチェックインさえもできなくなり，航空業界全体へ影響を及ぼした．原因は単純な開発者間のコミュニケーション不足によるソフトウェアの不具合であり，何度も繰り返される単純な過ちである．単純な過ちやミスが社会へ及ぼす影響が大きくなり，非常に深刻な状況に陥る．ソフトウェア開発において同じ単純なミスの過ちを繰り返さない手段が必要であった．特に，従来のようにテスト工程ですべてのソフトウェアの不具合を発見することは，複数システムの連携やネットワーク環境の複雑さに伴い，非常に困難になってきた．そこで，ソフトウェア開発中に不具合が埋め込まれることを防ぐため，ソフトウェア開発環境を改善させる必要があると考えた．

２．研究の目的

ソフトウェア開発において，同じ過ちを繰り返さないソフトウェア開発環境を構築することが本研究の目的である．つまり，過去のプロジェクトで発生した事象を将来のプロジェクトへの知識還元することである．過去の事象とは，ソフトウェアの不具合を誘因する開発者間のコミュニケーションミス等の人為的な事象のみならず，プログラムや設計の技術的ノウハウ，人員配置などのマネージメント系の経験から発生する知識なども含む．これらの過去のプロジェクトで発生した様々な事象を，特定の開発者やマネージャの暗黙知から，だれでも利用できる一般的な形式知へ昇華させ，将来のプロジェクトで同じ過ちを繰り返さない開発環境を構築する．同時に，産業界の開発者が開発を行う際の様々な事象のデータを自動収集し，研究者はその自動収集されたデータに基づいて，暗黙知を形式知へ昇華させることを支援できる，すなわち開発者と研究者が過去のプロジェクトの事象や知識を共有できるソフトウェア開発環境の構築が本研究の目的である．

３．研究の方法

過去のプロジェクトの知識を還元できるソフトウェア開発環境を構築するために次のアプローチをとった．

(1)過去のプロジェクトの再現するリプレイヤの開発と検証

過去のプロジェクトから知識を得る方法は，プロジェクトをそのまま再現してリプレイする方法を採用した．「リプレイする」とは開発データである CVS データやバグ情報，メールログ等を一括管理し，時系列でそれぞれのデータをアニメーションして再現する方法である．ビデオの再生のように「再生」ボタンや「巻き戻し」,「早送り」などができ，特定の時期のプロジェクトの状況が再現できるツールである．これによって，過去のプロジェクトで発生した様々な事象を再現し，開発者への知識伝達方法のひとつとした．実際に学生プロジェクトや実プロジェクトに適用させて，画像データ形式による容量問題の知識や，ソースコードのバックアップシステムが不十分のために，数週間分のソースコードの修正情報が消えるという問題の知識を獲得することができた．

(2)ハイブリッドシミュレータの開発と検証

過去のプロジェクトをそのまま再現することはできたが，途中で人員配置などの変更を加えたときのプロジェクトの仮想の進行をシミュレーションできる機能をユーザが要望した．つまり，リプレイヤは基本的に過去のプロジェクトをそのまま再現し，過去の事象を体験して知識を得ることが目的だが，シミュレータは「もし，ここで人員を追加した場合，どのようなプロジェクト進行になるのか？」を模擬実行する機能である．シミュレーションモデルには作業分割によるコミュニケーション工数モデルを採用し，人員追加した場合の並列作業による時間短縮に加え，作業分割による工数増を計算して，仮想の進捗の計算を試行した．これによって，様々なマネージメント系の知識獲得が可能となった．

(3)EPM による開発データの自動収集と研究者との共有

リプレイヤは実際のプロジェクトの開発データを自動収集したデータを利用する．そのための自動収集は EPM(Empirical Project Monitor)で行う．開発中のソースコードの変化の記録である CVS データやバグ情報やその修正履歴，さらにメールログ等を自動で収集した．また，研究者が収集した開発データに基づいてシミュレーションモデルを構築するために，収集した開発データを共有するレポジトリを用意し，Web ブラウザから用意にアクセスできる環境を準備した．

(4)ソフトウェアの複雑さの可視化マップ

これまでのリプレイヤはプロジェクト全体の進行の様子を大まかに把握するために役立ったが，個別のモジュール単位の複雑さの進化を表現するには向いていなかった．ユーザの「個別のモジュールの変化を目視確認したい」という要望を満たすために，モジュール結合と論理結合の複雑さの尺度を利用して，ソフトウェアの進化を可視化したマップを作成した．特に，プロダクトの複雑さであるモジュール結合とプロセスの複雑さである論理結合を組み合わせることで，新しいソフトウェアの複雑さの尺度を提案し，ソフトウェアの種類や開発者の個人的な開発方針に影響されない複雑さを計測できた．

(5)プロジェクト学習環境の整備

開発したツール群であるリプレイヤやハイブリッドシミュレータ，さらに情報共有レポジトリや複雑さの可視化マップなど，計算量が大きいツール群を快適に動作させるために新しい３層アーキテクチャに基づくプロジェクト学習環境を整えた．これは Web アプリケーションにてさまざまなデータを参照できる 1 層目，個別のツールを保持する 2 層目，さらに細かなデータを保持する 3 層目に分けて，拡張性が高く，並列処理が可能な実験環境を整備した．これによって，大規模なデータの効率的な保持や高い実行性能を維持することができた．

４．研究成果

本研究の成果はツール群の開発とその実証実験結果である．まず，ツール群を紹介した後にそのツールの実証実験結果を紹介する．

(1) ツール群

①リプレイヤ

図１　リプレイヤのスナップショット

図１に過去のプロジェクトを再現するリプレイヤを示す．リプレイヤは５つのペインを持ち，バグの発生などイベントを示すイベントリストビュー，プログラムモジュールの進捗を示すファイルビュー，ソースコードの累積を示すグラフビュー，メールの履歴を示すメールビュー，メンバ情報を示すメンバービューから構成される．リプレイヤの下部にはタイムバーがあり，再生，早送り，巻き戻し，特定時間へのジャンプなどができる．

②ハイブリッドシミュレータ

図２ハイブリッドシミュレータ

リプレイヤからプロジェクトシミュレータを開発した（図２参照）．過去のプロジェクトを再現しながら，プロジェクトメンバの作業の再割り当てや人員追加できる．

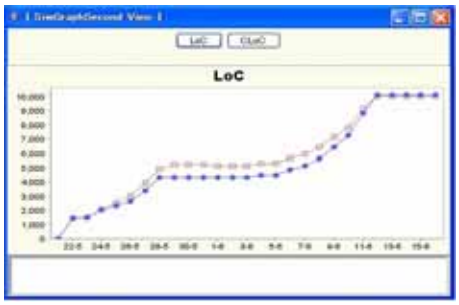

図３　シミュレーションのグラフビュー

図３にハイブリッドシミュレータのシミュレーションのグラフビューを示す．この図では，5 月 24 日に遅れている作業に一人を追加した場合，進捗がどのように変わるかを赤いラインでシミュレーションした例である．遅れた作業に新しい人を追加すると，この場合は当初は少し進捗が回復するが，終盤には他の作業と関係から終了時期は同じになるというシミュレーション結果である．シミュレーションモデルは，実際のプロジェクトから求めた作業分割によるコミュニケーション工数モデルを採用した．

③EPM での開発データの自動収集と情報共有

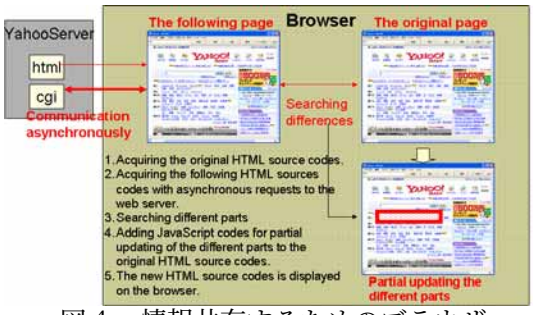

図４　情報共有するためのブラウザ

EPM で自動収集した開発データを研究者と共有するためのレポジトリを作成し，それをストレスなくアクセスするために，Ajax 機能を組み込んだブラウザを開発した（図４参照）．これによって，研究者でも開発者でも Ajax 機能を利用して，随時変化するレポジトリの情報をリアルタイムに取得することが可能となった．

④複雑さの可視化マップ

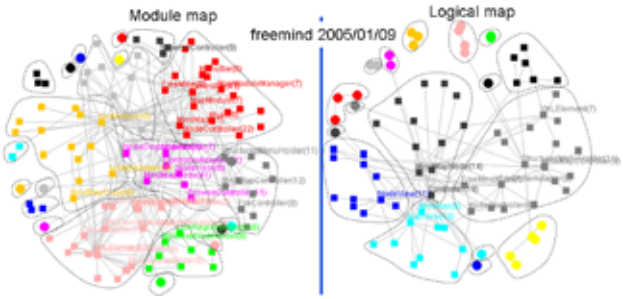

図５　複雑さの可視化マップ

個別のモジュールの複雑さを計測するためにモジュール結合関係と論理結合関係をあらわしたマップを表示するツールを図５に示す．マップが徐々に複雑になり，モジュール数が増えるとともに結合関係が強くなる様子が可視化できた．

⑤プロジェクト学習環境

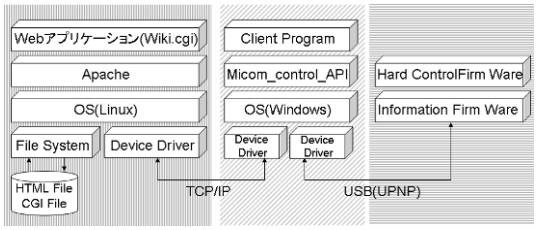

図６　３層アーキテクチャ

大規模な計算を効率的に行うためのプロジェクト学習実験環境を３層アーキテクチャに基づいて構築した．図６は３層アーキテクチャを組み込みシステムへ応用した時のシステム構成図であり，拡張性と汎用性の高いアーキテクチャにてデータ収集とシミュレーションを実現する環境を構築した．

(2) 実証実験結果

① リプレイヤ

リプレイヤにて小規模な学生プロジェクトと大規模な保守フェーズの実プロジェクトを再現した．過去を再現して得られた知識は以下の通りである．

- 画像データ形式の違いによるデータ容量と画像の鮮明度のトレードオフ
- ソースコードのバックアップミスによる修正箇所の喪失
- ソースコードの増加が停滞し，進捗がない期間は，ソースコードの修正箇所の復帰作業が実施され，生産性が一時期非常に落ちた．

② ハイブリッドシミュレータ

ハイブリッドシミュレータは試行段階の実証実験まで実施した．過去のプロジェクトの途中に人員を新しく追加したとき，作業分割によるコミュニケーションモデルに従ってシミュレーションした結果，一時期は進捗が回復するが，結局は終了時期は同じであるというシミュレーションとなった．これによって，安易な人員追加は効果が少ないということが明らかになった．

③ EPM での開発データの自動収集と情報共有

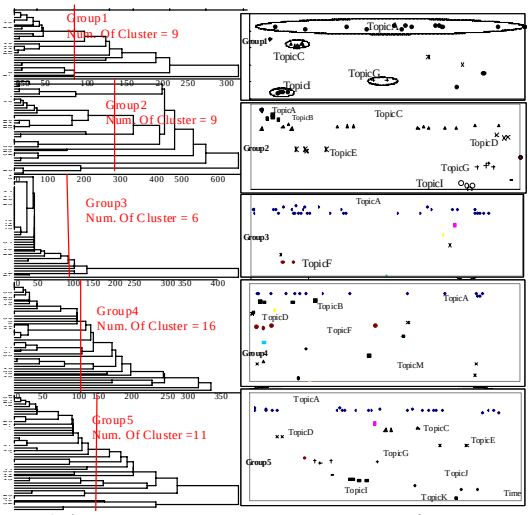

図７　コミュニケーションモデル

ハイブリッドシミュレータはシミュレーションモデルが必要となる．これは様々なプロジェクトの状態を進捗という数値に計算できるモデルである．工数計算の基本的なプロジェクトシミュレーションモデルをベースに，作業分割によるコミュニケーション工数を計算するモデルを新しく追加した（図７参照）．このコミュニケーション工数モデルは，EPM を使って実プロジェクトから自動収集した開発データ（CVS データ）に基づいて構築された．そのために開発プロジェクトからの開発データを蓄積するレポジトリが必要となる．さらに，レポジトリに蓄積されたデータを Web ブラウザで効率よく参照するための Alax を組み込んだブラウザを開発した．これによって，実際のコミュニケーション工数モデルを構築することができた．

④ 複雑さの可視化マップ

リプレイヤやハイブリッドシミュレータはプロジェクト全体の進捗を示すツールであった．複雑さの可視化マップはプログラムの複雑さの変化を可視化するツールであり，ソフトウェア進化を把握することに役立つ．また，モジュール結合と論理結合を組み合わせた新しいソフトウェア開発の複雑さのメトリクスを提案し，オープンソースプロジェクトのデータを EPM のレポジトリに蓄積し解析した結果，プロジェクトごとのソフトウェアの複雑さの変化の特徴が明らかになった（図８参照）．

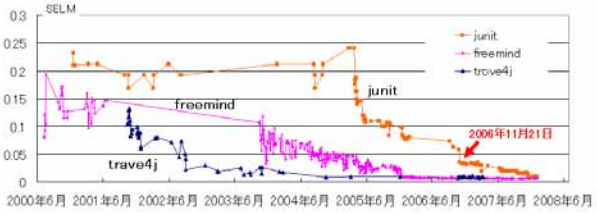

図８　３つのプロジェクトの複雑さの変化

⑤ プロジェクト学習環境

図９　プロジェクト学習環境の PC 群

これらのツールを装備したプロジェクト学習環境は５つの演算用コンピュータと一つのサーバ機と中間にノートコンピュータを配置する形式で構築された．これによってコミュニケーションモデルのシミュレーションや複雑さの計算が高速に行われ，過去のプロジェクトの変化を学習するとともに，シミュレーションを繰り返し実行できた．

５．主な発表論文等

（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計 15 件）

1. 花川典子, 論理結合マップとモジュール結合マップの重なりを用いたソフトウェア進化尺度の提案, コンピュータソフトウェア論文誌, ソフトウェア科学会, 採録決定. 査読有
2. Noriko Hanakawa, “A source-code based extraction way for micro processes influencing software complexity”, Proceeding of the 15th Asia-Pacific Software Engineering Conference, pp.239-246, Dec. 2008. 査読有
3. 尾花将輝, 花川典子：組込み機器を用いた３層 IT アーキテクチャの提案と実装, 第 15 回ソフトウェア工学の基礎ワークショップ(FOSE’08), pp.81-86 , Nov. 2008. 貢献賞受賞, 査読有
4. 花川典子：ソースコードの複雑さに影響を与えるソフトウェア開発マイクロプロセス抽出方法の提案, ソフトウェアエンジニアリング最前線 (社)情報処理学会ソフトウェア工学研究会, pp.95－102, Sep. 2008. 査読有
5. Noriko Hanakawa: An intelligent web browser plug-in for automatic translation to Ajax approach, The International Journal The IPSI BgD Transactions on Internet Research, Volume 3 Number 2 (ISSN 1820-4503), pp.23—31, July(2007). 査読有
6. Noriko Hanakawa, ” Visualization for software evolution based on logical coupling and module coupling”, Proceeding of the 14th Asia-Pacific Software Engineering Conference, pp.214-221, Dec. 2007. 査読有
7. 花川典子：“論理結合とモジュール結合度を用いたソフトウェア進化の可視化ツールの提案”, 第 14 回ソフトウェア工学の基礎ワークショップ(FOSE’07), pp.65-74, Nov. 2007. 貢献賞受賞,査読有
8. Noriko Hanakawa, “A tool-supported environment for knowledge feedback cycle in software development”, Proceeding of the 2nd International workshop on Supporting Knowledge Collaboration in Software Development, pp.33－34, Sep. 2006. 査読有
9. Noriko Hanakawa, Nao Ikemiya, “A new web browser including a transferable function to Ajax codes”, Proceeding of the 21th IEEE/ACM International Conference on Automated Software Engineering, Demo tool session , pp.351－352, Sep. 2006. 査読有
10. Noriko Hanakawa, “A web browser for Ajax approach with asynchronous communication model”, Proceeding of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, Dec. 2006. 査読有
11. Noriko Hanakawa, Mike Barker, “A case study of an empirical approach to component requirements in developing a plagiarism detection tool”, Proceeding of the 13th Asia-Pacific Software Engineering Conference(APSEC2006), pp.353－360, Dec. 2006. 査読有
12. Kimiharu Ohkura, Keita Goto, Noriko Hanakawa, Shinji Kawaguchi, and Hajimu Iida “Project Replayer with Email Analysis – Revealing Contexts in Software Development”, Proceeding of the 13th Asia-Pacific Software Engineering Conference(APSEC2006) , pp.453－460, Dec. 2006. 査読有
13. Keita Goto, Noriko Hanakawa, Hajimu Iida, “Project Replayer-An Investigation Tool to Revisit Process of Past Projects!”, Proceeding of International Software Process Workshop and International Workshop on Software Process Simulation and Modeling, (SPW/ProSim2006), pp.72—79, May 2006. 査読有
14. 大蔵君治, 後藤慶多, 川口真司, 花川典子, 飯田元：“ソフトウェア開発における知識還元のためのプロジェクト再現ツール”, ソフトウェアエンジニアリングシンポジウム 2006（SES2006）, pp.75－78, Oct. 2006. 査読有
15. 花川典子：“実証的アプローチによる要件定義を用いたレポート不正コピー検出ツールの開発”, ソフトウェアエンジニアリングシンポジウム 2006（SES2006）, pp.157－164, Oct. 2006. 査読有

〔学会発表〕（計１件）

1. 尾花将輝, 花川典子：マイコンを利用した３層 IT アーキテクチャ設計の提案 -DollHouse とマイコンによるホームネットワークの実装-, 第 52 回システム制御情報学会研究発表講演会, pp.213-214, May 2008. 京都

〔図書〕（計　５件）

1. ソフトウェア工学の基礎 XV　ソフトウェア科学会 FOSE　2008, 松下誠・川口真司編（執筆担当　組込み機器を用いた３層 IT アーキテクチャの提案と実装, pp.81–86）近代科学社，Nov. 2008.
2. ソフトウェアエンジニアリング最前線 (社)情報処理学会ソフトウェア工学研究会　飯田元・山本里枝子編（執筆担当　ソースコードの複雑さに影響を与えるソフトウェア開発マイクロプロセス抽出方法の提案, pp.95–102）近代科学社, Oct. 2008.
3. ソフトウェア工学の基礎 XIV　ソフトウェア科学会 FOSE　2007, 岸知二・野田夏子編（執筆担当：論理結合とモジュール結合度を用いたソフトウェア進化の可視化ツールの提案，pp.65–74），近代科学社，2007.11.
4. ソフトウェアエンジニアリング最前線 (社)情報処理学会ソフトウェア工学研究会　満田成紀，羽生田栄一編,（執筆担当　“ソフトウェア開発における知識還元のためのプロジェクト再現ツール”, pp.75–78），近代科学社，　Oct. 2006.
5. ソフトウェアエンジニアリング最前線 (社)情報処理学会ソフトウェア工学研究会　満田成紀，羽生田栄一編,（執筆担当　“実証的アプローチによる要件定義を用いたレポート不正コピー検出ツールの開発”，pp.157–164），近代科学社, Oct. 2006.

６．研究組織

(1)研究代表者

花川　典子（HANAKAWA NORIKO）
阪南大学・経営情報学部・教授
研究者番号：60351673

(2)研究分担者

なし

(3)連携研究者

なし