独立行政法人情報処理推進機構 委託

2013 年度ソフトウェア工学分野の先導的研究支援事業「IPA EPM-X の機能拡張によるプロアクティブ型プロジェクトモニタリング環境の構築　―次世代の定量的プロジェクト管理ツールとリポジトリマイニング研究基盤―」成果報告書

平成 27 年 2 月

国立大学法人和歌山大学

本報告書は独立行政法人情報処理推進機構　技術本部　ソフトウェア高信頼化センターが実施した「2013 年度ソフトウェア工学分野の先導的研究支援事業」の公募による採択を受けて国立大学法人和歌山大学システム工学部（研究責任者 和田俊和）が実施した研究の成果をとりまとめたものである.

目次

研究成果概要

１．背景

ソフトウェア開発環境のグローバル化による競争の激化から，開発の短納期化・低コスト化が同時に進むという状況にある．このような過酷な環境下において，ソフトウェア品質の確保や納期を遵守するためには，進行中のプロジェクトをリアルタイムにモニタリング（検出不具合数や工数の進捗具合などの計測と可視化）し，プロジェクト内で発生する問題を早期に検出し，対処する必要がある．ソフトウェア開発状況を定量的に把握しプロジェクト管理を支援するツール EPM(Empirical Project Monitor) [大平 2005, Ohira2004] や EPM-X [EPM-X]は,定量的プロジェクト管理の産業界への普及に重きを置いているため，基本的な定量データ（ソース規模，工数，進捗，品質等）の自動収集と，グラフ化によるプロジェクト管理支援機能が提供されているのみである（課題 A)．基本的にはプロジェクト管理者がグラフ等から問題の発生を目視で発見し対策を講じるというリアクティブなプロジェクト管理にならざるを得ないという問題がある（課題 B)．ソフトウェア開発の現場で求められているのは,プロジェクト内の異変や問題発生の予兆をリアルタイムに検出し，問題の発生を未然に防止するプロアクティブなプロジェクト管理と言える．

２．目的

本研究では，近年ソフトウェア工学分野で進展の目覚ましいリポジトリマイニング技術を定量的プロジェクト管理ツール EPM-X に取り入れること（研究課題 A)，さらに，リポジトリマイニング技術を発展させたプロアクティブマイニング技術を開発する（研究課題 B) ことを研究の達成目標とする．研究課題 A および B の成果を，EPM-X のプラグインとして実装することで，定量的な品質・進捗予測機能とプロアクティブ型のプロジェクト管理機能を実現する．具体的な研究目標は以下の通りである．

- **研究課題 A：リポジトリマイニング・プラグインの開発**
  - (SG-A-1)リポジトリマイニング技術の調査と選定
  - (SG-A-2)リポジトリマイニング・プラグイン 5 件の開発
- **研究課題 B：プロアクティブマイニング・プラグインの開発**
  - (SG-B-1)プロアクティブマイニング技術の調査と選定
  - (SG-B-2)プロアクティブマイニング・プラグイン 3 件の開発
  - (SG-B-3)プロアクティブマイニング・プラグインの有用性検証
  - (SG-B-4)実務者へのヒアリングを通じたニーズ調査および改良点の調査
  - (SG-B-5)プラグイン開発へのヒアリング調査結果の反映

３．実施内容

３．１ 研究全体

海外研究協力者から適宜フィードバックを得ながら，研究課題 A と B を平行して，それぞれ以下の手順で進めた．また，中間報告において得られたフィードバックに基づいて実装を見直した．さらに，複数の実務者へヒアリングをおこないニーズ調査を実施するとともに，プラグインの改良点について意見聴取し，プラグイン開発へフィードバックした．

３．２ 研究課題 A：リポジトリマイニング・プラグインの開発

1. 文献調査：ソフトウェア工学分野におけるリポジトリマイニング技術に関して，トップ会議・トップジャーナルを中心に過去 10 年分の文献を調査した．
2. 技術分類：文献調査に基づき，既存技術を以下の 3 つの観点で分類した．
   （ア）適用ドメイン（適用対象業務ごとに分類）
   （イ）対象リポジトリ（Subversion や Trac など，リポジトリごとに分類）
   （ウ）支援目的（実装支援や保守支援など，支援目的ごとに分類）
3. 技術選定：技術分類に基づいて，検証が十分に行われており，かつ，以下の条件を満たす技術を 5 件，選定した．
   （ア）現場のニーズに合っていること
   （イ）EPM-X と親和性の高いリポジトリを適用対象としていること
   （ウ）プラグインとして実装するコストが大きくなり過ぎないこと
4. 仕様決定：選定した技術の詳細が記述されている文献に基づき，リポジトリマイニングのアルゴリズムおよび入出力のためのデータ構造の仕様を決定した．
5. 実装・テスト：決定した仕様に基づき，プロトタイプとなるプラグインを 1 件実装した．その後，Redmine 版・Trac 版それぞれ 5 件合計 10 件のプラグインを実装する．オープンソースプロジェクトデータを用いて，実装したプラグインが調査した文献に記述されている通りに解析結果を出力できるかどうかをテストした．リポジトリマイニング技術の有効性そのものは検証されたものを利用するため，研究課題 A では検証実験を実施しなかった．

３．３ 研究課題 B：プロアクティブマイニング・プラグインの開発

1. 文献調査：他分野におけるプロアクティブマイニング技術に関して，トップ会議・トップジャーナルを中心に過去 10 年分の文献を調査した．
2. 技術分類：文献調査に基づき，既存技術が扱う問題を以下の 3 つの観点で分類した．
   （ア）外れ値検出問題
   （イ）変化点検出問題
   （ウ）異常状態検出問題
3. 技術選定：技術分類に基づき，ソフトウェア開発の支援に応用可能な技術を 3 件選定した．
4. 仕様決定：仕様の決定については，(研究課題 A-4)と同様に行った．
5. 実装・テスト：実装・テストについては，(研究課題 A-5)と同様に行った．
6. 検証実験：オープンソースプロジェクトデータを用いて，実装したプロアクティブマイニング技術の有効性を検証するための実験を行った．

４ 研究成果

４．１ (SG-A-1)リポジトリマイニング技術の調査と選定

過去 10 年分の文献調査を通じて既存リポジトリマイニング技術を分類し，有用性が高くかつプラグイン化可能なもの 5 件を選定した．リポジトリマイニング技術は，技術の体系化が進まない状態ですでに多数のものが乱立して提案されており，選定にあたってはソフ

トウェア工学におけるトップ会議やトップジャーナルを中心に，リポジトリマイニング技術に関する調査を行った．

また，技術選定にあたっては，門田らの技術解説[門田 13]を参考にして以下の技術的観点を設定し，特定の用途に偏ることがないように注意した．

- プログラミング：ソフトウェア部品検索，コードクローン分析など
- テスト：バグモジュール予測，テスト工数予測など
- 保守：バグ要因分析，エキスパート推薦，バグ特定など
- 管理：ライセンス分析，コミュニケーション分析など
- OSS 利用：品質・成熟度評価，修正時間予測など

以上の調査および技術分類に基づき，実務志向で有用性の高いもの（ただし，研究実施期間中に実現可能性なもの）として 5 種類のリポジトリマイニング技術を選定した．

４．２ (SG-A-2)リポジトリマイニング・プラグイン 5 件の開発

技術調査および選定結果に基づき，リポジトリマイニングのアルゴリズムおよび入出力のためのデータ構造の仕様を決定した．

次に，Redmine 版・Trac 版それぞれ 5 件計 10 件のプラグインを順に実装し，オープンソースプロジェクトデータを用いて，実装したプラグインが調査した文献に記述されている通りに解析結果を出力できるかどうかをテストした．リポジトリマイニング技術の有効性そのものは検証されたものを利用するため検証実験は行わなかった．

４．２．１ タスク担当者推薦プラグイン

（概要）

タスク担当者推薦プラグインは，過去の完了したタスクの情報に基づいて，新規タスクを担当すべき開発者を推薦するプラグインである．タスク担当者推薦プラグインを利用することによって，効率的なタスクの割当てを行うことができるため，タスク完了にかかる時間を短縮することが期待できる．

（アルゴリズム）

タスク担当者推薦プラグインでは，過去の完了したタスクに含まれる文章からキーワードを抽出し，抽出されたキーワードとそのタスクを完了した開発者の関係を学習する．学習に用いる機械学習アルゴリズムはナイーブベイズ法であり，学習データと同様に収集した未完了スクのキーワードのデータをテストデータとすることによって，そのタスクを担当すべき開発者を推薦する[Anvik06][Park11][Xuan12]．

（プラグイン画面）

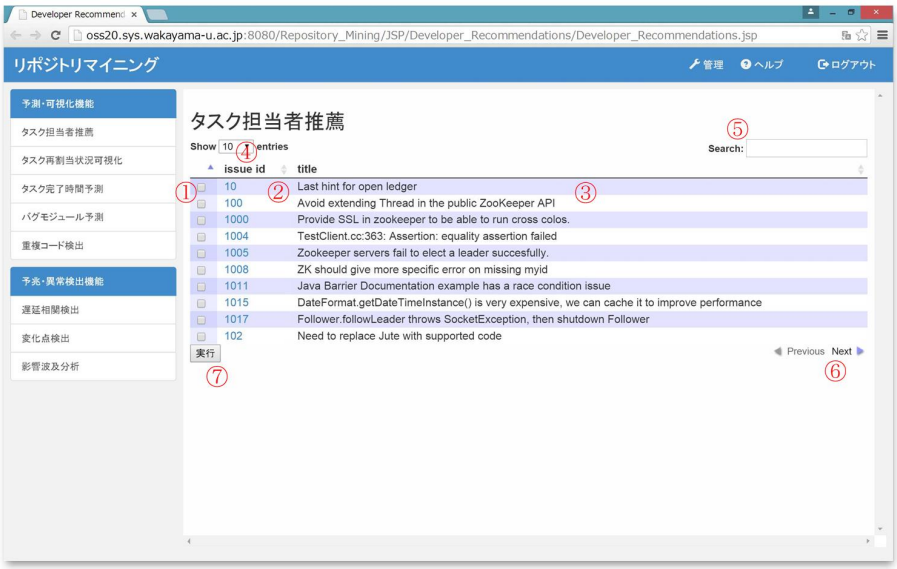


図 1 タスク選択画面（タスク担当者推薦プラグイン）

タスク担当者推薦プラグインを呼び出すと,完了していないタスクが issue_id の昇順で 10 件一覧出力される．図 1 は，タスク一覧の出力例を示している．①は，担当者を推薦したいタスクを選択するチェックボックスである(このチェックボックスは複数選択可能)．②は，タスクの ID を示している．issue_id ボタンを押すことで ID の昇順・降順を変更できる．また，ID のリンクをクリックすることで，その ID のタスクの詳細画面に遷移する．③は，タスクの題名のテキストを示している．title ボタンを押すことで題名の昇順・降順を変更できる．

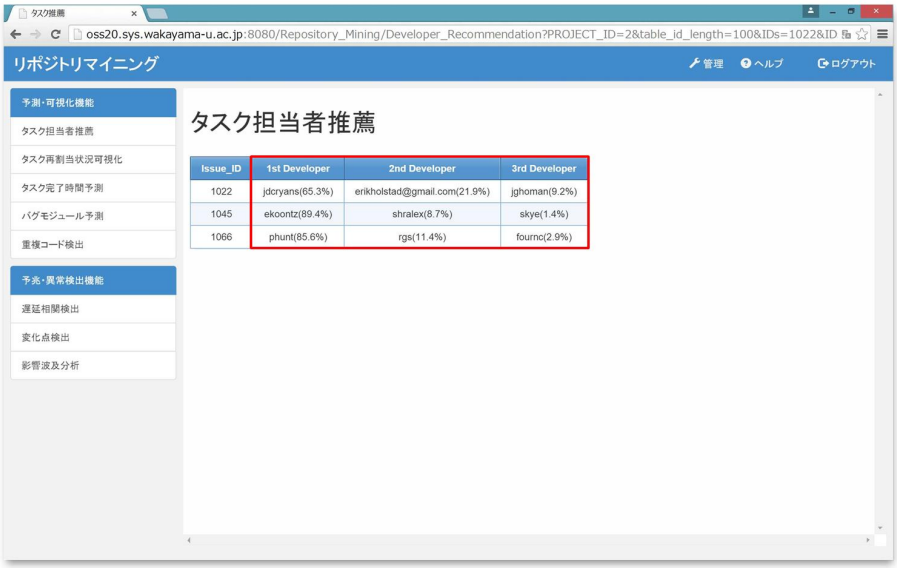


図 2 タスク担当者推薦プラグインの出力結果の例

図 2 に担当者推薦結果の出力画面の例を示す．図 1 に示したタスク選択画面においてチェックボックスでタスクが選択され，実行ボタンを押されるとこの出力画面に遷移する．issue_id に従い昇順で一覧表示される．赤枠部分にタスク選択画面において選択したタスクに対して推薦された担当者が上位 3 名まで表示されている．氏名および適任の度合いを確率としてパーセンテージ表示している．適任の度合いに応じて”1st Developer”，”2nd Developer”，”3rd Developer”の順に表示される．

４．２．２ タスク割当状況可視化プラグイン

（概要）

タスク再割当状況可視化プラグインは，過去のタスクの再割当情報を可視化するプラグインである．タスク再割当状況可視化プラグインを利用することによって，効率的なタスクの割当てを行うことができ，タスク完了にかかる時間を短縮することが期待できる．

（アルゴリズム）

タスク再割当状況可視化プラグインでは，タスクの再割当が発生する回数を減らすために,過去のタスク再割当の情報をマルコフモデルに準ずるネットワーク図上に可視化する．この手法は，Jeong らの研究で提案されたものであり，Tossing Graph と呼ばれる．Tossing Graph を用いることによって，開発者の関係やチームの構造を明らかにすることができることや，よりタスクの完了を見込むことのできる開発者にタスクを割当てることを支援することができる[Jeong09][Bhattacharya10][Bhattacharya12]．

（プラグイン画面）

タスク再割当状況可視化プラグインを呼び出すと，過去のタスクの再割当状況を確率モデルとして出力する．図 3 は，出力結果の例を示している．出力されたグラフにおける円の大きさは，割当てが行われた回数を示しており，円が大きいほど割当てられた回数が多いことを意味している．また，表示されている確率は，ある開発者から他の開発者に対してタスクの割当てが発生した際に，タスクが完了する確率を意味している．

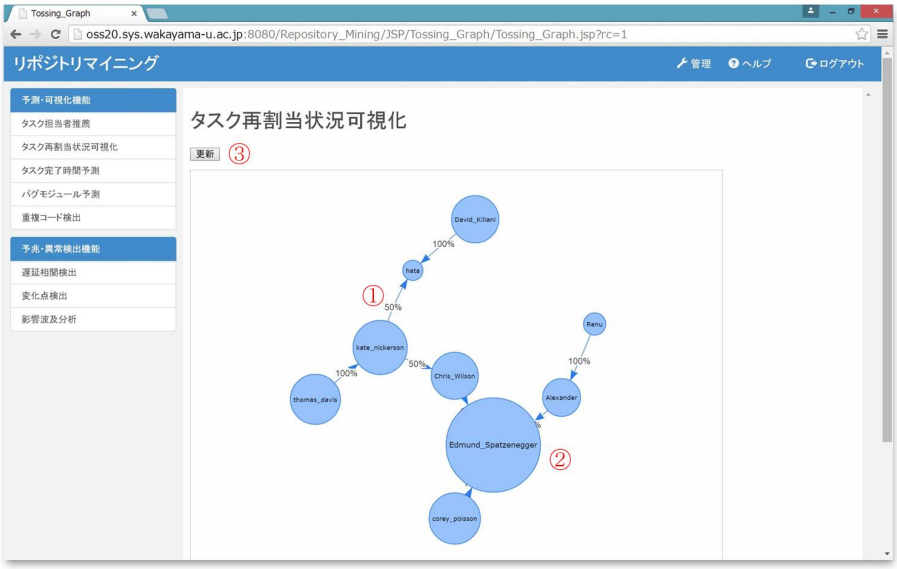


図 3 タスク再割当状況可視化プラグインの出力結果の例

４．２．３ タスク完了時間予測プラグイン

（概要）

タスク完了時間予測プラグインは，過去の完了したタスクの情報に基づいて，新規タスクが完了すると考えられる時間を予測するプラグインである．タスク完了時間予測プラグインを利用することによって，適切なタスクの見積もりを行うことができるため，効率的なタスク管理がおこなえると期待できる．

（アルゴリズム）

タスク完了時間予測プラグインでは，予測モデルを構築するために，不具合管理システムに登録されたチケット（不具合管理票）から抽出した 5 種類のメトリクス（コンポーネント名，優先度，報告者，担当者，概要の文字数）を用いる．指定期間内にタスクが完了するか否かの予測に目的変数を離散値として予測可能なランダムフォレスト法を用いる[Weiss07][Hewett09][Zhang13]．

（プラグイン画面）

図 4 タスク完了時間予測プラグインの出力結果の例

タスク担当者推薦プラグインと同様に，一覧表示されたタスクリストから任意のタスクを選択し実行すると，図 4 のように予測結果が出力される．①は，選択されたタスクの ID を示している．issue_id ボタンを押すことで，ID の昇順・降順を変更できる．②は，①の ID のタスクが完了する期間の予測結果をそれぞれ示している．予測完了時間ボタンを押すことで，時間の昇順・降順を変更できる．

４．２．４ バグモジュール予測プラグイン

（概要）

バグモジュール予測プラグインは，過去の不具合情報から，不具合が存在すると考えられるモジュールを予測するプラグインである．バグモジュール予測プラグインを利用する

ことによって，どのモジュールに不具合が含まれやすいのかを確認することができるため，不具合が含まれると考えられるモジュールに対して，重点的にテストを行うことなどによって，効率的なテストを行うことができる．

（アルゴリズム）

バグを含む可能性の高いファイルに対して重点的にテストを行うために，バグモジュール予測プラグインでは，リポジトリに含まれる対象ファイルすべてのバグを含む可能性を明らかにする[S'liwerski05][Nagappan06][Kim08]．そのため，バグモジュール予測プラグインの入力となるのは，リポジトリのパスであり，バグモジュール予測プラグインの出力となるのは，各ファイルがバグを含む可能性である．ここで，ファイルがバグを含む可能性とは，ファイルにバグが存在する確率であり，0 から 100 のパーセンテージの値で表される．

（プラグイン画面）

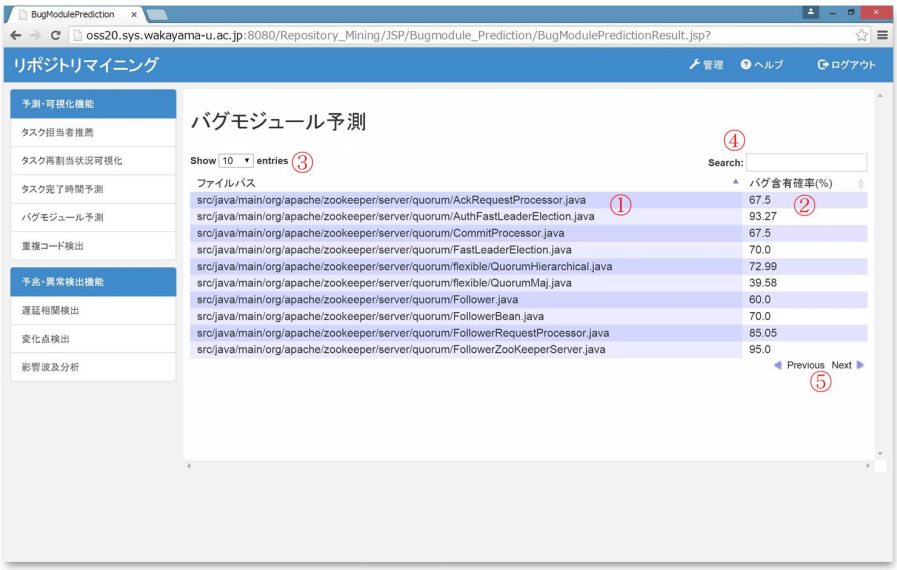


図 5 バグモジュール予測プラグインの出力結果の例

予測結果の出力画面の例を図 5 に示す．ファイルパスの昇順に一覧表示されている．①はファイルパスを示している．ファイルパスボタンを押すことでファイルパスの昇順・降順を変更できる．②は①のファイルにバグが含まれている確率をそれぞれパーセンテージで示している． バグ含有確率(%)ボタンを押すことで確率値の昇順・降順を変更できる．③は，一覧表示件数を変更するドロップダウンボタンである．10 件，25 件，50 件，100 件と変更できる．④は，検索窓である．ファイルパスとバグ含有確率の両方に対して単語や数値を検索できる．⑤は，表示ページの変更ボタンである．

４．２．５ 重複コード検出プラグイン

（概要）

重複コード検出プラグインは，現在開発中のソフトウェアのソースコードに含まれる重複した箇所を抽出するプラグインである．ソースコードに含まれる重複した箇所を知るこ

とは，保守性の高いソフトウェアを目指しリファクタリングを行う際などに役立つ．

（アルゴリズム）

重複コード検出プラグインでは，ソースコードを入力すると，重複するコード片を含むファイルセット，および，重複しているコード片を出力する．解析対象となるソースコードが大規模になればなるほど，重複するコード片のペアを抽出するのに時間がかかるため，単純に文字列を比較するような単一文字列検索アルゴリズムでは，現実的な時間で解析が終了しないと考えられる．そのため，効率的にソースコード中に含まれる文字列を比較するアルゴリズムを用いる必要がある．重複コード検出プラグインでは，ソースコード中に含まれる文字列を比較するアルゴリズムとして，ラビンカープ文字列検索アルゴリズム（ラビンカープ法）を用いる．ラビンカープ法は，ハッシュ関数を利用した文字列検索アルゴリズムである[Ducasse99][Lin14][Mondal14]．

（プラグイン画面）

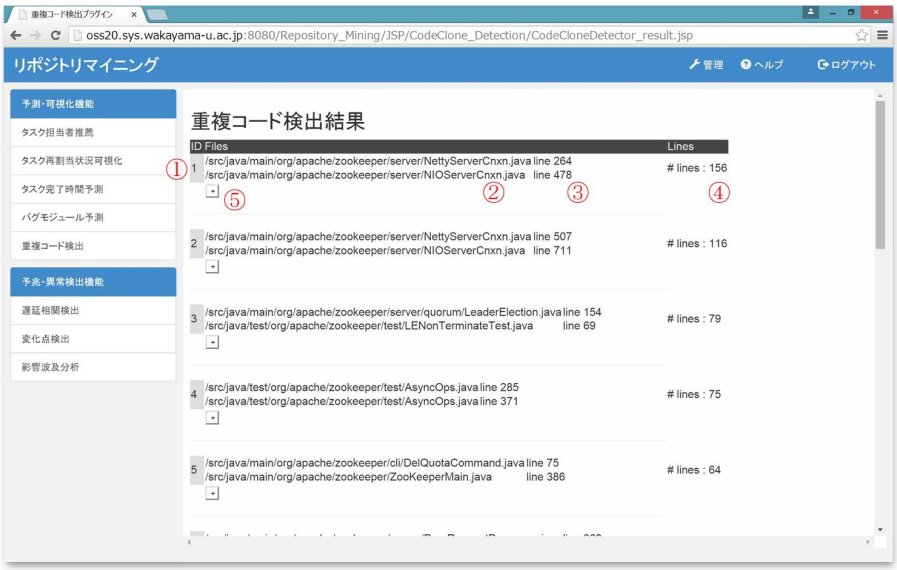


図 6 重複コード検出プラグインの出力結果の例

重複コード検出プラグインを呼び出すと，ソースコードから重複するコード片の情報が図 6 のように出力される．①は，類似するコード片のペアの ID を示している．②は，類似するコード片を含むファイルのファイルパスを示している．③は，それぞれのファイルの類似するコード片を含む箇所が始まる行数を示している．④は，類似する部分のコード行数を示している．⑤は，重複しているコード片の表示を行うボタンである．

４．３ (SG-B-1) プロアクティブマイニング技術の調査と選定

過去 10 年分の文献調査を通じて既存プロアクティブマイニング技術を分類し，有用性が高くかつプラグイン化可能なもの 3 件を選定した．ソフトウェア工学の分野ではほとんど適用事例がないため，他分野の調査を中心に技術の
選定を進めた．主に，以下の国際会議論文集および論文誌を調査した．

- 国際会議論文集

- ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
- SIAM International Conference on Data Mining
- Very Large Data Bases (VLDB) Conference
- ACM Internet Measurement Conference (IMC)

- 論文誌
  - Data Mining and Knowledge Discovery
  - IEEE Transaction on Information Theory
  - Machine Learning

また，技術選定にあたっては，山西の技術解説[山西 09]を参考にして以下の技術的観点からプロアクティブマイニング技術を分類した．

- 外れ値検出
- 変化点検出
- 異常行動検出

以上の調査および技術分類に基づき，実務志向で有用性の高いもの（ただし，研究実施期間中に実現可能性なもの）として以下の 3 種類のプロアクティブマイニング技術を選定した．

1) 遅延相関検出
2) 変化点検出
3) 影響波及分析

４．４ (SG-B-2)プロアクティブマイニング・プラグイン 3 件の開発

技術調査および選定結果に基づき，プロアクティブマイニングのアルゴリズムおよび入出力のためのデータ構造の仕様を決定した．

次に，Redmine 版・Trac 版それぞれ 3 件計 6 件のプラグインを順に実装し，オープンソースプロジェクトデータを用いて，プラグインの有効性を検証した．

４．４．１ 遅延相関検出プラグイン

（概要）

遅延相関検出プラグインは，リポジトリから計測されるメトリクス間の時間のずれる関係を明らかにするプラグインである．メトリクス間の時間のずれる関係とは，事象 A が発生すると一定期間後に事象 B が発生するといった関係を指し，メトリクス間の時間のずれる関係を把握することによって，今後プロジェクトに起こると考えられる事象を知ることができ，ソフトウェア開発・保守の品質および生産性の向上が見込まれる．

（アルゴリズム）

遅延相関検出プラグインは，時間的な遅延を伴う変数間の関係を明らかにする遅延相関分析手法をベースとしたものである．遅延相関分析手法は，一定期間の説明変数の値の変化が一定期間後の目的変数の値に影響を与える，という関係を検出するために相関分析を拡張したものである．遅延相関分析を行うことで，「開発者数が 2 か月多い状態が続くと，2 か月後に完了したタスク数が多くなる」といった関係を抽出できる[竹内 08][Yamatani14][山谷 15]．

（プラグイン画面）

図 7 は遅延相関検出プラグインの出力結果の例を示している．①は説明変数として用いたメトリクスの名称を示している．②は目的変数として用いたメトリクスの名称を示している．③は遅延して現れる関係を示している．④は遅延相関係数を示している．①～④のいずれも，列名のボタンを押すことで，昇順・降順を変更できる．⑤は分析期間を示している．⑥は一覧表示件数を変更するドロップダウンボタンである．10 件，25 件，50 件，100 件と変更できる．⑦は検索窓である．タスクの ID と題名のテキストの両方に対して単語や数値を検索できる．⑧は表示ページの変更ボタンである．

図 7 遅延相関検出プラグインの出力結果の例

４．４．２ 変化点検出プラグイン

（概要）

変化点検出プラグインは，ソフトウェア開発における開発状況の変化を早期に発見するために，変化点検出アルゴリズムに基づいてリポジトリから計測されるメトリクスの値の解析を行う．ソフトウェア開発における開発状況の変化を早期に発見することによって，（潜在的に）重大な問題に素早く対応できるようになると期待できる．

（アルゴリズム）

変化点検出プラグインでは，AR モデルのオンライン忘却型学習アルゴリズム SDAR を用いた時系列モデルの 2 段階学習に基づいた変化点検出アルゴリズムを用いる．時系列データが入力されると，第一段階の学習を行う．第一段階の学習では，AR モデルで予測される値と実際の値との差分である外れ値スコアを求める．次に，この外れ値スコアを平滑化する．平滑化とは，直近の数時点の外れ値スコアを平均することである．この平滑化によって，ノイズに反応した外れ値スコアを除去する．次に，平滑化されたスコアの時系列データに対して二段階目の学習を行い，外れ値スコアをもう一度求める．そして最後に，平滑後の外れ値スコアを再度平滑化して，最終的な各時点における変化点スコアを求める．平滑化によりノイズを除去することや，二段階学習をすることによって，本質的な変動のみを検出できる[Robles06][山西 09][久木田 15]．

（プラグイン画面）

図 8 変化点検出プラグインの出力結果の例

図 8 に変化点検出プラグインの出力結果の例を示す．選択したメトリクスの時系列データや変化点スコアを確認することができる．赤枠の中の赤い線が選択したメトリクスの時系列データあり，青の線が変化点スコアとなっている．

４．４．３ 影響波及分析プラグイン

（概要）

影響波及分析プラグインは，対象リポジトリに含まれるファイルを選択すると，同時に変更されたことのあるファイルおよび，その確率，およびファイルを修正するのにこれまでかかった時間が出力される．あるファイルを修正しようと考えている開発者は，修正にかかるコストを容易に見積もることができる．

（アルゴリズム）

影響波及分析プラグインでは，あるファイルを選択すると，同時に変更が行われる，論理的結合関係にあるファイルを変更履歴から抽出し，出力することができる．また，それらのファイルを修正するためにかかるコストも算出することができる[Zimmermann05][Acharya11]．そのためには，ソースコードの変更履歴と不具合修正履歴データの紐付けが必要となる．不具合修正履歴情報とソースコードの変更履歴情報をひも付ける作業は，SZZ アルゴリズムを用いた[Sliwerski05]．

（プラグイン画面）

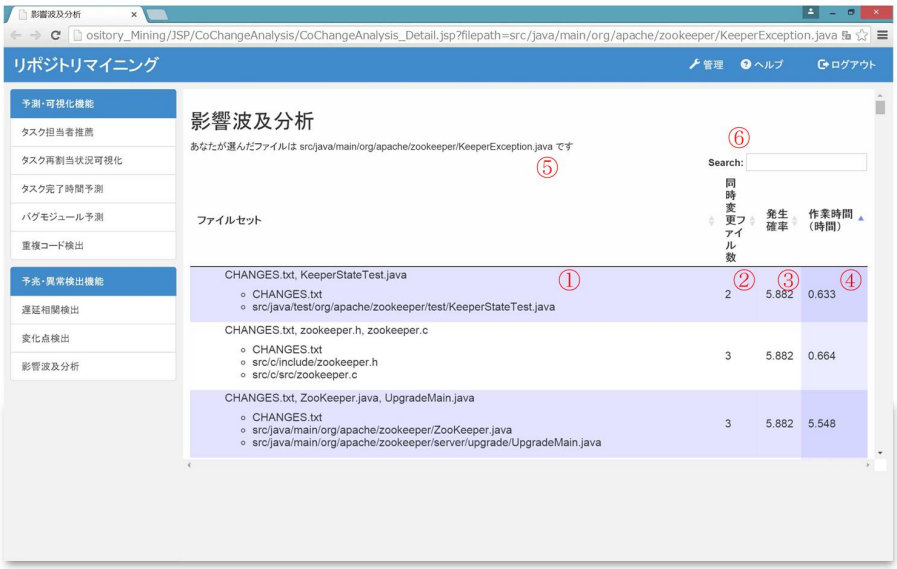


図 9 影響波及分析プラグインの出力結果の例

図 9 に影響波及分析プラグインの出力結果の例を示す．①はそれぞれ，同時変更されたファイルセットを示している．②は同時変更ファイル数を示している．③は同時変更の発生確率を示している．④はファイルセット全体の変更にかかったトータル作業時間の平均値を示している．①～④のいずれも，列名のボタンを押すことで，昇順・降順を変更できる．⑤はファイル一覧画面で選択したファイルのファイルパスを示している．⑥は検索窓である．すべてのカテゴリに対して単語や数値を検索できる．

４．５ (SG-B-3) プロアクティブマイニング・プラグインの有用性検証

オープンソースプロジェクトデータを用いて，実装したプロアクティブマイニング技術の有効性を検証するための実験を行った．なお，4.6 節で述べるように，影響波及分析プラグインは，実務者へのヒアリングから判明したニーズを踏まえて既存技術[Zimmermann05]を実装したものであるため，検証実験は行わずテストのみ実施した．

４．５．１ 遅延相関検出プラグインの検証実験

（概要）

実装した遅延相関検出プラグインの実践的な適用を想定して，Eclipse Platform プロジェクトを対象としたケーススタディを通じてプラグインの検証を行った．ケーススタディでは，相関分析のみを用いた従来の分析方法よりも有用性の高い分析が行えることを目標とした．2003 年 7 月から 2012 年 6 月までの約 10 年に渡るプロジェクトデータおよび 131 種類のメトリクスを用いて，抽出可能な関係を詳細に分析した．

（結果）

時間的順序関係を考慮しない通常の相関分析では，8,515 通りの相関を求めることができる．一方，遅延相関検出手法では，17,030 通り（${}_{131}P_2$）×12（加算係数）×13（遅延係数）通りの相関を求めることができる．組み合わせが膨大なため，相関係数 0.4 以上のも

ののみを抽出した結果，171 件に絞ることができた．171 件のうち，通常の相関分析では抽出できない関係が 31 件存在した．31 件の関係を詳細に分析した結果，「不具合の修正に多くの時間を要するようになると，3 ヶ月後に優先度の高い不具合数が増加する」といった時間的遅延を伴う関係が抽出できており，その原因理解を支援することができることがわかった．また，原因理解の支援により，望まない時間的関係が存在する場合はあらかじめ有効な予防策を講じることができるようになることがわかった[山谷 15]．

４．５．２変化点検出プラグインの検証実験

（概要）

実装した変化点検出プラグインの有効性を検証するために，Eclipse Platform プロジェクトの版管理システム Git，ニュースフォーラム Eclipse News Forums，不具合管理システムからそれぞれデータソースを抽出した．2002 年 1 月から 2012 年 12 月までの約 19 年にわたるデータを用いて，変化点スコアと 3 種類のメトクリク値（修正待ち不具合数，コード行数，サイクロマティック数の平均）との関係を詳細に分析した．

（結果）

分析の結果,図 10 のように各メトリクスの急激な変化に追従するように変化点スコアが大きな値をとることがわかり,期待した通りの出力が得られていることがわかった.また，大きな変化点スコアをとる前後の時期に対する LDA の結果（トピックの変化）から，急激な変化が起きた理由をある程度推定できることがわかった[久木田 15]．

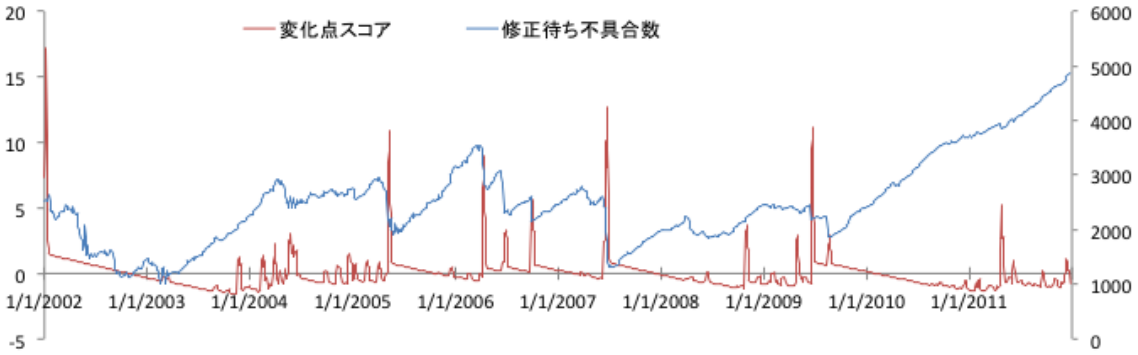


図 10 変化点検出の結果（修正待ち不具合数の時系列変化の例）

４．６ (SG-B-4)実務者へのヒアリングを通じたニーズ調査および改良点の調査

2014 年 1 月 29 日の中間報告後の RISE 委員会からの評価を受けて，実務者へのヒアリングを実施した．2014 年 4 月時点で完成しているプラグインに対する機能面（現場での利用状況に即した物になっているかなど）と使用性（インタフェースの使い勝手など）について，企業の実務担当者 5 名に意見を伺った．また，今後プラグイン化すべき技術についても議論し，現場ニーズを調査した．

ヒアリング調査の結果，既存プラグインの機能面・使用性については概ねポジティブな意見が得られたものの，すべての実務者の現場においてニーズがある訳ではないことがわかった．さらに，影響波及分析[Zimmermann05]については，多くの現場において強いニーズが存在することがわかった．

４．７ (SG-B-5)プラグイン開発へのヒアリング調査結果の反映

ヒアリング調査で得られた知見は，既存プラグインの機能追加およびインタフェースの改良のヒントとしてプラグイン開発へ反映した．また，ヒアリング調査前まではプロアクティブプラグインの１つとして異常行動検出のアルゴリズムを実装予定であったが，ニーズ調査により判明した影響波及分析を実装することとした．

５ まとめ

本研究では，近年ソフトウェア工学分野で進展の目覚ましいリポジトリマイニング技術を定量的プロジェクト管理ツール EPM-X に取り入れること，さらに，リポジトリマイニング技術を発展させたプロアクティブマイニング技術を開発することを研究の達成目標とし，Redmine 版・Trac 版それぞれに対して以下のプラグインを開発した．

- リポジトリマイニング技術
  - タスク担当者推薦
  - タスク割当状況可視化
  - タスク完了時間予測
  - バグモジュール予測
  - 重複コード検出
- プロアクティブマイニング技術
  - 遅延相関検出
  - 変化点検出
  - 影響波及分析

特に，本研究が終盤に差しかかりプラグインの数が増えるに従って，多くの方から利用希望や新規プラグインの開発希望などを頂いた．今後は，和歌山大学システム工学部ソフトウェアエンジニアリンググループ（大平研究室）の Web サイトにて，プラグイン配布のための Web サイトを公開（2015 年 2 月下旬公開予定）し，プラグインの普及活動を進めるとともに，産学連携の加速へ向けた取り組みを充実させていく予定である．

1 研究の背景および目的

1.1 背景

社会基盤として高度なソフトウェアシステムが広く普及した昨今，交通システムや金融システムの障害による業務・サービスの停止や中断は，我々の社会経済活動に多大なる損失を与えるようになっている．社会活動におけるソフトウェアシステムの重要性が今後も増々高まると予想される一方で，ソフトウェア開発環境のグローバル化による競争の激化から，開発の短納期化・低コスト化が同時に進むという状況にある．このような過酷な環境下において，ソフトウェア品質の確保や納期を遵守するためには，進行中のプロジェクトをリアルタイムにモニタリング（検出不具合数や工数の進捗具合などの計測と可視化）し，プロジェクト内で発生する問題を早期に検出し，対処する必要がある．

このような社会的要請に対する IPA(独立行政法人情報処理推進機構 技術本部 ソフトウェア高信頼化センター)の取り組みとして，ソフトウェア開発状況を定量的に把握しプロジェクト管理を支援するツール EPM-X [EPM-X](図 1-1)がある．EPM-X は，文部科学省のリーディングプロジェクト「e-Society 基盤ソフトウェアの総合開発」の一環の EASE(ソフトウェア工学へのエンピリカルアプローチ)プロジェクトにおいて開発された，EPM(Empirical Project Monitor) [大平 2005, Ohira2004] をモデルとしている．

ただし，現状の EPM-X は，定量的プロジェクト管理の産業界への普及に重きを置いているため，基本的な定量データ（ソース規模，工数，進捗，品質等）の自動収集と，グラフ化によるプロジェクト管理支援機能が提供されているのみである（課題 A)．したがって，基本的にはプロジェクト管理者がグラフ等から問題の発生を目視で発見し対策を講じるというリアクティブなプロジェクト管理にならざるを得ないという問題がある（課題 B)．前述したように，昨今の厳しい開発環境下においてソフトウェア開発の現場で求められているのは，プロジェクト内の異変や問題発生の予兆をリアルタイムに検出し，問題の発生を未然に防止するプロアクティブなプロジェクト管理と言える．

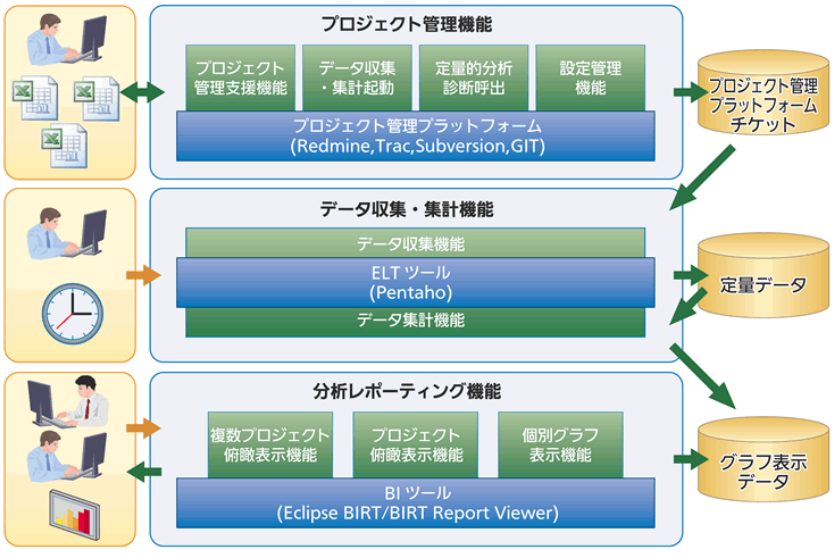


図 1-1 定量的プロジェクト管理ツール EPM-X の概要図（IPA HP より転載）

1.2 研究課題

本研究では，近年ソフトウェア工学分野で進展の目覚ましいリポジトリマイニング技術を定量的プロジェクト管理ツール EPM-X に取り入れること（研究課題 A），さらに，リポジトリマイニング技術を発展させたプロアクティブマイニング技術を開発する（研究課題 B）ことを研究の達成目標とする．研究課題 A および B の成果を，EPM-X のプラグインとして実装することで，定量的な品質・進捗予測機能とプロアクティブ型のプロジェクト管理機能を実現する．

研究目標の設定にあたって前提とした仮説は以下の通りである．

- (H-A-1) 文献調査により広く有用なリポジトリマイニング技術を明らかにすることができる．
- (H-A-2) 有用なリポジトリマイニング技術を EPM-X のプラグインとして実装することができる．
- (H-B-1) 文献調査により広く有用なプロアクティブマイニング技術を明らかにすることができる．
- (H-B-2) 有用なプロアクティブマイニング技術を EPM-X のプラグインとして実装することができる．
- (H-B-3) 実装したプロアクティブマイニング技術の有用性を示すことができる．

これらの仮説を確かめるために設定した具体的な研究課題は以下の通りである．

- **研究課題 A：リポジトリマイニング・プラグインの開発**
  - (SG-A-1)リポジトリマイニング技術の調査と選定
  - (SG-A-2)リポジトリマイニング・プラグイン 5 件の開発
- **研究課題 B：プロアクティブマイニング・プラグインの開発**
  - (SG-B-1)プロアクティブマイニング技術の調査と選定
  - (SG-B-2)プロアクティブマイニング・プラグイン 3 件の開発
  - (SG-B-3)プロアクティブマイニング・プラグインの有用性検証
  - (SG-B-4)実務者へのヒアリングを通じたニーズ調査および改良点の調査
  - (SG-B-5)プラグイン開発へのヒアリング調査結果の反映

1.2.1 研究課題 A：リポジトリマイニング・プラグインの開発

リポジトリマイニングとは，構成管理ツール（Subversion や Git など），プロジェクト管理ツール（Trac や Redmine など）および不具合管理ツール（Bugzilla など）に蓄積された膨大な開発履歴データベース（リポジトリ）に対するデータマイニング技術の総称であり，ソフトウェア開発に有益な知見を過去の定量データに基づいてフィードバックすることを目的としている．EPM-X においても上述のリポジトリが利用されており，適用への親和性が非常に高い技術である．

ソフトウェア工学におけるトップ会議やトップジャーナルである ICSE（International Conference on Software Engineering），FSE（Foundation of Software Engineering），TSE（IEEE Transaction on Software Engineering），ESEM（Journal of Empirical Software Engineering)などでは，若手研究者が次々と斬新なリポジトリマイニング技術を発表しており，近年最も発展の目覚ましい研究分野の 1 つとなっている．また，専門会議である MSR

(Mining Software Repository)は発足から 10 年が経過し，様々な技術が出揃ってきた段階と言える．ただし，研究の急速な進展により，少数のプロジェクトで検証しただけのものから，実務者が技術を実装し現場へ適用するには複雑すぎるものまで，多種多様な技術が乱立して提案されている状態であり，EPM-X のプラグインとして実装すべき技術を多くの開発現場にとって有益となるように選択するのは容易ではない．そこで本研究では，以下のようにサブ課題を設定し研究を進めた．

(SG-A-1)リポジトリマイニング技術の調査と選定

リポジトリマイニング技術は，大きく分類して以下のカテゴリのものが提案されている[門田 14]．

- プログラミング支援（ソフトウェア部品検索，コードクローン分析など）
- テスト支援（バグモジュール予測，テスト工数予測など）
- 保守・デバッグ支援（バグ要因分析，エキスパート推薦，バグ特定など）
- 管理支援（ライセンス分析，コミュニケーション分析など）
- OSS 利用ベンダ支援（品質・成熟度評価，修正時間予測など）

これら既存技術をトップ会議・トップジャーナルの中から幅広く文献調査し，

- ✧ 現場のニーズに合っているかどうか
- ✧ EPM-X と親和性の高いリポジトリが適用対象となるかどうか
- ✧ プラグインとして実装するコストが大きくなり過ぎないかどうか(*)

を調査した．また，調査に当たっては，研究メンバーの国際ネットワークを駆使し，第一線の海外研究協力者とメール・スカイプ・国際会議でのミーティング等を通じてアドバイスを得た．

(*) 本研究ではプラグイン1件（Redmine版・Trac版の計2件）の開発に，実装・テストそれぞれに1ヶ月，計2ヵ月以内での完了を想定した．

(SG-A-2)リポジトリマイニング・プラグイン 5 件の開発

研究課題(SG-A-1)で得られた調査結果に基づいて，実務志向で有用性の高いもの（ただし，研究実施期間中に実現可能性なもの）として以下の 5 種類のリポジトリマイニング技術を選定し，プラグイン化した．EPM-X は，Redmine 版および Trac 版が公開されているため，本研究では，Redmine 版・Trac 版それぞれに 5 種類のプラグイン（計 10 件のプラグイン）を開発した．

- タスク担当者推薦[Anvik06][Park11][Xuan12]
- タスク再割当状況可視化[Jeong09][Bhattacharya10] [Bhattacharya12]
- タスク完了時間予測[Weiss07][Hewett09][Zhang13]
- バグモジュール予測[S ´liwerski05][Nagappan06][Kim08]
- 重複コード検出[Ducasse99][Lin14][Monda114]

1.2.2 研究課題 B：プロアクティブマイニング・プラグインの開発

リポジトリマイニング技術は，蓄積された膨大なプロジェクトデータに基づいて次期プロジェクトの工数を予測したり，ある程度開発が進んだ後にバグが多く含まれているモジュールを予測する，などの用途に向いているが，進行するプロジェクトをリアルタイムに

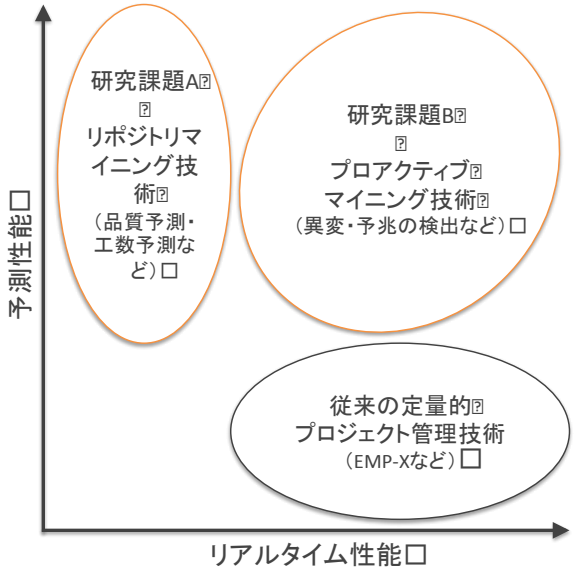


図 1-2 本研究のスコープと従来技術との違い

支援するためのものではない.

一方, EPM-X を代表とする従来の定量的プロジェクト管理技術は, プロジェクトの進捗をリアルタイムに把握するなどの用途に向くが, 問題が発生したことをできる限り早く察知するためのものであり, プロジェクト管理者は問題を認識してからリアクティブに対応せざるを得ない.

そこで研究課題 B では, 従来のリポジトリマイニング手法を発展させたプロアクティブマイニング技術を開発する. 本技術は, ビジネスインテリジェンス (BI) 技術の対比として, ソフトウェアインテリジェンス (SI) とも呼ばれる. しかしながら, SI の概念自体は Ahmed Hassan 氏 (クイーンズ大学, カナダ) および Tao Xie 氏 (イリノイ大学, 米国) によって提唱されている[Hassan2010]ものの, ソフトウェア工学の分野において確立した技術は現在のところほとんどない. 図 1-2 に示すように, 本研究により, 定量的プロジェクト管理技術のリアルタイム性能および予測性能の大きな向上を期待できる.

研究課題 B で開発するプロアクティブマイニング技術は, リポジトリマイニングの研究分野においても新規性の高い技術であるため, EPM-X のプラグインとしてそのまま実装できるようなものは存在しない. そこで本研究では, 以下のようにサブ課題を設定し, 十分な調査に基づいて研究を進めた.

(SG-B-1) プロアクティブマイニング技術の調査と選定

過去 10 年分の文献調査を通じて既存プロアクティブマイニング技術を分類し, 有用性が高くかつプラグイン化可能なもの 3 件を選定した. ソフトウェア工学の分野ではほとんど適用事例がないため, 他分野の調査を中心に技術の選定を進めた.

また, 技術選定にあたっては, 山西の技術解説[山西 09]を参考にして以下の技術的観点からプロアクティブマイニング技術を分類した.

- 外れ値検出[武田 00][Yamanishi04][Neal99]
- 変化点検出[赤池 94, 赤池 95][Yamanishi02][Takeuchi06]

- 異常行動検出[Baum70][Krichevsky81][Rissanen84][Viterbi06][Yamanishi07]

以上の調査および技術分類に基づき，実務志向で有用性の高いもの（ただし，研究実施期間中に実現可能性なもの）として 3 種類のプロアクティブマイニング技術を選定した.

(SG-B-2)プロアクティブマイニング・プラグイン 3 件の開発

研究課題(SG-B-1)で得られた調査結果に基づいて，実務志向で有用性の高いもの（ただし，研究実施期間中に実現可能性なもの）として以下の 3 種類のプロアクティブマイニング技術を選定し，プラグイン化した（Redmine 版・Trac 版それぞれ 3 種類計 6 件).

- 遅延相関検出[竹内 08][Yamatani14][山谷 15]
- 変化点検出[Robles06][山西 09][久木田 15]
- 影響波及分析[S ´liwerski05][Zimmermann 05][Acharya11]

(SG-B-3)プロアクティブマイニング・プラグインの有用性検証

実装したプロアクティブ技術は，ソフトウェア工学分野の適用事例がないことから，プラグインの実践的な適用を想定して，オープンソースプロジェクト（Eclipse Platform プロジェクト）を対象としたケーススタディを通じてプラグインの検証を行った．遅延相関検出プラグインに関しては，相関分析のみを用いた従来の分析方法よりも有用性の高い分析が行えることが示された[山谷 15]．変化点検出プラグインに関しては，各メトリクスの急激な変化に追従するように変化点スコアが大きな値をとることがわかり，期待した通りの出力が得られていることがわかった．また，大きな変化点スコアをとる前後の時期に対する LDA（Latent Dirichlet Allocation)の結果（トピックの変化）から，急激な変化が起きた理由をある程度推定できることがわかった[久木田 15]．影響波及分析プラグインに関しては，(SG-B-4)の実務者へのヒアリングを通じたニーズ調査により，ソフトウェア工学分野での定期用例が存在する技術[Zimmerman05]を実装したため,検証実験は行わなかった.

(SG-B-4)実務者へのヒアリングを通じたニーズ調査および改良点の調査

2014 年 1 月 29 日の中間報告後の RISE 委員会からの評価を受けて，実務者へのヒアリングを実施した．2014 年 4 月時点で完成しているプロアクティブマイニングプラグインに対する機能面（現場での利用状況に即した物になっているかなど）と使用性（インタフェースの使い勝手など）について，企業の実務担当者 5 名に意見を伺った．また，今後プラグイン化すべき技術についても議論し，現場ニーズを調査した.

ヒアリング調査の結果，既存プラグインの機能面・使用性については概ねポジティブな意見が得られたものの，すべての実務者の現場においてニーズがある訳ではないことがわかった．さらに，影響波及分析[Zimmermann05]については，多くの現場において強いニーズが存在することがわかった.

(SG-B-5)プラグイン開発へのヒアリング調査結果の反映

ヒアリング調査で得られた知見は，既存プラグインの機能追加およびインタフェースの改良のヒントとしてプラグイン開発へ反映した．また，ヒアリング調査前まではプロアクティブプラグインの 1 つとして異常行動検出のアルゴリズムを実装予定であったが，ニー

ズ調査により判明した影響波及分析を実装することとした.

1.3 研究の意義

リポジトリマイニング技術は，開発過程で自動的に蓄積されるデータに基づくものであり，定量的プロジェクト管理におけるデータ収集のコストを大幅に削減することを可能にする．また，近年最も研究が盛んに進んでいる研究分野であり多種多様な技術が提案されているため，定量的プロジェクト管理を支援する強力な手段となり得る．しかしながら，提案されている多数の技術の中から一般性が高くかつ有用なものを選定する必要がある，データマイニングおよび統計学の知識がある程度必要なため実装のコストが比較的高い，などの理由から，開発現場において十分には普及していないのが現状である.

EPM-X は，定量的プロジェクト管理のための環境を「手軽に」構築できる点に優位性があるものの，リポジトリマイニングのような高度なデータマイニング機能が利用できない点，基本的にはリアクティブなプロジェクト管理のみをターゲットとしている点から，競争や変化の激しい分野でのプロジェクト管理ツールとしては機能的に限界がある．欧米ではソフトウェ開発における大規模データの活用がすでに大企業を中心にはじまっており，現状のままでは我が国のソフトウェア開発は世界に遅れをとることになりかねない.

本研究は，EPM-X の機能拡張として，有用なリポジトリマイニング技術をプラグイン化することと，プロアクティブなプロジェクト管理を支援するためのプロアクティブマイニング技術をプラグイン化することにより，次世代の定量的プロジェクト管理ツールの構築と普及をめざすものであり，ビッグデータ時代のソフトウェア開発に対する社会的要請に応えるものである.

本研究で開発するリポジトリマイニング・プラグインおよびプロアクティブマイニング・プラグインの普及により，我々は，以下のような効果がソフトウェア開発現場にもたらされるものと期待している.

- 定量的プロジェクト管理における高度データマイニング技術の利用

  IPA EPM-X のプラグインとしてリポジトリマイニング技術を提供することにより，高度な予測機能を「手軽に」利用できるようになる．特に，テスト・保守工程においてのリソース割当の最適化に役立つものと期待される.

- プロアクティブなプロジェクト管理の実現

  異常／予兆検知技術をベースとしたプロアクティブマイニング技術によって，問題の発生を未然に防止するプロアクティブなプロジェクト管理が実施できるようになる．特に，プロジェクト内に発生する品質管理上の異常を早期に検出することが可能となると期待される.

- 新たなニーズ獲得と産学連携

  本研究によりリポジトリマイニング技術およびプロアクティブマイニング技術を手軽に利用することが可能になる．学術論文等の資料を読むのではなく，具体的な形でマイニング技術の有用性を試すことができるため，既存プラグインの改良要望や新たなプラグインの開発要望を実務者が出しやすくなるものと思われる．現状では現場問題を既存技術で解決可能であっても，産学の間の連携が十分ではなく，未解決な場合が多い．特に大学研究者が現場ニーズを正しく理解するのは困難な状況にあるため，

具体的な要望があれば，大学研究者が研究中の技術や既存技術の存在を開発現場にフィードバックできる．

2 実施内容

2.1 研究アプローチ

2.1.1 研究の全体像

海外研究協力者から適宜フィードバックを得ながら，研究課題 A（リポジトリマイニング・プラグインの開発）と研究課題 B（プロアクティブマイニング・プラグインの開発）を平行して進めた．具体的には，以下のような手順で研究を進めた．なお，中間報告において得られたフィードバックに基づいて実装の見直しも適宜行った．

研究課題 A：リポジトリマイニング・プラグインの開発

(A-1) 文献調査：ソフトウェア工学分野におけるリポジトリマイニング技術に関して，トップ会議・トップジャーナルを中心に過去 10 年分の文献を調査する．

(A-2) 技術分類:文献調査に基づき,既存技術を以下の 3 つの観点で分類する．

- 適用ドメイン（適用対象業務ごとに分類）
- 対象リポジトリ(Subversion や Trac など,リポジトリごとに分類)
- 支援目的（実装支援や保守支援など，支援目的ごとに分類）

(A-3) 技術選定：技術分類に基づいて，検証が十分に行われており，かつ，以下の条件を満たす技術を 5 件，選定する．

- 現場のニーズに合っていること
- EPM-X と親和性の高いリポジトリを適用対象としていること
- プラグインとして実装するコストが大きくなり過ぎないこと

(A-4) 仕様決定：選定した技術の詳細が記述されている文献に基づき，リポジトリマイニングのアルゴリズムおよび入出力のためのデータ構造の仕様を決定する．

(A-5) 実装・テスト：決定した仕様に基づき，プロトタイプとなるプラグインを 1 件実装する．その後，Redmine 版・Trac 版それぞれ 5 件合計 10 件のプラグインを実装する．オープンソースプロジェクトデータを用いて，実装したプラグインが調査した文献に記述されている通りに解析結果を出力できるかどうかをテストする．リポジトリマイニング技術の有効性そのものは検証されたものを利用するため，研究課題 A では検証実験は行わない．なお，中間報告において得られたフィードバックに基づいて実装の見直しも適宜行う．

研究課題 B：プロアクティブマイニング・プラグインの開発

(B-1) 文献調査：他分野におけるプロアクティブマイニング技術に関して，トップ会議･トップジャーナルを中心に過去10年分の文献を調査する．

(B-2) 技術分類：文献調査に基づき，既存技術が扱う問題を以下の 3 つの観点で分類する．

- 外れ値検出問題

・ 変化点検出問題
・ 異常状態検出問題

(B-3) 技術選定：技術分類に基づき，ソフトウェア開発の支援に応用可能な技術を 3 件選定する．

(B-4) 仕様決定：仕様の決定については，(A-4)と同様に行う．

(B-5) 実装・テスト：実装・テストについては，(A-5)と同様に行う．

(B-6) 検証実験：オープンソースプロジェクトデータを用いて，実装したプロアクティブマイニング技術の有効性を検証するための実験を行う．例えば，プロジェクト進捗が停滞する予兆を早期あるいはリアルタイムに検出できるかどうかなどについて検証を行う．なお，中間報告において得られたフィードバックに基づいて実装の見直しも適宜行う．

2.1.2 関連するこれまでの研究について

当該委託研究以前に実施されていた関連する研究テーマについて

当該委託研究以前に実施されていた研究のうち，本研究全体の到達目標と直接的に関連しているものは我々の知る限り存在しない．ただし，本研究で参考になる個別の要素技術は多数存在しており，中でも以下のような技術が参考になった．

研究課題 A（ソフトウェア工学分野におけるリポジトリマイニング技術[門田 14]）

- プログラミング支援（ソフトウェア部品検索，コードクローン分析など）
  - テスト支援（バグモジュール予測，テスト工数予測など）
  - 保守・デバッグ支援（バグ要因分析，エキスパート推薦，バグ特定など）
  - 管理支援（ライセンス分析，コミュニケーション分析など）
  - OSS 利用ベンダ支援（品質・成熟度評価，修正時間予測など）
  - プログラミング支援（ソフトウェア部品検索，コードクローン分析など）

- 研究課題 B（他分野における予兆検出技術[山西 09]）
  - プラント故障の予兆検出技術（早期の設備点検）
  - 計算機システムの障害予兆検知技術（早期の部品交換）
  - ネットワーク攻撃検出（不正アクセスの防止）
  - 渋滞予兆検知技術（渋滞の回避）
  - 入眠予兆検出技術（居眠り運転の防止）

研究責任者が当該委託研究以前に実施していた研究と当該委託研究との関係について

研究責任者はパターン認識および信号処理，コンピュータ・ビジョンの研究に取り組んできた．ソフトウェア工学分野での研究実績はないが，データマイニング等をはじめとする大規模データに対するアルゴリズムを研究している．研究分担者（大平雅雄）は，ソフトウェア工学分野においてリポジトリマイニング技術を 10 年以上研究してきており,データ処理の高度化を目的とする本研究では重要な役割を担っている．なお，研究分担者（大平雅雄）が以前実施していた研究と当該委託研究との関係については以下の通りである．

- リポジトリマイニングに基づく大規模 OSS 利用支援環境の構築（JSPS 科研費：基盤（C），2012 年 4 月 1 日～2015 年 3 月 31 日(予定)，研究代表者：大平雅雄）

  本研究は，ソフトウェア開発企業がオープンソースソフトウェア（OSS）を安心して自社製品へ利用できるようにするために，ボランティア開発者主導で開発される OSS の品質とプロジェクトの継続性をモニタリングできる支援環境を構築することを目的としている．具体的には，(1) OSS 開発で利用されている各種ソフトウェアリポジトリから自動的にデータを収集・統合するデータベース基盤の構築，(2) 大規模データを目的に応じて分析するためのリポジトリマイニング手法の分類，および，(3) プラグイン化，(4) 分析結果をモニタリングするための可視化ツールの開発，(5) ソフトウェア開発企業および OSS プロジェクトでの実証実験を通じた支援環境の有用性評価，をおこなう．当該委託研究では，IPA EPM-X をリポジトリマイニング基盤として利用しており直接的な関係はないが，本研究で得た知見を応用することで当該委託研究期間に 8 種類のプラグインを実装することを可能にした．
- EASE（ソフトウェア工学へのエンピリカルアプローチ）プロジェクト

  EPM-X は，文部科学省のリーディングプロジェクト「e-Society 基盤ソフトウェアの総合開発」の一環の EASE プロジェクトで開発された EPM (Empirical Project Monitor)をモデルとしている．研究分担者（大平雅雄）は，EASE プロジェクトへ初期段階からポスドク／助教として参画しており，EPM-X のアーキテクチャやコンセプトに対する理解が深く，EPM-X のプラグイン開発においてその知識を活用できた．

2.1.3 研究目標

本研究では，近年ソフトウェア工学分野で進展の目覚ましいリポジトリマイニング技術を定量的プロジェクト管理ツール EPM-X に取り入れること（研究課題 A），さらに，リポジトリマイニング技術を発展させたプロアクティブマイニング技術を開発する（研究課題 B）ことを研究の達成目標とした．研究課題 A および B の成果を，EPM-X のプラグインとして容易に利用可能な形で普及させることで，定量的な品質・進捗予測機能とプロアクティブ型のプロジェクト管理機能を実現することが狙いである．本研究では以下のようにまず２つの到達目標(G-A)および(G-B)を設定し，到達目標を(G-A-1)から(G-B-3)に分割した．

- 到達目標（G-A）：リポジトリマイニング・プラグインの開発（5 件）
  - (G-A-1) ソフトウェア工学分野における文献調査を幅広く実施し，EPM-X のプラグインとして実装すべきリポジトリマイニング技術を明らかにする．
  - (G-A-2) リポジトリマイニング・プラグインを 5 件（Redmine 版・Trac 版の計 10 件）開発し，容易に利用可能な形式で公開する．
  -
- 到達目標（G-B）：プロアクティブマイニング・プラグインの開発（3 件）

- (G-B-1) 文献調査を幅広く実施し，ソフトウェア開発支援に応用可能な他分野のプロアクティブマイニング技術を明らかにする
- (G-B-2) プロアクティブマイニング・プラグインを 3 件（Redmine 版・Trac 版の計 6 件）開発し，容易に利用可能な形式で公開する．
- (G-B-3) 開発したプロアクティブマイニング技術の有用性を検証し，EPM-X の有用性をアピールする．
-

また，各到達目標をそれぞれ達成するためのマイルストーンとして以下の研究目標を設定し，研究を進めた．

- 到達目標 G-A に向けた研究目標
  - (SG-A-1) リポジトリマイニング技術の調査と選定
  - (SG-A-2) リポジトリマイニング・プラグイン 5 件の開発

- 到達目標 G-B に向けた研究目標
  - (SG-B-1) プロアクティブマイニング技術の調査と選定
  - (SG-B-2) プロアクティブマイニング・プラグイン 3 件の開発
  - (SG-B-3) プロアクティブマイニング・プラグインの有用性検証
  - (SG-B-4) 実務者へのヒアリングを通じたニーズ調査および改良点の調査
  - (SG-B-5) プラグイン開発へのヒアリング調査結果の反映

特にプロアクティブマイニング技術は，産業界のニーズを把握する必要があるため，ヒアリング調査を実施し，調査結果をプラグイン開発に反映させることを心がけた．結果的に，当初予定していた異常行動検知技術ではなく影響波及分析を実装することができた．

2.2 研究の活動実績・経緯

本研究の活動実績を表 2-1～表 2-3 に示す．進捗の遅れはほとんど発生しなかったため，実績のみを示している．2013 年 6 月の研究開始直後，海外研究協力者と連絡をとり，本研究の趣旨を説明した．また，国際会議出席予定を確認し，インフォーマルな形式でのミーティングの予定を立てた．

- (SG-A-1)技術の調査と選定：2013 年 7 月より 8 月まで，リポジトリマイニング技術の調査を行い．実装すべき技術を選定した．
- (SG-A-2-1)プロトタイプ作成：2013 年 9 月より 10 月まで，プラグインを動作させるための基盤部品を実装し，タスク担当者推薦プラグインをプロトタイプとして実装した．実装上，性能上問題ないことを確認した．
- (SG-A-2-2)プラグイン実装・3 件×2 種：2013 年 11 月より 2014 年 4 年まで，3 種類計 6 件（Redmine 版・Trac 版）のプラグイン（タスク担当者推薦プラグイン，タスク再割当状況可視化プラグイン，タスク完了時間予測プラグイン）を実装した．
- (SG-A-2-3)プラグイン実装・2 件×2 種：2014 年 5 月より 2014 年 8 月まで，2 種類計 4 件（Redmine 版・Trac 版）のプラグイン（バグモジュール予測プラグイン，重複コ

ード検出プラグイン）を実装した.

- (SG-A-2-4)インストーラ開発：2014 年 11 月から 12 月まで，リポジトリマイニング・プラグインのインストーラをプロアクティブマイニング・プラグインと併せて開発した.
- (SG-B-1)技術の調査と選定：2013 年 7 月より 10 月まで，プロアクティブマイニング技術の調査を行い．実装すべき技術を選定した.
- (SG-B-2-1)プロトタイプ作成：2013 年 9 月より 10 月まで，プラグインを動作させるための基盤部品を実装し，遅延相関検出プラグインをプロトタイプとして実装した．実装上，性能上問題ないことを確認した.
- (SG-B-2-2)プラグイン実装・2 件×2 種：2013 年 11 月より 2014 年 4 年まで，2 種類計 4 件（Redmine 版・Trac 版）のプラグイン（遅延相関検出プラグイン，変化点検出プラグイン）を実装した.
- (SG-B-2-3)プラグイン実装・1 件×2 種：2014 年 5 月より 2014 年 7 月まで，1 種類計 2 件（Redmine 版・Trac 版）のプラグイン（影響波及分析プラグイン）を実装した.

表 2-1 研究活動の実績（中間報告 1 回目（11 月）まで）

実施項目	2013年 6月			7月			8月			9月			１０月			１１月		
	上	中	下	上	中	下	上	中	下	上	中	下	上	中	下	上	中	下
１．研究準備																		
海外研究協力者との調整	■	■	■															
２．中間目標の達成																		
(SG-A-1)技術の調査と選定				■	■	■	■	■	■									
(SG-A-2-1)プロトタイプ作成										■	■	■	■	■	■			
(SG-A-2-2)プラグイン実装・3件×2種																■	■	■
(SG-A-2-3)プラグイン実装・2件×2種																		
(SG-A-2-4)インストーラ開発																		
(SG-B-1)技術の調査と選定				■	■	■	■	■	■	■	■	■	■	■	■			
(SG-B-2-1)プロトタイプ作成										■	■	■	■	■	■			
(SG-B-2-2)プラグイン実装・2件×2種																■	■	■
(SG-B-2-3)プラグイン実装・1件×2種																		
(SG-B-3-1)検証実験																		
(SG-B-3-2)インストーラ開発																		
(SG-B-4)実務者へのヒアリング調査																		
(SG-B-5)プラグインの開発への反映																		
３．中間報告の準備とフォロー																		
1回目（2013年11月頃）の準備															■	■	■	
フィードバック（1回目）反映の検討																	■	■
2回目（2014年5月頃）の準備																		
フィードバック（2回目）反映の検討																		
3回目（2015年10月頃）																		
フィードバック（3回目）反映の検討																		
４．最終成果のとりまとめ																		
（１）成果報告書の構成案																		
（２）成果概要プレゼン資料																		
（３）成果報告書の作成																		
５．成果物の納品																		

表 2-2 研究活動の実績（中間報告２回目（５月）まで）

実施項目	１２月			2014年 １月			２月			３月			４月			５月		
	上	中	下	上	中	下	上	中	下	上	中	下	上	中	下	上	中	下
１．研究準備																		
海外研究協力者との調整																		
２．中間目標の達成																		
(SG-A-1)技術の調査と選定																		
(SG-A-2-1)プロトタイプ作成																		
(SG-A-2-2)プラグイン実装・3件×2種	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■			
(SG-A-2-3)プラグイン実装・2件×2種																■	■	■
(SG-A-2-4)インストーラ開発																		
(SG-B-1)技術の調査と選定																		
(SG-B-2-1)プロトタイプ作成																		
(SG-B-2-2)プラグイン実装・2件×2種	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■			
(SG-B-2-3)プラグイン実装・1件×2種																■	■	■
(SG-B-3-1)検証実験																		
(SG-B-3-2)インストーラ開発																		
(SG-B-4)実務者へのヒアリング調査														■	■	■	■	■
(SG-B-5)プラグイン開発への反映																		
３．中間報告の準備とフォロー																		
1回目（2013年11月頃）の準備																		
フィードバック（1回目）反映の検討																		
2回目（2014年5月頃）の準備															■	■	■	
フィードバック（2回目）反映の検討																	■	■
3回目（2015年10月頃）																		
フィードバック（3回目）反映の検討																		
４．最終成果のとりまとめ																		
（１）成果報告書の構成案																		
（２）成果概要プレゼン資料																		
（３）成果報告書の作成																		
５．成果物の納品																		

表 2-3 研究活動の実績（中間報告３回目（１０月）および最終成果発表まで）

実施項目	2014年 6月			7月			8月			9月			１０月			１１月			１２月			2015年 1月			2月		
	上	中	下	上	中	下	上	中	下	上	中	下	上	中	下	上	中	下	上	中	下	上	中	下	上	中	下
１．研究準備																											
海外研究協力者との調整																											
２．中間目標の達成																											
(SG-A-1)技術の調査と選定																											
(SG-A-2-1)プロトタイプ作成																											
(SG-A-2-2)プラグイン実装・3件×2種																											
(SG-A-2-3)プラグイン実装・2件×2種	■	■	■	■	■	■	■	■	■																		
(SG-A-2-4)インストーラ開発																■	■	■	■	■	■						
(SG-B-1)技術の調査と選定																											
(SG-B-2-1)プロトタイプ作成																											
(SG-B-2-2)プラグイン実装・2件×2種																											
(SG-B-2-3)プラグイン実装・1件×2種	■	■	■	■	■	■																					
(SG-B-3-1)検証実験							■	■	■	■	■	■	■	■	■												
(SG-B-3-2)インストーラ開発																■	■	■	■	■	■						
(SG-B-4)実務者へのヒアリング調査																											
(SG-B-5)プラグイン開発への反映	■	■	■	■	■	■																					
３．中間報告の準備とフォロー																											
1回目（2013年11月頃）の準備																											
フィードバック（1回目）反映の検討																											
2回目（2014年5月頃）の準備																											
フィードバック（2回目）反映の検討																											
3回目（2015年10月頃）												■	■	■													
フィードバック（3回目）反映の検討														■	■												
４．最終成果のとりまとめ																											
（１）成果報告書の構成案																						■	■				
（２）成果概要プレゼン資料																							■	■			
（３）成果報告書の作成																							■	■	■		
５．成果物の納品																										■	

- (SG-B-3-1)検証実験：2014 年 8 月より 2014 年 10 年まで，プロアクティブマイニング・プラグインの検証実験をおこなった.
- (SG-B-3-2)インストーラ開発：2014 年 11 月から 12 月まで，プロアクティブマイニング・プラグインのインストーラをリポジトリマイニング・プラグインと併せて開発した.
- (SG-B-4)実務者へのヒアリング調査：2014 年 4 月中旬から 5 月上旬にかけて，実務者に対してヒアリングを行い，プラグインに関するニーズを調査した.
- (SG-B-5)プラグインの開発への反映：2014 年 6 月から 7 月まで，ヒアリング調査の結果を受けて強いニーズを確認できた影響波及分析について検討し，プラグイン開発にフィードバックした.

なお，上記の活動とは別に，原則毎週水曜日の定例進捗報告ミーティングや，設定したマイルストーンごとの内部レビュー（全 15 回）も実施した.

研究期間中に参加した学会は以下の通りである．得られた知見は本研究の技術調査、プラグイン開発、インストーラ開発等にフィードバックした.

- 国際会議
  - SNPD'13: 14th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing
    - 知見：ソフトウェア工学に関する最新の研究動向について調査し，プログラム解析技術やリポジトリマイニング技術に関する知見を得た．得られた知見は，本研究の技術調査にフィードバックした.
  - ESEC/FSE'13: 9th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering
    - 知見：リポジトリマイニング技術の最新動向について調査し，Bug Assignment Optimization 技術に関する最新の研究動向についての知見を得た．得られた知見は，本研究の技術調査にフィードバックした.
  - ICSE'14: 36th International Conference on Software Engineering
    - 知見：リポジトリマイニング技術に関する最新動向を調査し，有益な知見を得た.得られた知見は,本研究のプラグイン開発にフィードバックした.
  - ICSME'14: 30th International Conference on Software Maintenance and Evolution
    - 知見：研究発表を行い，海外研究者から有益なコメントを得た．また，ソフトウェア保守技術に関する最新の知見を得た．特に，プロアクティブマイニング技術に関連する Bug Localization に関して多数の発表があり，得られた知見を本研究の検証実験にフィードバックした．また，Ahmed Hassan 氏，Bram Adams 氏，Emad Shihab 氏ら海外アドバイザとのミーティングをおこない，検証実験におけるデータセットの作成方法について助言を得た．得られた助言は本研究の検証実験にフィードバックした.
  - FSE'14: 22nd ACM SIGSOFT International Symposium on the Foundations of Software Engineering

- ✧ 知見：リポジトリマイニング技術およびプロアクティブマイニング技術についての国外・最新研究動向を調査した．特に，研究データの公開方法やツールの配布方法などの知見を得た．得られた知見は本研究のインストーラ開発にフィードバックした．

- 国内会議
  - SES'13:ソフトウェアエンジニアリングシンポジウム 2013
    - ✧ 知見：リポジトリマイニング技術についての国内・最新研究動向を調査した．特に，優秀論文賞を受賞した研究発表「リポジトリマイニングに対する Hadoop の導入に向けた性能評価，大坂陽，山下一寛，亀井靖高，鵜林尚靖（九州大学)」は，EPM-X に採用されている Pentaho とのインテグレーションが進められている Hadoop の性能評価に関する内容であり，プラグインのプロトタイプ実装の参考となる知見を得た．
  - FOSE'13：第 20 回ソフトウェア工学の基礎ワークショップ
    - ✧ 知見：リポジトリマイニング技術およびプロアクティブマイニング技術についての国内・最新研究動向を調査した．具体的には，本プロジェクトと関連のある「クラッシュリポジトリマイニングソースコード-欠陥箇所の特定に向けて- （亀井靖高，長本貴光，ラピュトシャシャンク，小須田光，伊原彰紀，鵜林尚靖)」や「メソッド進化分析のためのソースコード履歴グラフ構築とネットワーク分析（畑秀明，松本健一)」について，知見を得るとともに，著者らと直接議論し，本プロジェクトのプラグイン実装について意見を頂いた．得られた知見は本研究のプラグイン開発にフィードバックした．
  - WWS'14：ウィンターワークショップ 2014・イン・大洗
    - ✧ 知見：研究発表と情報収集をおこない，国内の産業界および学術界における最新の研究成果について幅広い知見を得た．得られた知見は本研究のプラグイン開発にフィードバックした．
  - SES'14：ソフトウェアシンポジウム 2014
    - ✧ 知見：研究発表と情報収集をおこない，国内の産業界および学術界における最新の研究成果について幅広い知見を得た．特に，本研究の「開発者推薦」プラグインに関連する研究発表『大規模 OSS 開発における不具合修正時間の短縮化を目的としたバグトリアージ手法（著者：柏祐太郎，大平雅雄（和歌山大学)，阿萬裕久（愛媛大学)，亀井靖高（九州大学)』をおこない，有益な意見を頂くとともに，最優秀論文賞を受賞した．得られた知見は本研究の参考情報としてフィードバックした．
  - FOSE'14：第 21 回ソフトウェア工学の基礎ワークショップ
    - ✧ 知見：ソフトウェア工学に関する最新の研究動向，特に，公開リポジトリに見られる矛盾のあるデータとその検出方法についての知見を得た．ツールおよびデータセットの公開の際に参考情報としてフィードバックした．

なお，本研究に関連する成果は，上記会議以外も含めて，以下の論文誌および会議にて発表した．

- 招待講演： ２件
    1. 大平 雅雄, ″ICSME (International Conference on Software Maintenance and Evolution) でのトレンド,″ ソフトウエア・メインテナンス・シンポジウム 2014, 2014 年 10 月.
    2. 大平 雅雄, ″ソフトウェアリポジトリマイニングでできること、できないこと,″ 第 9 回 SRA 技術セミナー『開発者視点でのビッグデータの活用--ソフトウェア開発を成功に導く魅力的な開発ログデータの利用へ向けて』, 2014 年 2 月.
- 表彰・受賞： ４件
    1. 最優秀論文賞：柏 祐太郎, 大平 雅雄, 阿萬 裕久, 亀井 靖高, ″大規模 OSS 開発における不具合修正時間の短縮化を目的としたバグトリアージ手法,″ ソフトウェアエンジニアリングシンポジウム 2014 論文集, pages 66--75, 2014 年 8 月.
    2. インタラクティブ賞（受賞者：柏 祐太郎）：柏 祐太郎, 大平 雅雄, ″不具合修正タスク量の最適化を目的としたバグトリアージ手法の提案,″ ソフトウェアエンジニアリングシンポジウム 2013 論文集, pages 1--2, 2013 年 9 月.
    3. 優秀論文賞：山谷 陽亮, 大平 雅雄, Passakorn Phannachitta, 伊原 彰紀, ″OSS システムとコミュニティの共進化の理解を目的としたデータマイニング手法,″ 情報処理学会 マルチメディア, 分散, 協調とモバイル (DICOMO2013) シンポジウム論文集, pages 1695--1703, 2013 年 7 月.
    4. ヤングリサーチャ賞（受賞者：山谷 陽亮）：山谷 陽亮, 大平 雅雄, Passakorn Phannachitta, 伊原 彰紀, ″OSS システムとコミュニティの共進化の理解を目的としたデータマイニング手法,″ 情報処理学会 マルチメディア, 分散, 協調とモバイル (DICOMO2013) シンポジウム論文集, pages 1695--1703, 2013 年 7 月.
- 学術論文： ３件（査読付き）
    1. 柏 祐太郎, 大平 雅雄, 阿萬 裕久, 亀井 靖高, 大規模 OSS 開発における不具合修正時間の短縮化を目的としたバグトリアージ手法, 情報処理学会論文誌（採録済）
    2. 吉行 勇人, 大平 雅雄, 戸田 航史, OSS 開発における管理者と修正者の社会的関係を考慮した不具合修正時間予測, コンピュータソフトウェア（採録済）
    3. 山谷 陽亮, 大平 雅雄, Passakorn Phannachitta, 伊原 彰紀, OSS システムとコミュニティの共進化の理解を目的としたデータマイニング手法, 情報処理学会論文誌, volume 56, number 1, pages 59—71, 2015 年 1 月.
- 国際会議： ３件（査読付き）
    1. Yosuke Yamatani and Masao Ohira, An Exploratory Analysis for Studying Software Evolution: Time-Delayed Correlation Analysis, In Proceedings

of 6th International Workshop on Empirical Software Engineering in Practice (IWESEP 2014), pages 13—18, November 2014.

2. Yutaro Kashiwa, Hayato Yoshiyuki, Yusuke Kukita, and Masao Ohira, A Pilot Study of Diversity in High Impact Bugs, In Proceedings of 30th International Conference on Software Maintenance and Evolution (ICSME2014), pages 536—540, 2014.
3. Masao Ohira and Hayato Yoshiyuki, A New Perspective on the Socialness in Bug Triaging: a Case Study of the Eclipse Platform Project, In Proceedings of 5th International Workshop on Social Software Engineering, pages 29—32, August 2013.

- 国内会議： １５件（査読付き１０件，査読なし５件）
  1. 久木田 雄亮，柏 祐太郎，大平 雅雄 ，″変化点検出とトピック分析を用いたリポジトリマイニング手法の提案,″ ウィンターワークショップ 2015・イン・宜野湾 論文集，pages 23--24，2015 年 1 月．（査読あり）
  2. 吉行 勇人，大平 雅雄 ，″不具合修正における優先度と修正作業の関係理解のための分析,″ ウィンターワークショップ 2015・イン・宜野湾 論文集，pages 19--20，2015 年 1 月．（査読あり）
  3. 大平 雅雄，柏 祐太郎，松本 明，山谷 陽亮，吉行 勇人，″プロジェクト管理ツール IPA EPM-X の機能拡張によるリポジトリマイニング研究基盤の構築,″ ウィンターワークショップ 2015・イン・宜野湾 論文集，pages 41--42，2015 年 1 月．（査読あり）
  4. 柏 祐太郎，吉行 勇人，大平 雅雄 ，″High Impact Bug が与える影響の分析に向けて,″ ウィンターワークショップ 2015・イン・宜野湾 論文集，pages 39--40，2015 年 1 月．（査読あり）
  5. 柏 祐太郎，大平 雅雄，阿萬 裕久，亀井 靖高，″大規模 OSS 開発における不具合修正時間の短縮化を目的としたバグトリアージ手法,″ ソフトウェアエンジニアリングシンポジウム 2014 論文集，pages 66--75，2014 年 8 月．（査読あり）
  6. 吉行 勇人，大平 雅雄，″修正者のタスク優先順位付けが不具合修正時間に与える影響,″ 情報処理学会シンポジウムシリーズ，ウィンターワークショップ 2014・イン・大洗，volume 2014，pages 99--100，2014 年 1 月．（査読あり）
  7. 山谷 陽亮，大平 雅雄，″時間的順序関係を考慮した相関分析手法の提案,″ 情報処理学会シンポジウムシリーズ，ウィンターワークショップ 2014・イン・大洗，volume 2014，pages 97--98，2014 年 1 月．（査読あり）
  8. 金城 清史，大平 雅雄 ，″コミッター候補者予測における OSS プロジェクトの開放性に関する考察,″ 情報処理学会シンポジウムシリーズ，ウィンターワークショップ 2014・イン・大洗，volume 2014，pages 33--34，2014 年 1 月．（査読あり）

9. 柏 祐太郎, 大平 雅雄, ″不具合修正時間の短縮化を目的としたバグトリアージ手法の提案,″ ソフトウェア工学の基礎XX,日本ソフトウェア科学会FOSE2013, volume 20, pages 053--058, 2013年11月. （査読あり）
10. 松本 明, 柏 祐太郎, 大平 雅雄, ″リポジトリマイニング手法の習得支援を目的としたシミュレーション環境の構築,″ ソフトウェアエンジニアリングシンポジウム2013論文集, pages 1--2, 2013年9月. （査読あり）
11. 金城 清史, 山谷 陽亮, 大平 雅雄, ″コミッター選出のためのプロジェクトメトリクスに関する定量的評価,″ 情報処理学会 マルチメディア,分散,協調とモバイル（DICOMO2014）シンポジウム論文集, pages 926--933, 2014年7月. （査読なし）
12. 山谷 陽亮, 大平 雅雄, ″ソフトウェア進化の理解を目的とした遅延相関分析手法：OSSプロジェクトデータへの適用,″ 情報処理学会報告 ソフトウェア工学研究会, volume 2014-SE-185, number 8, pages 1-8, 2014年. （査読なし）
13. 吉行 勇人, 大平 雅雄, ″OSS開発における管理者と開発者間の社会的関係がタスク遂行に与える影響の考察,″ 情報処理学会報告 ソフトウェア工学研究会, volume 2013-SE-181, number 6, pages 1-8, 2013年7月. （査読なし）
14. 山谷 陽亮, 大平 雅雄, 伊原 彰紀, ″OSS開発における共進化プロセスの理解のための遅延相関分析,″ 情報処理学会報告 ソフトウェア工学研究会, volume 2013-SE-181, number 3, pages 1-8, 2013年7月. （査読なし）
15. 柏 祐太郎, 大平 雅雄, ″OSS開発におけるタスク割当の最適化に関する考察,″ 情報処理学会報告 ソフトウェア工学研究会, volume 2013-SE-181, number 2, pages 1--8, 2013年7月. （査読なし）

など

2.3 研究実施体制

図 2-1 に研究実施体制を示す．また，表 2-4 に研究者のプロフィールを示す．

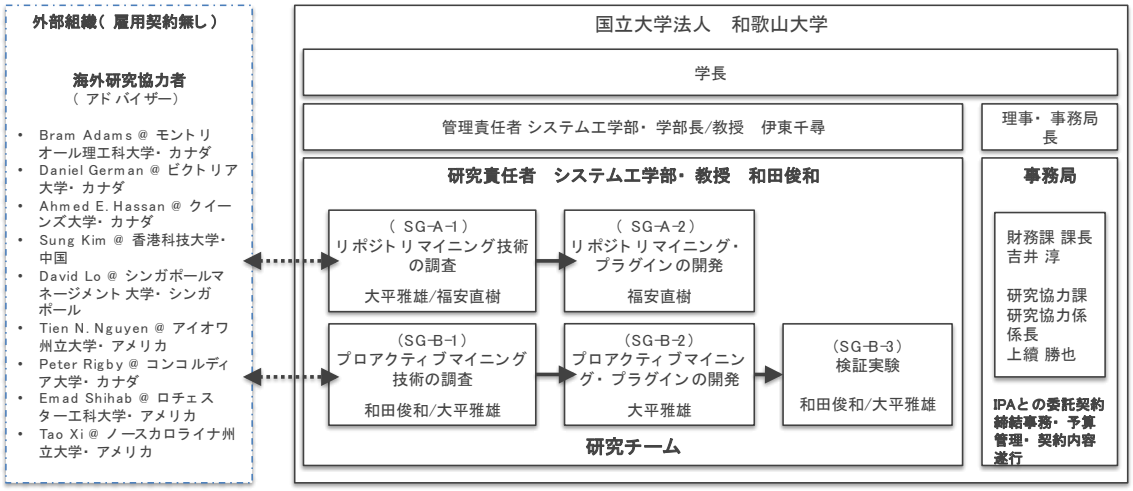


図 2-1 研究実施体制

表 2-4 研究者のプロフィール

氏名	和田俊和
最終学歴 主な経歴	表 2-5「研究責任者のプロフィール」を参照
主な研究分野	最近傍探索および事例ベースの画像解析・スパムメール検出，データマイニング等の研究に従事.
氏名	大平雅雄
最終学歴	奈良先端科学技術大学院大学情報科学研究科博士後期課程修了
主な経歴	平成 15 年 同大学・産学官連携研究員 平成 16 年 同大学・助手(平成 19 年より助教) 平成 24 年 和歌山大学システム工学部・講師 平成 25 年 和歌山大学システム工学部・准教授 博士(工学)(奈良先端科学技術大学院大学)
主な研究分野	エンピリカルソフトウェア工学, 特にリポジトリマイニング, オープンソースソフトウェア, ソフトウェア保守，品質改善等の研究に従事.
氏名	福安直樹
最終学歴	名古屋大学大学院工学研究科情報工学専攻博士後期課程修了
主な経歴	平成 12 年 和歌山大学システム工学・助手(平成 19 年より助教) 平成 26 年 和歌山大学システム工学・准教授 博士(工学)(名古屋大学)
主な研究分野	ソフトウェア開発環境，ウェブ工学，ソフトウェア工学教育等の研究に従事.

表 2-5 に研究者チームの役割分担を示す.

表 2-5 研究チームの役割分担

<table>
<tr><td>実施機関名</td><td colspan="3">国立大学法人 和歌山大学システム工学部</td></tr>
<tr><td>研究責任者</td><td colspan="3">教授 和田俊和</td></tr>
<tr><td>連絡担当者</td><td colspan="3">准教授 大平雅雄</td></tr>
<tr><td rowspan="12">主な実施場所
及び研究者</td><td colspan="3">国立大学法人 和歌山大学システム工学部
〒640-8510 和歌山市栄谷 930 番地</td></tr>
<tr><td>氏名</td><td>職種</td><td>担当内容</td></tr>
<tr><td>和田俊和</td><td>教授</td><td>全体取りまとめ, プロアクティブマイニングプラグインに関する技術調査／開発／検証実験および学生アルバイトへの作業指示</td></tr>
<tr><td>大平雅雄</td><td>准教授</td><td>リポジトリマイニングプラグインおよびプロアクティブマイニングプラグインに関する技術調査, プロアクティブマイニングプラグインの開発／検証実験／インストーラ開発／マニュアル作成および学生アルバイトへの作業指示</td></tr>
<tr><td>福安直樹</td><td>准教授</td><td>リポジトリマイニングプラグインに関する技術調査／開発／インストーラ開発および学生アルバイトへの作業指示</td></tr>
<tr><td>柏祐太郎</td><td>研究支援員</td><td>プロアクティブマイニング技術の調査と選定の補助／リポジトリマイニングプラグインの実装補助／プロアクティブマイニングプラグインの実装補助／検証実験の補助／マニュアル作成およびインストーラ開発の補助</td></tr>
<tr><td>永井琢也</td><td>研究支援員</td><td>リポジトリマイニング技術の調査と選定の補助／プロアクティブマイニングにおける技術の調査の選定の補助／リポジトリマイニングプラグインの実装補助</td></tr>
<tr><td>松井麻吏奈</td><td>研究支援員</td><td>リポジトリマイニング技術の調査と選定の補助／プロアクティブマイニングにおける技術の調査の選定の補助／リポジトリマイニングプラグインの実装補助</td></tr>
<tr><td>井村和樹</td><td>研究支援員</td><td>リポジトリマイニングプラグインのプロトタイプ作成補助</td></tr>
<tr><td>金岡弘道</td><td>研究支援員</td><td>リポジトリマイニングプラグインのプロトタイプ作成補助</td></tr>
<tr><td>福田翔</td><td>研究支援員</td><td>リポジトリマイニングプラグインのプロトタイプ作成補助</td></tr>
<tr><td>松村祐貴</td><td>研究支援員</td><td>プロアクティブマイニングプラグインのプロトタイプ作成補助／プロアクティブマイニングプラグインの実装補助／リポジトリマイニングプラグインの実装補助／検証実験の補助／マニュアル作成およびインストーラ開発の補助</td></tr>
</table>

表 2-5 研究チームの役割分担（つづき）

	森敦	研究支援員	プロアクティブマイニングに関するプロトタイプ作成補助／プロアクティブマイニングプラグインの実装補助／リポジトリマイニングプラグインの実装補助／検証実験の補助／マニュアル作成およびインストーラ開発の補助
	大谷洋平	研究支援員	プロアクティブマイニングに関するプロトタイプ作成補助／プロアクティブマイニングプラグインの実装補助
	湯浅圭太	研究支援員	プロアクティブマイニングに関するプロトタイプ作成補助／プロアクティブマイニングプラグインの実装補助／リポジトリマイニングプラグインの実装補助／検証実験の補助／マニュアル作成およびインストーラ開発の補助
	福井崇之	研究支援員	プロアクティブマイニングに関するプロトタイプ作成補助／プロアクティブマイニングプラグインの実装補助／リポジトリマイニングプラグインの実装補助／検証実験の補助／マニュアル作成およびインストーラ開発の補助
	松本明	研究支援員	リポジトリマイニングプラグインの実装補助／プロアクティブマイニングプラグインの実装補助／マニュアル作成およびインストーラ開発の補助
	山谷陽亮	研究支援員	リポジトリマイニングプラグインの実装補助／プロアクティブマイニングプラグインの実装補助／マニュアル作成およびインストーラ開発の補助
	吉行勇人	研究支援員	リポジトリマイニングプラグインの実装補助／プロアクティブマイニングプラグインの実装補助／マニュアル作成およびインストーラ開発の補助

3 研究成果

3.1 研究目標 1「リポジトリマイニング技術の調査と選定（SG-A-1)」

3.1.1 当初の想定

(1) 研究内容

（内容）

過去 10 年分の文献調査を通じて既存リポジトリマイニング技術を分類し,有用性が高くかつプラグイン化可能なものを明らかにし，その中から 5 件のリポジトリマイニング技術を選定する.

（想定される課題と解決策）

リポジトリマイニング技術は,技術の体系化が進まない状態で乱立して提案されており,選定のための調査が進まない可能性もわずかながら予想される．その場合は，TSE・ESEM のトップジャーナル，ICSE･FSE のトップカンファレンスで発表された技術のみを対象（数十件）として，実装する技術を選定することとした.

(2) 当初の到達目標と期待される効果

（当初の到達目標）

ソフトウェア工学分野における文献調査を幅広く実施し，EPM-X のプラグインとして実装すべきリポジトリマイニング技術を明らかにする(G-A-1).

（期待される効果）

多数提案されている技術の中から実務志向の有用性の高いリポジトリマイニング技術を選定することができる.

3.1.2 研究プロセスと成果

(1) 研究プロセス

以下の順序で研究を進めた.

1. 文献調査
2. 技術分類
3. 技術選定

文献調査では，ソフトウェア工学分野におけるリポジトリマイニング技術に関して，トップ会議・トップジャーナルを中心に過去 10 年分の文献を調査した.

技術分類では，文献調査に基づき，既存技術を以下の 3 つの観点で分類した.

- 適用ドメイン（適用対象業務ごとに分類）
- 対象リポジトリ（Subversion や Trac など，リポジトリごとに分類）
- 支援目的（実装支援や保守支援など，支援目的ごとに分類）

技術選定では，技術分類に基づいて，検証が十分に行われており，かつ，以下の条件を満たす技術を 5 件，選定した.

・　現場のニーズに合っていること
・　EPM-X と親和性の高いリポジトリを適用対象としていること
・　プラグインとして実装するコストが大きくなり過ぎないこと

(2) 具体的な研究成果（結果）

①文献調査

ソフトウェア工学におけるトップ会議やトップジャーナルを中心に，リポジトリマイニング技術に関する調査を行った．具体的には以下の国際会議論文集および論文誌を約 10 年分，約 50 編を調査した．

- 国際会議論文集
  - ICSE（International Conference on Software Engineering）
  - FSE（Symposium on Foundation of Software Engineering）
  - ASE（International Conference on Automated Software Engineering）
  - MSR（Working Conference on Mining Software Repositories）
- 論文誌
  - TSE（IEEE Transaction on Software Engineering）
  - TOSEM（ACM Transactions on Software Engineering and Methodology）
  - ESEM（Journal of Empirical Software Engineering）

②技術分類

技術選定にあたっては，門田らの技術解説[門田 13]を参考にして以下の技術的観点を設定し，特定の用途に偏ることがないように注意した．

- プログラミング：ソフトウェア部品検索，コードクローン分析など
- テスト：バグモジュール予測，テスト工数予測など
- 保守：バグ要因分析，エキスパート推薦，バグ特定など
- 管理：ライセンス分析，コミュニケーション分析など
- OSS 利用：品質・成熟度評価，修正時間予測など

③技術選定

調査結果に基づいて，実務志向で有用性の高いもの（ただし，研究実施期間中に実現可能性なもの）として以下の 5 種類のリポジトリマイニング技術を選定した．

- タスク担当者推薦[Anvik06][Park11][Xuan12]
- タスク再割当状況可視化[Jeong09][Bhattacharya10] [Bhattacharya12]
- タスク完了時間予測[Weiss07][Hewett09][Zhang13]
- バグモジュール予測[S ´liwerski05][Nagappan06][Kim08]
- 重複コード検出[Ducasse99][Lin14][Monda114]

3.1.3 課題とその対応

研究を進めるにあたって用いた技術分類はあくまでも学術的成果を分類したものであり，現場のニーズと強くマッチした分類となっていない可能性がある．また，2 年目におこっ

なった実務者向けのヒアリング調査においても，必ずしもすべての企業の現場において必要でないことも明らかになった.既存研究の多くは OSS データを用いて検証を行っており，現場ニーズと乖離した状況設定が含まれていたためと考えられる．例えば，大規模 OSS 開発では数百名のボランティア開発者がプロジェクトに参加することも珍しくないため，不具合修正などのタスクをどの開発者に割り当てるのが最もコスト的・品質的に効果的か，といった判断をプロジェクト管理者が常に正しく行うのは困難な状況にある（結果的に，現状の OSS 開発では，不具合修正タスクの多くは担当者が何度も変更される）．そのため，「タスク担当者推薦」技術は，プロジェクト管理者の意思決定を支援する上で非常に有用なものとして高く評価されてきた．しかしながら，ヒアリング調査では,「現場ではアサイン状況がほぼ確定しており、担当すべき要員も掌握しているため、推薦の必要がないケースが多い。」といった意見があり，現場ニーズと上手くマッチしてない場合があることがわかった．ただし,「管理要員や新規要員がプロジェクトの構成を把握する際に利用可能と思われる。」や「〇社では品質保証部門が別であり、総合試験は品証でやっています。システム担当なのか、ソフト担当なのか、ハード担当なのか割り振るという場面では使えるかもしれません。」などといった意見もあり，現場ニーズに沿うよう既存技術をカスタマイズすることで,利用価値が存在する技術であることもわかった．今後の普及と展開においては，現場のニーズをより広く調査し，利用価値の高いリポジトリマイニング技術を現場目線で選定し，プラグイン化していく必要がある．

3.2 研究目標 2「リポジトリマイニング・プラグイン 5 件の開発（SG-A-2)」

3.2.1 当初の想定

(1) 研究内容

（内容）

まず，SG-A-1 の技術調査および選定結果に基づき，リポジトリマイニングのアルゴリズムおよび入出力のためのデータ構造の仕様を決定する．次に，決定した仕様からリポジトリマイニング技術を EPM-X の拡張機能としてプラグイン化できることを確かめるために，実装の容易なタスク担当者推薦アルゴリズムを例にとりプラグインのプロトタイプを作成した.

次に，Redmine 版・Trac 版それぞれ 5 件、計 10 件のプラグインを順に実装し，オープンソースプロジェクトデータを用いて，実装したプラグインが調査した文献に記述されている通りに解析結果を出力できるかどうかをテストした．リポジトリマイニング技術の有効性そのものは検証されたものを利用するため検証実験は行っていない.

その後，テストにより問題がなければ，インストーラを開発し容易に利用可能な形式で公開できるようにする．得られた知見は，インストーラ開発終盤にプラグインに反映させた.

（想定される課題と解決策）

1 プラグインあたり実装 1 ヶ月・テスト 1 ヶ月，計 2 ヵ月での完成を予定した．既存技

術のプラグイン化であるため実装自体にさほど困難な点は考えられないが，プロジェクト開始当初はプラグイン構造の設計に多少手間取る可能性があった．用意されているドキュメント類を精査するとともに，必要があればIPAのEPM-X問い合わせ窓口に連絡し相談することとした．

(2) 当初の到達目標と期待される効果

（当初の到達目標）

リポジトリマイニング・プラグインを5件（Redmine版・Trac版の計10件）開発し，容易に利用可能な形式で公開する(G-A-2)．

（期待される効果）

開発したプラグインを公開することにより，リポジトリマイニング技術を容易に利用できるようになる．また，委託契約期間終了後もプラグインをさらに発展させるための知見を，早期にかつ多方面から収集することが可能になる．

3.2.2 研究プロセスと成果

(1) 研究プロセス

以下の順序で研究を進めた．

1. 仕様決定・プロトタイプ作成
2. プラグイン実装・テスト
3. インストーラ開発

(2) 具体的な研究成果（結果）

①仕様決定・プロトタイプ作成

選定した技術の詳細が記述されている文献に基づき，リポジトリマイニングのアルゴリズムおよび入出力のためのデータ構造の仕様を決定した．また，プラグインを動作させるための基盤部品を実装し，タスク担当者推薦プラグインをプロトタイプとして実装した．実装上，性能上問題ないことを確認した．

②プラグイン実装・テスト

決定した仕様に基づきプラグインをそれぞれ実装した．RedmineおよびTracで動作することを確かめるためのテストを実施した． 開発した5種類のリポジトリマイニング・プラグインの詳細を以下に説明する．

- **タスク担当者推薦プラグイン**

（概要）

タスク担当者推薦プラグインは，過去の完了したタスクの情報に基づいて，新規タスクを担当すべき開発者を推薦するプラグインである．タスク担当者推薦プラグインを利用す

ることによって，効率的なタスクの割当てを行うことができるため，タスク完了にかかる時間を短縮することが期待できる．

（アルゴリズム）

タスク担当者推薦プラグインでは，過去の完了したタスクに含まれる文章からキーワードを抽出し，抽出されたキーワードとそのタスクを完了した開発者を学習する．学習に用いる機械学習アルゴリズムはナイーブベイズ法であり，学習データと同様に収集した未完了スクのキーワードのデータをテストデータとすることによって，そのタスクを担当すべき開発者を推薦する[Anvik06][Park11][Xuan12]．

まず，TF-IDF 法に基づくキーワードの抽出の方法について詳しく説明する． 完了済みのタスクから，タスクの題名，タスクの概要，タスク担当者のテキストを関連付けて収集する．収集したデータに対して，単語ごとに TF-IDF 法で重み付けをする．

TF-IDF 法は，文書中の単語に関する重み付け手法の一種であり，主に情報検索や文章要約などの分野で利用される．TF-IDF 法の基本的な考えは以下の 2 つである．

1. 特定の文書に頻出する単語は，その文書を特徴付けているため，重みを大きくする(この重みを TF と呼ぶ)
2. 特定の文書だけでなく，他の文書にも頻出している単語は，その文書を特徴づける単語とはいえないため，重みを小さくする(この重みを IDF と呼ぶ)

なお，ストップワード(英語なら”a”，”the”，”I”，日本語なら“です”，“ます”，“こと”，“の”などのように，頻出するものの，それ自体が文書を特徴づけることはあまりないような語)はあらかじめ除いている．

次に，ナイーブベイズ法を用いた開発者推薦のための機械学習アルゴリズムについて説明する．ナイーブベイズ法は，特徴ベクトル間に条件付き独立性を仮定したベイズ定理に基づく分類モデルである．ナイーブベイズ法を学習アルゴリズムとすることで，高速でスケーラビリティの高いモデルの作成を実行できる．単語別に重み付けがなされたベクトルデータを学習データとしてモデルを作成した後，未完了のタスクをテストデータとしてナイーブベイズ法を適用して開発者ごとに分類する．タスク担当者推薦プラグインでは，ナイーブベイズ法の分類によって，各未完了タスクに対して，開発者名とその開発者が適任である確率が算出される．このうち，適任である確率が高い開発者の上位 3 名のみを担当者推薦結果の出力画面で表示する．

（プラグイン画面）

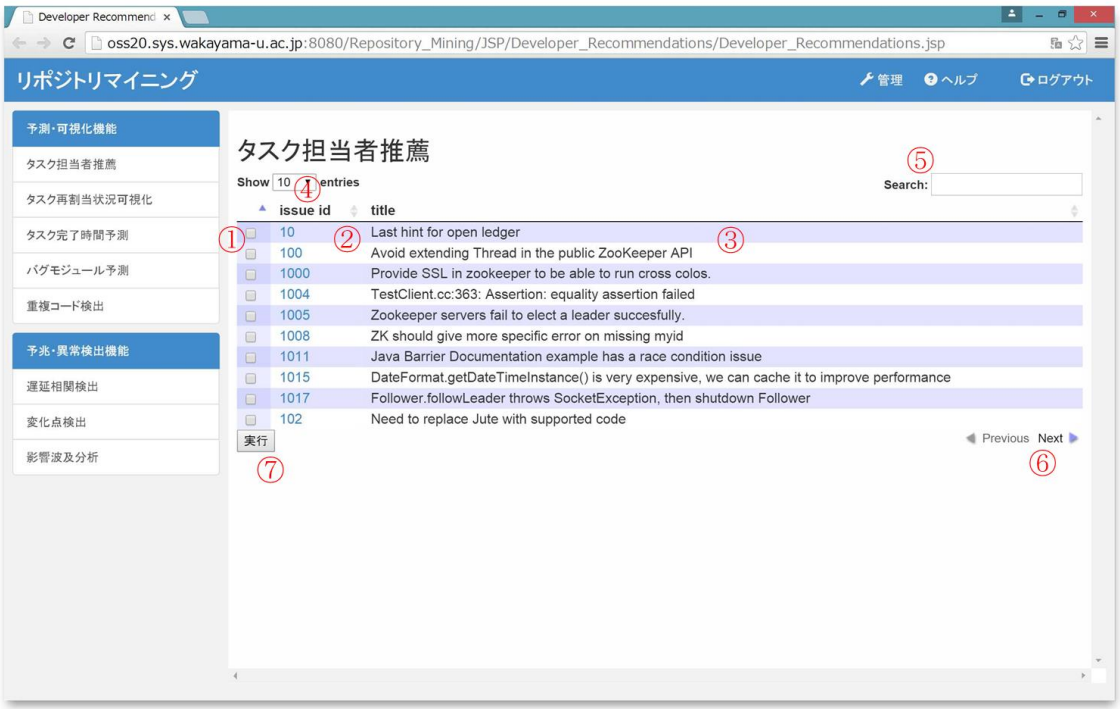


図 3-1 タスク選択画面（タスク担当者推薦プラグイン）

タスク担当者推薦プラグインを呼び出すと,完了していないタスクが issue_id の昇順で 10 件一覧出力される．図 3-1 は，タスク一覧の出力例を示している．①は，担当者を推薦したいタスクを選択するチェックボックスである(このチェックボックスは複数選択可能)．②は，タスクの ID を示している．issue_id ボタンを押すことで ID の昇順・降順を変更できる．また，ID のリンクをクリックすることで，その ID のタスクの詳細画面に遷移する．③は，タスクの題名のテキストを示している．title ボタンを押すことで題名の昇順・降順を変更できる．

図 3-2 に担当者推薦結果の出力画面の例を示す．図 3-1 に示したタスク選択画面においてチェックボックスでタスクが選択され,実行ボタンを押されると本出力画面に遷移する．

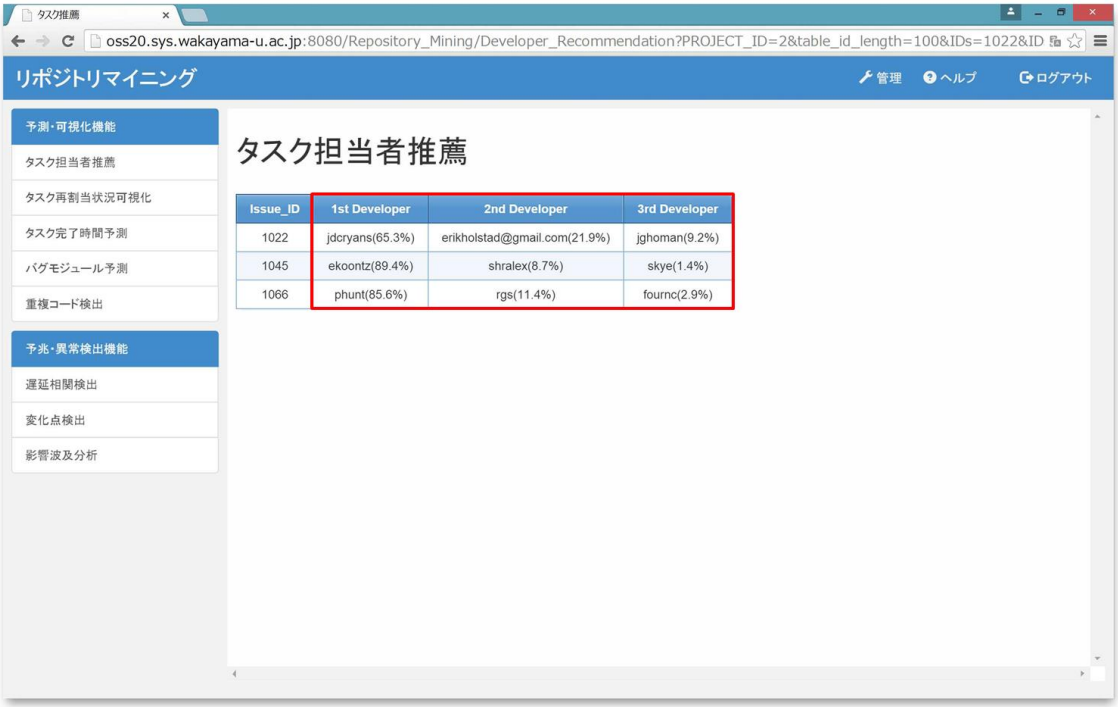


図 3-2 タスク担当者推薦プラグインの出力結果の例

issue_id に従い昇順で一覧表示される．赤枠部分にタスク選択画面において選択したタスクに対して推薦された担当者が上位 3 名まで表示されている．氏名および適任の度合いを確率としてパーセンテージ表示している．適任の度合いに応じて”1st Developer”，”2nd Developer”，”3rd Developer”の順に表示される．

- **タスク割当状況可視化プラグイン**

（概要）

タスク再割当状況可視化プラグインは，過去のタスクの再割当情報を可視化するプラグインである．タスク再割当状況可視化プラグインを利用することによって，効率的なタスクの割当てを行うことができるため，タスク完了にかかる時間を短縮することが期待できる．

（アルゴリズム）

タスク再割当状況可視化プラグインでは，タスクの再割当が発生する回数を減らすために，過去のタスク再割当の情報をマルコフモデルに準ずるネットワーク図上に可視化する．この過去のタスク再割当の情報をマルコフモデルに準ずるネットワーク図上に可視化する手法は，Jeong らの研究で提案されたものであり，Tossing Graph と呼ばれる．Tossing Graph を用いることによって，開発者の関係やチームの構造を明らかにすることができることや，よりタスクの完了を見込むことのできる開発者にタスクを割当てることを支援することが

できる[Jeong09][Bhattacharya10][Bhattacharya12].

Tossing Graph は，マルコフモデルに準ずるタスク割当てのネットワーク図である．マルコフモデルとは，状態の推移確率を記述したもので，現在の状態のみが次の状態に影響を及ぼすところに特徴がある．Tossing Graph を作成するために，タスク再割当状況可視化プラグインでは，以下の 3 つの処理を行う．

(Ⅰ)実際のタスク割当て順序の算出
(Ⅱ)ゴール志向のタスク割当て順序の算出
(Ⅲ)タスク割当て状況の可視化

次に，この 3 つの処理について説明する．

(Ⅰ)実際のタスク割当て順序の算出

実際のタスク割当て順序とは,ソフトウェア開発プロジェクトにおける過去の情報から,実際にどのようなタスク割当てが行われてきたのかを示すものとなっている．図 3-3 に,実際のタスク割当て順序を示すようなタスクの再割当状況の例を挙げる．

$$A \rightarrow B \rightarrow C \rightarrow D$$

図 3-3　タスクの再割当状況の例

図 3-3 は，あるタスクが開発者 A に割当てられ，その後，そのタスクが開発者 B，開発者 C，開発者 D の順に割当てられ，最終的に開発者 D によってタスクが完了されたことを示している．

図 3-3 のタスク割当て順序の例において，効率的なタスクの割当てを行い，タスクを完了するまでにかかる時間を短縮するためには，開発者 A に割当てられた後，開発者 B に割当てるのではなく，開発者 D に直接タスクを割当てることによって，タスクの完了までにかかる時間を短縮させることができると考えられる．しかしながら，開発者が数多く存在する大規模開発現場では，タスクを完了することのできる開発者に一度の割当てでタスクを割当てることは現実問題困難であるため，タスク担当候補者の中からタスクを完了することのできる開発者にタスクを直接割当てるための支援を行う．

（Ⅱ）ゴール志向のタスク割当て順序の算出

次に，実際のタスク割当て順序からゴール志向のタスク割当て順序を求める．ゴール志向のタスク割当て順序とは，タスクを完了させるための最短経路を示すタスク割当て順序である．例えば，図 3-4 に示すように図 3-3 に示したタスク割当て状況を考える．

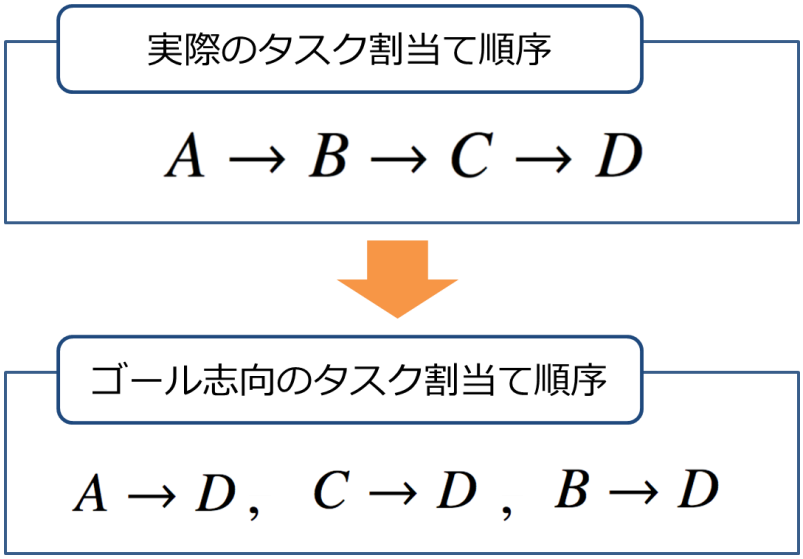


図 3-4　実際のタスク割当て順序からのゴール志向のタスク割当て順序の算出
（[Jeong09]の図を改編）

この状況では，開発者 A に割当てられたタスクが，その後，開発者 B，開発者 C に割当てられ，最終的に開発者 D によって完了されている．このタスク割当て状況におけるゴール志向のタスク割当て順序は，図 3-4 に示すように，開発者 A から開発者 D，開発者 C から開発者 D，開発者 B から開発者 D となる．つまり，開発者 A が完了することのできないタスクを開発者 D に割当てることで，タスク完了にかかる時間を短縮することができることがゴール志向のタスク割当て順序からわかる．

このような実際のタスク割当て順序からゴール志向のタスク割当て順序を算出する処理をすべてのタスクに対して行う．

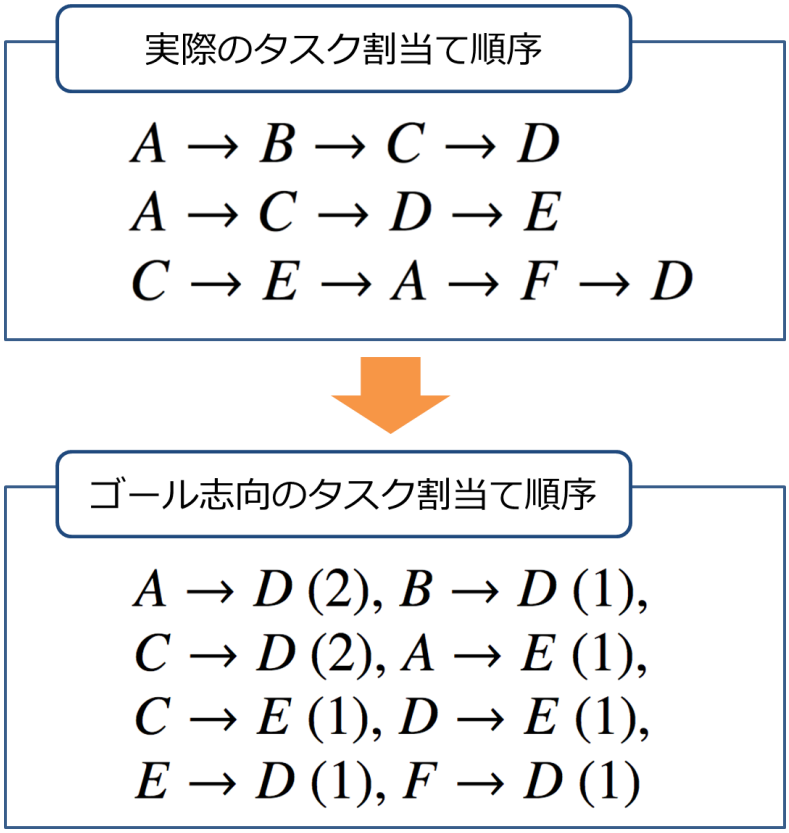


図 3-5　ゴール志向のタスク割当て順序の例（[Jeong09]から引用）

（Ⅲ）タスク割当て状況の可視化

最後に，効率的なタスク割当てを支援するために，タスク割当て状況の可視化を行う．可視化されるグラフは，タスクを割当てた際のタスクが完了するかどうかの確率をゴール志向のタスク割当て順序に基づき開発者ごとに示したものである．タスクを割当てた際のタスクが完了するかどうかの確率は，ゴール志向のタスク割当て順序により求まる．例えば，図 3-5 のゴール志向のタスク割当て順序において，開発者 A から開発者 D のパスの発生回数は 2 回，開発者 A から開発者 E のパスの発生回数は 1 回である．そのため，開発者 A からタスクを割当ててタスクを完了する確率は，開発者 D に割当てると 66.7%(=2/3)，開発者 E に割当てると 33.3%(=1/3)となる．つまり，この場合，開発者 A から開発者 D に割当てた方がタスクの完了する確率は高くなる．このような，タスクが完了する確率をすべてのタスク割当て順序に対して求める．

図 3-3 に示すような実際のタスク割当て順序の例からゴール志向のタスク割当て順序を算出する処理をすべてのタスクに対して行うと，図 3-5 のようになる．図 3-5 において，あるゴール志向のタスク割当て順序における数字は，発生回数を示している．例えば，開発者 A から開発者 D のゴール志向のタスク割当て順序は，図 3-5 の実際のタスク割当て順序の例において，上段と下段の 2 つのタスクで発生しているため，開発者 A から開発差 D のゴール志向のタスク割当て順序の発生回数は 2 となる．

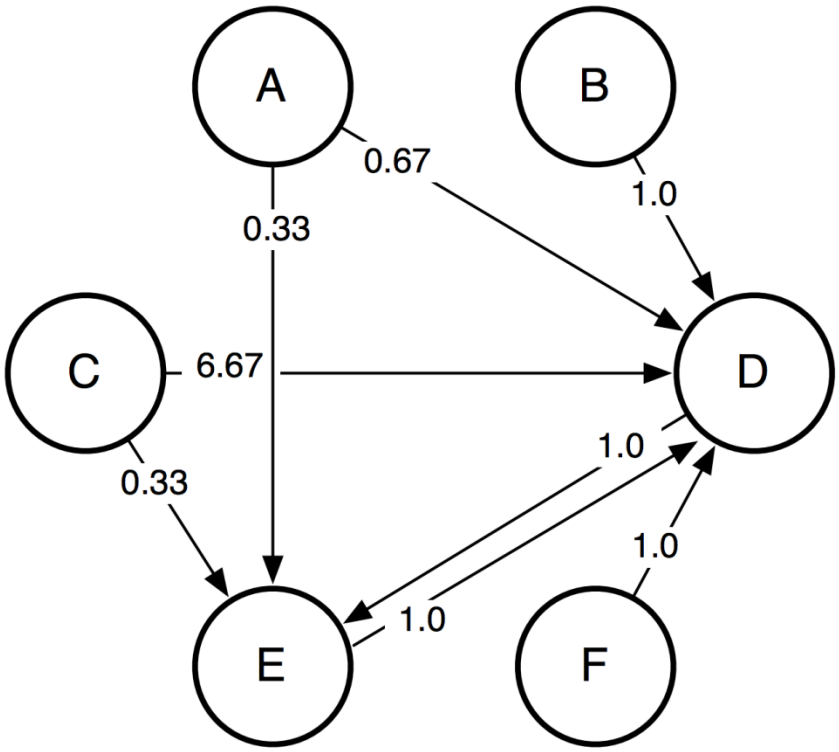


図 3-6　タスクが完了する確率モデル（[Jeong09]から引用）

図 3-6 は,図 3-3 に示すタスク割当て状況におけるタスクが完了する確率を示したモデルとなっている．先ほどの例のように，開発者 A から開発者 D にタスクを割当てた場合，タスクが完了する確率は約 67%であり，開発者 A から開発者 E にタスクを割当てた場合，タスクが完了する確率は約 33%である．

（プラグイン画面）

タスク再割当状況可視化プラグインを呼び出すと，過去のタスクの再割当状況を確率モデルとして出力する．図 3-7 は，出力結果の例を示している．出力されたグラフにおける円の大きさは，割当てが行われた回数を示しており，円が大きいほど割当てられた回数が多いことを意味している．また，表示されている確率は，ある開発者から他の開発者に対してタスクの割当てが発生した際に，タスクが完了する確率を意味している．

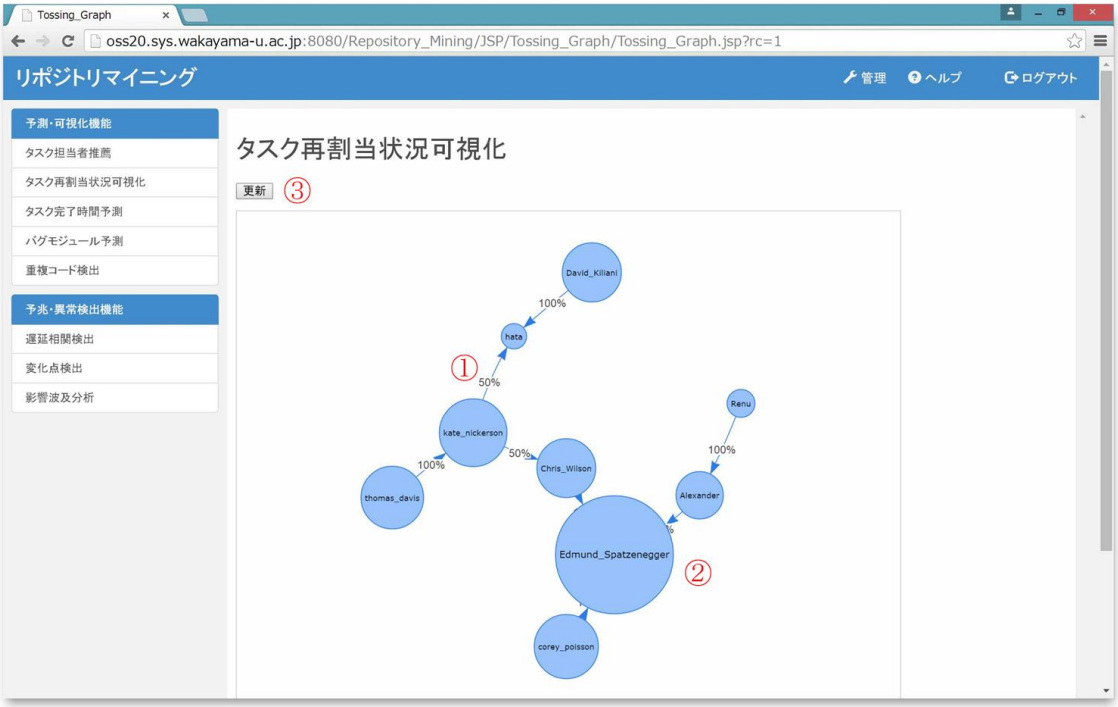


図 3-7 タスク再割当状況可視化プラグインの出力結果の例

- **タスク完了時間予測プラグイン**

（概要）

タスク完了時間予測プラグインは，過去の完了したタスクの情報に基づいて，新規タスクが完了すると考えられる時間を予測するプラグインである．タスク完了時間予測プラグインを利用することによって，適切なタスクの見積もりを行うことができるため，効率的なタスク管理がおこなえると期待できる．

（アルゴリズム）

タスク完了時間予測プラグインでは，予測モデルを構築するために，チケットから抽出した 5 種類のメトリクス（コンポーネント名，優先度，報告者，担当者，概要の文字数）を用いる．指定期間内にタスクが完了するか否かの予測に目的変数を離散値として予測可能なランダムフォレスト法を用いる[Weiss07][Hewett09][Zhang13]．これらのメトリクスは，チケットに関連するメトリクスと人に関連するメトリクスに分類される．この二つの分類について順に説明する．

チケットに関連するメトリクス

開発者は実施すべき作業，修正すべきバグなどの一つ一つのタスクを Redmine もしくは Trac のプロジェクトにチケットとして登録する．そして，このチケットに記録されている情報を基に，作業や修正を行う．開発者がタスクを迅速に完了するために，チケットに記録されている情報は重要である．タスク完了時間予測プラグインでは，このチケットに関

連するメトリクス群から，コンポーネント名，優先度，概要の文字数の情報を抽出し，予測モデルの構築に用いる．なお，コンポーネント名および優先度は名義尺度，概要の文字数は間隔尺度である．

人に関連するメトリクス

タスクに取り組むプロセスにおいて，報告者，管理者，担当者は主要な役割を担っている．いずれかの役割を担う開発者の能力や経験により，タスクの完了時間は異なると考えられる．例えば，多くのタスクを報告している開発者が報告者の場合，タスクの完了に必要な詳細な情報をチケットに登録することにより，担当者は迅速にタスクに取りかかることができるため，タスク完了時間は短くなると考えられる．この人に関連するメトリクス群から，報告者と担当者の情報を抽出し，予測モデルの構築に用いる．なお，報告者と担当者はともに名義尺度である．

また,タスク完了時間予測プラグインでは,タスクが完了するまでの期間を,“1 日以内”,“3 日以内”,“1 週間以内”,“2 週間以内”,“3 週間以内”,“1 ヶ月以内”,“2 ヶ月以内”,“3 ヶ月以内”,“4 ヶ月以内”,“5 ヶ月以内”,“6 ヶ月以内”,“7 ヶ月以内”,“8 ヶ月以内”,“9 ヶ月以内”,“10 ヶ月以内”,“11 ヶ月以内”,“1 年以内”,“1 年以上”と細かく分類して予測する．指定期間内にタスクが完了するか否かの予測に目的変数を離散値として予測可能なランダムフォレスト法を用いる．

ランダムフォレスト法は,回帰木を用いて集団学習を行う手法であり,2001 年に Breiman によって提案された．モデル構築用のデータセットに対して繰り返しランダムサンプリング(復元抽出)を行い，得られたサンプル群から多数の回帰木を構築する．そして，各回帰木の出力の平均により最終的な予測結果を得る．従来の集団学習ではモデル構築時に全ての説明変数を用いていたのに対して，ランダムフォレスト法では無作為に選択された説明変数を用いている．

既にタスクのステータスが完了済みとなっているチケットのデータを学習データ，タスクのステータスが新規で，タスク完了時間を見積もるために選択されたチケットのデータをテストデータとし，ランダムフォレスト法を用いて，それらのチケットのタスクが何日以内に完了するかの予測を行う．

（プラグイン画面）

タスク担当者推薦プラグインと同様に，一覧表示されたタスクリストから任意のタスクを選択し実行すると，図 3-8 のように予測結果が出力される．①は，選択されたタスクの ID を示している．issue_id ボタンを押すことで，ID の昇順・降順を変更できる．②は，①の ID のタスクが完了する期間の予測結果をそれぞれ示している.予測完了時間ボタンを押すことで，時間の昇順・降順を変更できる．

図 3-8 タスク完了時間予測プラグインの出力結果の例

- **バグモジュール予測プラグイン**

（概要）

バグモジュール予測プラグインは，過去の不具合情報から，不具合が存在すると考えられるモジュールを予測するプラグインである．バグモジュール予測プラグインを利用することによって，どのモジュールに不具合が含まれやすいのかを確認することができるため，不具合が含まれると考えられるモジュールに対して，重点的にテストを行うことなどによって，効率的なテストを行うことができる．

（アルゴリズム）

バグを含む可能性の高いファイルに対して重点的にテストを行うために，バグモジュール予測プラグインでは，リポジトリに含まれる対象ファイルすべてのバグを含む可能性を明らかにする[S'liwerski05][Nagappan06][Kim08]．そのため，バグモジュール予測プラグインの入力となるのは，リポジトリのパスであり，バグモジュール予測プラグインの出力となるのは，各ファイルがバグを含む可能性である．ここで，ファイルがバグを含む可能性とは，ファイルにバグが存在する確率であり，0 から 100 のパーセンテージの値で表される．

バグモジュール予測プラグインにおいて，入力されたリポジトリに含まれるすべてのファイルのバグが存在する確率を求めるために行う処理は以下の通りである．

① 不具合修正履歴とソースコードの変更履歴から，不具合の情報とその不具合を修正するために変更されたファイルを抽出する
② すべてのリビジョンにおける各ファイルのコードメトリクスを計測する
③ 特定のファイルのメトリクスの値と不具合の有無をランダムフォレスト法によって学習する
④ 入力されたリポジトリに含まれるファイルすべてに対してメトリクスを計測する
⑤ メトリクス値を説明変数として，不具合を含む可能性を各ファイルに対して求める

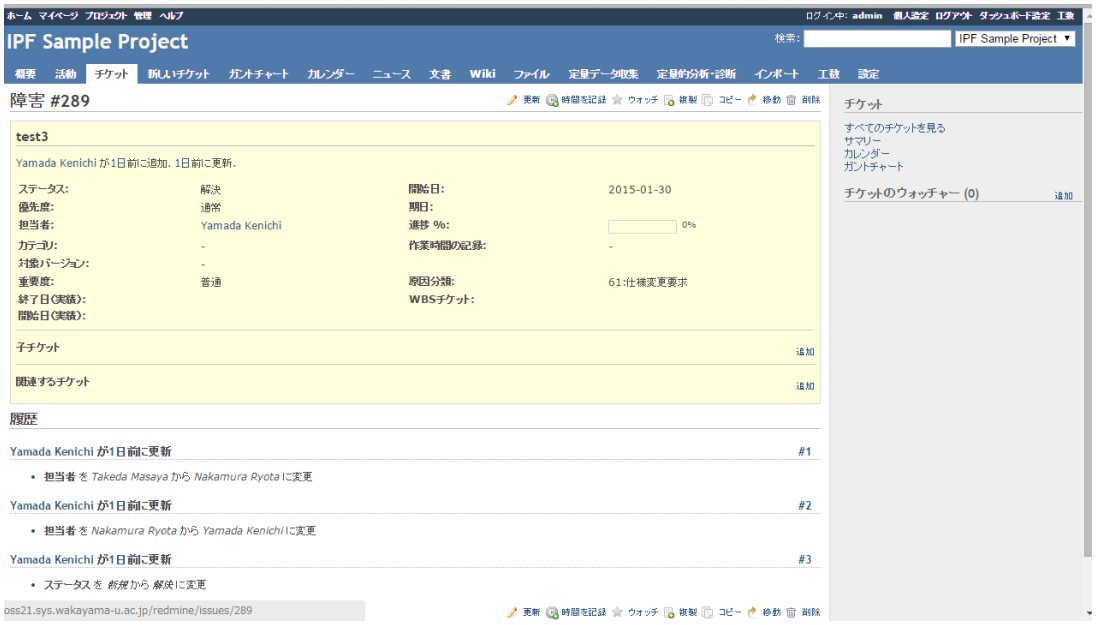


図 3-9 不具合修正履歴の例（Redmine 版）

図 3-10　不具合修正履歴の例（Trac 版）

まず，①の不具合修正履歴とソースコードの変更履歴から，不具合の情報とその不具合を修正するために変更されたファイルを抽出する処理について説明する．不具合修正履歴とは，Redmine であれば**図 3-9** に示すようなトラッカー種別が「障害」のチケット情報であり，Trac であれば**図 3-10** に示すような分類が「障害」のチケット情報を指す．不具合修正履歴情報には，不具合の症状や，不具合が報告された日時．完了した日時などの情報が記されている．

また，ソースコードの変更履歴とは，Git などのバージョン管理システムのログ情報を指しており，ソースコードの変更履歴の例を**図 3-**に示す．ソースコードの変更履歴からどの開発者がいつどのファイルをどれだけ変更したのかを明らかにすることができる．

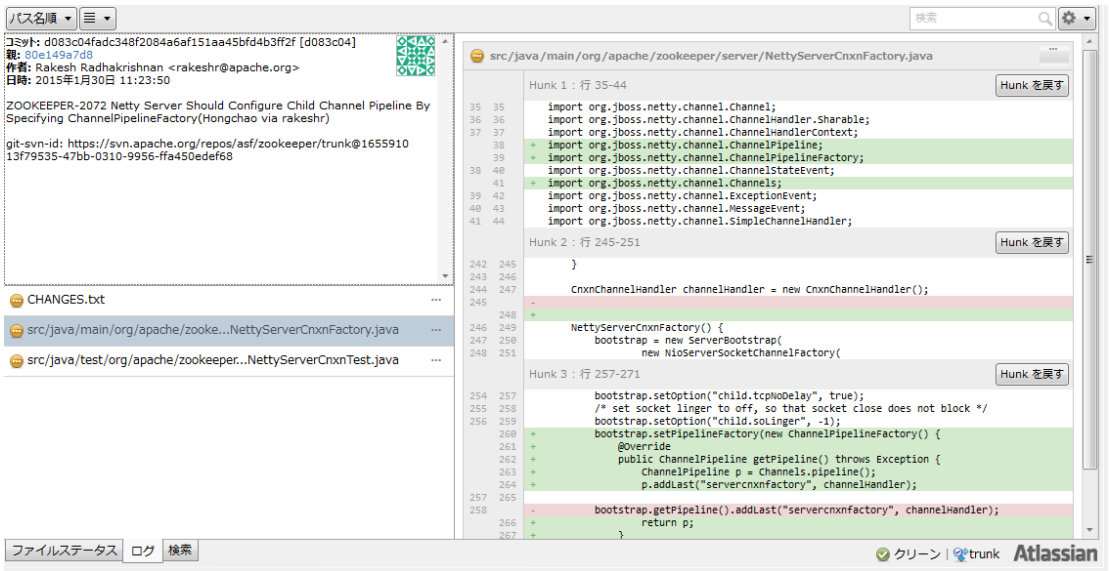


図 3-11　ソースコードの変更履歴の例

①の処理では，不具合修正履歴情報とソースコードの変更履歴情報をひも付ける作業を行う．不具合修正履歴情報とソースコードの変更履歴情報をひも付ける作業とは，どの不具合がどの時点のファイルに存在していたのかを明らかにするための処理であり，ソースコードの変更履歴から不具合の修正を目的としたソースコードの変更を特定し，その後，修正時に変更されたコードがいつ埋め込まれたのかを調べることで欠陥の混入時期を特定する．この不具合修正履歴情報とソースコードの変更履歴情報をひも付ける作業は，SZZ アルゴリズムと呼ばれており，Sliwerski らによって提案されたリポジトリマイニング手法である．

（プラグイン画面）

予測結果の出力画面の例を図 3-12 に示す.ファイルパスの昇順に一覧表示されている．①はファイルパスを示している．ファイルパスボタンを押すことでファイルパスの昇順・降順を変更できる．②は①のファイルにバグが含まれている確率をそれぞれパーセンテージで示している．バグ含有確率(%)ボタンを押すことで確率値の昇順･降順を変更できる．③は，一覧表示件数を変更するドロップダウンボタンである．10 件，25 件，50 件，100 件と変更できる．④は，検索窓である．ファイルパスとバグ含有確率の両方に対して単語や数値を検索できる．⑤は，表示ページの変更ボタンである．

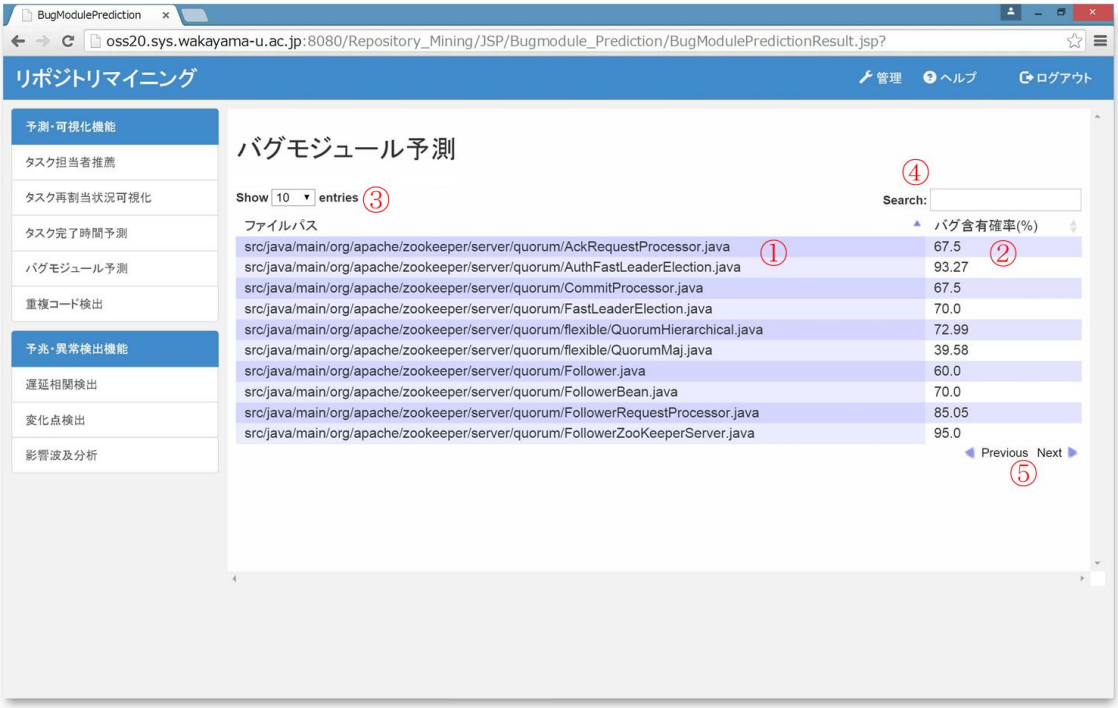


図 3-12 バグモジュール予測プラグインの出力結果の例

- **重複コード検出プラグイン**

（概要）

重複コード検出プラグインは，現在開発中のソフトウェアのソースコードに含まれる重複した箇所を抽出するプラグインである．ソースコードに含まれる重複した箇所を知ることは，保守性の高いソフトウェアを目指しリファクタリングを行う際などに役立つ．

（アルゴリズム）

重複コード検出プラグインでは，ソースコードを入力すると，重複するコード片を含むファイルセット，および，重複しているコード片を出力する．解析対象となるソースコードが大規模になればなるほど，重複するコード片のペアを抽出するのに時間がかかるため，単純に文字列を比較するような単一文字列検索アルゴリズムでは，現実的な時間で解析が終了しないと考えられる．そのため，効率的にソースコード中に含まれる文字列を比較するアルゴリズムを用いる必要がある．重複コード検出プラグインでは，ソースコード中に含まれる文字列を比較するアルゴリズムとして，ラビンカープ文字列検索アルゴリズム（ラビンカープ法）を用いる．ラビンカープ法は，ハッシュ関数を利用した文字列検索アルゴリズムである[Ducasse99][Lin14][Monda114]．

単一文字列検索アルゴリズムは，単純な文字列一致検索アルゴリズムであり，検索する文字列のパターンが存在する可能性をもつ全ての位置を順番にチェックする方法である．以下に単一文字列検索アルゴリズムの処理を示す．

① 検索対象文字列中での比較開始位置を決める
② 文字列の比較を行い，不一致が確定した時点で打ち切る
③ ②を検索対象文字列のすべての位置について行う

（プラグイン画面）

重複コード検出プラグインを呼び出すと，ソースコードから重複するコード片の情報が図 3-13 のように出力される．①は，類似するコード片のペアの ID を示している．②は，類似するコード片を含むファイルのファイルパスを示している．③は，それぞれのファイルの類似するコード片を含む箇所が始まる行数を示している．④は，類似する部分のコード行数を示している．⑤は，重複しているコード片の表示を行うボタンである．

図 3-13 における⑤の重複コード部分確認ボタンを押すと，図 3-14 のような画面が出力される．この画面において，背景が濃くなっている部分は，重複している箇所のソースコードである．このように，重複コード検出プラグインを用いることによって，重複しているコード片を簡単に確認することができる．

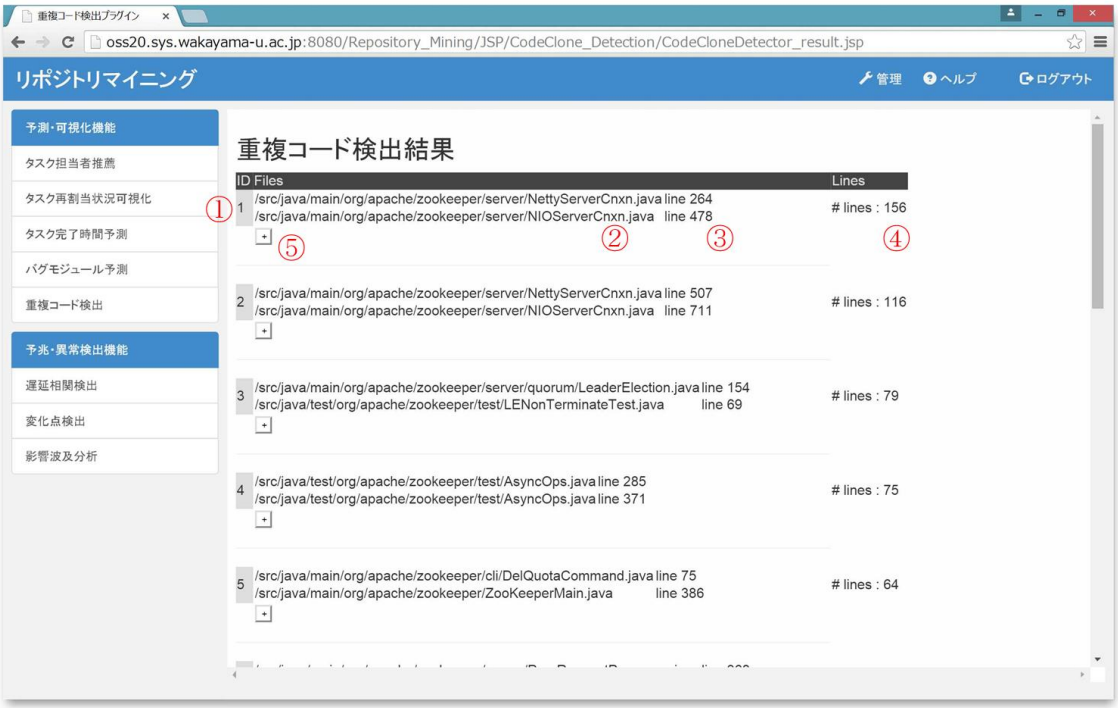


図 3-13 重複コード検出プラグインの出力結果の例

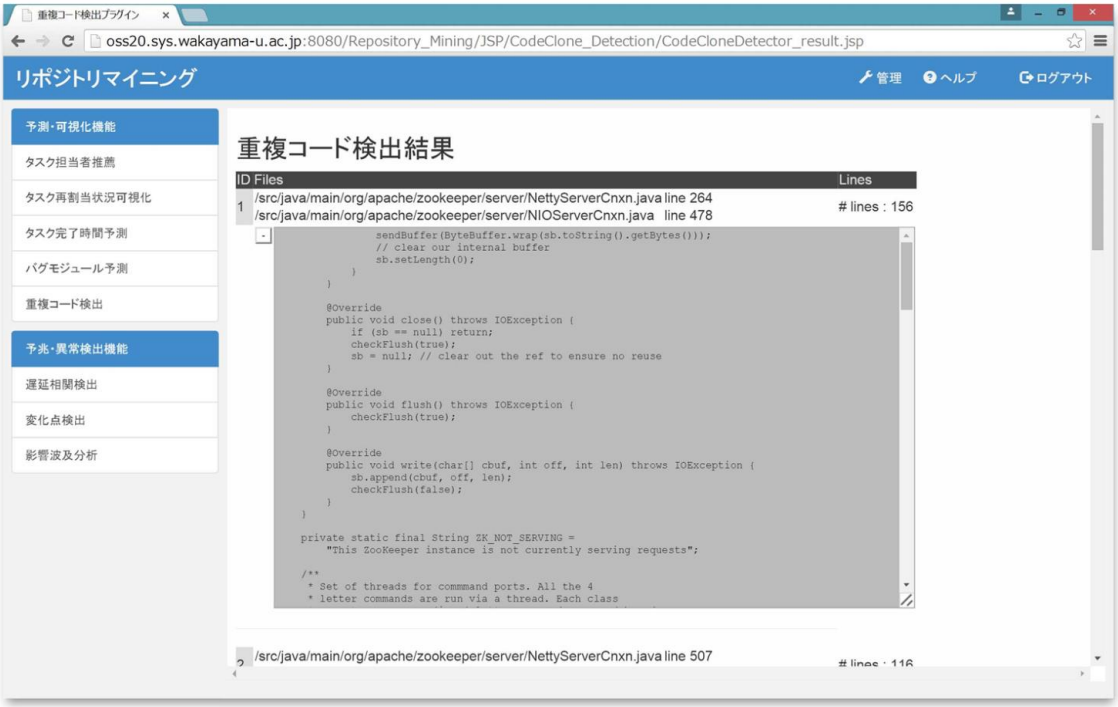


図 3-14 重複しているコード片の確認

③インストーラ開発

実装したプラグインの導入を支援するためのインストーラ（図 3-15）を開発した．Redmine 版・Trac 版それぞれが用意されている．EPX-X がすでにインストールされている環境において機能する．なお，本インストーラは，リポジトリマイニング・プラグインに加え，プロアクティブマイニング・プラグインのインストーラも兼ねている．

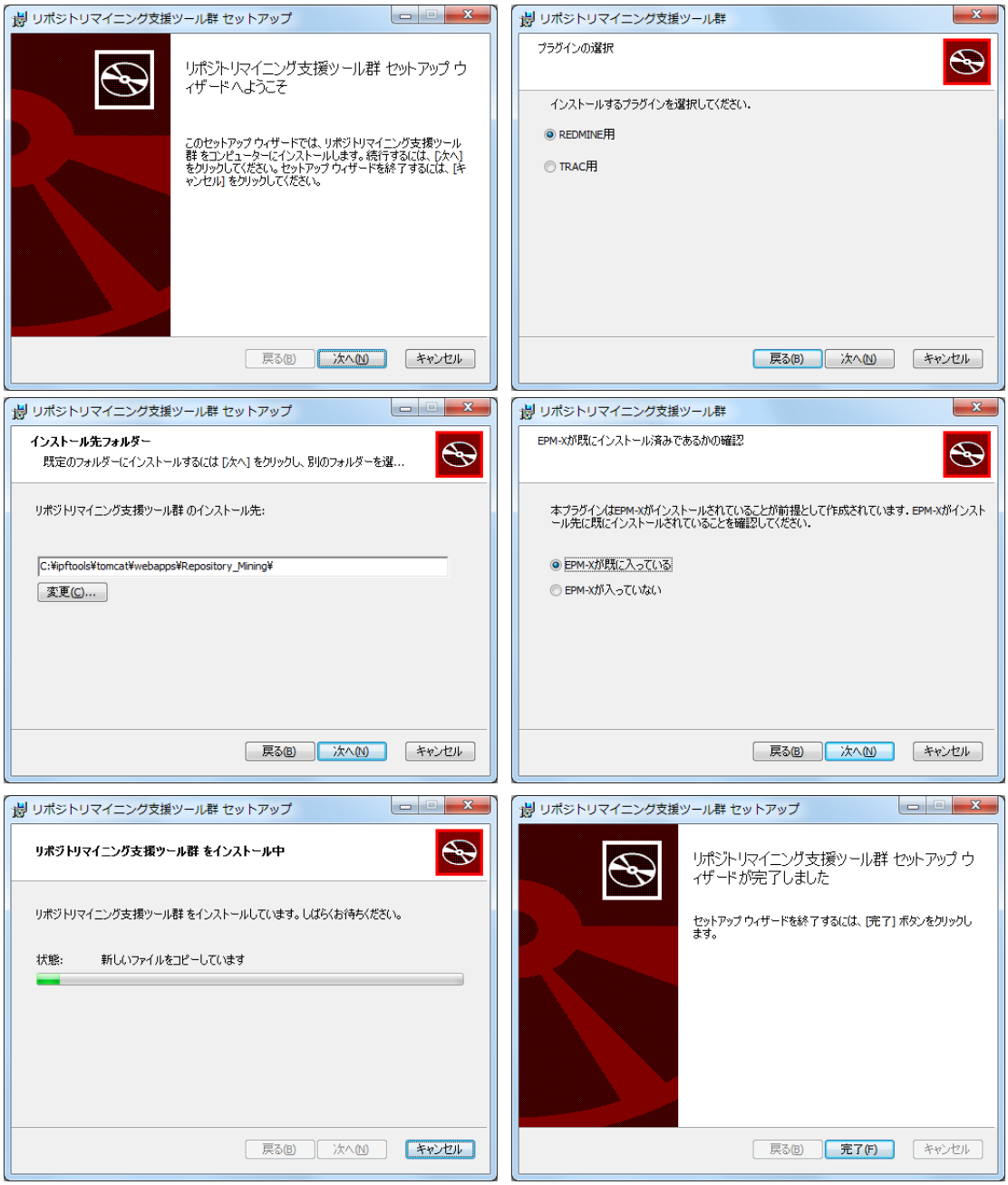


図 3-15 インストーラ画面

3.2.3 課題とその対応

プラグインの実装作業自体には大きな問題や課題は発生しなかった．しかし，学生アルバイトの多くに実装補助したものであり，コード規約の遵守や再利用性の観点に対する意識は低くならざるを得なかった．プラグイン自体は時間をかけてテストしているので，インストールして利用する限りにおいてはある程度の安定した動作が期待できるが，分析対象のリポジトリサイズによっては我々の想定を超えるものがある場合には上手く動作しない可能性がある．また，公開して配布する予定のソースコードを改変して利用する場合にも，コードの再利用性については課題がある．今後は，ソースコードも含めオープンソースライセンス（GPL）として公開し，利用者からのフィードバックを求めたい．また，本研究終了後は人的リソースを確保することができないので，ソースコードに問題を見つけて頂いた場合には，なるべく修正パッチを提供していただくよう協力を呼びかける予定である．

3.3 研究目標 3「プロアクティブマイニング技術の調査と選定（SG-B-1)」

3.3.1 当初の想定

(1) 研究内容

（内容）

過去 10 年分の文献調査を通じて既存プロアクティブマイニング技術を分類し,有用性が高くかつプラグイン化可能なものを明らかにし，その中から 3 件のプロアクティブマイニング技術を選定した．

（想定される課題と解決策）

ソフトウェア工学の分野ではほとんど適用事例がないため，他分野の調査を中心に技術の選定を進めた．海外研究協力者とも適宜相談しながら選定作業を進めるが，作業が捗らないことも予想された．その場合，すでに実用されている技術（故障予兆検出やネットワーク異常検出など）の分野の研究者に直接コンタクトをとり，技術指導を受けることも視野に入れた．

(2) 当初の到達目標と期待される効果

（当初の到達目標）

文献調査を幅広く実施し，ソフトウェア開発支援に応用可能な他分野のプロアクティブマイニング技術を明らかにする(G-B-1)．

（期待される効果）

他分野で適用が進んでいるプロアクティブマイニング技術の中から，プロジェクト管理支援にも応用可能なプロアクティブマイニング技術を選定できる．

3.3.2 研究プロセスと成果

(1) 研究プロセス

以下の順序で研究を進めた.

1. 文献調査
2. 技術分類
3. 技術選定

(2) 具体的な研究成果（結果）

①文献調査

過去 10 年分の文献調査を通じて既存プロアクティブマイニング技術を分類し,有用性が高くかつプラグイン化可能なもの 3 件を選定した. ソフトウェア工学の分野ではほとんど適用事例がないため, 他分野の調査を中心に技術の選定を進めた. 主に, 以下の国際会議論文集および論文誌から数十編を調査した.

- 国際会議論文集
  - ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
  - SIAM International Conference on Data Mining
  - Very Large Data Bases (VLDB) Conference
  - ACM Internet Measurement Conference (IMC)
- 論文誌
  - Data Mining and Knowledge Discovery
  - IEEE Transaction on Information Theory
  - Machine Learning

②技術分類

技術選定にあたっては, 山西の技術解説[山西 09]を参考にして以下の技術的観点からプロアクティブマイニング技術を分類した.

- 外れ値検出[武田 00][Yamanishi04][Neal99]
- 変化点検出[赤池 94,赤池 95][Yamanishi02][Takeuchi06]
- 異常行動検出[Baum70][Krichevsky81][Rissanen84][Viterbi06][Yamanishi07]

③技術選定

調査結果に基づき, 実務志向で有用性の高いもの（ただし, 研究実施期間中に実現可能性なもの）として 3 種類のプロアクティブマイニング技術を選定した.

- 遅延相関検出[竹内 08][Yamatani14][山谷 15]
- 変化点検出[Robles06][山西 09][久木田 15]
- 影響波及分析[S ´liwerski05][Zimmermann 05][Acharya11]

3.3.3 課題とその対応

ネットワークの異常検知やプラントの故障検知技術として利用されているプロアクティブマイニング技術は, ソフトウェア工学分野においての適用事例がほどんどなく, どのよ

うな状況で利用できそうなのかなど，具体的なシナリオを構築するのが難しい場合があった．幸いにも，遅延相関分析については企業実務者からの評判が高く，実装すれば役立つことがある程度の確信をもって取り組むことができた．

一方，変化点検出については，経済学分野で用いられるような指標（株価のトレンド変換を判断するための指標など）との明確な違いとそのメリットが我々にもまだはっきりとは分からない部分がある．今後は，対象となるメトリクスや検証実験の対象データを増やすとともに，他の指標との比較実験を通じて変化点検出アルゴリズムをソフトウェア工学分野で適用するメリットをより明確にする必要がある．特に，適用対象となる工程やメトリクスについて企業実務者を交えて議論することが今後は重要になると思われる．

影響波及分析については，当初異常行動検出アルゴリズムを実装予定であったが，実務者の強いニーズを受け，プロアクティブマイニング技術として選定することとした．今後は，必ずしも本調査によってリストアップされた技術や分類された技術にこだわらず，現場ニーズに即した技術選定をおこなう必要があると考えている．

3.4 研究目標 4「プロアクティブマイニング・プラグイン 3 件の開発（SG-B-2)」

3.4.1 当初の想定

(1) 研究内容

（内容）

まず，SG-B-1 の技術調査および選定結果に基づき，プロアクティブマイニングのアルゴリズムおよび入出力のためのデータ構造の仕様を決定した．次に，決定した仕様からプロアクティブマイニング技術を EPM-X の拡張機能としてプラグイン化できることを確かめるために，実装の容易な遅延相関検出アルゴリズムを例にとりプラグインのプロトタイプを作成した．

次に，Redmine 版・Trac 版それぞれ 3 件計 6 件のプラグインを順に実装し，オープンソースプロジェクトデータを用いて，実装したプラグインが想定通りに解析結果を出力できるかどうかをテストした．

（想定される課題と解決策）

1 プラグインあたり実装 1 ヶ月・テスト 2 ヶ月，計 3 ヵ月での完成を予定した．プロアクティブマイニングを実現するためのコアとなるアルゴリズムについてはソフトウェア工学の分野での適用事例がほとんどないため，テスト工数を多めに見積もった．プラグイン化のみでは技術の有用性が保証できないことが想定されるため，(SG-B-3)において OSS プロジェクトデータを用いた検証実験を行うことで課題を解決することとした．

(2) 当初の到達目標と期待される効果

（当初の到達目標）

プロアクティブマイニング・プラグインを 3 件（Redmine 版・Trac 版の計 6 件）開発する(SG-B-2)．

（期待される効果）
他分野での適用が進んでいるプロアクティブマイニング技術がプロジェクト管理支援のためのプラグインとして提供可能になる.

3.4.2 研究プロセスと成果

(1) 研究プロセス

以下の順序で研究を進めた.

1. 仕様決定・プロトタイプ作成(SG-B-2-1)
2. プラグイン実装・テスト(SG-B-2-2/SG-B-2-3)

(2) 具体的な研究成果（結果）

①仕様決定・プロトタイプ作成

選定した技術の詳細が記述されている文献に基づき，プロアクティブマイニングのアルゴリズムおよび入出力のためのデータ構造の仕様を決定した．また，プラグインを動作させるための基盤部品を実装し，遅延相関検出プラグインをプロトタイプとして実装した．実装上，性能上問題ないことを確認した.

②プラグイン実装・テスト

決定した仕様に基づきプラグインをそれぞれ実装した．Redmine および Trac で動作することを確かめるためのテストを実施した． 開発した 3 種類のプロアクティブマイニング・プラグインの詳細を以下に説明する.

- **影響波及分析プラグイン**

（概要）

遅延相関検出プラグインは，リポジトリから計測されるメトリクス間の時間のずれる関係を明らかにするプラグインである．メトリクス間の時間のずれる関係とは，事象 A が発生すると一定期間後に事象 B が発生するといった関係を指し，メトリクス間の時間のずれる関係を把握することによって，今後プロジェクトに起こると考えられる事象を知ることができ，ソフトウェア開発・保守の品質および生産性の向上が見込まれる.

（アルゴリズム）

遅延相関検出プラグインは，時間的な遅延を伴う変数間の関係を明らかにする遅延相関分析手法をベースとしたものである．遅延相関分析手法は，一定期間の説明変数の値の変化が一定期間後の目的変数の値に影響を与える，という関係を検出するために相関分析を拡張したものである．遅延相関分析を行うことで，「開発者数が 2 か月多い状態が続くと，2 か月後に完了したタスク数が多くなる」といった関係を抽出できる[竹内 08][Yamatani14][山谷 15].

遅延相関検出プラグインにおける処理の手順を以下に示す.

① ソフトウェア開発データからメトリクスを時系列データとして計測する
② 説明変数および目的変数にメトリクスを割当てる
③ 説明変数の移動平均を求める
④ 目的変数の値をずらして相関係数を求める
⑤ ③から④の処理をすべてのパラメータに対して行う
⑥ ②から⑤の処理をすべてのメトリクスの組み合わせについて行う

まず，①のソフトウェア開発データからメトリクスを時系列データとして計測する処理では，ソースコードを管理しているバージョン管理システムである Git および，チケット管理システムである Redmine あるいは Trac から，表 3-1 に示すメトリクスを計測する.

続いて，②の説明変数および目的変数にメトリクスを割当てる処理（図 3-16）では，説明変数および目的変数に表 3-1 メトリクス一覧に示すメトリクスの一つずつを割当てる. 遅延相関検出プラグインでは，ある二つのメトリクス間の関係を分析するため，説明変数および目的変数に一つずつメトリクスを割当てる.ここで説明変数となるメトリクスとは，メトリクス A が増加すると一定期間後にメトリクス B が増加するという関係におけるメトリクス A を指している. また，目的変数となるメトリクスとは，この関係におけるメトリクス B を指している.

表 3-1 メトリクス一覧

メトリクス名	説明
モジュールの数	モジュール（クラス）の数
行数	論理的コード行数
複雑度	McCabe's Cyclomatic 複雑度
コメント行数	コメントアウトの行数
コメントの割合	1 コメントあたりのコード行数
結合度	モジュール間結合度
障害数	チケット管理システムで記録されている障害の数
変更回数	ソフトウェアに変更を加えた回数
変更人数	ソフトウェアに変更を加えた開発者の数

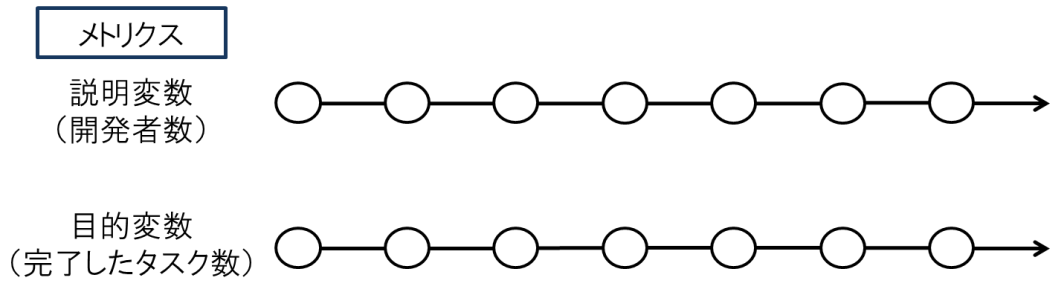


図 3-16 遅延相関検出アルゴリズム：②の処理

続いて③の説明変数の移動平均を求める処理では，②の処理で割り当てられたメトリクスの時系列データの移動平均を求める．ここで，説明変数の移動平均をとるウィンドウサイズをパラメータとして定義する．

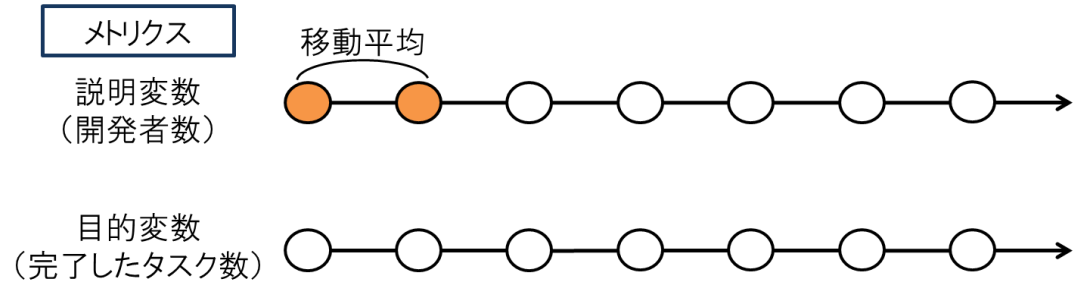


図 3-17　遅延相関検出アルゴリズム：③の処理

図 3-に説明変数の移動平均を求める処理のイメージを示す．図 3-に示す例では，説明変数の移動平均のウィンドウサイズは 2 であり，ウィンドウサイズが 2 であるということは，2 か月間の開発者数の平均を示している．

また，④の目的変数の値をずらして相関係数を求める処理では，③の処理で移動平均を求めた説明変数の時系列データと目的変数の時系列データに対して，目的変数の値を一定期間ずらして相関分析を行う．　図 3-に④の処理のイメージを示す．

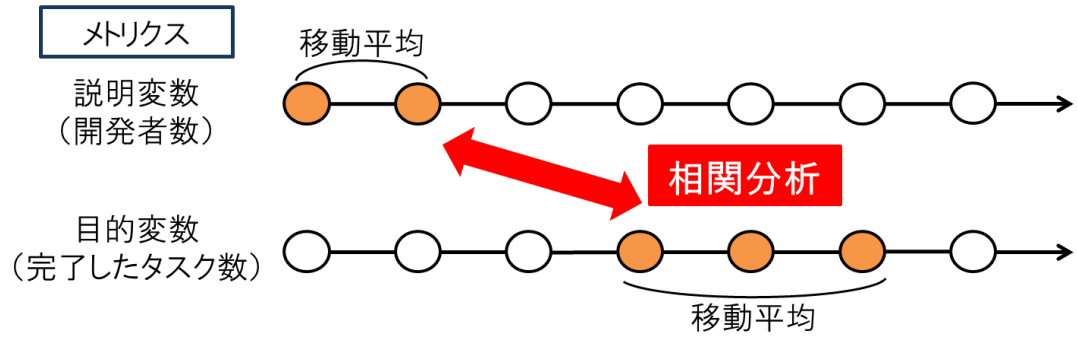


図 3-18　遅延相関検出アルゴリズム：④の処理

図 3-に示すように，目的変数の値をずらして相関分析を行うことで，時間的な遅延を伴う関係を明らかにすることができる．また，目的変数の値をずらす大きさをパラメータとして定義する．図 3-に示す例では，目的変数の値をずらす大きさは 2 となっている．つまり，図 3-に示す例では，「開発者数が 2 か月多い状態が続くと，2 か月後に完了したタスク数が多くなる」という関係を示している．

さらに，⑤の処理では，③と④の処理をすべてのパラメータの組み合わせについて行う．すべてのパラメータの組み合わせとは，移動平均をとるウィンドウサイズのパラメータお

よび，目的変数の値をずらす大きさを指しており，遅延相関検出プラグインでは，ウィンドウサイズの大きさの値は 1 から 3，遅延の大きさを示す値は 0 から 3 となっている．

ここで，ウィンドウサイズの大きさが 1，遅延の大きさが 0 というパラメータの組み合わせでは，遅延を考慮しない従来の相関分析と同じ結果になる．このことから，遅延相関検出アルゴリズムは，従来の相関分析アルゴリズムを含んでいることがわかる．

さらに，⑥の処理では，②から⑤の処理をすべてのメトリクスの組み合わせについて行う．遅延相関検出プラグインでは，9 種類のメトリクスを用いるため，72 通りのメトリクスの組み合わせについて分析を行う．そして，各メトリクスの組み合わせの中から最も相関係数の絶対値が大きくなるパラメータの組み合わせおよび，その際の相関係数の値を求める．

このように，すべてのメトリクスおよび，すべてのパラメータの組み合わせについて相関分析を行い，最も相関が強くなる場合を出力するため，分析者は，パラメータの値の組み合わせを気にすることなく分析することができる．

（プラグイン画面）

図 3-19 は遅延相関検出プラグインの出力結果の例を示している．①は説明変数として用いたメトリクスの名称を示している．②は目的変数として用いたメトリクスの名称を示している．③は遅延して現れる関係を示している．④は遅延相関係数を示している．①～④のいずれも，列名のボタンを押すことで，昇順・降順を変更できる．⑤は分析期間を示している．⑥は一覧表示件数を変更するドロップダウンボタンである．10 件，25 件，50 件，100 件と変更できる．⑦は検索窓である．タスクの ID と題名のテキストの両方に対して単語や数値を検索できる．⑧は表示ページの変更ボタンである．

図 3-19 遅延相関検出プラグインの出力結果の例

● **変化点検出プラグイン**

（概要）

変化点検出プラグインは，ソフトウェア開発における開発状況の変化を早期に発見するために，変化点検出アルゴリズムに基づいてリポジトリから計測されるメトリクスの値の解析を行う．ソフトウェア開発における開発状況の変化を早期に発見することによって，（潜在的に）重大な問題に素早く対応できるようになると期待できる．

（アルゴリズム）

変化点検出プラグインでは，AR モデルのオンライン忘却型学習アルゴリズム SDAR を用いた時系列モデルの 2 段階学習に基づいた変化点検出アルゴリズムを用いる．時系列データが入力されると，第一段階の学習を行う．第一段階の学習では，AR モデルで予測される値と実際の値との差分である外れ値スコアを求める．次に，この外れ値スコアを平滑化する．平滑化とは，直近の数時点の外れ値スコアを平均することである．この平滑化によって，ノイズに反応した外れ値スコアを除去する．次に，平滑化されたスコアの時系列データに対して二段階目の学習を行い，外れ値スコアをもう一度求める．そして最後に，平滑後の外れ値スコアを再度平滑化して，最終的な各時点における変化点スコアを求める．平滑化によりノイズを除去することや，二段階学習をすることによって，本質的な変動のみを検出できる[Robles06][山西 09][久木田 15]．

ここで変化点について，もう少し詳しく説明する．変化点とは，時系列データの振る舞いの急激な変わり目を指す．変化点によく似た性質を持つ時系列データの急激な変わり目として，外れ値が挙げられる．外れ値のイメージを図 3-に示す．

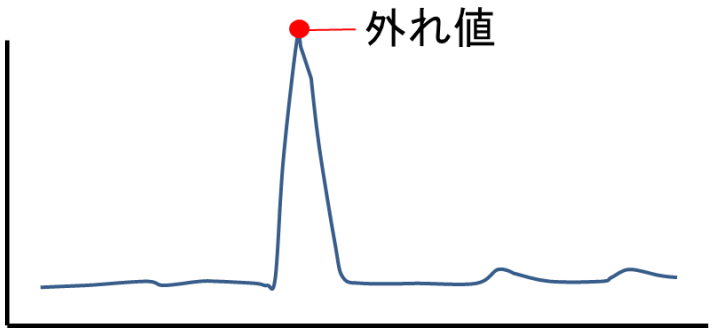


図 3-20　外れ値のイメージ

図 3-からわかるように，外れ値の前後では，時系列データの性質は変わっておらず，外れ値となる点だけ非常に高い値をとっていることがわかる．ソフトウェア開発データには，外れ値が混入しやすいものとなっている．例えば，開発データを入力する際の人為的なミスや，リリースなどのイベントによる影響を受けることにより，ソフトウェア開発データに外れ値が混入しやすくなっている．

変化点のイメージを**図 3-**に示す.

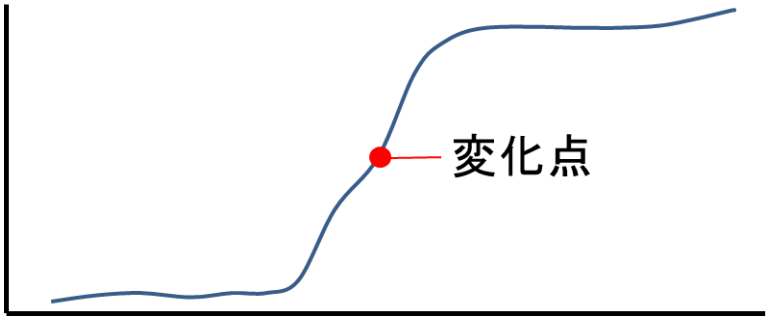


図 3-21　変化点のイメージ

図 3-に示すように，変化点の前後で時系列データの性質が変化している．プロジェクトの開発状況の変化を検出するためには，一点だけが変化する外れ値ではなく，時系列データの性質が変化する点を明らかにしたい．

そこで変化点検出プラグインでは，ChangeFinder を用いて変化点検出を行う．ChangeFinder は，AR モデルのオンライン忘却型学習アルゴリズム SDAR を用いた時系列モデルの 2 段階学習に基づいた変化点検出アルゴリズムであり，山西らによって提案されている．

以下に ChangeFinder の手順について説明する．まず，時系列データが入力されると，第一段階の学習を行う．第一段階の学習では，AR モデルで予測される値と実際の値との差分である外れ値スコアを求める．次に，この外れ値スコアを平滑化する．平滑化とは，直近の数時点の外れ値スコアを平均することである．この平滑化によって，ノイズに反応した外れ値スコアを除去する．

そして，平滑化されたスコアの時系列データに対して二段階目の学習を行い，外れ値スコアをもう一度求める．そして最後に，平滑後の外れ値スコアを再度平滑化して，最終的な各時点における変化点スコアを求める．このように，平滑化によりノイズを除去することや，二段階学習をすることによって，本質的な変動のみを検出できる．

ここまでは，ChangeFinder のアルゴリズムについて説明したが，次は，変化点検出プラグインのアルゴリズムについて説明する．変化点検出プラグインでは，まず，ソフトウェアリポジトリからメトリクスを計測する．変化点検出プラグインで用いるメトリクスの一覧を**表 3-2** に示す．LOC や複雑度といったメトリクスは，コードメトリクスであるため，メトリクス計測ツールである CCCC を用いて対象ソースのメトリクスを測定する．不具合数は，Redmine や Trac のデータベースを直接参照することによって，求めることができる．

表 3-2　メトリクス一覧

メトリクス名	説明
LOC	論理的コード行数
不具合数	Redmine / Trac に登録されている障害の数
複雑度	McCabe's Cyclomatic 複雑度

変化点検出プラグインで用いるメトリクスは月曜日を週の始まりとした週ごとのメトリクス値として扱われる．次に，それぞれのメトリクスに対して ChangeFinder を適用し，各メトリクスの変化点スコアを計算する．変化点スコアにおいても時系列データとして扱われ，最後に，メトリクス値および，変化点スコアの表示を行う．

（プラグイン画面）

図 3-22 に変化点検出プラグインの出力結果の例を示す.選択したメトリクスの時系列データや変化点スコアを確認することができる．赤枠の中の赤い線が選択したメトリクスの時系列データあり，青の線が変化点スコアとなっている．①は，データを再取得するためのボタン，②は，選択するメトリクスを示している．③は，データを 1 週間ごとにずらして閲覧するためのボタン，④は，表示期間を選択するためのプルダウンメニュー（図 3-22 では 6 か月間のデータが赤枠に表示されている），⑤は，メトリクス値および変化点スコアをグラフ上に重畳表示するかどうかを選択するためのプルダウンメニューである

図 3-22 変化点検出プラグインの出力結果の例

- **影響波及分析プラグイン**

（概要）

影響波及分析プラグインは，対象リポジトリに含まれるファイルを選択すると，同時に変更されたことのあるファイルおよび，その確率，およびファイルを修正するのにこれまでかかった時間が出力される．あるファイルを修正しようと考えている開発者は，修正にかかるコストを容易に見積もることができる．

（アルゴリズム）

影響波及分析プラグインでは，あるファイルを選択すると，同時に変更が行われる，論理的結合関係にあるファイルを変更履歴から抽出し，出力することができる．また，それらのファイルを修正するためにかかるコストも算出することができる[Zimmermann05][Acharya11]．そのためには，ソースコードの変更履歴と不具合修正履歴データの紐付けが必要となる．不具合修正履歴情報とソースコードの変更履歴情報をひも付ける作業は，SZZ アルゴリズムを用いた[Sliwerski05]．SZZ アルゴリズムは，ソースコードの変更履歴から不具合の修正を目的としたソースコードの変更を特定し，その後，修正時に変更されたコードがいつ埋め込まれたのかを調べることで欠陥の混入時期を特定するアルゴリズムである．

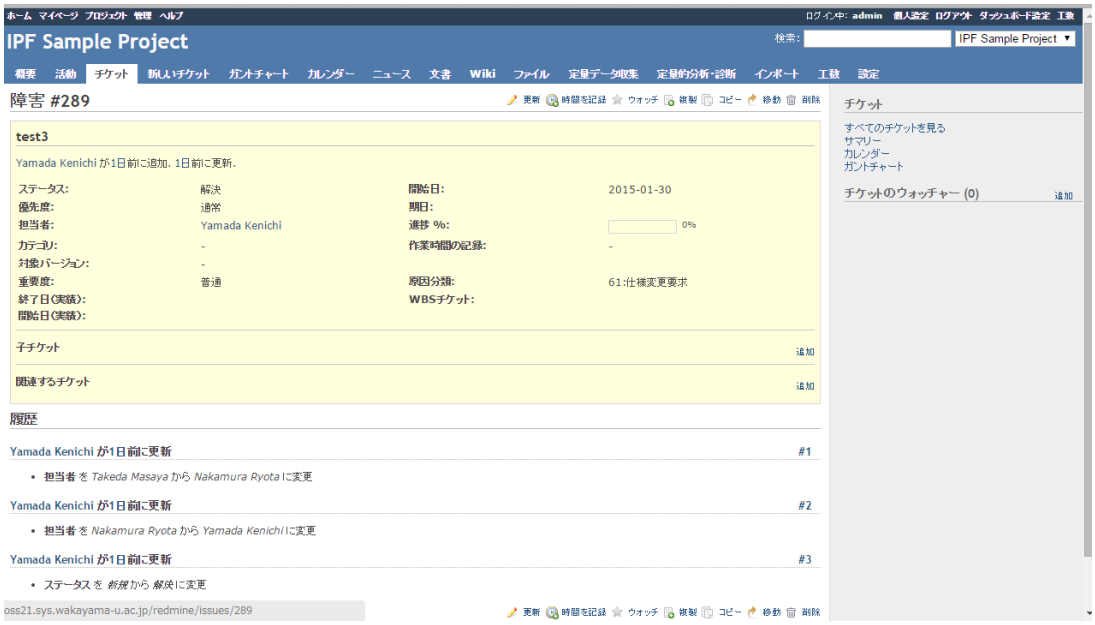


図 3-23　不具合修正履歴の例（Redmine 版）

図 3-24 不具合修正履歴の例（Trac 版）

不具合修正履歴とは，Redmine であれば図 3-23 に示すようなトラッカー種別が「障害」のチケット情報であり，Trac であれば図 3-24 に示すような分類が「障害」のチケット情報を指す．不具合修正履歴情報には，不具合の症状，不具合が報告された日時，完了した日時などの情報が記されている．

また，ソースコードの変更履歴とは，Git などのバージョン管理システムのログ情報を指している．ソースコードの変更履歴の例を図 3-25 に示す．ソースコードの変更履歴からどの開発者がいつ，どのファイルをどれだけ変更したのかを明らかにすることができる．

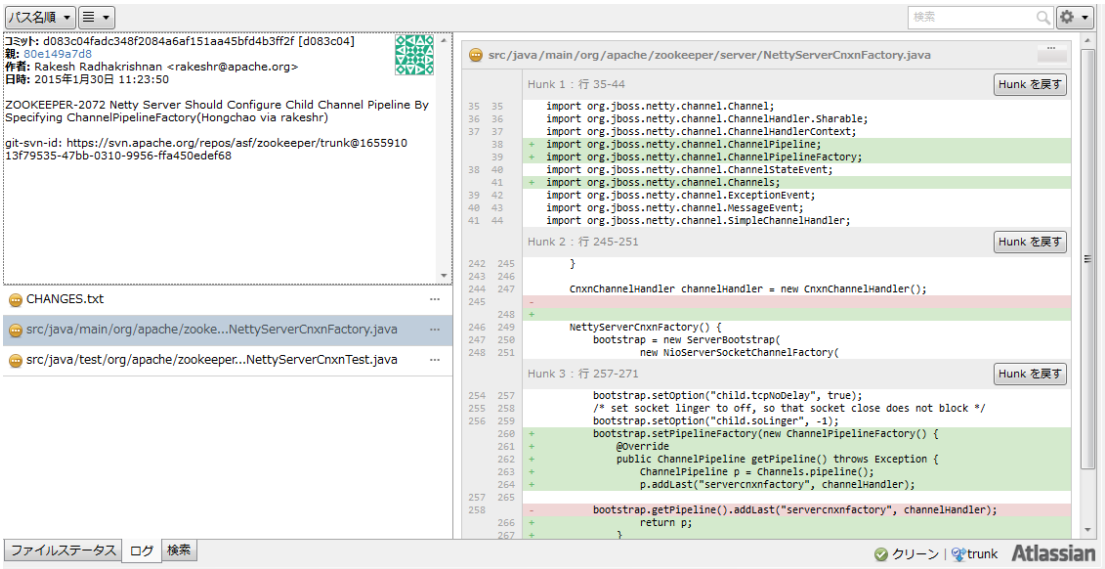


図 3-25 ソースコードの変更履歴の例

不具合修正履歴情報とソースコードの変更履歴情報をひも付ける作業とは，どの不具合がどの時点のファイルに存在していたのかを明らかにするための処理である．ソースコードの変更履歴から不具合の修正を目的としたソースコードの変更を特定し，その後，修正時に変更されたコードがいつ埋め込まれたのかを調べることで欠陥の混入時期を特定する．この不具合修正履歴情報とソースコードの変更履歴情報をひも付ける作業は，SZZ アルゴリズムと呼ばれており，Sliwerski らによって提案されたリポジトリマイニング手法である．

SZZ アルゴリズムによる処理をもう少し詳しく説明する．ソースコードの変更履歴情報には，ソースコードの変更に対する簡易的な説明が記述されている部分がある．この変更に対する簡易的な説明には，どの不具合を修正するための変更なのかなどの情報を示すために，不具合番号が記述されていることが多い．影響波及分析プラグインでは，この不具合番号が記述されているソースコードの変更説明文を取得し，不具合修正履歴情報とソースコードの変更履歴情報をひも付ける作業を行っている．そして，不具合修正履歴情報とソースコードの変更履歴情報をひも付けた情報をデータベースに格納している．データベースを参照することで，特定の不具合に対して同時変更されたファイルのパターンを抽出することができる．

また，影響波及分析プラグインでは，不具合修正履歴情報とソースコードの変更履歴情報をひも付けた情報から，同時変更ファイルパターンの発生確率およびファイルセットの変更にかかる時間の平均値を算出する．

表 3-3 ファイル同時変更履歴および変更作業時間の例

	ファイル A を含む同時変更履歴	変更作業時間（hour）
1	{ A,B,C }	50
2	{ A,K }	25
3	{ A,P,Q,R }	250
4	{ A,B,C }	86
5	{ A,X }	70
6	{ A,B,C }	110
7	{ A,Y,Z }	140
8	{ A,X }	60

表 3-3 はファイル A を含む同時変更ファイルの組み合わせと変更作業時間の例を示している．例えば 1 番目の変更履歴は，ファイル A，B，C が同時に変更されていて，これらのファイルの変更にかかったトータル時間が 50 時間であることを示している．このファイル

A，B，C が同時に変更されるというパターンは 8 回中 3 回みられるので，発生確率は，3/8=37.5%と算出できる．また，変更作業時間の平均値は，(50+86+110)/3=82 時間と算出できる．

（プラグイン画面）

影響波及分析プラグインを呼び出すと，図 3-26 に示す分析実行画面に遷移する．ディレクトリ構造がツリー形式で表示される．ファイル名のリンクをクリックすることで，分析結果画面に遷移する．

図 3-27 に影響波及分析プラグインの出力結果の例を示す．①はそれぞれ，同時変更されたファイルセットを示している．②は同時変更ファイル数を示している．③は同時変更の発生確率を示している．④はファイルセット全体の変更にかかったトータル作業時間の平均値を示している．①～④のいずれも，列名のボタンを押すことで，昇順・降順を変更できる．⑤はファイル一覧画面で選択したファイルのファイルパスを示している．⑥は検索窓である．すべてのカテゴリに対して単語や数値を検索できる．

図 3-26 影響波及分析プラグインの初期画面

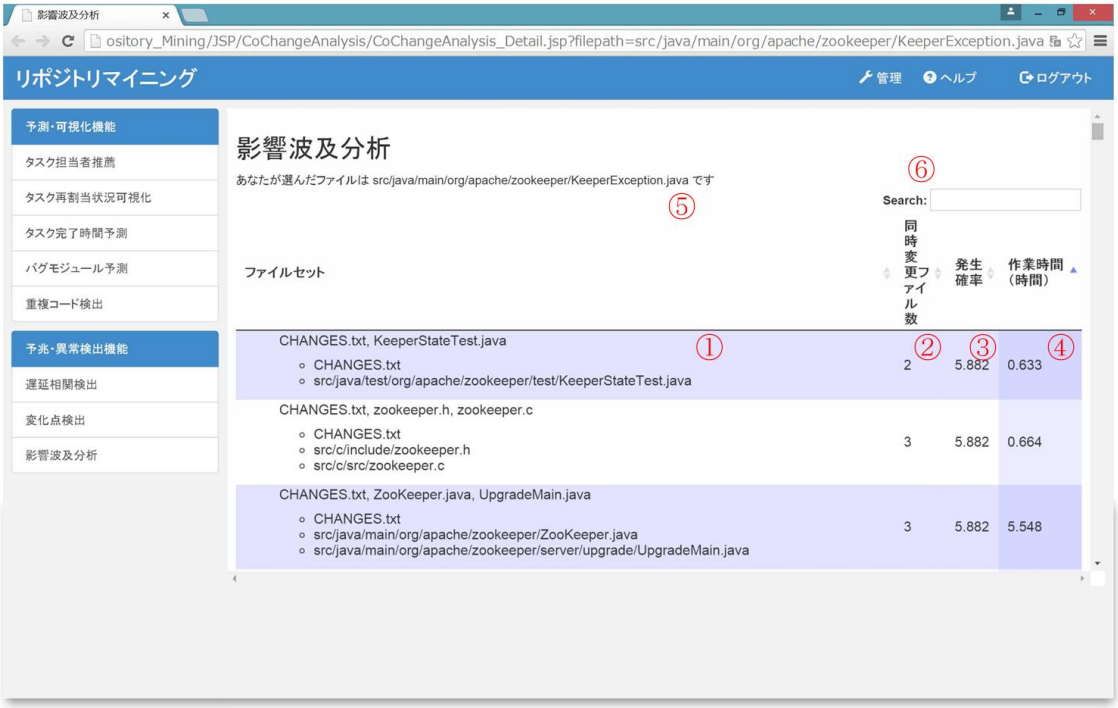


図 3-27 影響波及分析プラグインの出力結果の例

3.4.3 課題とその対応

3.3.3 項でも述べたように，影響波及分析の実装については実務者からの要望により当初予定していなかったものであった．ただし，異動行動検出のアルゴリズムを実装するのを取りやめたため，実装のスケジュール自体には問題は発生しなかった．

本研究の終盤で，リポジトリマイニング技術およびプロアクティブマイニング技術が合計 8 種類実装でき，学会やシンポジウム等で多くの実務者に具体的な成果物としてデモンストレーションを交えて技術を紹介できるようになった．その結果，新たなニーズや改善要望も多く頂けるようになってきており，3.2.3 項でも述べたように，今後の改良および新規開発のためには人的リソースの確保が重要な課題となる．また，プロアクティブマイニング技術の多くに関しては，現場での適用事例がほとんどない状況にある．今後は企業との共同研究や委託研究を加速させ，上記課題を解決していきたいと考えている．

3.5 研究目標 5「プロアクティブマイニング・プラグインの有用性検証（SG-B-3）」

3.5.1 当初の想定

(1) 研究内容

（内容）

オープンソースプロジェクトデータを用いて，実装したプロアクティブマイニング技術の有効性を検証するための実験を行う．例えば，プロジェクト進捗が停滞する予兆を早期あるいはリアルタイムに検出できるかどうかなどについて検証を行う．検証できたプラグインは，インストーラを開発し容易に利用可能な形式で公開できるようにする．

（想定される課題と解決策）

検証実験には大規模 OSS プロジェクト（Linux や Apache プロジェクトなど）のデータを用いる．大規模プロジェクトデータを用いて開発技術の検証を行ってきた実績があるため大きな困難は予想されないが，OSS 開発特有の習慣（仕様書・設計書がない，プロジェクトメンバーが流動的，厳守すべき納期がない，など）により検証結果の一般性が確保できるかどうかやや不明な点があった．その場合は，中心メンバーのほとんどが企業の開発者であるような Eclipse プロジェクトや MySQL プロジェクトなどのデータと比較しながら慎重に検証実験を行うこととした．

(2) 当初の到達目標と期待される効果

（当初の到達目標）

プロアクティブマイニング・プラグインを 3 件（Redmine 版・Trac 版の計 6 件）開発し，容易に利用可能な形式で公開する(G-B-3)．

（期待される効果）

開発したプラグインを公開することにより，プロアクティブマイニング技術を容易に利用できるようになる．また，委託契約期間終了後もプラグインをさらに発展させるための知見を，早期にかつ多方面から収集することが可能になる．

3.5.2 研究プロセスと成果

(1) 研究プロセス

以下の順序で研究を進めた．

1. 検証実験（SG-B-3-1）
2. インストーラ開発（SG-B-3-2）

(2) 具体的な研究成果（結果）

①検証実験

オープンソースプロジェクトデータを用いて，実装したプロアクティブマイニング技術の有効性を検証するための実験を行った．なお，3.6 節で述べるように，影響波及分析プラグインは，実務者へのヒアリングから判明したニーズを踏まえて既存技術[Zimmermann05]を実装したものであるため，検証実験は行わずテストのみ実施した．

- **遅延相関検出プラグインの検証実験**

（概要）

実装した遅延相関検出プラグインの実践的な適用を想定して，Eclipse Platform プロジェクトを対象としたケーススタディを通じてプラグインの検証を行った．ケーススタディでは，相関分析のみを用いた従来の分析方法よりも有用性の高い分析が行えることを目標とした．2003 年 7 月から 2012 年 6 月までの約 10 年に渡るプロジェクトデータおよび 131 種類のメトリクスを用いて，抽出可能な関係を詳細に分析した．

（結果）

時間的順序関係を考慮しない通常の相関分析では，8,515 通りの相関を求めることができる．一方，遅延相関検出手法では，17,030 通り（${}_{131}P_2$）×12（加算係数）×13（遅延係数）通りの相関を求めることができる．組み合わせが膨大なため，相関係数 0.4 以上のもののみを抽出した結果，171 件に絞ることができた．171 件のうち，通常の相関分析では抽出できない関係が 31 件存在した（表 3-4）．31 件の関係を詳細に分析した結果，「不具合の修正に多くの時間を要するようになると，3 ヶ月後に優先度の高い不具合数が増加する」といった時間的遅延を伴う関係が抽出できており，その原因理解を支援することができることがわかった．また，原因理解の支援により，望まない時間的関係が存在する場合はあらかじめ有効な予防策を講じることができるようになることがわかった[山谷 15]．

表 3-4 遅延相関検出の結果（[山谷 15]より転載）

説明変数	目的変数	加算係数	遅延係数	相関係数	p 値
新しいコミッタによる説明文の文字数	Duplicate になった不具合件数	0	7	−0.42	0.00
新しいコミッタによる説明文の文字数	Duplicate になった不具合件数	0	7	−0.42	0.00
1 コミットに対する変更したファイル数	スレッドの平均文字数	0	7	−0.48	0.00
1 コミットに対する変更したファイル数	Priority が 1（高）である不具合件数	1	3	−0.44	0.00
1 コミットに対する変更したファイル数	Wontfix になった不具合件数	0	11	−0.45	0.00
1 コミットに対する変更したファイル数	不具合を修正した人数	0	0	−0.49	0.00
コードオーナが削除した行数	新しいコミッタによる説明文の文字数	0	4	−0.42	0.00
コードオーナが削除した行数	関数・メソッドのコード行数の平均	0	5	−0.47	0.00
コードオーナが追加した行数	新しいコミッタが削除した行数	1	8	−0.43	0.00
新しいコミッタが削除した行数	不具合管理に関係があった件数	1	0	0.41	0.02
新しいコミッタが追加した行数	不具合の情報が変更された件数	1	0	0.42	0.03
新しいコミッタが追加した行数	不具合管理に関係があった件数	1	0	0.47	0.04
新しいコミッタが変更したファイル数	コミッタ数	1	9	−0.43	0.00
新しいコミッタが変更したファイル数	Priority が 1（高）である不具合件数	3	1	−0.45	0.00
コードオーナが行数を削除したコミット回数	Priority が 5（低）である不具合件数	10	12	−0.40	0.00
コードオーナが行数を追加したコミット回数	Priority が 5（低）である不具合件数	10	12	−0.4	0.00
コードオーナのコミット回数	Priority が 5（低）である不具合件数	11	12	−0.40	0.00
リプライの平均行数	スレッドの平均行数	0	2	−0.42	0.00
スレッドの平均文字数	Wontfix になった不具合件数	0	4	−0.44	0.00
平均文字数	スレッドの平均行数	0	2	−0.42	0.00
1 スレッドに対するリプライ数	Wontfix になった不具合件数	1	10	−0.42	0.00
報告されてから修正されるまでの時間	Priority が 1（高）である不具合の数	0	3	0.41	0.00
不具合修正にかかった時間	スレッドの平均行数	0	3	−0.41	0.00
不具合修正にかかった時間	**Priority が 1（高）である不具合の数**	**0**	**3**	**0.50**	**0.00**
不具合 1 件あたりの説明文の文字数	関数・メソッドのコード行数の平均	0	6	−0.48	0.00
不具合 1 件あたりの説明文の文字数	Wontfix になった不具合件数	0	11	−0.46	0.00
再割当てが発生した割合	Priority が 5（低）である不具合の数	11	11	−0.40	0.00
Severity が Trivial である不具合件数	スレッドの平均行数	0	11	−0.44	0.00
WorksForMe になった不具合件数	スレッドの平均行数	0	10	−0.47	0.00
Invalid になった不具合件数	スレッドの平均行数	0	10	−0.43	0.00
Verified になった不具合件数	スレッドの平均行数	0	12	−0.41	0.00
新しく不具合の情報を変更した人数	スレッドの平均行数	0	5	−0.41	0.00

● 変化点検出プラグインの検証実験

（概要）

実装した変化点検出プラグインの有効性を検証するために，Eclipse Platform プロジェクトの版管理システム Git，ニュースフォーラム Eclipse News Forums，不具合管理システムからそれぞれデータソースを抽出した．2002 年 1 月から 2012 年 12 月までの約 19 年にわたるデータを用いて，変化点スコアと 3 種類のメトリクス値（修正待ち不具合数，コード行数，サイクロマティック数の平均）との関係を詳細に分析した．

（結果）

分析の結果，図 3-28 のように各メトリクスの急激な変化に追従するように変化点スコアが大きな値をとることがわかり，期待した通りの出力が得られていることがわかった．また，図 3-29 に示すように，大きな変化点スコアをとる前後の時期に対する LDA の全結果（トピックの変化（図 3-30））から，急激な変化が起きた理由をある程度推定できることがわかった[久木田 15]．

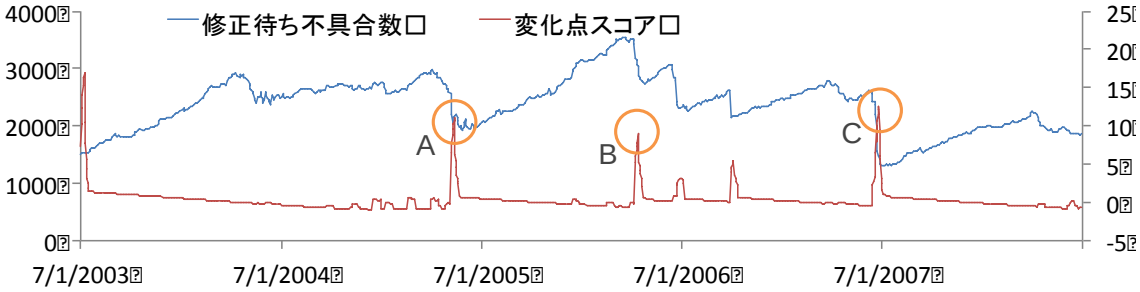
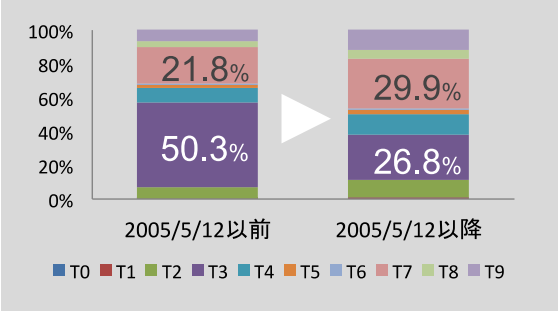
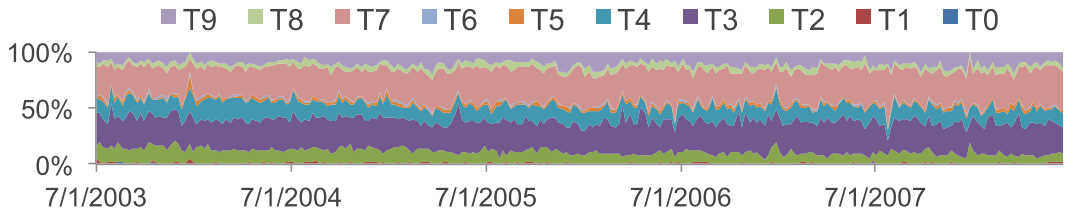


図 3-28 変化点検出の結果（修正待ち不具合数の時系列変化の例）



番号	ラベル	主な単語
T3	バグ・パッチ報告活動	bug created details attachment duplicate marked patch fixed verified fix head released verify
T7	修正活動	problem bug eclipse comment swt work time code don build fix gtk

図 3-29 LDA の結果（2005 年 5 月前後（図 3-13 の A の時期）のトピックの変化の例）



T0	バックグラウンド	line object core java run thread lang internal class worker wait
T1	ライブラリ	lib eclipse java usr gtk jar xp plugins vm opt system swt
T2	ビルド	file project eclipse cvs workspace build error files ant problem java log
T3	バグ・パッチ報告	bug created details attachment duplicate marked patch fixed verified fix head released
T4	UI	view editor text dialog menu window open key user show page problem
T5	イベント	swt shell display int public event void string table import eclipse item
T6	ワークベンチ	eclipse java org internal ui core swt run widgets main workbench jface
T7	修正活動	problem bug eclipse comment swt work time code don build fix gtk
T8	プラグイン	eclipse org ui http update plugins platform core plugin file jar xml
T9	API	api code method class change ui null case add content comment added

図 3-30 LDA の全結果

②インストーラ開発

実装したプラグインの導入を支援するためのインストーラを開発した．リポジトリマイニング・プラグインのインストーラも兼ねているので，図 3-15 と同じものである．なお，インストーラの配布にあたり，図 3-31 に示すようなマニュアルを作成した．マニュアルにはインストール方法と各プラグインの詳しい内容と利用方法を記述した．

また，本研究の成果物は以下のサイト（図 3-32～34）にて公開を予定している．

- 配布元
  - 和歌山大学システム工学部情報通信学科ソフトウェアエンジニアリンググループのウェブサーバ
  - http://oss.sys.wakayama-u.ac.jp/msr/
- 配布物
  - ソースコード一式
  - インストーラ（Redmine 用・Trac 用）
  - マニュアル一式
- ライセンス形式
  - GPL（GNU Public Lisence）
- 配布予定時期
  - 2015 年 2 月下旬

図 3-31 マニュアル外観

図 3-32 配布元ウェブサイト

図 3-33 インストーラ配布画面

図 3-34 マニュアル配布画面

さらに配布元ウェブサイト上にて，プラグインの機能を実際に利用して確認できるデモサイト（図 3-35）の公開を予定している．

図 3-35 デモサイト

3.5.3 課題とその対応

遅延相関検出プラグインの実証実験に関しては，学術論文としても成果が出ており，大きな課題はなかった．変化点検出プラグインに関しては，概ね意図していた通りの結果を得ることができたが，まだ十分な検証ができたとは言い切れない段階にあり，今後もメトリクスやプロジェクトデータを追加して検証を進めてきたいと考えている．

ただし，いずれにしても検証に用いたデータはオープンソースプロジェクトから収集したものであり，企業の現場でどれ程効果があるのかについては未知なままである．幸いにも，本研究終盤で実務者から本研究で開発したプラグインの公開を希望する声を多く頂いており，今後は企業の現場での適用を通じた実証実験の結果が得られるものと期待している．

なお，当初本研究では，マニュアルの整備や成果物の配布用ウェブサイトを準備することまでは計画していなかった．しかし，研究成果を広く社会に還元することが何よりも重要と考え，各種ドキュメントを準備し公開することとした．このような取り組みは継続して行っていくことが重要であると考えており，今後は EPM-X のセミナーを定期的に開催している PPMA(一般社団法人実践的プロジェクトマネジメント推進協会)様らと連携を図り，EPM-X およびプラグインの普及展開に貢献していきたいと考えている．

3.6 研究目標 6「実務者へのヒアリング調査の結果に基づいたプラグイン開発への反映（SG-B-5)」

3.6.1 当初の想定

(1) 研究内容

（内容）

2013 年度までに実装したプラグイン，さらには，2014 年度に実装予定のプラグインについて，実務者数名に対してヒアリング調査を実施した．特に，実装済み，あるいは，実装予定のプラグインが現場のニーズにマッチしているかどうか，実施の現場のニーズとしてはどの様な機能が求められているか，実装済み，および，実装予定のプラグインの改良点にはどのようなものが考えられるかなどについて把握することを目的とした．

（想定される課題と解決策）

国内大手ベンダの実務者数名からのヒアリングを実施する予定であった．しかし，社内決裁等の手続きが煩雑になることが予想され，現場からのニーズを迅速に本研究にフィードバックすることが困難になることが想定されることから，公式見解としてではなく実務者の個人的な見解としてヒアリング調査を取り纏め，プラグインを改良するための情報源とすることとした．

(2) 当初の到達目標と期待される効果

（当初の到達目標）

実務者へのヒアリングを通じて，開発したプラグインが現場のニーズにマッチしているかどうかを調査する．また，プラグインの改良点を調査する．

（期待される効果）

より有用性の高いプラグインにするために調査の結果をプラグイン開発にフィードバックすることができる

3.6.2 研究プロセスと成果

(1) 研究プロセス

以下の手順で研究を進めた．

- ヒアリング調査（SG-B-4-1）
- ヒアリング調査の取り纏め（SG-B-4-2）

(2) 具体的な研究成果（結果）

①ヒアリング調査

国内大手ベンダに勤務する 4 社計 5 名の実務者に対してヒアリングを実施した．プラグインを試用してもらった後，以下の点について意見を聴取した．

- プラグインの機能面について
  - *3 つのプラグインそれぞれについて，現場でどの様に活用できそうかどうかをお*

聞かせ下さい。また、現場で有効に活用するために必要な点などございましたらご自由にお聞かせ下さい。

- ✧ タスク担当者推薦プラグイン
- ✧ タスク再割当状況可視化プラグイン
- ✧ 遅延相関分析プラグイン

- インタフェースについて
  - *プラグインの選択画面や選択方法に関して、改良すべき点などございましたらご自由にお聞かせ下さい。*
- 今後の実装について
  - *数あるリポジトリマイニング手法の中でも特に現場で役立つと思われる手法（ニーズがある手法）があればお聞かせ下さい。また、すでに活用しているデータマイニング手法等がございましたら、差し支えのない範囲内で結構ですので、手法名と目的についてお聞かせ下さい。*

②ヒアリング調査の取り纏め

ヒアリングの結果，以下のような意見が得られた.

- プラグインの機能面について
  - タスク担当者推薦プラグイン
    - ✧ *現場ではアサイン状況がほぼ確定しており、担当すべき要員も掌握しているため、推薦の必要がないケースが多い。あるとすれば、管理要員や新規要員がプロジェクトの構成を把握する際に利用可能と思われる。*
    - ✧ *〇社では品質保証部門が別であり、総合試験は品証でやっています。システム担当なのか、ソフト担当なのか、ハード担当なのか割り振るという場面では使えるかもしれません。*
  - タスク再割当状況可視化プラグイン
    - ✧ *モジュールごとの有識者が把握できると良い。*
    - ✧ *緊急時に主担当者が対応できないときに使えそう。*
    - ✧ *プロパの人数が少ないため、再割当という概念とニーズがないかもしれません。*
  - 遅延相関分析プラグイン
    - ✧ *予兆をあらかじめつかむことができるので、ニーズはあると思われる。ただし、因果関係が説明できるものを絞り込む、結果を表すメトリクスを絞り込むなど、インタフェースの作り込みが必要。*
    - ✧ *現場でPJが失敗しそう、火を噴きそうかという判断で使えると思いました。ただ、メトリクスの設定が難しいかもしれません。*
    - ✧ *プロマネ毎、案件毎にグルーピングして結果を出せば、さらに使えるようになると思います。*
- インタフェースについて
  - *メトリクスごとの絞り込みがほしい。*
  - *IFは使いやすいと思います。*

- 今後の実装について
  - *モジュールの変更波及解析。*
  - *保守開発時に変更の影響がどの程度あるのかが事前にわかるとかなり助かる。予備調査のコストを大幅に減らせるだけではなく、品質もある程度担保できる。*
  - *後追いで対応している現場なので、状況が先読みできるような仕掛けは現場で大変喜ばれると思います。*

これからの結果から，既存プラグインに関しては，概ね前向きな意見が得られたが，必ずしもすべての実務者の現場で強いニーズがないことが分かった．インタフェースの使い勝手については，概ね良好であった．さらに，影響波及分析[Zimmerman05]に強いニーズが存在することが分かった．

これらの調査結果を踏まえ，メトリクスの絞り込み機能などインタフェースの改善に加え，プロアクティブマイニング技術として影響波及分析をプラグイン化することにした．

3.6.3 課題とその対応

前述したように，ヒアリングの結果，必ずしもすべてのプラグインの多くの現場で役立つものでないことが判明した一方で，ヒアリングによりプラグインの変更に間に合ったもの（影響波及分析）もあった．今後は，ソフトウェア技術者協会主催のソフトウェアシンポジウム（SS）や，一般財団法人日本科学技術連盟主催のソフトウェア品質シンポジウム（SQiP）をはじめとする実務者の多く集まるシンポジウム等に積極的に参加し，EPM-X プラグインの存在を知っていただくとともに，新たなニーズ獲得へ向けたヒアリング調査なども定期的に実施したいと考えている．

3.7 研究目標 7「プラグイン開発へのヒアリング調査結果の反映（SG-B-5)」

3.7.1 当初の想定

(1) 研究内容

（内容）

SG-B-4 のヒアリング調査の結果に基づいて，実装済みプラグインの改良および実装予定のプラグイン開発への反映を行った．

（想定される課題と解決策）

研究課題 A および研究課題 B では，学術界で有用性が認められたデータマイニング技術をプラグインとして実装するため，実務の観点から見ても有用性が高いものと，そうでないものとに選別される可能性があった．また，分野や業務内容の違いから，ヒアリング対象の実務者のニーズもそれぞれ異なる可能性がある．工数の制約からすべてのフィードバックを十分に反映できない状況が生まれた場合は，有用性が共通して高く，かつ，改良の余地のあるプラグインを優先して改良する．

(2) 当初の到達目標と期待される効果

（当初の到達目標）

ヒアリング調査よって得られた知見をプラグイン開発へフィードバックする．

（期待される効果）
実務者志向のより有用性の高いプラグインを提供することができる．

3.7.2 研究プロセスと成果

(1) 研究プロセス

以下の手順で研究を進めた

1. 実装済みプラグインの改良（SG-B-5-1）
2. 実装予定のプラグインの改良（SG-B-5-2）

(2) 具体的な研究成果（結果）

①実装済みプラグインの改良

ヒアリング調査の結果，実装済みのプラグインに関しては，主に，画面インタフェースの使い勝手について改良すべき点があることが分かった．プラグイン共通の画面インタフェースの機能として，メトリクスの絞り込み機能，テキスト検索機能，ソート機能を追加することで使い勝手を向上させた．

②実装予定のプラグインの改良

ヒアリング調査の結果，影響波及分析に強いニーズが存在することが分かった．厳密には，影響波及分析は本研究で行ったプロアクティブマイニング技術（異常検知技術）の分類ではないが，問題を前もって把握することを支援する技術ではあるため，当初実装予定であった，異常行動検出ではなく影響波及分析をプラグイン化することにした．

3.7.3 課題とその対応

ヒアリング調査によって得られた知見は可能な限りプラグイン開発へフィードバックすることができた．ただし，ヒアリング対象は大手ベンダ企業 4 社計 5 名の実務者を対象としたものであり，実務者から見た改善点はまだ数多く存在するものと思われる．今後は，プラグインのさらなる改良へ向けて実務者を対象とするヒアリング活動を積極的に展開していきたいと考えている．

4 考察

4.1 判明した効果や課題と今後の研究予定

本研究では,当初設定した目標である,リポジトリマイニング・プラグイン 5 件(Redmine 版・Trac 版合わせて計 10 件)の実装,プロアクティブマイニング・プラグイン 3 件(Redmine 版・Trac 版合わせて計 6 件）の実装，インストーラの実装を計画通り達成することができた．これら当初計画していたものに加えて，マニュアルおよび成果物配布サイトも作成することができ，概ね計画を上回る成果を上げることができたと言える．

新たに見出された課題としては，現場ニーズの把握の重要性が挙げられる．これまでも現場ニーズの把握に努めてきたが，明確な要望が得られることは少なかった．しかし，本研究では，特に研究終盤にかけて，実務者から改善要望や新たなニーズを数多くご指摘いただけたのは，本研究ではリポジトリマイニング技術およびプロアクティブマイニング技術の具体例を実際に試用できる状態にすることができたのが効果的であったものと思われる．今後は新たなニーズ獲得のためのツールとしても EPM-X プラグインを活用し，産業界での高度なデータマイニングツールの利用を薦めていきたいと考えている．

リポジトリマイニング技術をはじめとする高度なデータマイニング技術は，これまでも幾つかツール化されることがあった．しかし，単独の研究グループが開発したツールを公開するのみで，ツール毎に設定の異なるそれらいくつものツールを同じデータソースで試すようなことはできなかった．本研究では 8 種類のプラグインを同時に利用可能であり，実務者はカタログ的にリポジトリマイニング技術およびプロアクティブマイニング技術を試すことができる．異なる複数のツール（手法）を同じデータソースに適用するような試みは世界的に見ても例がなく，産業界との連携，特にデータ共有をより強化することにより，我が国のソフトウェア開発における生産性と品質の向上に大きく貢献できる可能性があると考えている．

本研究で開発したリポジトリマイニング技術は基本的には学術界で有用性が認められたものをベースにしており，学術的な新規性は少ない．一方，プロアクティブマイニング技術に関しては，ソフトウェア工学分野での適用事例がほどんど存在しない状況にあり，新たな知見が得られる可能性がある．実際,約 1 年半の研究期間を通じて,学術論文誌 3 編,査読付き国際会議論文 3 編，国内会議論文 15 編，表彰・受賞件数 4 件，招待講演 2 件と，決して少なくない数の学術的研究成果を得ることができた．

今後も引き続きリポジトリマイニング技術に関する研究を行っていくとともに，得られた学術的成果をプラグインとして実装し，実務者が広く利用可能な形で公開するよう努めたい．また，企業との共同研究や委託研究を加速させ，実証実験の結果として学術界にも技術適用の効果を広くフィードバックし，さらなる技術開発のヒントとして提供していきたいと考えている．

4.2 研究成果の産業界への展開について

本研究の研究成果は，これまでプロジェクトマネージャーの勘や経験に頼りがちであったソフトウェア開発をより高度化するものである．経験の浅いマネージャーの意思決定を強力に支援するだけではなく，経験豊富なマネージャーが自らの判断の正しさを客観的・

定量的な形でチームに伝えるためのツールとしても利用可能である．プロアクティブ技術を利用することでプロジェクトが危険な状態に入りかけていることをデータとして示すことが可能になり,プロジェクトや管理の見直しのための定量的な材料にすることができる．

ソフトウェア開発における大規模データの利用は，欧米では Google や Microsoft，IBM など大手企業においてすでに盛んに行われており，ICSE や FSE などの権威ある国際会議においても，大学との共同研究の成果としてすでに多数の事例報告がなされている．我が国においても各企業において勉強会を開催したり，ワーキングルグループを設置するなどの取り組みが始めれれているようである．EPM-X およびプラグインは，企業の現場に適用するために開発されたものであるが，これらのツールをプラットフォームとして，ソフトウェア開発における大規模データ利用について産業界をあげて議論することが，欧米での取り組みに遅れないようにするためには必要であるように思われる．

EPM-X プラグインは今後も引き続き改良および新規開発をおこないたいと考えている．しかし，プラグイン開発（プログラミング）自体は純粋な学術成果にはなり得ないため，本活動の勢いを落とさずに継続していくためには，人的リソースの確保が課題になる．本研究では，延べ 14 名の学生アルバイトを研究支援員として雇用した．活動レベルを落とさずにさらに発展させるためには，企業との共同研究や委託研究を加速させることが鍵となるのはいうまでもない．本研究に関与した学生は，最新技術に精通するだけではなく，企業ニーズと真摯に向き合うといった貴重な経験を積むことができたものと思われる．人材育成の観点からも本研究の取り組みは明らかに重要なものであり，産業界からの積極的な協力や支援もお願いしたい．

今後は，企業実務者が多く集まるシンポジウム等で EPM-X プラグインの存在を広く知っていただくための活動に取り組むとともに，企業と大学が連携を強化するための取り組みにも積極的に関与したいと考えている．我が国には若手研究者を中心いリポジトリマイニングを研究している大学教員が他国に引けを取らないほど存在しているが，おそらく企業には知られていない事実であると思われる．リポジトリマイニング技術は現在のところ個別性の高い技術であるため，1 企業と 1 大学研究グループでのマッチングという従来型の共同研究の形態では，お互いに効率が悪い．例えば，三ヶ月に一回程度，国内のリポジトリマイニング研究者を東京などに集め，企業の実務者を招待して研究会を開催し，企業が抱える相談に答えるといった取り組みを考えている．

参考文献

[Acharya 11] Acharya, M. and Robinson, B.: Practical Change Impact Analysis Based on Static Program Slicing for Industrial Software Systems, in Proceedings of the 33rd International Conference on Software Engineering (ICSE '11), pp. 746–755 (2011)

[Anvik 06] Anvik, J., Hiew, L., and Murphy, G. C.: Who Should Fix This Bug?, in Proceedings of the 28th International Conference on Software Engineering (ICSE'06), pp. 361–370 (2006)

[Baum 70] Baum, L. E., Petrie, T., Soules, G., and Weiss, N.: A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains, The Annals of Mathematical Statistics, Vol. 41, No. 1, pp. 164–171 (1970)

[Bhattacharya 10] Bhattacharya, P. and Neamtiu, I.: Fine-grained Incremental Learning and Multifeature Tossing Graphs to Improve Bug Triaging, in Proceedings of the 2010 IEEE International Conference on Software Maintenance (ICSM '10), pp. 1–10 (2010)

[Bhattacharya 12] Bhattacharya, P., Iliofotou, M., Neamtiu, I., and Faloutsos, M.: Graph-based Analysis and Prediction for Software Evolution, in Proceedings of the 34th International Conference on Software Engineering (ICSE '12), pp. 419–429 (2012)

[Ducasse 99] Ducasse, S., Rieger, M., and Demeyer, S.: A Language Independent Approach for Detecting Duplicated Code, in Proceedings of the IEEE International Conference on Software Maintenance(ICSM '99), pp. 109– (1999)

[EPM-X] 定量的プロジェクト管理ツール(EPM-X), http://sec.ipa.go.jp/tool/ipf/

[Hassan 10] Hassan, A. E. and Xie, T.: Software Intelligence: The Future of Mining Software Engineering Data, in Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research (FoSER '10), pp. 161–166 (2010)

[Hewett 09] Hewett, R. and Kijsanayothin, P.: On Modeling Software Defect Repair Time, Empirical Software Engineering, Vol. 14, No. 2, pp. 165–186 (2009)

[Jeong 09] Jeong, G., Kim, S., and Zimmermann, T.: Improving Bug Triage with Bug

Tossing Graphs, in Proceedings of the the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering (ESEC/FSE '09), pp. 111–120 (2009)

[Kim 08] Kim, S., Whitehead, E., and Zhang, Y.: Classifying Software Changes: Clean or Buggy?, IEEE Transactions on Software Engineering, Vol. 34, No. 2, pp. 181–196 (2008)

[Krichevsky 81] Krichevsky, R. and Trofimov, V.: The performance of universal encoding, IEEE Transactions on Information Theory, Vol. 27, No. 2, pp. 199– 207 (1981)

[Lin 14] Lin, Y., Xing, Z., Xue, Y., Liu, Y., Peng, X., Sun, J., and Zhao, W.: Detecting Differences Across Multiple Instances of Code Clones, in Proceedings of the 36th International Conference on Software Engineering (ICSE '14), pp. 164–174 (2014)

[Mondal 14] Mondal, M., Roy, C. K., and Schneider, K. A.: A Fine-Grained Analysis on the Evolutionary Coupling of Cloned Code, in 30th IEEE International Conference on Software Maintenance and Evolution (ICSME '14), pp. 51–60 (2014)

[Nagappan 06] Nagappan, N., Ball, T., and Zeller, A.: Mining Metrics to Predict Component Failures, in Proceedings of the 28th International Conference on Software Engineering (ICSE '06), pp. 452–461 (2006)

[Neal 99] Neal, R. M. and Hinton, G. E.: Learning in Graphical Models, chapter A view of the EM algorithm that Justifies incremental, sparse and other variants, pp. 355–368, Adaptive Computation and Machine Learning, MIT Press (1999)

[Ohira 04] Ohira, M., Yokomori, R., Sakai, M., Matsumoto, ichi K., Inoue, K., and Torii, K.: Empirical Project Monitor: a Tool for Mining Multiple Project Data, in International Workshop on Mining Software Repositories (MSR '04), pp. 42–46 (2004), Edinburgh, Scotland, UK

[Park 11] Park, J., Lee, M., Jinhan Kim, , Hwang, S., and Kim, S.: COSTRIAGE: A Cost-Aware Triage Algorithm for Bug Reporting Systems, in Proceedings of the Twenty-Fifth Conference on Artificial Intelligence (AAAI'11) (2011)

[Rissanen 84] Rissanen, J.: Universal coding, information, prediction, and

estimation, IEEE Transactions on Information Theory, Vol. 30, No. 4, pp. 629–636 (1984)

[Robles 06] Robles, G., Gonzalez-Barahona, J. M., Michlmayr, M., and Amor, J. J.: Mining Large Software Compilations over Time: Another Perspective of Software Evolution, in Proceedings of the 2006 International Workshop on Mining Software Repositories (MSR '06), pp. 3–9 (2006)

[S ´liwerski05]S ´liwerski, J., Zimmermann, T., and Zeller, A.: When Do Changes Induce Fixes?, in Proceedings of the 2nd International Workshop on Mining Software Repositories (MSR '05), pp. 1–5 (2005)

[Takeuchi 06] Takeuchi, ichi J. and Yamanishi, K.: A Unifying Framework for Detecting Outliers and Change Points from Time Series, IEEE Transactions on Knowledge and Data Engineering, Vol. 18, No. 4, pp. 482–492 (2006)

[Viterbi 06] Viterbi, A.: Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm, IEEE Transactions on Information Theory, Vol. 13, No. 2, pp. 260–269 (2006)

[Weiss 07] Weiss, C., Premraj, R., Zimmermann, T., and Zeller, A.: How Long Will It Take to Fix This Bug?, in Proceedings of the Fourth International Workshop on Mining Software Repositories (MSR '07), pp. 1– (2007)

[Xuan 12] Xuan, J., Jiang, H., Ren, Z., and Zou, W.: Developer prioritization in bug repositories, in Proceedings of the 34th International Conference on Software Engineering (ICSE'12), pp. 25–35 (2012)

[Yamanishi 02] Yamanishi, K. and Takeuchi, J.: A Unifying Framework for Detecting Outliers and Change Points from Non-stationary Time Series Data, in Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD ' 02), pp. 676–681 (2002)

[Yamanishi 04] Yamanishi, K., Takeuchi, ichi J., Williams, G., and Milne, P.: On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms, Data Mining and Knowledge Discovery, Vol. 8, No. 3, pp. 275–300 (2004)

[Yamanishi 07] Yamanishi, K. Y. and Maruyama, Y.: Dynamic Model Selection With its Applications to Novelty Detection, IEEE Transactions on

InformationTheory,Vol.53, No.6,pp.2180–2189(2007)

[Yamatani 14] Yamatani, Y. and Ohira, M.: An Exploratory Analysis for Studying Software Evolution: Time-Delayed Correlation Analysis, in Proceedings of 6th International Workshop on Empirical Software Engineering in Practice (IWESEP 2014), pp. 13–18 (2014)

[Zhang 13] Zhang, H., Gong, L., and Versteeg, S.: Predicting Bug-fixing Time: An Empirical Study of Commercial Software Projects, in Proceedings of the 35th International Conference on Software Engineering (ICSE '13), pp. 1042–1051 (2013)

[Zimmermann 05] Zimmermann, T., Weigerber, P., Diehl, S., and Zeller, A.: Mining Version Histories to Guide Software Changes, IEEE Transactions on Software Engineering, Vol. 31, No. 6, pp. 429–445 (2005)

[久木田 15] 久木田 雄亮, 柏 祐太郎, 大平 雅雄:変化 点検出とトピック分析を用いたリポジトリマイニ ング手法の提案, ウィンターワークショップ 2015・ イン・宜野湾 論文集, pp. 23–24 (2015)

[山西 09] 山西 健司:データマイニングによる異常 検知, 共立出版, 東京 (2009)

[山谷 15] 山谷 陽亮, 大平 雅雄, Phannachitta, P., 伊 原 彰紀:OSS システムとコミュニティの共進化の 理解を目的としたデータマイニング手法, 情報処 理学会論文誌, Vol. 56, No. 1, pp. 59–71 (2015)

[赤池 94] 赤池 弘次, 北川 源四郎(編):時系列解析 の実際 I, 統計科学選書, 朝倉書店 (1994)

[赤池 95] 赤池 弘次, 北川 源四郎(編):時系列解析 の実際 I, 統計科学選書, 朝倉書店 (1995)

[大平 05] 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, Vol. J88-D-I, No. 2, pp. 228–239 (2005)

[竹内 08] 竹内 裕之, 児玉 直樹:生活習慣と健康 状態に関する時系列データ解析手法の開発, in Proceedings of the 3th Forum on Data Engineering and Information Management (DEIM '11), pp. D7–5 (2008)

[武田 00] 武田 圭史, 磯崎 宏:ネットワーク侵入検知 —不正侵入の検出と対策, ソフトバンククリエイ ティブ (2000)

[門田 13] 門田 暁人, 伊原 彰紀, 松本 健一:「ソフト ウェア工学の実証的アプローチ」シリーズ第 5 回 ソフトウェアリポジトリマイニング, コンピュー タソフトウェア, Vol. 30, No. 2, pp. 52–65 (2013)