長期的大規模プロジェクトのドキュメントを事例としたトレーサビリティリンク回復の実践と課題

小形 真平[1,a)]　鷲崎 弘宜[2,b)]　石川 公一[1,c)]

概要：トレーサビリティの重要性は古くより認識されているにも関わらず，研究において利害関係者や産業界のニーズを十分に把握できていないため，産学連携により産業界の実際の文脈におけるトレーサビリティの成功や失敗を明らかとするケーススタディの必要性が指摘されている．そこで，本稿では長年にわたる大規模プロジェクトで作成されてきたドキュメントを事例として，産学連携によりトレーサビリティリンク回復を実践した結果について報告する．そして，実践内容を踏まえて，長年にわたり記述が未整理かつ未統一な状況において成果物間のトレーサビリティリンクを回復する現実的なツールおよびプロセスや，大規模プロジェクトで作成されてきたドキュメント群へ当該ツールおよびプロセスを適用することにより得られた知見，効果，課題について考察し，まとめる．

キーワード：トレーサビリティリンク回復，ケーススタディ，支援ツール

Practice and Challenges of Traceability Link Recovery for Documents in Long-Term Large Projects

1. はじめに

長期的に開発・保守される大規模システムでは，途上での技術革新や人員交代も見込まれ，それでもなお持続的な開発・保守活動に耐えうる品質を持つシステム仕様が不可欠である．そして，ドキュメントやソースコード等の成果物間のトレーサビリティは，その保守品質を底上げする重要な性質である，

トレーサビリティの重要性は古くより認識されているにも関わらず，研究において利害関係者や産業界のニーズを十分に把握できていないため，産学連携により産業界の実際の文脈におけるトレーサビリティの成功や失敗を明らかとするケーススタディの必要性が指摘されている [7]．たとえば，産業界では，トレーサビリティリンク維持・回復は高度な専門知識を要するため，当該知識の乏しい企業が豊富な企業に依頼することがある．そのようなケースでは，トレーサビリティリンクに関する技術的な問題を扱うだけでは不十分であり，機密情報の持ち出し禁止や知財等の契約，窓口担当者間のコミュニケーションの限界などの制約（以降，プロジェクト制約）を受けながら問題解決を目指さなければならない．このことがトレーサビリティリンク回復に影響していることは，目的や作業，環境に特化したツール群の利用が好まれている実態があることが報告されている [4], [7] ことからも推測される．

そこで，本稿では長年にわたる大規模プロジェクトで作成されてきたドキュメントを事例として，産学連携によりトレーサビリティリンク回復を実践した結果について報告する．図 1 に示すように適用事例は複数の組織 (組織 A,B,C) を跨ってやりとりされるドキュメント (各種サンプルドキュメントや全体データ) を対象にトレーサビリティリンク回復を行うものであり，そのプロセスの中で回復に必要なツール (ルールベース回復ツールや辞書ツール) 等を調整していったものである．段階的なサンプルドキュメントの提供や，組織間でのデータのやりとり，それらルールに応じた回復アプローチの選定やツールの作り込みといったプロセスは，純粋な技術的課題以上にプロジェクト制約に強く影響を受けるものである．この中で現実的なトレー

[1] 信州大学　〒 380–8553 長野県長野市若里 4–17–1
[2] 早稲田大学　〒 169–8555 東京都新宿区大久保 3–4–1
[a)] ogata@cs.shinshu-u.ac.jp
[b)] washizaki@waseda.jp
[c)] 18w2010f@shinshu-u.ac.jp

サビリティリンク回復ツールやプロセスを探究し，異なるアプローチを比較することは意義があるものと考える．

本論文の貢献は以下を示すことにある．

- 長年にわたり記述が未整理かつ未統一な状況において成果物間のトレーサビリティリンクを回復する現実的なツールおよびプロセス
- 長年にわたる大規模プロジェクトで作成されたドキュメント群へ上記トレーサビリティリンク回復ツールおよびプロセスを適用することにより得られた知見，効果，課題

2. 適用事例の課題と制約

本研究での適用事例は，長年にわたり開発され，進化してきた大規模システムのドキュメントである．その総容量が約 60GB に及ぶ中でトレーサビリティリンク回復を実践する．回復で具体的に達成すべきことは，分類 A（便宜上，仕様と呼ぶ）と分類 B（便宜上，技術と呼ぶ）それぞれの用語間に関係の有無を正確に与えることであった．

本事例でのトレーサビリティリンクの回復目的は，システム開発・保守の支援である．たとえば今後の派生開発等において仕様または技術の組み合わせが更改されたときにその影響波及範囲を容易に把握等できるように，仕様と技術のドキュメント間のトレーサビリティを保証するためのものである．回復の実践にあたっては，本プロジェクトでは以下のように幾つもの課題・制約が存在した．

- **ドキュメントのファイル形式や体裁，書式が多岐にわたる．**ファイル書式は一般的な Office スイートで作られたものもあれば，データベースで整理されたものもあり，体裁・書式については自然言語記述文章・図表表現が混在して形式性が低い．この結果，ファイル形式や体裁，書式に依らず統一的に情報を扱おうとすると文字列形式になるが，たとえば表形式の内容を無理矢理に文字列化するとその並びから意味を見出すことが困難な状態となる．表の表現例としては，用語間の関係を○×で表すものもあれば，文章により関係の有無や説明を記載するものもあった．

  この状況は，長年にわたる開発・保守の中でその体制の変化などを理由として生じており，大規模プロジェクトならば，なおさら生じやすい．このことから，自然言語記述を用い，かつ仕様記法・体制等が経年により変化するならば，これらの情報の表現方法を統一して維持することは，たとえ新規開発で一から記述する場合であっても困難であると予想される．このような困難さは適用事例外でも一般に生じるものと考えられるため，本制約下での回復の実践により得た知見は他の広範なプロジェクトにも通じるものと期待される．
- **ドキュメント作成者の専門知識に依存して同じ概念が全く異なる表現となることがある．**加えて，要素技術名の省略の仕方が異なることや，誤りなども存在し，当事者でなければ一貫した読み方が困難と推測されるものもあった．このことから，協働関係にある距離の遠い開発要員間で意思疎通が困難であることが指摘されていた．つまり，人が見ても対応が不明になりやすい語群が存在した．

  また，同じ概念が全く異なる表現となることは，表記上の類似性から同じ概念を表す用語群を同定することが困難なことを示唆する．適用事例は，数多くの部門が協働開発しているシステムであり，そのような開発体制を敷くプロジェクトは IoT 時代に入り，一般的な制約となるであろう．それ故に，本制約下での現実的な回復支援ツールならびにプロセスを探究することは意義がある．
- **仕様と技術のそれぞれの用語が識別されていない．**多数の部門により協働開発されるシステムでは様々な用語が存在するため，仕様または技術に該当する用語を全て把握している者はいなかった．そのため，用語の識別もタスクに含める必要があった．
- **ドメインエキスパートと協働することが困難である．**前項に関連して，地理的・業務的・要員数的に用語の識別などにドメインエキスパートから十分な支援を受けることが困難であった．そのため，適用事例の当事者でない組織が用語を識別する必要があり，用語の識別は多大な知識や作業時間がかかった．しかし，用語を識別してから回復を実践することは納期的に現実的でなかった．このことから，用語を漏らさないよう全てのドキュメントから全ての語を収集する必要があり，用語識別とリンク回復の作業は並行して行う必要があった．
- **データは段階的に提供される．**プロジェクト制約として，全てのデータは最初から提供されず，組織 A が判断してデータを段階的に提供していた．これにより，全ての単語の把握することが最初からはできず，また多量のデータを得ることもできなかった．各組織で望ましいとするデータ提供方法の考え方は異なっていたが，これは契約内容に基づく制約や，データ利活用法等の想定の差異によるものであった．そのような状況で，大規模データにも対応できるスケーラブルな回復方法が求められていた．
- **分散作業において情報・データ交換が制限される．**プロジェクトやデータの管理タスクや，トレーサビリティリンク回復手法の考案タスクを地理的に分散した要員が分担し，協働する状況にあった．しかし，プロジェクト制約としてオンラインでの情報・データ交換は制限されていた．そのため，多量のデータを用いて試行錯誤的に考案手法を改良する機会は限定されていた．

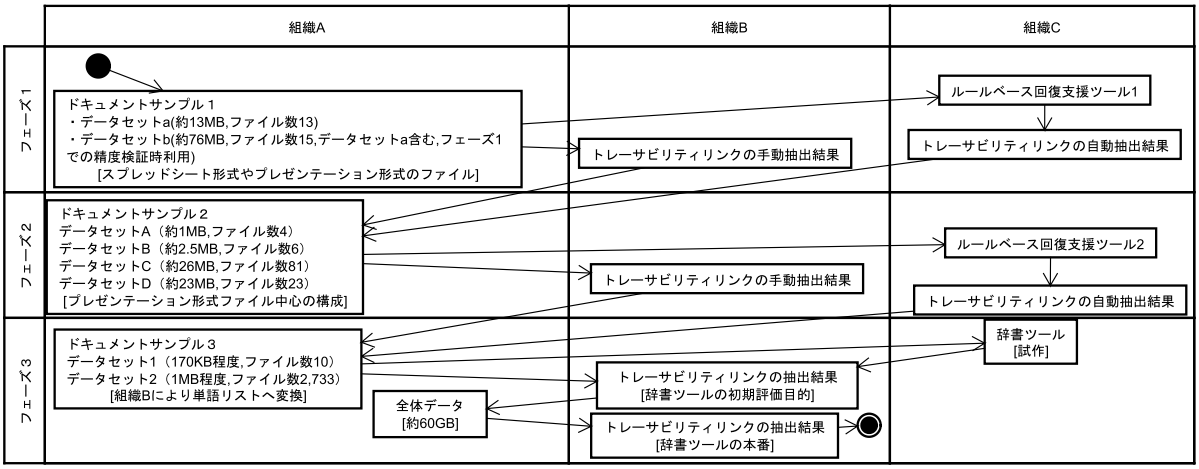

図 1 トレーサビリティリンク回復の実践におけるデータのやりとり

3. トレーサビリティリンク回復

トレーサビリティリンク回復とは，ソフトウェア開発プロセス改善を目的とした，成果物内の関連するドキュメント間のリンクを導出する作業である [17]．回復の結果は，ソースやシンクとなるドキュメント間の関係の有無を示すトレーサビリティマトリクスとして整理される [6], [13]．そのようなトレーサビリティマトリクスを基本として，付加的にドキュメント間の関連性を示すこと [13] や，関連の種類や信頼度を示すこと [6] がなされる．本研究でも，ソースとシンクとなる用語間に関係の有無を与えるものである．

トレーサビリティリンク回復には，Retrospective（後ろ向き）な方法と Prospective（前向き）な方法がある [3]．Retrospective（後ろ向き）な方法では，ソフトウェア成果物から遡ってトレーサビリティリンクの推論を行う．たとえば，テキストから候補リンクを抽出する方法や，コード解析や実行時監視により自動的にリンクを得る方法，一部の正解付きリンクを学習して正解が未知のリンクを予測する方法などが考えられる．一方，Prospective（前向き）な方法では，モデル変換の痕跡など，解析によってトレーサビリティリンクとなるものをしかるべき工程で与える．たとえば，成果物 ID をソースコードに埋め込む方法や，成果間の整合性を自動で維持する方法などが考えられる．本研究では，既に作成済みでトレーサビリティが十分に確保できていないドキュメントを対象とするため Retrospective な方法を採る．

Retrospective な方法としては，IR(Information Retrieval) 手法（たとえば [5]）やルールベース手法（たとえば [19]），機械学習アプローチ（たとえば [16]），深層学習アプローチ（たとえば [10]）がある．しかし，適用事例ではデータ提供に関する制約から，機械学習・深層学習アプローチは性能評価が困難であると判断され，除外された．そのため，本研究ではルールベース手法や IR 手法を優先的に検討・採用した．

一般にトレーサビリティリンク回復の対象となるドキュメントの種類は多岐にわたり，たとえばテストケースとユースケース [13], [17]，クラスとそのマニュアルページ [1]，要求とその実現クラス [1]，ユースケースとクラスコード [17] 等がある．本研究では，非形式な自然言語記述を中心としたドキュメントを対象とする．対象ドキュメントは表により構造化された記述がしばしば見受けられたが，ソースコード等の明確な構文規則に従う文章を前提としたアプローチは適さなかった．

ドキュメント中の表は，2 つの用語の関係の有無を表すものや，データ一覧を表すものなど異なる目的・様式で複数作成されており，解析には工夫を要する．他方で，文献 [14] で指摘されているように，日本語はアルファベット，漢字，ひらがな，カタカナ，西洋文字（ラテン文字，アラビア数字，各種記号）が複数混在する複雑な書記体系を持つことで知られている．本研究でも，このような特徴を全てもつ日本語の文や単語が主な解析対象となる．

4. トレーサビリティリンク回復の実践

明確なトレーサビリティリンクが失われた大規模システムのドキュメントに対し，仕様と技術間のトレーサビリティリンク回復を目的とする．本稿では，プロジェクト制約下で回復を実践して，従来の技術の適用可能性や，制約下で現実的なプロセスやツールを探究するケース・スタディの結果を報告する．なお，ドキュメントは office 系ファイルを含むが，その解析のために Apache の POI[2] を用いた Java プログラムを作成した．

4.1 プロセス

図 2 は実践したプロセスを示す．関連する組織は主には 3 つあり，ドメインエキスパート（組織 A），プロジェクト・データ管理（組織 B），トレーサビリティリンク回復方

法の考案（組織 C）という関係にあった．

当初に計画されたスケジュールでは，およそ 1 年という期間において，単一アプローチによる回復が想定されており，サンプルデータの追加によるフェーズ分けが想定されていた．そして，プロジェクト開始時点では組織 B と C にはドメイン知識が不足しており，正解となるトレーサビリティリンクを把握できていなかった．このことから，仕様と技術の間の関係に対する正確な読み解きが重視され，ルールベースのアプローチが採用された．しかし，図 2 に示す実際のスケジュールでは，ルールベースのアプローチではドキュメントの多様性に納期内で十分に対応できないことがフェーズ 2 の結果から推測された．その結果，フェーズ 3 で IR 的アプローチに切り替え，辞書ツールを実現・試用した．

4.2 フェーズ 1：ルールベース回復支援ツール 1 の試作

フェーズ 1 では少量なドキュメントサンプルに対して，リンク回復が試行された．ドキュメントサンプルは主にスプレッドシート形式で構成されており，ある程度規則性が見える表形式であった．そこで，表を認識し，ルールベースでリンクを導出するツールを実現して適用した．本ツールでは，スプレッドシート形式のドキュメントを与えると，仕様と技術のトレーサビリティマトリクスが得られる．

ドキュメントの詳細なところでは，スプレッドシートの行列で規則性は見えるが，具体的なファイル間で行列のズレや，構文規則の若干のブレが見受けられた．そのため，表の列名でも，たとえば，必ず A 行 3 列にあるとは言えず，B 行 3 列や A 行 4 列にあるファイルも存在した．また，1 ファイルに異なる種類の表が複数存在するケースもあった．表の形としては，2 つの項目間の関係を○×などで表すマトリクスや，データの一覧を表すもの，それらを複合したものがあった．さらに，トレーサビリティリンクの探索では，複数の表を跨ぐ追跡が必須であった．たとえば，2 つの表（項目 P 群-項目 Q 群のマトリクスと項目 R 群のデータ一覧があり，Q の一部の項目と R の一部の項目とが特定の解釈により対応づくもの）があったとき，マトリクス上で $(p_{\mathrm{i}}, q_{\mathrm{j}})$ に関係があった場合，一覧で q_{j} に対応する r_{k} を探して，最終的には $(p_{\mathrm{i}}, r_{\mathrm{k}})$ を得る必要があった．

ここでは，提供された少数のドキュメント間の共通な特徴を調査し，その特徴から表を認識・解析するツールを試作した．特徴とは例えば，列ヘッダ（列名部分）の 2 列目上部に同一の文字列が必ず与えられていることがある．また，データのまとまりを区切る意図であろう空行と，最終行の次行とを区別するための罫線等の装飾も特徴に挙げられる．

行と列のヘッダには異なる複数の方法で階層が与えられているものもあり，その解析も必要となった．たとえば，列ヘッダでは 3 行分を使い，1 行目を大分類，2 行目を中分類，3 行目を小分類とすることがあり，他方では行ヘッダは 1 列ではあるが，大分類の行の下から小分類の行が幾つか続く，といういことが繰り返される．これらは表種ごとに定まっているよう見受けられたため，解析時にファイル名等から表種を特定して処理を切り替えた．また，マトリクスでの値は○×から文字列まで様々であるため，データ全体を調査して，各値の有無の分類ルールを手動で分析してツールに組み込んだ．

4.3 フェーズ 2：ルールベース回復支援ツール 2 の試作

フェーズ 2 では，フェーズ 1 とは異なる少量なドキュメントサンプルに対して，リンク回復が試行された．本フェーズでのドキュメントは，主にプレゼンテーション形式で構成されており，前フェーズのルールの多くは再利用困難であった．そこで，プレゼンテーション形式中に見られる規則性を捉えて，新たなルールでリンクを導出するツールを試作した．本ツールでは，プレゼンテーション形式のファイルを与えると，仕様と技術のマトリクスが得られる．ファイル内の文章の形態素解析には MeCab[11] を用いた．

プレゼンテーション形式ファイルでは自然言語記述が中心的であった．その中では，たとえば “仕様⇒改良後仕様 ” という形式で関係が表されていたり，ファイル名に技術名が含まれていたりなどする．ファイル間で記述者が異なると，たとえば “仕様⇒改良後仕様 ” 部分の書式は類似していても，完全に一致する訳ではないものもあった．

本フェーズでは，スプレッドシート用のルールの多くが再利用できないにしても，プレゼンテーション用のルールも積み上げていくことで，その後の多くのドキュメントが処理可能であることが期待されたため，ルールベースのアプローチを変更しなかった．

4.4 フェーズ 3：辞書ツール

フェーズ 3 では，フェーズ 1 と 2 とはさらに書式が異なるドキュメントが提供された．このことから，ルールベースではコスト・納期の観点から現実的でないと判断し，あるドキュメントに関連するドキュメントを検索することに役立つ IR 手法に方向性を切り替えた．しかしながら，1 つのドキュメント内に仕様と技術の用語が多数存在しているのものがこの時点で確認されており，また，そのファイルを系統的に選別することが困難であった．そこで，図 3 に示すように，ドキュメント間の類似性に着目するのではなく，用語の分析（IR 手法における前処理）を支援する辞書ツールを実現した．辞書ツールではドキュメント内または間で取り上げた 2 つの単語に対して，後述のトレーサビリティリンクの種類とともに関係の有無を与える．図 3 に示す GUI を経由して，単語ペアを一度辞書化すれば，別のドキュメントから同一の単語ペアが得られても無視できるた

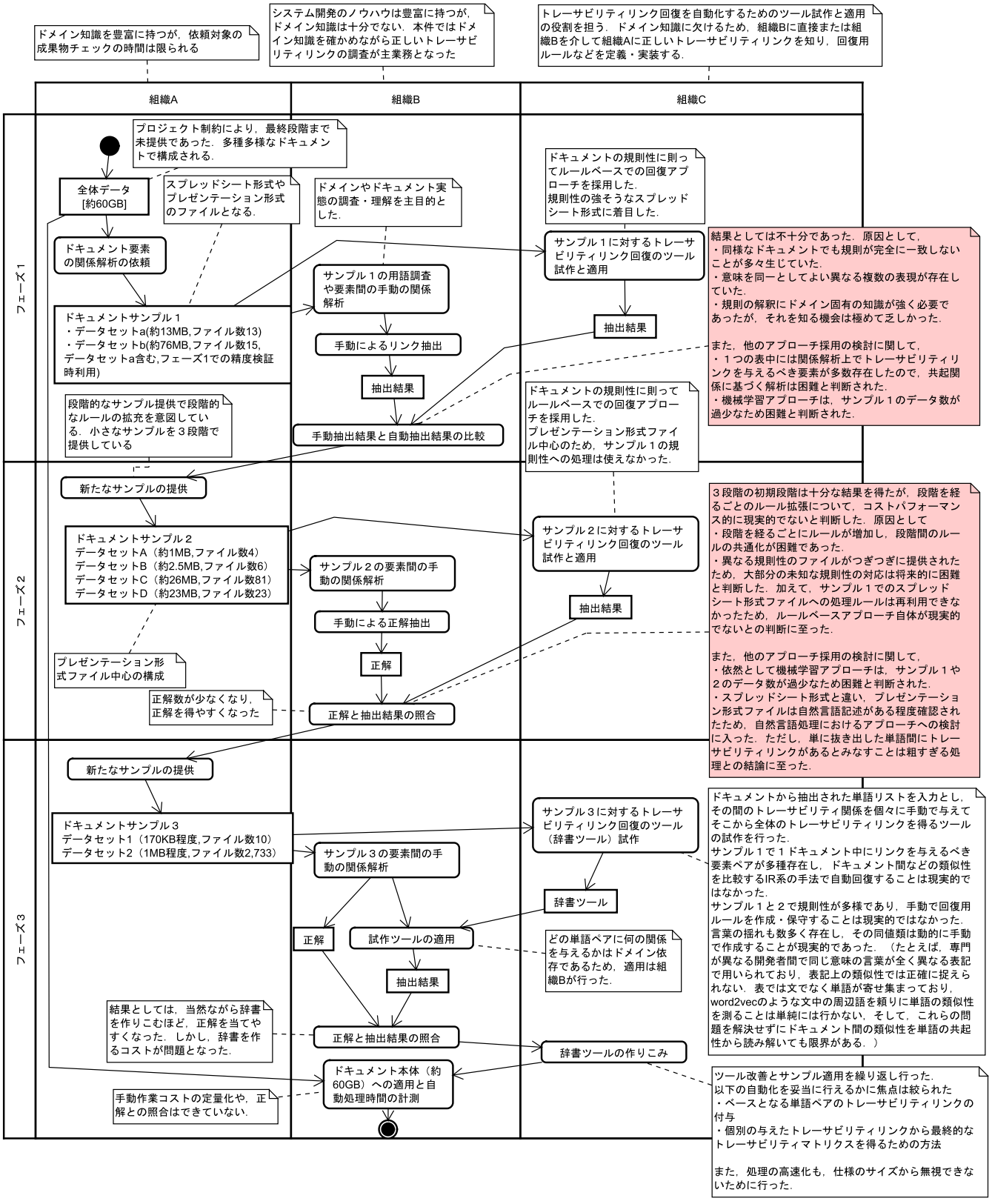

図 2　トレーサビリティリンク回復の実践プロセスの詳細

め，完全な手作業に比べて効率化が望める．

図 4 に辞書ツールの使用プロセスを示す．辞書ツールでは，辞書内の単語ペア（有効辺）がトレーサビリティリンクの候補となる．最終的なトレーサビリティリンクは推移的な関係追跡で得られた単語ペアの内，仕様と技術の用語で構成されるペアを指す．辞書ツールではドキュメント内の単語ペアに正確な関係を与えることが最終目的であるため，辞書とはツールユーザによるトレーサビリティリンク

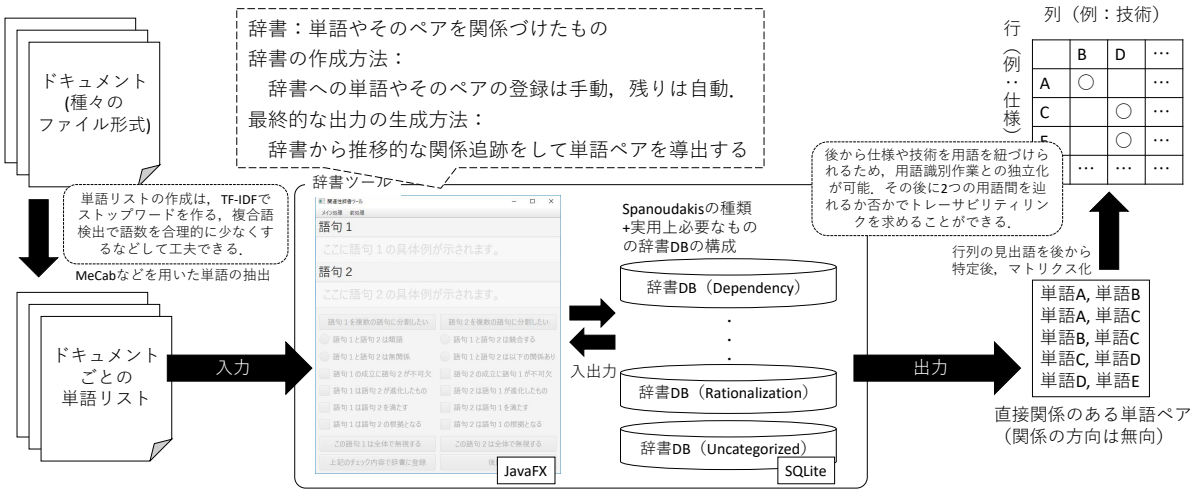

図 3 辞書ツールの構成・フローの概要

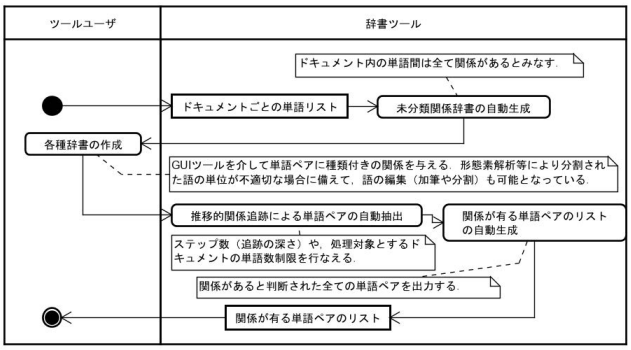

図 4 辞書ツールの使用プロセス

回復の作業状況を記録したものと言える．

辞書ツールへの入力はドキュメントごとに分けた単語リストの txt（1 行に 1 単語）ファイルであり，出力は各種辞書（単語ペアまたは単語のリスト）と推移的な関係追跡を行った結果（関係ありとされた単語ペアのリスト）である．そして，推移的な関係追跡の結果となる単語ペアに対して，後から仕様と技術それぞれの用語を識別することで，トレーサビリティマトリクスを作成できる．本辞書ツールの試作時点では，用語の識別は未完了であった．それ故に単語ペアを得ることと，単語を用語と捉えることは独立したプロセスとできるよう工夫した．

単語間の推移的な関係追跡では，関係ありとされた単語ペアを比較することで関係のある単語ペアをさらに得る．たとえば，単語 A から単語 B へ関係があり，単語 B から単語 C へ関係がある場合，単語 A から単語 C へ関係があるとみなす．このとき，単語ペアに双方向の関係がある場合，辞書では無向関係を与えておく．逆に，片方向の関係がある場合は，有向関係を与えておく．

推移的な関係追跡では，そのステップ数の上限値を与える．ステップ数とは，単語ペアの推移的追跡を何回行うかを表す．たとえば，ステップ数の上限値が 2 であれば，上記例のとおり，2 つの単語ペアを追跡して終了する．ステップ数が少ない場合は推移的追跡は十分とならず，多すぎれば処理時間が増大してあらゆる単語間が関係ありとされやすいことが想定されるため，回復の実践者にパラメータ調整は委ねられた．

また，あまりにも多量な単語が含まれるドキュメントは無関係な単語を関係づける可能性が高いため，単語数を閾値としてドキュメントを処理するかどうかを決定した．たとえば，閾値は 800 語（行）が挙げられるが，これは 800 語以内のドキュメントを処理対象とするということである．この閾値を導入する動機となった適用事例の特徴として，仕様となる用語が 1 万単語以上 1 シートに密集するスプレッドシートがあったことが挙げられる．このようなドキュメントを処理する場合，1 億に上る単語ペアを未分類関係として導出してしまい，不適切なリンク回復や過剰な単語ペア生成時間に繋がることが容易に予想されたために閾値を設けた．

辞書の保存媒体としては，SQLite を選択して読み書き処理が高速となるようチューニングした．この選択には，本ツールの使用状況がかなり限定的であることを前提としている．

4.4.1 トレーサビリティリンクの種類

Spanoudakis ら [18] は，次に示すようにトレーサビリティリンクの種類を整理しており，トレーサビリティリンクは最低限として 2 つの要素（たとえば，ドキュメント）間の関係で表される．

- Dependency（依存）: 依存する側から依存される側への有向関係で表す．
- Generalisation（汎化） / Refinement（洗練）: 汎化では部分から全体への有向関係で表し，洗練では全体から部分への有向関係で表す．
- Evolution（進化）: 進化前から進化後への有向関係で表す．

- Satisfiability（満足）: 要求を満たすものから要求への有向関係で表す．
- Overlap（重複）: 重複したものの間を無向関係で表す．
- Conflict（競合）: 競合したものの間を無向関係で表す．
- Rationalisation（根拠）: 根拠から根拠づける対象への有向関係で表す．
- Contribution（貢献）: 貢献するもの（例：ステークホルダ）から貢献対象（例：要求）への有向関係で表す．

本研究ではまず，Spanoudakis ら [18] の整理したこれらのトレーサビリティリンクの種類ごとに辞書を作る．それら辞書に登録する内容は，関係のある単語ペアであり，その単語ペアや推移的な関係追跡をしたものがトレーサビリティリンク候補となる．各辞書はツールユーザが手動で作成する．

我々は上記の辞書に加えて，次に示す内容を単語のペアやリストとして表す辞書も追加的に扱うこととした．

- 未分類関係（単語ペア）
- 無関係（単語ペア）
- ストップワード（単語リスト）

追加した辞書は，実践の過程で必要と判断されたものであり，トレーサビリティリンク候補を表す単語ペアに対する辞書登録や推移的な関係追跡を効率化支援するものである．

未分類関係とは，同一ドキュメント内の単語ペアには関係がある仮定した初期的な無向関係を表す．未分類関係は初期状態でも，ある程度のトレーサビリティリンク回復結果を得るためのものである．そのため，たとえば単語ペアが依存関係辞書に登録されたとき，そのペアの未分類関係は無効とみなす．ドキュメント内の全ての単語間でペアを作るため，本関係は自動生成できる．

無関係とは，単語ペアが無関係であることを無向関係として表す．この無関係性は，推移的な関係追跡時に，その単語ペア以降の追跡を中止するものである．これにより，あらゆる単語ペアが関係ありとみなされることを合理的に防ぐ．本関係はツールユーザが手動で与える．

ストップワードとは，頻出語などの理由で処理対象から除かれる単語群を表す．辞書ツールへの入力となる単語リストには，辞書作成の作業から除くことのできそうな頻出語が見受けられた．そこで，手動による辞書作成の負担を減らし，かつ，推移的な関係追跡の処理効率を高めるよう対処できるように，ストップワードを表す辞書を導入した．本リストはツールユーザが手動で与える．

4.5 結果

フェーズ 1 からフェーズ 3 までのサンプルドキュメントに対して，各フェーズに対応するツールを適用した結果を表 1 に示す．

4.5.1 ルールベース回復支援ツールの実践結果

フェーズ 1 と 2 はプロジェクト外の作業も並行する前提で，それぞれ 3 か月程度を費やした．その中では各種ツールの開発だけではなく，正解となるトレーサビリティリンクの調査や組織 A と B 間，または組織 B と C 間の議論等も含まれる．ツール開発自体はその改善作業を含め，およそ 1 か月程度かかった．

フェーズ 1 において，サンプルドキュメントセットの 1 つ（約 76MB）における 992 個の正解リンク数に対して，ルールベースの回復ツール 1 が指摘したリンク数は 62 個で，内 30 個が正解であった．そのため，Recall は約 3%であり，Precision は約 48%となった．同セットへの手動作業でも Recall は約 13%，Precision は約 20%であった．

フェーズ 1 と 2 それぞれで回復支援ツールでリンク候補を抽出した時点ではリンク候補が正解かどうかは不明であったからこそ，Recall を高めやすいと考えられたルールベースのアプローチをフェーズ 1 と 2 で採用した．しかし，結果的には初期評価の段階で手動作業でさえ低精度であったことを踏まえ，およそ 1 年という期限の中でドキュメントの多様性に対してツールの汎用性を高めることが現実的でないと組織 B と C で合意に至った．そして，ツールの完成度を高めて完全な評価を実施する前にアプローチを切り替えることとなった．

4.5.2 辞書ツールに関する処理時間

ここでは，辞書ツールの構成の違いによる処理時間について説明する．処理時間を計測するために，適用したデータセットは，データセット 1(170KB 程度)，データセット 2(1MB 程度)，データセット 3(全体データ : 60GB 程度) の 3 種である．これらのデータセットは，フェーズ 3 のみで提供されたものであった．

各データセットは複数の単語リストからなり，1 リストが 1 ドキュメントに対応する．ただし，プロジェクト上で試行不可能であった組み合わせがあるなど，網羅的な組み合わせとはなっていない．高負荷な処理として，未分類関係生成処理と推移的な関係追跡処理の 2 つが挙げられるので，その性能について述べる．

辞書ツールの構成 1 としては，辞書内容をオブジェクトで表し，処理結果をテキストファイルに保存して読み書きする構成である．データセット 1 では，未分類関係辞書生成では 1 秒程度かかった．未分類関係辞書のみを生成した状態で，ステップ数 2 での推移的な関係追跡を行った場合 7 秒程度かかり，ステップ数 5 での追跡では 15 秒程度かかった．データセット 2 では，未分類関係辞書生成では 1 分程度かかった．未分類関係辞書のみを生成した状態で，ステップ数 2 での推移的な関係追跡を行った場合 1 分程度かかり，ステップ数 5 での追跡では 7 分半程度かかった．

辞書ツールの構成 2 としては，辞書内容を hash table で表し，処理結果をテキストファイルに保存して読み書きする構成である．データセット 2 では，未分類関係辞書生成では数秒かかった．未分類関係辞書のみを生成した状態

で，ステップ数 5 での追跡でも数秒かかった．

辞書ツールの構成 3（最終的な構成）としては，辞書内容を hash table で表し，処理結果を関係データベースに保存して読み書きする構成である．データセット 3 では，未分類関係辞書生成では 44.5 時間程度かかった．

辞書ツールの構成 3 では，およそ 2 日間にて処理が完了しているが，未分類関係辞書の生成は一つのドキュメントセットに一度しか行なわないため有効時間内に完了していると考えられる．また，容量ベースで処理時間を比較した場合，未分類関係辞書の作成で KB あたり構成 1 が約 0.06 秒，構成 2 が約 0.03 秒，構成 3 が 2.67 秒でった．最終的な構成の方が遅いように思えるが，単語ペアを処理しているため，最悪で容量の二乗程度のデータを処理しなければならない．その関係を正確に示すためにも，今後は単語ペア数ベースでの処理時間の比較も行っていきたい．また，推移的な関係追跡による精度評価に関しては解析に時間を要するため，今後の課題として解析を進めていきたい．

4.5.3 辞書ツールに関する識別精度

精度については，以下のようなデータが得られた．ここでの識別精度とは，全ての正解中で何件の正解を当てられたかであり，いわゆる再現率にあたる．156 ファイルに対して，未分類関係辞書生成直後では 12%の識別精度であったものに対して，他の辞書追加後では 17%となっている．さらに別のケースとして，450 ファイルに対して，未分類関係辞書生成直後では 35%の精度であったものに対して，他の辞書追加後では 54%となっている．いずれのケースも，捉えるべき正解は同じもの (80 件程度) として精度を求めている．

これらのことから，辞書の質の向上によって精度が向上することがわかった．一方で，ドキュメントセットの特徴（用語の含有率）と辞書の性質（各種辞書の登録単語ペアの実態）を対照する分析を行なうべきことが示唆されたが，これは今後の課題としたい．

5. 教訓

本適用において得られた教訓は次のものがある．

- トレーサビリティリンクの回復アプローチを踏まえてプロジェクトの制約や進行を与えるべきである．たとえば，機械学習系アプローチを採る場合に，予め全てのデータを提供してもらえるようにプロジェクトを設定すべきである．本事例では，フェーズ 1 や 2，フェーズ 3 のデータセット 1 等各データセットのファイル数が 100 に満たないサンプルで回復を試行していた．そのために，終盤まで機械学習系アプローチを採るべきかどうかの検討が困難な側面があった．
  最終的には組織 A は全体データを提供していたため，進め方によっては全てのデータを初期から提供することは可能であったと考えられる．早期のデータ提供を阻害する技術的な要因としては，60GB の多様なドキュメントには DB に格納されているもの等もあり，そこからのデータ抽出など時間のかかるタスクが存在していたことがある．このことを踏まえ，計画的かつ早期なデータ提供方法はプロジェクト初期から検討されるべきであったと考えられる．
- データの多様性を初期段階で把握すべきである．たとえば，ルールベースアプローチを採ることが現実的かどうかを早期に判断できるべきである．これは，本事例では，フェーズ 2 の終了時までルールベースからアプローチを変更すべきか判断できなかったことによる．組織 A が段階的なデータ提供方式を採っていたが，これはサンプリングと捉えれば現実的な分析手順の一つとも捉えられる．前述のように，最終的には組織 A は全体データを提供していたことから，サンプリング戦略を検討できる余地はあった．本事例においては，様式が厳密には一致はしないものの，たとえば仕様の一覧や技術と製品のマトリクスを整理するなど目的ごとにドキュメントが分割されていた傾向は見られた．その場合のサンプリング戦略では，たとえば，ドキュメント全体に対して方向性が同じドキュメントを数個ずつ得られていれば，ルールベースアプローチが現実的かどうかは早期に判断できた可能性がある．本プロジェクトでは，方向性が同じドキュメントが数個ずつ提供されてはいたものの，ドキュメント全体からすると種類数が限定的であったために判断が遅れたと考えられる．目的や趣旨を考えずに無秩序な様式で多数のドキュメントを作成することは通常考え難いため，他プロジェクトでも考慮すべき事項であると考えられる．
- 分散環境における大規模なドキュメントに対するトレーサビリティリンクの回復では，ワークフローを考慮して各種作業を互いに独立にしておくとよい．本事例では，複数の組織が協働する形で回復を試みており，トレーサビリティリンクの正解の調査と，回復支援ツールの開発の担当は異なる組織が行っていた．より具体的には，辞書ツールの開発は組織 C のタスクであり，辞書ツールへの入力となる単語リストを合理化することや，辞書ツールが出力した単語ペアを仕様や技術に結び付けることは組織 B のタスクであった．これらのことから，たとえば，用語の識別や単語ペアの整理などの作業を，どちらを先に行っても良いようプロセスやツールを整備すべきである．これらの作業を独立化し，それぞれの作業結果を組み合わせて最終結果が得られるようにワークフローを構成することは，このような分散環境による協働作業を現実的に行う上で重要である．
- 辞書アプローチでは，ストップワードや用語（複合語）を如何に適切に得るか重要となる．多量のドキュメン

表 1　ツールの比較概要

手法	識別精度	スケーラビリティ	処理効率	説明
ルールベース回復支援ツール 1	低	低	低	識別制度は，書式等の間の細かい差異に網羅的に対応できず，また網羅することはコスト的に現実的でないため低い．スケーラビリティは，未知の書式等に対応できないため低い．処理効率は，Office ファイルの読み込みや，表インスタンスの作成に時間を要するため低いとしている．
ルールベース回復支援ツール 2	低	低	低	識別制度は，書式等の間の細かい差異に網羅的に対応できず，また網羅することはコスト的に現実的でないため低い．また，形態素解析の結果から正しい用語（複合語）を導くことも困難なケースもあったことも低い要因に挙げられる．スケーラビリティは，未知の書式等に対応できないため低い．処理効率は，Office ファイルの読み込みや，MeCab による形態素解析に時間を要するため低いとしている．
辞書ツール	低〜高	高	高	識別精度は，辞書や入力された単語リスト（用語を適切に与えられるかなど）によって変化するが，それらの質が高ければ，高い識別精度を得られる見込みは得た．スケーラビリティは，単語リストを用意すれば利用可能であるため高い．処理効率は，単語リストのみの処理に集中して比較的高速となるため高いとしている．これはスケーラビリティを高いとする要因にもなる．なお，組織 B からは多様性の高いドキュメントに対して辞書ツールのアプローチによりトレーサビリティリンク回復方法のビジョンが見えたという意見を得た．

識別精度，スケーラビリティ，処理効率は 2 値（低，高）で相対的に表現されている．

トには頻出語が登場する．また，本来は用語であるにも関わらず，形態素解析だけでは一般語・頻出語に分類されてしまう語も登場する．フェーズ 3 の辞書ツールでは，辞書への登録が手動であるため，その状況は無用な労力を生みやすい．そのため，入力となる単語の導出方法は重要であった，本事例では，辞書ツールへの入力単語リストに閾値を設けることとなったが，除外される単語リストを減らすためにも必要な処置と考えられる．

たとえば，頻出語と用語が区別できれば，ドキュメントごとの語の重要度を計れる TF-IDF[14] 等を使い，重要度の低い語をストップワード化することができる．また，用語が導出できれば，そもそも関係を与える単語数が減り，また表現の規則性からある程度は自動での辞書化ができる可能性も見込まれる．さらには，他の自動回復アプローチも取りやすいであろう．

- 初期から全てのデータで回復を試行すべきである．GB レベルのドキュメントに対しては，処理効率といった性能も当然重要となる．ツールの正しさは，サンプルを切り出して用いて確かめるとしても，性能に関しては全てのデータがなければ評価し難い．フェーズ 3 の終盤でのデータセット 2 では，2,000 を超える約 1MB のドキュメントが提供され，辞書ツールでの未分類関係生成時間がそれ以前のデータセットと比べると著しく増加した．一般論としても多量なドキュメントに対するアルゴリズムの調整は早期に行うべきであり，一度の試行にかかる時間が長くなることから短期間での性能改善は現実的とは言えない．このことを踏まえて，プロジェクトの計画を練る必要がある．

6. 関連研究

従来の Retrospective な手法では，IR（たとえば [5]）やルールベース（たとえば [19]），機械学習アプローチ（たとえば [16]），深層学習アプローチ（たとえば [10]）があるが，これらはいずれもトレーサビリティリンクの自動回復に焦点を当てている．もちろん，ソースとシンクにリンクを与える作業自体が労力が高いために，その支援は自然である．

しかしながら，大規模なドキュメントを対象とする実プロジェクトでは，精度をさておいたとしても結果を得れば終わりではない．たとえドメインエキスパートであっても一部の人員では全ての正解を把握できない状況下において，どのようにして正解を当て，また確認していくかが重要となる．本研究では，実プロジェクトの中で，回復の自動化以前の用語関係の整理に着目した現実的な支援手法を考案し，その適用結果をケース・スタディ的に示したことが新しい．また，プロジェクト制約から見た機械学習・深層学習アプローチを得ること，ルールベース手法の汎用化，類似ドキュメントに基づく用語の探索について，それぞれの困難さについても触れた．今後としては，入力となる単語リストを如何に適切に限定するかや，自動でのトレーサビリティリンクの回復手法と連係する現実的なプロセスを検討する必要があるであろう．

本研究で扱った辞書ツールへの入力データ（単語リスト）における単語数の合理的な削減に寄与する技術として

頻出語の重みを下げる TF-IDF[14] や，単語間の意味的類似性を考慮する LSI(Latent Semantic Indexing)[8], [9] や，word2vec[15], doc2vec[12] といったものがあり，これらを併用することで辞書ツールの利用方法を改善することが考えられる．併用の可能性は認識しつつも，今回は特に正解となるトレーサビリティリンクのほとんどが不明であるところを開始点としたため，このような技術を用いた単語数の削減のよる情報欠落のリスクを回避することが念頭にあった．今後としては，適用事例のデータに対するこれら技術の効果測定を実施することを検討していきたい．

7. まとめ

本稿では長年にわたる大規模プロジェクトで作成されてきたドキュメントを事例として，産学連携によりトレーサビリティリンク回復を実践した結果について報告した．そこでは，単に技術的な精度を高める以上にプロジェクトのコンテキストに合わせた技術の選定や回復手法の選定に細かい調整が必要であることが強く示唆されたものであった．また，これらの選定を早期に収集できることが，その後の回復手法の考案タスクのスケジューリングを適正化することに重要であるとの結論を得た．

今後の課題としては，考案した手法の網羅的な定量評価や，最終的な識別精度の評価などの評価を充実化することが挙げられる．また，今回導入したリンク種類は，実践者から必要となる可能性が指摘されたが，その実際を今後調査していく．そして，本研究で得られた知が他プロジェクトでも適可能なものであるかを適事例を増やして検証する．さらに，約 60GB に対する未分類関係生成処理が 44.5 時間かかるという結果は実用性や検証可能性に悪影響を与えるため，この改善手段として，入力となる単語リストを如何に適切に限定する方法や，自動でのトレーサビリティリンクの回復手法と連係する現実的なプロセスを検討する．

謝辞

本研究の実施にあたり，多大なるご協力・ご助言を賜りましたガイオ・テクノロジー株式会社の関係各位に深謝致します．また，本研究は JSPS 科研費 JP16H02804 の助成を受けたものです．

参考文献

[1] Antoniol, G., Canfora, G., Casazza, G., De Lucia, A. and Merlo, E.: Recovering traceability links between code and documentation, *IEEE Transactions on Software Engineering*, Vol. 28, No. 10, pp. 970–983 (2002).

[2] Apache: Apache POI, https://poi.apache.org/ (2019).

[3] Asuncion, H. U., Asuncion, A. U. and Taylor, R. N.: Software traceability with topic modeling, *2010 ACM/IEEE 32nd International Conference on Software Engineering*, Vol. 1, pp. 95–104 (2010).

[4] Bouillon, E., Mader, P. and Philippow, I.: A Survey on Usage Scenarios for Requirements Traceability in Practice, *Requirements Engineering: Foundation for Software Quality* (Doerr, J. and Opdahl, A. L., eds.), Berlin, Heidelberg, Springer Berlin Heidelberg, pp. 158–173 (2013).

[5] Cleland-Huang, J., Berenbach, B., Clark, S., Settimi, R. and Romanova, E.: Best Practices for Automated Traceability, *Computer*, Vol. 40, No. 6, pp. 27–35 (2007).

[6] Cleland-Huang, J., Hayes, J. H. and Domel, J. M.: Model-based traceability, *2009 ICSE Workshop on Traceability in Emerging Forms of Software Engineering*, pp. 6–10 (2009).

[7] Cleland-Huang, J., Gotel, O. C. Z., Huffman Hayes, J., Mader, P. and Zisman, A.: Software Traceability: Trends and Future Directions, *Proceedings of the on Future of Software Engineering*, FOSE 2014, pp. 55–69 (2014).

[8] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K. and Harshman, R.: Indexing by latent semantic analysis, *Journal of the American Society for Information Science*, Vol. 41, No. 6, pp. 391–407 (1990).

[9] Dumais, S. T.: Improving the retrieval of information from external sources, *Behavior Research Methods, Instruments, & Computers*, Vol. 23, No. 2, pp. 229–236 (1991).

[10] Guo, J., Cheng, J. and Cleland-Huang, J.: Semantically Enhanced Software Traceability Using Deep Learning Techniques, *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*, pp. 3–14 (2017).

[11] Kudo, T.: MeCab, https://taku910.github.io/mecab/ (2013).

[12] Le, Q. and Mikolov, T.: Distributed Representations of Sentences and Documents, *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, JMLR.org, pp. II–1188–II–1196 (2014).

[13] Lormans, M. and van Deursen, A.: Reconstructing Requirements Coverage Views from Design and Test Using Traceability Recovery via LSI, *Proceedings of the 3rd International Workshop on Traceability in Emerging Forms of Software Engineering*, TEFSE '05, pp. 37–42 (2005).

[14] Manning, C. D., Raghavan, P. and Schütze, H.: *Introduction to Information Retrieval*, Cambridge University Press, New York, NY, USA (2008).

[15] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. and Dean, J.: Distributed Representations of Words and Phrases and Their Compositionality, *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, Curran Associates Inc., pp. 3111–3119 (2013).

[16] Mills, C., Escobar-Avila, J. and Haiduc, S.: Automatic Traceability Maintenance via Machine Learning Classification, *2018 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pp. 369–380 (2018).

[17] Mills, C.: Towards the Automatic Classification of Traceability Links, *Proceedings of the 32Nd IEEE/ACM International Conference on Automated Software Engineering*, ASE 2017, pp. 1018–1021 (2017).

[18] Spanoudakis, G. and Zisman, A.: Software Traceability: A Roadmap, *Handbook of Software Engineering and Knowledge Engineering*, Vol. 3, pp. 395–428 (2005).

[19] Spanoudakis, G., Zisman, A., Perez-Minana, E. and Krause, P.: Rule-based generation of requirements traceability relations, *Journal of Systems and Software*, Vol. 72, No. 2, pp. 105 – 127 (2004).