Eメールアーカイブのクラスタリングによる開発コンテキストの可視化

大蔵 君治† 川口 真司†† 飯田 元†††



情報化の進む現代社会において，ソフトウェアはインフラの重要な構成要素となっている．しかしながら，ソフトウェア開発プロジェクトでは今もなおコスト超過や納期遅れといった“失敗”が相次いでいる．失敗の原因の1つに「プロジェクト分析の不足」が挙げられる．プロジェクトの事後分析は人的・時間的なコストが掛かるため，分析に十分な時間を確保することが難しいのが現状である．そこで，本研究ではソフトウェア開発においてよく用いられるメールに着目し，そのやり取りを時系列上に表示する手法を提案する．これにより，従来のメトリクスグラフで把握していたプロジェクトの「状態」に加え，「現場の開発背景」を同時に俯瞰することが可能となる．本稿では手法の詳細と共に，実際の開発プロジェクトに本手法を適用した3つのケーススタディを紹介する．

A Method for Visualizing Contexts in Software Development using Clustering Email Archives

Kimiharu Ohkura†, Shinji Kawaguchi†, and Hajimu Iida†

Software is a prominent factor of infrastructure in a modern information society. However failures in software development project such as cost or deadline over still remain. One of causes of the failures is considered a lack of postmortem analysis for project improvement. Because a postmortem analysis has high costs of time and human resource, it is difficult to perform the analysis continuously under the current circumstances. In this research, we focus on email conversations often used in software development, and we propose a method for visualizing them in the project. Our method helps to grasp "contexts" in development further to existing visualizing approaches such as metrics chart that can reveal only "state". In this paper, we describe details of our method and case studies of applying the method to actual projects.

1 はじめに

情報技術の発展によって，ソフトウェアは社会インフラの中枢をなす重要な構成要素となっている．それに伴い，ソフトウェア開発の需要は継続的に増え続けており，数々の開発プロジェクトが日々遂行されている．しかし，コスト超過や納期遅れに代表される「失敗プロジェクト」は，今もなお多く存在する [NIKKEI2008][データ白書 2009].

プロジェクトの失敗が減らない理由の１つとして，失敗の原因を分析することが難しいという点が挙げられる．プロジェクトの事後分析は，開発記録やプロジェクトメンバの記憶を頼りに行うこととなるが，膨大な記録から事後分析に有用な情報を取り出すには時間的なコストがかかる．また，プロジェクトメンバが開発中に起こった出来事をすべて思い出せるという保証は無い．このような理由から，開発プロジェクトの事後分析を継続的に行うことが困難となっている．

この問題に対し，開発過程で得られた定量データを用いて，プロジェクトの進捗やソフトウェアの品質を測る研究が数多くなされてきた [CHIDAMBER1994][KAMIYA2002][QUENTIN2006].

しかしながら，定量データから得られる情報はあくまでプロジェクトがそのときどのような「状態」であったかを数値で表したものであり，なぜプロジェクトがそのような状態になったのかという「原因」を調べるためには，プロジェクトメンバへのヒアリングが必要不可欠となる．ヒアリングは事後分析において有効な手段であるが，実施にかかるコストが分析者，開発者共に大きい．また，分散開発にお

† 奈良先端科学技術大学院大学 情報科学研究科 研究員, Graduate School of Information Science, NAIST(Nara Institute of Science and Technology), Reseacher
†† 奈良先端科学技術大学院大学 情報科学研究科 助教, Graduate School of Information Science, NAIST(Nara Institute of Science and Technology), Assistant Professor
††† 奈良先端科学技術大学院大学 情報科学研究科 教授,Graduate School of Information Science, NAIST(Nara Institute of Science and Technology), Professor

いては，モジュールごとの開発担当者があいまいであることも多く，ヒアリングの対象者を特定出来ないといった問題がある [ANVIK2006].

本研究ではこの問題に対し，開発に用いられたEメール（以下，メール）と，開発リポジトリから得られる定量データを対象とした分析手法を提案する．メールはソフトウェア開発プロジェクトにおいてよく用いられるコミュニケーションツールであり，数値データからは読み取ることの出来ない現場の情報が蓄積されていると考えられる．例えば，ミーティング時間の超過が開発の遅れの原因となっていた場合，開発が滞っているという「状態」はソースコード行数グラフ等，数値データの時系列上における変化から確認することが出来る．しかし，なぜ開発が滞っているかという「原因」をグラフから得ることは出来ない．このようなとき，同時期に交わされたメールのやり取りを確認すればミーティングの開催といった種々の情報が得られる可能性がある．

本研究ではこのような「数値データに直接現れない現場の情報」をコンテキストと総称し，中でも，蓄積されたメール群に存在するひとまとまりの話題（開発背景）として体現されるコンテキストを「メールに基づくコンテキスト情報」として可視化の対象とする．なお，簡略のため以下では，上記をメールコンテキストと呼ぶ．すべての有益なコンテキストがメールのやり取り中に記録されていているとは限らないが，開発者へのインタビューやドキュメントの精査を実地に行うことに比べると，はるかに低コストでの抽出が可能である．また，本論文の実験結果はメールコンテキストが十分有益な情報を含むことを示唆している．

人手でメールコンテキストを確認する作業は，メールの件数が多い場合の分析コストが高く，また，長期にわたって潜在的に存在する話題の把握が困難である．本研究では，以下に示すようなアプローチによってメールコンテキストを自動的に可視化し，分析を支援する．

・ベクトル空間法とクラスタリングを用いたメールコンテキストの自動抽出
・メールコンテキストの時系列上へのプロット

メールコンテキストを時系列に置くことで，ソースコード行数グラフをはじめとする他の時系列グラフとの重畳表示が可能となる．これにより，開発プロジェクトの「状態」と「原因」を相互に分析しやすくなる．例えば，数値データから開発が滞っているという「状態」が判明したときに，その時点に存在するメールコンテキストを見ることで，「原因」の推測がより容易になる．

本研究は開発プロジェクトの事後分析を支援するためのものであるが，当該プロジェクトのメンバ，及び外部分析者の両方を利用者に想定している．以下に，提案する手法を用いた事後分析のユースケースを簡単に示す．

(1)　外部分析者による利用

当該プロジェクトに関する予備知識に乏しい外部の分析者（研究者等）が，プロジェクトの要因分析を行ったり，状況把握を行ったりする場合には，分析の糸口を得るという観点から，本手法は非常に有用であろう．

(2)　当該プロジェクトのメンバによる利用

プロジェクトの当事者が失敗の原因を調査したり，過去のプロジェクトで発生した事例を調べたりするために本手法を利用出来る．当事者であればメール以外の開発データへのアクセスも容易であるため，多種の開発記録と本手法の出力結果を併せて，プロジェクトの精査が可能であると考えられる．

以降，第2節ではメール分析を扱った関連研究と既存のツールを使った事後分析方法について述べる．第3節では手法の詳細について説明し，第4節では実際に手法を適用した3つのケーススタディについて述べる．最後に，第5節にてまとめと今後の展望について述べる．

2　関連研究

本節ではメールの分析に関する関連研究と，本手法と既存のプロジェクト情報管理ツールとの位置付けについて述べる．

2.1　メールの分析に関する研究

メールを対象とした分析手法は，これまでにも数多く行われてきた．ソフトウェア開発分野におけるメールの分析手法は，大別すると，送信数等のメタ情報を用いたソーシャルネットワーク分析（以下，SNA※1）と，自然言語解析を用いた手法に分けられる．

Birdらは，SNAアプローチを用いて大規模なオープンソースソフトウェア（以下，OSS※2）プロジェクトに対して調査を行った [BIRD2006]．調査は開発用メーリングリストに

脚注

※1　SNA：Social Network Analysis，ソーシャルネットワーク分析
※2　OSS：Open Souce Software，オープンソースソフトウェア

対して行われ，各開発者のメール送信数や被送信数，媒介中心性等のメトリクスを用いて分析を行った．調査の結果，各開発者のコミット数とメール送信数には高い相関があることや，メーリングリストへの投稿は一部のアクティブな開発者らによって大半が占められていること等が明らかとなった．このように，SNA 手法は主に開発者に着目した分析を行うために用いられる．そのため，SNA によって各開発者の特性が明らかになったとしても，どの開発者がどのモジュールを担当していたかといった，ソフトウェア成果物との関連を調べるためには手動での調査が必要となる．Sarma らはこの点に着目し，SNA によって描かれた開発者同士のネットワークと，モジュール依存度を基に描かれたファイルネットワークを関連付けて提示するツール“Tesseract”を開発した [SARMA2009]．例えば，ある開発者に着目し，その開発者と関連の強い他の開発者，及びモジュールを同時に提示するといったことが可能となる．しかし，ファイルネットワークと開発者ネットワークの関連付けはすべて手動で行われているため，分析のための事前準備にかかるコストは高い．

また，時系列情報を持つ文書の到着頻度に着目し，そのトピックの活発性を測る研究も行われている [KLEINBERG 2002][崔 2004]．これらの研究は文書（記事）の到着時間間隔を手掛かりに分析を行っているため，本文の影響を受けずにトピック活性度の計測が可能である．しかし，これらの手法はリアルタイムに文書が到着することを想定している上，各種パラメータの設定は経験的に決められており，調整が極めて困難であると思われる．

Rigby らは，メール本文を解析し，その言語的手掛かりによってプロジェクトの動向を分析した [RIGBY2007]．分析には LIWC※3[LIWC] と呼ばれるツールを用いている．LIWC は辞書登録された文章中のキーワードをカウントし，その文章を 70 種類の心理的特徴に分類するツールである．例えば“may be”といったキーワードは心理的特徴「あやふや（tentativeness）」に，“important”等のキーワードは「確信（certainty）」にそれぞれ関連付けられている．Rigby らは分析の中で，優秀な開発者と一般の開発者で感情変化にどのような差があるか等を調べ，優秀な開発者ほど外向性が低いことや，感情性（神経質さ）は優秀な開発者と一般の開発者で差が無いこと等を示した．しかし，このような言語的手掛かりを用いた手法は，くだけた会話やコミュニティ依存の専門用語等を使用するインフォーマルな会話には適用しにくいため，分析対象は一部の OSS プロジェクト等に限定される．

また，メール本文の特徴語に着目した討議構造の検出に関する手法も提案されている [仁野 2008][中山 2008]．これらの手法は，メール本文の構造がある程度形式的であり，時間や場所に関連付けられた単語が本文から抽出可能であること，また，「ところで」，「なぜかというと」といった発話上の言語的手掛かりが本文中に存在することを前提としている．そのため，開発プロジェクト内でのみ用いられる専門的な用語やスラング，あるいはインフォーマルな会話には不向きである．

本研究はメールの本文を対象としてクラスタリングを行う．提案手法では意味解析，構文解析，討議構造の解析等は行わず，単語の統計情報のみを用いる．そのため，分析用の辞書を作成する必要は無く，異なる言語圏でも同じ手法を適用することが出来る．

2.2 プロジェクト管理ツールによる事後分析

今日ではソフトウェア開発に様々なツールが用いられており，その代表的なものに Bugzilla（バグジラ）[BUGZILLA] や GNATS（グナッツ）[GNATS] 等の障害管理ツールや，Microsoft Project 等の進捗管理ツールが挙げられる．このようなツールを導入しているプロジェクトにおいては，メールを確認せずともミーティングのスケジュールやバグの詳細情報を正確に把握することが可能である．しかしながら，事後分析を行う者が常にそれらのツールを利用出来るとは限らず，データの互換性にも問題が生じる可能性がある．

本研究で入力として用いるメールアーカイブは，一般的な MUA※4 で読み書き可能な標準的なフォーマットであるため互換性の問題は発生しにくく，単一のファイルであるため可搬性にも優れている．更に，提案手法では「ミーティングの日程」といった単純な情報だけでなく，「なぜその日程に決定したか」といったコンテキストを抽出出来る可能性がある．一般に，このような管理ツールには形式的な情報（結果）が記録されるのみであり，スケジューリングのミスといった事象（原因）はログに残されていない．提案手法は，そのようなインフォーマルなやり取りにおいてのみ記録される情報を抽出出来る可能性がある．ただし，本研究は従来の分析手法に置き換わるものではなく，従来の分析結果に重ね合わせることによって，事後分析における情報量を増加させることを目的とする．

3 分析手法

本手法は，単一のメールアーカイブを対象とする．メールコンテキストの抽出手順は次の通りである．

① 各メールのベクトル化

すべてのメールの本文に対して形態素解析を行い，単語を抽出する．そして，抽出された単語の頻度情報を要素として持つベクトルを，すべてのメールに対して生成する．

② クラスタリングによる話題の自動分類

ベクトル化された各メールは，ベクトル演算によって類似度（距離）の計測が可能である．このとき，「類似度が高い」ということは「似通った単語が出現するメール同士」であることを意味する．すべてのベクトル間で類似度を計算し，クラスタリングアルゴリズムに基づいて各ベクトルを類似度の高いもの同士でカテゴライズしていく．

③ クラスタの時系列上へのプロット

クラスタリングによって得られた各クラスタは，似通った内容を持つメールの集合である．本研究ではこれを１つの話題であると見なし，メールコンテキストと呼ぶ．クラスタの要素であるメールには，送信日時が記録されているため，これを用いて時系列上にプロットする．

時系列上にプロットされた各クラスタを，他の時系列グラフと重畳表示することで，プロジェクトの状態とメールコンテキストを同時に俯瞰することが出来る．概念図を図１に示す．以降では，各手順の詳細と，実装したツールについて説明する．

3.1 各メールのベクトル化

各メールは，ベクトル空間モデル [SALTON1975] に基づいてベクトル表現へと変換される．ベクトル空間モデルでは，メール中に出現する単語（索引語と呼ぶ）の頻度がベクトルの要素となる．また，生成された各メールのベクトル表現のことをここでは文書ベクトルと呼ぶ．図２に，文書ベクトルの概念を示す．

このとき，単純に単語の出現頻度をそのまま文書ベクトルの要素とした場合，どのメールにも普遍的に表れる単語（例えば「私」等）が類似度に大きく影響してしまう．この問題を回避するために，TF-IDF※6 [SALTON1987] による重み付けを行う．TF-IDF とは情報検索等で用いられている単語のスコアリング手法である．単語の出現頻度である TF に，その単語が検索対象内においていかに希少かを示す値である IDF を乗ずることで，普遍的な単語の影響を少なくすることが出来る．TF-IDF の式を以下に示す．

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i + 1}\right)$$

$w_{i,j}$：文書 j における単語 i の重み（スコア）
$tf_{i,j}$：文書 j における単語 i の出現頻度
df_i：単語 i を含む文書 j の数
N：その文書空間における文書の総数

本研究における文書空間はメールアーカイブ全体であり，文書は各メールの本文に相当する．

提案手法では，あらかじめすべてのメールの単語に対して，TF-IDF によるスコアリングを行っている．その際，各メールでスコアの高かった単語をキーワードとしてリストアップしておき，それらのキーワードを連結したものを索引語として用いている．これは，索引語に普遍的な単語が入らないようにするためである．なお，次元数が増加すると計算量が膨大になるため，各メールからリストアップするキーワードの数は２とした．また，予備実験の結果から，抽出する単語は名詞と未知語のみとした．形態素解析システムには Sen[SEN] を用いている．

3.2 クラスタリングによる話題の自動分類

文書ベクトル間の類似度計算後は，クラスタリングアルゴリズムに従って各メールを話題別に分類していく．クラスタリングアルゴリズムには Ward 法※7，k-means 法※8，最長距離法等様々なものがあるが，どのアルゴリズムが最

図1　コンテキストの重畳表示（概念図）

索引語：	コンパイル	点検	例外	停電	エラー
文書１	0	1	0	2	0
文書２	2	0	3	0	1
・・・					

索引語の出現頻度

d_1=(0, 1, 0, 2, 0)
d_2=(2, 0, 3, 0, 1)

図2　文書ベクトルの概念

脚注

※3 **LIWC**：Linguistic Inquiry and Word Count
※4 **MUA**：Mail User Agent
※5 **LOC**：Lines of Code
※6 **TF**：Term Frequency，**IDF**：Inverse Document Frequency
※7 **Ward 法**：Ward's Method，ウォード法
※8 **k-means 法**：日本語では「K 平均法」と呼ばれる非階層型クラスタリングアルゴリズムの１つ。

適であるかは生成された文書ベクトルによって異なる．本研究では，使用するクラスタリングアルゴリズムやその閾値についてはとくに定めず，何度かの試行によって最適なアルゴリズムを分析者が判断し，選択することとする．類似度に用いる尺度についても同様に，ユークリッド距離，コサイン距離，マハラノビス距離等から最も精度が良いと判断したものを選択する．

3.3 クラスタの時系列上へのプロット

クラスタリングによって得られたクラスタは，類似度の高い文書ベクトルの集合である．時系列上へのプロットは，各文書ベクトルに対応するメールの Date ヘッダから日付情報を取得して行う．メールを個別にプロットするのではなくクラスタ単位でプロットすることで，メールコンテキストを俯瞰しやすくなる他，長期間にわたって散在する話題や，定期的に行われるイベント等の把握を容易にする．

3.4 ツールの実装

本手法は次の 3 つのツールによって実現している．

・データコンバータ（Java，コンソール）
・クラスタリングツール（C#，GUI）
・プロジェクトリプレイヤ（C#，GUI）

データコンバータとクラスタリングツールは，本研究のために独自に開発したものである（形態素解析システムを除く）．入力から可視化までの流れを図 3 に示す．データコンバータは mbox 形式のメールアーカイブをベクトルデータに変換する他，クラスタリングツールが出力するログデータから樹形図を画像ファイルとして出力したり，クラスタリング結果を html 出力する機能を備えている．図 4, 図 5 に，それぞれの例を示す．クラスタリングツールは，ベクトルデータを読み込み，クラスタリングを実行する GUI プログラムである．GUI では，使用するクラスタリングアルゴリズムや距離尺度，閾値等を設定出来る．設定画面を図 6 に示す．プロジェクトリプレイヤ（以下，リプレイヤ）は，本研究チームで開発を行っているプロジェクト分析支援ツールである [OHKURA2006]．リプレイヤは，自動収集された定量データを基に，開発プロジェクトをビデオのように再生する機能を持つ．リプレイヤが提示出来る情報の 1

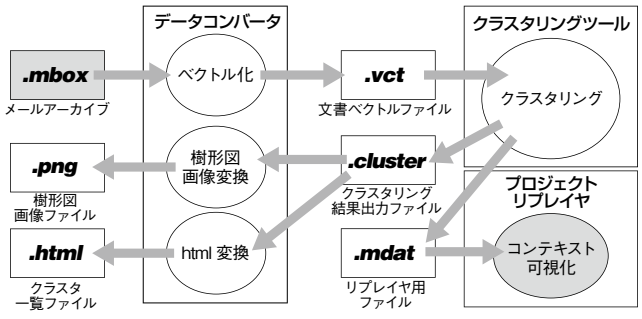

図3　コンテキスト可視化までの流れ

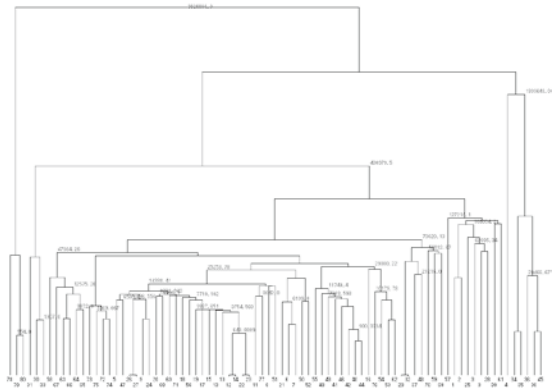

図4　出力される樹形図（画像ファイル）の例

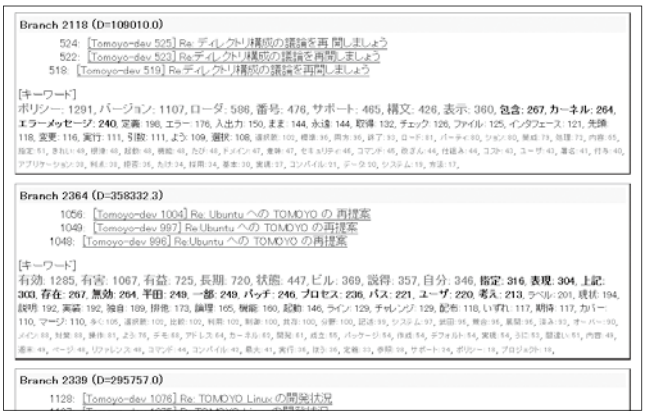

図5　クラスター覧のhtml出力

図6　クラスタリングツールの設定画面

つとして，ソースコードメトリクスのラインチャートがある．我々は，このラインチャートにクラスタを重畳表示出来るよう，リプレイヤに改良を加えた．クラスタを重畳表示したラインチャートを図7に示す．水平に延びている線が1つのメールコンテキスト（クラスタ）を表す．実装上の問題で，キーワードをチャート上に表示することは出来なかったが，線上をマウスオーバすることで，そのクラスタを構成するメール中の，スコアの高い単語を表示することが出来る．クラスタ数が多くて視認性が低い場合は，右に表示されたクラスタ一覧から，見たいクラスタだけを選択して表示させることが出来る．黒い縦線は現在の日付を表す．現在の日付をまたがって存在するクラスタ（縦線と交差している横線）に含まれる個々のメールは，別ウィンドウの「メールビュー」にて確認することが出来る（図8）．

4 ケーススタディ

我々は本手法の有効性を検証するために，3つの開発プロジェクトデータに対して適用実験を行い，以下に挙げる3つの観点から有効性を検証した．

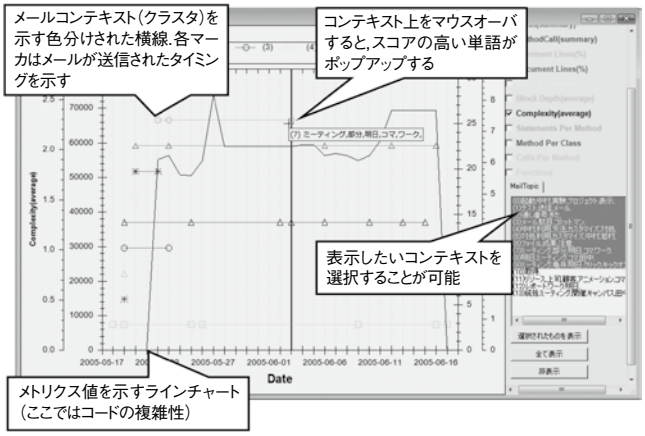

図7　メールコンテキストの重畳表示（リプレイヤ）

図8　メールビュー（リプレイヤ）

・提案手法の実現可能性
・提案手法の優位性
・得られたクラスタの正確性

以降4.1 ～ 4.3節にそれぞれの観点からの評価を示す．

4.1 小規模開発への適用

我々はまず，本手法の実現可能性（実際にメールコンテキストに相当する話題が抽出出来るかどうか）を確認するために，小規模なソフトウェア開発プロジェクトに対して適用を行った．クラスタリングアルゴリズムには，試行の結果，最もバランスの良い樹形図を描いたWard法を使用した．クラスタリングを終了させるための閾値にはクラスタ間類似度の平均値を使用し，類似度はWard法の定義によりユークリッド距離を用いた．対象プロジェクトの詳細は表1の通りである．なお，開発要員はすべて奈良先端科学技術大学院大学の学生で，開発は主に学内で行われた．

当該プロジェクトの開発用メーリングリストに対して手法を適用した結果，リプレイヤのラインチャート上にクラスタを重畳表示させることが出来た（図7）．我々は，1人の被験者（研究者，当該プロジェクトとは無関係）を用意し，このプロジェクトのメールコンテキストを分析してもらった．その結果，数値データには現れない以下のような開発状況を，30分程度の調査で把握することが出来た．

・各開発者の役割
・ミーティングスケジュールの調整
・コード修正のアナウンス
・リソースのアップロード報告

当該プロジェクトのメールコンテキストを表2に示す．表2のメールコンテキストは，当該プロジェクトの開発メ

表1　対象プロジェクト（小規模開発）

プロジェクト	ゲーム開発（大学オープンキャンパス用）
開発要員	7人
開発期間	27日
メールサイズ	272KB，81件

表2　プロジェクトメンバが手動で抽出したメールコンテキスト一覧

番号	メールコンテキストの内容
1	EPMの導入について
2	メーリングリストの設定議論
3	メーリングリストへのテスト送信
4	EPMの不具合について
5	開発ミーティングの内容調整
6	CVSコミットメールに関する話題
7	リソース（画像）作成に関する報告
8	開発するフレームワークに関する打ち合わせ
9	統括ミーティングの日程調整

ンバの 1 人によって手動で抽出されたものである．インタビューの結果，「ミーティングスケジュールの調整」，「コード修正のアナウンス」，「リソースのアップロード報告」は，順に表 2 における 5, 6, 7 番にあたるとメンバは判断した．「各開発者の役割」は表 2 のメールコンテキストのいずれにも該当しないが，被験者は複数のメールコンテキストを見ることでそれぞれの開発者の役割を判断した．また，その役割の判断が正しいものであるかをメンバによって確認してもらった．その結果，被験者が判断した各開発者の役割は正しいものであることが確認出来たが，インタビューを行ったメンバも一部のメンバについてはその役割の詳細を忘れていた．回顧を促すためにクラスタリング結果の html を閲覧してもらったところ，短時間で各メンバの詳細な役割を思い出すことが出来た．

また，被験者は「全くメールコンテキストが存在しない期間」にも着目した．同時期のコード行数グラフにも動きが見られないことから，被験者は何らかの理由でこの期間は開発が停滞していたと予想した．被験者は，停滞期間の前週に「試験」等の単語を含むメールコンテキストが存在することから，「学科試験のために開発が中断していた」という推測を行った．この推測は正しいものであったが，事実確認には当該プロジェクトメンバへのヒアリングが必要であった．ただし，メールの内容から各開発者の役割が大まかに把握出来ていたため，ヒアリングにかかる時間を抑えることが出来た．

4.2 企業開発（保守工程）への適用

我々は 4.1 節の実験において本手法，及び本手法を実装した一連のツールを用いて実際の開発プロジェクトを分析可能であることを確認した．本実験では単に MUA でメールスレッドを巡覧した場合と比較した際の提案手法の優位性を検証するために，ソフトウェア開発企業から提供を受けた大規模なデータに対して提案手法の適用を行った．対象となるデータが大規模であるため，クラスタリングアルゴリズムには計算量の少ない k-means 法を用いた．プロジェクトの詳細を表 3 に示す．なお，守秘義務により開発用メーリングリスト以外の各種メトリクスデータは，別ツールを用いて表示したもののスクリーンショットとなる．そのため，重畳表示ではなくグラフとメールコンテキスト表示（リプレイヤ）を並列に並べて分析を行った．

表3　対象プロジェクト（企業）

プロジェクト	企業開発プロジェクト（保守工程）
開発要員	10 人以上
データ収集期間	およそ 500 日
メールサイズ	65.1MB，38,365 件

実験では，提供されたコード行数グラフに見られる特徴的な遷移に着目した（図 9）．特徴的な遷移を見せているのは，10 月 28 日と 2 月 28 日の 2 箇所で，それぞれ急上昇，及び急降下している．我々は 2 人の被験者（研究者，当該プロジェクトとは無関係）に，この 2 箇所について各被験者 1 箇所ずつ原因を探ってもらった．1 時間程度の調査の結果，以下のように原因を正しく判定することが出来た．

① 10 月 28 日に行数が急上昇した原因

開発データの中央集約化の際に行った CVS 環境の新構築が原因であった．この情報は 10 月 28 日付近のメールコンテキストから確認出来た．また，同じメールコンテキストに含まれるメールから，中央集約化の実施に至った理由が，旧サーバのハードディスク障害であることが判明した．「CVS 環境の新構築」と「旧サーバのハードディスク障害」は時間的に離れた別々のメールスレッドに書かれた情報であるため，単に 10 月 28 日付近のメールスレッドを確認するだけではハードディスク障害について知ることは出来ない．また，本節で分析対象としているメールアーカイブは保守工程という性質上，ユーザからの質問や各種依頼のメールが多数転送されてきており，開発者同士のやり取りとユーザの質問に対する回答のメールが混在している．ユーザからのメールはすべて独立したスレッドであるため，開発者同士のやり取りを確認したい場合はノイズとなる．提案手法を用いた場合，同じような内容の問い合わせで 1 つのクラスタを形成するため，メールスレッドを巡覧するよりも効率良くコンテキストを把握出来ると考えられる．

② 2 月 28 日に行数が急降下した原因

開発言語の変更に伴い，不要となったファイルを一斉削除したことが原因であった．この情報は 2 月 28 日付近のメールから確認出来たが，「なぜ開発言語が変更になったのか」という情報は掴めなかった．

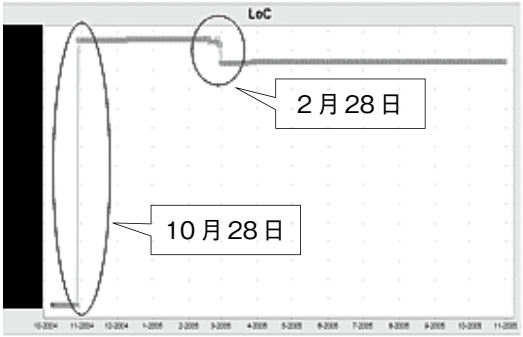

図9　コード行数グラフに見られる特徴的な推移

これら2つの事象に対する「原因」は明確にメールに書かれていたため，推測ではなく事実とした．ただし，これらの事象以外にも原因がある可能性があるため，2箇所の特徴的な推移に対するすべての原因が分析出来たとは言えない．

4.3 オープンソースソフトウェアプロジェクトへの適用

最後に，我々はクラスタリング結果が人にとってどれだけ直感的であるかを検証するための実験を行った．対象は，表4に示すOSSプロジェクトである．実験手順は次の通りである．

① 被験者に開発用メーリングリストを手動で分類してもらう
② 提案手法を用いて同メーリングリストを自動分類する
③ 手動分類と自動分類の結果と比較し，提案手法がどれだけ手動分類を再現出来ているかを調べる

表4　対象プロジェクト(OSS)

プロジェクト	TOMOYO プロジェクト（Linux アクセス制御）
開発要員	20 人
データ収集期間	2005/10/27 ～ 2009/11/16
メールサイズ	2.77MB，1,256 件

一般に，メールを返信した際には返信ヘッダが付与され，メールスレッド構成する．スレッドは既に1つの話題の集合であると考えられるので，我々はこれをメールコンテキストの最小単位とした．被験者（大学院生）には図10のような記入例を示した入力用データシートを渡し，スレッドを更に抽象化した中分類と大分類にメールコンテキストを分割してもらった．結果の一部を抜粋して表5に掲載する．被験者は，当該メーリングリストのスレッドを計24個の中分類と4個の大分類に分類した．本実験では提案手法によって中分類を再現出来るかどうかを検証する．なお，再現出来ているかどうかの判定は，クラスタ一覧を出力したhtml（図5）を被験者に閲覧してもらい，インタビュー形式で行った．クラスタリングアルゴリズムにはWard法を使用した．

【記入例】　※ルート(最初の発言者)のMLナンバーを記入して下さい

大分類	中分類	該当スレッド	キーワード（スペース区切り）
開発	○○機能実装 △△のバグ修正 etc..	392, 882, 501 22, 57, 212, 333 （以下略）	プレビュー jpg 画像 フォーマット ヒープ メモリ 不足 （以下略）
イベント	オフ会の開催 etc..		
アナウンス	リリース情報 etc..		
その他の相談	メンバーの追加 etc..		

図10　入力用データシートの記入例

表5　被験者による手動分類の結果(一部抜粋)

大分類	中分類	該当スレッド	キーワード（スペース区切り）	評価	HTMLとの相違点
開発	ファイルのパーミッション・アクセスに関する話題	751, 756, 776, 830, 831	ファイル パーミッション アクセス 許可	2	クラスタ2456に多く含まれている
	ファイルの置換・移動に関する話題	113, 587, 612	置換 衝突 議論 移動	3	全て大きなクラスタ2409に含まれる
	ツールのポリシーに関する話題	100，200，370，529，540，755，874, 931	ポリシー 環境変数 プログラム パラメータ	2	
	開発のメインラインを定義するための議論	369，438，448，490，522，551，856, 937, 988, 1095, 1103, 1028	ドメイン ディレクトリ構成 名前の由来 css tomoyo 仕様変更	3	369のスレッドの内容が多岐にわたるため，いくつかのクラスタに分散している
	コードのクリーンアップ	553, 662, 682, 687, 836	コード クリーンアップ	3	大きなクラスタ2453に多く含まれている
	メモリの使用量に関する話題	797, 944	メモリ 使用量 ポリシー 制限	1	クラスタ2421
	csstoolsに関する話題	73, 586, 900, 1089	csstools ポリシー ロード	4	csstoolsに関する話題ではあるが，その詳細はそれぞれ異なっている
	次期バージョンに向けての仕様の議論	1，8，84，96，101，561，587，908，1050, 1179	仕様 提案	4	どのような機能を実装するかによって，分類されるクラスタは異なっている
	カーネルのコンパイル	54，88，116，132，143，209，642，710, 757, 780, 813, 840, 1029	カーネル コンパイル テスト リポジトリ 協力者 リリース	2	
アナウンス	リリース情報	14，81，115，137，162，206，250，361，376，533，571，656，675，676，824，846，868，882，884，885，924，927，943，1030，1041，1081，1082，1085，1092，1177，1200, 1202, 1203	パッチ コンパイル リリース 変更点 サポート規則 アップデート	3	リリースは各機能の改善と一緒のクラスタに含まれていた
	バグ報告	8，368，383，794，818，852，925，1077, 1104, 1109	スペルミス バグ 報告 修正 特有	4	バグ報告はそれぞれの機能と結びついている
	ドキュメント修正に関する話題	350, 509, 826, 896, 915	GUI インストールドキュメント 文字コード	4	ドキュメントの修正も個々の機能の話題と結びついている

表6　インタビュー結果

質問	手動で分類した中分類に合致するクラスタが存在したかどうかを，以下の評価基準に従って各項目ごとに評価してください．
評価1	ほぼ合致するするクラスタが存在する
評価2	部分的に合致するクラスタが存在する
評価3	どちらともいえない
評価4	合致するクラスタは存在しない

回答	件数
評価1	2
評価2	6
評価3	8
評価4	8

閾値は数回の試行によって得られた中分類に適した値を手動で設定した．被験者の評価結果を表 6 に示す．ここで,「部分的に合致するクラスタ（表 6 の評価 2)」とは，手動で分類したスレッドのうちおおむね半分以上が含まれているクラスタを指す．

評価 1，評価 2 を正解とすると，提案手法は手動分類結果の 33.3%（8/24）を抽出出来ているものの，出力されたクラスタの総数は 56 となっており，提案手法が手動分類と十分に適合しているとは言い難い．しかし，4.2 節のケーススタディが示すように，提案手法は膨大な数のメールアーカイブに対して分析に有用なレベルの分類結果を返しており，本評価結果は手法の有用性を否定するものではない．とりわけ，時間的コストの観点から評価すると，被験者は 1,256 件のメールを手動で分類する作業におよそ 6 時間を費やしているのに対し，本手法は小規模なプロジェクトであれば数秒から数十秒，大規模（10,000 件～）なメールアーカイブであっても数分から数十分の間に分類を終わらせることが可能（PC スペック CPU：AMD Athlon 64 X2 4800+, RAM：2GB の場合）であるため，時間的コストにおけるアドバンテージは極めて高く，分析の糸口として用いるには十分に有用であると言える．今後，精度の改善を行うことで更に有用性の向上が期待出来る．

4.4 妥当性

本節では 3 つのケーススタディについて述べ，それぞれのケースにおいて提案手法の有用性を確認するための実験を行ってきた．しかし, いずれの実験も被験者の数が少なく，結果を一般化することは出来ない．

また，メールの分類にかかるコストについては手動分類との比較を行ったが，分析にかかるコストについては比較を行っていないため，検証が必要である．具体的には，単純にメールクライアント等を用いて分析した場合と，提案手法を用いた場合とで時間的コスト，対象プロジェクトへの理解度等にどのような差が出るかを検証する必要がある．

5 まとめと今後の展望

本論文ではソフトウェア開発プロジェクトのコンテキストを，メールアーカイブのクラスタリングと，クラスタの時系列プロットによって可視化する手法を提案した．また，提案手法を実現するためのツールを開発し，3 つのプロジェクトに対して手法を適用し，その有効性について検証した．本論文では第 1 節にて，定量データのみを用いた事後分析には以下のような問題があると述べた．

・観測事象に対応するコンテキストの特定が困難
・ヒアリング対象者の特定が困難
・開発者の記憶があいまい

以上のそれぞれについて，ケーススタディの中で解決出来たこと，解決出来なかったことについて次にまとめる．

① 観測事象に対応するコンテキストの特定が困難

4.2 節で述べた企業開発データへの適用実験において, ソースコード行数が特徴的な遷移を見せた 2 箇所についてそれぞれ原因を明らかにすることが出来た．しかし，明らかになった 2 箇所の原因のうち，1 つは本手法を用いずとも該当日付周辺のメールスレッドを確認することで明らかに出来た．よって，この結果をもって「単にメールを巡覧したときと比べて，本手法を用いた分析の方が常に優れている」と言うことは出来ない．しかし，メールの件数が膨大になればなるほどスレッドの巡覧にかかるコストは増大する．本手法はクラスタリングの際に閾値を調整することによって，スレッドよりも抽象的な範囲で話題を抽出することが可能であり，数年にわたるような長期プロジェクトであっても短期間でメールコンテキストの把握が可能である．

② ヒアリング対象者の特定が困難

4.1 節で述べた小規模プロジェクトの分析において，被験者は本手法を用いて各プロジェクトメンバの役割を把握出来ていた．分析対象のメーリングリストには，当該プロジェクトメンバ以外から送信されたメールも多数含まれていたが，ミーティング調整やコード修正アナウンスのメールコンテキストから，誰がプロジェクトメンバであるかを判断することが出来た．このように開発者の役割が判断出来ていれば, ヒアリング対象者を絞ることが容易となる．ただし，4.1 節は小規模なプロジェクトのみの検証であるため，一般性を述べるためには適用プロジェクトのサンプルを更に増やす必要がある．

③ 開発者の記憶があいまい

4.1 節にて行ったプロジェクトメンバへのインタビューは 2009 年に行われたが，当該プロジェクトは 2005 年に実施されており，プロジェクトメンバは開発の詳細を部分的に忘れていた．しかし，インタビューの際にクラスタリング結果の html を閲覧することで詳細を思い出すことが出来た．ただし，これについても当該プロジェクトが小規模であるため，提案手法を用いずともメールスレッドの巡覧のみでプロジェクトを回顧出来た可能性が高いため，この点をもって手法を一般化することは出来ない．しかし，メールスレッドから更に抽象化されたメールコンテキスト群は話題の俯瞰に適しており，MUA を用いたメールスレッドの巡覧よりも短時間で回顧が可能であると考えられる．

以上のように，本手法の一般性は今後の追加実験によって更なる検証を行う必要があるものの，現時点においても一定の有効性は確認出来たと言える．また，本論文で述べた3つのケーススタディは異なるスケール（小規模～大規模）のプロジェクトを対象としており，本手法が一定のスケーラビリティを保していることを示している．今後は更に大規模な実プロジェクトへ適用し，本手法がその対象規模にかかわらず適用可能であることを示す必要があると考えられる．

本手法は開発形態や組織形態にかかわらず，様々な実開発プロジェクトに適用可能であると考えられるが，プロジェクトによってメールの利用目的は様々であり，どのような開発においてどのようにメールが利用されているかの調査が必要である．また，手法の精度向上，及び信頼性向上のために適用サンプルを増やしていく必要がある．しかし，多くの開発組織は機密保持を理由に開発データを外部に提供しないという現状があるため，今後は産学の連携をより一層強めていくことが求められる．

また，索引語抽出のアルゴリズムや重み付けの調整についても，より人間にとって直感的なコンテキストの分類が出来るよう手法を改善していこうと考えている．更に，分析にかかる時間的コストをより小さくするために，閾値別の分析結果をシームレスに切り替えて表示するためのインターフェースを用意していきたいと考えている．

謝辞

本研究の一部は，文部科学省「次世代IT基盤構築のための研究開発」の委託に基づいて行われた．また，本研究は日本学術振興会特別研究員奨励費の助成を受けたものである．

参考文献

[ANVIK2006] John Anvik, Lyndon Hiew, and Gail C. Murphy : Who should fix this bug? In ICSE '06 : Proceedings of the 28th international conference on Software engineering, pp.361-370, New York, NY, USA, 2006, ACM

[BIRD2006] Christian Bird, Alex Gourley, Prem Devanbu, Michael Gertz, and Anand Swaminathan : Mining email social networks. In MSR '06 : Proceedings of the 2006 international workshop on Mining software repositories, pp.137-143, New York, NY, USA, 2006, ACM

[BUGZILLA] http://www.bugzilla.org/

[CHIDAMBER1994] S. R. Chidamber and C. F. Kemerer : A metrics suite for object oriented design, IEEE Trans. Softw. Eng., Vol.20, No.6, pp.476-493, 1994

[GNATS] http://www.gnu.org/software/gnats/

[KAMIYA2002] Toshihiro Kamiya, Shinji Kusumoto, and Katsuro Inoue : Ccfinder : a multilinguistic token-based code clone detection system for large scale source code, IEEE Trans. Softw. Eng., Vol.28, No.7, pp.654-670, 2002

[KLEINBERG2002] Jon Kleinberg : Bursty and hierarchical structure in streams, pp. 91-101, ACM Press, 2002

[LIWC] http://www.liwc.net/

[NIKKEI2008] 中村 建助，矢口 竜太郎：2008年情報化実態調査プロジェクトの成功率は31.1%，日経コンピュータ 2008年12月1日号，pp.38-53，December 2008

[OHKURA2006] Kimiharu Ohkura, Keita Goto, Noriko Hanakawa, Shinji Kawaguchi, and Hajimu Iida : Project replayer with email analysis - revealing contexts in software development, In Proceedings of the 13th Asia Pacific Software Engineering Conference, pp.453-460, December 2006

[QUENTIN2006] Quentin W. Fleming and Joel M. Koppelman : Earned Value Project Management, Project Management Institute, 2006

[RIGBY2007] Peter C. Rigby and Ahmed E. Hassan : What can oss mailing lists tell us? a preliminary psychometric text analysis of the apache developer mailing list, In MSR '07 : Proceedings of the Fourth International Workshop on Mining Software Repositories, p.23, Washington, DC, USA, 2007, IEEE Computer Society

[SALTON1975] G. Salton, A. Wong, and C. S. Yang : A vector space model for automatic indexing, Commun, ACM, Vol.18, No.11, pp.613-620, 1975

[SALTON1987] Gerard Salton and Chris Buckley : Term weighting approaches in automatic text retrieval, Technical report, Ithaca, NY, USA, 1987

[SARMA2009] Anita Sarma, Larry Maccherone, Patrick Wagstrom, and James Herbsleb : Tesseract : Interactive visual exploration of socio-technical relationships in software development, In ICSE '09 : Proceedings of the 2009 IEEE 31st International Conference on Software Engineering, pp.23-33, Washington, DC, USA, 2009, IEEE Computer Society

[SEN] Sen Project : https://sen.dev.java.net/

[崔 2004] 崔 春花，北川 博之：到着頻度と関連性を考慮した時系列文書の連続的トピック分析（時系列とコンテンツ）（夏のデータベースワークショップdbws2004），情報処理学会研究報告，データベース・システム研究会報告，Vol.2004，No.72，pp.315-322，20040714

[データ白書 2009] IPA/SEC：ソフトウェア開発データ白書2009 ～2327プロジェクト　定量データ分析で分かる開発の最新動向～，日経BP社，2009

[中山 2008] 中山 祐貴，宮寺 庸造，横山 節雄，中村 勝一：電子メール中の議論過程抽出手法の提案（教育・学習評価／一般），電子情報通信学会技術研究報告，ET，教育工学，Vol.108，No.354，pp.35-40，20081206

[仁野 2008] 仁野 裕一，野田 潤，中尾 敏康：特徴語の共起性を利用した携帯電話メールの関係性検出手法（ライフログ活用技術とその課題，オフィス情報システム，デジタルドキュメント，一般），電子情報通信学会技術研究報告，OIS，オフィスインフォメーションシステム，Vol.108，No.156，pp.71-76，20080717