最先端ソフトウェア工学ゼミ[個別ゼミ2] 成果報告

2021年12月9日

塚田 祥弘 (キヤノン株式会社)
関 堅吾 (株式会社NTTデータ)

1. 設定したテーマとその理由

- テーマ：機械学習を用いた、要求仕様書からの設計ドキュメントの自動生成に関する研究の調査
- 設定した理由
  - 本ゼミ参加者は、いずれも業務で上流工程のレビュアを務めることがあり、設計ドキュメントの品質改善に関心があるため
    - お互いの現場のレビューで重要視されるのは

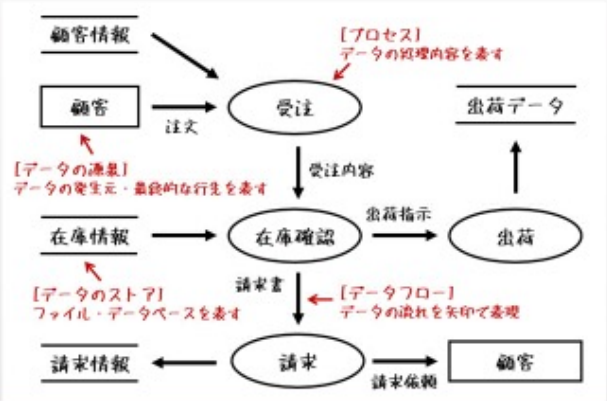
      データ関連の設計図（DFD; DataFlowDiagram, DLD; DataLineageDiagram）

      である。
  - 要求仕様から設計ドキュメントを執筆する際に有識者が用いる暗黙知を、機械学習を用いてシステム化することで、以下の効果が期待できると考えた
    - 誤った解釈・要件見落し防止による、成果物の品質向上
    - ドキュメント生成の自動化による、執筆者の負担軽減

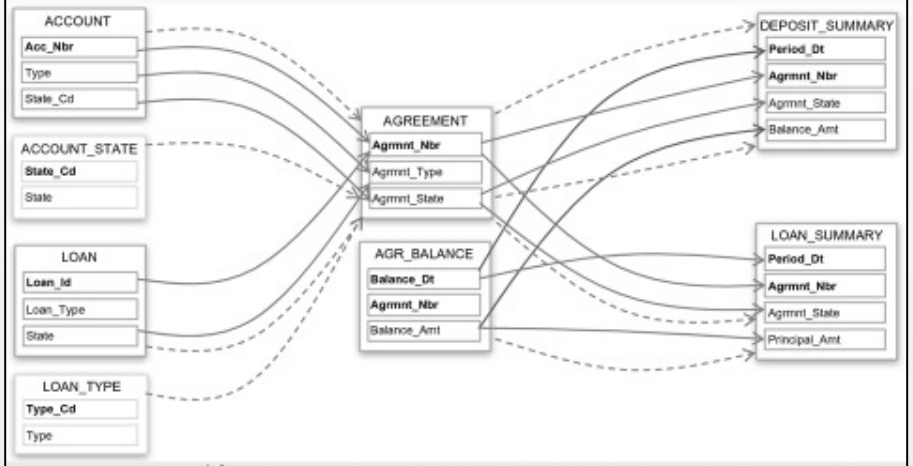
自動生成対象の図面はDFD

DFDが対象の設計書の場合

- 要求仕様書から、データの源泉、プロセス、データのストア、データフローを特定する必要がある。



参考：Data Lineageを可視化した図



出典：Tomingas K., et al. (2019) Computing Data Lineage and Business Semantics for Data Warehouse.
https://doi.org/10.1007/978-3-319-99701-8_5

個別ゼミ2開始までのまとめ

Top SE
EDUCATION PROGRAM FOR TOP SOFTWARE ENGINEERS
6

上期まとめ

- 本ゼミでは、機械学習を用いて、要求仕様書から設計ドキュメント(特にデータの流れに着目したもの)を自動生成する研究について、ここ4年ほどの論文の調査を行った。
- 自然言語で記述された要求仕様書から設計ドキュメントを自動生成する先行研究としては、ユースケース図・フィーチャーモデル・クラス図などを対象としたものがあり、それらの手法を適用することで、**DFDの構成要素の抽出**と**要素の種類の識別**が、ある程度の精度でできそうなことが判明した。
- 論文の一つに、要求仕様書からフィーチャモデルを生成するツールを公開しているものがあったため試してみたが、日本語に未対応・コードの品質が悪い・同一のエンティティがモデルの異なる場所に複数回現れる、といった問題があり、そのまま使うのは難しいことがわかった。
- 個別ゼミ2では、今回学んだ手法を活用することで、日本語で記述された要求仕様書からDFDを自動生成するツールの開発に取り組みたい。

Top SE
EDUCATION PROGRAM FOR TOP SOFTWARE ENGINEERS
7

DFDの自動生成に関する報告

- DFDを用いた要求仕様の洗練化のためのトレーサビリティ構築と条件の抜け漏れ検出
  - 情報処理学 会. 第181 回ソフトウェア工学研究会(SIGSE), 2013
  - 和田 大輝 弓倉 陽介 鷲見 毅 藤本 宏 村田 由香里
- データフローダイアグラムを利用した 要求仕様の品質向上技術
  - 東芝レビューVol.68 No.11 (2013)
  - 村田 由香里 弓倉 陽介 和田 大輝

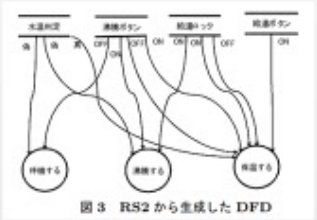
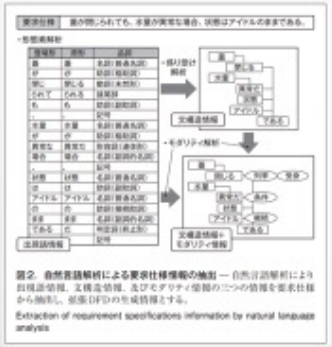


図 3 RS2 から生成した DFD



要求仕様書に対して形態素解析・係り受け解析を行うことでDFDを生成。

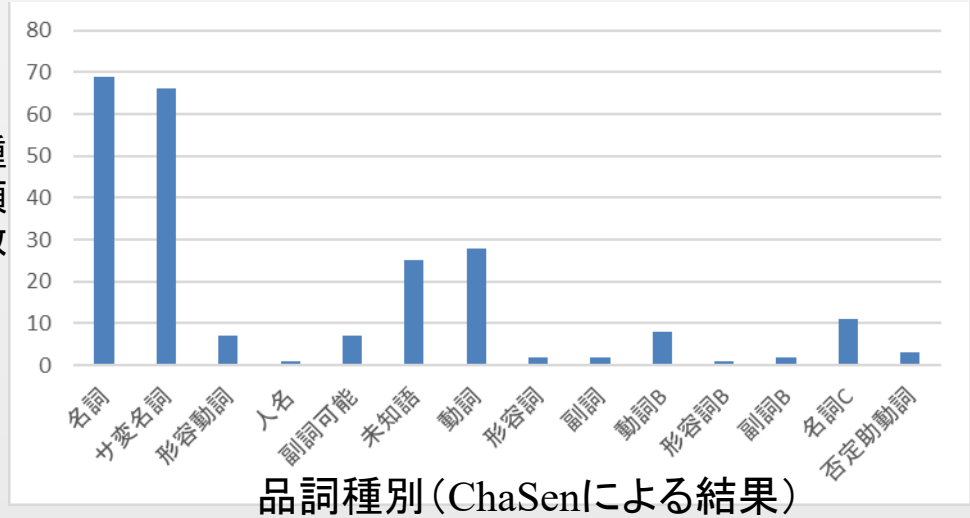
上期の調査により、
　DFDの構成要素それぞれは抽出可能であることが判明。
その後の調査により、
　係り受け解析によりDFD自動生成を目指す報告もあった。

ゼミ2の全体概要

- 目的
  - 誤った解釈・要件見落し防止による、成果物の品質向上
    - 課題：人が作業することによる誤った解釈・要件見落し。
  - ドキュメント生成の自動化による、執筆者の負担軽減
    - 課題：作業工数がかかる。
- 解決手段
  - 要求仕様書からDFDの自動生成
- DFD自動生成の実施計画
  - 10/7-10/28　スプリント1
    - 要求仕様書の内容を全て反映したDFD作成の検討
      - 文章のルールパターンすべてに対応する手法では、様々なケースで課題解決できない。
      - 様々なケースが発生しないようにDFD作成工程を分解し、自動化対象を絞り込むべき。
  - 11/4-11/25　スプリント2
    - 抽象度の高いDFD作成の検討
      - 課題解決：ターミネータを取りこぼさない、設計初期工数の削減。

DFD自動生成対象

- 話題沸騰ポット要求仕様書（GOMA-1015型）第6版
  - 全18ページ
  - 約3600文字
  - 総単語数 232



品詞種別（ChaSenによる結果）


話題沸騰ポット（GOMA-1015型）
要求仕様書

胡麻印まほうびん（株）

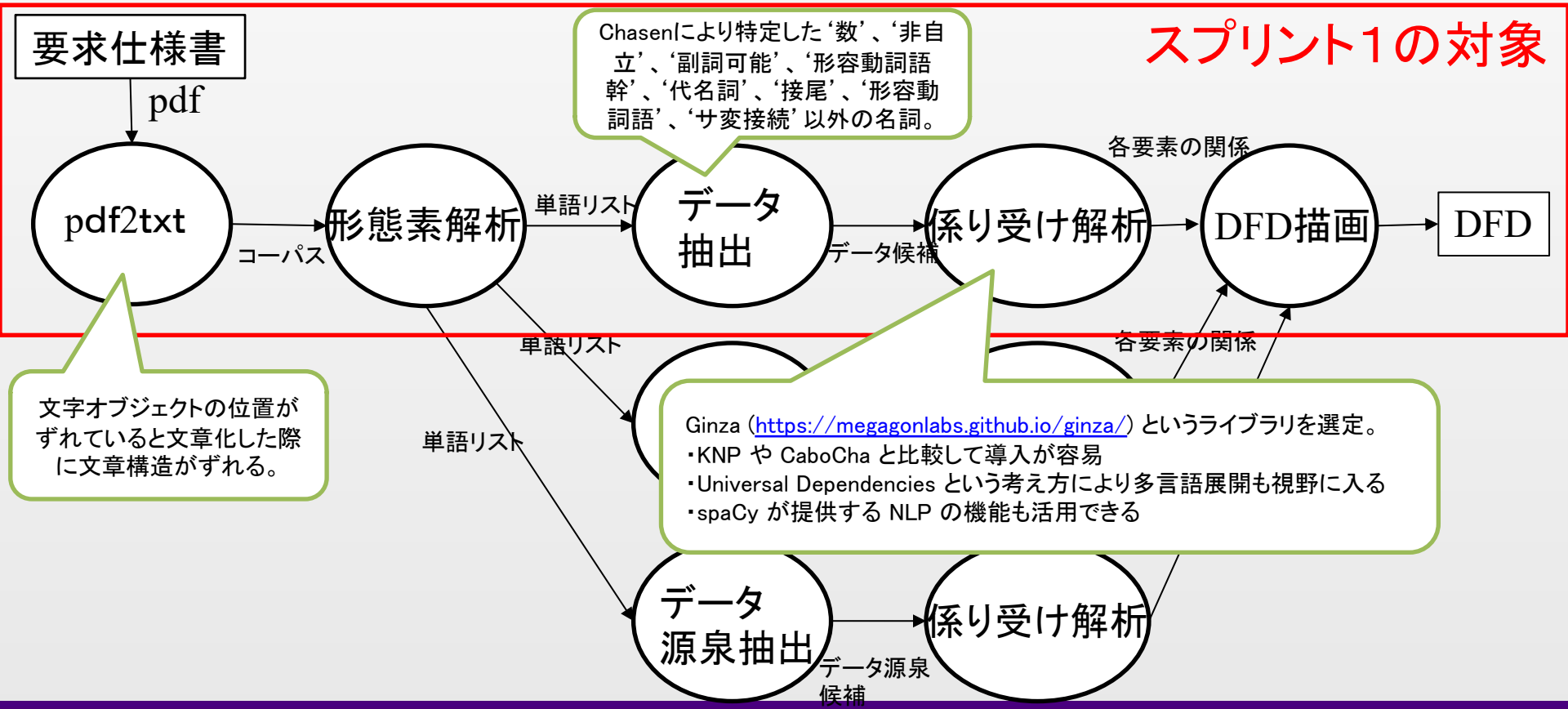
話題沸騰ポット（GOMA-1015 型）要求仕様書第6 版, 組込みソフトウェア管理者・技術者育成研究会,2005.

ゼミ2の全体概要

- 目的
  - 誤った解釈・要件見落し防止による、成果物の品質向上
    - 課題：人が作業することによる誤った解釈・要件見落し。
  - ドキュメント生成の自動化による、執筆者の負担軽減
    - 課題：作業工数がかかる。
- 解決手段
  - 要求仕様書からDFDの自動生成
- DFD自動生成の実施計画
  - 10/7-10/28　スプリント1
    - 要求仕様書の内容を全て反映したDFD作成の検討
      - 文章のルールパターンすべてに対応する手法では、様々なケースで課題解決できない。
      - 様々なケースが発生しないようにDFD作成工程を分解し、自動化対象を絞り込むべき。
  - 11/4-11/25　スプリント2
    - 抽象度の高いDFD作成の検討
      - 課題解決：ターミネータを取りこぼさない、設計初期工数の削減。

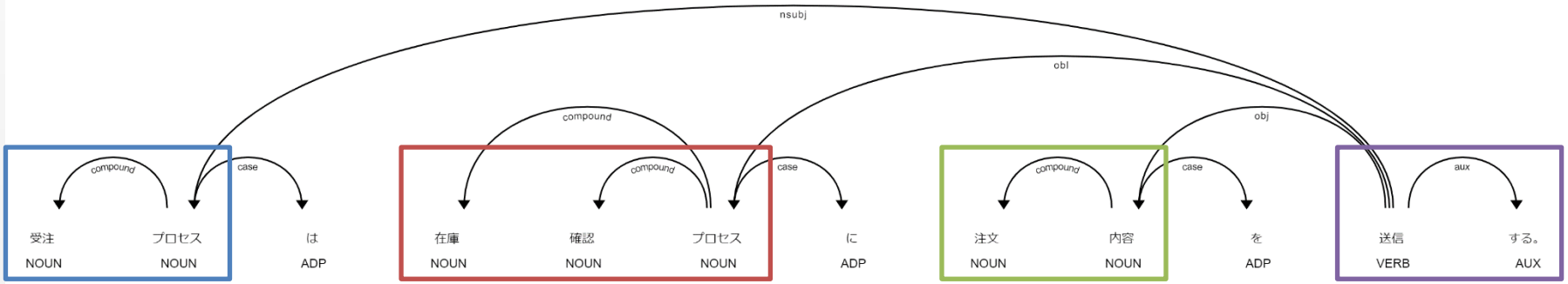
スプリント1
要求仕様書の内容を全て反映したDFD作成の検討

- 要求仕様書からDFD生成までのデータフロー。
- プロセス間はテキストファイル出力とし、事前にフォーマットを確定した
  - 非同期での作業を可能にするため。



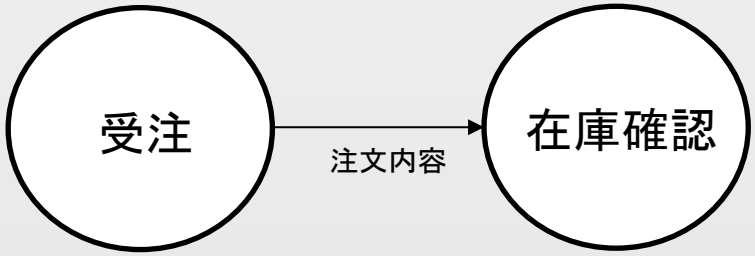
スプリント1:基本的なアイデアと直面した課題

「受注プロセスは、在庫確認プロセスに注文内容を送信する」という文を例に取ると・・



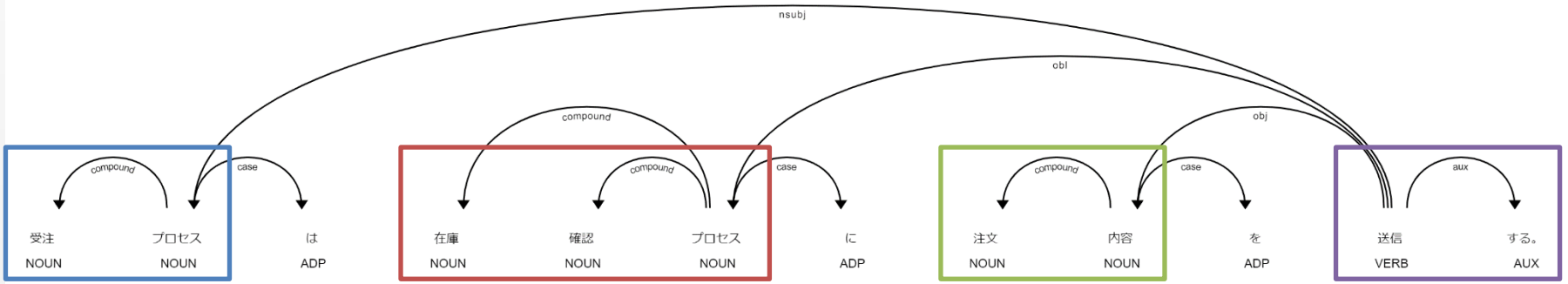
②それに係っている主語・目的語・斜格を特定すれば、
主語・斜格(ニ格の場合)はデータの受け渡し元と先、
目的語は受け渡すデータを表現しているのではないか。

①まず係り受け解析の
root要素となる
動詞を特定し、

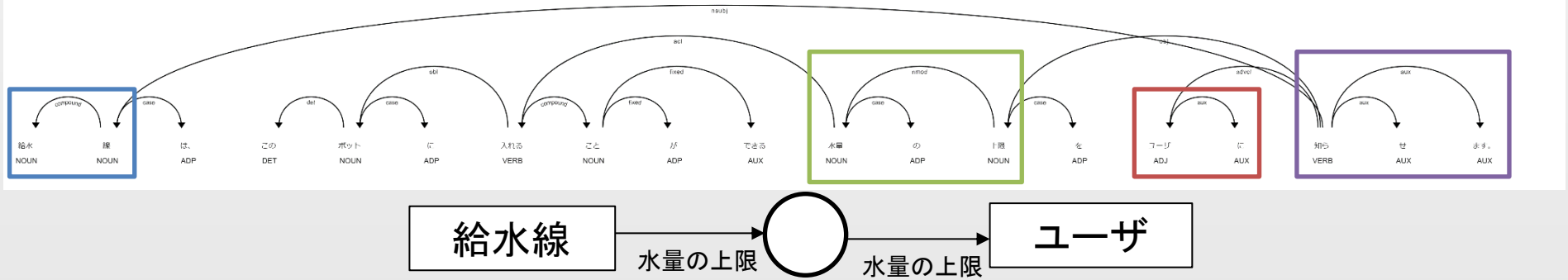


スプリント1:基本的なアイデアと直面した課題

「受注プロセスは、在庫確認プロセスに注文内容を送信する」という文に倣うと、

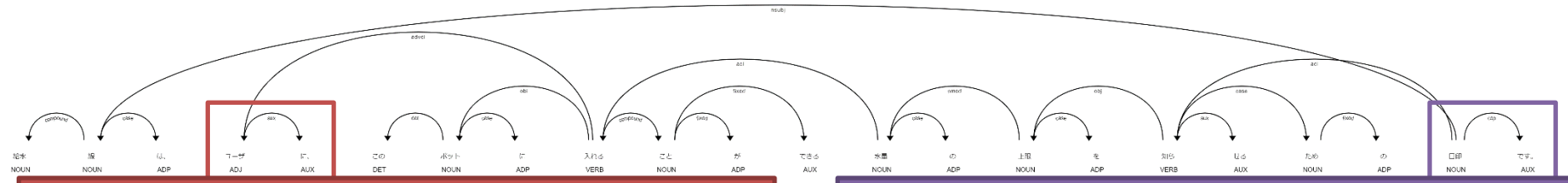


「給水線は、このポットに入れることができる水量の上限をユーザに知らせます」という文なら、確かに同じ構造になるので正しく解釈できそうに思える。



スプリント1:基本的なアイデアと直面した課題

しかし、実際の要求仕様書に書かれている文は
「給水線は、ユーザに、このポットに入れることができる水量の上限を知らせるための目印です。」
この文を係り受け解析して先述のルールを当てはめても、期待したようには解釈できない。



「ユーザに」は、本来は「知らせる」に係ってほしいのに、「入れる」に係るものと判断されてしまっている

文のroot要素が「目印です」になるため、直接これに係っている目的語や斜格が存在しなくなってしまう

加えて、以下のような問題も存在することを認識。

- 格助詞の表層格だけに着目しても、データの受け渡し元／先かどうか正しく判断することはできない

  例)『蓋センサは(中略)蓋が閉じている時にonになります。』→「名詞(句)」+「二格」だが、着点ではなく時間を表している

- 主語や目的語が省略、もしくは代名詞化される場合がある。また説明が一文で完結せず、複数の文に跨る場合がある

  例)『システム全体として、以下の動作仕様を満たさなければなりません。(中略)ユーザが設定したタイマのタイムアウト時、及び沸騰状態終了時には、ブザーを3回鳴らします。』

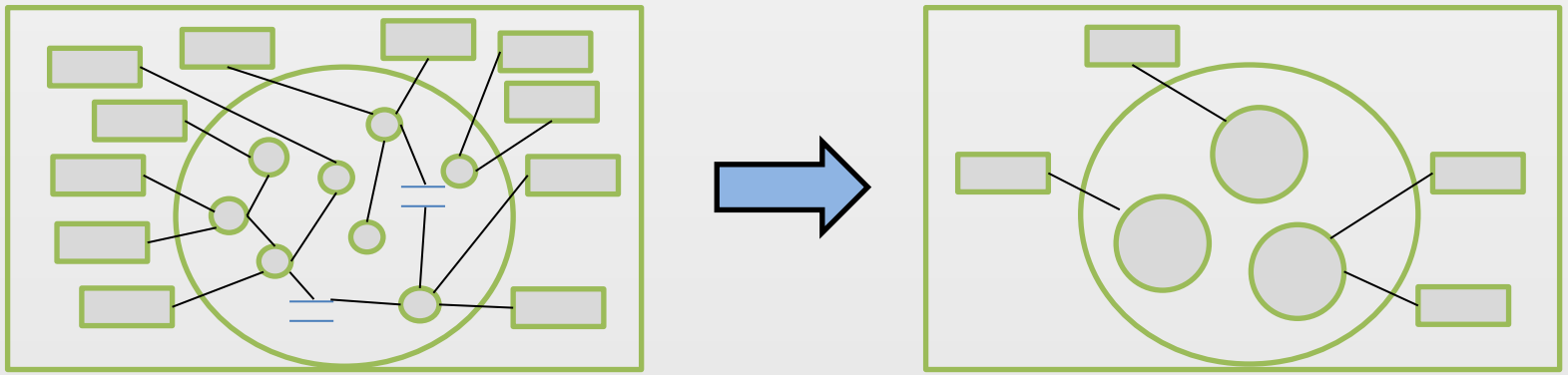
  →後半の主語は「システム全体」(もしくは別の箇所で述べられているキッチンタイマ機能)だが、一つの文に閉じた処理ではそのことはわからない

→これらの問題に、その都度ルールを追加し続けて対応していくのは難しいと判断

スプリント1振り返り

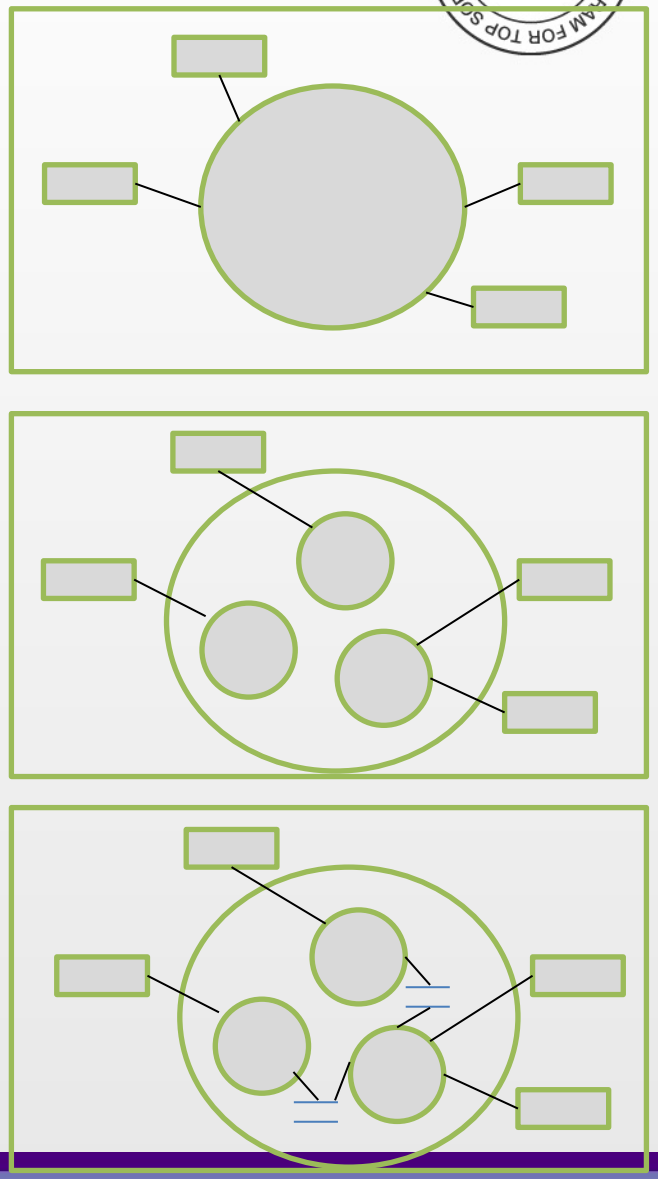
- 文章のルールパターンすべてに対応する手法では、様々なケースで課題解決できない。
- 改善点
  - 様々なケースが発生しないようにDFD作成工程を分解し、自動化対象を絞り込むべき。
    - 全要素の関連性は、期待する文法パターンで記述されていないと抽出は難しい。
    - 正しい文法で記載されていれば、確かに本手法で対応できるはずだが、
      対象の要求仕様書が正しい文法で記載されている前提では、実現場での使用は難しい。
    - 既存の文章を、全て正規化する難しさ、文章を記載する人（顧客や非技術系人材）の教育の難しさ。
    - プロセスと、データの源泉間の関係に絞るなどすれば、対処は可能。

抽出対象を限定する。



DFD作成工程(トップダウン法)

- ステップ1
  - データの源泉、ターミネータを洗い出し
- ステップ2
  - 抽象度の高いプロセス定義
  - プロセスとデータ源泉/ターミネータ等を結ぶ関連線
- ステップ3
  - データストアの定義
- プロセスを段階的に詳細化していく。



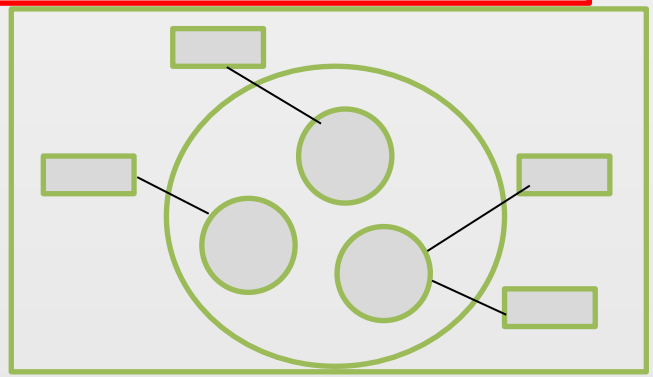
ゼミ2の全体概要

- 目的
  - 誤った解釈・要件見落し防止による、成果物の品質向上
    - 課題：人が作業することによる誤った解釈・要件見落し。
  - ドキュメント生成の自動化による、執筆者の負担軽減
    - 課題：作業工数がかかる。
- 解決手段
  - 要求仕様書からDFDの自動生成
- DFD自動生成の実施計画
  - 10/7-10/28　スプリント1
    - 要求仕様書の内容を全て反映したDFD作成の検討
      - 文章のルールパターンすべてに対応する手法では、様々なケースで課題解決できない。
      - 様々なケースが発生しないようにDFD作成工程を分解し、自動化対象を絞り込むべき。
  - 11/4-11/25　スプリント2
    - 抽象度の高いDFD作成の検討
      - 課題解決：ターミネータを取りこぼさない、設計初期工数の削減。

スプリント2の設計：抽象度の高いDFD作成の検討

スプリント2の対象

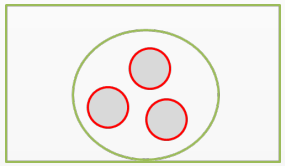
- ステップ1
  - データの源泉、ターミネータを洗い出し
    スプリント1の成果物を利用し復号語の名詞抽出で対応
- ステップ2
  - 抽象度の高いプロセス定義
    - 話題沸騰ポットの場合は、仕様書に明記されていた。
    - プロセス候補のクラスタリングで対応できそうだが、今回は割愛する。
  - プロセスとデータ源泉/ターミネータ等を結ぶ関連線
    - 関連性を抽出する処理を検討する必要あり。
- ステップ3
  - データストアの定義
- ステップ1‘
  - プロセスのブレイクダウン



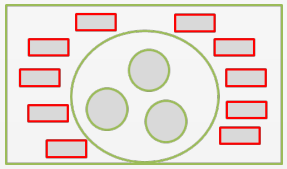
スプリント2の設計

:人間が実施
:自然言語処理で実施

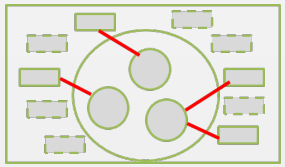
①



②



③



①要求仕様書の主要プロセスを特定

②形態素解析・品詞タグ付けを行い、
名詞・名詞句をDFD中の要素の候補として抽出

③名詞の場合
・係り受け解析により文中の主語特定
・主語に対応する動詞・目的語と①プロセス名との類似度(Word2Vecを使用)から関連度を判定

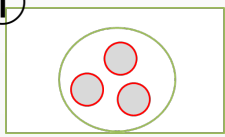
③名詞句の場合
・①プロセスと関連があるかを、同一の文中にプロセス名と共起しているかどうかに基づき判定

④抽出されたDFD中の要素とプロセスとの関連を可視化

スプリント2の設計①②

- 要求仕様書の1章に記載されている以下の3機能をDFD上の主要なプロセスとし、これらに関連する「データの源泉」や「データストア」を特定することを目標とする。
  - 沸騰・保温
  - 給湯
  - タイマ

①



1. 機能要件

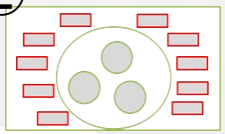
今回設計する沸騰ポットとは、ユーザに以下の機能を提供する家電製品です。

- ポット内の水を沸騰・保温する機能
- ポット内の水を給湯する機能
- ユーザが指定した時間がきたら、ブザーを鳴らして知らせるキッチンタイマ機能

以降の章では、このポットに要求される機能の詳細を説明します。

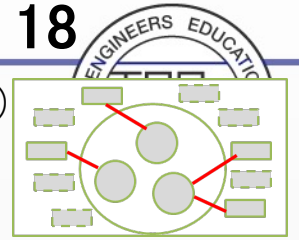
- 「データの源泉」や「データストア」の候補として、要求仕様書の2・3章から、名詞・名詞句を抽出。それぞれ後述するルールに従い、関連するプロセスを判定。

②



- 蓋センサ
  蓋が開いているかどうかを検出します。蓋が閉じている時に on になります。
- サーミスタ
  ポット内の水温を検出します。
- ヒータ
  ポット内の水を加熱します。
- ヒータ用電源
  ヒータへ電力を供給します。通常は on で、ヒータに異常が発生した時に off にして電力を遮断します。
- 給水線
  ユーザに、このポットに入れることができる水量の上限を知らせるための目印です。満水センサの位置よりも若干下にあります。

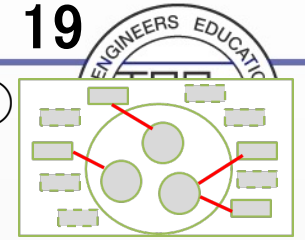
- タイマボタン
  このボタンを押すとタイマが起動し、１回押す毎に１分追加されます。
- タイマ残り時間表示窓
  タイムアップまでの残り時間（分単位に切り上げ）が表示されます。
- 保温設定ボタン
  このボタンを押すと、保温モードを高温（98℃保温）、節約（90℃保温）、ミルク（60℃保温）モードに設定します。1 回押す毎に高温→節約→ミルク→高温とモードが変わります。
- 温度／モード表示窓
  現在の水温と、設定されている保温モード（図中の▼）が表示されます。
- 解除ボタン
  給湯口のロック／解除を行います。ロック中は、給湯ボタンを押しても水は出ません。ロック中に押すとロックは解除され、解除されている時に押すと給湯口をロックします。また、給湯中はロックできません。



ステップ2の設計(続き)③名詞

抽出した名詞は、以下の方法でどのプロセスに関連しているか(あるいはどのプロセスにも関連していないか)を判定した。

- 課題1:明らかなストップワード(例:「以下」「次」「時」「下」…)を除去しても、単体ではDFD中の要素になり得るかどうか判断が難しいもの(例:「外観」「意味」「通常」「異常」「位置」…)が多数抽出されてしまう。
- 工夫1:文中で重要な役割を果たしている名詞を特定するため、係り受け解析を行い、文中で主語となっているものに対象を限定。
- 課題2:単純なキーワードマッチでは、関連するプロセスを判断できない場合がある。
  - たとえば、「ヒータは水を沸騰させます」という記述があれば、単純なキーワードマッチで沸騰プロセスと関連があることを判断できるが、実際には「ヒータは水を加熱します」「ヒータは水を温めます」「ヒータは湯を沸かします」…などと、多様な表現が可能
- 工夫2:表現の揺らぎを吸収するため、係り受け解析を行って主語に対応する動詞及び目的語を特定し、それらの語が各プロセスの名称(「沸騰」、「保温」、「給湯」、「タイマ」のいずれか)と一定以上類似していれば、そのプロセスに関連していると判定。
  - 類似度の評価には下記ページで公開されているWord2Vecの学習済み日本語モデルを用い、閾値は0.5以上とした。https://aial.shiroyagi.co.jp/2017/02/japanese-word2vec-model-builder/



ステップ2の設計(続き)③名詞句

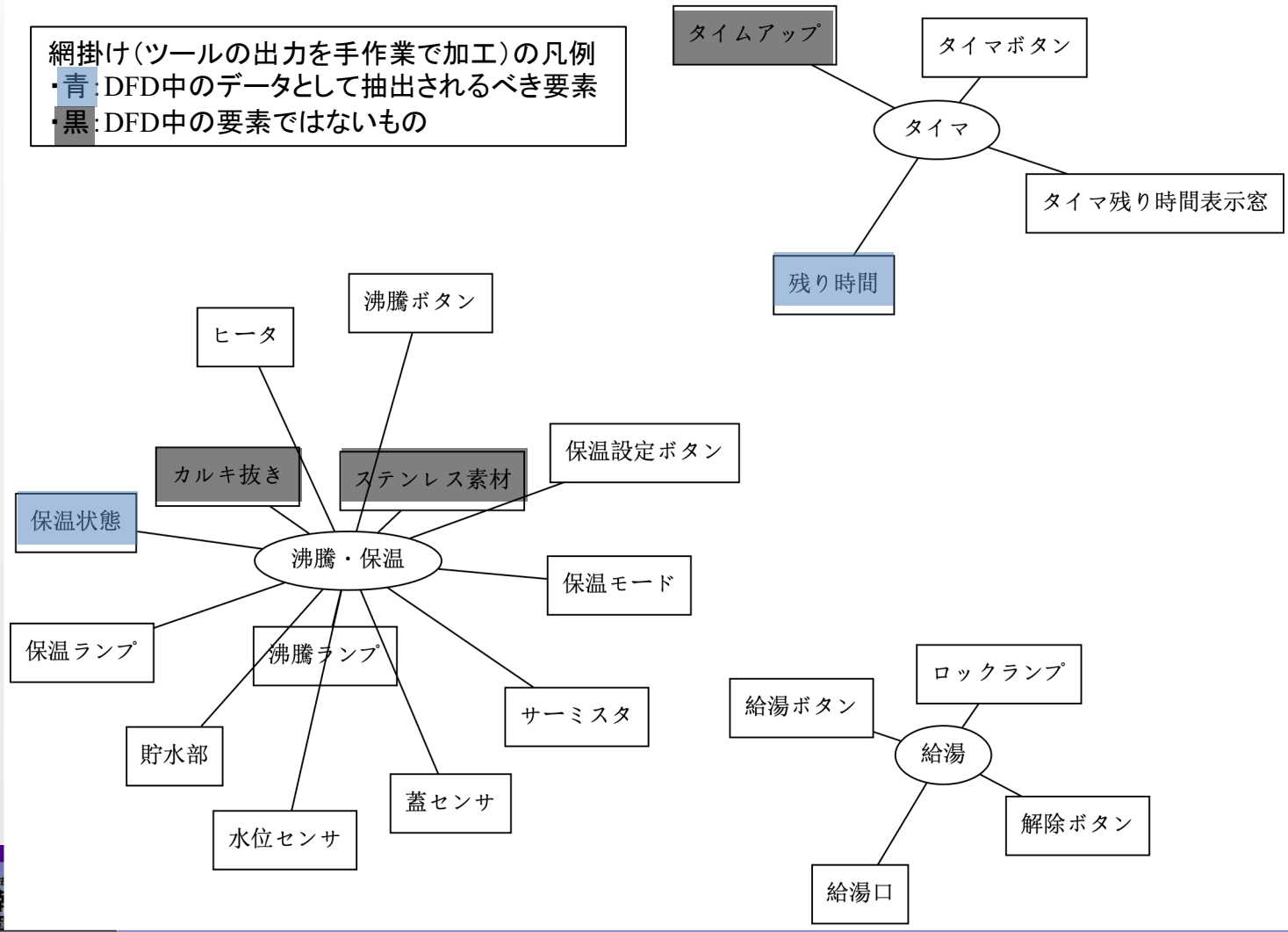
名詞句については、仕様書内のいずれかの文中でプロセス名(「沸騰」、「保温」、「給湯」、「タイマ」)と共起していればそのプロセスと関連しているとみなしたが、以下の工夫を行った。

- 課題3:説明が複数の文に分かれている場合、それらの文同士の関係を考慮できていない。
  - たとえば、「蓋センサは、蓋が開いているかどうかを検出します。」「蓋が閉じられた場合は、温度制御可能な水位ならば沸騰状態に移行し、ポット内の水を加熱します。」という2つの文があった場合、蓋センサは沸騰プロセスと関連があるが、それぞれの文を独立に分析していたのでは見出すことができない。

- 工夫3:名詞句の末尾の名詞に同じものが複数存在した場合(今回は「センサ」「ボタン」「ランプ」が該当)、語尾より前の部分の名詞(句)でもプロセス名との共起判定を行う。
  - システムや機器に(「センサ」「ボタン」「ランプ」といった)同じ種類の要素が複数存在する場合、名詞句の前方部分がそれらを識別するための情報であり、役割を端的に表した言葉として重要な役割を担っていると考えた。

出力結果

数件の誤検出はあるものの、「データの源泉」「データストア」の抽出と、プロセスとの関連付けが、概ね正しく実現できていることがわかる。

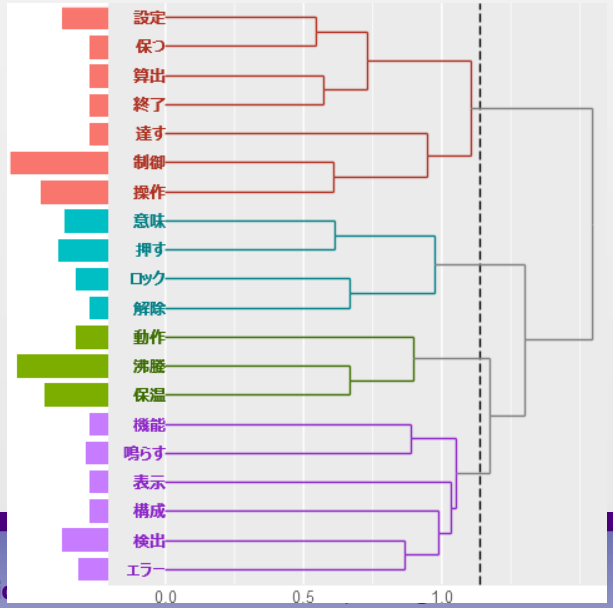


本手法による課題に対しての解決度

- 我々の目的に対してどうか
  - 誤った解釈・要件見落し防止による、成果物の品質向上
    - **文章に存在するすべての要素を抽出可能。**
    - 必要ない要素も含まれる可能性がある。
    - 日本語特有の省略をどこまで許容できるかの検証が必要。
  - ドキュメント生成の自動化による、執筆者の負担軽減
    - **DFD作成時の、各要素作成、プロセス一各要素間接合の手間軽減。**
    - 段階的細分化は現状では未対応。
- DFD作図の際の初期コスト軽減の補助ツールとして現場導入可能。

考察・展望

- ツールの展望
  - 今回は抽象度の高いプロセスが要求仕様書に明記されていたので、それをよりどころにDFD自動生成をすすめた。
    - 本来であれば、要求仕様書から抽象度の高いプロセスも導けると対応可能なケースが増えて良い。
      - 試しに、動詞になりえる単語に対してクラスタリングしてみた。
      - Jaccard係数、Ward法でクラスタリング。
        - 給湯関連、沸騰・保温関連、鳴らす関連、それ以外でクラス分類できた。
        - たまたまかもしれないが、追加検討の余地あり。
  - 抽出結果（ステンレス素材など）を取り除いたり、（現状では恣意的な）閾値を調整したりといったフィードバックをかけることで対応可能。



考察・展望

- 日本語の難しさは他の仕様書解析のケースでも活かせそう。
  - 主語や目的語が省略、もしくは代名詞化される場合がある
  - 文の構造が複雑化すると、誤った品詞タグが付与される場合がある
  - 説明が一文で完結せず、複数の文に跨って記述される場合がある
  - 同じ内容を多種多様な表現で記述できるので、どういった品詞の組合せが揃っていればよいかを大量に定義する必要がある

- 設計図の自動生成の論文は多く見たが、実用化に至るものはなかった。
  - 今回、設計図作成の工程分解し、対象を絞ることで、現場にとって意味のある結果を出すことができた。
  - 上期に調査した論文で対応されていたFeatureツリー、ドメインクラス図、ユースケース図なども同様に作成工程を分析し現場導入。

上期まとめ

- 本ゼミでは、機械学習を用いて、要求仕様書から設計ドキュメント（特にデータの流れに着目したもの）を自動生成する研究について、ここ4年ほどの論文の調査を行った。

- 自然言語で記述された要求仕様書から設計ドキュメントを自動生成する先行研究としては、ユースケース図・フィーチャーモデル・クラス図などを対象としたものがあり、それらの手法を適用することで、**DFDの構成要素の抽出**と**要素の種類の識別**が、ある程度の精度でできそうなことが判明した。

- 論文の一つに、要求仕様書からフィーチャモデルを生成するツールを公開しているものがあったため試してみたが、日本語に未対応・コードの品質が悪い・同一のエンティティがモデルの異なる場所に複数回現れる、といった問題があり、そのまま使うのは難しいことがわかった。

- 個別ゼミ2では、今回学んだ手法を活用することで、日本語で記述された要求仕様書からDFDを自動生成するツールの開発に取り組みたい。

DFDの自動生成に関する報告

- DFDを用いた要求仕様の洗練化のための トレーサビリティ構築と条件の抜け漏れ検出
  - 情報処理学 会, 第 181 回ソフトウェア工学研究会 (SIGSE), 2013
  - 和田 大輝　弓倉 陽介　鷲見 毅　藤本 宏　村田 由香里
- データフローダイアグラムを利用した 要求仕様の品質向上技術
  - 東芝レビューVol.68 No.11(2013)
  - 村田 由香里　弓倉 陽介　和田 大輝

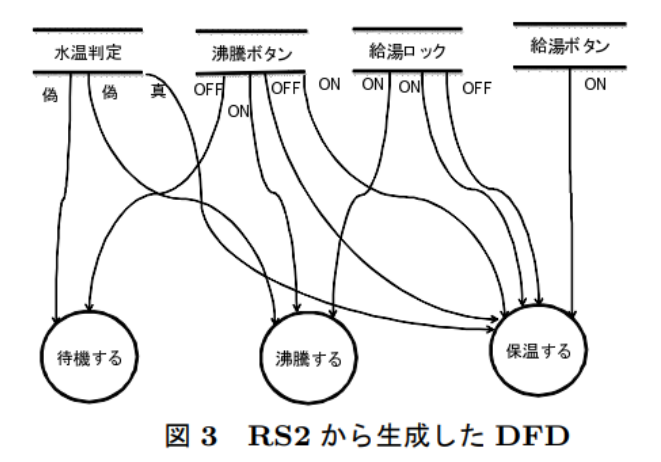


図 3　RS2 から生成した DFD

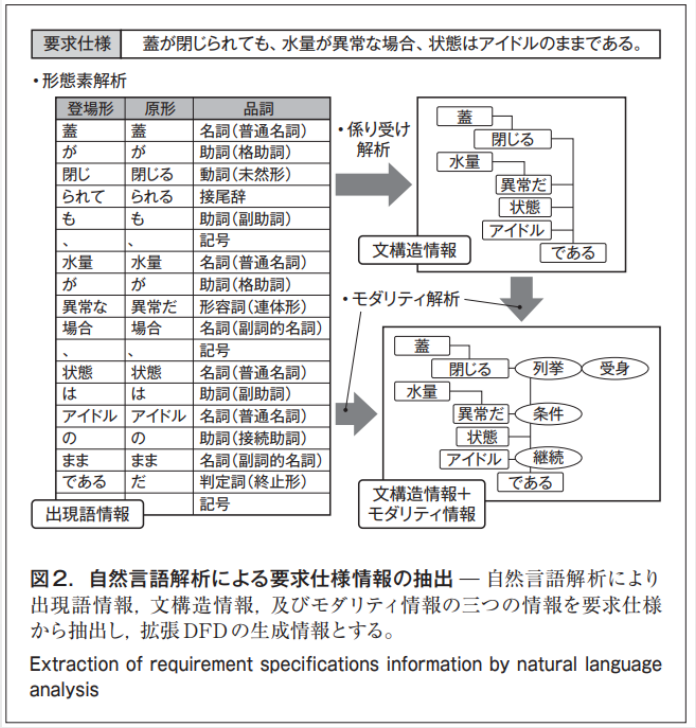


図2. 自然言語解析による要求仕様情報の抽出 — 自然言語解析により出現語情報，文構造情報，及びモダリティ情報の三つの情報を要求仕様から抽出し，拡張DFDの生成情報とする。

Extraction of requirement specifications information by natural language analysis

要求仕様書に対して形態素解析・係り受け解析を行うことでDFDを生成。

2. 調査方法

説明　1分

- Scopus から抽出した, “requirement engineering” と “machine learning” の両方に関連する, 2017年以降の国際論文のリスト250件から調査。

- 対象論文のabstract を分析し、要求仕様からの設計図生成に関連しそうな23論文に絞り込んだ。
  - 量が多い・対象分野に詳しくないため調査方法を工夫した。
    - クラスター分析で的を絞った。(20クラスに分割)
      - Noun, Proper Noun, Verb のみ対象。
      - Ward法, 距離：Cosine, 値：TF-IDF
    - 引用、被引用論文検索。

3. 調査結果

説明　0.5分

調査の結果、
「要求仕様からデータ関連の設計図（DFD,DLD）を生成できる論文」
は発見できなかった。

ただし、他の設計図を生成する論文はいくつか見られたので、関連しそうな3パターンの論文を共有します。

関連設計図	対象論文
UC図	Identifying Use Case Elements from Textual Specification: A Preliminary Study
FeatureModel	Extracting Software Product Line Feature Models from Natural Language Specifications
クラス図	Towards Queryable and Traceable Domain Models

3－1．ユースケースの抽出に関する研究

説明　2分

- Identifying Use Case Elements from Textual Specification: A Preliminary Study
（Tiwari S., Rathore S.S., Sagar S., Mirani Y.）

- テキストで書かれた要求仕様書からユースケースの要素を特定する予備研究。

研究内容

学習データ作成：
サンプルシステムのデータセット28種に対して手動タグ付けで、UC名、アクター名、その他の3カテゴリに分類。

学習データを元に、UC要素の推論：
右図の手順のように、抽出したワードに対して、学習済みデータを用いて予測。

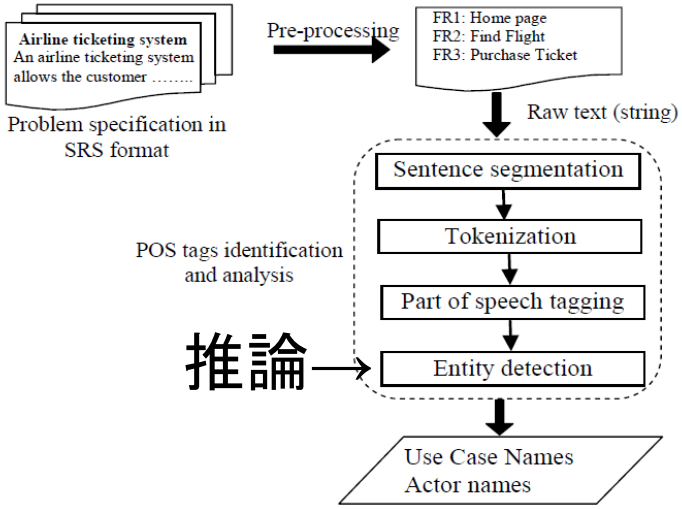
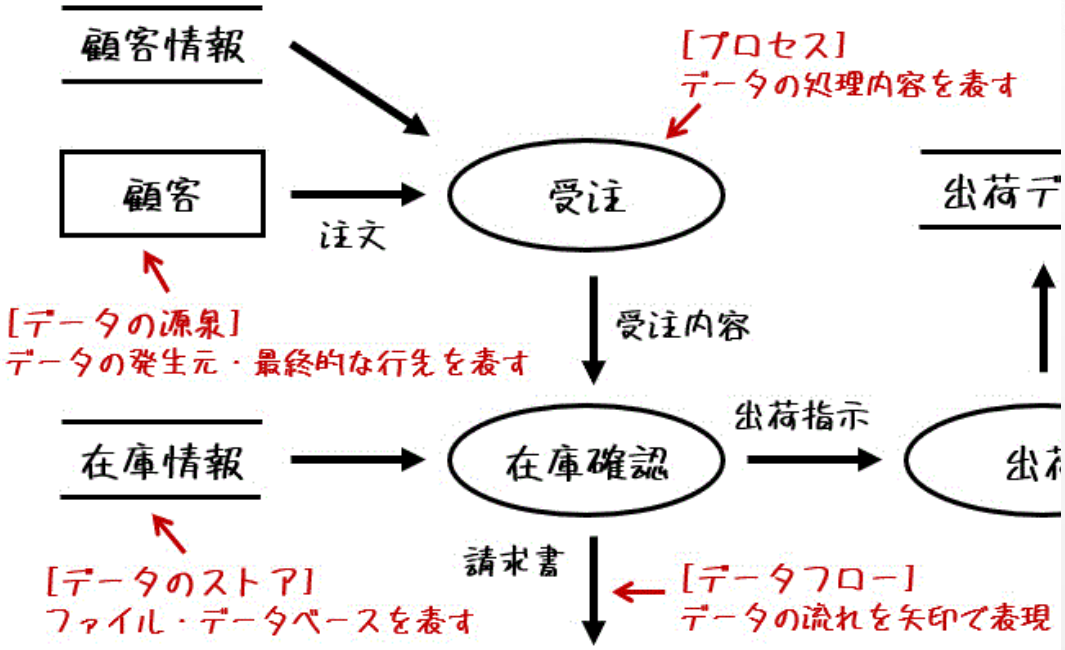


Fig. 1. Overview of the approach

- 結論
  - 基本、代替、例外フロー抽出は本アプローチの拡張では不可。
  - アクター名予測は精度、再現率、Fスコア高い。
    - 精度0.91 再現率1.00 F値0.92
  - UC名予測は精度、再現率、Fスコア低い。

Class labels	Multinomial Naïve Bayes			Perceptron			Linear classifier with SGD			Passive Aggressive classifier		
	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
UC-NAME	0.34	0.57	0.43	0.56	0.22	0.32	0.54	0.18	0.27	0.38	0.50	0.43
P-ACTOR	0.70	1.00	0.82	0.91	0.86	0.88	0.91	0.87	0.89	0.86	1.00	0.92

所感

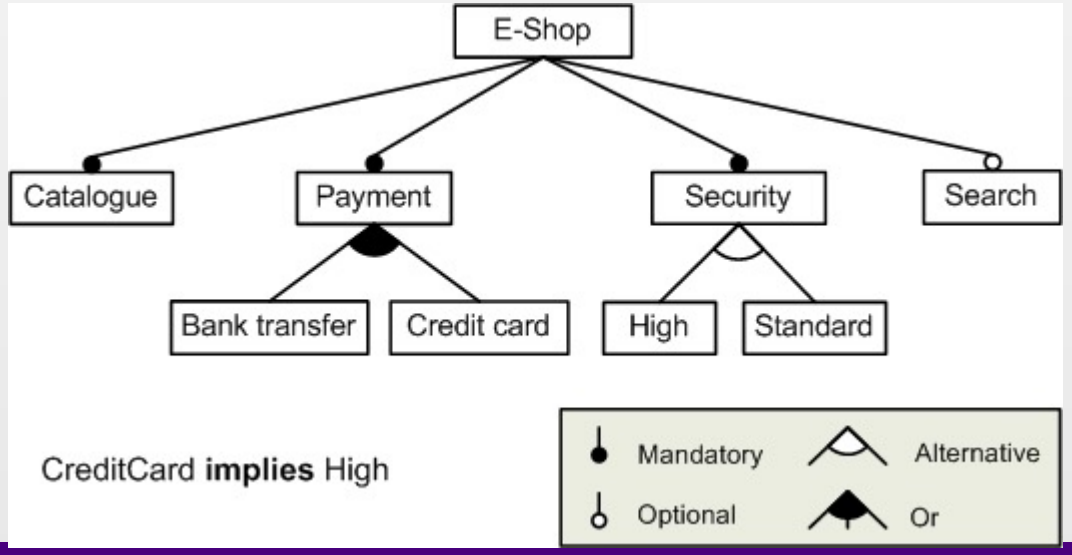


- データ系設計図への活用案
  - 本手法を活用して抽出したアクター名はDFDのデータの源泉に使用できそう。
    - 例：顧客は注文する。この場合、アクター名は、顧客となり、「データの源泉」にあたる。
  - プロセスなど他のDFD要素の特定はできない。
  - 学習データはどのように用意すべきか。
    - 本論文の結果は英語のデータセットで学習した結果である。
    - 同様の方法をとる場合は、改めて日本語での教師データを用意してラベリングする必要がある。

3－2. フィーチャーモデルの抽出に関[illegible]（１）

説明 2分

- Extracting Software Product Line Feature Models from Natural Language Specifications (A Sree-Kumar, et al., 2018)
- https://doi.org/10.1145/3233027.3233029

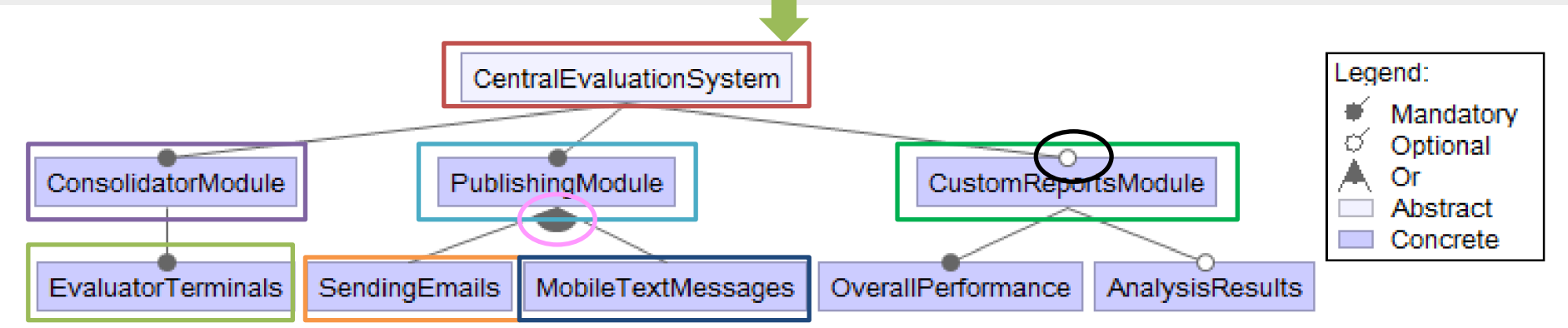


フィーチャモデルの例
(https://en.wikipedia.org/wiki/Feature_model)

研究の目的

- 自然言語で記述されたSPL仕様書からフィーチャモデルを抽出

*"The **central evaluation system** collects all the evaluation results from the various **evaluator terminals** using the **consolidator module** and enters them into the results database where the scores for various courses are segregated or grouped based on candidates. These results will be used by the **publishing module** which is responsible for **sending emails** or **mobile text messages** to the candidates with their complete score cards. The publishing module can be configured using a **custom reports module** for generating customized views of the reports which can be used for understanding the **overall performance** of the class and provide informational **analysis results** on the exam to the teachers and other administrators."*



先行研究の課題

- 先行研究は非公開のツールを用いてなされており結果を再現できない
- 本研究では、一般に利用可能な以下のツールを組み合わせて、フィーチャモデルを抽出・構築可能なフレームワークである FeatureX を提案 (https://github.com/5Quintessential/FeatureX)
  - NLTK (https://www.nltk.org/)
  - Pattern (https://github.com/clips/pattern)
  - WordNet (https://wordnet.princeton.edu/)
  - PDFMiner (https://github.com/euske/pdfminer)

FeatureXの処理フロー

FeatureXの処理の流れは以下の通り。

1. 字句解析でトークン化や品詞のタグ付けを行い、名詞句を抽出してモデル中のフィーチャの候補とする
2. 全フィーチャ候補の TF-IDF を計算して、もっともスコアが高かったものを最上位のフィーチャとする
3. 文章中で主語・目的語として出てくる順序に従い、フィーチャモデルのツリーを構成していく
4. あらかじめ定めた品詞の並びが見つかった場合は、それに応じてモデルに情報（必須・任意・排他など）を付与したり、ツリー構造を修正したりしていく。以下のような要素をもとに判断。
   - 助動詞: can, could, may, might, must, shall, should, will, would
   - 副詞: often, never, always, frequently, normally, usually, generally, regularly, occasionally, sometimes, hardly, rarely
   - 限定詞: all, every, each, any, some
   - if-then 構文 など

結果

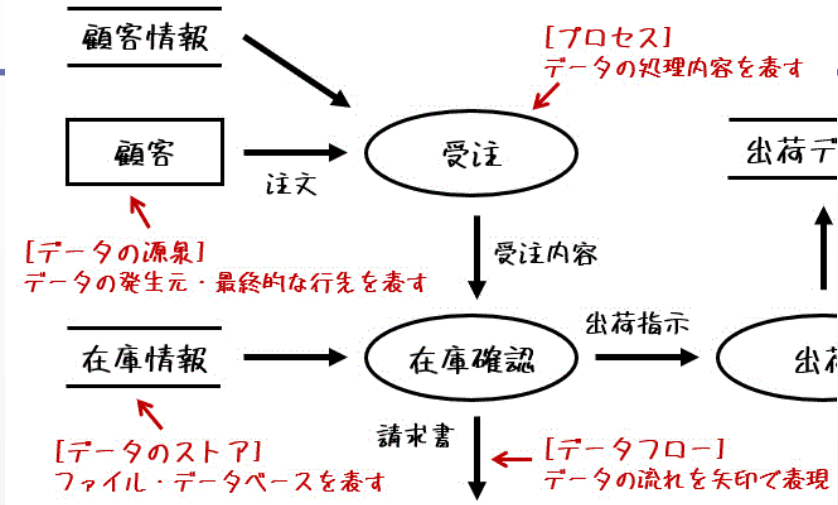
- 先行研究と比較して、フィーチャおよびフィーチャ間関係のrecallが改善し、precisionも同程度（と著者らは主張）
- 先行研究と異なり、本研究のツールはPythonで実装されており、広く一般に利用可能

Table 8: Comparing feature and relationship extraction between FeatureX and state-of-the-art approaches.

	FeatureX								Other tools							
	Features				Relationships				Features				Relationships			
Case study	Rel	Extr	**Precision**	**Recall**	Rel	Extr	**Precision**	**Recall**	Rel	Extr	**Precision**	**Recall**	Rel	Extr	**Precision**	**Recall**
CS(1a)	15	32	0.44	0.82	14	31	0.45	0.75	-	-	-	-	-	-	-	-
CS(1b)	20	27	0.73	0.77	23	26	0.87	0.76	-	-	-	-	-	-	-	-
CS(2a)	13	32	0.41	0.88	12	31	0.41	0.58	-	-	-	-	-	-	-	-
CS(2b)	15	26	0.59	0.65	16	25	0.65	0.68	7	13	0.53	0.55	6	12	0.50	0.37
CS(2c)	6	16	0.40	0.57	8	15	0.54	0.48	13	24	0.54	0.45	12	23	0.57	0.43
CS(2d)	13	29	0.46	0.93	17	28	0.63	0.52	16	27	0.59	0.57	15	26	0.66	0.55

Rel: Number of extracted features considered relevant by a domain engineer. **Extr**: Total number of extracted features (both relevant and irrelevant).

所感

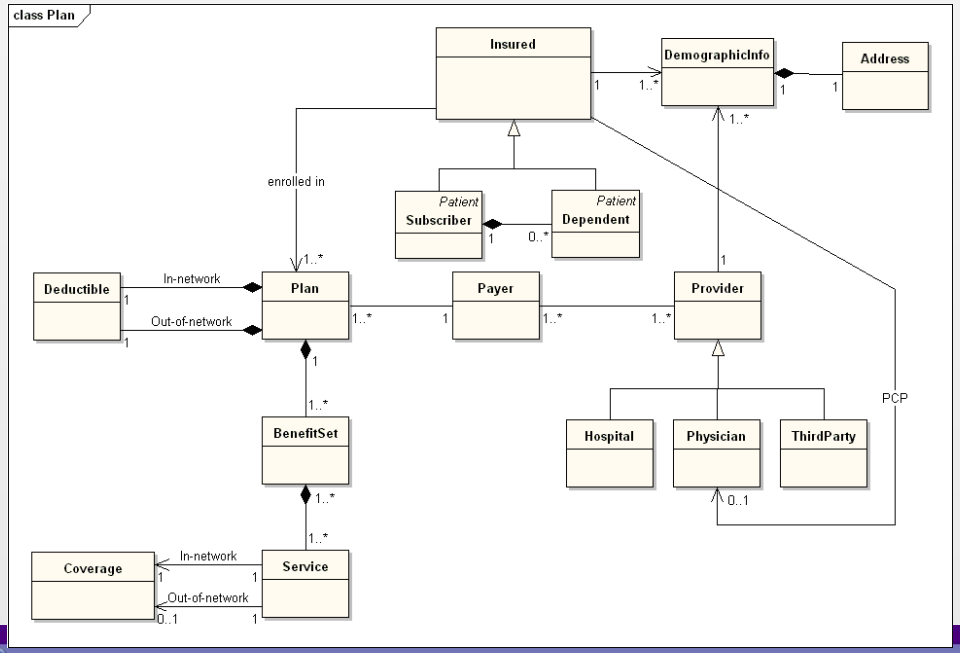


- 文章中の主語を抽出することで、(システム内の処理の主体である)プロセスの候補を特定できそう
  - 例:「**受注システム**は顧客から注文を受け取り、顧客情報DBと突合して受注内容を在庫システムに送信する」
  - 書き方によってはデータの源泉も主語になりそうなので、識別する手段が必要(例:「**顧客**は受注システムに注文を送信する」)→3-3で議論
- 同様に目的語を抽出することで、データフローの候補を特定できそう
  - 例:「受注システムは顧客から**注文**を受け取り、顧客情報DBと突合して**受注内容**を在庫システムに送信する」
  - 書き方によってはデータストアも目的語になりそうなので、識別する手段が必要(例:「受注システムは**顧客情報DB**を参照し、顧客が存在することを確認」)→3-3で議論
- 抽出された候補が単なる一般名詞か、システム中の固有名詞かは、名詞句かそうでないかと、TF-IDF によるスコアリングである程度判別できそう
  - 例:「受注(システム|プロセス)」「顧客情報データベース」

3－3. クラス図の抽出に関する研究（1）

説明　2分

- Towards Queryable and Traceable Domain Models (R. Saini, et al., 2020)
- https://ieeexplore.ieee.org/document/9218176



クラス図の例
(https://ja.wikipedia.org/wiki/%E3%83%89%E3%83%A1%E3%82%A4%E3%83%B3%E3%83%A2%E3%83%87%E3%83%AB)

概要

- ドメインの問題を記述した文書から、ドメインモデルを自動で抽出するためのフレームワークを提案
- NLPベースの既存研究に、MLを導入することで精度を改善

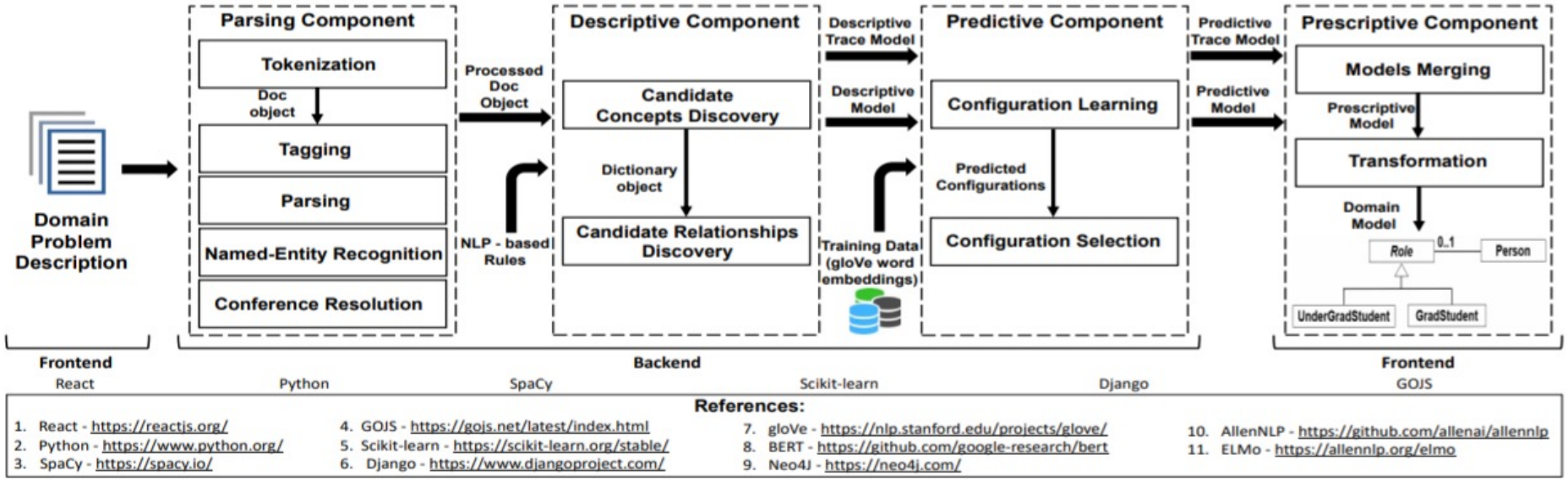
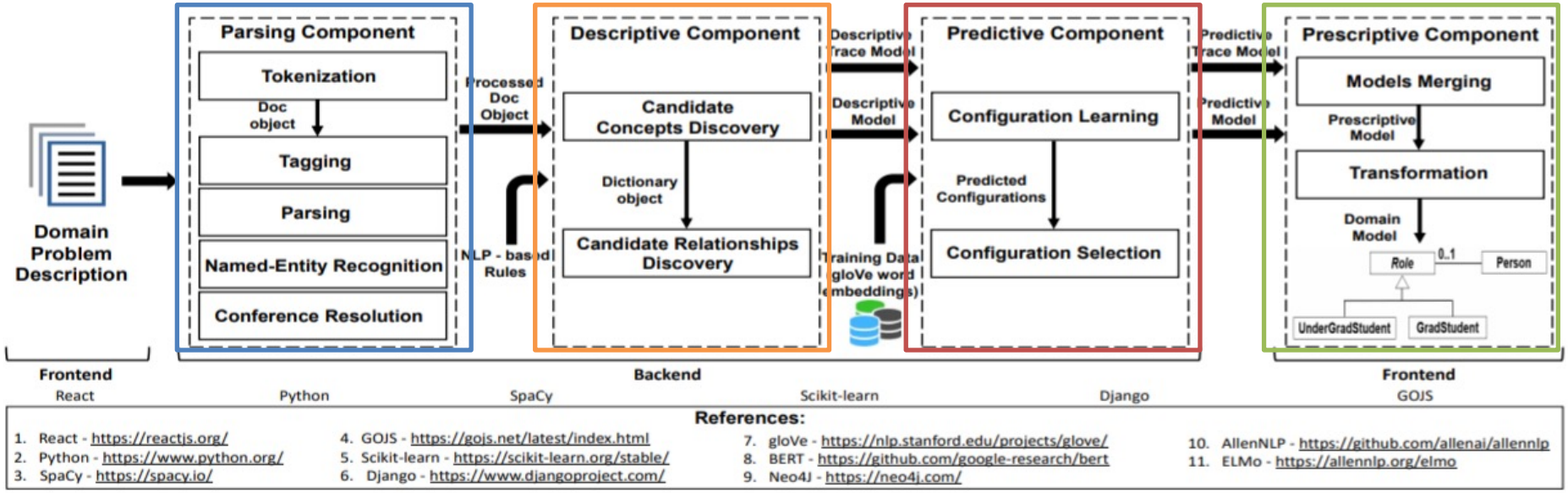


Fig. 1. Our Approach for Queryable and Traceable Domain Models

処理フロー

1. トークン化、品詞タグ付けなどの字句解析
2. 名詞句を抽出することで、ドメイン内のエンティティ候補を発見。また主語—動詞—目的語の関係から、エンティティ間の関連の候補を発見。
3. それぞれのエンティティをクラスと属性のどちらとしてモデル化するか、後者の場合データ型は何にするかを、機械学習を用いて判定
4. 3までで構築したモデルをクラス図として可視化し出力



References:

1. React - https://reactjs.org/
2. Python - https://www.python.org/
3. SpaCy - https://spacy.io/
4. GOJS - https://gojs.net/latest/index.html
5. Scikit-learn - https://scikit-learn.org/stable/
6. Django - https://www.djangoproject.com/
7. gloVe - https://nlp.stanford.edu/projects/glove/
8. BERT - https://github.com/google-research/bert
9. Neo4J - https://neo4j.com/
10. AllenNLP - https://github.com/allenai/allennlp
11. ELMo - https://allennlp.org/elmo

Fig. 1. Our Approach for Queryable and Traceable Domain Models

結果：先行研究との比較

- Arora et al. (2016) をベースラインとして比較すると、すべての課題について正答率が向上した
- 特に、MLを組み合わせることで属性推定の精度が大幅に改善した

Fig. 5. Evaluation of Extracted Domain Models (3 Case Studies)

所感

- 名詞句をエンティティとして抽出したり、主語ー動詞ー目的語の組合せからエンティティ間の関係を見出す手法は、3-2の研究と同様。
- 抽出したエンティティの種類（クラス or 属性）を判定するのに ML を使って精度を上げているのが特徴。
  - 3-2の所感で、「品詞に基づく判断だけではプロセスとデータの源泉、データフローとデータストアの識別が難しいかもしれない」と述べたが、この課題にもMLが使えるのではないか。
  - 先に挙げた例で言えば、「注文を『受け取る』」「受注内容を『送信する』」なら、目的語が指すものはデータフローだが、「顧客情報DBを『参照する』」ならデータストアである。
  - 人間は後続する動詞に基づいてこういった判断を行っているが、類語は多数存在し、それらに対してルールを逐一記述するのは現実的でないため、MLで例を与えて学習させるのが適していそうに思える。

3. 調査結果(まとめ)

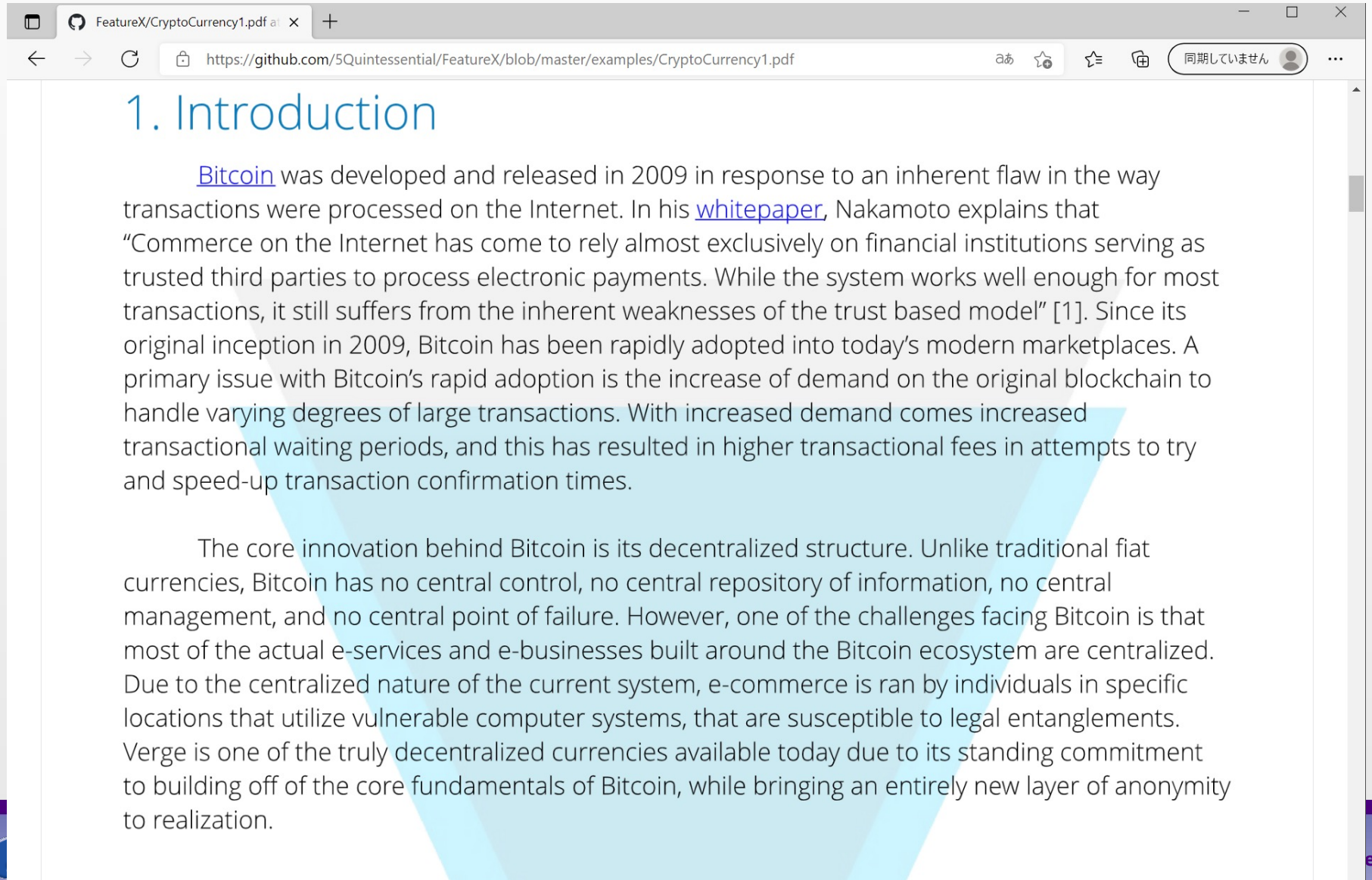
説明　1分

- 調査結果のまとめ
  - DFDの各要素を特定できそうな手段を見つけることができた。
    - データの源泉(3-1)
    - プロセス(3-2)
    - データストア(3-2,3-3)
    - データフロー(3-2,3-3)
  - FeatureX(3-2)の仕組みで、DFDの要素は概ね抽出可能であるため、FeatureXを参考にDFD抽出手法を確立する。
    - Step1:FeatureXを用いたフィーチャモデル抽出の実践
    - Step2:FeatureXをベースに3-1,3-3の手法を取り込む。(こちらはゼミ2)

4. FeatureXを用いたフィーチャモデル抽出の実践

- 目的
  - 3-2で紹介したFeatureXを実際に動かしてみることで、自然言語処理や機械学習を体験するとともに、FeatureXが我々の目的に使えそうかどうか、そのままでは使えない場合はどういった課題が存在するかを明確にする。
- 入出力
  - 入力：英語で記述された暗号通貨の仕様書
  - 中間生成物：品詞タグ付け結果、フィーチャ候補群、最上位フィーチャ
  - 出力：ツリー状のフィーチャモデル
- 所感
  - 使用しているNLPのライブラリが日本語に対応していない
  - コードの品質も特に高いわけではなく、そのまま使うのは難しそう
    - 明らかなバグや論文の記述との乖離も存在し、そのうちいくつかは本体にフィードバックを実施 (https://github.com/5Quintessential/FeatureX/pull/2, https://github.com/5Quintessential/FeatureX/issues/3).

入力


FeatureX/CryptoCurrency1.pdf
https://github.com/5Quintessential/FeatureX/blob/master/examples/CryptoCurrency1.pdf
同期していません
1. Introduction
Bitcoin was developed and released in 2009 in response to an inherent flaw in the way transactions were processed on the Internet. In his whitepaper, Nakamoto explains that "Commerce on the Internet has come to rely almost exclusively on financial institutions serving as trusted third parties to process electronic payments. While the system works well enough for most transactions, it still suffers from the inherent weaknesses of the trust based model" [1]. Since its original inception in 2009, Bitcoin has been rapidly adopted into today's modern marketplaces. A primary issue with Bitcoin's rapid adoption is the increase of demand on the original blockchain to handle varying degrees of large transactions. With increased demand comes increased transactional waiting periods, and this has resulted in higher transactional fees in attempts to try and speed-up transaction confirmation times.
The core innovation behind Bitcoin is its decentralized structure. Unlike traditional fiat currencies, Bitcoin has no central control, no central repository of information, no central management, and no central point of failure. However, one of the challenges facing Bitcoin is that most of the actual e-services and e-businesses built around the Bitcoin ecosystem are centralized. Due to the centralized nature of the current system, e-commerce is ran by individuals in specific locations that utilize vulnerable computer systems, that are susceptible to legal entanglements. Verge is one of the truly decentralized currencies available today due to its standing commitment to building off of the core fundamentals of Bitcoin, while bringing an entirely new layer of anonymity to realization.

中間生成物

Activities Text Editor Jul 8 18:39

Open ~/repos/Featu Save

品詞タグ付け結果

	WORD	TAG	CHUNK	ROLE	ID	PNP	LEMMA
2							
3	Because	IN	PP	-	-	-	because
4	the	DT	-	-	-	-	the
5	routing	VBG	VP	-	-	-	rout
6	of	IN	PP	-	-	PNP	of
7	communication	NN	NP	SBJ	1	PNP	communication
8	is	VBZ	VP	-	1	-	be
9	partly	RB	VP ^	-	1	-	partly
10	concealed	VBN	VP ^	-	1	-	conceal
11	at	IN	PP	-	-	PNP	at
12	every	DT	NP	-	-	PNP	every
13	hop	NN	NP ^	-	-	PNP	hop
14	in	IN	PP	-	-	PNP	in
15	the	DT	NP	-	-	PNP	the
16	Tor	NNP	NP ^	-	-	PNP	tor
17	circuit	NN	NP ^	-	-	PNP	circuit

Plain Text Tab Width: 8 Ln 1, Col 1 INS

フィーチャ候補群

Open CandidateTerms. ~/repos/FeatureX/example

```
1 'routing', 'communication', 'concealed', 'hop', 'tor', 'circuit',
  'method', 'single point', 'communicating peers', 'be',
  'determined', 'network surveillance', 'source', 'destination',
  'ip integration ip', 'built', 'provide', 'hidden services',
  'people', 'host', 'servers', 'unknown locations', 'ip', 'many',
  'same benefits', 'anonymous access', 'online', 'content', 'use',
  'pp-style routing structure', 'layered', 'encryption',
  'designed', 'network', 'internet', 'see', 'figure', 'traffic',
  'contained', 'borders', 'performs', 'based', 'opposed', 's
  circuit based', 'benefit', 'route', 'congestion', 'service
  interruptions', 'manner', 'similar', 's ip', 'level',
  'reliability', 'redundancy', 'first time', 'client', 'contact',
  'query', 'distributed', 'network database', 'custom structured',
  'distributed hash table', 'dht', 'kademlia', 'algorithm [ ]',
  'done', 'find', 'other client', 'inbound tunnels', 'subsequent
  data', 'information', 'further network database lookups',
  'required', 'obfuscated', 'service', 'ipv', 'verge', 'data',
```

Plain Text Tab Width: 8 Ln 1, Col 419 INS

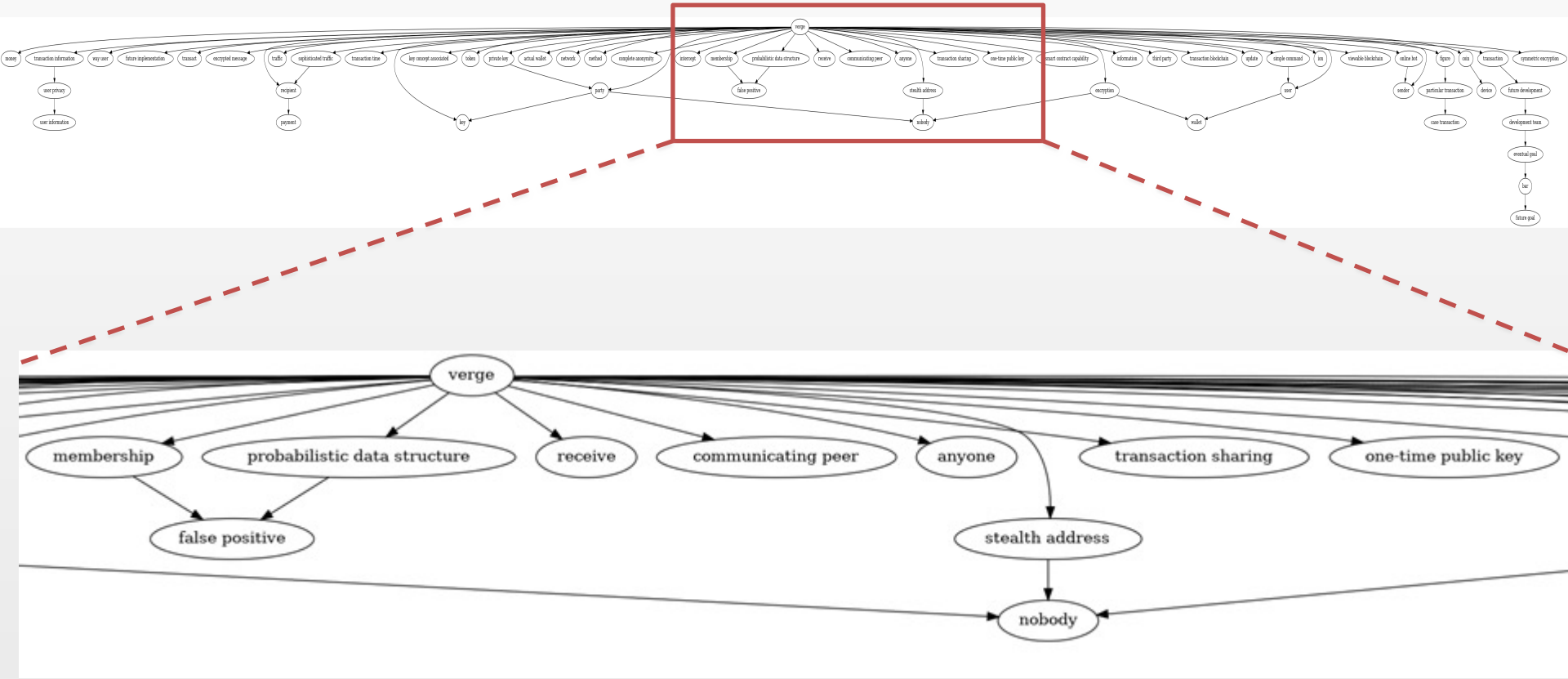
最上位フィーチャ

Open Save

```
1 Verge
```

Plain Text Tab Width: 8 Ln 1, Col 6 INS

出力



- 実際のマニュアルを対象にした際のFeatureXの実力を確認するために
  キヤノン製品Canon i-SENSYS MF742Cdwの欧州向けユーザマニュアル（英語）に対してFeatureXによる解析を実行。

対象は機体管理の章
（p.359-p456）

Outline

Managing the Machine

3S21-06X

To reduce the various risks associated with the use of this machine, such as leaks of personal information or unauthorized use by third parties, constant and effective security measures are required. An administrator should manage important settings, such as access privileges and security settings, to ensure that the machine is used safely.

■Configuring the Basic Management System

Setting Access Privileges(P. 361)

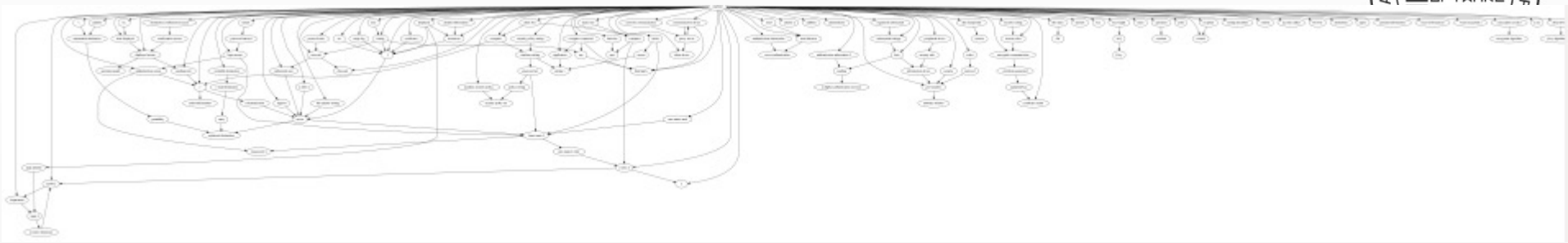
Configuring the Network Security Settings(P. 376)

■Preparing for Risks from Negligence or Misuse

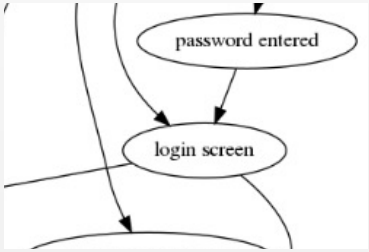
Restricting the Machine's Functions(P. 410)

Increasing the Security of Documents(P. 424)

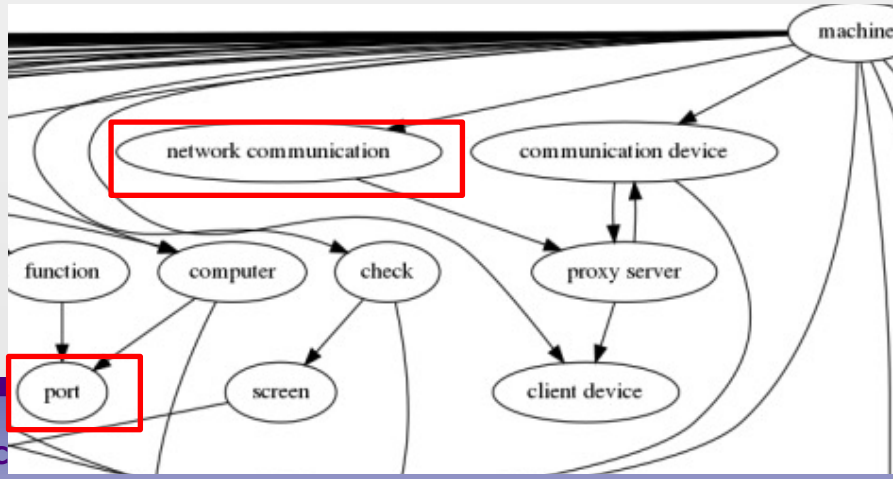
■Ensuring Effective Management



- 結果:
  - rootFeatureがmachineであることを特定。
  - DFDのデータフローとプロセスに関連するものを関連付けれていた。
    例:「password entered 」と「login screen」の関連。



- 問題点:
  - ドメインエンジニア観点では、関連を示す線が不足している。
    - ネットワークのportがNetwork communicationと関連していない。
  - 同義語などがノードとして出現するので、ツリー構造が複雑化。
- 考察:
  - ドメインのFeature関連図を反映。
    - 別のFeatureModelとの結合。
    - 他の周辺機器のマニュアルの結果も考慮する。
  - 同義語をまとめる仕組みを考える必要がある。
    - 辞書(手動で作ったリスト、MLを使って作るなど)



まとめ

- 本ゼミでは、機械学習を用いて、要求仕様書から設計ドキュメント（特にデータの流れに着目したもの）を自動生成する研究について、ここ4年ほどの論文の調査を行った。

- 自然言語で記述された要求仕様書から設計ドキュメントを自動生成する先行研究としては、ユースケース図・フィーチャーモデル・クラス図などを対象としたものがあり、それらの手法を適用することで、DFDの構成要素の抽出と要素の種類の識別が、ある程度の精度でできそうなことが判明した。

- 論文の一つに、要求仕様書からフィーチャモデルを生成するツールを公開しているものがあったため試してみたが、日本語に未対応・コードの品質が悪い・同一のエンティティがモデルの異なる場所に複数回現れる、といった問題があり、そのまま使うのは難しいことがわかった。

- 個別ゼミ2では、今回学んだ手法を活用することで、日本語で記述された要求仕様書からDFDを自動生成するツールの開発に取り組みたい。

全体設計の再設計

- ステップ１　抽象度の高いプロセスをクラス分けで定義
- ステップ２
  - データ特定
  - データの源泉、ターミネータ、抽象度の高いプロセス、それらを結ぶ関連線
- ステップ３
  - データストアの定義
- ステップ１‘
  - プロセスのブレイクダウン
  - ;