Vol.2019-GN-106 No.24
Vol.2019-CDS-24 No.24
Vol.2019-DCC-21 No.24
2019/1/24

統合ナレッジ管理に向けた会話スレッド自動抽出手法の検討

角田 啓介[1,a)]　坂井 俊之[1]　鈴木 貞重[1]　宮下 直也[1]　箕浦 大祐[1]

概要：企業内には業務に役立つ情報がドキュメント，コミュニケーションログ，システムログ等様々な形式で蓄積されている．我々はそれらを一元的に管理し，ユーザの必要に応じて最適なナレッジを提示できる統合ナレッジ基盤の確立を目指している．その一環として本稿では，コミュニケーションログであるメッセンジャサービスにおける会話ログを対象とし，会話ログから意味のある一連の会話である「会話スレッド」を自動的に抽出する手法を提案する．また，提案手法によって抽出した会話スレッドの活用事例として，会話スレッドとドキュメントを関連付けを実施し，その効果について考察する．

A Study on Automatic Conversation Thread Extraction for Integrated Knowledge Management System

1. 緒論

企業は共通の目標を持った複数の人間で構成され，目標達成のために活動している．企業が効率的かつ効果的に目標を達成するために，近年では業務に役立つ情報である「ナレッジ」を企業内で管理し，ユーザとなる企業構成員が業務において自在に活用できるようにする「ナレッジ管理(ナレッジマネジメント)」が注目されてきた．ナレッジには形式知と暗黙知とがあり，暗黙知はユーザにとって個人的，主観的かつ経験的ないわゆるアナログ的知識，形式知は企業的，客観的かつ理性的ないわゆるデジタル的知識と定義されている [1]．そして，ナレッジ管理の代表的なモデルである SECI モデル [1] ではそれらのナレッジの変換として，共同化，表出化，結合化，内面化の 4 つが挙げられている．

一方，情報システムを用いて SECI モデルで述べられたようなナレッジ管理を支援しようとする試みも数多くなされてきた．例えば，企業内にドキュメントという形で作成・蓄積されたナレッジを共有した上で，スケジュール管理等と連携した活用を実現するグループウェア [2]，大量のナレッジを検索するための全文検索技術 [3]，ユーザ個人の持つ暗黙知や，個人的・主観的だが記述可能なナレッジである仲介知の共有を促す企業内 SNS [4] などが挙げられる．しかしながら，従来技術やサービスには，本来業務以外でもユーザの能動的な操作を必要とする点，様々な箇所に様々な形式として散在したナレッジを組みわせて活用するのが難しい，といった課題がある．

我々は従来技術やサービスにおける課題を解決するため，ドキュメント，コミュニケーションログといった様々な形式のナレッジを一つのシステムへ自動的に取り込み，整理した上で，相互の関連づけや検索，活用を可能とする統合ナレッジ管理の確立を目指している．本稿では統合ナレッジ管理の実現に向けた 1 つの課題として，コミュニケーションログの中でも近年普及が著しい企業内メッセンジャサービス [5] における会話ログの管理を取り上げる．メッセンジャサービスにおける会話ログは電子メールのメッセージと異なり，非常に短文かつ簡潔なメッセージから成り立っており，それらを密に交換することでコミュニケーションが成立している．これまで，メッセンジャにおける会話ログの分析は，含まれている単語の頻度を分析することによるトピック推定 [6] などが行われてきたが，従来手法では会話ログ自体に含まれるであろう議論や意思決定過程を抽出し，他の形式のナレッジと共に扱うことは困難であった．しかも，会話ログの 1 つ 1 つは簡潔な短文であるため，含まれている情報量が非常に少なく，ドキュメント等と同様に言語処理技術に基づいて整理し，活用することが困難である．

そこで本稿では，まず，従来のナレッジ管理における課題を挙げ，それらを解決するための概念である統合ナレッ

[1] NTT コムウェア株式会社
NTT Comware Corp., Minato, Tokyo 108–8019, Japan
[a)] tsunoda.keisuke@nttcom.co.jp

ジ管理を提案する．その上で，統合ナレッジ管理における目標に基づき，会話ログに含まれる議論や意思決定過程をナレッジとして管理するため，発言者と発言時刻に着目した，意味のある一連の会話である会話スレッドを自動で抽出する手法を提案する．そして，実際のソフトウェア開発業務で利用された会話ログに適用することで，その有効性と課題について述べる．

2. ナレッジ管理

2.1 形式知，暗黙知と SECI モデル

野中によると，人の持つナレッジは形式知と暗黙知に分類され，形式知はドキュメントやマニュアルに記載された企業として活用できるナレッジ，暗黙知は個人が持っている，ドキュメントやマニュアルなどに変換していない主観的なナレッジとされている [1]．その上で野中は，形式知と暗黙知の変換モデルとして SECI モデルを提案している [1]．SECI モデルでは，ナレッジ管理において個人の持つ暗黙知から企業の形式知へと昇華し，かつそれを個人が理解するというプロセスとして，共同化 (Socialization)，表出化 (Externilization)，連結化 (Combination)，内面化 (Internalization) の 4 つを挙げている．

SECI モデルに基づき，これまで様々な情報システムによるナレッジ管理支援，特にナレッジの共有に関する技術やサービスが提案されてきた．代表例としては，様々なファイルをアップロードし，全文検索技術 [3] によって共有・活用できるようにするシステム [7] や，それらをスケジューラなどと連携させて活用することができるグループウェア [2] が挙げられる．

2.2 企業内 SNS と仲介知

山本らは，企業内で電子的にやりとりされている内容を考察し，形式知でも暗黙知でもない知として「仲介知」という概念を提唱した [8]．仲介知とは，形式知のように記述可能であり SNS へ投稿可能であるが，暗黙知のように個人が持っているだけで何に役立つかは不明瞭なナレッジとしている．このようなナレッジが企業内 SNS へ投稿された上で，それを他の構成員が参照し，自身の業務に役立てるケースがあるため，ナレッジ管理において企業内 SNS は有効であると考えられる．一方で，企業内 SNS は投稿数が減ると参照者が減り，ますます投稿数が減るといった悪循環に陥ってしまったという失敗事例が多いことも指摘されている [9]．

2.3 ナレッジの形式

本稿では，ドキュメントに記載されている，会話に含まれているといった，ナレッジが表現形態をナレッジの形式と呼ぶこととする．

ナレッジの形式として最も一般的なのはドキュメントであり，企業内では業務マニュアル，ソフトウェア仕様書，稟議文書など無数と言えるほど存在する．多くのドキュメントは Microsoft Office などのアプリケーションで作成され，PC や共有ファイルサーバに保存されている．また，人と人のコミュニケーションにも，議論の過程やその決定内容といった重要なナレッジが含まれる．コミュニケーションの形態として，人同士の会話，会議，メール，近年普及が進む Slack [5] 等のメッセンジャサービスが挙げられる．このうち，主な会議の内容は議事録となりドキュメントとして残り，メールに関してもメールサーバに情報が蓄積される．一方，メッセンジャといったコミュニケーションの内容は，一つ一つの発言が断片的であるためナレッジとして活用されることは少なく，従来の技術やサービスでは，発言に含まれる頻出語やその変化傾向であるトレンドの抽出し提示する機能 [6] や，会話に含まれた画像などのマルチメディアデータを抽出し，再利用を促す手法 [10] が提案されてきた．しかしながら，複数人での会話自体をナレッジとして活用する方法は未だ提案されていない．

2.4 課題

上述した従来のナレッジ管理には 2 つの課題があると考えられる．

1 つは多くのシステムはユーザに対し，ドキュメントのアップロードやタグ付与をはじめとしたナレッジの整理といった能動的な作業を要求するため，ユーザの利用頻度が減少するとナレッジも陳腐化・散乱してしまい，システム自体の有用性が低下してしまう点である．

もう 1 つは，ナレッジにはドキュメント，SNS の書き込み，システムログ，コミュニケーションログなど多種多様であるが，多くの場合はそれらが独立したシステムで管理・活用されており，ナレッジの形式を超えた活用が困難な点である．

3. 提案

3.1 統合ナレッジ管理

我々はナレッジの形式を超えて 1 つのシステムでナレッジを統合的に管理し，活用できる統合ナレッジ管理を提案する．統合ナレッジ管理の概念図を図 1 に示す．統合ナレッジ管理における目標は下記の通りである．

(1) ユーザに対して本来活動以外でのシステムに対する能動的な操作は極力要求しない
(2) ナレッジの形式に関わらず 1 つのシステムで統合的に管理し，相互の関連性を算出し，それに基づいて活用できるようにする

そのため統合ナレッジ管理では以下 2 機能を備える．本稿ではそのうち，機能 1 の一部である，メッセンジャでの会話ログを意味のある 1 つの会話スレッドとして自動抽出する手法を提案する．

Vol.2019-GN-106 No.24
Vol.2019-CDS-24 No.24
Vol.2019-DCC-21 No.24
2019/1/24

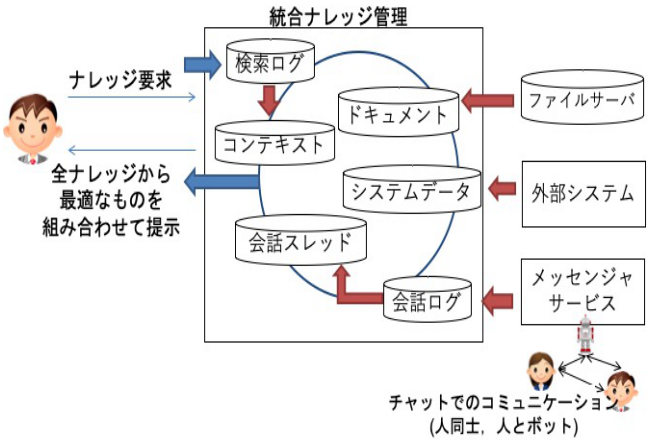

図 1　目指す統合ナレッジ管理の全体像
Fig. 1　Overview of integrated knowledge management system

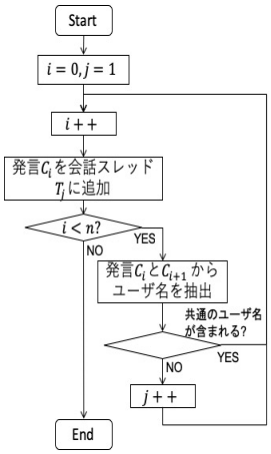

図 2　発言者に着目した処理フロー
Fig. 2　Extraction method based on speaker

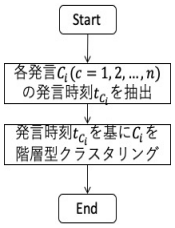

図 3　発言時刻に着目した処理フロー
Fig. 3　Extraction method based on time

機能 1　在来のシステムであるファイルサーバ，メッセンジャサービスなどからファイルや会話ログなどを自動的に収集・整理し，ナレッジとして 1 つのシステムへ取り込む機能

機能 2　機能 1 で取り込まれたナレッジを相互に関連付け，活用できる機能

3.2　会話スレッド抽出手法

本節では会話スレッドの抽出方法について述べる．まず前提として，会話は単一のメッセンジャサービスを通じて人間同士で行われ，その中におけるユーザの発言の集合を会話ログと定義する．そして，発言には発言者名，発言時刻．発言内容が含まれているものとする．また，発言の集合である会話ログは，メッセンジャサービスのサーバより取得できるものとする．

本手法では，会話における発言者および発言時刻に関するいくつかの考え方を基にしている．まず発言者に関しては，同じ話題を扱い関連のある発言においては，発言中に返答を期待する相手，または返信先の発言の発言者名が含まれている可能性が高いという点である．そして発言時刻に関しては，メッセンジャを含むコミュニケーションにおいて近い時刻に発言された内容は関連している可能性が高いという点と，ある一つの話題の会話はある短い時間的に集中してなされる可能性が高いという点にある．そこで発言者と発言時刻を用いて，教師なし学習であるクラスタリング手法を適用することで．会話スレッドの自動抽出を試みる．そして，発言者，発言時刻をそれぞれ用いた場合と，両方用いた場合の結果について比較する．

上記の考え方に基づいた，発言者に着目した処理フローを図 2 に，発言時刻に着目した処理フローを図 3 にそれぞれ示す．

4.　評価

4.1　評価データ

評価にはアジャイル開発の手法を用いたソフトウェア開発中に slack で交わされた会話 394 件を用いた．期間は 2018 年 5 月 13 日から 6 月 27 日まで，開発者数は 7 名である．

4.2　評価指標

会話スレッド抽出結果が正解となる会話スレッドとどの程度類似しているかを評価するために，単純にスレッドの切り替え点の一致率を算出することも可能であるが，そうすると切り替え点がわずか 1 発言だけ正解と異なった場合も，全く正解とかけ離れた結果であっても同じように不一致として評価されてしまう問題がある．また，1 発言=1 会話スレッドと推定された場合も，上記の手法では正解率が 100%となってしまう問題があり，会話スレッド抽出手法の評価として不十分と考えられる，そこで本稿では情報検索における評価指標を参考に，次に述べる 3 つの指標を用いた．1 つは，各推定会話スレッドにおいて，スレッド内の会話が同じ正解会話スレッドに含まれる割合であり，情報検索における適合率と類似している概念であることから本稿ではこれをスレッド適合率と呼ぶ．2 つ目は各正解会話スレッド内の会話が同じ推定会話スレッドに含まれる割

表 1　評価結果
Table 1　Evaluation result

手法	スレッド適合率	スレッド再現率	F 値
1	0.555	0.909	0.689
2	0.623	0.898	0.736
3	0.681	0.773	0.724

合であり，情報検索における再現率と類似している概念であることから本稿ではこれをスレッド再現率と呼ぶ．3 つ目はこれら 2 つの調和平均であり，本稿ではこれを F 値と呼ぶ．

4.3　評価方法

評価データに下記 3 パタンの手法を適用した結果と，開発に参加したメンバが手動で作成した会話スレッドに基づいた評価指標を基に評価を行った．なお，手法 2, 3 における発言時刻に基づくクラスタリングにはウォード法を用い，閾値はそれぞれ 0.7, 0.5 とした．

手法 1　図 2 で示した発言者に着目した手法のみで会話スレッドを抽出．

手法 2　図 3 で示した発言時刻に着目した手法のみで会話スレッドを抽出．

手法 3　発言者に着目した手法で抽出した会話スレッドを新たな会話の集合とみなし，さらに発言時刻に着目した手法を適用して会話スレッドを抽出．

4.4　結果

表 1 に結果を示す．表 1 をより，F 値でみると最も精度が高いのは発言時刻のみに着目した手法 2 であるが，手法 3 との差異は小さいこと．また手法 1,2 のみではスレッド適合率よりスレッド再現率が高い傾向があり，手法 3 ではその 2 指標の差は小さくなっていることがわかる．

4.5　考察

表 1 をより，手法 2 が最も精度が高いが手法 3 との差は小さいこと，また手法 1, 2 を用いた場合は手法 3 と比較してスレッド再現率が高い傾向にあることが明らかになった．本節では，具体的なスレッド抽出結果を基に，各手法における結果の特徴について詳しく考察する．なお，以下に示す結果におけるスレッドは，同一手法内で同じ数字であればそれらの発言は同じ会話スレッドに属することを表す．また，企業内の機密情報に関する発言内容については伏せ字を利用している．

まず，手法 1 によって正しく推定できた例を表 2 に示す．表 2 の例では，3 つ目の A の発言で一度会話が途切れ，1 時間半後の B の発言により会話が再開していると考えられる．このようなケースにおいて手法 2 では発言時間間隔が相対的に開いてしまうところで別の会話スレッドと推定されてしまうが，手法 1, 3 では本文中に発言先となるユーザ名が含まれていることから，同一会話スレッドと正しく推定される．このように，手法 1 は明確に発言先が指定されるような，会話への参加者が固定的な会話に適しているといえる．

次に，手法 2 によって正しく推定できた例を表 3 に示す．表 3 の例では開発した API の仕様を確認しているが，各発言は 1 から 5 分間隔でなされており，手法 2 によって正しく会話スレッドが推定されている一方，A と B が会話を始めた後，それを見た第三者である C が横から発言しており (4 発言目)，各発言に必ずしも発言先が指定されているわけではないため，手法 1 では正しく会話スレッドが抽出されていないことがわかる．このように，手法 2 は短時間で会話がなされるケースであれば，2 者間の発言に第三者が割り込んでくるような，発言者が固定的でないケースにおいても有効であるといえる．

また，手法 1, 2 では正しく推定できなかったが，手法 3 によって正しく推定できた例を表 4 に示す．表 4 の例ではロジックにおける DB の参照方法について確認しているが，1 発言目と 2 発言目の時間間隔が他と比べて開いていること，4 発言目は直前の発言を補足するために JSON ファイルを貼り付けており発言先ユーザ名が含まれていないことから，手法 1, 2 いずれもこれらの発言を 1 つの会話スレッドとして推定できていないことがわかる．しかしながら，手法 3 では発言先と発言時刻の両方を用いて会話スレッドを推定しているため，このようなケースにおいても正しく推定できていることがわかる．このように，手法 3 は時間間隔が開く発言と，発言先を省略された発言を含む，2 者間での比較的コンテキストが高い発言において有効であるといえる．

最後に，いずれの手法でも正しく会話スレッドを推定できなかった例を表 5 に示す．表 5 の例では画面に表示する時刻フォーマットを確認しているが，2 発言目だけが DB リセットの周知であり，その前後とは関係のない発言であることがわかる．そのため，本稿で提案した，発言者に基づいた手法 1，発言時刻に基づいた手法 2，およびそれらを組み合わせた手法 3 のいずれをもっても 3 発言目を別の話題と判別できないため，結果として正しく会話スレッドを抽出できなかったことがわかる．

まとめると，本手法は評価用データのように，途中で話題の割り込みが少ない業務でのコミュニケーションに対しては，一定の精度 (F 値 0,7 以上) で会話スレッド抽出ができるといえる．特に，メッセンジャ特有の現象として，発言において発言先が自明であるため，わざわざ言及しないケースも多いため，手法 2，3 のように発言時刻を用いた方が手法 1 のように発言者を用いるより高い精度で会話スレッドを抽出できたと考えられる．一方で，様々な話題が交錯する会話ログにおける会話スレッド抽出には，異なる

表 2　スレッド抽出結果例 (1)
Table 2　Example of extracted thread(1)

発言時刻	発言者	本文	会話スレッド			
			正解	手法 1	手法 2	手法 3
16:14:43	A	@B @C xxxx 用データの蓄積および、割り振り可能判定の改修まで終了しました。 @B capacities テーブルのバッチとの結合についての打ち合わせがあると認識していますが、あっていますか?	1	1	1	1
16:18:30	B	@A ありがとうございます。打ち合わせしましょうか、今大丈夫ですか？	1	1	1	1
16:18:49	A	@B 大丈夫です	1	1	1	1
17:49:33	B	@A 先程の xxx 推定用の時刻をどこから起算するかですが、C と相談した結果、assigned_at (今のまま) から起算することになりました。	1	1	2	1
17:49:56	A	@B 承知しました	1	1	2	1

表 3　スレッド抽出結果例 (2)
Table 3　Example of extracted thread(2)

発言時刻	発言者	本文	会話スレッド			
			正解	手法 1	手法 2	手法 3
10:56:44	A	現在、xxx API のうち、新しくドキュメントを作成する API が叩けないでいます。以下のように叩いても、生の html で 404 が返ってきます。 splash などは叩けるので、叩き方が間違っていると思うのですが... (コマンドの中身)	1	1	1	1
10:58:28	B	@A 確認してみますねー	1	1	1	1
10:59:19	C	‘/v1/conversations/conversations‘ じゃないでしょうか。splash が通るのは ‘/v1/conversations/splash‘ ですよね	1	2	1	1
10:59:41	A	@C 確認します	1	2	1	1
10:59:55	B	そうかも？すみません、実装したの私なのに忘れました	1	3	1	1
11:00:27	C	‘/v1/conversations‘ までが basePath だった気がします	1	4	1	1
11:01:10	B	その通りでした！	1	5	1	1
11:01:22	A	そのとおりでした	1	6	1	1
11:07:46	A	@C 以前頂いた swagger の html をテキストエディタで確認したところ、‘”basePath”: ”vi/conversations”‘ だったのでおっしゃるとおりでした。	1	7	1	1

話題の発言を抽出するような手法を導入する必要があると考えられる．

5. 会話スレッド抽出の応用例

会話スレッドを抽出した結果，ひとまとまりの会話に含まれる意思決定過程や議論を，他のナレッジであるドキュメントなどと同等に文書として扱うことができるようになる．本節ではその活用例として，ソフトウェアの仕様が記述されたドキュメントの TF-IDF 値が高い単語で抽出した会話スレッドを検索し，ドキュメントと検索結果に含まれる会話スレッドを関連づけた結果，ドキュメントに記述された仕様の決定過程を把握できるようになった例を挙げる．表 6 に，仕様変更に関するドキュメント (開発におけるチケット) の例を，表 7 に提案手法によって抽出され，ドキュメントにおいて TF-IDF 値が高い単語を含んでいることで当該ドキュメントと関連付けられた会話スレッドの例を示す．なお，表 7 に示した会話スレッドは，手法 1, 2, 3 のいずれにおいても正しく抽出されている．

表 6 はあるアプリケーションにおけるユーザ通知メッセージの内容を記述したものであるが，関連付けられた表 7 の会話スレッドにはこのメッセージ変更は営業担当が示した案から決定された経緯が含まれていることがわかる．このように，単純な単語に基づいたドキュメントと抽出された会話スレッドの関連付けであっても，ドキュメントとそこに記載された内容を決定した経緯や根拠を含む会話スレッドを関連付けることができ，従来のナレッジ管理では実現できなかった，いわゆる know-why の理解を支援できる可能性があることがわかる．これは，会話ログにおける単一の発言は短い上にコンテキストに基づいた表現の省略が多く，単体では処理が難しかったが，提案手法に基づいて複数の発言を会話スレッドとしてまとめることで，会話スレッドに含まれる単語が増え，関連付けのような処理が可能になったためである．今後は会話スレッドとドキュメントに対し word2vec [11] や doc2vec [12] のような技術を適用することで，ドキュメントと会話ログといったナレッジの形式を超えた活用が容易になると考えられる．

6. 結論

本稿では，ユーザにファイルアップロードやタグ付けといった本来業務以外の作業を極力強いず，かつナレッジの

Vol.2019-GN-106 No.24
Vol.2019-CDS-24 No.24
Vol.2019-DCC-21 No.24
2019/1/24

表 4　スレッド抽出結果例 (3)
Table 4　Example of extracted thread(3)

発言時刻	発言者	本文	会話スレッド			
			正解	手法 1	手法 2	手法 3
13:10:04	A	@B 帳票割り振りロジックを作っていますが，「現在誰がどの種別の帳票を確認中か」という情報は DB で言えばどのフィールドを参照すると良いのでしょうか?なんとなく xxx DB の checker や task テーブルから取得できそうなのですが，念のため確認させてください．	1	1	1	1
13:33:39	B	@A 以下のように確認できます。(詳細な手順とソースコード)	1	1	2	1
13:36:08	B	@A タスクのステータスが、xxx のチケットステータスと紐付いている関係上、環境によって done 状態に紐づく整数値が変化します。現在デモで使用している環境だと、以下のような設定です。(URL)	1	1	2	1
13:37:41	B	(JSON ファイル)				
13:37:50	A	@B ありがとうございます．割り当て時に毎回この処理をするのはそこそこ時間がかかりそうですが，やってみます．	1	2	2	1

表 5　スレッド抽出結果例 (4)
Table 5　Example of extracted thread(4)

発言時刻	発言者	本文	会話スレッド			
			正解	手法 1	手法 2	手法 3
16:50:31	A	@B @C 先ほどのデモを受けて、私と xxx から仕様に関する質問がございます。業務状況画面の「最終操作からの経過時間」が、チケットステータスが完了の場合「00:00:00」になっている (算出アルゴリズム上正しい) がこれを「–:–:–」にしたほうがいいか?	1	1	1	1
16:55:05	A	xxx のメッセージ背景色のローテーションアルゴリズム改修のため、DB を一度リセットします。 問題がある方がいましたらご連絡ください	2	2	1	2
17:17:41	B	@A @C 質問、改善案ありがとうございます。以下回答します。–:–:–のほうが混乱を招きにくそうなので「–:–:–」に修正いただけますか？	1	2	2	2
17:23:34	C	@B 修正・デプロイしたので、確認いただけますか	1	2	2	2
17:54:59	B	@C すごく良くなりました。ありがとうございます！	1	2	3	2

形式を超えた活用を容易とするための統合ナレッジ管理の実現を目指し，実現に向けた一機能としてメッセンジャサービスでなされた会話ログから一連の話題に関する会話スレッドを自動抽出する手法を検討した．実際にソフトウェア開発においてなされた会話スレッドに対し，発言者，発言時刻に着目した抽出手法を適用した結果，それぞれの手法には一長一短があるものの，同一話題に関して発言時刻の間隔が開くケースより，単一の発言には必ずしも発言先が含まれてケースが多かったため，全体的に見ると発言時刻に着目した手法の方が高精度に会話スレッドを抽出できること，また両手法とも途中に新たな話題に関する発言が割り込んできたケースには対応できないことが確認された．さらに抽出手法の活用例として，ソフトウェアの仕様変更に関するドキュメントと抽出された会話スレッドを含まれている単語を基に関連付けることで，ドキュメントとそこに記載された仕様変更の決定経緯を含む会話スレッドを関連付けることができ，本手法はナレッジの形式を超え，いわゆる know-why の理解を支援できる可能性があることを示した．

今後の課題として，本手法で対応できなかった新たな話題の発言が割り込んできた際にそれを検出する手法の検討や，ドキュメントと会話スレッドを単語だけでなく doc2vec [12] 等の手法により意味に基づいて関連付ける手法の提案及び評価が挙げられる．

参考文献

[1] Nonaka, I. and Konno, N.: The concept of “Ba”: Building a foundation for knowledge creation. California management review, Vol.40, No.3, pp.40–54(1998). ISO 690
[2] Google LLC: G Suite available from ⟨https://gsuite.google.com⟩ (browsed at 2018.12.19).
[3] Apache software fundation: Apache Solr, available from ⟨http://lucene.apache.org/solr/⟩ (browsed at 2018.12.19).
[4] Microsoft: Yammer available from ⟨https://products.office.com/ja-jp/yammer/yammer-overview/⟩ (browsed at 2018.12.19).
[5] Slack Technologies: Slack, available from ⟨https://slack.com/⟩(browsed at 2018.12.19).
[6] Slack Technologies: Slack Analytics, available from ⟨https://slack.com/apps/category/At0G5YTKU2-analytics/⟩(browsed at 2018.12.19).
[7] 野村直之: ナレッジマネジメントツールの配備, 実践動向と次世代技術, 人工知能学会誌, Vol.16, No.1, pp.33–41 (2001).

表 6　仕様変更内容を記述したドキュメント例
Table 6　Example of document with the specification

タイトル	本文
マークがついた時のユーザ側への通知	以下の 2 種類を実装 1. 一番最初にマークがついた時　「△さんが、受付番号*** の対応を開始しました。未対応: 0、対応中:1、完了: 0」 2. マーク状態に変化があった時で、1 以外のとき「△さんが、受付番号*** の対応を進めています。未対応: 0、対応中:0、完了: 1」

表 7　仕様の決定経緯を含む会話スレッドの例
Table 7　Example of extracted thread with decision process

発言時刻	発言者	本文
11:14:37	A	@B 処理者の処理状況の表示について相談があります． 先ほどの打ち合わせでの話とちょっと矛盾しますが，例えば 5 枚の画像を送信した際，その状態変化があるたびにユーザへ 「受付番号 xx 未着手: 4，対応中: 1，完了: 0」といった通知を送るような機能は可能ですか?
11:20:39	B	@A 可能です。
11:21:15	A	@B ありがとうございます．実は営業担当よりつい先ほど最新版のメッセージ案が共有され，ぜひそれを表示したいということになっていたので・・・
11:27:10	B	@A 承知しました。 現状、マークの変化があるたびにメッセージを送信するようになっているので、これで意図されている動作になっているか 見ていただきたいのですが、可能でしょうか?

[8] 山本修一郎, 神戸雅一: CMC が拓く知識流通ネットワーク. 人工知能学会誌, Vol.25, No.5, pp.715–725 (2010).
[9] 佐別当隆志, 小谷美佳: エンタープライズソーシャルネットワークを活用したナレッジマネジメント, 情報の科学と技術, Vol.62, No.7, pp.296–301 (2012).
[10] Meixner, B., Lee, M., and Carter, S.: Chat2Doc: From Chats to How-to Instructions, FAQ, and Reports. *Proc. the 2017 ACM Workshop on Multimedia-based Educational and Knowledge Technologies for Personalized and Social Online Training* pp. 27-30, ACM (2017).
[11] Mikolov, T., Chen, K., Corrado, G., and Dean, J.: Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781 (2013)
[12] Le, Q., and Mikolov, T.: Distributed representations of sentences and documents. *Proc. International Conference on Machine Learning*, pp.1188–1196 (2014).