セマンティックWebコンテンツ構築支援ツールの実装と評価

2007年度

森田 武史

要旨

現在の Web コンテンツは，人間が理解することを目的として構築されているため，ソフトウェアによる理解は困難である．一方，次世代 Web の候補の一つであるセマンティック Web は，ソフトウェアが意味理解可能な辞書に基づいて，Web コンテンツにソフトウェア可読なメタデータを付与することによって，ソフトウェアが Web コンテンツを理解し，推論することを可能にしようという試みである．セマンティック Web の実現により，知的な情報検索やアプリケーションを横断したデータ統合および再利用などが可能となる．

しかしながら，ソフトウェアが Web コンテンツを理解するためのセマンティック Web 標準技術である，RDF，RDFS，OWL により記述されるセマンティック Web コンテンツは，現状では人手によって構築されており，多大なコストを要している．本論文では，その構築コストを低減するためのセマンティック Web コンテンツ構築支援ツールを実装し，評価することを目的とする．

以下，本論文の構成について述べる．

はじめに，第 1 章において，本研究の背景，問題，目的について述べる．

第 2 章では，本研究の関連技術として，セマンティック Web の技術階層，オントロジーの定義および具体例，オントロジー構築方法論，オントロジー構築支援環境について述べると共に，それらの関連研究についても紹介する．

第 3 章では，RDF(S) コンテンツ構築支援ツール MR^3 の設計，実装，評価について述べる． MR^3 は，RDF(S) コンテンツ管理機能と RDF(S) コンテンツの視覚的編集機能を持つ． MR^3 では，RDF(S) コンテンツ管理機能を「RDF コンテンツおよび RDFS コンテンツをモデルとオントロジーの関係として捉え，RDF コンテンツと RDFS コンテンツを分離し，両者の整合性を半自動的に管理する機能」として捉えている．さらに，関連ツールとの機能比較，実験を通して RDF(S) コンテンツ管理機能および RDF(S) コンテンツの視覚的編集機能を評価し， MR^3 の有用性を確認する．

第 4 章では，領域オントロジー（OWL コンテンツ）構築支援ツール DODDLE-OWL の設計，実装，評価について述べる．DODDLE-OWL は，領域専門文書を入力として，既存オントロジーを参照しながら，領域オントロジーにおける階層関係およびその他の関係の半自動構築を支援する．DODDLE-OWL を評価するために，関連研究との比較，DODDLE-OWL と手動構築の比較実験，ロケット運用分野における大規模オントロジーの構築，法律分野における既存オントロジー獲得とランキングの実験を行い，DODDLE-OWL の有用性を確認する．

最後に第 5 章では，本論文のまとめと今後の課題および展望について述べる．

Title: Implementing Semantic Web Contents Development Tools with Case Studies

Abstract:

Since current Web contents are intended for people, it is difficult for machine to understand them. On the other hand, Semantic Web enables machine to understand and even infer Web contents, giving machine-readable metadata to them. Thus intelligent search and data integration over many applications will come up with Semantic Web.

However it takes many costs for users to develop Semantic Web contents described by RDF, RDFS, and OWL by hands. So this paper focuses on the intelligent tools to support users in constructing them. This paper has been organized as follows:

Chapter 1 describes backgrounds, issues, and purpose of this study.

Chapter 2 introduces Semantic Web technologies, definitions and examples of ontology, an ontology development method, and ontology development environment as related technologies and studies around this paper.

Chapter 3 describes the design, implementation, and evaluation of the RDF(S) contents development tool called MR^3 . MR^3 has graphical facilities for RDF(S) contents and RDF(S) contents management facilities to maintain both RDF contents and RDFS contents mutually. In order to evaluate MR^3 , we compared MR^3 with other related tools. Case studies in the domain of business have shown us that MR^3 goes well.

Chapter 4 describes the design, implementation, and evaluation of the domain ontology (OWL contents) development environment called DODDLE-OWL. DODDLE-OWL makes reuse of existing ontologies and supports the semi-automatic construction of taxonomic and other relationships in domain ontologies from documents. In order to make sure how much well DODDLE-OWL goes, we compared DODDLE-OWL with popular ontology development tools. In order to evaluate the scalability of DODDLE-OWL, we constructed a large scale of ontology with 34,000 concepts or more in the field of rocket operation using DODDLE-OWL. The case studies have shown us that we have build up the large scale of the rocket operation ontology within 30 hours.

Finally, in chapter5, we have concluded this paper and pointed out future issues.

目 次

図 目 次

表 目 次

第1章　序論

1.1　背景と目的

近年，次世代 Web の候補の一つとして，セマンティック Web [1, 2] が多くの企業および研究者から注目を集めている．現在の Web は人間を情報参照の対象とし，人間が理解可能なコンテンツを提供することを目的としているため，ソフトウェアによる理解は困難である．一方，セマンティック Web は，ソフトウェアが意味理解可能な辞書に基づき，Web コンテンツにソフトウェア可読なメタデータを付与することによって，ソフトウェアが Web コンテンツの意味を理解し，推論することを可能にしようという試みである．ここで,「Web コンテンツの意味を理解する」とは，後述するセマンティック Web 技術により，Web コンテンツにどのような概念が含まれているか，また，それらの概念間にどのような関係があるかをソフトウェアにより処理できることとする．Web コンテンツの意味が理解できることで，Web コンテンツをより多くの異なる状況で利用することが可能となる．セマンティック Web の実現により，知的な情報検索，動的な Web サービス連携，様々なアプリケーションを横断したデータの統合および再利用などが可能となり，Web がさらに便利なものとなることが期待されている．

セマンティック Web は，W3C (World Wide Web Consortium) を中心として関連技術の仕様策定が行われている．URI (Uniform Resource Identifier) [3]，Unicode，XML (Extensible Markup Language) [4] といった構文上の基礎となる仕様のもと，主語，述語，目的語の三つ組みによって，Web 上のリソース間の関係を表現するためのデータモデルと構文を提供する RDF (Resource Description Framework) [5] が標準化されている．RDF により記述される RDF コンテンツは，RDFS (RDF Vocabulary Description Language: RDF Schema) [6] および OWL (Web Ontology Language) [7] により記述されるソフトウェアが意味理解可能な辞書（RDFS コンテンツおよび OWL コンテンツ）に基づいて構築される．また，RDF，RDFS，OWL を検索するための技術として SPARQL [8] も標準化されている．さらに，RIF (Rule Interchange Format), Logic, Proof, Trust が標準化されつつあり，これらの技術により，推論規則の定義，ソフトウェアによる推論，推論結果が正しいことの証明，Web コンテンツの信頼性の保証などが可能となる．本論文では，セマンティック Web 関連技術の中から，ソフトウェアが Web コンテンツを理解するために必要な技術である，RDF，RDFS，OWL に着目する．

セマンティック Web を実現するためには，上述したように様々な技術の標準化が必要となるが，RDF，RDFS，OWL により記述されるセマンティック Web コンテンツの普及が不可欠である．しかしながら，セマンティック Web コンテンツは，現状では人手によって

構築されており，多大なコストを要している．本論文では，その構築コストを低減するためのセマンティック Web コンテンツ構築支援ツールを実装し，評価することを目的とする．本論文では，セマンティック Web コンテンツの構築を支援するために二つのツールを提案する．一つ目は，RDF(S)（RDF および RDFS）コンテンツ構築支援ツールであり，二つ目は，領域オントロジー（OWL コンテンツ）構築支援ツールである．ここで，領域オントロジーとは，特定の領域（法律やビジネスなど）に存在する概念とその間の関係を定義したものである．領域オントロジーは，ソフトウェアが RDF コンテンツを理解する際に，辞書的な役割を果たす．

1.2 RDF(S) コンテンツ構築支援

RDF(S) コンテンツの構築を支援するために，様々な研究が行われている．

RDF(S) コンテンツの記述を支援するために，数多くのグラフィカルエディタが提案されている．それらの多くの主な機能は，単純に XML 記述を支援するのではなく，“主語-述語-目的語” の三つ組みにより表現可能な RDF データモデルに基づいて XML ベースの記述を視覚的に表示および編集することにある．グラフィカルエディタにより，RDF(S) コンテンツを視覚的に理解することが可能となり，RDF データモデルを XML 形式で容易に記述することが可能となる．しかしながら，RDF(S) コンテンツを RDF データモデルに忠実に視覚化を行うことには問題がある．RDF および RDFS コンテンツは，同一構文で記述されるため，忠実に RDF データモデルを視覚化すると，両者の区別を行うことが困難となる．RDFS コンテンツは，現実世界のオブジェクトからは離れて，対象の概念化を考慮して構築が行われる．一方，RDF コンテンツは現実世界のオブジェクトを考慮して構築が行われる．このように，RDF コンテンツおよび RDFS コンテンツは，それぞれ異なる観点から構築されるため，両者の区別を行うことが困難な場合には，ユーザの RDF(S) コンテンツ構築コストは高くなるという問題がある．

その他の RDF(S) コンテンツの記述を支援するための研究として，RDF コンテンツのためのモデルベースのフレームワークも提案されている（N3 [9], TRIPLE [10], Turtle [11] など）．これらのフレームワークは，XML 形式に比べて RDF(S) コンテンツの記述を容易に行うことが可能であるが，依然としてモデルベースのフレームワークであるため，RDFS コンテンツと RDF コンテンツの区別を行うことは困難である．

RDFS コンテンツを管理するために，セマンティック Web 登場以前から知識工学の分野ではオントロジー構築支援ツール（OntoEdit [12], Protégé [13] など）が開発されている．これらのツールは，主にオントロジーの構築およびオントロジーベースの意味記述の管理に注力している．そのため，OWL および RDFS コンテンツの構築支援は行われているが，RDF コンテンツの構築支援は十分に行われていない．

以上より，RDF(S) コンテンツの構築を支援するために様々な研究が行われているが，RDF コンテンツおよび RDFS コンテンツを同時に構築する際に生じる問題に焦点を絞った研究は行われていない．RDF コンテンツおよび RDFS コンテンツの間の意味的な対応を

とらえるためには，より詳細なセマンティクスが必要となる．そこで，本論文では，RDF コンテンツと RDFS コンテンツを分離し，両者の間の整合性を保ちながら視覚的に編集する機能を持つ RDF(S) コンテンツ構築支援ツール MR^3 (Meta-Model Management based on RDFs Revision Reflection) を提案する．

1.3 領域オントロジー構築支援

人間とソフトウェアの間での共通理解において，オントロジーが必要であることが認識されつつあり [14]，オントロジーは様々なアプリケーション領域で注目を集めている．セマンティック Web は，最もオントロジーの利用が進んでいるアプリケーション領域の一つであり，多くのオントロジーが OWL で記述されつつある．しかしながら，他のアプリケーション領域と同様，領域オントロジーの構築と保守には多大なコストがかかっている．さらに，日本語を対象とした，セマンティック Web における領域オントロジー構築支援環境は少ない．

テキストや汎用オントロジーなどの既存情報資源から自動的に領域オントロジーを構築するために，多くの研究は，知識工学，自然言語処理，データマイニングの技術を用いている [15, 16]．しかしながら，領域オントロジーの構造は専門家の対象領域の見方に依存して変化するため，領域オントロジーを全自動で構築することは困難である．実用的な領域オントロジー構築支援を行うためには，全自動ではなく，ユーザとのインタラクションを通して，半自動的に領域オントロジーの構築を支援することが重要である．さらに，セマンティック Web コミュニティに広く領域オントロジー構築支援環境を利用してもらい，ユーザビリティやオントロジー構築支援機能に対するフィードバックを得ることができるように，オープンソースソフトウェアとして公開することも重要である．

以上より，本論文では，インタラクティブで実用的な領域オントロジー構築支援環境 DODDLE-OWL (a Domain Ontology rapiD DeveLopment Environment - OWL extension) を提案する．DODDLE-OWL は，領域専門文書を入力として，WordNet [17] や EDR 電子化辞書 [18] などの既存オントロジーを参照しながら，領域オントロジーにおける階層関係およびその他（階層以外）の関係の半自動構築を支援する．DODDLE-OWL は，DODDLE-II [19] を基礎として，オープンソースソフトウェアとして Java 言語で実装されており，領域オントロジーを OWL 形式でエクスポートする機能や日本語領域オントロジーの構築支援を行うための機能など，様々な機能が拡張されている．

1.4 論文の構成

以降，本論文の構成は次のとおりである．第 2 章では，本研究の関連技術として，セマンティック Web の技術階層，オントロジーの定義および具体例，オントロジー構築方法論，オントロジー構築支援環境について述べると共に，それらの関連研究についても紹介する．第 3 章では，RDF(S) コンテンツ構築支援ツール MR^3 の設計，実装，評価に

ついて述べる． MR^3 の評価として，関連ツールとの機能比較，RDF(S) コンテンツ管理機能および RDF(S) コンテンツの視覚的編集機能の評価実験について述べる．第 4 章では，領域オントロジー構築支援環境 DODDLE-OWL の設計，実装，評価について述べる．DODDLE-OWL の評価として，関連研究との比較，DODDLE-OWL と手動構築の比較実験，ロケット運用分野における大規模オントロジーの構築，法律分野における既存オントロジー獲得とランキングの実験について述べる．最後に，第 5 章では，本論文のまとめと今後の課題および展望について述べる．

第2章　関連研究

2.1　概要

本章では，本論文に関連する技術および関連研究を紹介する．はじめに，2.2 節では，セマンティック Web の概要を述べ，その中から本論文と関連の深い技術である，URI，XML，RDF，RDFS，OWL について説明する．2.3 節では，オントロジーの定義を述べ，本研究において主に利用する汎用オントロジーについて説明する．2.4 節では，オントロジー構築方法論について述べる．最後に，2.5 節では，現在開発されているオントロジー構築支援環境を紹介する．

2.2　セマンティック Web

近年の知識工学における大きな流れとして，セマンティック Web [1, 2] が挙げられる．Web 上のリソースにメタデータを付与し，計算機がそれを理解して推論を行うなど，Web そのものを知識ベースとして扱えるようにすることが大きな目標であり，各種関連技術の策定が行われている．以下では，セマンティック Web の技術的な全体像を述べた後，本研究における関連技術である URI，XML，RDF，RDFS，OWL について説明する．

本節の RDF，RDFS，OWL に関する説明は，[20] を参考にした．

2.2.1　セマンティック Web の概要

現在の Web は人間を情報参照の対象とし，人間が理解可能なコンテンツを提供することを目的としているため，ソフトウェアによる理解は困難である．一方，セマンティック Web は，ソフトウェアが意味理解可能な辞書に基づき，Web コンテンツにソフトウェア可読なメタデータを付与することによって，ソフトウェアが Web コンテンツの意味を理解し，推論することを可能にしようという試みである．ここで，「Web コンテンツの意味を理解する」とは，後述するセマンティック Web 技術により，Web コンテンツにどのような概念が含まれているか，また，それらの概念間にどのような関係があるかをソフトウェアにより処理できることとする．Web コンテンツの意味が理解できることで，Web コンテンツをより多くの異なる状況で利用することが可能となる．セマンティック Web の実現により，知的な情報検索，動的な Web サービス連携，様々なアプリケーションを横断したデータの統合および再利用などが可能となり，Web がさらに便利なものとなることが期待されている．

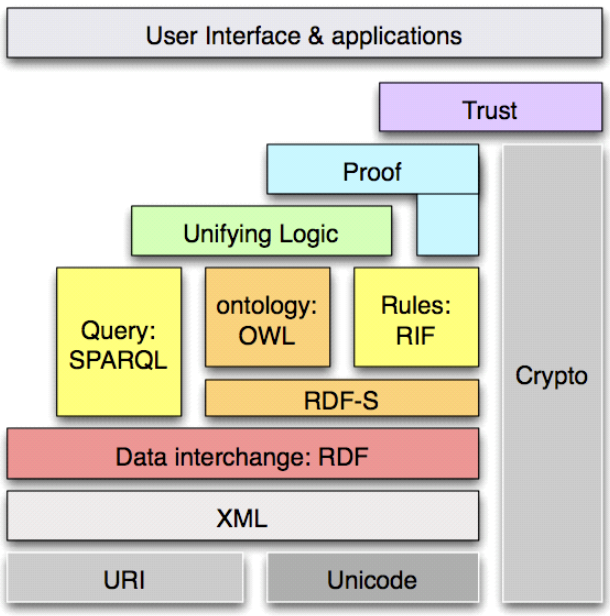


図 2.1: セマンティック Web を実現する技術のロードマップ

セマンティック Web は，Web の創始者でもある Tim Berners-Lee によって提唱され，W3C (World Wide Web Consortium)[1]を中心として関連技術の仕様策定が行われている．セマンティック Web を実現する技術のロードマップを図 2.1 に示す[2]．URI (Uniform Resource Identifier) [3]，Unicode，XML (Extensible Markup Language) [4] といった構文上の基礎となる仕様のもと，主語，述語，目的語の三つ組みによって，Web 上のリソース間の関係を記述するための言語 RDF (Resource Description Framework) [5] が標準化されている．RDF により記述される **RDF コンテンツ**は，RDFS (RDF Vocabulary Description Language: RDF Schema) [6] および OWL (Web Ontology Language) [7] により記述されるソフトウェアが意味理解可能な辞書（**RDFS コンテンツ**および **OWL コンテンツ**）に基づいて構築される．本論文では，RDF コンテンツ，RDFS コンテンツ，OWL コンテンツを合わせて，**セマンティック Web コンテンツと呼ぶ**．また，RDF，RDFS，OWL を検索するための技術として SPARQL [8] も標準化されている．さらに，RIF (Rule Interchange Format), Logic, Proof, Trust が標準化されつつあり，これらの技術により，推論規則の定義，ソフトウェアによる推論，推論結果が正しいことの証明，Web コンテンツの信頼性の保証などが可能となる．

以下では，上記のセマンティック Web 関連技術の中から，本論文と関連の深い技術である，URI，XML，RDF，RDFS，OWL について説明する．

[1] http://www.w3.org

[2] 出典: Emerging Web Technologies to Watch
http://www.w3.org/2006/Talks/1023-sb-W3CTechSemWeb/Overview.html#(19)

2.2.2 URI

URI (Uniform Resource Identifier) [3] とは，抽象的あるいは物理的なリソースを識別するための簡潔な文字列である．ファイル（電子文書，画像，動画）やサービスなど，Web 上に存在し，URI により識別可能なものはリソースである．また，Web 上に存在するものばかりでなく，人間，企業，図書館にある装丁された書籍などもリソースとみなすことができる．

URI の構文は，リソースにアクセスするための手段を表すスキーム (scheme) により異なる．http, ftp, tel, urn など様々なスキームが存在する．ここでは，セマンティック Web コンテンツにおいてよく利用される，http スキームにおける URI の構文について説明する．

http スキームにおける URI は，主にスキーム (scheme)，ホスト名 (host)，パス (path) から構成される．スキームは，前述のとおり，リソースにアクセスするための手段を表す．ホスト名は，リソースを提供するホストマシン名を表す．パスは，ホストマシン内部でのリソースが存在する場所を表す．例えば，`http://www.ietf.org/rfc/rfc2396.txt` では，`http` がスキーム，`www.ietf.org` がホスト名，`/rfc/rfc2396.txt` がパスとなる．

2.2.3 XML

XML (Extensible Markup Language) [4] とは，マークアップ言語を定義するためのメタ言語である．マークアップ言語とは，文書の一部を特殊な文字列（タグ）で囲むことにより，文書の修飾情報や論理構造などを記述するための言語である．XML は，修飾情報や論理構造についての制約条件を記述する機構を提供する．XML により記述される文書は，XML 文書と呼ばれる．

XML 文書は，一つ以上の要素を含む．要素の境界は，開始タグおよび終了タグによって区切る．要素は，親子関係を持つことができ，親要素は複数の子要素を持つことができる．また，要素は，一つ以上の属性を持つことができる．属性は，名前および値を持つ．

図 2.2 は，慶應義塾大学 矢上キャンパスの住所を XML により記述した文書の例である．図 2.2 の住所要素は，郵便番号属性を持つ．また，住所要素は子要素として県名，市名，区名，町名を持つ．

```
<住所 郵便番号="223-8522">
  <県名>神奈川</県名>
  <市名>横浜</市名>
  <区名>港北</区名>
  <町名>日吉</町名>
</住所>
```

図 2.2: XML 文書の例

XML 名前空間

XML 名前空間 (XML namespaces) [21] は，XML における要素名および属性名を URI で修飾することにより，一意に定まる名前をつけるための方法を提供する．XML 名前空間を用いた場合，リソースは**名前空間 URI** および**ローカル名**により表現される．名前空間 URI は，ある目的のために作られた XML における要素名および属性名の集合（語彙）を識別するための URI である．ローカル名は，各語彙に属する名前を表す文字列である．例えば，`http://www.example.com/Book` という URI があったとき，`http://www.example.com/` の部分が名前空間 URI であり，`Book` の部分がローカル名である．ここで，すべての要素名および属性名を URI で記述すると煩雑になる．そこで，XML 名前空間では，名前空間 URI に対して，**名前空間接頭辞** (namespace prefix) と呼ばれる簡潔な別名を割り当てる方法を定めている．例えば，`http://www.example.com/`の名前空間接頭辞を `ex` とすると，`http://www.example.com/Book` は，`ex:Book` と記述することができる．ここで，名前空間接頭辞およびローカル名で表現される名前を**修飾名** (qualified name) と呼ぶ．

2.2.5 項および 2.2.6 項で説明する RDFS および OWL では，XML 名前空間を用いることで，ユーザが定義する様々な語彙を区別できるようにしている．

以下，本節において，`ex` は，`http://www.example.com/`の名前空間接頭辞をあらわす．

2.2.4 RDF

セマンティック Web の目標である「計算機に理解可能な情報」の表現のためには，メタデータなどのリソース相互の関係を，特定のアプリケーションに依存しない形で叙述的に示す共通の方法が必要である．**RDF** (Resource Description Framework) [5] は，Web 上のリソースに関する情報を表現するための言語である．RDF は，主語（subject），述語（predicate），目的語（object）の三つ組み（**トリプル**）によって，関係の連鎖を辿ることができるような **RDF データモデル**と，それを具体的に表現する XML 構文を定めている．RDF データモデルは，特定の構文によらず RDF のリソース記述を表現する方法を規定したものである．RDF データモデルは次の四つから構成される．

リソース

Web 上に存在し，URI により識別可能なもの．ステートメントにおける主語および目的語になることができる．

プロパティ

リソースを記述するために用いる，特徴，関係，属性．ステートメントにおける述語になることができる．それぞれのプロパティは，その意味，許される値の範囲，記述できるリソースのタイプ，他のプロパティとの関係などが 2.2.5 項で述べる RDFS によって定められており，URI により参照される．

表 2.1: RDF ステートメントの例

主語 (subject)	リソース	ex:民情一新
述語 (predicate)	プロパティ	ex:著者
目的語 (object)	リテラル	福沢諭吉

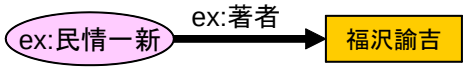


図 2.3: RDF データグラフの例

リテラル

文字列や数値などのデータを直接表現したもの．ステートメントにおける目的語になることができるが，主語や述語になることはできない．

ステートメント

あるリソースとそのプロパティおよび値の組み合わせを結びつけたもの．この三つの要素はそれぞれ主語，述語，目的語と呼ばれる．ステートメントの目的語には，別のリソースもしくはリテラル（文字列あるいは XML の基本データ型）がなることができる．

次に，具体例を示す.「民情一新の著者は，福沢諭吉である」という文を，RDF ステートメントを用いて表現する場合，表 2.1 のようにとらえることができる [3].

RDF データグラフでは，RDF データモデルをラベル付き有向グラフ (labeled directed graph) を用いて図によって表現する．図 2.3 は,「民情一新の著者は，福沢諭吉である」という文を RDF データグラフで表したものである．RDF データグラフでは，リソースを楕円で，プロパティを矢印で，リテラルを矩形で表現する．矢印は主語を始点とし，目的語を終点とする．

リソースのタイプ

RDF では，いくつかのリソースを**クラス** (Class) の**インスタンス** (instance) としてグループ化することができる．ここで，クラスとは，リソースのグループを表す．クラスに属するリソースを，クラスのインスタンスと呼ぶ．リソースがあるクラスに属することは，`rdf:type` プロパティを用いて表す．あるリソースにおける `rdf:type` プロパティの値を，そのリソースのタイプと呼ぶ．

[3] 自然言語における主語，述語，目的語と厳密には一致しないが，RDF におけるステートメントはリソース間の関係定義であるため，表 2.1 のようにとらえることができる．

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:ex="http://www.example.com/">
  <rdf:Description rdf:about="http://www.example.com/民情一新">
    <ex:著者>福沢諭吉</ex:著者>
  </rdf:Description>
</rdf:RDF>
```

図 2.4: RDF/XML の記述例

RDF/XML

RDF の XML 構文は，“RDF/XML Syntax Specification” [22] で定義されており，略して RDF/XML と呼ばれる．RDF データモデルを RDF/XML で表現するためには，主語，述語，目的語をそれぞれ XML の要素として記述する．表 2.1 に示す RDF ステートメントを RDF/XML で記述すると図 2.4 のようになる．RDF/XML では，rdf:RDF 要素内に，RDF データモデルの内容を記述する．ここで，`rdf` は RDF の名前空間 URI `http://www.w3.org/1999/02/22-rdf-syntax-ns#`を表す．RDF/XML では，主語および目的語は，rdf:Description 要素を用いて記述する．rdf:Description 要素の rdf:about 属性により，主語または目的語を表すリソースの URI を参照する．述語は，その名前を修飾名（名前空間接頭辞とローカル名の組み合わせ）で表したプロパティ要素として記述する．目的語がリテラルの場合には，rdf:Description 要素を用いずに，プロパティ要素の内容として直接リテラル文字列を記述する．

RDF/XML は，上記以外にも様々な省略構文などが定義されており，[22] に詳細に説明されている．

2.2.5 RDFS

RDFS は，RDF データモデルに語彙表現の仕組みを加えて拡張するものである．RDF データモデルは，プロパティを用いてリソース間の関係を表すが，そのプロパティ自身を定義する方法は定めていない．また，リソースがどのようなグループに属するかを表すクラスについても定義する方法を定めていない．**RDFS** (RDF Vocabulary Description Language: RDF Schema) [6] は，これらのプロパティおよびクラスを定義するための手段を提供する．また，RDFS はクラスの IS-A 関係 (`rdfs:subClassOf`)，プロパティの IS-A 関係 (`rdfs:subPropertyOf`)，プロパティの定義域 (`rdfs:domain`) および値域 (`rdfs:range`) を記述するための手段も提供している．

RDFS では，あらかじめ用意されたクラスおよびプロパティ（基本クラスおよび基本プロパティ）を用いて語彙を定義していく．以下では，基本クラスおよび基本プロパティについて説明する．また，`rdfs` は RDFS の名前空間 URI `http://www.w3.org/2000/01/rdf-schema#` を表す．

基本クラス

基本クラスとは，RDFS があらかじめ用意している，クラスを表現するためのクラスである．

rdfs:Resource
: 最も基本的なクラス．すべてのリソースは，rdfs:Resource クラスのインスタンスである．また，すべてのクラスは rdfs:Resource クラスの下位クラスとなる．

rdfs:Class
: クラスを定義するためのクラス（クラスクラス）．RDFS で表現するクラスは，すべて rdfs:Class クラスのインスタンスとなる．

rdf:Property
: プロパティを定義するためのクラス（プロパティクラス）．RDFS で表現するプロパティは，すべて rdf:Property クラスのインスタンスとなる[4]．

rdfs:Literal
: 文字列などのリテラルを表すクラス．

rdfs:XMLLiteral
: タグを含んだ XML 文字列を表すデータ型のクラス．

rdfs:Datatype
: 独自のデータ型を定義するためのクラス．

基本プロパティ

基本プロパティとは，RDFS があらかじめ用意している，クラス間およびプロパティ間の IS-A 関係，プロパティの定義域および値域，リソースの見出しや説明を記述するためのプロパティである．以下では，rdf:type プロパティの厳密な定義も示す．

rdf:type
: 主語リソースが目的語クラス（rdf:type プロパティの値）のインスタンスであることを示すプロパティ．主語リソースは目的語クラスの全ての性質を備える．クラスを定義する場合は，rdf:type プロパティの値を rdfs:Class クラスとする（クラスは rdfs:Class クラスのインスタンスとなる）．

rdfs:subClassOf
: 主語となるリソースが，目的語となるリソースの下位クラスであることを表すプロパティ．rdfs:subClassOf プロパティの主語および目的語となるリソースはクラスである．あるクラスをより精密に定義するときに用いる．

[4] クラスという概念は RDFS で導入されるが，プロパティは RDF のトリプルを構成する基本概念であるため，rdf:の名前空間に属するものとして表現される．

rdfs:subPropertyOf
主語となるリソースが，目的語となるリソースの下位プロパティであることを表すプロパティ．rdfs:subPropertyOf プロパティの主語および目的語となるリソースはプロパティである．あるプロパティをより精密に定義するときに用いる．

rdfs:range
プロパティの主語となるリソースのクラス（定義域）を表すプロパティ．

rdfs:domain
プロパティの目的語となるリソースのクラス（値域）を表すプロパティ．

rdfs:label
リソースに人間が読むための名前（見出し）を付けるためのプロパティ．

rdfs:comment
リソース定義に人間が読むための説明を加えるためのプロパティ．

クラスの定義

図 2.5 は，rdfs:Resource クラスの下位クラスとして，ex:書籍クラスと ex:人間クラスを定義したことを表している．なお，親クラスを持たないクラスは，rdfs:Resource クラスの下位クラスとなる．

図 2.5 は，RDF データモデルにより表現できる．例えば,「ex:書籍は，rdfs:subClassOf プロパティを持ち，その値は rdfs:Resource である」と表現できる．図 2.5 のクラス定義を，RDF/XML により記述すると図 2.6 のようになる．

プロパティの定義

プロパティは，rdf:Property クラスのインスタンスとして定義する．プロパティがどのような性質を持つかを定めるために，rdfs:domain および rdfs:range プロパティを用いて範囲の制約を課す．

ここでは例として，ex:著者というプロパティを定義してみる．図 2.7 は，ex:著者というプロパティの定義域が ex:書籍クラスであり，値域が ex:人間クラスであることを RDF/XML により記述したものである．

プロパティをより精密化するするためには，rdfs:subPropertyOf プロパティを用いる．

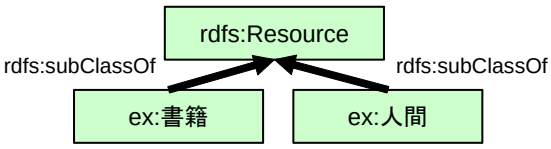


図 2.5: クラス定義の例

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
   <rdfs:Class rdf:about="http://www.w3.org/2000/01/rdf-schema#Resource"/>
   <rdfs:Class rdf:about="http://www.example.com/人間">
     <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
   </rdfs:Class>
   <rdfs:Class rdf:about="http://www.example.com/書籍">
     <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
   </rdfs:Class>
</rdf:RDF>
```

図 2.6: クラス定義の RDF/XML による記述例

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
   <rdf:Property rdf:about="http://www.example.com/著者">
     <rdfs:domain rdf:resource="http://www.example.com/書籍"/>
     <rdfs:range rdf:resource="http://www.example.com/人間"/>
   </rdf:Property>
</rdf:RDF>
```

図 2.7: プロパティ定義の RDF/XML による記述例

2.2.6 OWL

RDFS は基本的なクラス（IS-A 階層）とプロパティ定義の手段を提供しているが，計算機が様々な領域で意味を理解しながら活動するためには，より詳細で厳密な知識を記述可能なオントロジー記述言語が必要となる．セマンティック Web では，様々な場所で語彙や知識ベースが構築されうるため，これら相互の関係を示し，統合や相互利用ができるような仕組みが重要となる．セマンティック Web におけるオントロジー記述言語として，**OWL** (Web Ontology Language) [7] が W3C により勧告されている．OWL では，RDFS のようなクラス定義，プロパティ定義に加え，クラスの論理的組み合わせ，プロパティの制約条件，他のオントロジーとの関係など，セマンティック Web におけるオントロジーを記述するうえで必要となってくる機能を備えている．利用目的に応じて，OWL-Lite, OWL-DL，OWL-Full の三つを使い分けるようになっている．

OWL-Lite

クラス階層の構築と基本的な制約条件の記述が主目的である場合に使える，OWL-DL のサブセット．OWL-DL/OWL-Full で使える語彙の一部が利用できなかったり，語彙の使い方が制限されている．OWL の利用を促進するため，ツール作成者にとって使いやすい言語の提供を目的としている．

OWL-DL

OWL の語彙をすべて利用しつつ，記述論理 [23] に基づき，論理計算の完全性と決定可能性を確保するよう設計されている．このために，OWL-DL ではタイプの区別を厳密にしており，クラスでありかつインスタンスでもあるといったリソースの存在を認めていない．厳密な推論を行うシステムのためのオントロジーを記述できる．

OWL-Full

OWL の語彙と RDF の柔軟性を最大限利用するための言語．OWL の中で最も表現力があるが，計算の完全性，決定可能性は保障されない．OWL-Full は，RDFS などで表現されている語彙をより詳細に記述したり相互に関連づけたりするために用いることができる．

OWL コンテンツは，オントロジー全体の情報を記述する**ヘッダ**，クラスの定義を記述する**クラス公理**，プロパティのグローバルな定義を記述する**プロパティ公理**，クラスのインスタンス（個体）に関する公理を記述する**事実**の四つから構成される．以下では，それぞれについて簡単に述べる．

ヘッダ

ヘッダには，外部オントロジーを取り込むための記述やバージョン情報を記述する．**owl:Ontology** クラスのインスタンスにオントロジー・プロパティ (**owl:OntologyProperty**) と注釈型プロパティ (**owl:AnnotationProperty**) を用いて記述する．オントロジープロパティには，**owl:imports**，**owl:priorVersion**，**owl:backwardCompatibleWith** などがあり，外部オントロジーの取り込みや古いバージョンのオントロジーとの関連を記述するために用いられる．注釈型プロパティには，**rdfs:comment**，**owl:versionInfo** などがあり，バージョン番号，作成日付，コメントなど，人間のための情報を記述するために用いられる．これらは，オントロジーの連動や発展性の確保のために必要となる．

クラス公理

クラスは，同様の性質を持つリソースをグループ化し，その性質を論理的に表現するための機能を提供する．OWL では基本語彙 **owl:Class** によってクラスを表現する．クラスのインスタンスは**個体** (Individual)，インスタンスの集合はクラスの**外延**と呼ばれる．OWL のクラスは，クラス表現という構成要素を組み合わせ，クラス公理として定義される．OWL のクラス表現には，URI 参照によるクラスの名前付け，クラスのインスタンスとなる個体の列挙，プロパティの制約，二つ以上のクラス表現の積，二つ以上のクラス表現の和，クラス表現の否定の 6 通りがある．URI 参照によるクラスの名前付け以外のクラス表現は，クラス外延のメンバーであるための条件を記述する．

プロパティ公理

プロパティ公理は，RDFS 語彙を用いたグローバルな定義域，値域，階層関係の他，OWL の語彙を用いた論理的性質の記述で構成される．OWL のプロパティには，個体同士の関係を記述する個体値型プロパティ (**owl:ObjectProperty**) と，個体をデータ値と関係づけるデータ型プロパティ (**owl:DatatypeProperty**) がある．また，オントロジーの記述に用いる特殊なタイプとして注釈型プロパティ (**owl:AnnotationProperty**) とオントロジープロパティ (**owl:OntologyProperty**) が用意されている．プロパティの公理は，この四つのタイプのいずれかを示すトリプルを持ち，RDFS の **rdfs:domain**，**rdfs:range** で定義域，値域を加えて記述することができる．

事実

OWL では，個体に関する公理を**事実** (fact) と呼ぶ．事実は，ある個体がどのようなクラスに属するか，また，どのようなプロパティ値を持つかを記述することにより表される．

2.3 オントロジー

本節では，オントロジーの定義および本研究と関連のある汎用オントロジーについて説明する．

2.3.1 定義

オントロジーとは，本来，哲学の用語であり,「存在に関する体系的な理論（存在論）」という意味を持つ．一方，人工知能の立場からは,「概念化の明示的な記述」という定義がなされる．ここで概念化とは，対象（世界）に関して，興味を持つ概念とそれらの間の関係の明確化を指す．我々が世界を認識し，そのモデルを計算機内に作ろうとするときには，必ず，世界を概念化する．例えば，ビジネスプロセスのモデルを構築する際には，ビジネス自体を構成する概念を洗いだし，それらの間の関係を整理し，概念を特徴付け，他と区別する属性を明確にすることでそれらの概念を記述する．概念のなかには，オブジェクト（名詞に対応）だけではなく，活動を表す動詞的なものも存在する．そして，活動が対象とするオブジェクトとそれらを関係付ける制約など，これら全てを体系化したものをオントロジーと呼ぶ．知識べースの立場からは,「人工システムを構築する際のビルディングブロックとして用いられる基本概念/語彙の体系（理論）」という定義がなされる．知識ベースは対象とする世界における問題解決モデルを構築するときに不可欠となるものである．問題解決を対象とするので，オントロジーは問題解決過程に固有の概念化であるタスク（メソッド）に関するオントロジーであるタスクオントロジー（およびメソッドオントロジー），そしてタスクが実行される領域（ドメイン）に関わるドメインオントロジーの大きく 2 種類に分かれる．また，オントロジーは複数の人の間で共有される合意内容で

あるといった見方もある．共有される合意内容としては，ドメイン知識のモデル化の概念フレームワーク，相互に交信し合うエージェントの会話の内容に関わる通信規約，ドメインの知識を記述するときの合意事項，などが含まれる．知識共有の考えのなかでは，オントロジーは表現のための語彙の定義の形で記述される．最も簡単な場合としては，カテゴリとそれらの間の上下関係（包含関係）を規定したもの（概念階層）であり，関係データベースにおける概念スキーマがある．また，自然言語理解の研究分野では，一般的かつ網羅的な概念関係を定義した汎用オントロジーの開発が精力的に行われている．

上記の定義は，オントロジー工学調査委員会 [24] で挙げられた定義を参考にした．

2.3.2 汎用オントロジー

自然言語理解の研究分野では，電子化辞書 (MRD:a Machine Readable Dictionary) の開発が精力的に行われており，オントロジーというと電子化辞書を指す場合が多い．よく知られている電子化辞書としては，WordNet（プリンストン大学），ROGET's Thesaurus (MICRA)，LDOCE (Longman's Dictionary)，SENSUS（南カリフォルニア大学情報科学研究所），EDR 電子化辞書（情報通信研究機構）などがある．電子化辞書の特徴として，定義される概念が一般的かつ多くの分野にわたっている点があげられる．そのため，電子化辞書は汎用オントロジーとしてとらえることができる．

以下では，概念階層構造が整っていることから広く使われている WordNet，階層構造としての IS-A 関係だけでなく他の概念関係子もサポートしている EDR 電子化辞書，既存の電子化辞書をマージして構築された SENSUS について概略を述べる．

WordNet

WordNet [17] (version 2.0) は，図 2.8 に示すように，名詞句辞書，動詞句辞書，形容詞句辞書，副詞句辞書，見出し句辞書から構成されており，総計約 15 万個の語彙を保持し

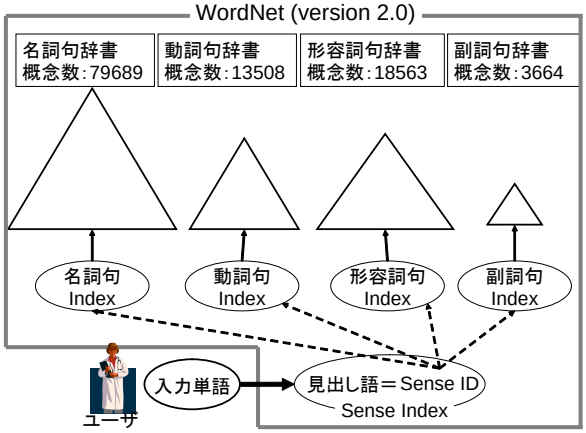


図 2.8: WordNet の概観

表 2.2: WordNet (version 2.0) の辞書サイズ

辞書名	見出し（語彙）数	意味（概念）数
名詞句辞書	114648	79689
動詞句辞書	11306	13508
形容詞句辞書	21436	18563
副詞句辞書	4669	3664
見出し句辞書	152059	115424

ている．各々の辞書に記録されている見出し数および概念数を表 2.2 に示す．

見出し句辞書は，見出し句，意味情報としての概念 ID，辞書編集情報，品詞情報などから構成されている．概念 ID は，見出し句辞書と各辞書の間のリンクとして機能している．同じ概念を意味するいくつかの単語見出しが，同じ概念 ID によって一つの概念にまとめられており，この集合を **synset (synonym set)** と呼ぶ．WordNet 内では，この synset を単位として階層・定義の記述が成されている．

名詞句辞書と動詞句辞書のみが階層構造を持つ．また，一部の概念 ID には，反対概念の概念 ID，`part of`，`member of`，`substance of` 関係の概念 ID なども与えられている．

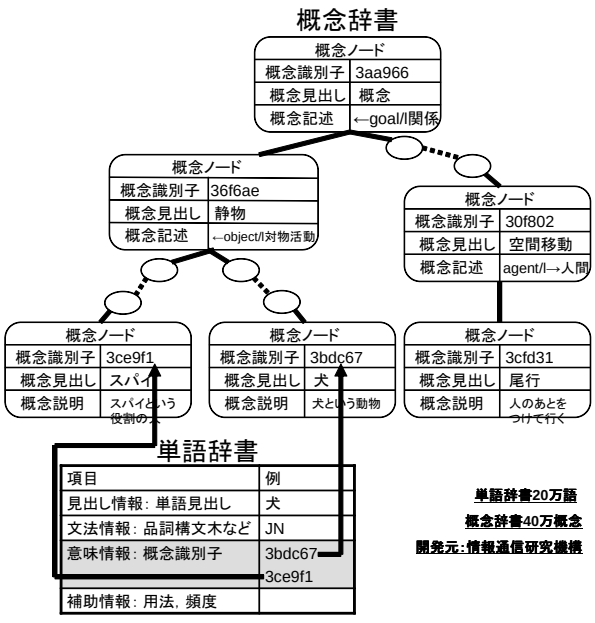


図 2.9: EDR 電子化辞書の概観

EDR 電子化辞書

EDR 電子化辞書 [18] は，単語辞書，対訳辞書，概念辞書，共起辞書，専門用語辞書（情報処理），EDR コーパス，から構成される．本論文では，図 2.9 に示すように，単語辞書

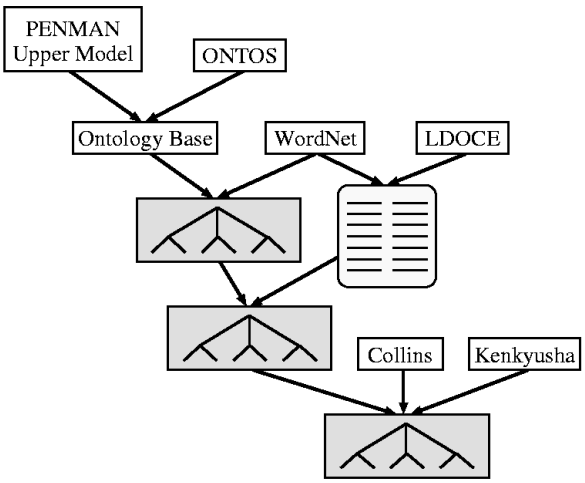


図 2.10: SENSUS の概観

と概念辞書を利用している．単語辞書には日本語単語辞書と英語単語辞書がある．単語辞書は，見出し情報，文法情報，意味情報，補助情報から構成されているが，意味情報には，概念辞書の各概念ノードを識別するための概念識別子が割り当てられ，単語辞書と概念辞書を結合する働きを持っている．一方，概念辞書には，多重継承を許す概念階層関係を定義した概念体系辞書と，agent（動作主体）, object（対象）, goal（目標）, implement（道具・手段）, cause（原因）, place（場所）, scene（場面）, a-object（属性）という 8 種類の概念関係子による概念間関係を定義した概念記述辞書がある．各概念は，主に，概念識別子，概念見出し，概念の説明を持つ．

SENSUS

SENSUS は，図 2.10 に示すように，PENMAN Upper Model [25] と ONTOS [26] をマージして，上位オントロジーとして Ontology Base を構築した後，WordNet や LDOCE を順次その下位にマージしていくことによって構築された電子化辞書である．構築プロセスは，完全に自動化されていないが，二つの概念の定義文における共有単語の有無によって，二つの辞書間で対応する概念を見出したり，対応する概念の上位下位概念群に注目して，さらに対応する概念を見出すような照合アルゴリズムが利用されており，マージ作業を効率化している．

2.4 オントロジー構築方法論

本節では，M.Uschold ら [27] によるオントロジー開発のための総合的方法論を紹介する．彼らはオントロジー開発を図 2.11 に示す段階に分けて整理した．

はじめに，オントロジーのスコープと目的を明らかにし，利用者の特徴付けを行う．オントロジーのスコープや目的を明らかにすることはオントロジー構築のための明確な目

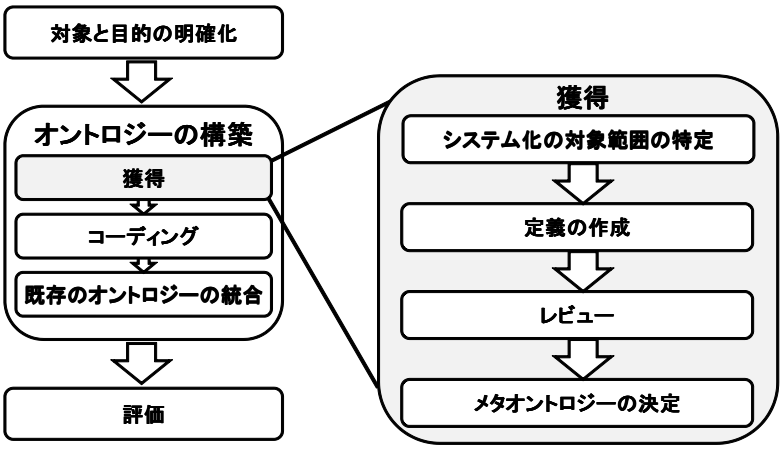


図 2.11: オントロジー開発の流れ

標を提供することに役立つ．オントロジー構築は**獲得**，**コーディング**，**既存オントロジーの統合**の三つのステップがある．以下，各ステップについて述べる．

2.4.1 オントロジーの獲得

オントロジー獲得は重要概念の切り出しとその定義からなる．オントロジー構築の主要な作業はこのオントロジー獲得であり,「オントロジー構築」はオントロジー獲得を指す場合が多い．以下，四つの段階について順に述べる．

システム化の対象範囲の特定

はじめに，すべての関連する可能性のある用語を洗い出す．この際に，概念の粒度について十分に注意しなければならない．次に，切り出された概念を，類似あるいは参照関係にある概念同士でまとめて分類する－これをワークエリアと呼ぶ．その後，その領域内の意味的な相互関係を特定する．

定義の作成

前述の処理で，重要な概念のほとんどの語彙が識別される．オントロジー構築の主な作業は定義の作成である．次に定義を作成する際に，まず注意すべきことを示す．

メタオントロジー

概念の熟考を行い，どのような class, object, relationship などを導入するかを決定する．様々な可能性を心に留めておき，一貫した手法で適切なところで切り出した重要概念である単語および熟語（role, entity, relationship, type, instance など）を使用する．

ワークエリア
他のワークエリアとの重なりなどを考慮して，複雑なものから順に用語の定義に取り組む．

用語
各ワークエリアで最も基本的な用語を定義し，その語より抽象的または固有な用語へ進むミドルアウトアプローチをとる．ミドルアウトアプローチは，その概念の詳細度によってバランスが取れるため，基本的な概念を特殊化または一般化することによって必要な概念のみを表せる．

表現方法
技術的な正確さと明確さの適切なバランスをとる．定義の作成の時点では，概念や概念間の関係の定義は，自然言語により行う．コーディング時に，技術的に正確な定義を形式言語で行うことができるように，正確かつ明確に記述を行う．

あいまいな用語の処理
オントロジーに本当に必要なものを考え，できるだけ少ない技術的用語によって各概念の定義を行う．

定義段階全般にわたるガイドライン
用語の使用においては一貫性を確実にし，定義における循環は避ける．また適切な例を与える．

レビューおよびメタオントロジーの決定

定義を批判的にレビューし，適切に訂正する．その後，絶対的な要求仕様として，自然言語の定義を使ってメタオントロジーを考案する．

2.4.2 コーディング

コーディングとは，形式言語によって，獲得された概念化を明示的に表現することである．この段階は以下の流れで行われる．

- オントロジーで使われるような基礎用語（class, object, relation など），メタオントロジーの決定
- メタオントロジーをサポートできる表現言語の選択
- コード（形式言語の命令を記述する文書）の記述

2.4.3 既存オントロジーの統合

構築中のオントロジーに，既存オントロジー中の概念が存在しなければ，同義語を同定し，既存オントロジーを再利用して，構築中のオントロジーを拡張することは容易である．しかしながら，既存オントロジーに，構築中のオントロジーに含まれる概念と類似する概念が存在する場合，そのような概念をどのように再利用すればよいかは，明確ではない．

2.5 オントロジー構築支援環境

オントロジー構築支援方法としては，主に，手動構築支援と（半）自動構築支援がある．

手動構築支援では，オントロジーエディタを用いた視覚的編集により，特定のオントロジー記述言語の記述形式でのオントロジー構築を支援する．しかしながら，手動構築は構築するオントロジーの規模が大きくなるとユーザの負担が大きくなることから，既存情報資源を活用して（半）自動的にオントロジー構築を支援する研究が行われている．(半）自動的にオントロジー構築を支援する際に利用される既存情報資源としては，電子化辞書，既存領域オントロジー，専門文書などがある．

自然言語理解の研究分野では，電子化辞書 (MRD:a Machine Readable Dictionary) の開発が精力的に行われており，計算機で利用可能な辞書の環境が整いつつある．電子化辞書で提供されている概念階層構造は，多数の概念を網羅しており，領域オントロジー概念階層構造を構築するための基盤として利用可能である．ところが，電子化辞書は言語処理の観点から整理された概念階層構造であることから，特定の問題領域における概念階層構造と比較すると，種々の差異があると推定される．この差異は，タスクやコンテキストに依存して概念の意味が変化する概念変動 (Concept Drift) の一種であり，それを管理する機構が課題となる．

本節では，まず，領域オントロジー構築において課題となる概念変動について説明する．次に，手動構築支援ツール，電子化辞書を利用した概念階層構築支援環境，既存オントロジーを利用した領域オントロジー構築支援手法，専門文書を利用した概念階層・概念定義の学習手法を紹介する．

2.5.1 概念変動

機械学習の分野においても概念変動 (Concept Drift) という概念が使われる．コンテキストが不安定な領域における学習において概念変動の現象が起こる．コンテキストは時間によって変化する．漸次，または不意のコンテキストの変化はしばしば，概念変動，つまり概念の意味変化として明らかになる．

コンテキストは，対象領域が変化することによっても変化すると考えられる．よって，対象領域の変化による，概念の意味変化も概念変動であるといえる．概念の意味変化は，概念階層の変化として明らかになる．以下に，電子化辞書を利用してある領域世界を表す概念階層を構築する際に発生する概念変動について，例を挙げて説明する．

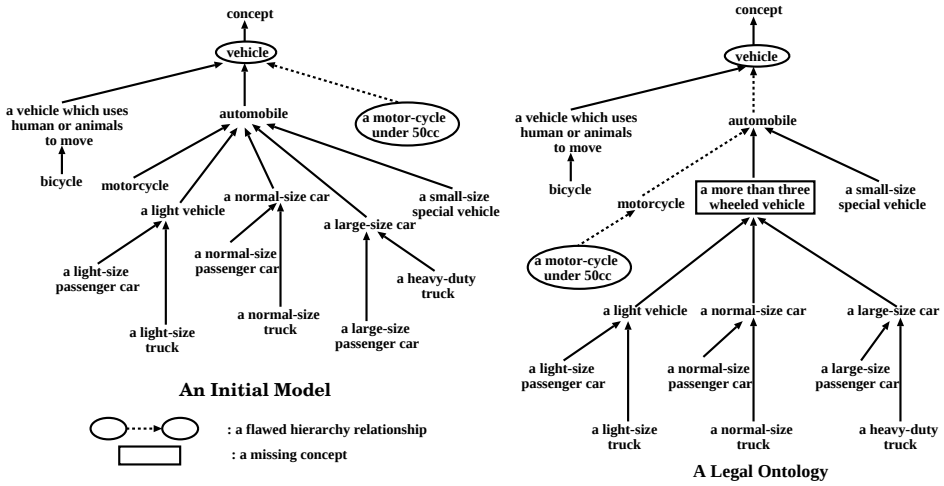


図 2.12: Ontological Bugs

ある領域世界の概念に関連する情報を電子化辞書から抽出したものを，ここでは，初期モデル (An Initial Model) と呼ぶ．電子化辞書は一般的な概念の定義がなされているため，電子化辞書から抽出した初期モデルの階層関係も一般的な（特定の領域に依存しない）定義となっている．図 2.12 は，道路交通法の世界を表した，初期モデルおよび法律オントロジー (A Legal Ontology) における概念階層を示している．初期モデルには，領域固有の概念が不足していたり，領域固有の観点からみると概念階層の構造に不備があるなど，領域オントロジーとしては十分なものとはいえない．初期モデルを領域オントロジーとして再利用する際の問題として，不足概念，不要な概念および不要な概念定義，領域オントロジーとは異なる階層関係および概念定義などがあげられる．

図 2.12 の初期モデルを道路交通法の領域オントロジーとして利用するためには，二カ所を修正する必要がある．ここでは，初期モデルを領域オントロジーとして再利用する際に，概念変動により修正すべき箇所のことを，Ontological Bugs と呼んでいる．図 2.12 の法律オントロジー中の長方形で囲まれている “A more than three wheeled vehicle” は，初期モデルに不足している概念であり，追加する必要がある．もう一つは，初期モデル中では親子関係にある “vehicle” と “a motor-cycle under 50cc” の階層関係である．道路交通法という領域から見れば，法律オントロジーの点線のパスで示されるような，先祖-子孫の関係として定義されるべきである．上記の二カ所を修正することによって，初期モデルは，領域オントロジーとなる．

次に，階層関係以外の関係（その他の関係）である概念定義構築への電子化辞書の利用を考察する．ある概念の概念定義は，概念関係子と概念関係子の値の組で表現される．概念定義において，一方の概念から見たときの関係を概念関係子，他方の概念をその概念関係子の値と呼ぶ．例えば,「人間は行為の agent である」という関係では，人間という概念から見たとき，agent が概念関係子であり，行為が概念関係子の値になる．

EDR 電子化辞書は概念記述辞書を持っており，階層構造以外の概念関係子もサポートしている．EDR 電子化辞書は，約 40 万個の概念の定義を与えており，網羅する概念数は多

いといえる．しかしながら，8 種類の比較的一般的な概念関係子 (agent, object, a-object, goal, implement, cause, place, scene) が与えられているのみで，領域固有の概念関係子に関する情報は保持されていない．また，保持されている概念関係子の値についても，その値は例示レベルの値が直接的に数多く記述され，集約作業は何ら行われていないため，抽象化などの作業が必要となる [28].

概念定義における概念変動はこの概念関係子の変動と概念関係子の値の変動が考えられる．ある概念が電子化辞書で定義されているとき，それをある対象領域に適用すると，利用可能な概念関係子と利用不可能な概念関係子が現れる．新たに必要になる概念関係子も現れる．概念定義構築の基盤として電子化辞書を利用しようとすると，こうした概念関係子の変動と概念関係子の値の変動の両方を管理しなければならない．また，その変化についての知識は専門家しか持っていないため，専門家に構築を頼ることになる．

2.5.2 手動構築支援

主なオントロジー手動構築支援ツールとして，Protégé [13] と法造 [29] がある．

Protégé は，スタンフォード大学で開発されている，2007 年現在，世界で最も利用されているオントロジーエディタである．Protégé は，主に OWL コンテンツの構築を支援しており，OWL-DL レベルのオントロジーを構築可能となっている．また，プラグインによる拡張機能を備えており，オントロジーの視覚化プラグインなど様々なプラグインが開発されている．

法造は，大阪大学 溝口研究室で開発されている，オントロジーの基礎理論に関する考察に基づいたオントロジーエディタである．オントロジーを構築するうえで重要な意味を持つ**ロール概念**を明示的に支援している点が特徴である．また，法造は，オントロジーの分散開発なども支援している．

2.5.3 電子化辞書の利用

電子化辞書を利用した代表的なツールの一つとして，南カリフォルニア大学情報科学研究所で開発されているオントロジーブラウザ Ontosaurus がある [30]．Ontosaurus では，図 2.13 に示すように，空軍作戦 (Air Campaign) に関連した入力語彙と 2.3.2 項で述べた電子化辞書 SENSUS 間の照合をとり，照合されたノード (ACP Seed Term) と根節点間のパス (Path to Root) を中心に分析し，そのパス上に入力語彙以外の空軍概念 (Military Terms) が見出されたり，あるいは，その空軍概念の上位概念 (Frequent Parent) やその下位部分木に含まれる概念 (Subtree Terms) に新しい空軍概念が見出されて，空軍概念階層木が構築されていく．Ontosaurus は，人手に委ねられる部分が依然大きいが，電子化辞書を利用した最初の領域オントロジー構築支援環境として意義があるといえる．

Nvigli らが提案している OntoLearn [31] は，電子化辞書 WordNet と自然言語処理の技術を利用して領域オントロジー構築を支援している．彼らのアプローチでは，自然言語処理の技術を用いて WordNet の概念に追加・削除を行うことで，WordNet の概念を領域

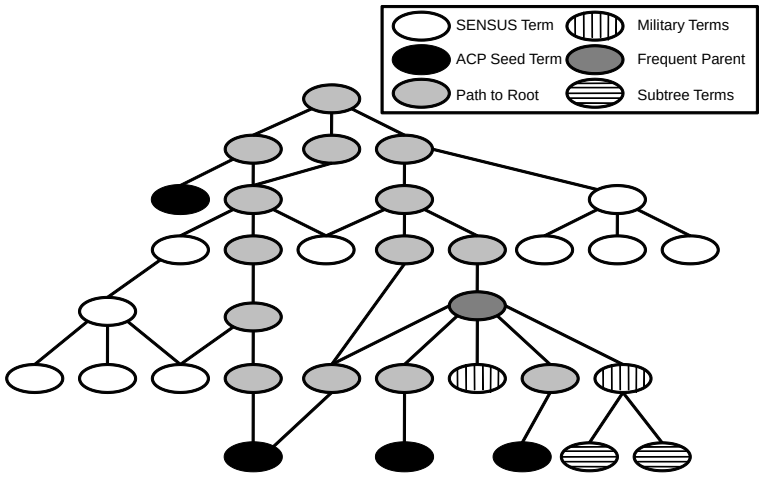


図 2.13: 電子化辞書を利用した領域オントロジーの構築

概念に適合させる．オントロジーの自動構築は，ターミノロジー翻訳アプリケーションのケーススタディにおいて実際に用いることができると，彼らは主張している．

2.5.4 既存オントロジーの利用

既存オントロジーを利用した領域オントロジー構築支援の研究の一つに，アムステルダム大学の G.Heijst らが検討しているオントロジーライブラリがある [32]．彼らのアプローチでは，クラス，関係，関数，インスタンスの集合から構成される定理を基本単位として，汎用性の高い定理集合としてのコアライブラリとそれ以外の定理集合としての周辺ライブラリを事前に構築しておく．そして，新しい問題領域におけるドメイン階層とメソッド階層を構築したとき，それらをドメイン固有性とタスク（メソッド）固有性の観点から，この二つのライブラリに関連づけることによって，オントロジーライブラリを拡張していく．

2.5.5 専門文書の利用

概念定義を構築するためのリソースとして，その対象領域に関する専門文書が利用できる．汎用的な情報しか持たない電子化辞書と比べ，その対象領域に関する専門文書からは，より有用な情報を得ることが期待できる．テキスト情報の例としては，法律オントロジーを構築するのであれば，その法律の条文が挙げられる．また，現在，精力的にオントロジーの構築が行われているエンタープライズにおいては，自然言語でなされた定義が存在する [33]．適切なテキスト情報が存在しない場合であっても，オントロジーを構築する場合，まずは自然言語で定義するのが一般的である．

動詞指向の手法を用いた研究では，主に動詞とそれに構文的に関わる名詞との関係を記述し，そこから概念関係を導いている．[16] では，機械学習手法による階層関係 (taxonomic

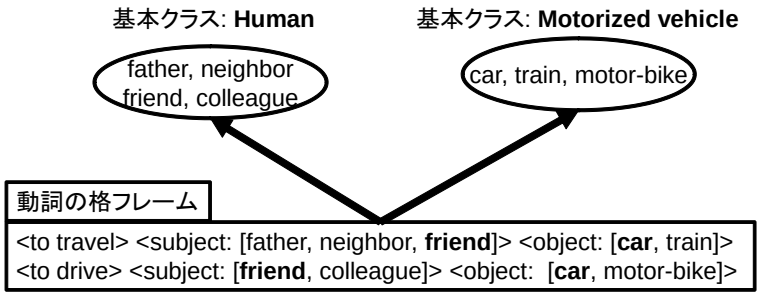


図 2.14: ASIUM の概念クラスタリング

relation) および動詞の格フレームの抽出を行っている．このシステム ASIUM で用いられている概念クラスタリング手法は，抽出された格フレームのフレーム名とスロットが，2 種類以上の格フレームで共通する名詞がある場合，その名詞を含むスロット値に含まれる名詞を一つの概念としてまとめ，これを基本クラス (base class) にするというものである．図 2.14 では，<to travel> <subject: [father, neighbor, friend]> <object: [car, train]> と <to drive> <subject: [friend, colleague]> <object: [car, motor-bike]> の 2 種類の格フレームが抽出された場合の例を示している．ここで，to travel および to drive がフレーム名，subject および object がスロットとなる．図 2.14 では，friend および car が，フレーム名とスロットが，2 種類以上の格フレームでに共通する名詞である．friend を含むスロット subject のスロット値である，father, neighbor, friend, colleague をまとめて，基本概念 Human としている．また，car を含むスロット object のスロット値である，car, train, motor-bike をまとめて，基本概念 Motorized vehicle としている．基本クラスを更にクラスタリングすることでオントロジー（概念階層）を構築する．

データマイニングの手法を用いた研究として，相関ルールアルゴリズムを用いた概念定義学習手法が [15] によって提案されている．構文解析から得られた用語間の係り受け情報から概念対を抽出し，抽出された概念対をトランザクションとした相関ルールを Apriori アルゴリズムを用いて求めている．相関ルールから得られる概念関係は，出現頻度を強く反映したものとなるため，必ずしも高い精度での学習が可能であるとはいえないが，簡易テキスト処理にヒューリスティクスを加えることで，比較的軽い処理でトランザクション抽出が行える．

2.6 まとめ

本章では，はじめにセマンティック Web の概要を述べ，本論文に関連の深い技術である URI，XML，RDF，RDFS，OWL について説明した．また，セマンティック Web コンテンツにおけるオントロジー構築に関連して，オントロジーとその開発方法論について述べた．最後に，関連研究として，オントロジー構築支援に関する研究を紹介した．

第3章　RDF(S)コンテンツ構築支援ツール

本章の内容は，文献 [34] に基づいている．

3.1　概要

セマンティック Web は，ソフトウェアが Web コンテンツを理解するためのセマンティック Web 標準技術である RDF (Resource Description Framework) [5]，RDFS (RDF Vocabulary Description Language: RDF Schema) [6]，OWL (Web Ontology Language) [7] を基礎とした，次世代 Web の最も有力な候補の一つである．セマンティック Web の目的は，人間のためだけでなくソフトウェアが意味理解可能な Web コンテンツを構築することにある．セマンティック Web の実現により，知的な情報検索や様々なアプリケーションを横断したデータ統合および再利用が可能となる．

セマンティック Web を実現するためには，RDF，RDFS，OWL により記述されるセマンティック Web コンテンツの普及が不可欠である．しかしながら，セマンティック Web コンテンツは主に RDF/XML で記述されるため，XML 構文の厳密さが原因となり，ソフトウェアが処理することは容易だが，人間が読解し，記述し，修正を行うことは容易ではない．そのため，セマンティック Web コンテンツの構築を支援するために様々な研究が行われている．

RDF(S)（RDF および RDFS）コンテンツの記述を支援するために，数多くのグラフィカルエディタが提案されている．それらの多くの主な機能は，単純に XML 記述を支援するのではなく，“主語-述語-目的語” の三つ組みにより表現可能な RDF データモデルに基づいて XML ベースの記述を視覚的に表示および編集することにある．グラフィカルエディタにより，RDF(S) コンテンツを視覚的に理解することが可能となり，RDF データモデルを XML 形式で容易に記述することが可能となる．しかしながら，RDF(S) コンテンツを RDF データモデルに忠実に視覚化を行うことには問題がある．RDF および RDFS コンテンツは，同一構文で記述されるため，忠実に RDF データモデルを視覚化すると，両者の区別を行うことが困難となる．RDFS コンテンツは，現実世界のオブジェクトからは離れて，対象の概念化を考慮して構築が行われる．一方，RDF コンテンツは現実世界のオブジェクトを考慮して構築が行われる．このように，RDF コンテンツおよび RDFS コンテンツは，それぞれ異なる観点から構築されるため，両者の区別を行うことが困難な場合には，ユーザの RDF(S) コンテンツ構築コストは高くなるという問題がある．

その他の RDF(S) コンテンツの記述を支援するための研究として，RDF コンテンツのためのモデルベースのフレームワークも提案されている（N3 [9], TRIPLE [10], Turtle [11] など）．これらのフレームワークは，XML 形式に比べて RDF(S) コンテンツの記述を容易に行うことが可能であるが，依然としてモデルベースのフレームワークであるため，RDFS コンテンツと RDF コンテンツの区別を行うことは困難である．

RDFS コンテンツを管理するために，セマンティック Web 登場以前から知識工学の分野ではオントロジー構築支援ツール（OntoEdit [12], Protégé [13] など）が開発されている．これらのツールは，主にオントロジーの構築およびオントロジーベースの意味記述の管理に注力している．そのため，OWL および RDFS コンテンツの構築支援は行われているが，RDF コンテンツの構築支援は十分に行われていない．

以上より，RDF(S) コンテンツの構築を支援するために様々な研究が行われているが，RDF コンテンツおよび RDFS コンテンツを同時に構築する際に生じる問題に焦点を絞った研究は行われていない．RDF コンテンツおよび RDFS コンテンツの間の意味的な対応をとらえるためには，より詳細なセマンティクスが必要となる．

そこで，本論文では，RDF コンテンツと RDFS コンテンツを分離し，両者の間の整合性を保ちながら視覚的に編集する機能を持つ RDF(S) コンテンツ構築支援ツール　MR^3 (Meta-Model Management based on RDFs Revision Reflection) を提案する．

なお，本章では，表 3.1 に示す名前空間接頭辞を用いて説明を行っている．

以降，本章の構成は次のとおりである．3.2 節では，はじめに，本研究におけるモデルとオントロジーについて説明し，次に，RDF(S) コンテンツ管理および RDF(S) コンテンツ管理のシナリオについて説明する．3.3 節では，RDF(S) コンテンツ構築支援ツールの設計および機能について述べる．3.4 節では，RDF(S) コンテンツ構築支援ツールの機能の中で特に重要な RDF(S) コンテンツ管理機能の詳細について説明する．3.5 節では，RDF(S) コンテンツ構築支援ツールの実装について述べる．3.6 節では，RDF(S) コンテンツ構築支援ツールの評価として，関連ツールとの機能比較，RDF(S) コンテンツ管理機能および RDF(S) コンテンツの視覚的編集機能の評価実験について述べる．3.7 節では，オープンソースソフトウェアとしての RDF(S) コンテンツ構築支援ツールについて述べる．最後に，3.8 節では，本章のまとめと今後の課題について述べる．

表 3.1: 名前空間接頭辞および名前空間 URI

名前空間接頭辞	**名前空間 URI**
ex	http://www.example.com/
mr3	http://www.yamaguti.comp.ae.keio.ac.jp/mmm/mr3/
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	http://www.w3.org/2000/01/rdf-schema#
owl	http://www.w3.org/2002/07/owl#

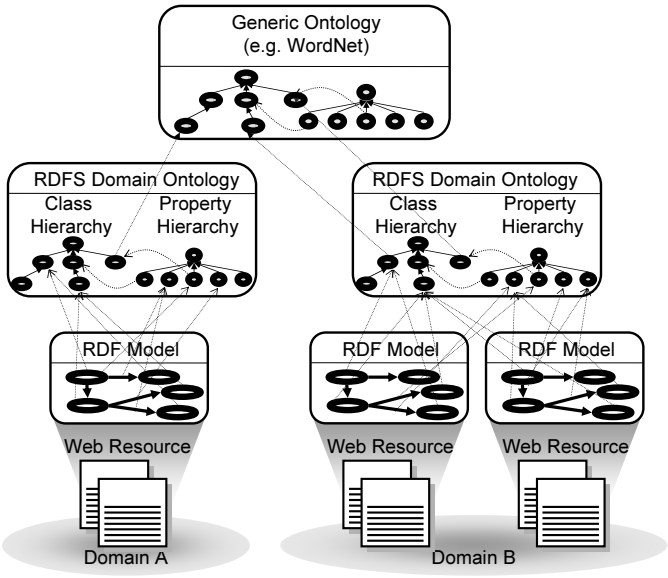


図 3.1: セマンティック Web における Web リソース，モデル，オントロジーの関係

3.2 RDF(S) コンテンツ管理

3.2.1 モデルとオントロジー

本章における，モデルとオントロジーの関係は，次のように説明することができる．

モデル

Web コンテンツから些末な情報を排除し，機械処理させたい本質的なコンテンツを RDF により記述したもの．具体的には，あるリソースがどのクラスに属するか，また，どのようなプロパティおよびプロパティ値を持つかを RDF により記述したもの．モデルは，オントロジーによって提供されるモデル構成要素によって構成される．セマンティック Web では，モデルは RDF コンテンツに相当する．また，モデルは OWL における事実 (fact) と同義である．オントロジーによって提供される，RDF におけるモデル構成要素は，RDF リソースにおける `rdf:type` プロパティの値（RDF リソースのタイプ）および RDF プロパティである．

オントロジー

クラスおよびプロパティとそれらの間の関係を定義したもの．セマンティック Web では，オントロジーは RDFS および OWL コンテンツに相当する．RDFS コンテンツでは，RDF コンテンツの構成要素である，RDF リソースのタイプを RDFS クラスとして，RDF プロパティを RDFS プロパティとして定義する．

Web リソース，モデル，オントロジーの関係を図 3.1 に示す．

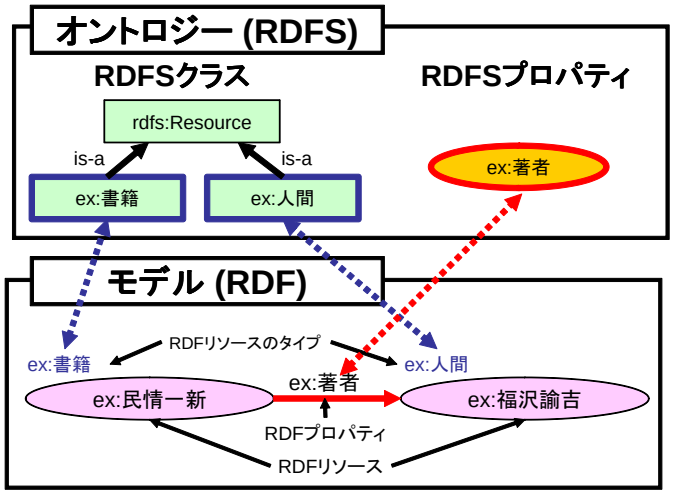


図 3.2: RDF(S) コンテンツ管理

各 Web リソースには，それらの内容をモデル化した RDF データモデルを用意する．RDF データモデルを XML などの特定の構文で記述したものが RDF コンテンツである．RDF データモデル構築のために，領域オントロジーや汎用オントロジー（WordNet [17], EDR 電子化辞書 [18] など）を参照する．図 3.1 中の `Domain B` のように，複数の Web リソースでも対象領域が同じであれば，同一の領域オントロジーを参照することが可能である．

3.2.2 RDF(S) コンテンツ管理の概念

本節では，本研究における RDF(S) コンテンツ管理の概念について述べる．

RDF および RDFS コンテンツは共に RDF/XML により記述されるため，RDF および RDFS コンテンツが混在する場合には，両者を区別することは困難である．ユーザは RDF および RDFS コンテンツを区別するために，リソースのタイプに着目する必要があり，このことはユーザの負担となる．

RDFS よりも詳細なオントロジーにおける記述（クラス公理やプロパティ公理など）が可能である，セマンティック Web のためのオントロジー記述言語 OWL の標準化についても，W3C が行っている．しかしながら，OWL も RDFS と同様に単一のフレームワーク（RDF/XML による記述）を採用しており，オントロジーとモデルの間の対応関係を管理するための仕組みは標準化されていない．

本研究では，上記の問題を解決するために，RDF(S) コンテンツ管理を提案する．RDF(S) コンテンツ管理とは，RDF コンテンツと RDFS コンテンツをモデルとオントロジーの関係として捉え，両者を明確に区別し，両者の間の整合性を（半）自動的に管理することと定義する．RDF(S) コンテンツ管理では，特に，オントロジーにおける RDFS クラスおよび RDFS プロパティと，モデルにおける RDF リソースのタイプおよび RDF プロパティを適切に管理する．

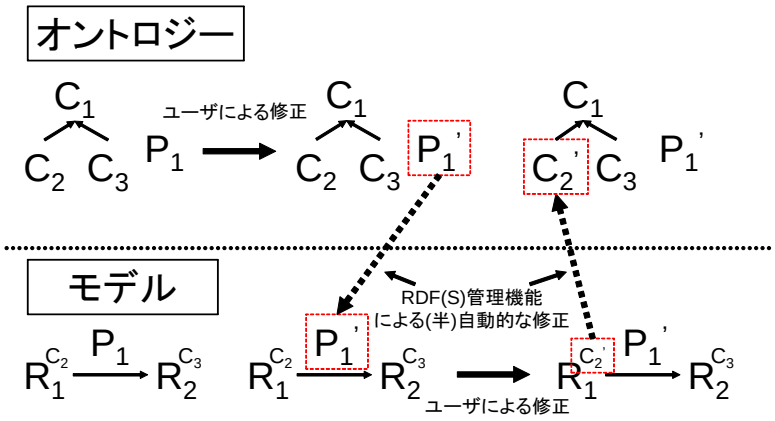


図 3.3: RDF(S) コンテンツ管理のシナリオ

図 3.2 に RDF(S) コンテンツ管理の概念図を示す.

3.2.3 RDF(S) コンテンツ管理のシナリオ

セマンティック Web 実現のためには，Web リソースに対して，オントロジー（RDFS および OWL コンテンツ）に基づいて，モデル（RDF コンテンツ）を構築する必要がある．しかし，モデル構築に適切なオントロジーが常に存在するとは限らない．その場合，モデラは，モデルおよびオントロジーを同時に構築することが考えられる．

オントロジーおよびモデルの構築は，実際には二つの異なる観点から行われることが考えられる．オントロジーは，現実世界のオブジェクトからは離れて，対象の概念化を考慮して構築が行われる．一方，モデルは現実世界のオブジェクトを考慮して構築が行われる．モデルの修正は，オントロジーの修正をもたらすことがあり，また，その逆もいえる．適切なオントロジーおよびモデルを構築するためには，モデラはモデルおよびオントロジーを相互に繰り返し修正しなければならない．そのような状況では，モデルおよびオントロジーの両方に頻繁に修正が発生するため，モデラの修正コストは大きくなる．

本研究で提案する RDF(S) コンテンツ構築支援ツールは，RDF(S) コンテンツの視覚的編集機能および RDF(S) コンテンツ管理機能により，モデルおよびオントロジーの双方向に頻繁に発生する修正を支援する．RDF(S) コンテンツの視覚的編集機能は，モデルおよびオントロジーを明確に区別し，視覚的に表示および編集を可能にする機能であり，モデラがモデルとオントロジーの関係を容易にとらえることを可能にする．RDF(S) コンテンツ管理機能は，モデラが，モデル編集モードとオントロジー編集モードを頻繁に切り替えることなく，モデルまたはオントロジーの編集に集中することを可能にする．

図 3.3 に RDF(S) コンテンツ管理のシナリオを示す．図 3.3 の上部は，オントロジー修正プロセスを示している．図 3.3 の下部はモデル修正プロセスを示している． オントロジーにおける C_1，C_2，C_3 は，RDFS クラスを表す．オントロジーにおける P_1 は，RDFS プロパティを表す．モデルにおける R_1 および R_2 は RDF リソースを表す．モデルにおけ

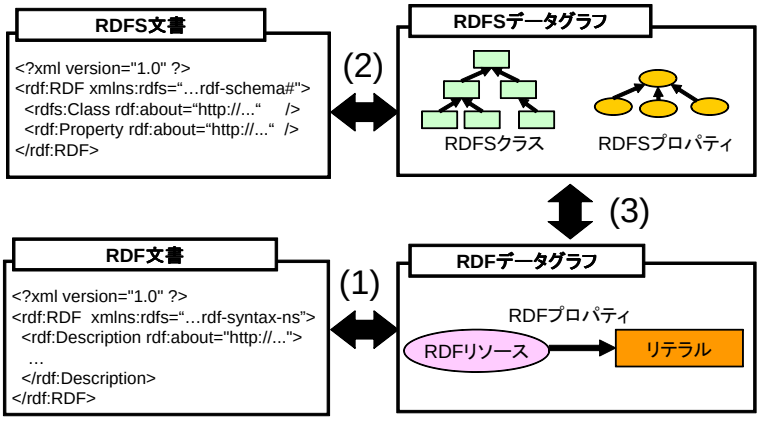


図 3.4: RDF(S) コンテンツ構築支援ツールの機能概要

る RDF リソースの右上に描いている C_2 および C_3 は，RDF リソースのタイプを表す．モデルにおける P_1 は，RDF プロパティを表す．

図 3.3 では，はじめにオントロジーにおける P_1 がユーザにより P_1' に修正される．それに伴い，RDF(S) コンテンツ管理機能により，対応するモデルにおける RDF プロパティが自動的に修正される．次に，RDF リソースのタイプであるモデルにおける C_2 がユーザにより C_2' に修正される．それに伴い，対応するオントロジーにおける RDFS クラスが RDF(S) コンテンツ管理機能により半自動的に修正される．RDF(S) コンテンツ管理機能の詳細については，3.4 節で述べる．

3.3 RDF(S) コンテンツ構築支援ツールの設計

3.3.1 RDF(S) コンテンツ構築支援ツールの機能概要

RDF(S) コンテンツ構築支援ツールの機能概要を図 3.4 に示す．RDF(S) コンテンツ構築支援ツールは，(1) RDF コンテンツの視覚的編集機能，(2) RDFS コンテンツの視覚的編集機能，(3)RDF(S) コンテンツ管理機能の主に三つの機能を持つ．

RDFS 文書は，クラスおよびプロパティの階層関係，プロパティの定義域および値域が RDF 構文（RDF/XML, Notation 3, N-Triples など）により記述された文書を表す．RDF 文書は，クラスのインスタンスに関するステートメントが RDF 構文により記述された文書を表す．RDF(S) データグラフとは，RDF(S) データモデルを視覚的に表現したものである．RDF 要素は，RDF リソース，RDF プロパティ，RDF リテラルを表す．RDFS 要素は，RDFS クラスおよび RDFS プロパティを表す．

以下では，RDF(S) コンテンツ構築支援ツールの各機能について説明する．

3.3.2 RDF コンテンツの視覚的編集機能

図 3.4 の (1) は，RDF コンテンツの視覚的編集機能を表している．RDF コンテンツの視覚的編集機能は，RDF 文書から RDF データグラフへの変換機能，RDF データグラフから RDF 文書への変換機能，RDF データモデルに基づいた RDF 要素の視覚的編集機能の三つから構成される．RDF 要素の視覚的編集機能では，ステートメント（RDF リソース，RDF プロパティ，RDF リテラルで表される三つ組み）の編集を行うことができる．

3.3.3 RDFS コンテンツの視覚的編集機能

図 3.4 の (2) は，RDFS コンテンツの視覚的編集機能を表している．RDFS コンテンツの視覚的編集機能は，RDFS 文書から RDFS データグラフへの変換機能，RDFS データグラフから RDFS 文書への変換機能，RDFS データモデルに基づいた RDFS 要素の視覚的編集機能の三つから構成される．RDFS 要素の視覚的編集機能では，クラスおよびプロパティの階層関係，プロパティの定義域および値域の編集を行うことができる．

3.3.4 RDF(S) コンテンツ管理機能

図 3.4 の (3) は，RDF(S) コンテンツ管理機能を表している．RDF(S) コンテンツ管理機能により，オントロジー (RDFS コンテンツ) とモデル (RDF コンテンツ) 間の整合性を管理することができる．

RDF(S) コンテンツ管理機能は，**O→M** または **M→O** で表すことができる．**O→M** は，オントロジー (RDFS クラスおよび RDFS プロパティ) の変更をモデル (RDF リソースのタイプおよび RDF プロパティ) に反映させる機能を表す．**M→O** は，モデルの変更をオントロジーに反映させる機能を表す．RDFS クラスの編集および RDFS プロパティの編集は，**O→M** で表される．RDF リソースタイプの変更および RDF プロパティの変更は，**M→O** で表される．

RDF(S) コンテンツ管理機能の詳細については，3.4 節で述べる．

3.3.5 その他の機能

URI 識別補助機能

URI 識別補助機能とは，RDF 要素および RDFS 要素の識別を補助するための機能である．ユーザは RDF 要素および RDFS 要素を URI によって識別する．URI は一般的に数十文字になるため，膨大な数のリソースを扱う場合，ユーザが URI を識別および編集することが困難である．そこで，RDF(S) コンテンツ構築支援ツールではユーザが指定した名前空間 URI を指定した名前空間接頭辞に置換して表示する機能をもつ．また，RDF 要素および RDFS 要素に `rdfs:label` プロパティの値（見出し）が定義されている場合，URI の代わりに見出しを表示する機能をもつ．

メタクラスの設定機能

RDF(S) コンテンツ構築支援ツールでは，ユーザはメタクラスを設定することができる．メタクラスには，クラスクラスとプロパティクラスの 2 種類がある．あるリソースのタイプがクラスクラスの場合，そのリソースはクラスとなる．また，あるリソースのタイプがプロパティクラスの場合，そのリソースはプロパティとなる．この機能により，ユーザはどのリソースをクラスまたはプロパティとして扱うかを設定することができる．例えば，ユーザが `owl:Class` をクラスクラスとして，`owl:ObjectProperty` と `owl:DatatypeProperty` をプロパティクラスとして設定した場合，RDF(S) コンテンツ構築支援ツールは OWL のクラスおよびプロパティ階層を扱うことができる．RDF(S) コンテンツ構築支援ツールでは，初期クラスクラスとして `rdfs:Class` が，初期プロパティクラスとして `rdf:Property` が設定されている．

整合性チェック機能

整合性チェック機能により，RDF(S) コンテンツ構築中の任意の時点で，RDFS プロパティの定義域および値域に違反する RDF コンテンツのステートメントをユーザに提示することができる．RDF(S) コンテンツ構築支援ツールは，RDF プロパティの定義を容易に行うことができるように[1]，RDFS プロパティの定義域および値域の整合性チェックをリアルタイムに行わない．整合性チェックには，vOWLidator [35] を用いている．

3.4 RDF(S) コンテンツ管理機能の詳細

3.4.1 O→M: RDFS クラスの編集

RDF(S) コンテンツ管理機能がはたらく RDFS クラスの編集には，RDFS クラス名 (URI) の変更と RDFS クラスの削除がある．

RDFS クラス名の変更

RDFS クラス名を変更する場合，その RDFS クラスを参照している RDF リソースのタイプ名 (URI) も同時に変更される．図 3.5 は，RDFS クラス名を変更する場合の具体例を表している．図 3.5 の左側は RDFS クラス **ex:書籍**を変更する前の状態を表しており，右側が RDFS クラス **ex:書籍**を **ex:書物**に変更した後の状態を表している．以下，RDF(S) コンテンツ管理機能の具体例を表す図中の RDFS における矩形はクラス，楕円はプロパティ，矢印はクラスの上位・下位関係を表す．矢印は，下位クラスから上位クラスに向

[1]詳しくは 3.6.1 項で述べるが，オントロジーベースのツールでは，RDFS プロパティの定義域および値域を定義しなければ，RDF コンテンツ構築時に RDF プロパティが利用できない．RDF(S) コンテンツ構築支援ツールでは，RDFS プロパティの定義域および値域の定義を行うことなく，RDF プロパティの定義を行うことができる．

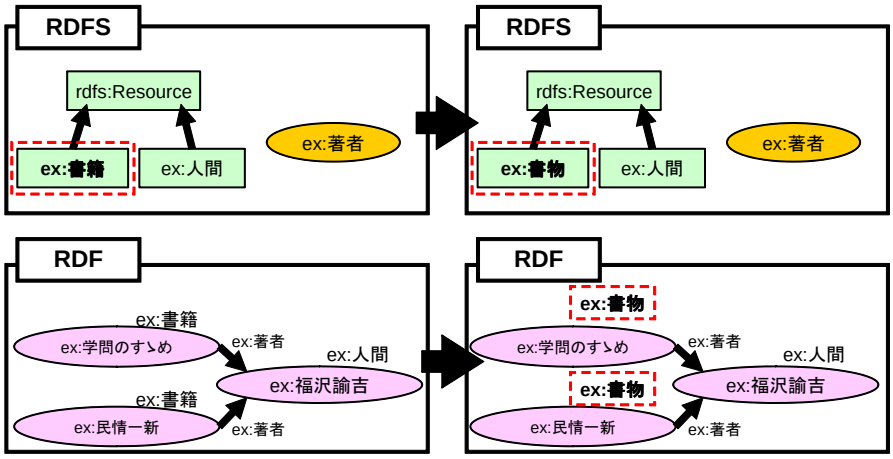


図 3.5: RDFS クラス名の変更

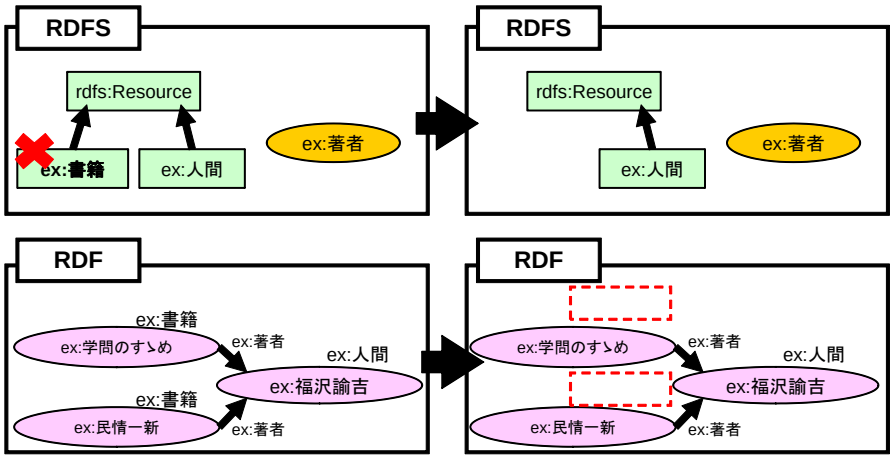


図 3.6: RDFS クラスの削除

かって伸びている．同様に，図中の RDF における楕円は RDF リソース，矢印は RDF プロパティ，楕円右上のラベルは，RDF リソースのタイプを表す．**ex:学問のすゝめ**と **ex:民情一新**は，**ex:書籍**をタイプとする RDF リソースである．RDFS クラス **ex:書籍**を **ex:書物**に変更すると，**ex:書籍**に対応する RDF リソースのタイプにも自動的に変更が反映される．つまり，RDF リソース **ex:学問のすゝめ**および **ex:民情一新**のタイプは，**ex:書物**に自動的に変更される．

RDFS クラスの削除

RDFS クラスを削除する場合，その RDFS クラスをタイプとする RDF リソースの一覧を示し，ユーザは RDF リソースのタイプとして，他の RDFS クラス（または空[2]）を選択することができる．図 3.6 は，RDFS クラスを削除する場合の具体例を表している．ユー

[2]RDF リソースのタイプが空の場合，**rdfs:Resource** クラスをタイプとして持つことになる．

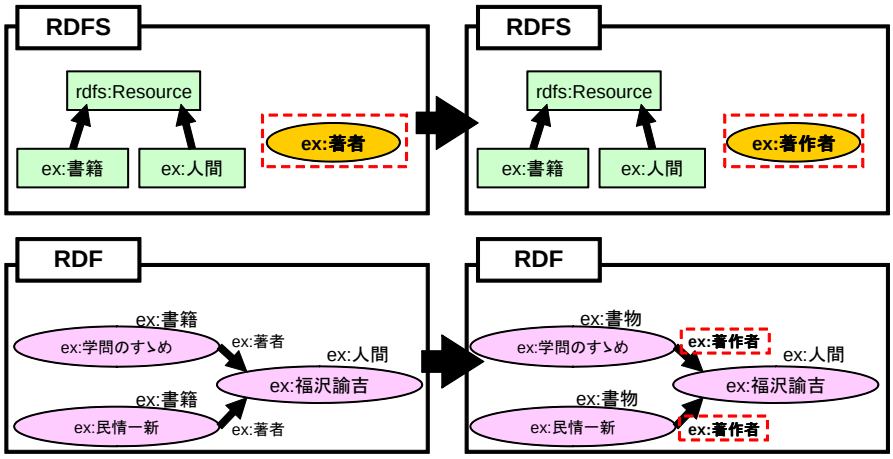


図 3.7: RDFS プロパティ名の変更

ザが RDFS クラス **ex:書籍**を削除したとする．**ex:学問のすゝめ**および **ex:民情一新**は，**ex:書籍**をタイプとする RDF リソースであるため，**ex:書籍**が削除されると RDF コンテンツと RDFS コンテンツ間の整合性を保つことができない．図 3.6 では整合性を保つために，RDF リソース **ex:学問のすゝめ**および **ex:民情一新**のタイプを空にしている．

3.4.2 O→M: RDFS プロパティの編集

RDF(S) コンテンツ管理機能がはたらく RDFS プロパティの編集には，RDFS プロパティ名 (URI) の変更と RDFS プロパティの削除がある．

RDFS プロパティ名の変更

RDFS プロパティ名を変更する場合，その RDFS プロパティを参照している RDF プロパティ名 (URI) も同時に自動的に変更される．図 3.7 は，RDFS プロパティ名を変更する場合の具体例を表している．図 3.7 の左側は RDFS プロパティ **ex:著者**を変更する前の状態を表しており，右側は RDFS プロパティ **ex:著者**を **ex:著作者**に変更した後の状態を表している．**ex:学問のすゝめ**および **ex:民情一新**は，**ex:著者**プロパティをもつ RDF リソースである．ユーザが RDFS プロパティ **ex:著者**を **ex:著作者**に変更すると，**ex:著者**と対応する RDF プロパティにも変更が反映される．つまり，RDF リソース **ex:学問のすゝめ**および **ex:民情一新**がもつプロパティ **ex:著者**は **ex:著作者**に自動的に変更される．

RDFS プロパティの削除

RDFS プロパティを削除する場合，その RDFS プロパティを参照している RDF プロパティの一覧を示し，ユーザは RDF プロパティとして，他の RDFS プロパティ（あるいは

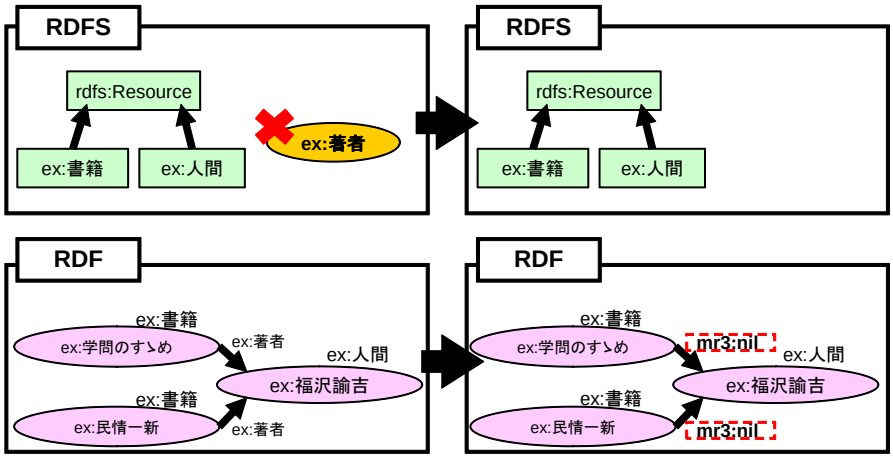


図 3.8: RDFS プロパティの削除

初期のプロパティ[3]）を選択することができる．図 3.8 は，RDFS プロパティを削除する場合の具体例を表している．ユーザが RDFS プロパティ**ex:著者**を削除したとする．**ex:学問のすゝめ**と**ex:民情一新**は，**ex:著者**をプロパティとしてもつ RDF リソースであるため，**ex:著者**が削除されると RDF コンテンツと RDFS コンテンツ間の整合性を保つことができない．図 3.8 では，整合性を保つためにユーザは，RDF リソース**ex:学問のすゝめ**および**ex:民情一新**がもつ**ex:著者**プロパティを初期のプロパティ (`mr3:nil`) にしている．

3.4.3 M→O: RDF リソースのタイプの変更

ユーザが変更した RDF リソースのタイプが RDFS クラスで定義されている場合は，RDF(S) コンテンツ構築支援ツールは，RDF リソースのタイプとそれに対応する RDFS クラスを対応づける．ユーザが変更した RDF リソースのタイプに対応する RDFS クラスが定義されていない場合，ユーザは次の二つのどちらかを選択できる．一つは，ユーザは RDF リソースのタイプを変更する前に参照していた RDFS クラス名を変更することができる．もう一つは，ユーザは，定義されていない RDFS クラスを新規に作成することができる．ユーザが変更する前の RDF リソースのタイプが空の場合には，RDFS クラス名の変更は選択できない．図 3.9 および図 3.10 は，それぞれ，RDF リソースのタイプを変更した場合の具体例を表している．

参照している RDFS クラス名の変更

図 3.9 では，RDFS クラス**ex:書物**が定義されていない状態で，ユーザが RDF リソース**ex:民情一新**のタイプ**ex:書籍**を**ex:書物**に変更している．ここでは，整合性を保つために

[3]本論文で提案する RDF(S) コンテンツ構築支援ツールでは，初期のプロパティを mr3:nil としている．RDFS プロパティが定義されていない状態で RDF リソース間の関係を定義した場合，その RDF リソース間の関係は初期のプロパティとなる．

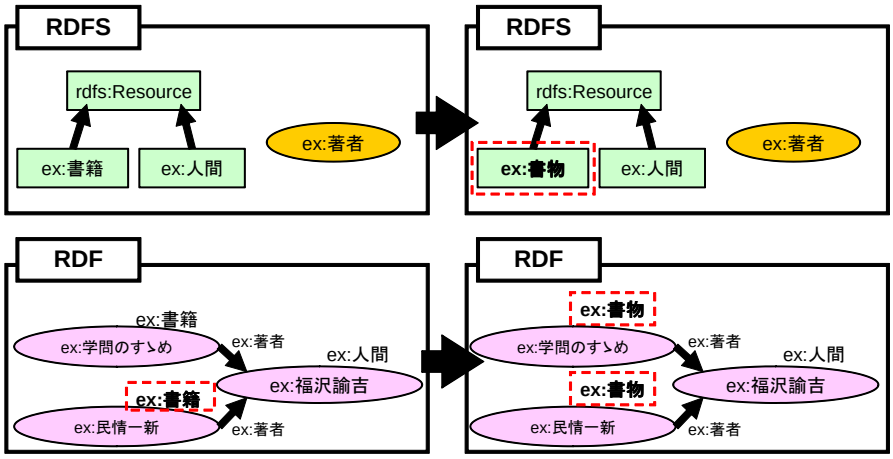


図 3.9: 参照している RDFS クラス名の変更

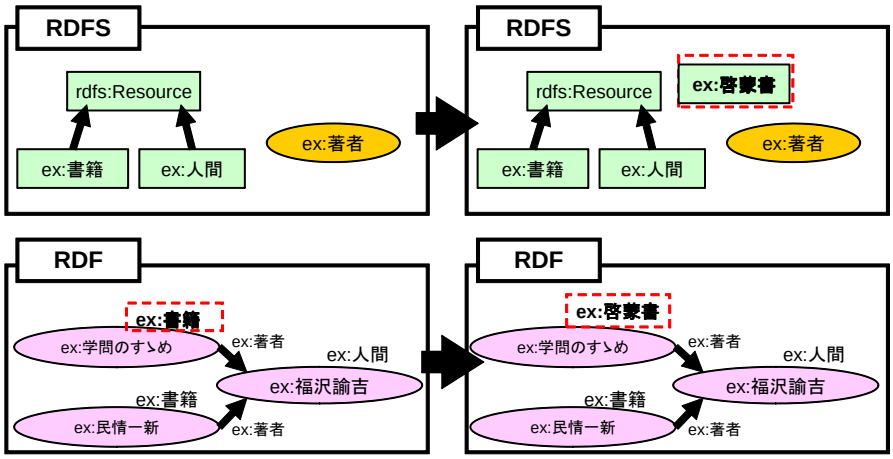


図 3.10: RDFS クラスの新規作成

RDFS クラス名の変更をユーザが選択している．RDFS クラス **ex:書籍**を **ex:書物**に変更することにより，**ex:学問のすゝめ**のタイプも **ex:書物**に半自動的に変更される．

RDFS クラスの新規作成

図 3.10 では，RDFS クラス **ex:啓蒙書**が定義されていない状態で，ユーザが RDF リソース **ex:学問のすゝめ**のタイプ **ex:書籍**を **ex:啓蒙書**に変更している．ここでは，整合性を保つために RDFS クラスの新規作成をユーザが選択している．**ex:啓蒙書**クラスを新規に作成することによって，RDF コンテンツと RDFS コンテンツ間の整合性が保たれる．RDFS クラス **ex:啓蒙書**は，**rdfs:Resource** のサブクラスとなる[4]．

[4] **rdfs:subClassOf** プロパティが定義されていないクラスは，暗黙的に **rdfs:Resource** のサブクラスとなる．

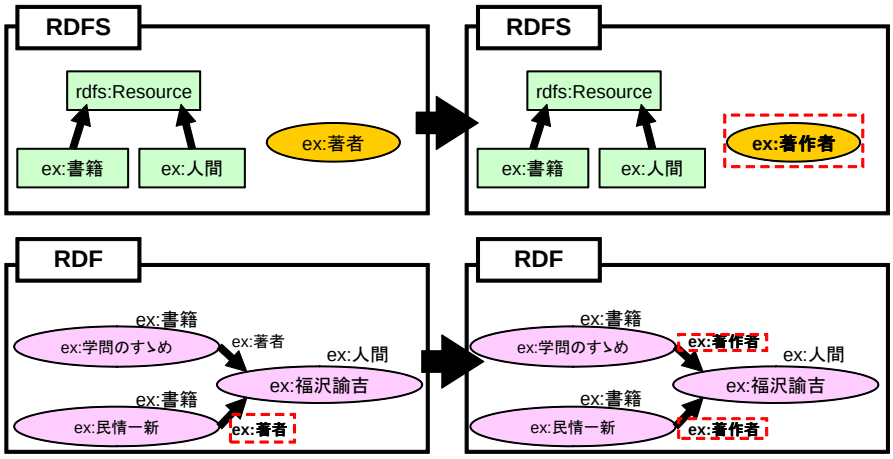


図 3.11: 参照している RDFS プロパティ名の変更

3.4.4 M→O: RDF プロパティの変更

ユーザが変更した RDF プロパティが RDFS プロパティで定義されている場合は，RDF(S) コンテンツ構築支援ツールは，RDF プロパティとそれに対応する RDFS プロパティを対応づける．ユーザが変更した RDF プロパティに対応する RDFS プロパティが定義されていない場合，ユーザは次の二つのどちらかを選択できる．一つは，ユーザは RDF プロパティを変更する前に参照していた RDFS プロパティ名を変更することができる．もう一つは，ユーザは，定義されていない RDFS プロパティを新規に作成することができる．図 3.11 および図 3.12 は，それぞれ，RDF プロパティを変更した場合の具体例を表している．

参照している RDFS プロパティ名の変更

図 3.11 では，RDFS プロパティ **ex:著作者**が定義されていない状態で，ユーザが RDF リソース **ex:民情一新**が持つプロパティ **ex:著者**を **ex:著作者**に変更している．ここでは，整合性を保つために RDFS プロパティ名の変更をユーザが選択している．RDFS プロパティ **ex:著者**を **ex:著作者**に変更することにより，**ex:学問のすゝめ**が持つプロパティ **ex:著者**も **ex:著作者**に半自動的に変更される．

RDFS プロパティの新規作成

図 3.12 では，RDFS プロパティ **ex:著作者**が定義されていない状態で，ユーザが RDF リソース **ex:民情一新**が持つプロパティ **ex:著者**を **ex:著作者**に変更している．ここでは，整合性を保つために RDFS プロパティの新規作成をユーザが選択している．RDFS プロパティ **ex:著作者**を新規に作成することによって，整合性を保つことができる．

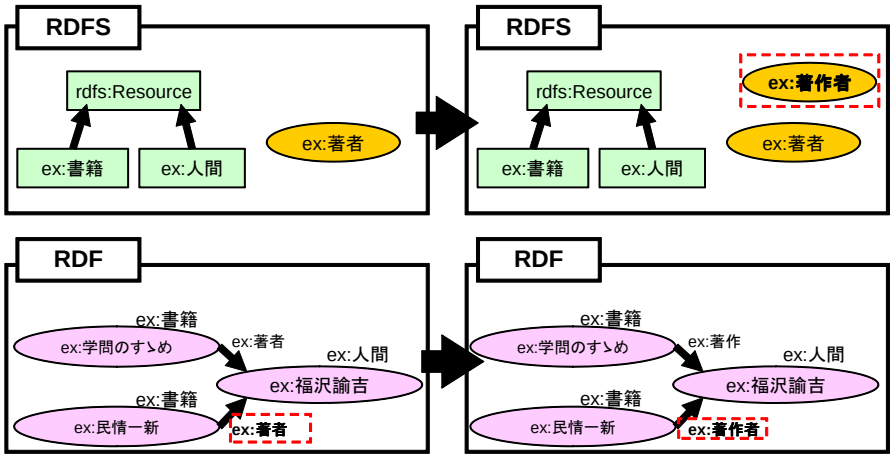


図 3.12: RDFS プロパティの新規作成

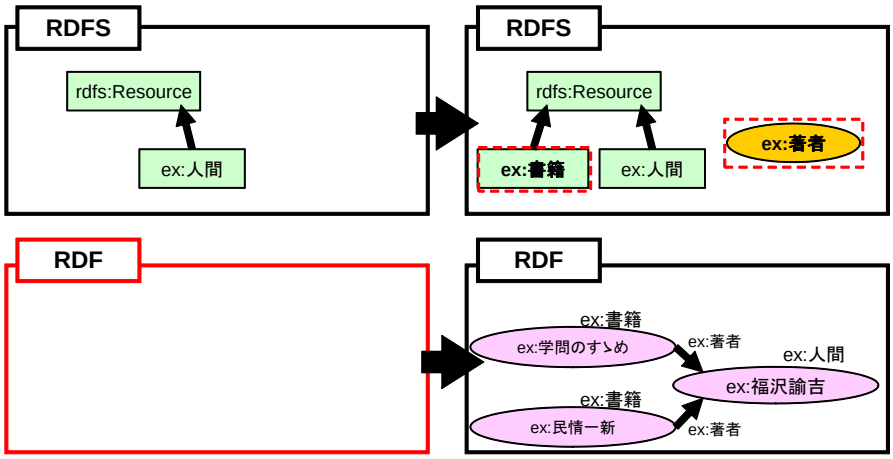


図 3.13: RDF 文書のインポート

3.4.5 RDF 文書のインポート

RDF 文書を RDF(S) コンテンツ構築支援ツールにインポートする場合，RDF リソースのタイプおよび RDF プロパティが，RDFS クラスおよび RDFS プロパティとして定義されていない場合がある．その場合，整合性を保つために RDFS クラスとして定義されていない RDF リソースのタイプは，**rdfs:Resource** クラスのサブクラスとして新規作成する．同様に，RDFS プロパティとして定義されていない RDF プロパティについても新規作成する．

図 3.13 の左側は，RDF 文書をインポートする前の状態を表しており，右側は RDF 文書をインポートした後の状態を表している．RDF リソース **ex:学問のすゝめ**と **ex:民情一新**のタイプ **ex:書籍**は，RDFS クラスとして定義されていない．RDF プロパティ**ex:著者**も同様に，RDFS プロパティとして定義されてない．整合性を保つために，RDFS クラス **ex:書籍**と RDFS プロパティ**ex:著者**は，インポート時に自動的に作成される．

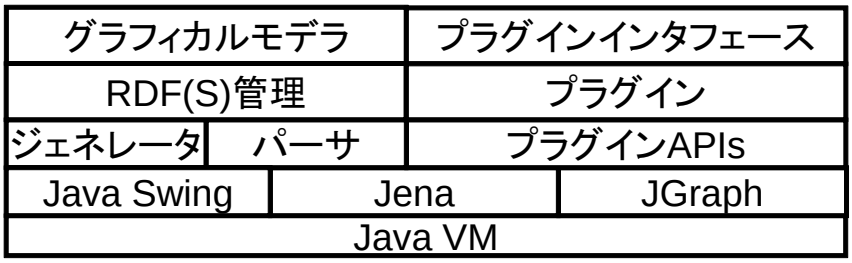


図 3.14: MR^3 の実装アーキテクチャ

3.4.6 要素名の一意性確保

要素名の一意性確保は，RDF 要素および RDFS 要素の名前変更および新規作成を行う際に，他の要素名と重複しないようにするための機能である．RDFS 要素名の重複を許すと，RDF リソースのタイプおよび RDF プロパティに対応する RDFS 要素が複数存在する場合が生じるため，整合性を保つことができない．RDF コンテンツと RDFS コンテンツ間の整合性を保つために，RDF(S) コンテンツ構築支援ツールでは，要素名の一意性を確保する．

3.5 RDF(S) コンテンツ構築支援ツールの実装

3.5.1 実装の概要

3.3 節および 3.4 節で述べた RDF(S) コンテンツ構築支援ツールの設計を基に，RDF(S) コンテンツ構築支援ツール MR^3 (Meta-Model Management based on RDFs Revision Reflection) の実装を行った．図 3.14 に MR^3 の実装アーキテクチャを示す． MR^3 は GUI コンポーネントとして Java Swing を用いて，Java 言語で実装を行った． MR^3 は，RDF および RDFS データグラフを視覚化するために JGraph [36] ライブラリを用いている．図 3.14 のグラフィカルモデラの実装には JGraph を用いている．また， MR^3 は，セマンティック Web コンテンツ（RDF，RDFS，OWL コンテンツ）のインポート，編集，エクスポートを行うために，Jena: A Semantic Web Framework [37] を用いている．図 3.14 のパーサとジェネレータの実装には Jena API を用いている．他のセマンティック Web アプリケーションと連携を行うことができるように， MR^3 はプラグイン機構を備えている．MR^3 はプラグイン開発のために主に 2 種類の API を提供している．一つは， MR^3 のデータグラフを Jena のモデルオブジェクトに変換する API であり，もう一つは，Jena のモデルオブジェクトを MR^3 のデータグラフに変換する API である．これらの API を用いることで， MR^3 で構築した RDF および RDFS コンテンツを別のセマンティック Web アプリケーションに入力することや，逆に他のセマンティック Web アプリケーションの出力を MR^3 に入力することが可能となる．将来的には，RDF コンテンツと RDFS コンテンツ間の整合性管理に関する API も提供することを考えている．

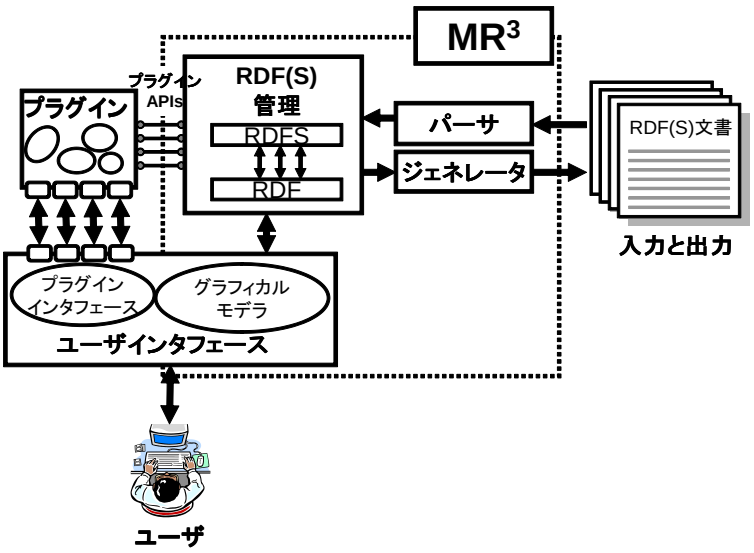


図 3.15: MR^3 のシステム構成

図 3.15 に MR^3 のシステム構成を示す． MR^3 は，パーサモジュール，ジェネレータモジュール，RDF(S) コンテンツ管理モジュール，プラグイン，ユーザインタフェースから構成される．ユーザインタフェースには，プラグインインタフェースおよびグラフィカルモデラがあり，ユーザはグラフィカルモデラを介して，RDF および RDFS コンテンツを視覚的に編集することができる．また，ユーザは，プラグインインタフェースより，プラグインの機能にアクセスすることができる． MR^3 の入力と出力は RDF(S) 文書である．パーサは RDF(S) 文書を解析して，Jena のモデルオブジェクトに変換し，その後，RDF(S) コンテンツ管理を行うことができるように MR^3 の内部データに変換する．ジェネレータは， MR^3 の内部データを Jena のモデルオブジェクトに変換し，その後，RDF(S) 文書に変換する．

表 3.2 に MR^3 の各モジュールのソースコードの行数を示す[5]．

図 3.16 に MR^3 のグラフィカルモデラのスクリーンショットを示す．グラフィカルモデ

表 3.2: MR^3 のソースコードの規模

モジュール名	行数
パーサ	826
ジェネレータ	239
RDF(S) コンテンツ管理	4378
グラフィカルモデラ	9540
その他	3607
合計	18590

[5]ソースコードの行数は，Eclipse Metrics Plugin 1.3.6 を用いて数えた．
URL: http://metrics.sourceforge.net/

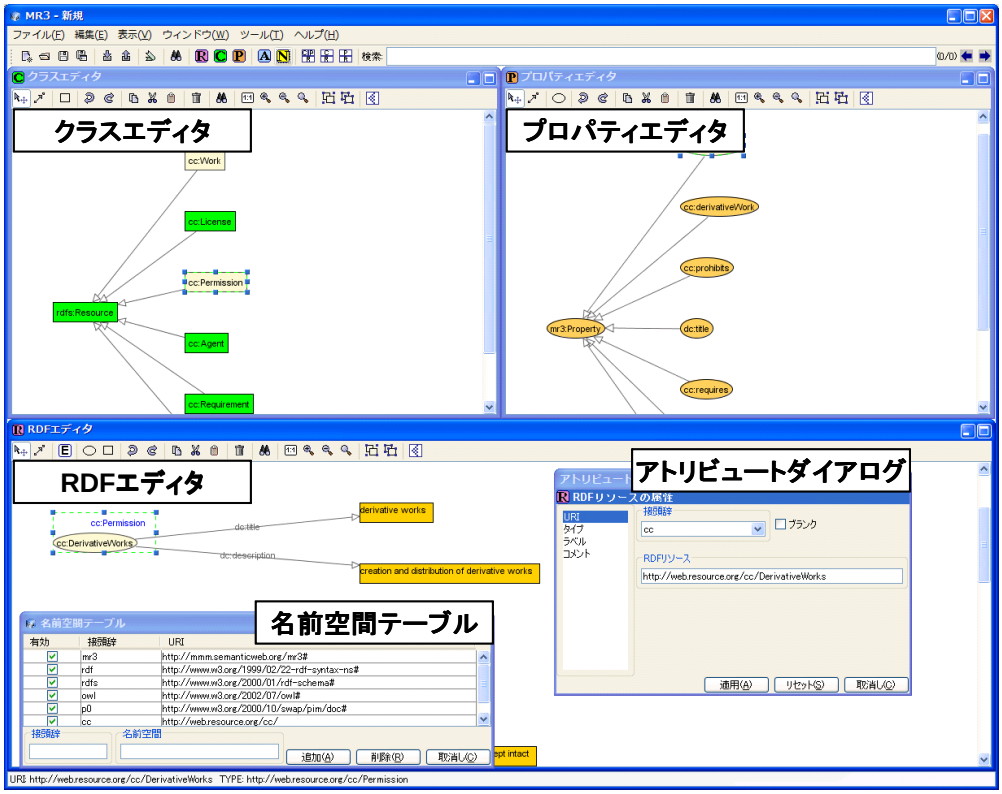


図 3.16: MR^3 のグラフィカルモデラのスクリーンショット

ラは，RDF エディタ，クラスエディタ，プロパティエディタ，アトリビュートダイアログ，名前空間テーブル，削除ダイアログ，リソース検索ダイアログ，インポートダイアログ，エクスポートダイアログ，設定ダイアログから構成される．

以下では，各エディタ，テーブル，ダイアログの実装の詳細について説明する．

3.5.2 RDF エディタの実装

RDF エディタでは，RDF リソース，RDF リソースのタイプ，RDF プロパティ，RDF リテラルの編集を行う．図 3.17 に示すように，RDF リソースは楕円，RDF プロパティは矢印，RDF リテラルは矩形で RDF エディタ内に表示される．RDF リソースのタイプは

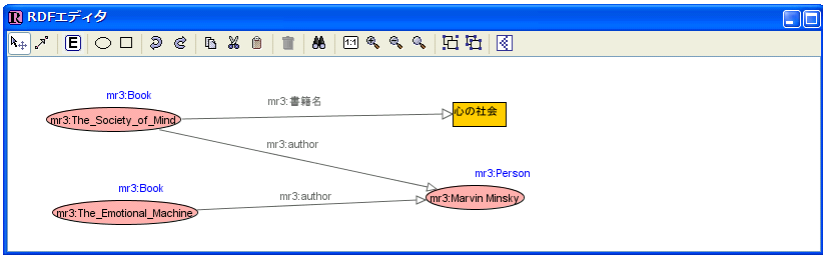


図 3.17: RDF エディタ

図 3.18: アトリビュートダイアログ（RDF リソースの URI）

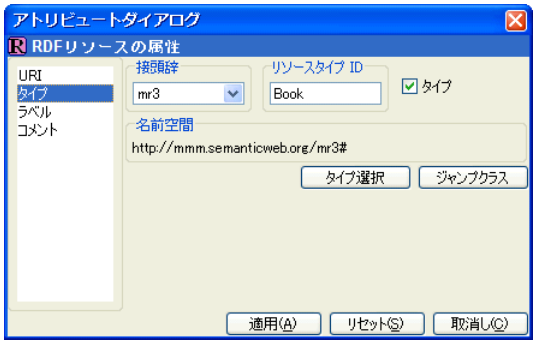


図 3.19: アトリビュートダイアログ（RDF リソースのタイプ）

RDF リソースの右上に表示される．

RDF リソースの属性編集

RDF エディタ内の RDF リソースを選択すると，アトリビュートダイアログに RDF リソースの属性が表示される（図 3.18 から図 3.23）．RDF リソースの属性編集では，RDF リソースの URI，RDF リソースのタイプ，RDF リソースが空白ノード (blank node) かどうかの選択，RDF リソースのラベルおよびコメントを編集することができる．ここで，空白ノードとは，URI により名前付けしない RDF リソースを表す．空白ノードは外部から参照することはできない．しかし，ステートメントにおける主語または目的語になることができ，URI により名前付けしにくい RDF リソースを記述したり，複数の RDF プロパティを構造化した RDF コンテンツを表現するために用いられる．

RDF リソースの URI を設定したい場合には，図 3.18 に示すように，アトリビュートダイアログ左側の一覧から「URI」を選択する．「接頭辞」コンボボックスには，3.5.5 項で述べる，名前空間テーブルに登録されている名前空間接頭辞の一覧が表示される．「接頭辞」コンボボックスから名前空間接頭辞を選択すると，対応する名前空間が「RDF リソース」テキストフィールドに出力される．ユーザは，「RDF リソース」テキストフィールドに設定したい RDF リソースの URI を入力する．RDF リソースを空白ノードとしたい場合に

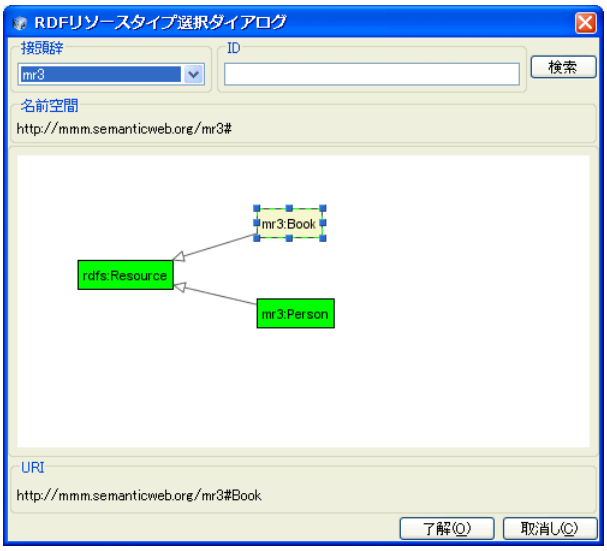


図 3.20: RDF リソースタイプ選択ダイアログ

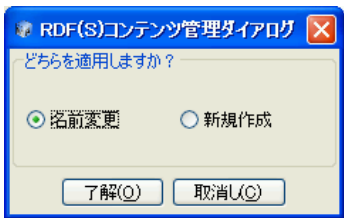


図 3.21: RDF(S) コンテンツ管理ダイアログ

は,「ブランク」チェックボックスにチェックをいれる.

RDF リソースのタイプを設定したい場合には，図 3.19 に示すように，アトリビュートダイアログ左側の一覧から「タイプ」を選択する．RDF リソースタイプを入力するには,「タイプ」チェックボックスをチェックする．RDF リソースのタイプを空にするには,「タイプ」チェックボックスのチェックをはずす.「タイプ選択」ボタンをクリックすると，図 3.20 に示す「RDF リソースタイプ選択」ダイアログが表示される.「RDF リソースタイプ選択」ダイアログには，クラスエディタで構築したクラス階層が表示される．RDF リソースのタイプとしたいクラスを選択すると,「RDF リソースタイプ選択」ダイアログの URI ラベルに，選択したクラスの URI が表示される.「了解」ボタンをクリックすると，アトリビュートダイアログの「接頭辞」コンボボックスと「リソースタイプ ID」テキストフィールドに,「RDF リソースタイプ選択」ダイアログで選択した URI が設定される．クラスエディタで定義されていないクラスの URI を RDF リソースのタイプとして入力した場合，RDF(S) コンテンツ管理機能が働き，図 3.21 に示す「RDF(S) コンテンツ管理」ダイアログが表示される．RDF(S) コンテンツ管理ダイアログでは，RDFS クラス名の変更または RDFS クラスの新規作成のどちらかをユーザは選択することができる.「ジャンプクラス」ボタンをクリックすると，その RDF リソースのタイプと対応するクラスエディ

図 3.22: アトリビュートダイアログ（RDF リソースのラベル）

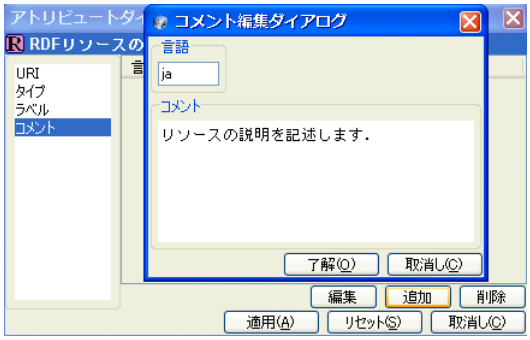


図 3.23: アトリビュートダイアログ（RDF リソースのコメント）

タ内の RDFS クラスへジャンプする．また，アトリビュートダイアログにその RDFS クラスの属性が表示される．

RDF リソースに対して，`rdfs:label` プロパティを用いて RDF リソースのラベルを記述したい場合には，図 3.22 に示すように，アトリビュートダイアログ左側の一覧から「ラベル」を選択する.「言語」テキストフィールドに言語を,「ラベル」テキストフィールドにラベルを入力し,「追加」ボタンを押すと，アトリビュートダイアログ中央のテーブルに言語とラベルが追加される．テーブルの行を選択し,「削除」ボタンを押すと選択したラベルを削除することができる．

RDF リソースに対して，`rdfs:comment` プロパティを用いて RDF リソースのコメントを記述したい場合には，図 3.23 に示すように，アトリビュートダイアログ左側の一覧から「コメント」を選択する.「追加」ボタンを押すと,「コメント編集」ダイアログが表示される.「言語」テキストフィールドに言語を,「ラベル」テキストフィールドにラベルを入力し,「了解」ボタンを押すと，アトリビュートダイアログ中央のテーブルに言語とコメントが追加される．テーブルの行を選択し,「編集」ボタンを押すと「コメント編集」ダイアログが表示され，コメントを編集することができる．テーブルの行を選択し,「削除」ボタンを押すと選択したコメントを削除することができる．

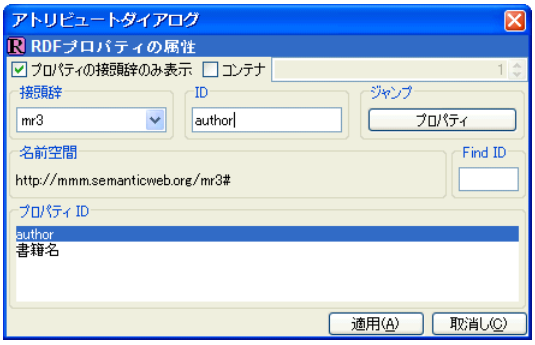


図 3.24: アトリビュートダイアログ（RDF プロパティ）

RDF プロパティの属性編集

RDF エディタ内の RDF プロパティを選択すると，アトリビュートダイアログに RDF プロパティの属性が表示される（図 3.24）．RDF プロパティの属性編集では，RDF プロパティの URI を編集することができる.「コンテナ」チェックボックスにチェックをして，数値を入力すると rdf:_1...rdf:_n プロパティの入力を行うことができる.「プロパティの接頭辞のみ表示」チェックボックスにチェックをいれると，プロパティエディタで定義されているプロパティがもつ名前空間 URI に対応する名前空間接頭辞のみを「接頭辞」コンボボックスから選択可能となる．チェックをはずすと，名前空間テーブルに登録されているすべての接頭辞を選択可能となる．プロパティエディタで定義されていないプロパティの URI を入力した場合，RDF(S) コンテンツ管理機能が働き，図 3.21 に示す「RDF(S) コンテンツ管理」ダイアログが表示される．RDF(S) コンテンツ管理ダイアログでは，RDFS プロパティ名の変更または RDFS プロパティの新規作成のどちらかをユーザは選択することができる．プロパティエディタ内で定義されているプロパティの中で，名前空間接頭辞に対応する名前空間 URI をもつプロパティの ID が「プロパティID」リストに表示される.「プロパティ」ボタンをクリックすると，選択した RDF プロパティに対応するプロパティエディタ内の RDFS プロパティにジャンプし，アトリビュートダイアログにその RDFS プロパティの属性が表示される．

RDF リテラルの属性編集

RDF エディタ内の RDF リテラルを選択すると，アトリビュートダイアログに RDF リテラルの属性が表示される（図 3.25）．RDF リテラルの編集では，リテラルの内容，言語属性（xml:lang 属性），データタイプの編集を行うことができる.「リテラル」テキストエリアには，リテラルの内容を入力する.「言語」テキストフィールドには，リテラルの記述言語を入力する．リテラルのデータタイプを設定したい場合には,「タイプ」チェックボックスにチェックを入れて,「タイプ」コンボボックスからデータタイプを選択することができる．リテラルのデータタイプを必要としない場合には,「タイプ」チェックボックス

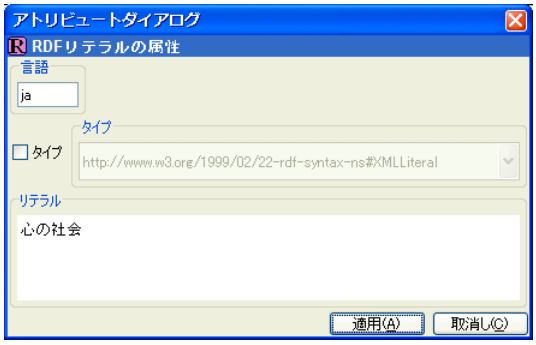


図 3.25: アトリビュートダイアログ（RDF リテラル）

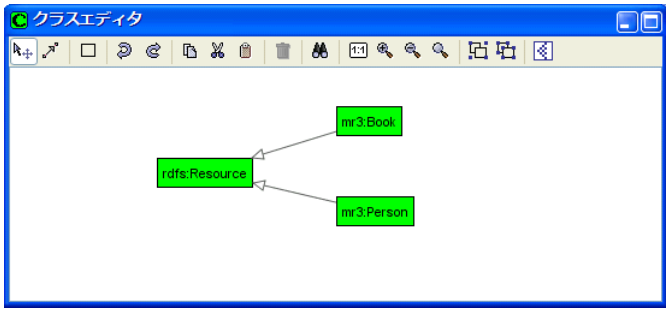


図 3.26: クラスエディタ

のチェックをはずす．言語とリテラルのタイプは排他的であり，どちらか一方しか設定することはできない．

3.5.3 クラスエディタの実装

クラスエディタでは，RDFS クラスの階層関係および RDFS クラスの属性編集を行うことができる．図 3.26 にクラスエディタのスクリーンショットを示す．

クラスエディタの属性編集

クラスエディタ内の RDFS クラスを選択すると，RDFS クラスの属性がアトリビュートダイアログに表示される（図 3.27，図 3.28，図 3.29）．アトリビュートダイアログ左側のリストから「基本」,「ラベル」,「コメント」,「インスタンス」,「上位クラス」項目を選択し，RDFS クラスの属性の編集を行う．「基本」では，リソースタイプおよび URI を編集することができる（図 3.27）．リソースタイプで選択できる項目は，3.5.10 項で述べる「設定」ダイアログのクラスクラスリストから追加および削除を行うことができる．「ラベル」では，`rdfs:label` プロパティの編集を行うことができる．「コメント」では，`rdfs:comment` の編集を行うことができる．ラベルとコメントの編集方法は，RDF リソースと同様である．「インスタンス」には選択した RDFS クラスのインスタンスのリストが表示される（図

図 3.27: アトリビュートダイアログ（RDFS クラスの基本）

図 3.28: アトリビュートダイアログ（RDFS クラスのインスタンス）

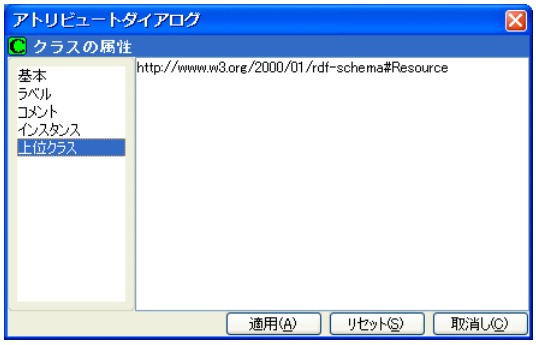


図 3.29: アトリビュートダイアログ（RDFS クラスの上位クラス）

3.28）．リスト中の項目をクリックすると対応する RDF エディタ内の RDF リソースにジャンプし，その RDF リソースの属性がアトリビュートダイアログに表示される．「上位クラス」には，選択した RDFS クラスの上位クラスのリストが表示される（図 3.29）．

3.5.4 プロパティエディタの実装

プロパティエディタでは，RDFS プロパティの階層関係および RDFS プロパティの属性編集を行うことができる．図 3.30 にプロパティエディタのスクリーンショットを示す．

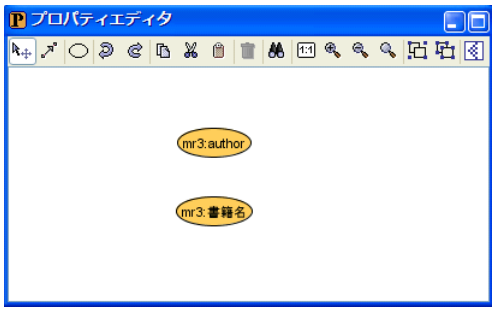


図 3.30: プロパティエディタ

図 3.31: アトリビュートダイアログ（RDFS プロパティの範囲）

プロパティエディタの属性編集

プロパティエディタ内の RDFS プロパティを選択すると，RDFS プロパティの属性がアトリビュートダイアログに表示される（図 3.31，図 3.32，図 3.33）．アトリビュートダイアログ左側のリストには,「基本」,「ラベル」,「コメント」,「範囲」,「インスタンス」,「上位プロパティ」項目が表示される.「基本」,「ラベル」,「コメント」の編集項目は RDFS クラスと同様である.「基本」のリソースタイプは,「設定」ダイアログのプロパティクラスリストから追加および削除を行うことができる.「範囲」には RDFS プロパティの定義域および値域をクラスエディタから選択できる（図 3.31）.「インスタンス」には選択した

図 3.32: アトリビュートダイアログ（RDFS プロパティのインスタンス）

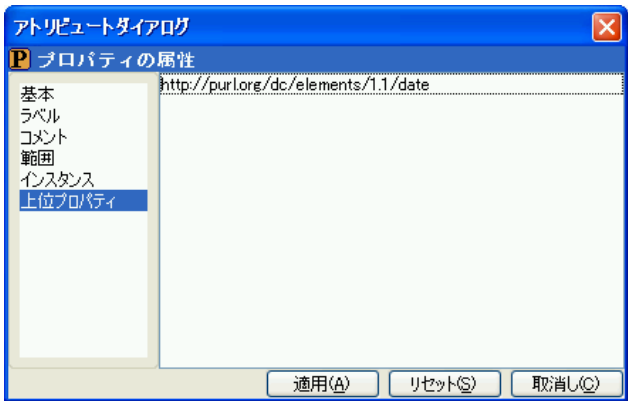


図 3.33: アトリビュートダイアログ（RDFS プロパティの上位プロパティ）

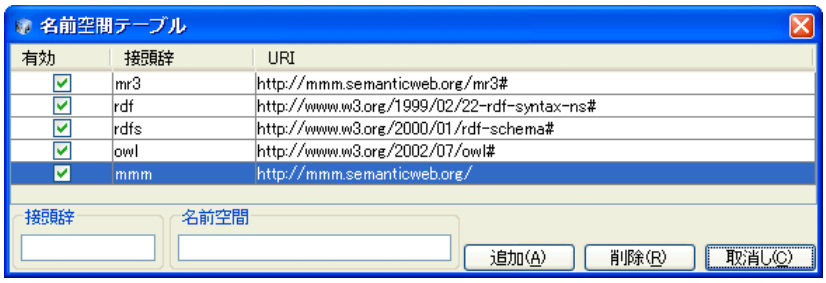


図 3.34: 名前空間テーブル

RDFS プロパティを有する RDF リソースのリストが表示される（図 3.32）．リスト中の項目をクリックした際の動作は，RDFS クラスの場合と同様である．「上位プロパティ」には選択した RDFS プロパティの上位プロパティが表示される（図 3.33）．

3.5.5 名前空間テーブルの実装

名前空間テーブルでは，名前空間 URI を名前空間接頭辞に置換して表示するために，名前空間接頭辞と名前空間 URI の管理を行う．図 3.34 に名前空間テーブルのスクリーンショットを示す．「接頭辞」テキストフィールドに名前空間接頭辞を，「名前空間」テキストフィールドに名前空間 URI を入力して，「追加」ボタンを押すと名前空間接頭辞と名前空間 URI の対応がテーブルに追加される．対応を削除したい場合には，削除したいテーブルの行を選択し，「削除」ボタンを押す．「有効」チェックボックスをチェックすると，URI 表示にしている場合に名前空間 URI が名前空間接頭辞で置換される．「有効」チェックボックスのチェックをはずすと名前空間接頭辞が名前空間 URI に置換される．

RDF(S) 要素の URI を入力する際に名前空間テーブルが参照される．RDF(S) 要素の URI を入力するためには，名前空間接頭辞と ID（ローカル名）を入力する必要がある．名前空間テーブルに登録されている名前空間接頭辞をコンボボックスから選択することができる．入力したい URI の名前空間 URI および，その名前空間接頭辞をあらかじめ名前空

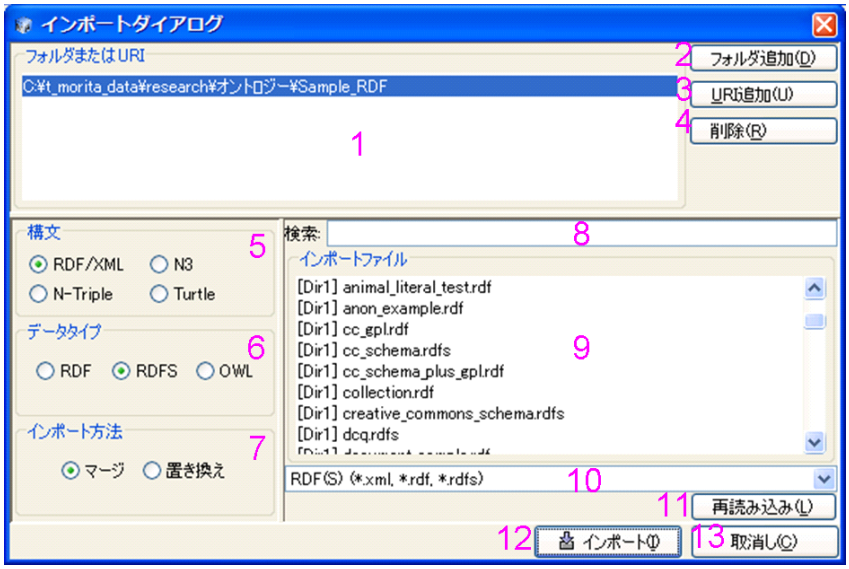


図 3.35: インポートダイアログ

間テーブルに登録する必要がある．名前空間接頭辞を選択すると対応する名前空間 URI が「名前空間」ラベルに表示される．

3.5.6 インポートダイアログの実装

インポートダイアログでは，RDF/XML, N3, N-Triple, Turtle 形式で記述された RDF(S) 文書を MR^3 にインポートすることができる．図 3.35 にインポートダイアログのスクリーンショットを示す．また，以下ではインポートダイアログの各部分（図 3.35 内の 1 から 13）について説明する．

1. コンテナリスト

RDF，N3, N-Triple，Turtle 形式で記述された RDF(S) 文書が保存されたフォルダまたは URI のリストを表示する．

2. フォルダ追加

「1. コンテナリスト」にフォルダを追加する．

3. URI 追加

「1. コンテナリスト」に URI を追加する．

4. 削除

「1. コンテナリスト」内の選択したフォルダまたは URI を削除する．

5. 構文

インポートしたい RDF(S) 文書の構文 (RDF/XML，N3，N-Triple，Turtle) を選択する．

6. データタイプ
インポートしたい文書のタイプ (RDF, RDFS, OWL) を選択する.

7. インポート方法
編集中の RDF(S) コンテンツにインポートしたい RDF(S) 文書の内容をマージするか，または，編集中の RDF(S) コンテンツをインポートしたい RDF(S) 文書の内容に置換するかを選択する.

8. 検索
「検索」テキストフィールドに入力したキーワードによって,「9. インポートファイル」内に表示されているファイルを絞り込む.

9. インポートファイル
「1. コンテナリスト」内で選択されたフォルダ内のファイル一覧を表示する.

10. 拡張子
拡張子によって,「9. インポートファイル」内に表示されているファイルを絞り込む.

11. 再読み込み
「1. コンテナリスト」で選択されているフォルダの内容を再読込し，最新のファイル一覧を「9. インポートファイル」に表示する.

12. インポート
設定した条件（構文，データタイプ，インポート方法，インポートファイル）で RDF(S) 文書を MR^3 にインポートする.

13. 取消し
インポートを中止し，インポートダイアログを閉じる.

3.5.7 エクスポートダイアログの実装

エクスポートダイアログでは， MR^3 で構築した RDF(S) データグラフを RDF/XML, N3, N-Triple, Turtle 形式で RDF(S) 文書にエクスポートすることができる．図 3.36 にエクスポートダイアログのスクリーンショットを示す．また，以下ではエクスポートダイアログの各部分（図 3.36 内の 1 から 8）について説明する.

1. 構文
RDF(S) 文書としてエクスポートしたい構文（RDF/XML，N-Triple，Turtle，N3 など）を選択する.

2. データタイプ
各チェックボックスでチェックされたエディタ内のデータグラフを RDF(S) 文書にエクスポートする.（例：クラスとプロパティにチェックをつけた場合，クラスエディタとプロパティエディタ内のデータグラフを RDF(S) 文書にエクスポートする.）

図 3.36: エクスポートダイアログ

3. オプション

「Encode(UTF-8)」チェックボックスにチェックをいれた場合，エクスポート時に URI の規則を定める RFC3986 に基づいて，各要素の URI 文字列に含まれる非予約文字データを URL エンコードする.「Selected」チェックボックスにチェックをいれた場合，範囲選択されているサブデータグラフをエクスポートする.「Abbrev」チェックボックスにチェックをいれた場合，省略表記方法でエクスポートする.「XMLbase」チェックボックスにチェックをいれると，XMLBase 宣言が RDF(S) 文書内に追加される.

4. ファイル

設定した条件（構文，データタイプ，オプション）で，RDF(S) 文書にエクスポートする.

5. 画像

「3. データタイプ」の「RDF」,「クラス」,「プロパティ」チェックボックスのうちどれか 1 つを選択し,「画像」ボタンを押すと，選択されたエディタのデータグラフを png 形式でファイルに保存することができる.

6. 再読み込み

設定した条件で最新のデータグラフを「8. ソース表示」に表示する.

7. 取消し

エクスポートを中止し，エクスポートダイアログを閉じる.

図 3.37: 削除ダイアログ

8. ソース表示

エクスポートされる RDF(S) 文書の内容が表示される.

3.5.8 削除ダイアログの実装

削除対象の RDFS クラスを，RDF リソースのタイプまたは RDF プロパティの定義域または値域が参照している場合，その RDFS クラスを削除すると整合性を保つことができない．RDF プロパティが削除対象の RDFS プロパティを参照している場合も同様である．これらの場合，RDF(S) コンテンツ管理機能によって，図 3.37 に示す「削除」ダイアログが表示される．図 3.37 上部の「削除」リストには，削除対象の RDFS クラスのうち，削除すると整合性を保つことができない RDFS クラスまたは RDFS プロパティのリストが表示される．図 3.37 下部の参照リスト内の「RDF」タブには，削除対象の RDFS クラスをタイプとして参照している RDF リソースのリストまたは，削除対象の RDFS プロパティを参照している RDF プロパティのリストが表示される.「プロパティ」タブには，定義域または値域として，削除対象の RDFS クラスを参照している RDFS プロパティの一覧が表示される．参照リストの「削除」チェックボックスは，RDFS クラスまたは，RDFS プロパティの参照をやめるかどうかの決定に用いる.「削除」チェックボックスをチェックした状態で「適用」ボタンをクリックすると，削除対象の RDFS クラスまたは RDFS プロパティへの参照がたたれる.「削除」チェックボックスのチェックをはずした項目については,「適用」ボタンを押したときに整合性のチェックを行う.「全選択」ボタンは，すべての「削除」チェックボックスにチェックをつける.「全解除」ボタンは，すべての「削除」チェックボックスのチェックをはずす.「反転」ボタンは，現在チェックされているものとされていないものを反転する.「ジャンプ」ボタンは，選択したリソースへジャンプする．RDF リソースのタイプ，定義域，値域を変更したい場合は，ジャンプボタンで該当するリソースへジャンプして変更を行うことができる.

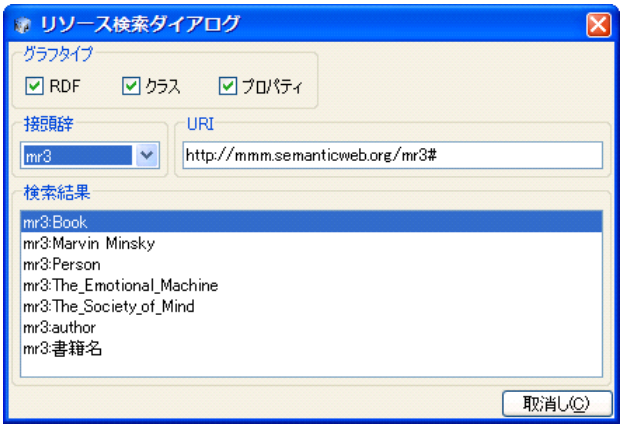


図 3.38: リソース検索ダイアログ

3.5.9 リソース検索ダイアログの実装

「リソース検索」ダイアログでは，RDF エディタ，クラスエディタ，プロパティエディタ内に定義されている RDF(S) 要素の検索を行うことができる．図 3.38 にリソース検索ダイアログを示す.「グラフタイプ」内の「RDF」,「クラス」,「プロパティ」チェックボックスのいずれか一つを選択し，検索範囲を設定する.「URI」テキストフィールドに検索したい URI を入力すると,「グラフタイプ」内で選択されたチェックボックスの検索範囲内でリソースを検索する．該当するリソース一覧が，図 3.38 下部の「検索結果」リストに表示される.「検索結果」リストの項目を選択すると，選択したリソースを含むエディタ内のノードへジャンプし，アトリビュートダイアログにそのリソースの属性が表示される．

3.5.10 設定ダイアログの実装

設定ダイアログでは，基本，ディレクトリ，プロキシ，メタクラス，レイアウト，レンダリングについて設定を行うことができる．

基本

設定ダイアログの「基本」タブ（図 3.39）では，言語，UI 言語，出力エンコーディング，フォント，ベース URI，ログファイルの設定ができる．言語はラベル表示を行う際に優先して表示する言語を設定できる．UI 言語は，メニューなどに表示する言語を設定できる．言語は，ja（日本語），en（英語），zh（中国語）を選択できる．出力エンコーディングは，エクスポート時に RDF(S) 文書を保存するファイルエンコーディングを設定する．フォントは，各エディタ内のノード内の文字列を表示する際のフォントを設定する．ベース URI には，エクスポート時の初期の名前空間 URI を設定する．ログファイルには，MR^3 使用中の各種ログを保存するファイルを設定する．

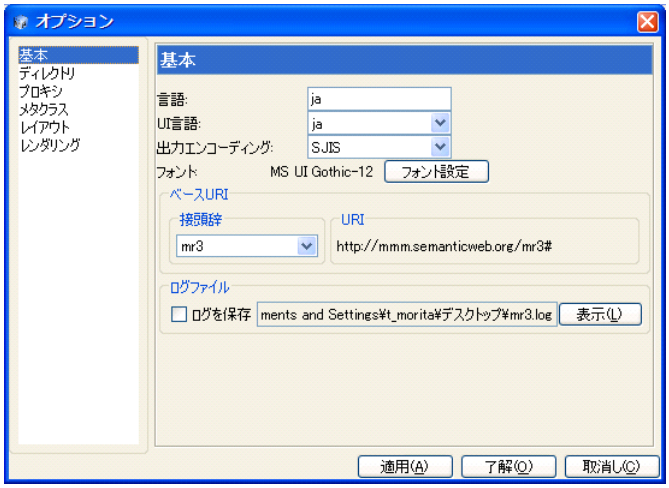


図 3.39: 設定ダイアログ: 基本

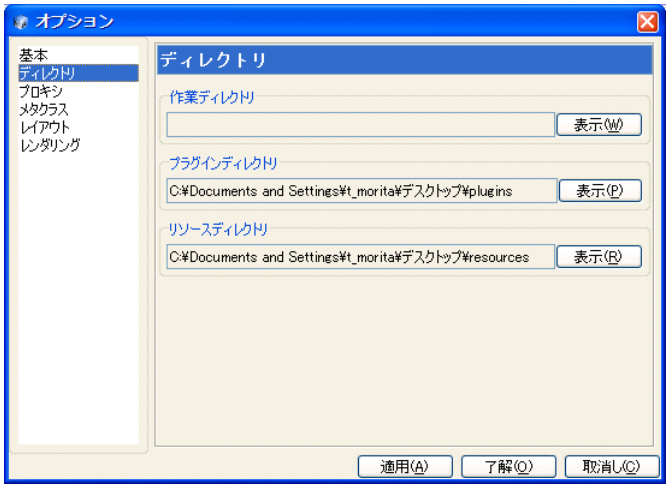


図 3.40: 設定ダイアログ: ディレクトリ

ディレクトリ

設定ダイアログの「ディレクトリ」タブ（図 3.40）では，作業ディレクトリ，プラグインディレクトリ，リソースディレクトリを設定ができる．作業ディレクトリには，インポートダイアログにおいて，RDF(S) 文書を含むフォルダを選択する際に最初に開くフォルダを設定する．プラグインディレクトリには， MR^3 のプラグインが保存されているフォルダを設定する．リソースディレクトリは， MR^3 のプロパティファイル（メニューなどに表示する言語ごとの設定ファイル）が保存されているフォルダを設定する．

プロキシ

設定ダイアログの「プロキシ」タブ（図 3.41）では，プロキシサーバを利用している場合に，プロキシサーバのホスト名とポート番号の設定を行うことができる．インポートダ

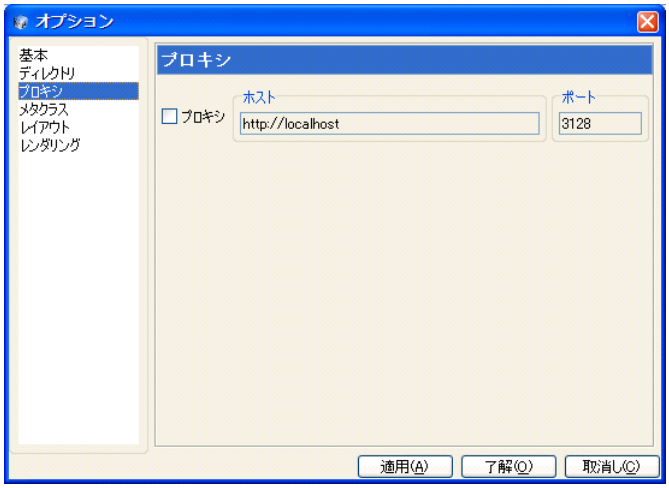


図 3.41: 設定ダイアログ: プロキシ

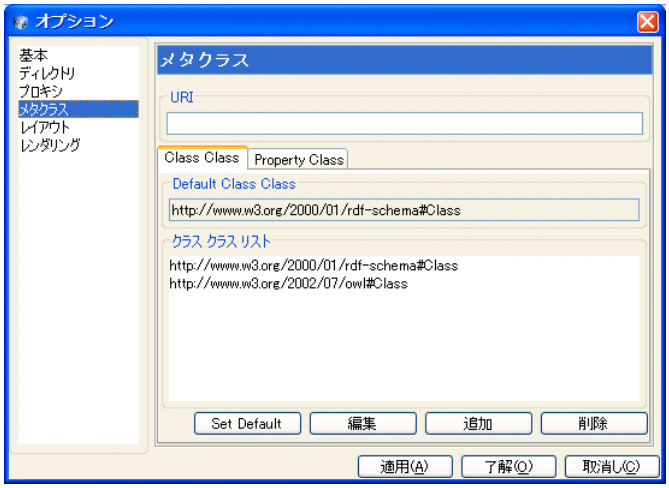


図 3.42: 設定ダイアログ: メタクラス

イアログから URI を指定して，RDF(S) 文書をインポートする場合に設定が必要となる場合がある．

メタクラス

設定ダイアログの「メタクラス」タブ（図 3.42）では,「クラス クラス」および「プロパティ クラス」の設定をすることができる.「クラス クラス」に設定したクラスを `rdf:type` プロパティの値とするリソースを， MR^3 はクラスとして認識し，クラスエディタにインポートすることができる．同様に,「プロパティ クラス」に設定したクラスを `rdf:type` プロパティの値とするリソースを， MR^3 はプロパティとして認識し，プロパティエディタにインポートすることができる．

初期状態では,「クラス クラス」には `rdfs:Class` が設定されている．また,「プロパティ

図 3.43: 設定ダイアログ: レイアウト

図 3.44: 設定ダイアログ: レンダリング

クラス」には，`rdf:Property` が設定されている．`owl:Class` や `owl:ObjectProperty` をタイプとするリソースを，クラスまたはプロパティとして MR^3 にインポートしたい場合には，図 3.42 の「Class Class」タブまたは「Property Class」タブで設定を行う．

レイアウト

設定ダイアログの「レイアウト」タブ（図 3.43）では，各エディタ内のデータグラフのレイアウト方法を設定する．

レンダリング

設定ダイアログの「レンダリング」タブ（図 3.44）では，RDF リソース，RDF リテラル，RDFS クラス，RDFS プロパティの各ノードの色や，ノード選択時の色，エディタウィ

表 3.3: MR^3 と関連研究の比較

	ツール名	開発元	RDF コンテンツの視覚的編集機能	RDF(S) コンテンツ管理機能		RDF プロパティの定義容易性	インポート/エクスポート/編集	
				O→M	M→O		RDF	RDFS
RDF ベース	IsaViz	W3C	[1]			✓	✓	✓
	RDFAuthor	Demian Steer	✓			✓	✓	[4]
	MR^3	慶應義塾大学, 静岡大学, 産業技術総合研究所	✓	✓	✓	✓	✓	✓
オントロジーベース	KAON OI-modeler	FZI Research Center and AIFB Institute, University of Karlsruhe	[1]	✓			[3]	✓
	OntoEdit	Ontoprise GmbH	[2]	✓			[3]	✓
	Protégé OWL Plugin	Stanford Medical Informatics, Stanford University		✓			[3]	✓

[1] これらのツールはオントロジーとモデルを分離して表示していない．[2] OntoEdit は，RDF コンテンツ全体を表示しながら RDF 要素を編集できない．[3] これらのツールは，空白ノードを扱うことができない．[4] RDFAuthor は，RDFS コンテンツの編集とエクスポートを支援していない．

ンドウの背景職をカスタマイズすることができる.「ノードに色をつける」チェックボックスのチェックをはずすと，すべてのノードが無色となる.「アンチエイリアス」チェックボックスにチェックをいれると各ノードの縁が滑らかに表示される.

3.6 RDF(S) コンテンツ構築支援ツールの評価

3.6.1 関連研究との比較

本項では，RDF(S) コンテンツ構築支援ツールと関連研究との比較について述べる．近年，セマンティック Web 技術（RDF, RDFS, OWL など）の標準化に伴い，商用およびオープンソースのセマンティック Web コンテンツ構築支援ツールが数多く開発されている．[38] では，現在利用可能な 94 のセマンティック Web コンテンツ構築支援ツールが紹介されている．本研究で開発を行った MR^3 についても [38] の調査結果の中で紹介されている．本項では，[38] の調査結果の中から，代表的な三つのオントロジーベースのツールと二つの RDF ベースのツールを選択し，RDF(S) コンテンツ構築支援ツールとの比較を行った.

表 3.3 は，RDF(S) コンテンツ構築支援ツール (MR^3) と他の関連研究との比較結果を示している．表 3.3 で比較を行った機能は，モデルとオントロジーの双方向に発生する修正コストを削減するという本研究目的にとって必須の機能である．表 3.3 におけるチェックの有無は，本研究の目的を達成するために必要な機能を比較ツールが有しているかどうかによって行った.

MR^3 の他ツールに比べて優位な点は，RDF コンテンツの視覚的編集機能と RDF(S) コンテンツ管理機能を有している点である．表 3.3 のツールは，RDF ベースのツールとオントロジーベースのツールの二つに分類している．IsaViz [39]，RDFAuthor [40]， MR^3 は RDF ベースのツールとして分類している．RDF ベースのツールは RDF コンテンツ構築に特化したツールである．そのため，これらのツールは優れた RDF コンテンツの視覚的編集機能を有している．KAON OI-Modeler [41]，OntoEdit [12]，Protégé OWL Plugin [13] は，オントロジーベースのツールとして分類している．オントロジーベースのツールはオントロジーの構築に特化している．そのため，これらのツールは，RDF コンテンツの視覚的編集機能をもっていないか，または，部分的にしか持っていない．IsaViz と KAON OI-modeler は RDF および RDFS コンテンツを混在して表示している．これらのツールを利用する場合，ユーザは RDF 要素と RDFS 要素を区別することが困難であるため，オントロジーとモデルを同時に構築する場合には，ユーザのオントロジーおよびモデル構築コストは大きくなる．RDFAuthor， MR^3 ，OntoEdit はオントロジーとモデルを分離して表示することができる．RDF(S) コンテンツの視覚的編集機能の評価については 3.6.3 項で述べる．

表 3.3 の O→M と M→O は，RDF(S) コンテンツ管理機能のサブ機能を表している．O→M は，オントロジー（RDFS クラスおよび RDFS プロパティ）の修正結果をモデル（RDF リソースのタイプおよび RDF プロパティ）に反映させる機能をあらわす．M→O は，モデルの修正結果をオントロジーに反映させる機能をあらわす．IsaViz と RDFAuthor は RDF(S) コンテンツ管理機能を持っていない．オントロジーベースのツールは，O→M の機能を持っている．しかしながら，オントロジーベースのツールは，M→O の機能を持っていない．よって， MR^3 の他ツールと比較して優れている点として，RDF ベースのツールであり，かつ，RDF(S) コンテンツ管理機能を持っている点があげられる．特に，M→O の機能は MR^3 独自の機能である．

表 3.3 の RDF プロパティの定義容易性は，RDFS プロパティの定義域および値域を定義することなく，RDF プロパティ（RDF リソース間の関係）を定義可能かどうかを意味する．オントロジーベースのツールは，RDF プロパティを定義するためには，あらかじめ RDFS プロパティにおいて，定義域および値域を定義しなければならない．RDF ベースのツールは，RDFS プロパティの定義域および値域を定義することなく，RDF プロパティを定義することができる．RDF コンテンツ構築の観点からは，RDF プロパティの定義をオントロジーの定義に依存することなく，ユーザが自由に行うことができることが望ましいと考えられる．

RDF コンテンツの構築方法として主に以下の二つが考えられる．一つは，あらかじめ定義されたオントロジーを用いて RDF コンテンツを構築する方法である（トップダウン）．もう一つは，オントロジーを参照することなく，RDF コンテンツを構築する方法である（ボトムアップ）[6]．M→O および RDF プロパティの定義容易性により，オントロジーと

[6]ボトムアップによる RDF コンテンツ構築のみでは，適切なオントロジーの構築を行うことはできない．ボトムアップにより生成したクラスやプロパティは，トップダウンにより階層関係の定義などを行う必要がある．

モデルの整合性を保ちながら，ユーザはボトムアップに RDF コンテンツの構築を行うことができる．M→O により，ユーザが RDF コンテンツ構築時にオントロジーに定義されていないクラスまたはプロパティを参照しようとした場合に，それらのクラスまたはプロパティは自動的にオントロジーに定義される．

表 3.3 のインポート/エクスポート/編集は，RDF(S) コンテンツのインポート，エクスポート，編集機能を示している．これらの機能について，RDF ベースのツールとオントロジーベースのツールの主な違いは，RDF コンテンツに関するインポート，エクスポート，編集を行うことができるかどうかである．オントロジーベースのツールは，空白ノードを含む RDF コンテンツの扱いを支援していない．空白ノードは，複数のリソースをまとめて構造化するためによく用いられる．既存の RDF コンテンツ（RSS [42] など）の多くは空白ノードを含んでいるため，RDF コンテンツ構築のために空白ノードを扱うことができることは重要となる．RDF ベースのツールは，空白ノードを扱うことができる．また，表 3.3 で示しているツールは，すべて RDFS コンテンツを扱うことができる．しかしながら，RDFAuthor は，RDFS コンテンツをインポートすることができるが，RDFS コンテンツの編集およびエクスポートは支援していない．

3.6.2 RDF(S) コンテンツ管理機能の評価

実験の概要

RDF(S) コンテンツ管理機能の有用性を確認するために，RDF(S) コンテンツ管理機能が働く場合と働かない場合の両方で，2 種類の異なる領域について， MR^3 を用いて RDF(S) コンテンツの構築を行った．評価実験は，RDF，RDFS， MR^3 に精通している 1 人の被験者が行った．被験者は，理工学部の 4 年生で，4ヶ月程度 RDF，RDFS，OWL についてのゼミに参加している．また，その間に， MR^3 を含むいくつかのセマンティック Web コンテンツ構築支援ツールを用いて，数回程度，セマンティック Web コンテンツの構築を経験している．被験者は，本評価実験で構築する RDF(S) コンテンツの領域専門家ではないため，領域知識を十分には持っているとはいえない．しかし，被験者は領域専門文書を読んで領域について理解することはできる．実験 1 および 2 の概要を図 3.45 に示す．被験者は，実験 1 を行った後に，実験 2 を行った．

実験 1 では，RDF(S) コンテンツ管理機能を有する MR^3 を用いて，被験者は自動車販売業務に関する RDF(S) コンテンツを構築した．RDF(S) コンテンツ構築時に，被験者は自動車販売業務の作業手順を参照した．自動車販売業務の作業手順は 862 語を含む日本語で記述されている．

実験 2 では，RDF(S) コンテンツ管理機能を取り除いた MR^3 を用いて，被験者は果樹園通信販売についての RDF(S) コンテンツを構築した．RDF(S) コンテンツ構築時に，被験者は果樹園通信販売の作業手順を参照した．果樹園通信販売の作業手順は 1245 語を含む日本語で記述されている．

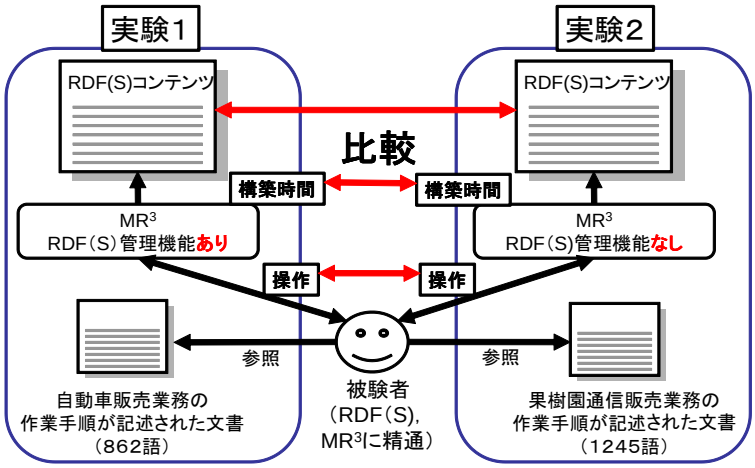


図 3.45: 実験 1 および 2 の概要

表 3.4: 実験 1 および 2 で構築されたステートメント数の比較

	実験 1	実験 2
RDFS クラス数	18	21
RDFS プロパティ数	16	16
RDFS 部分のステートメント数	65	79
RDF 部分のステートメント数	31	26
全ステートメント数	96	105

被験者の MR^3 の操作履歴をとるために，評価実験中は MR^3 の操作ログ保存機能が有効となっている．

実験の結果

表 3.4 から表 3.8 に実験結果を示す．表 3.4 は，実験 1 および 2 で構築されたステートメント数の比較を示している．表 3.5 は，実験 1 および 2 における RDF(S) コンテンツ構築時間を示している．表 3.6 は，実験 1 および 2 における MR^3 の操作回数の比較を示している．表 3.6 の操作とは，RDF 要素および RDFS 要素の挿入，編集，削除とする．**O→M** に関する操作は自動的に行われるため，表 3.6 には **O→M** の操作回数は含まれていない．しかし，**M→O** は半自動的に行われるため，表 3.6 には，**M→O** の操作回数は含まれて

表 3.5: 実験 1 および 2 における RDF(S) コンテンツ構築時間

実験 1	実験 2
27 分 40 秒	42 分 00 秒

表 3.6: 実験 1 および 2 における MR^3 の操作回数の比較

	実験 1	**実験 2**
モデル構築における操作回数	83	69
オントロジー構築における操作回数	49	96
全操作回数	132	165

表 3.7: 実験 1 における RDF(S) コンテンツ管理機能の使用回数

RDF(S) コンテンツ管理機能		**回数**
タイプ	**機能名**	
O→M	RDFS クラス名の変更	0
	RDFS プロパティ名の変更	2
M→O	RDF リソースタイプの変更（RDFS クラスの新規作成）	4
	RDF リソースタイプの変更（RDFS クラス名の変更）	2
	RDF プロパティの変更（RDFS プロパティの新規作成）	15
	RDF プロパティの変更（RDFS プロパティ名の変更）	0

いる．表 3.7 は，実験 1 における RDF(S) コンテンツ管理機能の使用回数を示している．表 3.8 は，実験 1 および実験 2 におけるオントロジーおよびモデル編集モードの切り替え回数の比較を示している．

実験の考察

表 3.4 より，実験 1 で構築された RDF(S) コンテンツの規模は，実験 2 で構築された RDF(S) コンテンツの規模とほぼ同程度といえる．表 3.5 より，RDF(S) コンテンツ管理機能を用いたほうが RDF(S) コンテンツ管理機能を用いないよりも早く RDF(S) コンテンツを被験者は構築できたといえる．表 3.6 より，RDF(S) コンテンツ管理機能により RDF(S) コンテンツ構築における操作回数を軽減することができたといえる．表 3.8 より，RDF(S) コンテンツ管理機能によりモデルおよびオントロジー間の編集モード切替回数を軽減することができたといえる．

表 3.8: 実験 1 および 2 におけるオントロジーおよびモデル編集モードの切替回数の比較

	実験 1	**実験 2**
モデル編集モードからオントロジー編集モードへの切替回数	13	34
オントロジー編集モードからモデル編集モードへの切替回数	12	34
全編集モードの切替回数	25	68

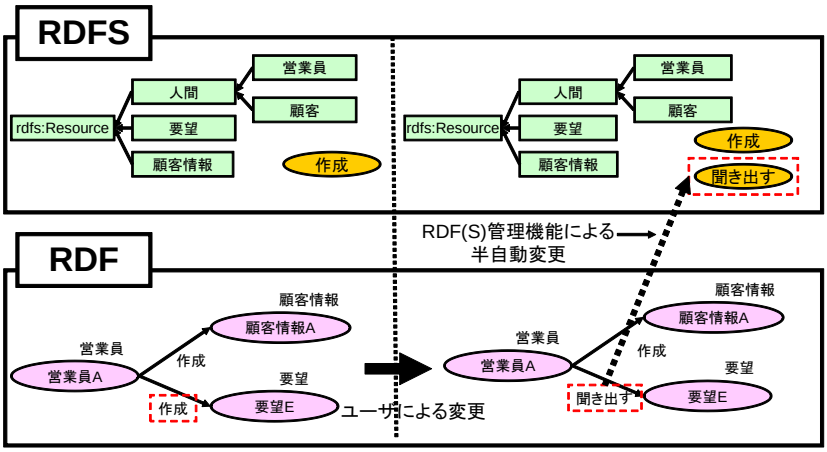


図 3.46: 実験 1 における RDF(S) コンテンツ管理機能の使用例

被験者からは，RDF(S) コンテンツ管理機能を用いない場合には，RDFS コンテンツの編集を行うたびに，整合性をとるために RDF コンテンツを修正することが面倒であったという感想が得られた．逆に，RDF(S) コンテンツ管理機能を用いた場合には，整合性が（半）自動的に保たれるため，RDF(S) コンテンツの修正を容易に行うことができたという感想が得られた．

図 3.46 は，実験 1 における RDF(S) コンテンツ管理機能の使用例を示している．図 3.46 の上部は，RDFS コンテンツを示しており，図 3.46 の下部は RDF コンテンツを示している．図 3.46 は，実験 1 において構築された RDF および RDFS コンテンツ全体の中から，RDF(S) コンテンツ管理機能が使用された場面に関連する部分の RDF および RDFS コンテンツを取り出したものとなっている．RDFS コンテンツにおける矩形は RDFS クラスを表しており，楕円は RDFS プロパティを表している．RDF コンテンツにおける楕円は RDF リソースを表しており，ラベル付き矢印は RDF プロパティを表している．また，RDF コンテンツにおける RDF リソースの右上のラベルは，RDF リソースのタイプを表している．図 3.46 では，RDF リソース「営業員 A」と RDF リソース「要望 E」の間の RDF プロパティ「作成」が，被験者により RDF プロパティ「聞き出す」に修正された．その際に，RDFS プロパティ「聞き出す」は RDFS コンテンツ中に定義されていないため， MR^3 は被験者に RDFS プロパティ「作成」の名前を変更するか，新規に RDFS プロパティ「聞き出す」を新規作成するかを確認した．ここで，被験者は RDF リソース「営業員 A」と RDF リソース「顧客情報 A」の間の関係は，RDF プロパティ「作成」のまま保持したいと考え，被験者は RDFS プロパティ「聞き出す」の新規作成を選択した．その結果，RDF(S) コンテンツ管理機能により，RDFS プロパティ「聞き出す」が RDFS コンテンツに半自動的に定義された．

RDF(S) コンテンツの構築は創造的な作業であるため，二人の被験者が同一領域の RDF(S) コンテンツを構築する場合でも，全く同一の RDF(S) コンテンツを構築することは困難である [43]．また，領域知識やツールの習熟度などの被験者の能力も構築結果に大きな影響を及ぼす．さらに，構築された RDF(S) コンテンツの質の評価を定量的に行うことも困

表 3.9: 評価データのリソース数，リテラル数，ステートメント数

	リソース数	リテラル数	ステートメント数
RDF 部分	15	35	59
RDFS 部分	34	46	131
全体	49	81	190

表 3.10: 評価データのクラス数およびプロパティ数

クラス数	プロパティ数
13	19

難である．そのため，RDF(S) コンテンツ構築支援ツールの評価を定量的に行うことは困難である．しかしながら，本実験によって，構築時間の削減，操作回数の軽減，オントロジーおよびモデル編集モードの切替回数の軽減という点において，RDF(S) コンテンツ管理機能の有用性を示すことができたといえる．

3.6.3 RDF(S) コンテンツの視覚的編集機能の評価

本項では，表 3.3 に示したツールと MR^3 の RDF(S) コンテンツの視覚的編集機能の比較について述べる．評価データとして，RSS のスキーマ[7]，RDFS の基本語彙[8]，研究室 Web ページの RSS データを用いた．表 3.9 に評価データのリソース数，リテラル数，ステートメント数を示す．表 3.10 に評価データのクラス数およびプロパティ数を示す．上記の三つの RDF コンテンツおよび RDFS コンテンツが混在した評価データを，表 3.3 に示したツールにインポートした．Protégé OWL Plugin は，RDF コンテンツの視覚的編集機能を持っていないため，比較評価を行うことができなかった．評価データには空白ノードが含まれていたため，KAON OI-modeler および OntoEdit は，評価データをインポートすることができなかった．よって，以下では表 3.3 に示したツールのうち，評価データをインポートすることができた，IsaViz，RDFAuthor， MR^3 の三つのツールについて，RDF(S) コンテンツの視覚的編集機能の比較評価を示す．

図 3.47 は，IsaViz に評価データをインポートした際のスクリーンショットを示している．図 3.47 では，RDFS 要素の階層関係をデータグラフから読み取ることは困難である．IsaViz では，RDF 要素および RDFS 要素が，同一色の楕円もしくは矩形で表示されている．IsaViz において RDFS 要素を識別するためには，ユーザは `rdfs:Class` または `rdf:Property` リソースを値とする `rdf:type` プロパティを持つリソースを検索する必要がある．図 3.48 および図 3.49 は，それぞれ，RDFAuthor および MR^3 に評価データをインポートした際の

[7] http://web.resource.org/rss/1.0/schema.rdf
[8] http://www.w3.org/TR/rdf-schema/rdfs-namespace

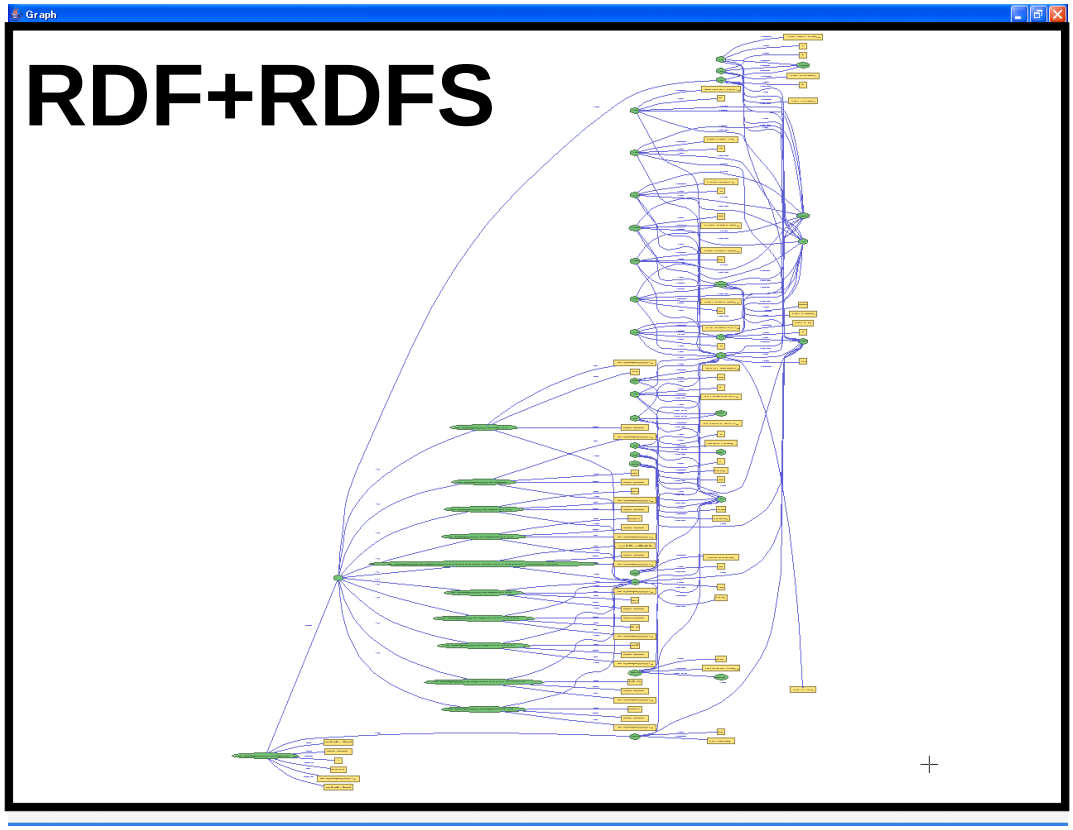


図 3.47: IsaViz による RDF(S) コンテンツの視覚化

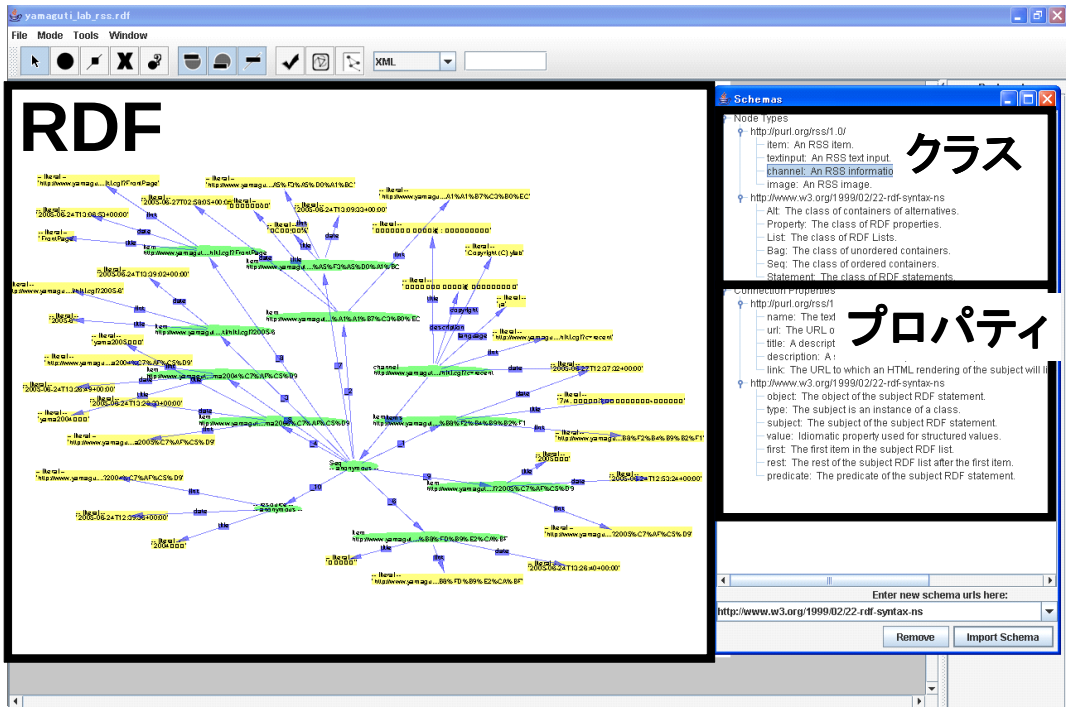


図 3.48: RDFAuthor による RDF(S) コンテンツの視覚化

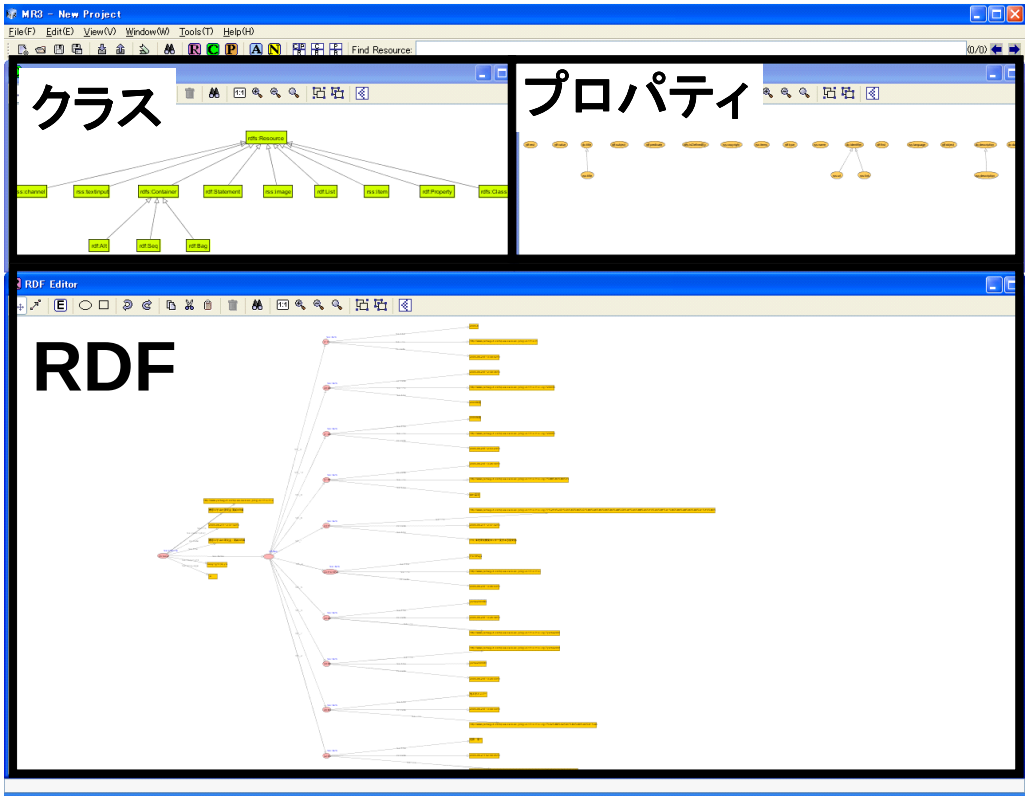


図 3.49: MR^3 による RDF(S) コンテンツの視覚化

スクリーンショットを示している．RDFAuthor および MR^3 は RDF 要素および RDFS 要素を分離して表示するため，ユーザは RDF 要素と RDFS 要素を容易に区別することができる．RDFAuthor は，RDFS クラスおよび RDFS プロパティのリストを名前空間 URI ごとに分類して，表示している．しかしながら，RDFAuthor は，RDFS クラスおよび RDFS プロパティの階層関係の表示および編集機能は支援していない．一方， MR^3 は，RDFS クラスおよび RDFS プロパティの階層関係を表示および編集することができる．

以上より，IsaViz，RDFAuthor， MR^3 における RDF(S) コンテンツの視覚的編集機能の比較評価では，RDFAuthor および MR^3 のデータグラフが IsaViz のデータグラフと比べて，RDF 要素および RDFS 要素の識別を容易に行うことが確認できた．RDF 要素および RDFS 要素の識別を容易に行うことができた理由の一つは，RDF 要素および RDFS 要素を分離して表示している点である．特に， MR^3 は，データグラフをユーザが容易に理解できるように，不要な要素はデータグラフ上に表示しないようにしている．例えば， MR^3 では，`rdfs:label` プロパティや `rdfs:comment` プロパティなどの RDF リソースの属性はデータグラフ上には表示せずに，アトリビュートダイアログを通して編集を行うことができるようにしている．同様に，`rdfs:label`，`rdfs:comment`，`rdfs:domain`，`rdfs:range` などの RDFS クラスおよび RDFS プロパティの属性や関係も，データグラフ上には表示せずに，アトリビュートダイアログを通して編集できるようにしている．`rdfs:label` プロパティは，主に人間がリソースを識別する際に用いるラベルであり，`rdfs:comment` プロパティは，主に人間がリソースについて理解するための説明を記述する際に用いられ

図 3.50: MR^3 プロジェクト Web サイト

る．これらは，RDF(S) コンテンツ構築においては付加的な情報であり，データグラフ上に表示してしまうと，その他のモデリングにとって重要なリソース間関係との区別が困難となる． MR^3 では，`rdfs:label` プロパティの値は，各ノードを識別するために URI の代わりにノード上に表示することもできるようになっている．

以上より， MR^3 を用いることで，RDF コンテンツおよび RDFS コンテンツの中でモデリングに重要なリソース間関係の視覚的な表示および編集にユーザは注力することができるといえる．

3.7 オープンソースソフトウェアとしての RDF(S) コンテンツ構築支援ツール

RDF(S) コンテンツ構築支援ツール MR^3 をセマンティック Web コミュニティに利用してもらえるように， MR^3 のシステムおよびソースコードを Web 上で公開している（図 3.50）[9]．2007 年 12 月現在，約 40 カ国，約 750 のユーザ登録がなされている． MR^3 を公開することによって，様々なユーザからコメントを得ることができている．例えば，厳密なフレーム指向（クラスおよびプロパティを定義してからでなければ，モデルを構築できない）ではなく，RDF コンテンツを構築できる点が良いというコメントや，具体化

[9] http://www.yamaguti.comp.ae.keio.ac.jp/mmm/mr3/

(reification)[10]や数千クラスを扱うことができるようにしてほしいという要望を得ることができた．また，rdf:Bag などのコンテナモデルを扱うためのプラグインや RDF データベース Sesame [44] と接続するためのプラグインなどが MR^3 のユーザにより開発されている．ユーザからのコメントは，将来的に MR^3 に反映させていきたいと考えている．

3.8 まとめ

RDF(S) コンテンツ構築支援を目的として，RDF(S) コンテンツの視覚的編集機能および RDF(S) コンテンツ管理機能を持つ，RDF(S) コンテンツ構築支援ツールの提案を行った．RDF コンテンツおよび RDFS コンテンツはモデルとオントロジーの関係として捉えることができることから，RDF(S) コンテンツ構築支援ツールでは，RDF コンテンツおよび RDFS コンテンツを分離し，それらの間の関係を半自動的に管理することができる．本章では，RDF(S) コンテンツ管理を，モデルとオントロジーの間の関係（特にオントロジーにおけるクラスおよびプロパティとモデルにおける RDF リソースのタイプおよび RDF プロパティの対応）を適切に管理することと定義した．また，RDF(S) コンテンツ構築支援ツールのプロトタイプ実装である MR^3 (Meta-Model Management based on RDFs Revision Reflection) を Java 言語を用いて実装した． MR^3 と他の関連研究との比較評価実験を行い， MR^3 の RDF(S) コンテンツ管理機能の有用性および RDF(S) コンテンツの視覚的編集機能の有用性を示すことができた．

MR^3 は，プラグイン機構を用いて，4 章で述べる領域オントロジー構築支援環境 DODDLE-OWL と連携可能となっている．DODDLE-OWL により，半自動的に構築された OWL 形式のオントロジー（クラス階層，プロパティ階層，プロパティの定義域および値域）をインポートし，視覚的な編集行うことができる．しかしながら，OWL については現状では MR^3 は部分的にしか対応していないため，今後の課題として，OWL Lite レベルのオントロジーの視覚的編集を支援することがあげられる．また，RDF(S) コンテンツ管理機能についても，現状では RDFS レベルの支援にとどまっている．OWL のクラス公理，プロパティ制約などが定義されているオントロジーとそのモデルの間に発生する種々の整合性管理機能を整理することは，今後の課題である．

[10]RDF におけるステートメントそのものをリソースとして，ステートメントを記述できるようにすること．ステートメントに関する評価などを記述する際に用いられる．

第4章　領域オントロジー構築支援環境

本章の内容は，文献 [45] に基づいている．

4.1　概要

人間とソフトウェアの間での共通理解において，オントロジーが必要であることが認識されつつあり [14]，オントロジーは様々なアプリケーション領域で注目を集めている．セマンティック Web は，最もオントロジーの利用が進んでいるアプリケーション領域の一つであり，多くのオントロジーが Web オントロジー記述言語 OWL (Web Ontology Language) [7] で記述されている．しかしながら，他のアプリケーション領域と同様，領域オントロジーの構築と保守には多大なコストがかかっている．さらに，日本語を対象とした，セマンティック Web における領域オントロジー構築支援環境は少ない．

テキストや汎用オントロジーなどの既存情報資源から自動的に領域オントロジーを構築するために，多くの研究は，知識工学，自然言語処理，データマイニングの技術を用いている [15, 16]．しかしながら，領域オントロジーの構造は専門家の対象領域の見方に依存して変化するため，領域オントロジーを全自動で構築することは困難である．実用的な領域オントロジー構築支援を行うためには，全自動ではなく，ユーザとのインタラクションを通して，半自動的に領域オントロジーの構築を支援することが重要である．さらに，セマンティック Web コミュニティに広く領域オントロジー構築支援環境を利用してもらい，ユーザビリティやオントロジー構築支援機能に対するフィードバックを得ることができるように，オープンソースソフトウェアとして公開することも重要である．

以上より，本論文では，インタラクティブで実用的な領域オントロジー構築支援環境 DODDLE-OWL (a Domain Ontology rapiD DeveLopment Environment - OWL extension) を提案する．DODDLE-OWL は，領域専門文書を入力として，WordNet [17] や EDR 電子化辞書 [18] などの既存オントロジーを参照しながら，領域オントロジーにおける階層関係およびその他（階層以外）の関係の半自動構築を支援する．DODDLE-OWL は，DODDLE-II [19][1]を基礎として，オープンソースソフトウェアとして Java 言語で実装されており，領域オントロジーを OWL 形式でエクスポートする機能や日本語領域オントロジーの構築支援を行うための機能など，様々な機能が拡張されている．特に，ユーザ中心

[1]DODDLE-II は，知識システムのためのオントロジー構築を目的として設計されている．DODDLE-II はオープンソースソフトウェアではなく，セマンティック Web の標準的なオントロジー記述言語 OWL でのエクスポートや日本語領域オントロジー構築に対応していなかった．また，DODDLE-II は領域オントロジー構築における各モジュールの統合が不十分であった．

表 4.1: 名前空間接頭辞および名前空間 URI

名前空間接頭辞	名前空間 URI
doddle	http://www.yamaguti.comp.ae.keio.ac.jp/mmm/doddle/
daml	http://www.daml.org/2001/03/daml-ont#
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	http://www.w3.org/2000/01/rdf-schema#
owl	http://www.w3.org/2002/07/owl#
skos	http://www.w3.org/2004/02/skos/core#
wn20schema	http://www.w3.org/2006/03/wn/wn20/schema/

のインタラクティブな領域オントロジー構築支援を行うために，DODDLE-OWL では各モジュールにユーザとインタラクションをとるための機能が備わっている．

なお，本章では，表 4.1 に示す名前空間接頭辞を用いて説明を行っている．

以降，本章の構成は次のとおりである．4.2 節では，領域オントロジー構築支援環境の分析について述べる．4.3 節では，領域オントロジー構築支援環境の基本設計について述べる．4.4 節から 4.9 節では，領域オントロジー構築支援環境の各モジュールについて述べる．4.10 節では，領域オントロジー構築支援環境の実装について述べる．4.11 節では，DODDLE-OWL と関連研究との比較について述べる．4.12 節では，DODDLE-OWL の有用性の評価として，DODDLE-OWL と手動構築の比較評価について述べる．4.13 節では，DODDLE-OWL のスケーラビリティの評価として，ロケット運用に関する大規模オントロジーの構築に関するケーススタディについて述べる．4.14 節では，既存オントロジーの獲得およびランキングの実験について述べる．4.15 節では，オープンソースソフトウェアとしての DODDLE-OWL について述べる．最後に，4.16 節では，本章のまとめと今後の課題について述べる．

4.2 領域オントロジー構築支援環境の分析

領域オントロジー構築支援を行う方法として，大きく分けて手動構築支援と半自動構築支援の二つが考えられる．手動構築支援は，概念数が膨大な場合にユーザの負担が大きくなることから，本章で提案する領域オントロジー構築支援環境 DODDLE-OWL では，領域オントロジーの半自動構築支援を試みる．

[46] では，半自動的にオントロジー構築を行う際の学習機能を，用語，同義語，概念，概念階層，関係，関係階層，公理スキーマ，汎用公理の八つに分類している．ここでは，これらの学習機能の中から特にオントロジーにおいて重要である，概念階層および関係に焦点を絞る．

2.5 節でも述べたように，オントロジーの半自動構築支援には，電子化辞書，専門文書，既存オントロジーなどの既存情報資源が活用されている．DODDLE-OWL も，既存研究

と同様，既存情報資源を活用することとする．しかしながら，これらの既存情報資源を利用して領域オントロジーを構築する場合には，いくつかの問題がある．

電子化辞書を利用した領域オントロジー構築支援における問題として，[47] では電子化辞書内の概念を十分に利用することが困難という経験が示されている．[47] で紹介されているケーススタディでは，一般的なオントロジー (WordNet) と領域オントロジーの間で概念変動と呼ばれる上位概念と下位概念の入れ替えが生じている．そのような場合，概念変動の生じている自動構築されたオントロジーを専門家に見せることは混乱をもたらす．このことから，領域オントロジー構築支援環境では，主導権を計算機ではなく専門家にもたせる必要があると考えられる．また，概念変動を管理するための機能が領域オントロジー構築支援環境には必要だと考えられる．

専門文書を利用した領域オントロジー構築支援における問題として，自動的に獲得された概念間関係の洗練の問題がある．自然言語処理やデータマイニングなどの技術を用いて獲得される概念間関係は，領域オントロジーの構造が専門家の見方によって変化する以上，完全であることはない．そのため，必ず獲得された概念間関係の洗練を行う必要がある．このことから，領域オントロジー構築支援環境では，ユーザとインタラクションをとりながら，自動的に獲得された概念間関係を洗練するための機能が必要であると考えられる．

既存オントロジーの利用については，利用可能な既存オントロジーをどのように獲得するかという問題がある．Web 上に散在する既存オントロジーの中から，ユーザが構築しようとしている領域オントロジーに類似するオントロジーを発見することは，ユーザの負担となる．このことから，領域オントロジー構築支援環境では，既存オントロジーを獲得するための機能が必要であると考えられる．

本章では，上記の問題を考慮した領域オントロジー構築支援環境を提案する．

4.3 領域オントロジー構築支援環境の基本設計

図 4.1 に DODDLE-OWL のシステムフローを示す．DODDLE-OWL は，オントロジー選択モジュール，入力モジュール，オントロジー構築モジュール，オントロジー洗練モジュール，視覚化モジュール，変換モジュールの主に六つのモジュールから構成される．

階層構築モジュールおよび階層洗練モジュールは階層関係構築支援のために，DODDLE-I [47] で提案された．関係構築モジュールおよび関係洗練モジュールは，その他の関係構築支援のために，DODDLE-II [19] で提案された．オントロジー選択モジュール，入力モジュール，視覚化モジュール，変換モジュールが DODDLE-OWL で新たに提案するモジュールである．また，DODDLE-OWL では，DODDLE-I および DODDLE-II で提案された各モジュールを統合し，インタラクティブな領域オントロジー構築支援環境を実現している．

DODDLE-OWL では一つ以上の領域における専門文書があることを前提としている．また，ユーザは領域にとって重要な用語（入力語）を選択可能な知識をもっているものとする．

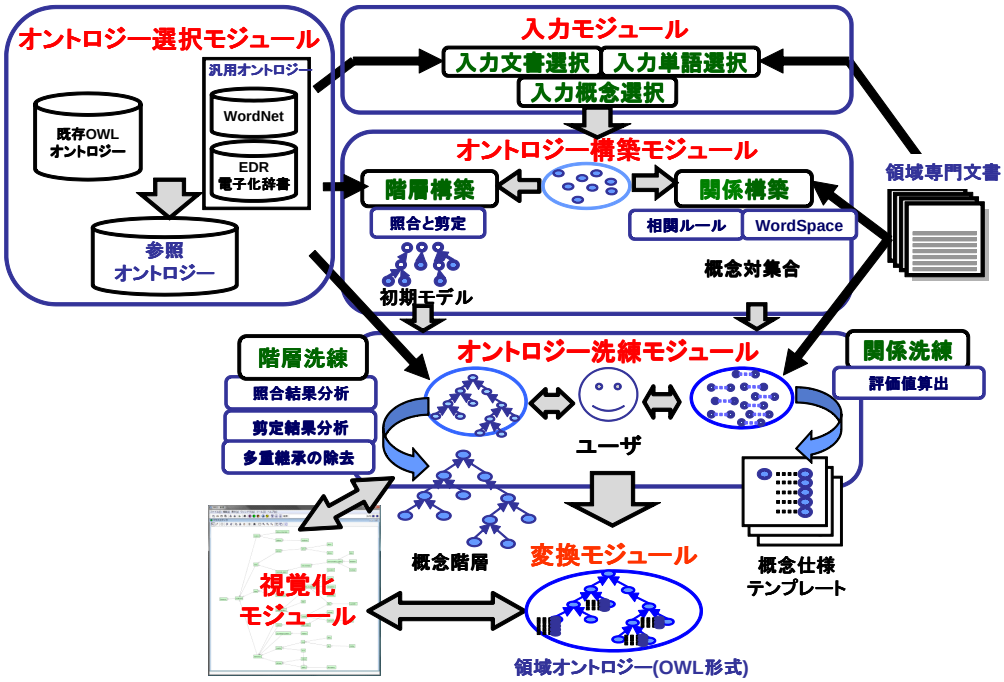


図 4.1: DODDLE-OWL のシステムフロー

以下，4.4 節から 4.9 節では，図 4.1 のシステムフローに従って，DODDLE-OWL の各モジュールの設計について説明する．

4.4 オントロジー選択モジュールの設計

オントロジー選択モジュールでは，ユーザは参照オントロジーの選択を行う．参照オントロジーは領域オントロジーを構築するための基礎となるオントロジーであり，DODDLE-OWL の各モジュールから参照される．DODDLE-OWL では，WordNet [17]（英語汎用オントロジー）および EDR [18]（日本語汎用オントロジー）を参照オントロジーとして利用可能である．さらに，OWL 形式で記述された既存オントロジーも DODDLE-OWL は参照オントロジーとして利用可能である．WordNet や EDR などの汎用オントロジーは，一般的かつ網羅的に定義がなされているため，領域オントロジー構築に利用する際には，領域に特化した構造へ，階層関係の修正や不要概念の除去などの洗練を行う必要がある．このことはユーザの負担となる．構築対象に関連する領域オントロジーがすでに存在する場合は，それらを再利用するほうが汎用オントロジーを参照オントロジーとするよりも，領域オントロジーを洗練するコストを軽減できると考えられる．また，DODDLE-OWL を用いて構築されるオントロジーは OWL 形式であるため，それらも参照オントロジーとして利用することができる．よって，DODDLE-OWL では，既存領域オントロジーが存在しない，または，既存領域オントロジーが，ユーザが構築対象とする領域オントロジーを網羅できていない場合には，汎用オントロジーを利用することができ，既存領域オントロジーが存在する場合には，より容易に対象とする領域オントロジーの構築支援を行うこ

とができる．

Web 上に存在する既存オントロジーを参照オントロジーとして再利用するために，オントロジー選択モジュールには，オントロジー検索エンジンを用いた既存領域オントロジーの獲得機能がある．以下では，オントロジー検索エンジンを用いた既存領域オントロジー獲得方法について述べる．

4.4.1 オントロジー検索エンジンを用いた既存領域オントロジーの獲得

既存領域オントロジーを再利用するためには，対象領域に関連する既存オントロジーを Web 上から検索する必要がある．既存のオントロジー検索エンジンとして，OntoSelect [48] や Swoogle [49] がある．OntoSelect は，Web 上のオントロジーを検索，選択，閲覧することを支援している．Swoogle には，2007 年現在，1 万以上のオントロジーが登録されており，クラス単位，プロパティ単位の検索やオントロジー内に明示的に記述されていない逆リンクの関係を検索することができる．

既存のオントロジー検索エンジンは，対象領域に関連する既存オントロジー検索に利用可能であるが，領域オントロジー構築の観点からはいくつか問題がある．既存のオントロジー検索エンジンは，クラス単位やプロパティ単位での検索を支援しているが，複数のクラスやプロパティを含むオントロジーの検索や対象領域と関連の深いオントロジーの検索は支援していない．Swoogle には，google のページランク [50] に類似したオントロジーのためのランキングの仕組み (OntoRank) や，クラスおよびプロパティのためのランキングの仕組み (TermRank) が導入されている．OntoRank や TermRank では，より多くのセマンティック Web コンテンツから参照されているクラスやプロパティを高くランキングしているが，領域オントロジー構築の観点からは，必ずしも多数のセマンティック Web コンテンツから参照されているオントロジーが利用可能とは限らない．ユーザが構築の対象としている領域に関連する既存オントロジーを適切に検索するための仕組みが必要となる．また，既存オントロジーを修正することなく再利用できることは少ないため，領域オントロジー構築支援環境とオントロジー検索エンジンは連携できたほうが望ましいと考えられる．

Swoogle は，オントロジーを検索するための 19 種類の REST 形式の Web サービス（Swoogle Web サービス）を提供している．DODDLE-OWL から既存オントロジーを再利用するためには，ソフトウェアからオントロジーを検索することが必要となるため，Web サービスを提供している Swoogle を，既存領域オントロジーの獲得に利用する．

オントロジー選択モジュールでは，図 4.2 に示す手順で，Swoogle を用いて既存オントロジーの獲得およびランキングを行う．

1. 入力語を URI のローカル名または `rdfs:label` プロパティの値として持つクラスおよびプロパティ（入力概念）の獲得

2. 手順 1 で獲得したクラスを定義域または値域とするプロパティの獲得

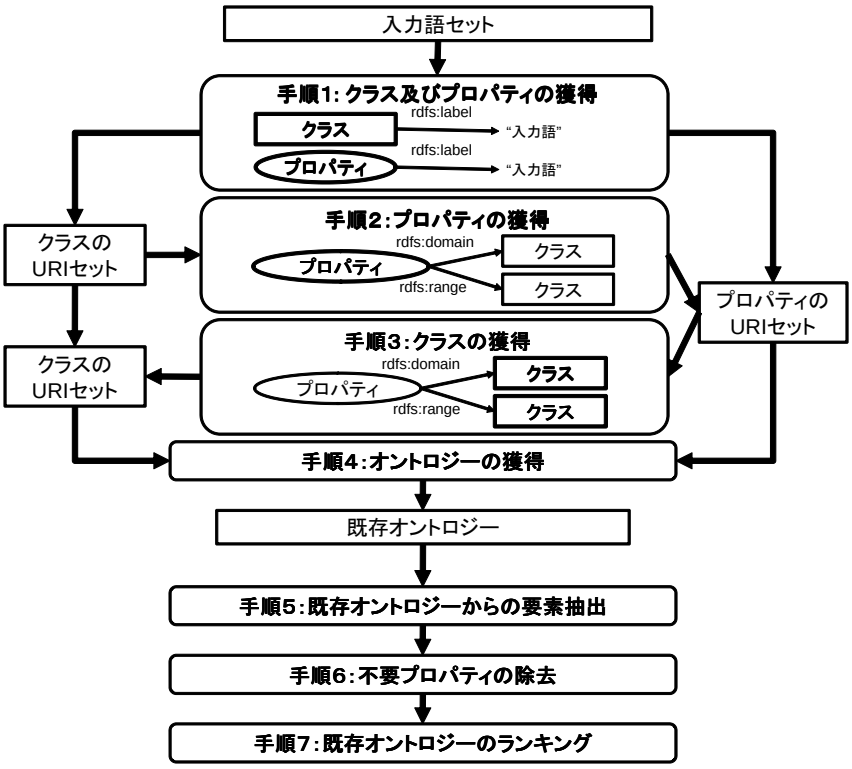


図 4.2: Swoogle を用いた既存オントロジーの獲得およびランキング

3. 手順 1 および 2 で獲得したプロパティの定義域および値域（クラス）を獲得
4. 手順 1 から 3 で獲得したクラスおよびプロパティを定義しているオントロジーの獲得
5. 手順 4 で獲得したオントロジーからの要素抽出
6. 手順 4 で獲得したオントロジーに定義されたプロパティの中で，定義域および値域が入力概念または入力概念の上位概念でないものの削除
7. 手順 4 で獲得した既存オントロジーのランキング

手順 5 の既存オントロジーからの要素抽出の詳細については，4.4.2 項で述べる．手順 7 の既存オントロジーのランキングの詳細については，4.4.3 項で述べる．

4.4.2 既存オントロジーからの要素抽出

OWL オントロジーを参照オントロジーとして領域オントロジー構築支援で利用するためには，オントロジー構築支援に利用可能な要素を OWL オントロジーから抽出する必要がある．DODDLE-OWL では，領域オントロジーにおける階層関係およびその他の関係の定義を支援している．概念関係およびその他の関係を定義するために必要な OWL オントロジーの要素は，概念（クラスおよびプロパティ），概念の見出し，概念の説明，階層関係，その他の関係である．概念の抽出は，領域オントロジー構築において必須である．

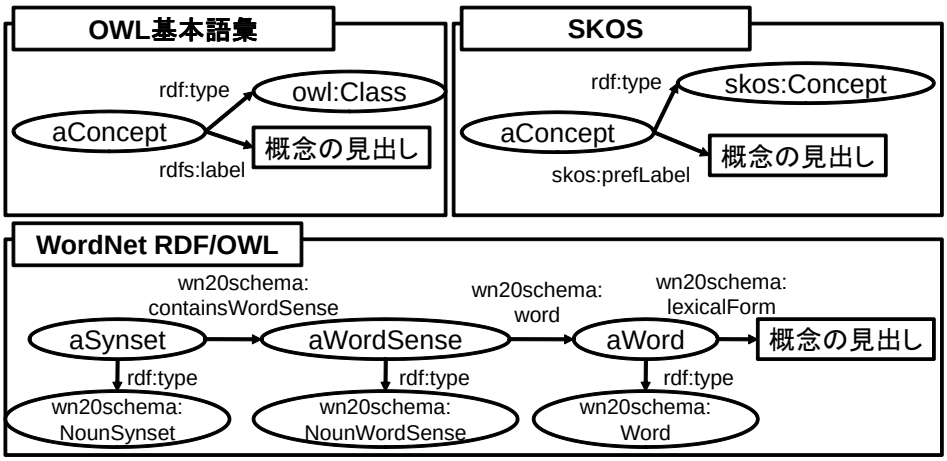


図 4.3: OWL 基本語彙，SKOS，WordNet RDF/OWL スキーマにおける “概念の見出し” の表現方法

概念の見出しは，入力語と概念を対応づける（入力概念選択）ために必要である．概念の説明は，入力語に対応する概念が複数ある場合に，入力概念をユーザが選択する際の参考となる．階層関係はクラスおよびプロパティ階層を構築する際に必要である．その他の関係を定義するために，プロパティの定義域および値域を抽出する必要がある．RDFS, DAML, OWL などのオントロジー記述言語は，上記のオントロジーの要素を定義するために基本的なクラスやプロパティを提供している．

Swoogle では，現状では RDFS, DAML, OWL が提供している基本的なクラスおよびプロパティに基づいて，オントロジーの検索を行うことができるようになっている．例えば，Swoogle ではクラスを，次の (X, Y, Z) というステートメントを満たす X と定義している．

- X は匿名リソース（空白ノード）ではない
- Y は `rdf:type` プロパティである
- Z は以下のいずれかのクラスである - `rdfs:Class`, `owl:Class`, `owl:Restriction`, `owl:DataRange`, `daml:Class`, `daml:Datatype`, `daml:Restriction`

Swoogle が扱う範囲のクラスやプロパティのみを OWL オントロジーから抽出する場合には，Swoogle の定義に従ってオントロジーの要素を抽出すればよい．しかし，それ以外の形式で定義された汎用オントロジーやシソーラスが存在する． [51] では WordNet や EDR の OWL 化について検討しており，それらは Swoogle で定義しているクラスやプロパティとは構造が異なっている．図 4.3 に OWL 基本語彙，SKOS (Simple Knowledge Organisation System) [52]，WordNet RDF/OWL における “概念の見出し” の表現方法を示す． [53] では，Wikipedia に対して Web マイニングを行う手法である Wikipedia マイニングを提案し，シソーラス辞書を構築している [2] ．構築された Wikipedia シソーラス

[2] http://wikipedia-lab.org/

表 4.2: OWL 基本語彙，SKOS, WordNet RDF/OWL におけるオントロジーの要素を特定するクラスおよびプロパティ一覧

オントロジーの要素	オントロジーの要素を特定するクラスおよびプロパティ一覧
概念	rdfs:Class, owl:Class, rdf:Property, owl:ObjectProperty, etc skos:Concept wn20schema:WordSense, wn20schema:NounWordSense, etc
概念の見出し	rdfs:label skos:prefLabel, skos:altLabel, skos:hiddenLabel wn20schema:lexicalForm
概念の説明	rdfs:comment skos:definition wn20schema:gloss
階層関係	rdfs:subClassOf, rdfs:subPropertyOf skos:broader, skos:narrower wn20schema:hypernymOf, wn20schema:hyponymOf
その他の関係	rdfs:domain, rdfs:range skos:related wn20schema:antonymOf, wn20schema:partMeronymOf, etc

を表現するための語彙として SKOS を用いている．SKOS では，概念を表すクラスとして `skos:Concept`，上位概念を表すプロパティとして `skos:broader` などの語彙が定義されており，OWL 基本語彙とは異なっている．表 4.2 に OWL 基本語彙，SKOS, WordNet RDF/OWL におけるオントロジーの要素を特定するクラスおよびプロパティ一覧を示す.

DODDLE-OWL では，多用な形式のオントロジーからオントロジーの要素を抽出するために，クラス抽出テンプレート，プロパティ抽出テンプレート，見出しと説明抽出テンプレート，階層関係抽出テンプレート，その他の関係抽出テンプレートの 5 種類のテンプレートを RDF クエリー言語 SPARQL [8] を用いて記述し，OWL オントロジーと対応づけている.

4.4.3 既存オントロジーのランキング

DODDLE-OWL は，現状では Swoogle の `OntoRank` および `TermRank`，オントロジー中の入力概念を含む割合，オントロジー中の入力概念に関するその他の関係数の四つをランキングのための指標としている．`OntoRank` および `TermRank` は， [49] で提案されているオントロジーとクラスおよびプロパティをランキングするための指標である．本研究では，入力概念をより多く含むオントロジーを対象領域に関連するオントロジーと仮定する．また，入力概念に関するその他の関係をより多く定義しているオントロジーも対象領域に関連するオントロジーと仮定する．入力概念を含む割合が同程度のオントロジーについては，`OntoRank` を参考にすることで，ユーザはより多くのセマンティック Web 文書で

参照されているオントロジーを再利用することが可能となる．語の多義性により，入力概念の候補が複数ある場合には，TermRank を参考にすることで，ユーザはより多くのオントロジーで参照されている入力概念を再利用することが可能となる．

4.4.4 既存オントロジー再利用における課題

詳細については，4.6 節で述べるが，DODDLE-OWL では，階層関係構築支援を行うために，参照オントロジーから入力概念に関連するパスを抽出し，合成および不要概念の剪定を行う．Web 上に散在する異種のオントロジーのパスを合成する際には，上位概念階層の構造の違いにより単純に合成することは困難である．そのため，オントロジーアライメントによる類似概念の同定が必要となる．現状では，オントロジーアライメントを用いた階層関係構築支援は実現できていない．オントロジーアライメントについては，オントロジーアライメントのコンテストが活発に行われており[3]，ツールも多数公開されている．オントロジーアライメントツールと DODDLE-OWL の連携については，今後の課題である．

4.5 入力モジュールの設計

入力モジュールは，領域専門文書集合を入力として，参照オントロジーを参照し，入力概念集合を出力する．入力モジュールは，入力文書選択モジュール，入力語選択モジュール，入力概念選択モジュールから構成される．図 4.4 に入力モジュールのシステムフローを示す．以下では，各モジュールについて説明する．

4.5.1 入力文書選択モジュール

入力文書選択モジュールでは，英語または日本語で記述された領域に関する専門文書集合（入力文書集合）をユーザが選択し，入力文書集合の中から領域にとって重要な語（入力語）の候補となる用語集合を抽出する．入力文書選択モジュールでは，形態素解析器を用いて，専門文書中のユーザが指定した品詞（名詞，動詞，その他の品詞など）の語を抽出できる．また，専門用語自動抽出システム [54] や日本語係り受け解析器[4]を用いて複合語の抽出を行うこともできる．テキスト文書だけでなく，PDF，Microsoft Word, Excel, PowerPoint など様々な形式のファイルからテキストを抽出することもできる．

入力文書選択モジュールのもう一つの役割として，入力文書中の 1 文の区切りの同定がある．1 文の区切りの同定は，オントロジーにおけるその他の関係構築支援手法の一つである相関ルールを適用する際に必要となる．1 文の区切りを丸（。），ピリオド（.），改行などから自動的に入力文書選択モジュールは判別するが，丸やピリオドが入力文書に含

[3]http://oaei.ontologymatching.org/
[4]CaboCha: http://chasen.org/ taku/software/cabocha/

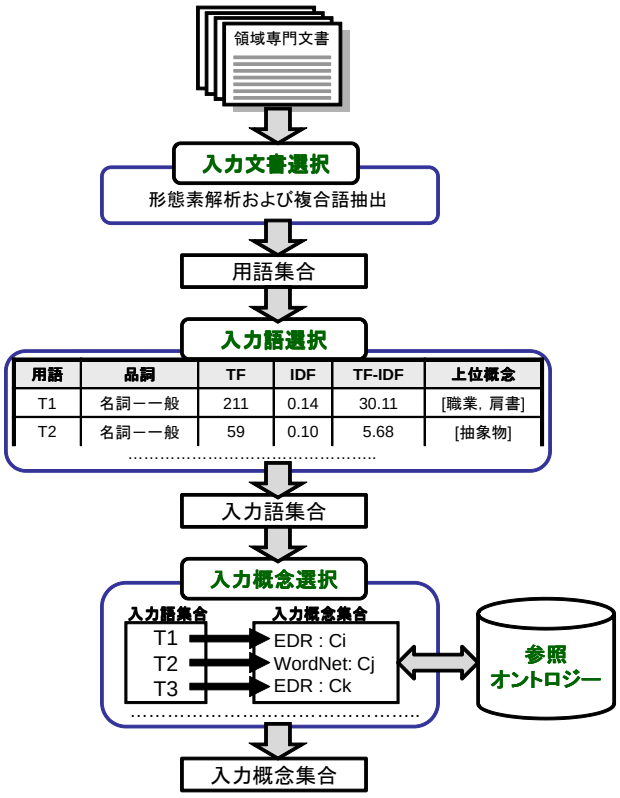


図 4.4: 入力モジュールのシステムフロー

まれない場合は，誤って 1 文を判別してしまう．このことは，相関ルールを用いた関係構築の精度の低下をもたらす．上記の問題を解決するため入力文書選択モジュールでは，1 文の区切りをユーザが手動で修正することができるようになっている．

4.5.2 入力語選択モジュール

入力語選択モジュールでは，入力文書選択モジュールにより自動抽出された用語集合から，複合語，品詞，TF (Term Frequency), IDF(Inverse Document Frequency), TF-IDF, 上位概念を考慮しながら，ユーザは入力語を選択する．ここで上位概念とは，参照オントロジーにおける概念階層の上位部分に存在する概念を表す．上位概念は，あらかじめユーザが手動で設定する．用語とその上位概念を同時に参照することにより，自動抽出された用語を抽象化してユーザは理解することができる．例えば，EDR を参照オントロジーとして,「具体物」を上位概念に設定した場合,「具体物」の下位概念の見出しと一致した自動抽出された用語については，その用語の上位概念として「具体物」を表示する．

入力文書中に入力語が含まれていない場合や，入力文書選択モジュールが自動抽出し損ねた入力語については，入力語選択モジュールでは，ユーザが手動で追加できるようになっている．また，入力文書からの入力語の選択漏れを防ぐために，抽出した入力語と入力文書中の出現箇所の対応関係がわかるようになっている．

4.5.3 入力概念選択モジュール

入力概念選択モジュールでは，ユーザはオントロジー選択モジュールで選択した参照オントロジー中の概念と入力語を対応づけることによって，入力語の意味を同定する．用語は複数の意味を持つ場合があるため，ある用語を見出しとしてもつ概念が複数存在する．入力概念選択モジュールでは，入力語とそれに対応する概念の候補をユーザに提示する．ユーザはその中から入力語に対応する，領域にとって最も適切な概念（入力概念）を選択する．

大部分の複合語は，それを見出しとして持つ概念が参照オントロジー中に存在しない．入力概念選択モジュールでは部分照合を行うことによって，より多くの複合語の入力概念選択を可能にしている．入力概念選択モジュールの入力概念選択方法は完全照合と部分照合の 2 種類がある．完全照合は，入力語と参照オントロジー中の概念の見出しが完全に一致することを意味する．部分照合は，入力語と参照オントロジー中の概念の見出しが部分的に一致することを意味する．完全照合しなかった入力語については，形態素解析を行い，先頭の形態素を順に除いて参照オントロジー中の概念と対応付けを試みる．ここで，先頭の形態素を順に除く理由は，複合名詞の語尾にあたる語のほうが，語頭にあたる語よりも重要（複合名詞の中心的な意味を表す）であると仮定しているためである．これは，複合名詞では一般的に，語尾にあたる語を，語尾以前の語が修飾することが多いという経験則を参考にしている．つまり，入力語中の語尾を含むように参照オントロジー中の概念の見出しと部分照合するようにしている．最終的に，最長一致した用語に対応する概念と対応付けを行う．部分照合した複合語については，対応する概念の下位概念または別見出し（同義語）として階層構築を行う．

例えば,「ロケット発射装置」という入力語について入力概念選択を行うことを考える．「ロケット発射装置」が完全照合しない場合，形態素解析を行い,「ロケット」と「発射」と「装置」に分解する．はじめに,「発射装置」について照合を試みる．次に「装置」について照合を試みる．この例では,「発射装置」を見出しとしてもつ概念は参照オントロジー中に存在せず,「装置」を見出しとして持つ概念が参照オントロジー中に存在する．よって,「ロケット発射装置」の意味として,「装置」を見出しとして持つ概念を候補としてユーザに提示する．その際に,「ロケット発射装置」を「装置」概念の下位概念とするか,「装置」概念の別見出しとするかをユーザは選択できる．

参照オントロジー中の概念に照合しなかった入力語は未定義語に分類され，オントロジー洗練モジュールにおいて階層中の適切な位置にユーザが手動で階層関係の定義を行う．また，参照オントロジー中の概念に照合はしたが，意味的に一致する概念が存在しない入力語が存在する．そのような入力語は，入力概念選択時に「該当なし」を選択することによって，未定義語に分類され，参照オントロジー中の概念に照合しなかった入力語と同様に，ユーザが階層中の適切な位置に手動で階層関係の定義を行う．

入力概念選択の半自動化

入力語数が多い場合や入力語が多くの意味を持つ場合，入力概念選択はユーザの負担となる．入力概念選択モジュールでは，主に 2 種類の自動概念選択方法を用いて入力概念選択の支援を行う．両手法共に入力語に対応する概念候補の評価値を求めてランキングを行い，評価値の高い概念から順番にユーザに提示することにより，ユーザが入力概念選択を行うことを支援する．

一つ目の評価値の計算方法は以下のとおりである．

- 対象とする概念からそのルート概念までの各パスに出現する概念のうち，入力語集合（入力語彙）を見出しとして持つ概念の総数の最大値
- 対象とする概念の全ての下位概念のうち，入力語彙を見出しとして持つ概念の総数
- 対象とする概念の兄弟概念のうち，入力語彙を見出しとして持つ概念の総数

以上，三つの中からユーザは一つ以上の指標を選択し，選択した指標により得られた評価値の総和を用いて，入力語に対応する概念候補をランキングする．

二つ目の方法における評価値の計算方法は以下のとおりである．

1. 入力語に対応する概念候補となる概念集合を得る
2. 概念集合から二つの組み合わせを求め，それぞれの概念間距離を求める
3. ある概念と組み合わせ関係にある概念集合との概念間距離の逆数の総和をその概念の評価値とする

多重継承している場合には，概念間距離の計算方法が複数考えられる．その場合には，最短，最長，平均のどれかをユーザは選択することができる．

部分照合する用語の入力概念選択を簡略化するために，ある部分照合した用語の入力概念選択結果を，同様に部分照合するすべての用語の入力概念選択結果に反映させることが，入力概念選択モジュールでは可能である．例えば,「バッテリ充電装置」,「ノイズ測定装置」,「バルブ作動点検装置」がそれぞれ,「装置」で部分照合していた場合，装置の入力概念選択結果を，上記三つの入力語の入力概念選択結果とすることができる．

4.6 オントロジー構築モジュールの設計

オントロジー構築モジュールは，階層構築モジュールおよび関係構築モジュールから構成される．階層構築モジュールでは，参照オントロジーの概念階層を参照し，概念階層初期モデルを構築する（階層構築）．関係構築モジュールでは，入力文書および入力概念集合から，共起性に基づく手法により概念対集合を獲得する（関係構築）．概念階層初期モデルおよび概念対集合は，初期領域オントロジーであり，オントロジー洗練モジュールにおいてユーザインタラクションを通して洗練される．

以下では，階層構築モジュールおよび関係構築モジュールについて説明する．

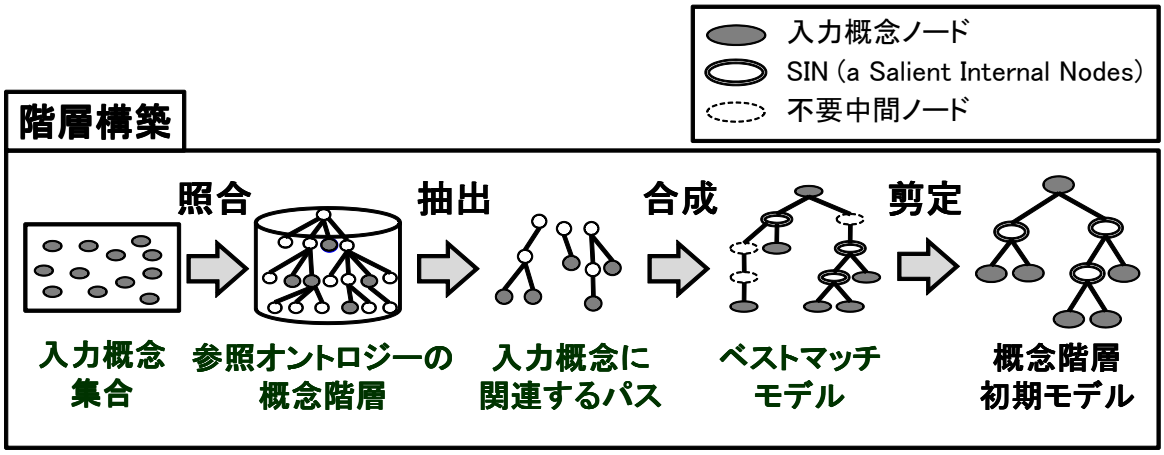


図 4.5: 完全照合概念の階層構築工程

4.6.1 階層構築モジュール

階層構築モジュールでは，参照オントロジーの概念階層を参照し，領域オントロジーの基礎となる概念階層初期モデルを構築する．入力モジュールにおいて，入力語と完全照合した入力概念（完全照合概念）と部分照合した入力概念（部分照合概念）により，階層構築方法が異なる．以下では，完全照合概念と部分照合概念のそれぞれについて，階層構築方法を説明する．

完全照合概念の階層構築

図 4.5 に完全照合概念の階層構築工程を示す．はじめに，参照オントロジーから，入力モジュールにより獲得した完全照合概念を末端ノードとするルート概念までのパスを抽出し，合成する．これをベストマッチモデルと呼ぶ．

図 4.5 のベストマッチモデルは，1 重線で囲まれたノードである入力概念ノード，2 重線で囲まれたノードである SIN (a Salient Internal Nodes)，点線で囲まれたノードである不要中間ノードの 3 種類のノードから構成される．入力概念ノードは，ユーザが選択した入力語に対応する参照オントロジー中の概念であり，領域にとって必須である．参照オントロジーから抽出したノードのうち，入力概念ノード以外のノードは SIN または不要中間ノードとなる．SIN は，入力概念ノードを一つ以上子ノードとして持つノードである．SIN は，各入力概念間の位相関係（祖先・親子・兄弟関係）を保持することに貢献する．一方，不要中間ノードは，入力概念ノードを子ノードとして持たないノードである．不要中間ノードは SIN とは異なり，各入力概念間の位相関係を保持することに貢献しないため，階層構築モジュールは階層構築において不要な概念であると見なし，ベストマッチモデルから削除する．不要中間ノードを削除する工程を剪定と呼ぶ．剪定によって得られた入力概念ノードと SIN のみから構成される概念階層を概念階層初期モデルと呼ぶ．概念階層初期モデルは，4.7.1 項で述べる概念階層洗練手法を用いて，ユーザとのインタラクションにより洗練され，最終的な領域オントロジーにおける概念階層となる．

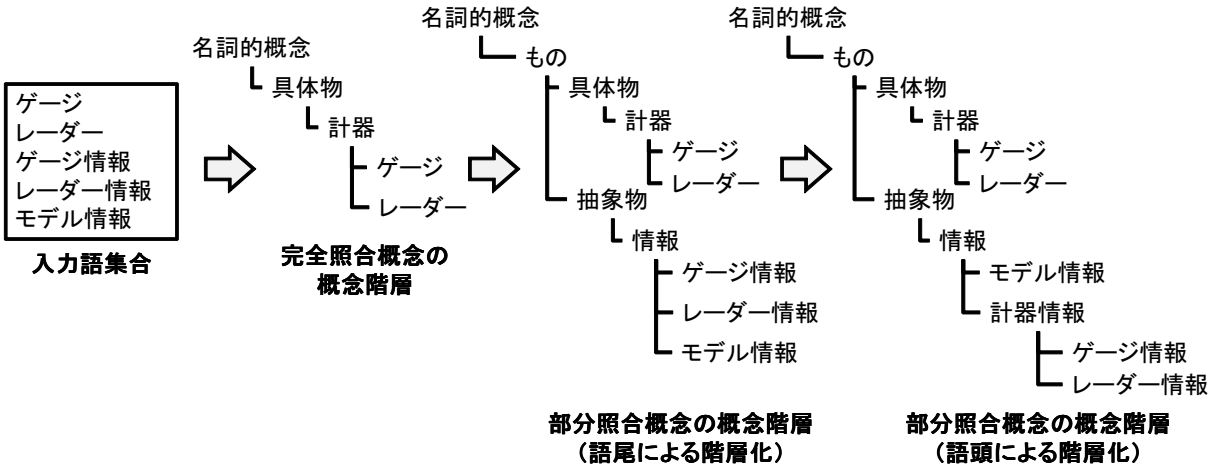


図 4.6: 部分照合概念の階層構築例

部分照合概念の階層構築

階層構築モジュールでは，部分照合概念について語尾および語頭による階層化を行う．図 4.6 に部分照合概念の階層構築例を示す．ここで，部分照合概念とは，参照オントロジー中の概念の見出しと部分的に照合する入力語を概念化したものである．4.5.3 項で説明したように，入力語が完全照合しなかった場合，入力語を形態素解析し，語尾を含むように部分照合を行っている．ここで，部分照合概念の見出しについて，語尾を含むように照合された部分を語尾部分，それ以前の部分を語頭部分と呼ぶことにする．例えば,「ゲージ情報」という入力語が参照オントロジー中の「情報」概念と部分照合した場合,「ゲージ」を語頭部分,「情報」を語尾部分と呼ぶ．また，4.5.3 項で説明したように，入力概念選択モジュールにおいて，ユーザは部分照合した入力語を照合した概念の別見出しとするか，下位概念とするかを選択する．ここでは，下位概念とするほうをユーザが選択したものとして説明する．

図 4.6 では，はじめに，ユーザは，入力語として「ゲージ」,「レーダー」,「ゲージ情報」,「レーダー情報」,「モデル情報」を選択した.「ゲージ」および「レーダー」は，参照オントロジー中にそれらを見出しとする概念が存在するため，図 4.5 に示した完全照合概念の階層構築工程に従って階層構築される.「ゲージ情報」,「レーダー情報」,「モデル情報」は，参照オントロジー中の「情報」概念と部分照合した．語尾による階層化により，はじめに,「情報」概念が完全照合概念の階層構築工程に従って階層構築され，次に,「ゲージ情報」,「レーダー情報」,「モデル情報」が概念化され,「情報」概念の下位概念として定義される．さらに，語頭による階層化では，部分照合概念の語頭部分に着目し，語頭部分を見出しとして持つ概念が構築中の概念階層内に存在する場合，その概念の上位概念と部分照合概念の語尾部分と照合した概念の見出しを組み合わせた見出しを持つ概念を新たに作成する．語頭部分が照合した部分照合概念は，新たに作成された概念の下位概念として階層関係が再定義される．部分照合概念の語頭部分は，部分照合概念を修飾していることが多い．そのため，語頭による階層化により，語尾による階層化のみに比べて，より詳細な階層構

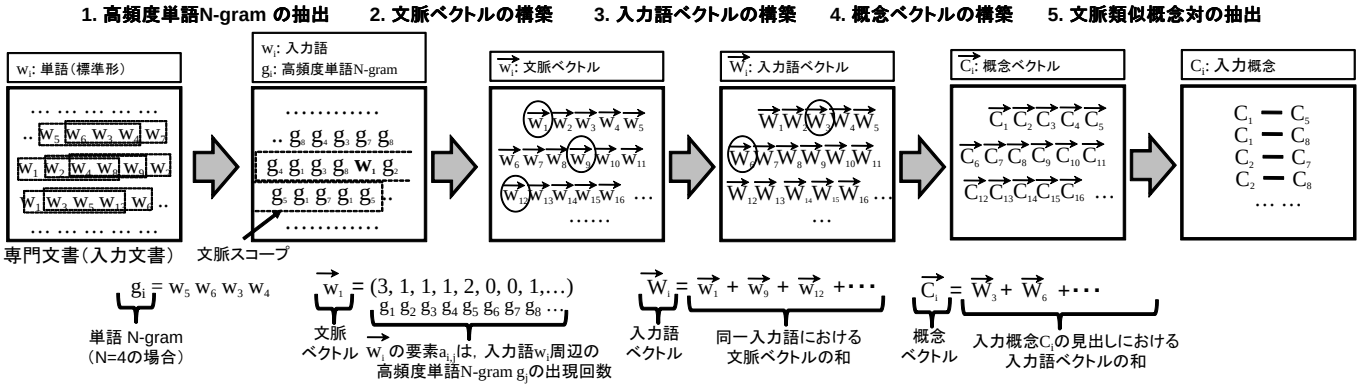


図 4.7: 文脈類似概念対の獲得手順

築を行うことができると考えられる．

図 4.6 の語尾による階層化により構築された概念階層では，部分照合概念である「ゲージ情報」概念および「レーダー情報」概念の語頭部分にあたる「ゲージ」および「レーダー」を見出しとして持つ,「ゲージ」概念および「レーダー」概念が「計器」概念の下位概念として定義されている．ここで，語頭による階層化により,「計器」概念と「情報」概念を組み合わせた「計器情報」概念が新規に作成され,「ゲージ情報」概念および「レーダー情報」概念の上位概念として，階層化が行われる.「計器情報」概念を定義することにより,「モデル情報」概念と「ゲージ情報」概念および「レーダー情報」概念という計器に関する情報を分類することができる．

4.6.2 関係構築モジュール

その他の関係の定義を支援するために，関係構築モジュールでは，WordSpace と相関ルールの二つの共起性に基づく手法を用いて，入力文書および入力語彙からその他の関係の候補となる概念対を獲得する．

WordSpace による概念対の抽出

共起統計の計算手法として WordSpace [55] を利用する．WordSpace とは，語彙の共起統計から大規模な単語群の意味表現を誘導するコーパスに基づく方法である．WordSpace によって，出現語句を共起情報を含むベクトルとして表現できる．この単語ベクトルの集合である多次元ベクトル空間が WordSpace であり，2 ベクトル間の内積は出現語句の文脈類似度の指標となる．WordSpace から得られる共起情報を基に，文脈類似概念対を入力文書から獲得し，その他の関係定義に関わる可能性のある概念対として利用する．“文脈の類似は，その語句間の何らかの概念関係の存在を示唆している” と仮定する．

以下では，WordSpace に基づく文脈類似概念対の獲得手順（図 4.7）について説明する．

1. **高頻度単語 N-gram の抽出**

   専門文書中から N 個の単語から構成される句（**単語 N-gram**）を抽出し，共起の最小単位として用いる．文字単位の N-gram 統計を取るのに比べ意味の無い文字列の共起情報を除外でき，より専門文書の文脈表現に役立つ情報が抽出できる[5]．この際抽出される句は，標準形に変換し，同形のものをまとめることで重複を排除している．ここで抽出された単語 N-gram 集合の中から，専門文書における出現頻度の高い単語 N-gram（高頻度単語 N-gram）を WordSpace の構築に用いる．これにより入力文書は高頻度単語 N-gram の配列とみなせる．関係構築モジュールでは，高頻度単語 N-gram を抽出する際に，単語 N-gram の単語数 N および出現数をユーザは設定することができる．

2. **文脈ベクトルの構築**

   次に，ある二つの入力語の文脈を比較するために，**文脈ベクトル (context vector)** を構築する．文脈ベクトルとは，ある入力語周辺の高頻度単語 N-gram の出現回数をベクトルで表現したものである．文脈ベクトル $\overrightarrow{w_i}$ の要素 $a_{i,j}$ は，入力語 w_i の出現場所周辺（**文脈スコープ**）の高頻度単語 N-gram g_j の出現回数である．関係構築モジュールでは，文脈スコープとして，入力語 w_i の前後何語以内に含まれる高頻度単語 N-gram を文脈ベクトルの構築に用いるかをユーザは設定することができる．

3. **入力語ベクトルの構築**

   次に，文脈ベクトルから入力語のベクトル表現である**入力語ベクトル (input term vector)** を導く．入力語ベクトル $\overrightarrow{W_i}$ は，専門文書において，入力語 w_i の全出現場所についての文脈ベクトル $\overrightarrow{w_i}$ の和によって表される．

4. **概念ベクトルの構築**

   次に，入力語ベクトルから入力概念のベクトル表現である**概念ベクトル (concept vector)** を導く．4.5.3 項で述べた入力概念選択モジュールによって，入力語に対応する参照オントロジー中の概念（入力概念）は特定されている．入力概念の見出し（入力語）における入力語ベクトルの和が概念ベクトルとなる．

   概念ベクトル $\overrightarrow{C}$ は，式 4.1 で表される．$\mathcal{A}(w)$ は，入力語 w の専門文書における全出現場所を表す．$\overrightarrow{w}(i)$ は，入力語 w の専門文書中の位置 i における文脈ベクトルを表す．$synset(C)$ は，概念 C における見出し集合を表す．

[5] [55] においては文字単位の共起を用いて WordSpace の構築を行っているが，関係構築モジュールでは単語単位 N-gram の共起を最小単位として扱う．従って，通常の WordSpace 構築時に文字単位共起をある程度まとまった形で表現するために行う 4-gram ベクトル構築工程は行わない．

$$\overrightarrow{C} = \sum_{w \in synset(C)} (\sum_{i \in \mathcal{A}(w)} \overrightarrow{w}(i)) \tag{4.1}$$

5. **文脈類似概念対の獲得**

以上の処理より，全入力概念について概念ベクトルを得ることができる．概念ベクトル間の内積は，概念間の文脈類似度となる．関係構築モジュールでは，文脈類似度に対してある一定の閾値をユーザは設定することができる．ユーザが指定した閾値を越える値を持つ概念対を文脈類似概念対として獲得する．

概念ベクトル $\overrightarrow{C_1}$，$\overrightarrow{C_2}$ 間の文脈類似度 $sim(\overrightarrow{C_1}, \overrightarrow{C_2})$ は，式 4.2 を用いて計算する．

$$sim(\overrightarrow{C_1}, \overrightarrow{C_2}) = \frac{\sum_i c_{1,i} c_{2,i}}{\sqrt{\sum_i {c_{1,i}}^2 \sum_i {c_{2,i}}^2}} \tag{4.2}$$

概念間の関係を明示する概念関係子は推定されていないため，推定前の初期値として概念関係子 ***non-TAXONOMY*** を割当てる．獲得された文脈類似概念対の中には，階層関係が含まれる可能性がある．そのため，概念階層において既に定義されている階層関係については，文脈類似概念対集合の中から除外する．

相関ルールによる概念対の抽出

専門文書からその他の関係定義の候補となる概念対を獲得するもう一つの方法として，相関ルールを利用する．相関とは，ある事象が発生すると別の事象が発生しやすいという共起性を意味する．また，$A \Rightarrow B$ という相関ルールは，A という事象が起こると B という事象も起こりやすいことを意味する．相関ルールの抽出は代表的なデータマイニング技術の一つであり，その他の関係定義にも利用されている [15]．ここでは，入力文書内の 1 文中に同時に出現する入力語の組み合わせを相関ルールとして抽出し，その他の関係定義の候補となる概念対として利用する．抽出された相関ルールに含まれる概念間に，何らかの概念関係が存在すると仮定する．

以下では，相関ルールの定義および相関ルール抽出アルゴリズム Apriori について述べる．相関ルールおよび Apriori アルゴリズムの説明は，データマイニングの基礎 [56] 2.5 節を参考にした．

相関ルールの定義　相関ルールは，式 4.3 に示す**トランザクション集合** (transaction set) T から抽出される．**トランザクション** (transaction) t_i は，データベース内でのデータのまとまりの単位を表す．ここでは，入力文書内の 1 文をデータのまとまりの単位としているため，トランザクション集合の要素数 n は，入力文書に含まれる文の数を表す．

$$T := \{t_i \mid i = 1 \ldots n\} \tag{4.3}$$

T の要素 t_i は，アイテム集合 (item set) である．ここでは，アイテムは入力語とする．つまり，t_i は，入力文書の i 番目の文に含まれる入力語の集合として表される．t_i は，式 4.4 で表される．式 4.4 の C は，入力文書に含まれる全入力語の集合を表す．

$$t_i = \{a_{i,j} \mid j = 1 \ldots m, a_{i,j} \in C\} \tag{4.4}$$

k 個のアイテムを含むアイテム集合 X_k と Y_k について，相関ルールは，$X_k \Rightarrow Y_k (X_k, Y_k \subset C, X_k \cap Y_k = \emptyset)$ で表される．ここで，X_k を条件部，Y_k を結論部と呼ぶ．条件部，結論部共に複数アイテムを含んでいてもよい．

相関ルールの重要性を測る指標として，**支持度** (support) と**確信度** (confidence) がある．

支持度とは，相関ルールが全トランザクションでどの程度出現するかを表す割合である．$X_k \Rightarrow Y_k$ の支持度 $support(X_k \Rightarrow Y_k)$ は，T の中で X_k と Y_k を共に含むトランザクションの割合により定義される（式 4.5）．

$$support(X_k \Rightarrow Y_k) = \frac{\mid \{t_i \mid X_k \cup Y_k \subseteq t_i\} \mid}{n} \tag{4.5}$$

確信度とは，条件部が起こったときに結論部が起こる割合である．$X_k \Rightarrow Y_k$ の確信度 $confidence(X_k \Rightarrow Y_k)$ は，T において X_k を含むトランザクションの中で，Y_k が出現する割合により定義される（式 4.6）．

$$confidence(X_k \Rightarrow Y_k) = \frac{\mid \{t_i \mid X_k \cup Y_k \subseteq t_i\} \mid}{\mid \{t_i \mid X_k \subseteq t_i\} \mid} \tag{4.6}$$

相関ルールの抽出では，支持度と確信度にある一定の閾値を設けないと，組み合わせ爆発を起こし，多数の無意味なルールが生成されてしまう．そのため，相関ルールの抽出では，支持度と確信度に閾値を設け，その値以上の支持度と確信度を有する相関ルールのみを抽出する．ここで，それぞれの閾値を**最小支持度** (minimum support)，**最小確信度** (minimum confidence) と呼ぶ．また，ユーザから与えられた最小支持度以上の支持度を有するアイテム集合を**多頻度アイテム集合** (frequent item set) と呼ぶ．

通常，相関ルールの条件部には複数のアイテムを許すが，ここでは概念対を抽出したいため，条件部と結論部共に一つずつのアイテム，つまり入力語の対を獲得する．WordSpace を用いた概念対の抽出と同様に，概念間の関係を明示する概念関係子は推定されていないため，初期値として概念関係子 ***non-TAXONOMY*** を割当てる．

相関ルール抽出アルゴリズム Apriori　相関ルールは，次の二つのステップにより抽出される．

ステップ 1:　多頻度アイテム集合 F を獲得する．

ステップ 2:　F から最小確信度以上の確信度を有する相関ルールを導出する．

ステップ 2 は，ステップ 1 により求めた F からルールを導出する処理であり，その負荷は比較的小さい．一方，ステップ 1 は，T を繰り返し検索し，数多くのアイテム集合の支

持度を調べるため，その負荷は大きい．そのため，ステップ 1 の効率の良いアルゴリズムを開発することが，実用的な相関ルール抽出アルゴリズムにつながると考えられてきた．この課題をはじめて解決した方法が，IBM アルマデン研究所の Rakesh Agrawal らによって提案された Apriori アルゴリズム [57] である．Apriori アルゴリズムは，現在最も広く利用されている相関ルール抽出アルゴリズムであり，本研究でも関係構築モジュールの実装に用いている．

以下では，Apriori アルゴリズムについて説明する．

Apriori アルゴリズムでは,「A が多頻度アイテム集合であれば，その部分集合 B は多頻度アイテム集合である」および，その対偶をとって「B が多頻度アイテム集合でなければ，B を含むような集合 A も多頻度アイテム集合でない」というアイテム集合の支持度の逆単調性を利用している．これらの性質を利用することにより，効率よく枝刈りを実行して，多頻度アイテム集合を求めることができる．例えば，$\{1,2\}$ が多頻度アイテム集合でなければ，$\{1,2\}$ を含むいかなるアイテム集合（$\{1,2,3\}$ など）も多頻度アイテム集合ではないため，その支持度を調べる必要はない．

Apriori アルゴリズムでは，要素数の少ないアイテム集合から支持度を計算し，あるアイテム集合の支持度が最小支持度より小さくなったとき，この逆単調性を利用して，そのアイテム集合を含むようなアイテム集合は，多頻度アイテム集合の候補とはせずに枝狩りする．

要素数 k の多頻度アイテム集合を F_k，多頻度アイテム集合の候補集合を C_k とする時，Apriori アルゴリズムの処理手順は以下のようになる．

(1) F_k から C_{k+1} を作成する．この際に，C_{k+1} の各要素について，要素数 k のアイテム集合からなる各部分集合がすべて F_k に含まれるかどうかを点検し，そうでなければその要素を C_{k+1} から削除する．

(2) T を検索し，C_{k+1} における各要素の支持度を求める．

(3) C_{k+1} から F_{k+1} を抽出する．

(4) 新たな多頻度アイテム集合が空となるまで，(1) から (3) の処理を繰り返す．

図 4.8 に，最小支持度 0.50 ($2/4 = 0.50$) における，Apriori アルゴリズムによる多頻度アイテム集合抽出の例を示す．図 4.8 では，T には四つのトランザクションが含まれているため，T の中で 2 回以上出現するアイテム集合が，多頻度アイテム集合となる．はじめに，T から要素数 1 のアイテム集合がトランザクションに含まれる回数を数え上げ，C_1 を作成する．C_1 の中から最小支持度以上の支持度を有するアイテム集合を抽出し，F_1 を求める．次に，F_1 から C_2 を作成する．ここでは，C_2 の各要素について，要素数 1 のアイテム集合からなる各部分集合は，すべて多頻度アイテム集合となるため，要素の削除は行われない．T を検索し，C_2 から F_2 を求める．次に，F_2 から C_3 を作成する．ここで，F_2 からは，$\{1,2,3\}$ および $\{1,3,5\}$ といったアイテム集合も C_3 の候補として抽出される．しかし，これらの部分集合である $\{1,2\}$ および $\{1,5\}$ は，それぞれ多頻度アイテム集合では

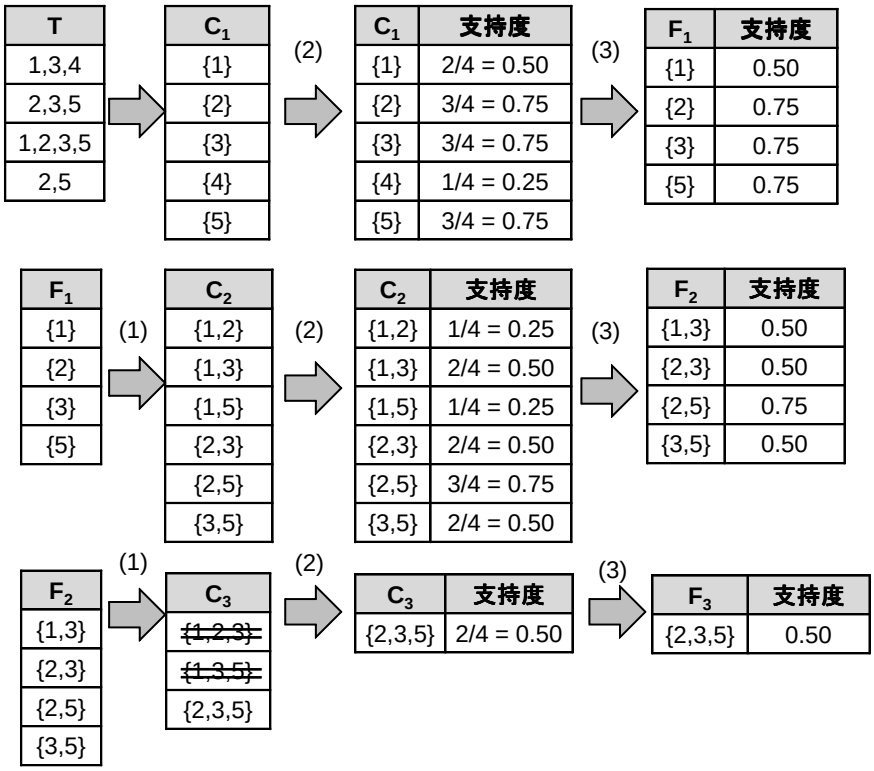


図 4.8: Apriori アルゴリズムによる多頻度アイテム集合抽出の例

ないため，$\{1,2,3\}$ および $\{1,3,5\}$ も多頻度アイテム集合ではないことがわかり，C_3 から削除される．よって，C_3 は $\{2,3,5\}$ のみとなる．T を検索すると，$\{2,3,5\}$ の出現数が 2 であり，支持度は 0.50 以上となる．よって，F_3 は $\{2,3,5\}$ となる．$\{2,3,5\}$ からは，C_4 を作成することができないため，ここで停止することとなる．

4.6.3 EDR 概念記述辞書を用いたプロパティ階層の構築およびその他の関係定義

オントロジー構築モジュールは，EDR 概念記述辞書を用いてプロパティ階層の構築およびその他の関係定義を行うことができる．EDR 概念記述辞書には動詞的概念が名詞的概念を支配する場合の格関係を中心に，`agent`，`object`，`goal`，`implement`，`a-object`，`place`，`scene`，`cause` の 8 種類の概念関係が定義されている．オントロジー構築モジュールは EDR 概念記述辞書に定義されている動詞的概念およびその下位概念を OWL におけるオブジェクトプロパティとみなし，階層構築時に名詞的概念階層（クラス階層）とは分離してプロパティ階層構築を行う．

また，オントロジー構築モジュールは，8 種類の概念関係のうち `agent` 関係がある名詞的概念をプロパティの定義域，`object` 関係がある名詞的概念をプロパティの値域として定義する．

プロパティ階層構築にも，クラス階層構築における完全および部分照合概念階層化と同様のアルゴリズムが適用可能である．完全照合概念を階層化する際には，不要概念の剪定が行われる．そのため，以下の場合にその他の関係定義の整合性が保持できなかったり，その他の関係定義が欠落してしまう問題が発生する．

1. クラス階層中の剪定された概念が agent または object の値として定義されている場合

2. プロパティ階層中の剪定された概念に agent または object 関係が定義されている場合

オントロジー構築モジュールでは，1. については，agent または object の値を，剪定された概念の下位概念に置換することで整合性を保持している．2. については，剪定されたプロパティの下位概念に定義域および値域を継承させることによりその他の関係定義が欠落しないようにしている．

4.7 オントロジー洗練モジュールの設計

オントロジー洗練モジュールは，階層洗練モジュールおよび関係洗練モジュールから構成される．オントロジー洗練モジュールでは，オントロジー構築モジュールで構築した**概念階層初期モデル**と，その他の関係定義のための**概念対集合**を基に，ユーザとのインタラクションを通してオントロジーの洗練を行う．

以下では，階層洗練モジュールおよび関係洗練モジュールについて説明する．

4.7.1 階層洗練モジュール

参照オントロジー（特に汎用オントロジー）から半自動構築された初期概念階層は一般的な階層関係が定義されているため，ユーザは概念変動（対象領域の変化による概念の意味変化）と呼ばれる問題を考慮しながら，初期概念階層を特定の領域に調整する必要がある．概念変動管理のために，階層洗練モジュールは戦略 1：照合結果分析，戦略 2：剪定結果分析，戦略 3：多重継承の除去の三つの戦略を適用する．図 4.9 に概念階層洗練工程を示す．戦略 1 は入力概念集合と汎用オントロジーとの照合結果の観点から, 戦略 2 は剪定結果の観点から，戦略 3 は多重継承から概念変動を同定する戦略である．以下では，それぞれの戦略の詳細を説明する．

戦略 1: 照合結果分析

戦略 1 では，概念階層初期モデルにおいて，入力概念の位置関係から再利用可能な領域と不可能な（概念変動が発生していると推定される）領域に分割し，再利用不可能な領域

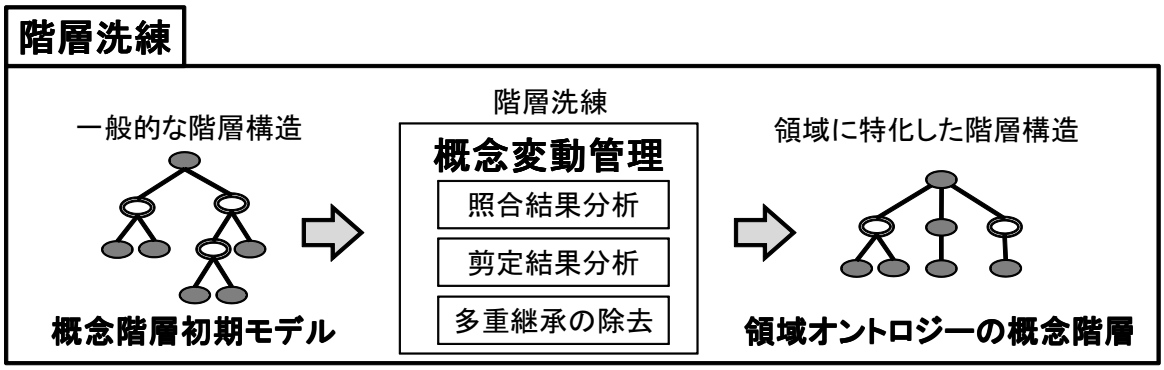


図 4.9: 概念階層洗練工程

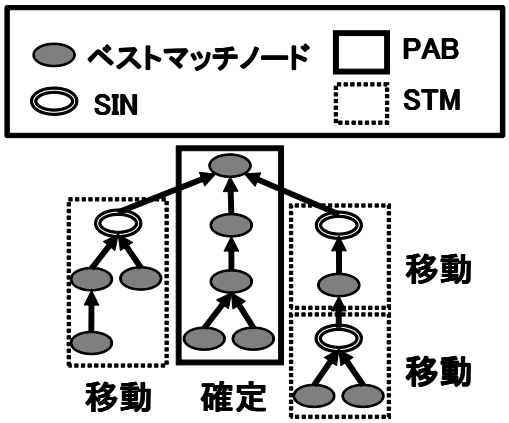


図 4.10: 戦略 1：照合結果分析

を移動することによって概念変動を解消する．ここで，移動するとは，再利用不可能な領域に含まれる概念を，他の適切な概念の下位概念として再定義することを意味する．

入力概念（ベストマッチノード）は，問題領域から考えてほぼ妥当と考えられた概念のため，それらが連続するパスは，妥当な概念が集中していると考え，再利用可能なパスとみなせる．このパスを **PAB (PAths including only Bestmatches)** と呼ぶ．一方，SIN が含まれる領域は，概念構造の差異（概念変動）が生じている可能性があるため，移動すべき領域とみなせる．この領域を **STM (SubTrees manually Moved)** と呼ぶ．PAB と STM の定義を以下に示す．

PAB の定義

ルート概念から入力概念（ベストマッチノード）が複数個連続しているパス．

STM の定義

SIN をサブルートとし，その下位ノードがすべてベストマッチノードで構成される部分木．

図 4.10 に PAB と STM の例を示す．実線で囲まれた部分木が PAB，破線で囲まれた部分木が STM である．ユーザーは STM を移動することで概念階層初期モデルを洗練し，領

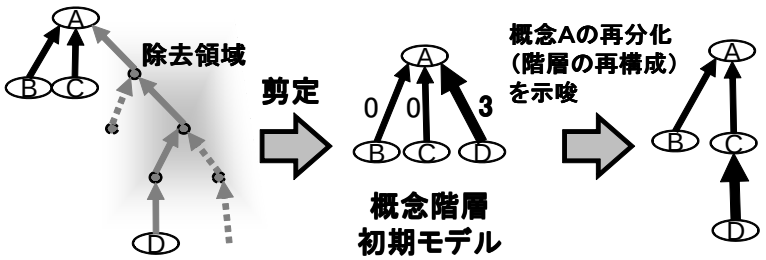


図 4.11: 戦略 2：剪定結果分析

域概念階層を構築する．STM の移動先についてはユーザが決定し，移動する必要がないと判断した場合は移動しない．移動時にユーザーが不必要と判断した STM のルートノードは削除してもよい．戦略 1 は，照合結果を分析することによって得られた戦略のため，**照合結果分析 (Matched Result Analysis: MRA)** と呼ぶ．

戦略 2: 剪定結果分析

戦略 2 では，概念階層初期モデルにおいて，同じ親ノード（上位概念）を持つ兄弟ノード間で，剪定において取り除かれた中間概念数の差が大きい場合，その階層関係を再構成するよう示唆する．

剪定工程で，削除された中間概念とそれにつながるベストマッチノード以外の概念を含む領域が全て削除されることは，参照オントロジーによる概念の分化の方法が問題領域の概念の分化の方法と異なっていることを示しているといえる．そのような部分木に対して分化の再構成をユーザに促す．剪定の際の削除数の差が概念階層初期モデルのルート概念から末端概念までの距離の 1/3 以上であった親子ノードに対し，再構成をユーザに示唆する．ルート概念から末端概念までの剪定の際の削除数は，ユーザによって任意に設定することもできる．戦略 2 は関連情報の剪定結果の分析によって行なわれる戦略のため，**剪定結果分析 (Trimmed Result Analysis: TRA)** と呼ぶ．

剪定結果分析の適用例を図 4.11 に示す．図 4.11 のベストマッチモデルを剪定した結果，概念 A と概念 D 間の領域が全て削除された．このような変化は概念 A の分類属性が，対象となる問題領域では異なった形で分化に利用されている可能性があることを意味し，ここに概念変動が発生していることが考えられる．この例では，対象となる問題領域では，概念 D は概念 A の下位概念ではなく，概念 C の下位概念として概念階層を再構成している．

戦略 3: 多重継承の除去

WordNet や EDR 電子化辞書などの汎用オントロジーは，網羅的に階層関係を定義するために，多重継承を多用している．汎用オントロジーにおける多重継承関係は，様々なコンテキストを考慮して定義されている．そのため，大部分の継承関係は特定の領域においては不要な継承関係となる．階層洗練モジュールでは，多重継承している概念の一覧を提

図 4.12: 戦略 3：多重継承の除去

示し，どの概念を上位概念として持つかをユーザに提示することにより，多重継承の除去を容易に行うことができる．

図 4.12 に多重継承の除去の例を示す． 図 4.12 では，汎用オントロジー中で概念 D は概念 A，概念 B，概念 C の 3 つの概念を上位概念として多重継承している．ここでは，概念 A と概念 C は上位概念として不要であるとみなし，ユーザが継承関係を除去している．

4.7.2 関係洗練モジュール

関係洗練モジュールでは，関係構築モジュールで WordSpace および相関ルールにより獲得した概念対集合から，ユーザが概念間関係を定義するのを支援する．関係洗練モジュールでは，WordSpace と相関ルールにおけるパラメータの調節や，結果の合成，正解または不要概念対の選択，概念対間の関係の定義を行うことができる．

4.8 視覚化モジュールの設計

セマンティック Web における領域オントロジーの質を高めるために，DODDLE-OWL には視覚化モジュールを導入している．視覚化モジュールには，3 章で述べた RDF(S) コンテンツ構築支援ツール MR^3 を利用している． MR^3 は RDF コンテンツと RDFS コンテンツをモデルとオントロジーの関係としてとらえ，両者の視覚的編集および一貫性を（半）自動的に管理するツールである． MR^3 のプラグイン機構を用いて，DODDLE-OWL と相互に OWL データの交換を行う．

領域オントロジー構築における視覚化モジュールの役割は二つある．一つ目は階層洗練モジュールにおける概念変動管理を視覚的に支援する機能である．視覚化モジュールのクラスエディタおよびプロパティエディタに，階層構築モジュールで構築された概念階層初期モデルを表示し，ユーザは階層洗練モジュールにより同定された概念変動が生じている可能性のある概念階層の部分を編集することができる．二つ目はオントロジーの外在化である．オントロジーの外在化とは階層関係とその他の関係を DODDLE-OWL 以外の見方によって視覚的に表示することを意味する．DODDLE-OWL 以外の見方の例として，階層関係のグラフ表示やオントロジーとインスタンスの関係を同時に見ることができることがあげられる．階層関係をグラフ表示することにより，多重継承関係をユーザが把握しやすくなる．また，オントロジーとインスタンスの関係を同時に見ることで，クラスおよびプロパティ定義の不足や誤りを発見しやすくなる．オントロジーの外在化を行うこと

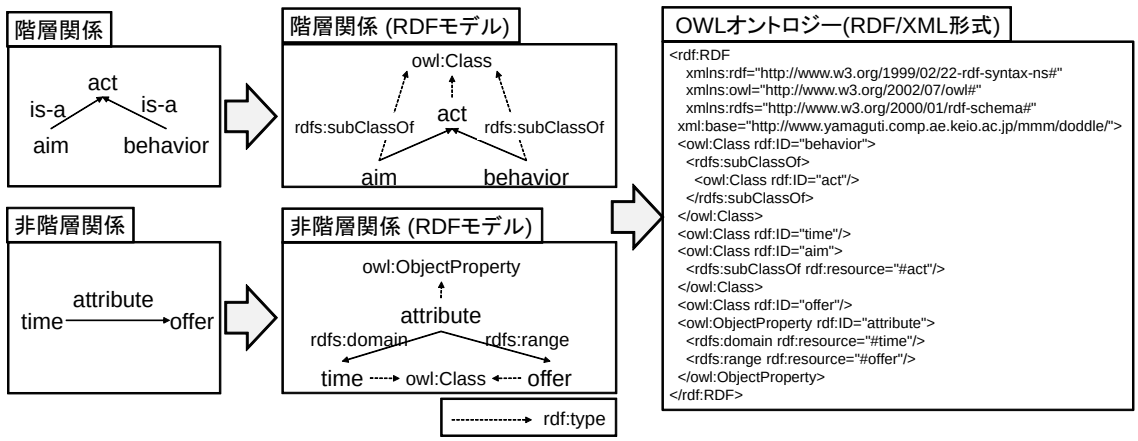


図 4.13: 変換モジュールによる領域オントロジーの OWL 形式への変換例

によって，オントロジー全体（クラス階層，プロパティ階層，その他の関係，インスタンス）のバランスを見ながら領域オントロジーの調整を行い，領域オントロジーの質を向上させることができる．

4.9 変換モジュールの設計

DODDLE-OWL によって構築される領域オントロジーは，階層関係とその他の関係から構成される．クラスの is-a 階層は，OWL が提供する `owl:Class` クラスおよび `rdfs:subClassOf` プロパティにより定義する．クラスの has-a 階層は，`owl:Class` クラスおよび `doddle:partOf` プロパティにより定義する．プロパティの is-a 階層は，`owl:ObjectProperty` クラスおよび `rdfs:subPropertyOf` プロパティにより定義する．プロパティの has-a 階層は，`owl:ObjectProperty` クラスおよび `doddle:partOf` プロパティにより定義する．その他の関係は，概念対の間の関係を OWL におけるプロパティ，概念対をプロパティの定義域および値域としてとらえ，OWL が提供する `owl:ObjectProperty` クラス，`rdfs:domain` および `rdfs:range` プロパティにより定義する．

図 4.13 の上部は，概念関係の定義の例として,「act」クラスの下位クラスとして「aim」と「behavior」クラスが定義された状態を，OWL 形式に変換する方法を示している．図 4.13 の下部は，その他の関係の定義の例として,「time」と「offer」クラスの間に「attribute」プロパティという関係がある状態を，OWL 形式に変換する方法を示している．

また，DODDLE-OWL では概念の見出しを `rdfs:label` プロパティ，概念の説明を `rdfs:comment` プロパティ，概念の表示見出しを `skos:prefLabel` プロパティを用いて定義している．概念の表示見出しは，概念に複数の見出しが定義されている場合に，概念階層を表示する際に優先的に表示する見出しのことである．

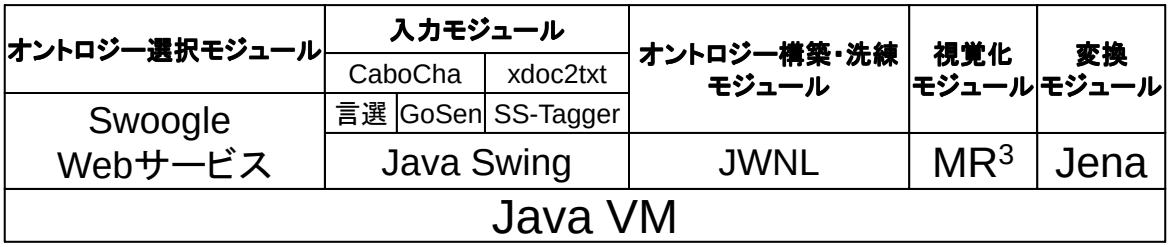


図 4.14: DODDLE-OWL の実装アーキテクチャ

4.10 領域オントロジー構築支援環境の実装

4.10.1 実装の概要

4.3 節から 4.9 節で述べた領域オントロジー構築支援環境の設計を基に，DODDLE-OWL (a Domain Ontology rapiD DeveLopment Environment - OWL extension) を実装した．図 4.14 に DODDLE-OWL の実装アーキテクチャを示す．DODDLE-OWL は，GUI コンポーネントとして Java Swing を用いて，Java 言語で実装した．DODDLE-OWL は，オントロジー選択モジュール，入力モジュール，オントロジー構築・洗練モジュール，視覚化モジュール，変換モジュールから構成される．実装上は，オントロジー構築およびオントロジー洗練は，同一パネル上で操作できるようにしている．

Web 上の既存オントロジーを獲得するために，オントロジー選択モジュールでは Swoogle Web サービスを利用している．入力モジュール，オントロジー構築・洗練モジュールでは，WordNet を参照するために Java WordNet Library(JWNL) [6]を利用している．入力モジュールでは，日本語の形態素解析および品詞同定を行うために，日本語形態素解析器 GoSen [7]を用いている．英語の品詞同定を行うために英語タガー SS-Tagger [58] を用いている．英語および日本語の複合語を抽出するために専門用語自動抽出システム言選 [54] を用いている．日本語の複合語抽出には，言選以外に日本語係り受け解析器 CaboCha[8]を用いることもできる．xdoc2txt [9]と連携することにより，テキスト文書のみでなく，PDF, Microsoft Word, Excel, PowerPoint など様々な形式のファイルからテキストを抽出することができる．視覚化モジュールには MR^3 を用いている．変換モジュールでは，OWL 形式のオントロジーのインポートおよびエクスポートを支援するために，Jena セマンティック Web フレームワーク [10]を用いている．

表 4.3 に DODDLE-OWL の各モジュールのソースコードの行数を示す[11]．

以下では，各モジュールの実装の詳細について説明する．

[6] http://jwordnet.sourceforge.net/
[7] http://itadaki.org/wiki/index.php/GoSen
[8] http://chasen.org/ taku/software/cabocha/
[9] http://www31.ocn.ne.jp/ h_ishida/xdoc2txt.html
[10] http://jena.sourceforge.net/downloads.html
[11] ソースコードの行数は，Eclipse Metrics Plugin 1.3.6 を用いて数えた．
URL: http://metrics.sourceforge.net/

表 4.3: DODDLE-OWL のソースコードの規模

モジュール名	行数
オントロジー選択モジュール	4127
入力モジュール	7016
オントロジー構築・洗練モジュール	8261
変換モジュール	752
その他	5648
合計	25804

表 4.4: 領域オントロジー構築支援に利用可能な Swoogle Web サービスとその入出力

タイプ	Swoogle Web サービス	入力	出力
(1)	Search ontology	検索キーワード	検索キーワードに関連する SWO のリスト
(3)	Search terms	検索キーワード	検索キーワードに関連する SWT のリスト
(4)	Digest semantic web document	SWD	SWD の Swoogle メタデータ
(13)	List documents using term	SWT	SWT を定義，参照，populate している SWD のリスト
(16)	List domain classes of a property	プロパティ	入力したプロパティの定義域のリスト
(17)	List properties of a domain class	クラス	入力したクラスを定義域とするプロパティのリスト
(18)	List range classes of a property	プロパティ	入力したプロパティの値域のリスト
(19)	List properties of a range class	クラス	入力したクラスを値域とするプロパティのリスト

4.10.2 オントロジー選択モジュールの実装

Swoogle を用いた既存オントロジーの獲得

本項では，4.4.1 項で述べたオントロジー検索エンジンを用いた既存オントロジーの獲得の実装について述べる．

オントロジー検索エンジン Swoogle は 19 種類の REST 形式の Web サービス（Swoogle Web サービス）を提供している．ユーザは URL を用いてクエリーを作成し，RDF/XML 形式の検索結果を得ることができる．表 4.4 に領域オントロジー構築支援に利用可能な Swoogle Web サービスとその入出力を示す．表 4.4 の SWT (Semantic Web Term) はクラスまたはプロパティを表す．SWD (Semantic Web Document) は RDF/XML，N-Triple, N3 形式で記述された RDF 文書を表す．SWO (Semantic Web Ontology) はクラスおよびプロパティの定義の割合が 8 割以上の SWD を表す．

Swoogle Web サービスを呼びだす際には，`queryType`, `searchString`, `key` の主に三つのパラメータを与える．`queryType` には，呼びだす Web サービスを特定するためのサービスタイプを指定する．`searchString` には，サービスの入力となる検索キーワードなどを与える．`key` は出力する検索結果数に制限をかけるために用いる．

表 4.5 は，4.4.1 項，図 4.2 で示した既存オントロジー獲得の手順 1 から 4 の各手順で利用する Swoogle Web サービスのタイプおよび実行条件を示す．表 4.5 の手順は，4.4.1 項，図 4.2 の手順と一致している．表 4.5 の各手順で利用する Swoogle Web サービスのタイプ

表 4.5: 既存オントロジー獲得の各手順で利用する Swoogle Web サービスのタイプおよび実行条件

手順	各手順で利用する Swoogle Web サービスのタイプ	実行条件
1	(3)	各入力語について，獲得するクラスおよびプロパティ数は， TermRank によりランク付けされた上位 5 個までとする．
2	(17), (19)	手順 1 で獲得したクラスを rdfs:domain または rdfs:range プロパティの値として持つ プロパティの獲得数は，各クラスごとに上位 100 個までとする．
3	(16), (18)	手順 1 および 2 で獲得したプロパティの定義域および値域の獲得数は， 各プロパティごとに上位 100 個までとする．
4	(1), (4), (13)	各入力語について獲得するオントロジー数は， OntoRank でランク付けされた上位 10 個までとする．

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX daml03: <http://www.daml.org/2001/03/daml+oil#>
PREFIX daml10: <http://www.w3.org/2001/10/daml+oil#>

SELECT ?class
WHERE {
   {?class rdf:type rdfs:Class} UNION {?class rdf:type owl:Class} UNION  {?class rdf:type owl:Restriction} UNION {?class rdf:type owl:DataRange} UNION
   {?class rdf:type daml03:Class} UNION {?class rdf:type daml03:Datatype} UNION {?class rdf:type daml03:Restriction} UNION
   {?class rdf:type daml10:Class} UNION {?class rdf:type daml10:Datatype} UNION {?class rdf:type daml10:Restriction}
}
```

図 4.15: RDFS，DAML，OWL 基本語彙におけるクラス抽出テンプレート

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl:  <http://www.w3.org/2002/07/owl#>
PREFIX daml03: <http://www.daml.org/2001/03/daml+oil#>
PREFIX daml10: <http://www.w3.org/2001/10/daml+oil#>

SELECT ?property
WHERE {
   {?property rdf:type rdf:Property} UNION {?property rdf:type owl:ObjectProperty} UNION  {?property rdf:type owl:DatatypeProperty} UNION
   {?property rdf:type owl:AnnotationProperty} UNION {?property rdf:type owl:FunctionalProperty} UNION {?property rdf:type owl:InverseFunctionalProperty} UNION
   {?property rdf:type owl:SymmetricProperty} UNION {?property rdf:type owl:OntologyProperty} UNION {?property rdf:type owl:TransitiveProperty} UNION
   {?property rdf:type daml03:Property} UNION {?property rdf:type daml03:ObjectProperty} UNION {?property rdf:type daml03:DatatypeProperty} UNION
   {?property rdf:type daml03:TransitiveProperty} UNION {?property rdf:type daml03:DatatypeProperty} UNION {?property rdf:type daml03:UniqueProperty}  UNION
   {?property rdf:type daml10:Property} UNION {?property rdf:type daml10:ObjectProperty} UNION {?property rdf:type daml10:DatatypeProperty} UNION
   {?property rdf:type daml10:TransitiveProperty} UNION {?property rdf:type daml10:DatatypeProperty} UNION {?property rdf:type daml10:UniqueProperty}
}
```

図 4.16: RDFS，DAML，OWL 基本語彙におけるプロパティ抽出テンプレート

は，表 4.4 のタイプの番号と一致している．また，計算時間を削減するために，各手順において実行条件を設定している．

SPARQL テンプレートを用いたオントロジー要素抽出

図 4.15 から図 4.19 に RDFS，DAML，OWL 語彙におけるオントロジーの要素を抽出するための SPARQL で記述したテンプレートを示す．図 4.17 の見出しと説明抽出テンプレートを直接 SPARQL のクエリーとした場合，OWL オントロジー中のすべての rdfs:label および rdfs:comment プロパティの値を抽出してしまう．オントロジー選択モジュールでは

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX daml03: <http://www.daml.org/2001/03/daml+oil#>
PREFIX daml10: <http://www.w3.org/2001/10/daml+oil#>

SELECT ?label ?description
WHERE {
  {?concept rdfs:label ?label} UNION {?concept rdfs:comment ?description} UNION  {?concept daml03:label ?label} UNION
  {?concept daml03:comment ?description} UNION {?concept daml10:label ?label} UNION  {?concept daml10:comment ?description}
}
```

図 4.17: RDFS，DAML，OWL 基本語彙における見出しおよび説明抽出テンプレート

```
PREFIX  rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX daml03: <http://www.daml.org/2001/03/daml+oil#>
PREFIX daml10: <http://www.w3.org/2001/10/daml+oil#>

SELECT ?subConcept
WHERE {
  {?subConcept rdfs:subClassOf ?concept} UNION {?subConcept rdfs:subPropertyOf ?concept} UNION {?subConcept daml03:subClassOf ?concept} UNION
  {?subConcept daml03:subPropertyOf ?concept} UNION {?subConcept daml10:subClassOf ?concept} UNION {?subConcept daml10:subPropertyOf ?concept}
}
```

図 4.18: RDFS，DAML，OWL 基本語彙における階層関係抽出テンプレート

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX daml03: <http://www.daml.org/2001/03/daml+oil#>
PREFIX daml10: <http://www.w3.org/2001/10/daml+oil#>

SELECT ?property ?domain ?range
WHERE {
  {?property rdfs:domain ?domain} UNION  {?property rdfs:range ?range} UNION {?property daml03:domain ?domain} UNION
  {?property daml03:range ?range} UNION {?property daml10:domain ?domain} UNION {?property daml10:range ?range}
}
```

図 4.19: RDFS，DAML，OWL 基本語彙におけるその他の関係抽出テンプレート

?concept 変数の部分を取得したい概念（クラスまたはプロパティ）の URI に置換することにより，特定の概念の見出しおよび説明のみを抽出できるようにしている．他のテンプレートも同様にテンプレートを直接 SPARQL のクエリーとして用いるのではなく，変数部分をオントロジー選択モジュールが適切な URI に置換したものを最終的な SPARQL のクエリーとしている．?concept，?subConcept，?class，?property，?label，?description，?domain，?range 変数を用いてトリプルのパターンを各オントロジーの要素を抽出するテンプレートに記述し，テンプレートを OWL オントロジーに対応づけることで，様々なクラス，プロパティ，構造により表現されたオントロジーの要素を抽出することが可能となる．

汎用オントロジー選択パネル

図 4.20 に汎用オントロジー選択パネルを示す．図 4.20(1) に示す，3 種類の汎用オントロジー（EDR 一般辞書，EDR 専門辞書，WordNet）の中から参照オントロジーを選択する．チェックボックスにチェックをつけた汎用オントロジーを用いて，その後領域オントロジーにおける概念階層を構築する．複数の汎用オントロジーが選択可能な利点として

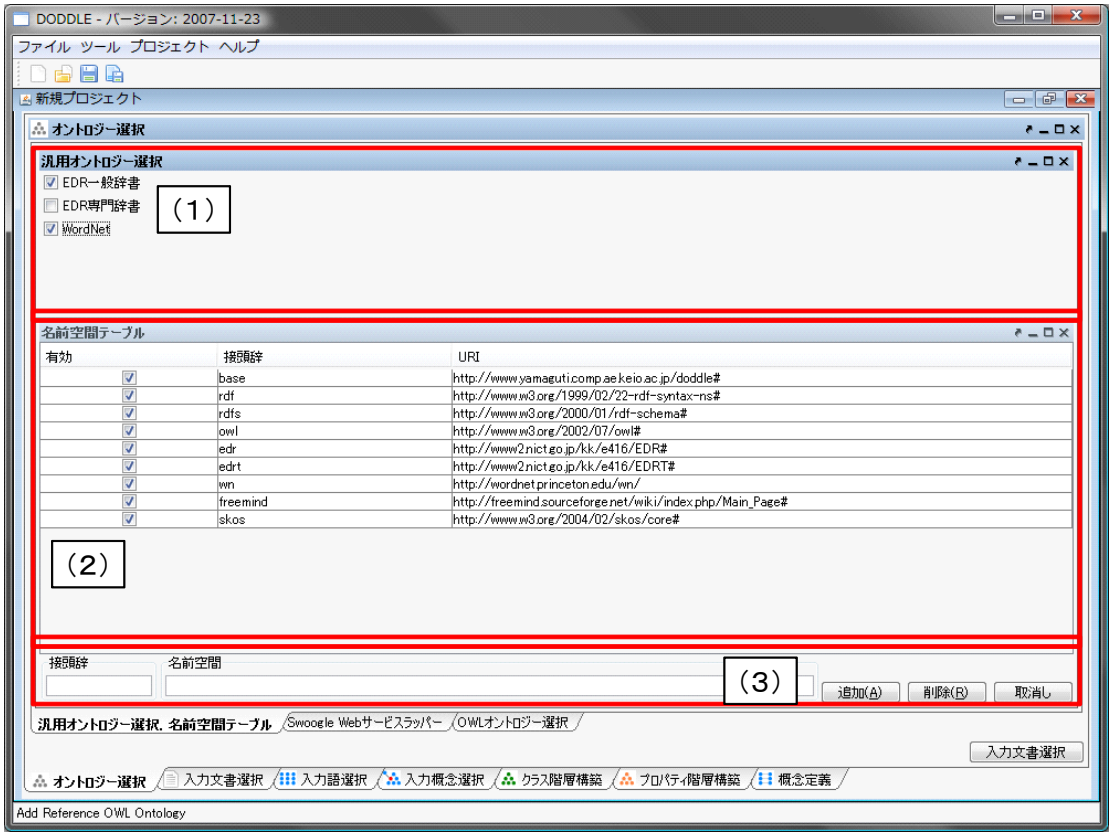


図 4.20: 汎用オントロジー選択パネル

は，領域によっては，一つの汎用オントロジーだけでは語彙を網羅しきれない場合があるため，複数の汎用オントロジーを組み合わせて利用できるようにしている．図 4.20(2) の名前空間テーブルは，名前空間 URI とその名前空間接頭辞の対応関係を管理している．図 4.20(3) に接頭辞と名前空間を入力し，図 4.20(3) 右側の「追加」ボタンで追加することができる．

OWL オントロジー選択パネル

図 4.21 に OWL オントロジー選択パネルを示す．図 4.21(1) の「追加（ファイル）」または「追加 (URI)」ボタンにより，参照オントロジーとする既存 OWL オントロジーを選択する．図 4.21(3) には，図 4.21(1) のオントロジーリスト中で選択したオントロジーの OWL メタデータが表示される．また，図 4.21(2) において，OWL オントロジー中から抽出する要素を決定するための SPARQL テンプレートを指定する．SPARQL テンプレートの種類として，4.10.2 項で述べた 5 種類が利用できる．

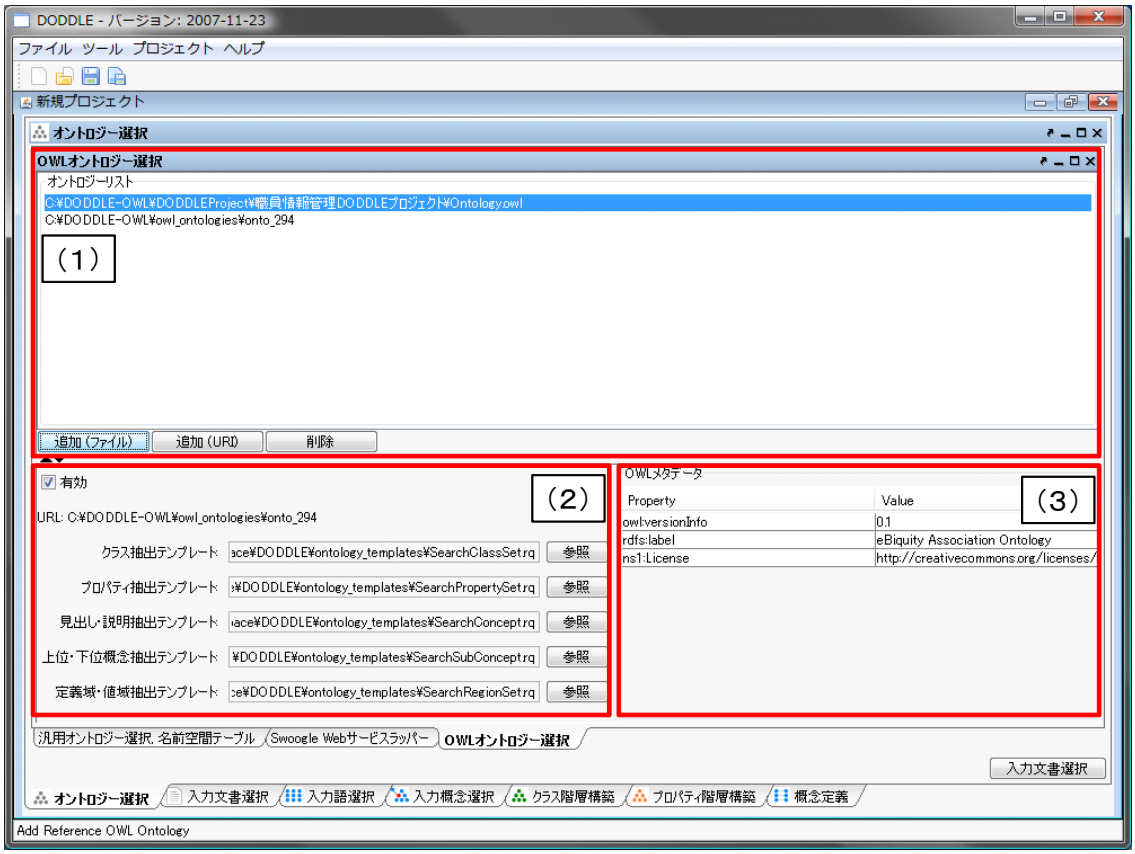


図 4.21: OWL オントロジー選択パネル

4.10.3 入力文書選択モジュールの実装

図 4.22 に入力文書選択パネルを示す．入力文書選択パネルでは，領域に関連する英語または日本語で記述された文書を選択する．入力文書選択パネルでは，xdoc2txt というツールを用いて，様々な形式（Word, Excel, PowerPoint, PDF など）のファイルからテキストデータを抽出できる．単語を抽出する際には，抽出する単語の品詞を指定できるようにしている．名詞，動詞，その他の品詞，複合語のいずれかを抽出したり，1 文字だけの領域オントロジー構築に不要となる語を除去することができる．以下に図 4.22 の各部分について説明する．

(1) 入力文書のリストを表示する．

(2) 入力文書の記述言語（日本語または英語）の選択と入力文書の追加および削除を行う．

(3) 1 文の区切り文字を設定する．

(4) (1) の入力文書リストの中から選択された文書の内容を表示する．

(5) 抽出する語の品詞，複合語を抽出するかどうか，1 文字の語を抽出するかどうかを選択する．

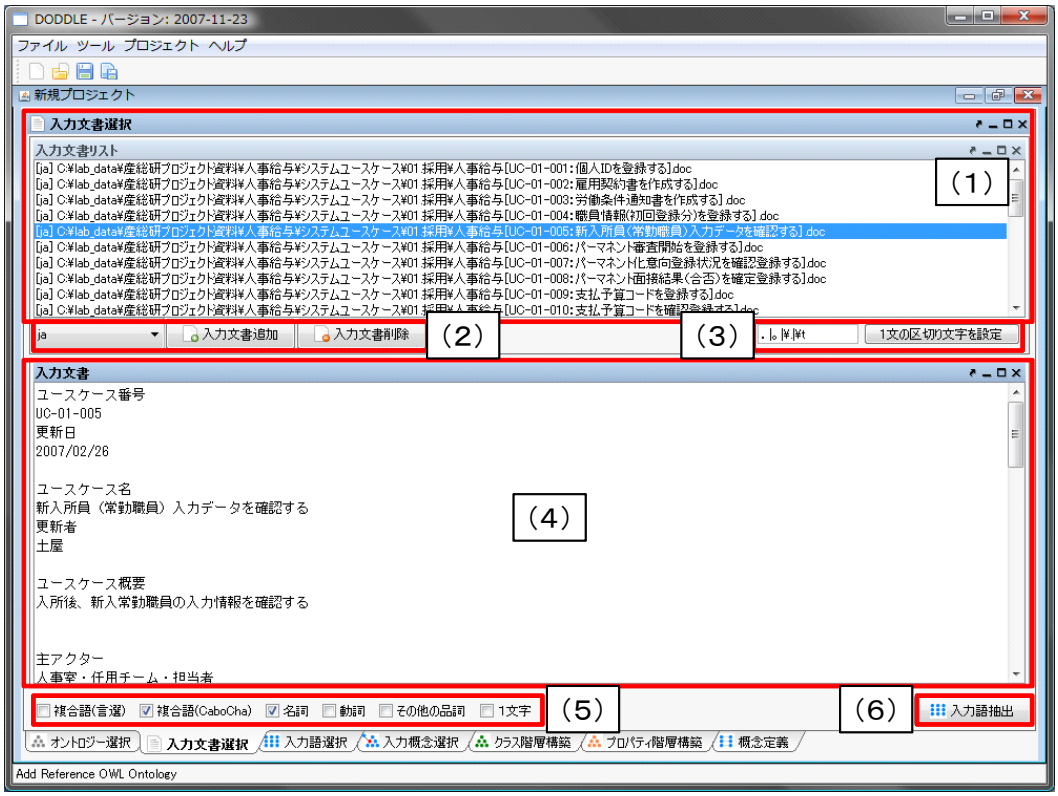


図 4.22: 入力文書選択パネル

(6) (1) の入力文書リストで選択された文書中から (5) で指定した条件の語を抽出する．

4.10.4 入力語選択モジュールの実装

入力語選択モジュールの実装として，入力文書ビューア，入力語情報テーブル，削除語情報テーブルを実装した．

入力文書ビューア

入力文書ビューアでは，入力文書の内容を見ながらユーザは入力語の選択を行うことができる．図 4.23 に入力文書ビューアのスクリーンショットを示す．以下では，入力文書ビューアの各部分について説明する．

(1) 入力文書リストを表示する．

(2) (1) で選択した入力文書の内容を (3) に表示する際に，何行目から何行目までを表示するかを選択する．

(3) (1) で選択した入力文書の内容を表示する．表示される行範囲は (2) で選択される．入力文書中のハイパーリンクが張られている語をクリックすることで，入力語か不

図 4.23: 入力文書ビューア

要語かを選択することができる．青色リンクは入力語を，灰色リンクは不要語を表している．

(4) (3) のハイパーリンクにマウスカーソルを合わせた際に，ハイパーリンクが張られている語の用語名，品詞，TF，IDF，TF-IDF，上位概念が表示される．

(5) (1) で選択した入力文書の内容を分割して (3) に表示する際の分割行数を設定する．

(6) 自動用語抽出により，抽出できなかった用語を手動で追加することができる．(3) において用語を範囲選択し，マウスを右クリックすることでも，同様に手動で用語を追加することができる．追加された用語は，(3) において青色のハイパーリンクが張られる．

(7) (3) に表示される入力文書の内容にハイパーリンクを張る用語の種類（複合語，名詞，動詞，その他の品詞）を選択する．

入力語情報テーブル

入力語情報テーブルでは，入力文書から自動抽出された語から入力語を選択することができる．図 4.24 に入力語情報テーブルのスクリーンショットを示す．以下では，入力語情報テーブルの各部分について説明する．

図 4.24: 入力語情報テーブル

(1) ユーザが入力した用語で (3) に表示する用語情報リストを絞り込む.

(2) ユーザが入力した品詞で (3) に表示する用語情報リストを絞り込む.

(3) 入力文書から自動抽出された用語情報を表示する．用語情報には，用語名，品詞，TF，IDF，TF-IDF，上位概念があり，それぞれの観点からリストをソートすることができる．抽出された語が，あらかじめユーザが用意した参照オントロジー中の概念の下位概念の見出しに含まれる場合，その概念の見出しを上位概念に表示する．概念階層中の上位概念を設定しておくことで，抽出された語を「もの」「場所」「時間」などに分類して表示することができ，入力語選択を支援することができる．

(4) (3) の中で選択された用語情報の用語の入力文書中の出現箇所を表示する.

(5) 最終的にユーザが決定した入力語のリスト．テキストエリアになっているため，入力文書に出現しなかった入力語の追加をユーザは行うことができる.

(6) 「入力語リストに追加」ボタンを押すと，(3) の中で選択された行の用語を (5) の入力語リストに追加する.「削除」ボタンを押すと，(3) の中で選択された用語情報の用語を「削除語テーブル」に移す.

図 4.25: 削除語情報テーブル

(7) (5) に入力された入力語を設定し，入力概念選択パネルに移る.「入力語彙をセット」ボタンを押した場合は，新規に入力語リストを入力概念選択パネルに設定する.「入力語彙を追加」ボタンを押した場合は，設定済みの入力語リストに新たに入力語を追加する.

削除語情報テーブル

削除語情報テーブルには，入力語情報テーブルから削除された用語情報のリストが表示される．図 4.25 に削除語情報テーブルのスクリーンショットを示す．削除語情報テーブルの各部分は，入力語情報テーブルと同様である．異なる点は,「戻す」ボタンと「完全削除」ボタンである.「戻す」ボタンにより，誤って削除語情報テーブルに移動させてしまった用語情報を入力語情報テーブルに戻すことができる.「完全削除」ボタンにより，用語情報をリストから完全に削除することができる.

4.10.5 入力概念選択モジュールの実装

入力概念選択モジュールの実装として，入力概念選択パネルを実装した．図 4.26 に入力概念選択パネルを示す．入力概念選択パネルでは，入力語と参照オントロジー中の概念との対応付けを行う．語には多義性があり，ある入力語を見出しとして持つ概念が複数存

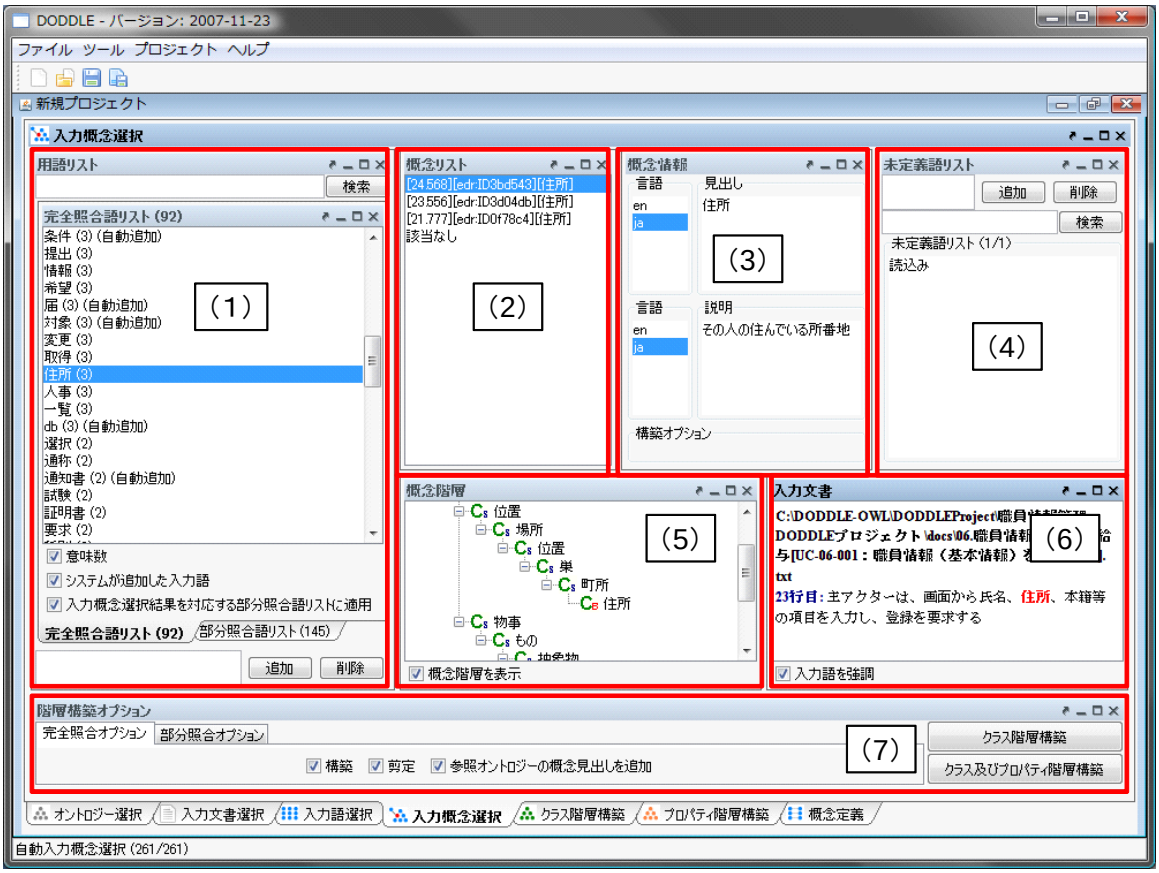


図 4.26: 入力概念選択パネル

在する可能性がある．入力概念選択パネルでは，対象領域にとって最も適切な入力語に対応する概念を選択する際の支援を行う．以下に入力概念選択パネルのの各部分の説明を示す．

(1) 用語リスト

入力語彙の中で参照オントロジー中の概念見出しと完全照合または部分照合した用語のリストを表示する．

(2) 概念リスト

(1) で選択された語を見出しとしてもつ参照オントロジー中の概念のリストを表示する．

(3) 概念情報

(2) で選択された概念の見出しおよび説明を言語ごとに分類して表示する．

(4) 未定義語リスト

参照オントロジー中の概念の見出しと照合しなかった入力語（未定義語）を表示する．

(5) 概念階層

(2) で選択された概念の参照オントロジー中の階層階層を表示する．

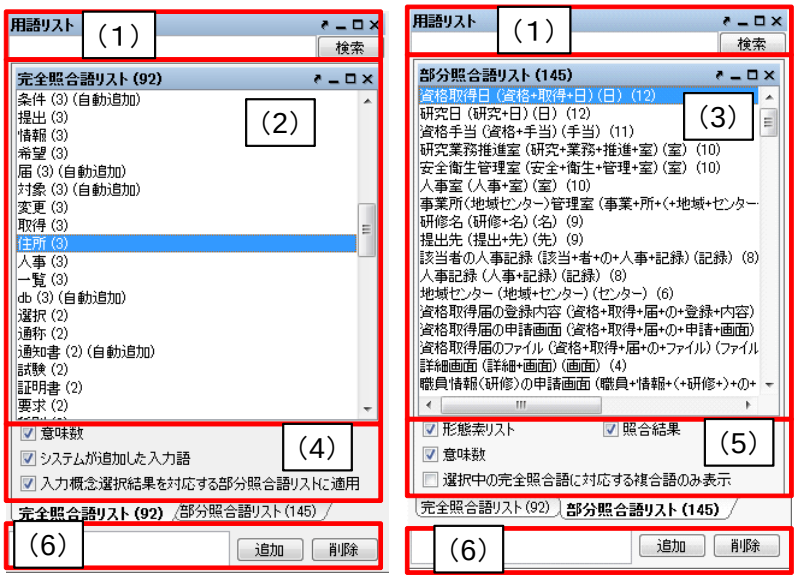


図 4.27: 入力概念選択パネル：用語リスト

(6) 入力文書

(1) で選択された語の入力文書中の出現箇所を表示する．

(7) 階層構築オプション

階層構築における条件を設定する．

用語リスト

図 4.27 は図 4.26(1) 用語リストを拡大した図である．以下では，入力概念選択パネルの用語リストの各部分について説明する．

(1) テキストフィールドに検索キーワードを入力し，検索ボタンを押すと (2) および (3) の完全照合語リストおよび部分照合語リストに検索キーワードを含む入力語のみが表示される．

(2) 完全照合語リストを表示する．1 番目の括弧内には，入力語を見出しとする参照オントロジー中の概念の数が表示される．入力語がシステムが自動的に追加した場合には，2 番目の括弧内に「自動追加」と表示される．

(3) 部分照合語リストを表示する．1 番目の括弧内には，部分照合語を形態素解析し，各形態素を「+」記号で結合した結果が表示される．2 番目の括弧内には，参照オントロジー中の概念の見出しと照合した部分照合語内の語が表示される．3 番目の括弧内には，2 番目の括弧内に表示された語を見出しとする参照オントロジー中の概念の数が表示される．

(4) 完全照合語リストに関する設定を行うことができる．

「意味数」チェックボックスは，完全照合語リスト中の各語を見出しとする参照オントロジー中の概念の数を表示するかどうかを設定するオプションである．

「システムが追加した入力語」チェックボックスは，システムが自動的に追加した語かどうかを完全照合語リスト中の語に提示するかどうかを設定するオプションである．部分照合語の中で参照オントロジー中の概念と照合した語を，ユーザが入力語として追加していなかった場合に，システムはその語を自動的に完全照合語として完全照合語リストに追加する．例えば,「資格取得日」をユーザが入力語として選択した場合,「資格取得日」自体は参照オントロジー中の概念の見出しに存在しないため，部分照合語となる.「資格取得日」の「日」に対して部分照合したとする．ここで，ユーザが「日」を入力語として選択している場合には問題ない．しかし,「日」をユーザが入力語として選択していなかった場合には,「日」が自動的に完全照合語リストに追加される．システムが自動的に追加した語には,「(自動追加)」と表示される．

「入力概念選択結果を対応する部分照合語リストに適用」チェックボックスは，完全照合語の入力概念選択結果を，その完全照合語に照合した部分照合語リストの入力概念選択に反映させるかどうかを設定するためのオプションである．例えば，完全照合語「日」に対して入力概念選択を行った結果を，部分照合語リスト中の「資格取得日」や「研究日」などにも反映させるかどうかを設定することができる．

(5) 部分照合語リストに関する設定を行うことができる．

「意味数」チェックボックスは (4) の完全照合語リストのオプションにおける「意味数」と同様である．

「形態素リスト」チェックボックスは，部分照合語を形態素解析器で形態素に分割したときの分割のされ方を表示するか否かを設定するためのオプションである．このオプションを有効にした場合，例えば,「資格取得日」に対して,「(資格+取得+日)」が表示される.「+」記号は形態素の区切りをあらわす.

「照合結果」チェックボックスは，部分照合語の形態素リストの中で，参照オントロジー中の概念と照合した形態素リストを表示するか否かを設定するオプションである．このオプションを有効にした場合，例えば,「資格取得日」は,「日」で照合しているため,「(日)」と表示される．

「選択中の完全照合語に対応する複合語のみ表示」チェックボックスは，完全照合語リストで選択した語を照合語とする部分照合語のみを表示するか否かを設定するためのオプションである．このオプションを有効にした場合，例えば，完全照合語リスト中の「日」を選択した場合,「資格取得日」や「研究日」など「日」と照合した部分照合語のみが部分照合語リストに表示される．

(6) 入力語の追加および削除を行うことができる．

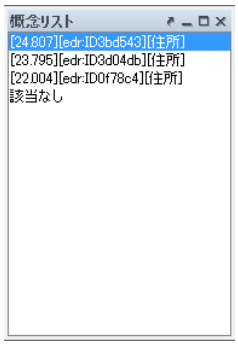


図 4.28: 入力概念選択パネル：概念リスト

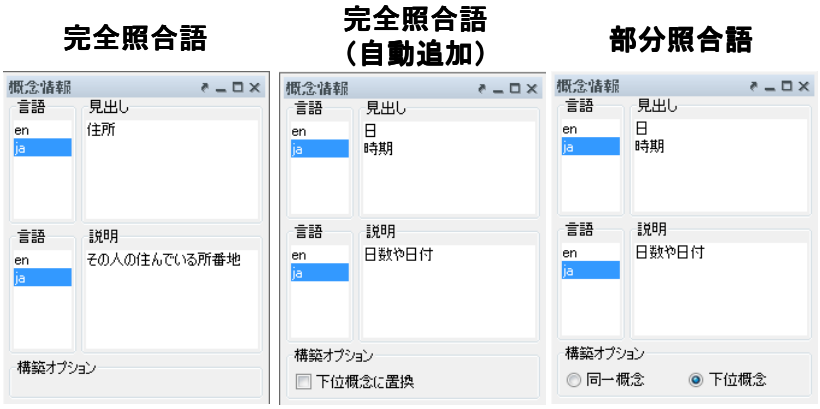


図 4.29: 入力概念選択パネル：概念情報

概念リスト

図 4.28 は図 4.26(2)「概念リスト」を拡大した図である．概念リストは，図 4.27(2) または (3) で選択した完全照合語または部分照合語を見出しとして持つ参照オントロジー中の概念のリストを表示する．図 4.28 は,「住所」を見出しとして持つ参照オントロジー（この例では EDR を参照オントロジーとしている）中の概念リストを示している．リストの項目は，三つの部分から構成されている．左側は，4.5.3 項で述べた，自動概念選択方法により求めた，入力語に対応する概念候補の評価値を示す．入力語に対応する概念候補は，評価値の降順に並び替えて表示される．評価値が高い概念ほど，より入力概念となる可能性が高い概念となる．中央は概念の ID をあらわす．概念の ID は URI で表され，画面上には名前空間接頭辞とローカル名の組み合わせが表示される．「edr:」は EDR の名前空間接頭辞を示しており，ここで表示される接頭辞は，図 4.20(2) で示した名前空間テーブルで設定した名前空間接頭辞となる．右側には，概念の見出しが複数ある場合，そのうちのいずれか一つが表示される．

概念情報

図 4.29 は図 4.26(3)「概念情報」を拡大した図である.「概念情報」には，図 4.28 の「概念リスト」で選択された概念の見出しと説明が表示される.「言語」リストで選択した言語の見出しおよび説明が「見出し」リストおよび「説明」リストに表示される．図 4.29 下部の「構築オプション」では，概念階層の構築方法を設定することができる.「構築オプション」には，図 4.27「用語リスト」で選択する用語の種類に応じて 3 種類の表示方法がある．図 4.27(2) で完全照合語を選択した場合，図 4.29 左側のように「構築オプション」には何も表示されない．図 4.27(2) でシステムが自動的に追加した完全照合語[12]（「自動追加」が表示される完全照合語）を選択した場合には，図 4.29 中央のように「構築オプション」には「下位概念に置換」するかどうかを選択するチェックボックスが表示される．図 4.27(3) で部分照合語を選択した場合には図 4.29 右側のように「構築オプション」には,「同一概念」か「下位概念」かの選択をするためのラジオボタンが表示される．

図 4.29 中央の「構築オプション」の例として,「資格取得日」のみを入力語とした場合を考える．この場合,「資格取得日」は部分照合語となり,「日」と照合するため,「日」はシステムにより自動的に完全照合語リストに追加される.「日」の入力概念選択を行う際に,図 4.29 中央の「構築オプション」として「下位概念に置換」というチェックボックスが表示される．ここでは,「日」はシステムが自動的に追加した語であるため，ユーザがあえて「日」を入力語としなかったのか，入力語にし忘れたかの確認をしている．ユーザがあえて「日」を入力語にしなかった場合，概念階層中に「日」は含まれるべきではない.「構築オプション」の「下位概念に置換」をチェックすることにより,「資格取得日」は「日」の下位概念とはならず，概念階層中に表示されない．ユーザが「日」を入力語に追加し忘れた場合には,「構築オプション」の「下位概念に置換」にチェックをいれなければ,「資格取得日」は「日」の下位概念として概念階層が構築される．

図 4.29 右側の「構築オプション」の例として,「日」と「資格取得日」を入力語とした場合を考える．上記と同様に「資格取得日」は「日」で照合する部分照合語である.「資格取得日」の入力概念選択を行う際に，図 4.29 右側の「構築オプション」が表示される.「同一概念」のほうを選択した場合は，概念階層構築時に「資格取得日」は「日」と同一概念として扱われる．つまり,「資格取得日」は「日」概念に対応する参照オントロジー中の概念の別見出しとして概念階層が構築される．一方,「下位概念」のほうを選択した場合は,「資格取得日」は「日」とは異なる概念，ここでは,「日」の下位概念として概念階層が構築される．初期状態において，部分照合語を「同一概念」とみなすか,「下位概念」とみなすかは，オプションダイアログにより設定することができる．

階層構築オプション

図 4.30 は図 4.26(7)「階層構築オプション」を拡大した図である.「階層構築オプション」では，4.10.6 項および 4.10.7 項で述べる，クラスおよびプロパティ階層構築モジュールに

[12]部分照合語の照合部分の語をユーザが入力語としていない場合には，システムは自動的にその語を入力語として追加する．これを完全照合語（自動追加）と呼ぶ.

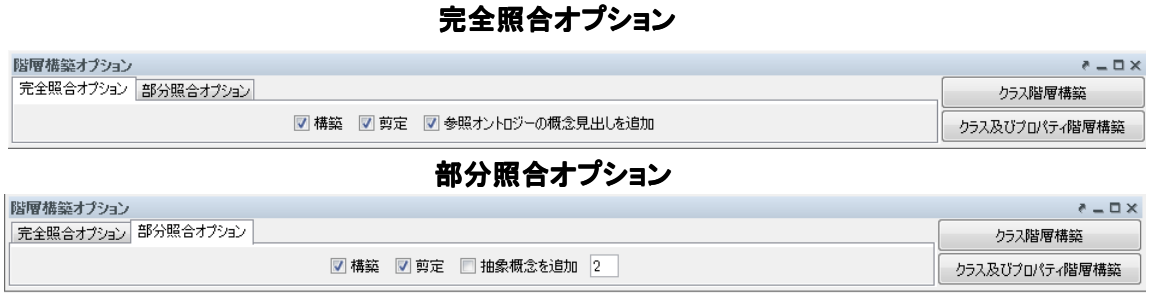


図 4.30: 入力概念選択パネル：階層構築オプション

おいて，クラスおよびプロパティ階層を構築する際のパラメータの設定を行う.「階層構築オプション」は,「完全照合オプション」および「部分照合オプション」から構成される.

図 4.30 の「完全照合オプション」では，完全照合語リストから概念階層を構築する際の設定を行う.「構築」チェックボックスでは，完全照合語リストから概念階層を構築するかどうかを選択する.「剪定」チェックボックスでは，概念階層構築時に剪定を行うかどうかを選択する.「参照オントロジーの概念見出しを追加」チェックボックスでは，概念階層構築時に，各概念の見出しとして，入力語として与えた語のみを概念の見出しとするか,対応する参照オントロジー中の概念の見出しをすべて利用するかどうかを選択する.

図 4.30 の「部分照合オプション」では，部分照合語リストから概念階層を構築する際の設定を行う.「構築」チェックボックスでは，部分照合語リストから概念階層を構築するかどうかを選択する.「剪定」チェックボックスでは，概念階層構築時に剪定を行うかどうかを選択する.「抽象概念を追加」チェックボックスでは，部分照合語リストから概念階層を構築する際に，語頭による階層化を行うかどうかを選択する．このチェックボックス右側のテキストフィールドには，いくつ以上グループ化できる場合に共通の上位概念を挿入するかを設定する.

図 4.30 右端にある「クラス階層構築」ボタンを押すと，上記の階層構築オプションに基づいて，クラス階層構築パネルにクラス階層のみが構築される.「クラスおよびプロパティ階層構築」ボタンを押すと，上記の階層構築オプションに基づいて，クラス階層構築パネルおよびプロパティ階層構築パネルに，クラス階層およびプロパティ階層が構築される.クラス階層とプロパティ階層の両方を構築するためには，参照オントロジーとして EDR 一般辞書またはプロパティ階層を含む OWL オントロジーを設定しなければならない.

4.10.6 クラス階層構築・洗練モジュールの実装

階層構築モジュールと階層洗練モジュールの設計を基に，クラス階層構築パネルを実装した．図 4.31 にクラス階層構築パネルを示す．以下に各部分の説明を示す.

(1) 未定義語リスト

参照オントロジー中の概念に照合しなかった入力語リスト．リストから語を選択し,

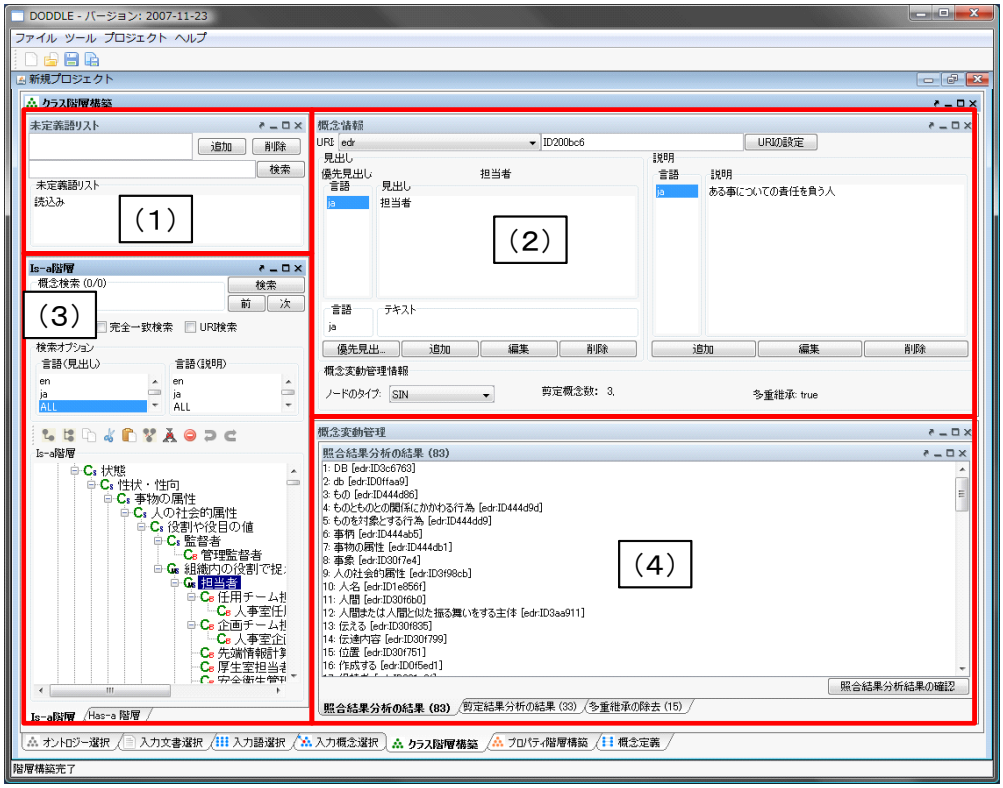


図 4.31: クラス階層構築パネル

「Is-a 階層パネル」にドラッグ＆ドロップすると，未定義語を概念として Is-a 階層に追加できる．

(2) 概念情報パネル

概念階層中の選択された概念の URI，優先見出し（階層中に表示する見出し），見出し，説明，概念変動管理情報を表示する．見出しと説明については，言語属性の付与と追加，編集，削除ができる．

(3) 概念階層パネル

Is-a 階層と Has-a 階層．概念の検索，追加，削除などを行うことができる．

(4) 概念変動管理パネル

照合結果分析結果，剪定結果分析結果，多重継承している概念をリストで表示し，各項目を選択すると Is-a 階層中の修正候補箇所が選択される．

以下では，図 4.31(2) から (4) の詳細を説明する．

概念情報パネル

図 4.32 は図 4.31(2) 概念情報パネルを拡大した図である．以下では，概念情報パネルの各部分について説明する．

図 4.32: クラス階層構築パネル: 概念情報パネル

(1) 名前空間接頭辞をコンボボックスから選択し，ローカル名をテキストフィールドに入力し,「URI の設定」ボタンを押すことで，選択した概念の URI を変更することができる．図 4.20(2) で示した名前空間テーブルに定義された接頭辞が選択可能である．

(2) 概念の見出しを編集するための領域である.「言語」リストの項目を選択することで，選択した言語の見出しが「見出し」リストに表示される．図 4.32(2) では，日本語見出しとして「担当者」が表示されている．図 4.32(2) 下部の「言語」と「テキスト」テキストフィールドに追加したい見出しの言語とテキストを入力し,「追加」ボタンを押すことで概念の見出しを追加することができる．また，選択した見出しを編集したい場合には「編集」ボタンを，削除したい場合には「削除」ボタンを押すことにより，見出しの編集および削除を行うことができる．また,「優先見出しの設定」ボタンを押すことで，選択された見出しが Is-a 階層および Has-a 階層パネルの概念の表示用の見出しとなる．

(3) 概念の説明を編集するための領域である．見出しと同様に「言語」リストの項目を選択することで，選択した言語の説明が「説明」リストに表示される．

(4) 概念変動管理情報を表示・編集するための領域である.「ノードのタイプ」は，編集対象のノードが SIN（参照オントロジーから抽出した概念）かベストマッチノード（入力概念）かを表示する．SIN の中でベストマッチノードとしたいノードについては，ここでノードのタイプを SIN からベストマッチに変更することができる.「剪定概念数」は，階層構築時の剪定により，選択された概念とその上位概念の間の概念がいくつ削除されたかを表示している.「多重継承」は，編集対象のノードが多重継承をしているかしていないかを表している．多重継承をしている場合は「true」，していない場合は「false」と表示される．

(5) (3) の「追加」または「編集」ボタンを押すと表示される.「言語」と「説明」を入力し,「OK」ボタンを押すと，概念の説明の追加や編集を行うことができる．また，

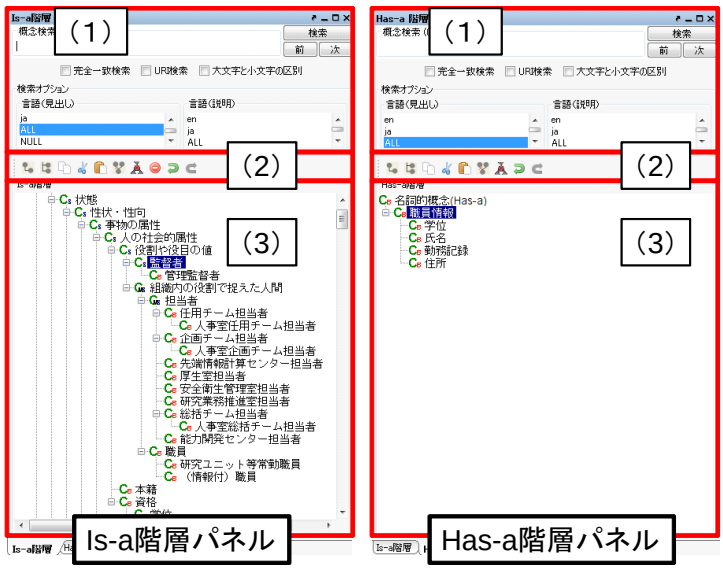


図 4.33: クラス階層構築パネル: Is-a 階層パネルと Has-a 階層パネル

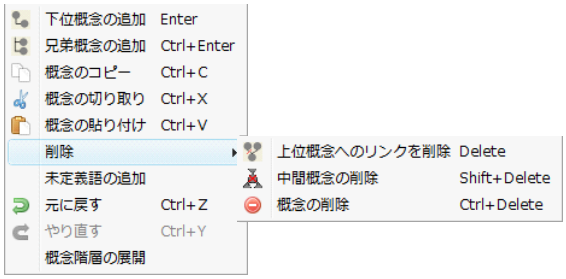


図 4.34: クラス階層構築パネル: ポップアップメニュー

「削除」ボタンにより選択された概念の説明を削除することができる.

Is-a 階層および Has-a 階層パネル

図 4.33 は図 4.31(3) を拡大した図である. 図 4.33 の左側が Is-a 階層パネルを右側が Has-a 階層パネルを示している. 以下では, Is-a 階層パネルと Has-a 階層パネルの各部分について説明する.

(1) 概念階層中の概念を検索するための領域である. テキストフィールドに検索キーワードを入力し,「検索」ボタンを押すと検索オプションを満たす概念が選択される. 候補が複数ある場合には,「次」ボタンまたは「前」ボタンで別の概念候補に移動できる. 検索オプションとしては, 言語, 概念の見出し, 概念の説明が選択できる. また,「完全一致検索」チェックボックスにチェックをいれると, 入力した検索キーワードと完全に一致する見出しや説明を含む概念のみが検索される.「完全一致検索」チェックボックスにチェックが入っていない場合は部分一致検索となり, 検索キーワードを

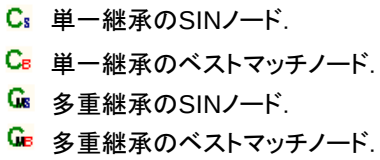


図 4.35: クラス階層構築パネル: ノードのアイコン

見出しまたは説明の一部に含む概念が検索される.「URI 検索」チェックボックスにチェックをいれると，概念の URI も検索対象となる.「大文字と小文字の区別」チェックボックスにチェックをいれると，英語見出しまたは説明を検索する際に，大文字と小文字を区別して検索する.

(2) Is-a 階層および Has-a 階層の編集に利用可能なツールバー．ツールバーは，階層中の概念をマウスで右クリックした際に表示される，図 4.34 のポップアップメニューと同様の機能を持つ.

(3) Is-a 階層と Has-a 階層を表示・編集するためのパネル．(2) のツールバーまたは概念を選択して，マウスを右クリックすることで表示されるポップアップメニューから，概念の追加，削除などを行うことができる.

図 4.34 は Is-a 階層パネルのポップアップメニューを示している．Is-a 階層パネルと Has-a 階層パネルの主な違いとして，Has-a 階層パネルでは Is-a 階層パネルで定義された概念を用いて Has-a 関係を定義する点が異なる．また，Has-a 階層では，以下で説明する「概念の削除」を行うことはできない.

DODDLE-OWL における概念の削除は 3 種類ある.「概念の削除」は削除対象のノードと同一 URI を持つノードおよびその下位ノードをすべて削除する.「上位概念へのリンクを削除」は，多重継承している場合に削除対象のノードとその上位ノードの間の関係を削除する.「中間概念の削除」は，削除対象のノードを削除し，その下位ノードを削除対象のノードの上位ノードの下位ノードとして定義する.

クラス階層構築パネルにおける Is-a 階層パネルと Has-a 階層パネルのクラスには，図 4.35 に示す 4 種類がある.

概念変動管理パネル

図 4.36 は図 4.31(4) 概念変動管理パネルの各タブを展開し，拡大した図である．以下では，概念変動管理パネルの各部分について説明する.

(1) 照合結果分析の結果をリストで表示する．リストの項目は SIN ノードであり，項目を選択すると Is-a 階層中の該当する部分木が選択される．また，照合結果分析結果を確認し修正する必要がない場合，もしくは，修正後に「照合結果分析結果の確認」ボタンを押すことで，選択した項目をリストから削除することができる.

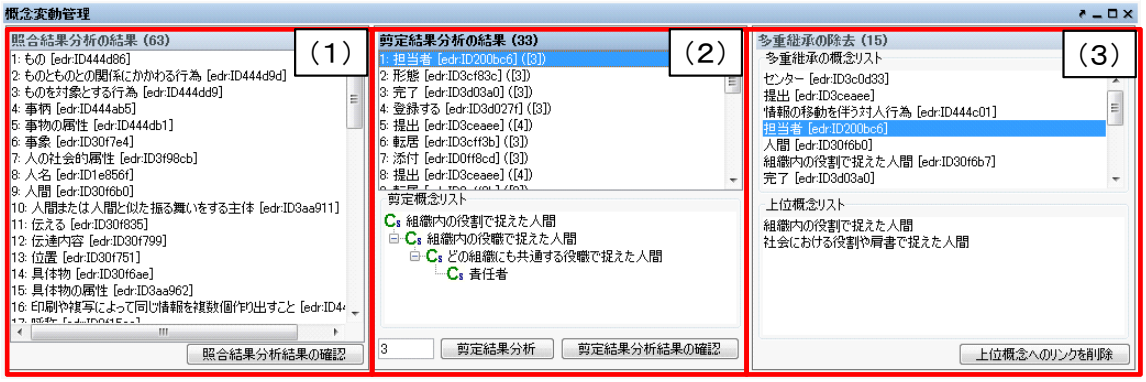


図 4.36: クラス階層構築パネル: 概念変動管理パネル

(2) 剪定結果分析の結果をリストで表示する．(2) 下部の「剪定概念リスト」は，概念階層構築時に剪定された，選択した概念とその上位概念の間の概念が提示されている.「剪定結果分析」ボタンを押すと，ボタン左側のテキストフィールドに指定した数よりも多くの中間概念が削除された概念をリストに表示する．また，剪定結果分析結果を確認し，修正する必要がない場合，もしくは，修正後に「剪定結果分析結果の確認」ボタンを押すことで，選択した項目をリストから削除することができる．（当該概念の剪定概念数がゼロとなる）

(3) 多重継承している概念のリストを表示する．リストの項目を選択すると，(3) 下部に多重継承しているノードのリストが表示される．このノードを選択すると，Is-a 階層パネル中の概念に移動し，ノードを選択する.「上位概念へのリンクを削除」ボタンを押すと，選択した概念と上位概念の間の関係が削除される．

4.10.7 プロパティ階層構築・洗練モジュールの実装

階層構築モジュールおよび階層洗練モジュールの設計を基に，プロパティ階層構築パネルを実装した．図 4.37 にプロパティ階層構築パネルを示す．プロパティ階層構築パネルの構成要素の大部分は，クラス階層構築パネルと同様である．異なる点は，図 4.37(1) の概念定義パネルがある点である．概念定義パネルは，汎用オントロジーとして EDR 一般辞書を指定し，プロパティ階層を構築した場合，EDR 概念記述辞書における，`agent` および `object` の関係にある概念を定義域および値域として自動的に定義している．また，クラス階層を参照し，定義域および値域の追加を行うことも可能である．

プロパティ階層構築パネルにおける Is-a 階層パネルと Has-a 階層パネルのプロパティには，図 4.38 に示す 4 種類がある．

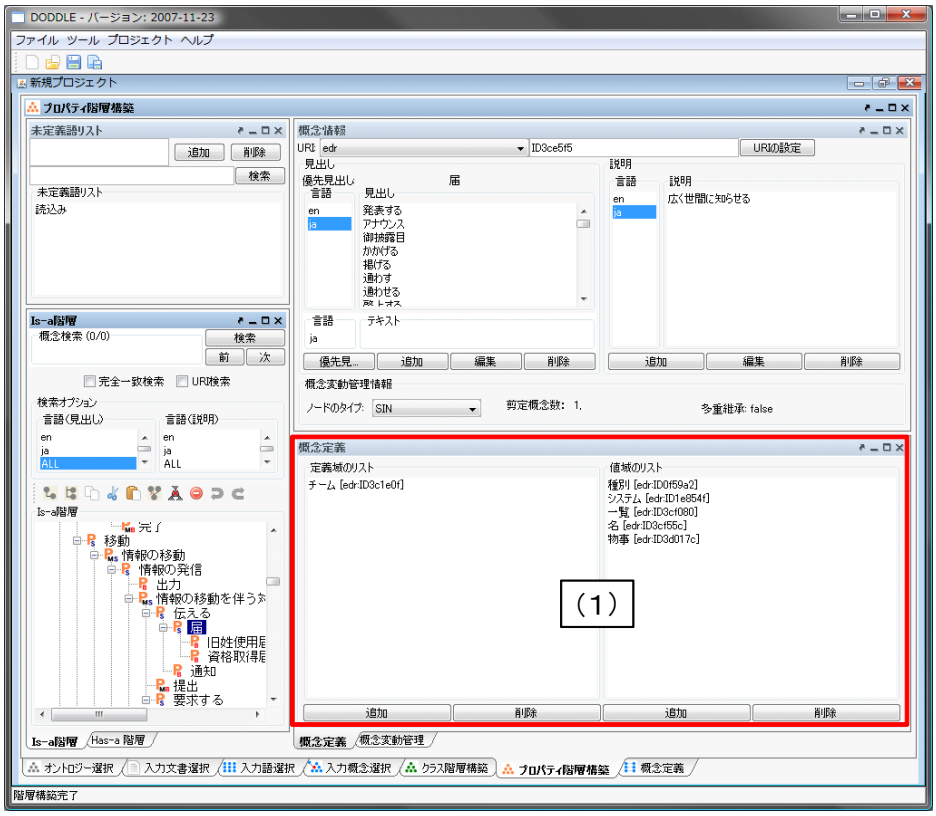


図 4.37: プロパティ階層構築パネル

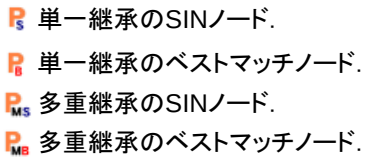


図 4.38: プロパティ階層構築パネル: ノードのアイコン

4.10.8 関係構築・洗練モジュールの実装

関係構築モジュールおよび関係洗練モジュールの設計を基に，概念定義パネルを実装した．図 4.39 に，概念定義パネルのスクリーンショットを示す．以下では，概念定義パネルの各部分について説明する．

(1) WordSpace パラメータの設定を行う．WordSpace のパラメータとしては，N-gram，N-gram 出現頻度，文脈スコープ（前，後 N 語），文脈類似度の閾値を設定できる．「WordSpace の実行」ボタンを押すと結果が (5) に表示される．

(2) Apriori パラメータの設定を行う．Apriori のパラメータとしては，最小支持度および最小確信度を設定できる．「Apriori」の実行ボタンを押すと結果が (5) に表示される．

(3) 入力語選択パネルで選択した入力語が表示される．

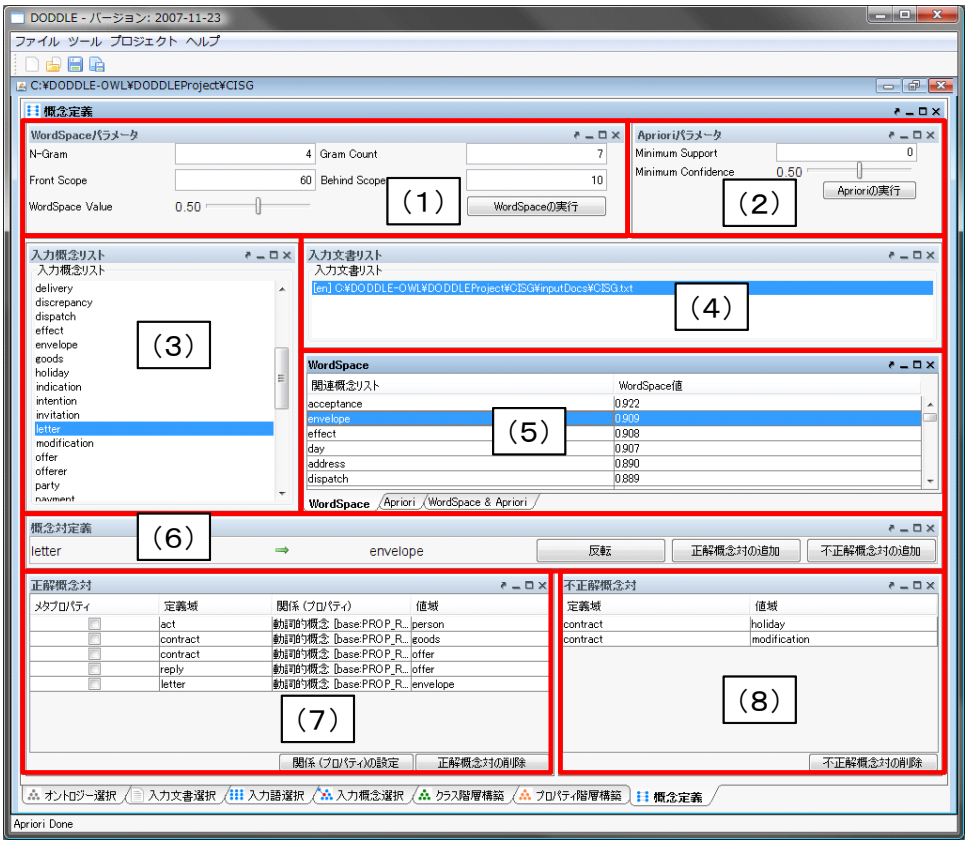


図 4.39: 概念定義パネル

(4) 入力文書選択パネルで選択した入力文書が表示される．

(5) (3) で選択した入力語と関連のある入力語を関係値と共に表示する．関係値の高い順に表示される．WordSpace，Apriori，WordSpace および Apriori のアルゴリズムの関係値をタブで切り替えて表示することができる．

(6) (3) で選択した入力語と関連のある (4) で選択された語を表示し，正解概念対または不正解概念対として (7) または (8) に追加する．矢印の向きによって，定義域と値域が変化する．

(7) 定義域，プロパティ，値域が表示される．プロパティは，プロパティ階層構築パネルから選択することができる．

(8) 不要な概念対が表示される．不要な概念対は，概念定義の候補となる概念対集合から削除されるため，残りの概念定義を行いやすくなっている．

4.11 関連研究との比較

本節では，オントロジーの（半）自動構築（オントロジー学習）における，学習機能，参照知識，入力，自動化に関する関連研究との比較について述べる．

表 4.6: 学習機能における関連研究との比較

組織	システム	学習機能							
		用語	同義語	概念	概念階層	関係	関係階層	公理スキーマ	汎用公理
AIFB, Univ. Karlsruhe	Text2Onto	✓	clusters	int.	✓	✓			
	AEON							✓	
Amir Kabir Univ. Tehran	HASTI	✓			✓	✓			✓
CNTS,Univ. Antwerpen	OntoBasis		clusters	clusters					
DFKI	OntoLT/RelExt	✓			✓	✓			
Economic Univ. Prague	TextToOnto++					labels			
ISI, USC	CBC		clusters	clusters					
	DIRT								✓
Keio Univ.	DODDLE-OWL	✓			✓	✓			
NRC-CNRC	PMI-IR		✓						
Univ. de Paris-Sud	ASIUM/Mo'k		clusters	clusters	✓	✓			
Univ. di Roma	OntoLearn	✓	✓	int.	✓	✓			
Univ. of Salford	ATRACT	✓	clusters	clusters					
Univ. Zürich	Parmenides	✓			✓				

4.11.1 学習機能における関連研究との比較

表 4.6 は，EACL2006 (11th Conference of the European Chapter of the Association for Computational Linguistics) のチュートリアル [13]において紹介されたオントロジーの（半）自動構築に関する研究の一覧を示している．表 4.6 では，各研究が，オントロジーにおけるどのような要素の（半）自動構築を支援しているかをまとめている．オントロジー学習における学習機能には，用語，同義語，概念，概念階層，関係，関係階層，公理スキーマ，汎用公理がある．

用語は，オントロジー学習の基礎となる学習機能で，品詞のパターンによる名詞句の抽出や TF-IDF などの統計的手法による用語の重み付けなどが行われる．Text2Onto [59]，HASTI [60]，OntoLT/RelExt [61]，OntoLearn [31] など，様々なツールが用語の学習を支援している．DODDLE-OWL では，複合語抽出モジュールや TF-IDF を用いて用語の獲得を支援している．

同義語の学習では，同一概念を表す用語を同定する．言語が異なるが同じ意味を表す用語についても同定する．同義語の学習には，WordNet の Synset を用いた分類や用語間の共起性を用いたクラスタリングの手法が用いられる．Text2Onto，OntoBasis [62]，ASIUM [16]，OntoLearn などが同義語の学習を支援している．DODDLE-OWL では，同義語の学習は支援していない．しかしながら，参照オントロジーを用いて，入力概念選択をユーザが手動で行うことで，同義語の同定を行っている．

概念の学習では，概念の定義を WordNet の説明 (gloss) などを用いて自動的に獲得する．また，テキストなどから概念のインスタンスの獲得も支援する．同義語の学習を支援しているツールは，概念の学習も支援している．同義語の同定と同様，DODDLE-OWL では参照オントロジーを用いて，入力概念選択をユーザが手動で行うことで，概念の定義

[13]http://www.aifb.uni-karlsruhe.de/WBS/pci/EACL_OL_Tutorial_06/

を獲得している．

概念階層の学習では，構文解析を用いた階層関係の獲得や，WordNet などの汎用オントロジーを基にした，階層関係の拡張を支援する．Text2Onto，HASTI，OntoLT，ASIUM，OntoLearn などが概念階層の自動構築を支援している．DODDLE-OWL では，部分照合概念の階層構築時に，参照オントロジーを基に階層関係の拡張を行っている．完全照合概念の階層構築については，参照オントロジーから階層関係を抽出，合成，剪定を行っているのみで，新たな階層関係の定義は行っていない．しかしながら，階層関係構築時の照合や剪定の情報を用いて，階層関係の洗練支援を行っている点は，関連研究と比較して独自性があるといえる．

関係の学習では，自然言語処理やデータマイニングの技術を用いて，その他の関係の自動構築を支援する．概念階層構築支援と同様，Text2Onto，HASTI，OntoLT，ASIUM，OntoLearn などが関係の自動構築支援を行っている．DODDLE-OWL では，共起性に基づく手法を用いて概念対の自動獲得を行っている．しかしながら，概念間関係については自動的に獲得は行っておらず，ユーザが手動で定義している．現状では，概念間関係の獲得は自然言語処理などの技術を用いても高い精度で行うことができないため，DODDLE-OWL ではユーザに定義をゆだねている．

公理の学習としては，自然言語処理を用いて語の依存関係などを解析し，推論ルールの発見や排他的関係の獲得などを支援する．AEON [63]，HASTI，DIRT [64] が公理の学習を支援している．DODDLE-OWL では，公理の学習は支援していない．

4.11.2 参照知識，入力，自動化の度合いにおける関連研究との比較

表 4.7 は，Amir Kabir 工科大学の Shamsfard らによるオントロジー学習に関する比較研究論文 [65] で紹介されている，オントロジー学習ツールの一覧を示している．ここでは，オントロジー学習における，学習機能，参照知識，入力，自動化の度合いに着目してまとめている．

学習機能については，4.11.1 項でも述べたように，DODDLE-OWL は関連研究と比較して，独自の要素の自動構築を支援していない．

参照知識については，OWL 形式の既存オントロジーが利用できる点が関連研究と比較して優れた点である．

入力については，非構造化 文書が大部分の関連研究で利用されている．しかしながら，各研究ごとに，支援している入力文書の記述言語が異なっている．これは，自然言語処理技術を用いてオントロジーの自動構築を支援しているため，記述言語の特性（構文など）に依存するためである．DODDLE-OWL では，英語と日本語で記述された文書を入力とすることができる．日本語に関するオントロジー構築支援が可能な点は，関連研究と比べて独自な点である．

自動化の度合いについては，大部分のツールがオントロジーの自動または半自動構築（パラメータの設定程度）を支援しているのに対して，DODDLE-OWL はユーザとのインタラクションによりオントロジーの構築を支援している．（半）自動構築したオントロ

表 4.7: 参照知識，入力，自動化の度合いにおける関連研究との比較

組織	システム	学習機能	参照知識	入力	自動化の度合い
Keio University	DODDLE-OWL	用語，階層関係，その他の関係	WordNet，EDR，既存オントロジー (OWL 形式)	非構造化 領域専門文書（日本語および英語）	ユーザインタラクション，半自動オントロジー洗練
Paris-Sud University	ASIUM	動詞の下位範疇化フレーム，階層関係	Linguistic K.	非構造化 文書（フランス語）	半自動
Amir Kabir University of Technology	HASTI	用語，概念，階層関係，その他の関係，公理	ほぼなし（小さなカーネル）	非構造化 文書（ペルシャ語）	半自動
CNRS laboratory	SVETLAN’	名詞クラス	なし	SEGAPSITH の入力となる（非）構造化文書（フランス語）	自動
University of Albert-Ludwigs	SYNDIKATE	用語，概念，階層関係，その他の関係	汎用オントロジー，領域語彙	非構造化 文書（ドイツ語）	自動
University of Karlsruhe	TEXT-TO-ONTO	概念，階層関係，その他の関係	語彙 DB，領域語彙	非構造化 文書（ドイツ語），非構造化文書 (HTML, XML)	半自動
Distributed Systems Technology Centre	WEB → KB	クラスおよび関係のインスタンス，インスタンス認識ルール	学習されるインスタンスのためのオントロジー	オントロジー，インスタンスの訓練データ (HTML)	自動

ジーの質は手動構築と比べて高くないため，ユーザがオントロジーを洗練する必要がある．しかしながら，自動構築されたオントロジーを修正することは，ユーザの負担となる．DODDLE-OWL では，オントロジー構築における各工程（入力文書選択，入力語選択，入力概念選択，階層構築および洗練，関係構築および洗練）においてユーザとのインタラクションをとることによって，ユーザがオントロジーを洗練する負担を軽減している．

4.12 DODDLE-OWL と手動構築との比較評価

4.12.1 実験方法

DODDLE-OWL の有用性を評価するために，DODDLE-OWL と手動によるオントロジー構築の比較を行った．手動構築とは，オントロジーを半自動構築可能なツールではなく，オントロジーエディタを用いたオントロジー構築のことを指す．本実験では，現在最も普及しているオントロジーエディタである Protégé [13] を用いた．本実験における手動構築では，オントロジーを構築する際に，被験者が汎用オントロジーや辞書などを参照することを許している．また，本実験におけるオントロジー構築とは，is-a 関係の構築を意味する．

実験は，2 人の被験者（被験者 A および被験者 B）が行った．2 人の被験者は共に，理工系の大学院生で，4ヶ月程度の RDF，RDFS，OWL に関するゼミに 3 回参加している．また，DODDLE-OWL および Protégé を用いて，数百概念規模のオントロジー構築を数回行った経験がある．

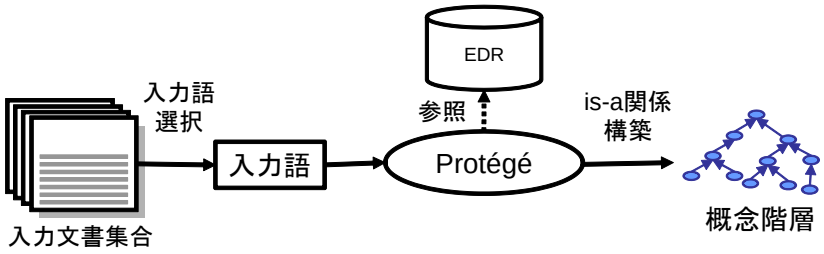


図 4.40: 実験における手動によるオントロジー構築手順

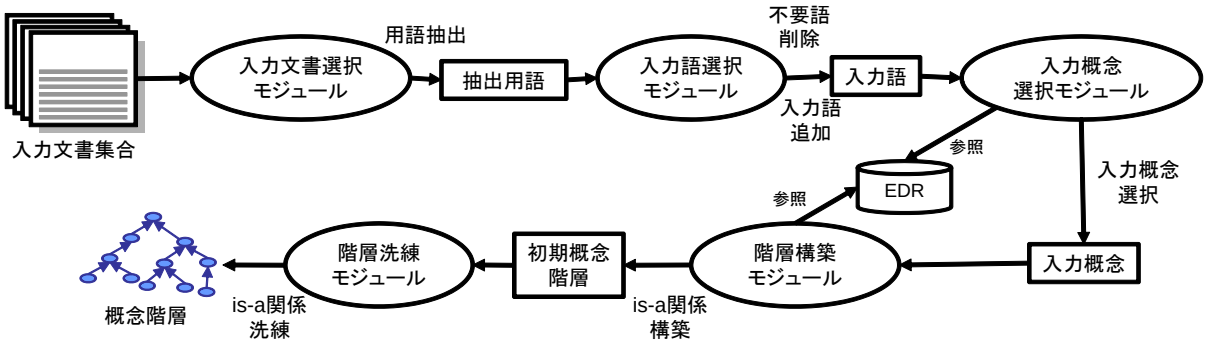


図 4.41: 実験における DODDLE-OWL を用いたオントロジー構築手順

対象ドメインは，財務会計および人事とした．被験者は対象ドメインについての専門家ではないが，対象ドメインの文書を読んで内容を理解するだけの知識は持っている．対象ドメインの文書は被験者らの母国語である日本語で記述されている．このことから，階層構築時に DODDLE-OWL が参照する参照オントロジーとして，本実験では日本語汎用オントロジーである EDR 電子化辞書を用いた．

図 4.40 に，本実験における手動によるオントロジー構築手順を示す．手動によるオントロジー構築では，はじめに被験者は入力文書集合から入力語を手動で抽出する．次に，EDR を参照しながら，入力語から Protégé を用いて is-a 関係を構築する．

図 4.41 に，本実験における DODDLE-OWL を用いたオントロジー構築手順を示す．DODDLE-OWL を用いたオントロジー構築は以下の五つの手順で行われる．第 1 に，被験者は入力文書集合から入力文書選択モジュールを用いて，用語を自動抽出する．第 2 に，抽出された用語から入力語選択モジュールを用いて，被験者は不要語の削除および自動抽出できなかった入力語の追加を行い，入力語を決定する．第 3 に，EDR を参照しながら，入力語から入力概念選択モジュールを用いて入力概念の選択を行う．第 4 に，入力概念から階層構築モジュールを用いて初期概念階層を自動構築する．第 5 に，初期概念階層を階層洗練モジュールを用いて洗練し，is-a 関係を構築する．

DODDLE-OWL と手動によるオントロジー構築の比較を行う方法として，主に以下の二つの方法が考えられる．比較方法 1 は，各被験者が各ドメインについて DODDLE-OWL および手動によりオントロジーを構築し，構築時間を比較する方法である．比較方法 1 を

表 4.8: DODDLE-OWL と手動によるオントロジー構築の比較方法 1

	DODDLE-OWL	**手動構築**
被験者 A	財務会計	財務会計
	人事	人事
被験者 B	財務会計	財務会計
	人事	人事

表 4.9: DODDLE-OWL と手動によるオントロジー構築の比較方法 2

	DODDLE-OWL	**手動構築**
被験者 A	財務会計	人事
被験者 B	人事	財務会計

表 4.8 に示す．比較方法 2 は，各ドメインについて，一方の被験者が DODDLE-OWL を用いた場合，もう一方の被験者が手動によりオントロジーを構築し，それぞれの構築時間を比較する方法である．比較方法 2 を表 4.9 に示す．

比較方法 1 では，同一被験者が同一ドメインの同一文書集合から，DODDLE-OWL および手動によりオントロジーを構築する．オントロジー構築における習熟度の影響が小さい場合には正確な評価を行うことができる．しかし，オントロジー構築における習熟度の影響が大きい場合には，1 回目のオントロジー構築経験が 2 回目のオントロジー構築時間の短縮につながる．そのため，DODDLE-OWL と手動を用いてオントロジーを構築する順番により，それぞれのオントロジー構築時間が変化するという問題が生じる．よって，比較方法 1 を用いる場合には，あらかじめドメインごとのオントロジー構築における各被験者の習熟度の影響を知る必要がある．

比較方法 2 では，各被験者は各ドメインの文書集合から 1 回だけ，DODDLE-OWL または手動によりオントロジーを構築するため，比較方法 1 のようなオントロジー構築における習熟度の影響はない．しかし，被験者 A と被験者 B の知識レベルの差（ドメインに関する知識，オントロジー構築経験，ツールの習熟度などによる，オントロジー構築時間の差）が同程度でなければ，DODDLE-OWL と手動によるオントロジー構築時間の差が，知識レベルの差によるものなのか，DODDLE-OWL によるものなのかを明確に知ることができない．そのため，一方の被験者の DODDLE-OWL を用いたオントロジー構築時間ともう一方の被験者の手動によるオントロジー構築時間を直接比較することは困難である．よって，比較方法 2 を用いる場合には，あらかじめドメインごとのオントロジー構築における被験者 A と被験者 B の知識レベルの差を知る必要がある．被験者 A と被験者 B の知識レベルの差がわかれば，手動構築した被験者のオントロジー構築時間から，DODDLE-OWL を用いてオントロジーを構築した被験者の予想手動構築時間をおおよそではあるが求めることができる．同一ドメインかつ同一規模の文書集合からオントロジーを構築する場合でも，対象文書集合の内容によりオントロジー構築時間は変化することがあるため，予想手動構築時間は正確にはかることはできない．しかしながら，目安に

表 4.10: 予備実験および本実験に用いた財務会計および人事ドメインの各文書集合における文書数および語数

	ドメイン	文書数	語数
予備実験	財務会計（調達）	19	20041
	人事（採用）	25	20035
本実験	財務会計（支払）	19	20388
	人事（給与支給）	29	20286

はなると考えられる．DODDLE-OWL を用いたオントロジー構築時間が，予想手動構築時間と比べて大きく短縮されていれば，DODDLE-OWL の有用性を示すことができると考えられる．

本評価実験では，上記の問題を解決するために予備実験を行い，それを基に本実験を行った．予備実験では，財務会計および人事ドメインについて，本実験とは内容が異なるが規模は同程度の文書集合を用意し，被験者 A と被験者 B が手動により，それぞれ 2 回ずつオントロジーを構築した．予備実験では，被験者 A と B 共に，財務会計（調達）ドメイン，人事（採用）ドメインの順に 1 回目の手動によるオントロジー構築を行った 2〜3 日後に，財務会計（調達）ドメイン，人事（採用）ドメインの順で 2 回目の手動によるオントロジー構築を行った．

表 4.10 に，予備実験および本実験で用いた財務会計および人事ドメインの各文書集合における文書数および語数を示す．予備実験を行う理由は二つある．一つ目は，財務会計および人事ドメインのそれぞれについて，被験者 A と被験者 B が手動でオントロジー構築する場合の知識レベルの差を調べることである．ここで，財務会計と人事ドメインのそれぞれについてオントロジー構築を行う理由は，ドメインにより被験者 A と被験者 B の知識レベルの差が異なることが考えられるためである．二つ目は，同一ドメインの同一文書集合から各被験者が 2 回ずつオントロジーを構築することにより，習熟度によるオントロジー構築時間の変化を調べることである．予備実験の結果により，比較方法 1 と比較方法 2 のどちらを本実験に用いるかを決定することにした．習熟度によるオントロジー構築時間の差が小さい場合には，比較方法 1 を用いて本実験を行う．習熟度によるオントロジー構築時間の差が大きい場合には，比較方法 1 では DODDLE-OWL と手動構築を比較することは困難であると考えられる．その場合には，予備実験により求めたドメインごとの被験者 A と被験者 B の知識レベルの差を用いて，比較方法 2 により DODDLE-OWL と手動構築の比較を行う．

4.12.2 予備実験の結果

財務会計（調達）ドメインにおける予備実験の結果を表 4.11 および表 4.12 に示す．表 4.11 は，予備実験: 財務会計（調達）ドメインにおける 1 回目と 2 回目の被験者 A および被験者 B の手動オントロジー構築時間をあらわしている．表 4.12 は，予備実験: 財

表 4.11: 予備実験: 財務会計（調達）ドメインにおける 1 回目と 2 回目の被験者 A および被験者 B の手動オントロジー構築時間

	被験者 A		被験者 B	
	1 回目	2 回目	1 回目	2 回目
入力語抽出時間 （分）	65	43	42	34
is-a 関係構築時間 （分）	116	76	104	75
合計時間 （分）	181	119	146	109

表 4.12: 予備実験: 財務会計（調達）ドメインにおいて，1 回目と 2 回目に被験者 A および被験者 B が手動構築したオントロジーの入力語数と概念数

	被験者 A		被験者 B	
	1 回目	2 回目	1 回目	2 回目
入力語数	210	208	207	173
概念数	258	256	235	232

務会計（調達）ドメインにおいて被験者 A および被験者 B が手動構築したオントロジーの入力語数と概念数をあらわしている．

人事（採用）ドメインにおける予備実験の結果を表 4.13 および表 4.14 に示す．表 4.13 は，予備実験: 人事（採用）ドメインにおける 1 回目と 2 回目の被験者 A および被験者 B の手動オントロジー構築時間をあらわしている．表 4.13 は，予備実験: 人事（採用）ドメインにおいて被験者 A および被験者 B が手動構築したオントロジーの入力語数と概念数をあらわしている．

4.12.3 予備実験の考察

表 4.11 および表 4.13 より，財務会計（調達），人事（採用）ドメインともに，被験者 A と被験者 B は 1 回目よりも 2 回目のオントロジー構築時間は約 25 分から 60 分程度短縮されていた．よって，習熟度によるオントロジー構築時間への大きな影響が見られたため，本実験では表 4.9 で示した，比較方法 2 を用いることにした．

表 4.13: 予備実験: 人事（採用）ドメインにおける 1 回目と 2 回目の被験者 A および被験者 B の手動オントロジー構築時間

	被験者 A		被験者 B	
	1 回目	2 回目	1 回目	2 回目
入力語抽出時間（分）	50	34	41	39
is-a 関係構築時間（分）	87	70	108	84
合計時間（分）	137	104	149	123

表 4.14: 予備実験: 人事（採用）ドメインにおいて，1 回目と 2 回目に被験者 A および被験者 B が手動構築したオントロジーの入力語数と概念数

	被験者 A		被験者 B	
	1 回目	2 回目	1 回目	2 回目
入力語数	191	205	198	205
概念数	228	252	245	235

表 4.15: 予備実験: 財務会計ドメインオントロジー構築における被験者 A および被験者 B の知識レベルの差

	被験者 A	被験者 B
入力語抽出	1.548	1.000
is-a 関係構築	1.115	1.000

表 4.12 および表 4.14 より，財務会計（調達）および人事（採用）ドメインそれぞれについて，被験者 A と被験者 B が手動でオントロジーを構築する場合に，オントロジーの規模（入力語数と概念数）に大きな差は見られなかった．構築されたオントロジーの規模が同程度であるため，財務会計（調達）および人事（採用）ドメインそれぞれの 1 回目の被験者 A と被験者 B のオントロジー構築時間の比により，被験者 A と被験者 B の知識レベルの差をおおよそではあるがはかることが可能であると考えられる．表 4.15 および表 4.16 に，被験者 A と被験者 B の知識レベルの差を，財務会計（調達）および人事（採用）ドメインそれぞれの 1 回目のオントロジー構築時間の比から求めた結果を示す．表 4.15 および表 4.16 では，本実験において手動構築するほうの被験者の入力語抽出および is-a 関係構築の値を 1.000 として，被験者 A と被験者 B の各領域におけるオントロジー構築時間の比を求めた．

以下では，表 4.15 および表 4.16 を参照し，表 4.9 で示した比較方法 2 を用いて行った本実験の結果および考察を示す．

4.12.4 本実験の結果

本実験では，被験者 A は財務会計（支払）ドメインを DODDLE-OWL を用いて，人事（給与支給）ドメインを手動でオントロジーの構築を行った．また，被験者 B は財務会計

表 4.16: 予備実験: 人事ドメインオントロジー構築における被験者 A および被験者 B の知識レベルの差

	被験者 A	被験者 B
入力語抽出	1.000	0.820
is-a 関係構築	1.000	1.241

表 4.17: 本実験: 財務会計（支払）ドメインにおける被験者 A と被験者 B のオントロジー構築時間

	被験者 A DODDLE-OWL	被験者 A 手動構築（予想）	被験者 B 手動構築
入力語抽出時間（分）	22	63	41
入力概念選択時間（分）	8	0	0
is-a 関係構築時間（分）	35	91	82
合計時間（分）	65	154	123

表 4.18: 本実験: 財務会計（支払）ドメインにおいて被験者 A と被験者 B が構築したオントロジーの入力語数および概念数

	被験者 A DODDLE-OWL	被験者 B 手動構築
入力語数	275	177
概念数	326	215

（支払）ドメインを手動で，人事（給与支給）ドメインを DODDLE-OWL を用いてオントロジーの構築を行った．被験者 A と B 共に，DODDLE-OWL を用いてオントロジーを構築した後に，手動によりオントロジーを構築した．

評価方法としては，はじめに，予備実験で求めた表 4.15 および表 4.16 の財務会計（調達）および人事（採用）ドメインそれぞれについての被験者 A と被験者 B の知識レベルの差を用いて，手動構築した被験者のオントロジー構築時間から DODDLE-OWL を用いた被験者の予想手動構築時間を求める．次に，予想手動構築時間と DODDLE-OWL を用いたオントロジー構築時間を比較した．

表 4.17 に，本実験: 財務会計（支払）ドメインにおける入力語抽出時間，入力概念選択時間，is-a 関係構築時間，合計時間を示す．表 4.18 に，本実験: 財務会計（支払）ドメインにおけるオントロジーの入力語数と概念数を示す．表 4.19 に，本実験: 財務会計（支払）ドメインの DODDLE-OWL を用いた入力語抽出における自動抽出語数，被験者 A が追加した入力語数，正答率，再現率を示す．

表 4.19: 本実験: 財務会計（支払）ドメインの DODDLE-OWL を用いた入力語抽出における自動抽出語数，被験者 A が追加した入力語数，正答率，再現率

自動抽出語数	537
被験者 A が追加した入力語数	18
入力語数	275
正答率	0.48
再現率	0.93

表 4.20: 本実験: 人事（給与支給）ドメインにおける被験者 A と被験者 B のオントロジー構築時間

	被験者 B DODDLE-OWL	被験者 B 手動構築（予想）	被験者 A 手動構築
入力語抽出時間（分）	40	35	43
入力概念選択時間（分）	8	0	0
is-a 関係構築時間（分）	38	109	88
合計時間（分）	86	144	131

表 4.21: 本実験: 人事（給与支給）ドメインにおいて被験者 A と被験者 B が構築したオントロジーの入力語数および概念数

	被験者 B DODDLE-OWL	被験者 A 手動構築
入力語数	300	212
概念数	393	272

表 4.20 に，本実験: 人事（給与支給）における入力語抽出時間，入力概念選択時間，is-a 関係構築時間，合計時間を示す．また，表 4.21 に，本実験: 人事（給与支給）におけるオントロジーの入力語数と概念数を示している．表 4.22 に，本実験: 人事（給与支給）ドメインの DODDLE-OWL を用いた入力語抽出における自動抽出語数，被験者 B が追加した入力語数，正答率，再現率を示す．

表 4.19 および表 4.22 における正答率および再現率は，式 4.7，式 4.8，式 4.9 を用いて求めた．式 4.7，式 4.8，式 4.9 において，SI は DODDLE-OWL が獲得した入力語数を，ST は自動抽出語数を，UI は被験者が追加した入力語数を，I は入力語数をそれぞれ表している．

表 4.22: 本実験: 人事（給与支給）ドメインの DODDLE-OWL を用いた入力語抽出における自動抽出語数，被験者 B が追加した入力語数，正答率，再現率

自動抽出語数	783
被験者 B が追加した入力語数	63
入力語数	300
正答率	0.38
再現率	0.79

$$SI = I - UI \tag{4.7}$$

$$\text{正答率} = \frac{SI}{ST} \tag{4.8}$$

$$\text{再現率} = \frac{SI}{I} \tag{4.9}$$

4.12.5 本実験の考察

表 4.17 および表 4.20 より，被験者 A と被験者 B ともに予想手動構築時間よりも DODDLE-OWL を用いたほうが，時間では約 1 時間程度早くオントロジーが構築できた．本実験からは，入力語数および概念数と DODDLE-OWL を用いることで短縮できるオントロジー構築時間の相関まではわからない．しかしながら，本実験結果より 200 から 400 概念程度の is-a 関係構築には，DODDLE-OWL を用いることでオントロジー構築コストが削減できるといえる．

以下では，オントロジー構築のどの過程で DODDLE-OWL が有効であったかをより詳細に分析する．

入力語抽出に関しては，被験者 A は DODDLE-OWL を用いたほうが抽出時間が短縮できたが，被験者 B は DODDLE-OWL を用いたほうが抽出時間が多くかかっている．表 4.10 より，財務会計（支払）と人事（給与支給）ドメインの各文書集合に含まれる語数はほぼ同様であったが，DODDLE-OWL の入力語抽出精度に差があったためである．表 4.19 および表 4.22 より，財務会計（支払）ドメインにおける DODDLE-OWL を用いた入力語抽出精度は，人事（給与支給）ドメインにおける DODDLE-OWL を用いた入力語抽出精度よりも高かった．よって，人事（給与支給）ドメインにおいて DODDLE-OWL を用いた被験者 B のほうが，財務会計（支払）ドメインにおいて DODDLE-OWL を用いた被験者 A よりも，入力語候補からの不要語の削除や抽出できなかった入力語の追加に多くの時間がかかったと考えられる．上記の結果より，文書集合やドメインにより DODDLE-OWL の入力語抽出精度が低い場合には，DODDLE-OWL を用いることにより入力語抽出時間は短縮できないことがあるといえる．しかしながら，表 4.18 および表 4.21 より，被験者 A と被験者 B ともに DODDLE-OWL を用いたほうが入力語を多く抽出できている．これは，手動で入力語抽出をする際には，類似語などを見落とすことが多いためである．このことから，DODDLE-OWL を用いることで入力語の抽出漏れを防ぐことができると考えられる．本実験において，被験者 A と被験者 B ともに DODDLE-OWL を用いたほうがオントロジーに含まれる概念数が多くなっているのも入力語の抽出漏れが低減し，入力語数が増えたためである．

入力概念選択時間については，被験者 A と被験者 B ともに 8 分程度であり，オントロジー構築時間全体から見るとコストは高くないといえる．

is-a 関係構築については，被験者 A と被験者 B ともに DODDLE-OWL を用いたほうが構築時間が短縮できているため，DODDLE-OWL が有効に機能しているといえる．

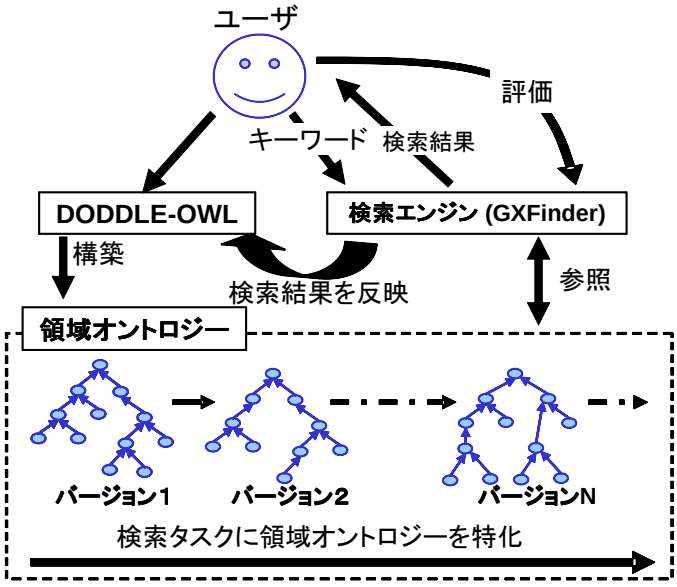


図 4.42: オントロジーベースの検索システム

被験者からは，手動構築の場合には入力文書からの入力語抽出や入力語をクラスとして Protégé へ入力する手間など，オントロジー構築に本質的には重要でない作業に時間がかかるという感想が得られた．一方，DODDLE-OWL を用いてオントロジーを構築する場合には，それらの手間をツールが支援するため，被験者は本質的にオントロジー構築に重要な作業に集中することができたという感想が得られた．このことから，質の高いオントロジー構築を DODDLE-OWL が支援できる可能性があるといえる．

4.13 ケーススタディ

4.13.1 概要

DODDLE-OWL のスケーラビリティを評価するために，DODDLE-OWL を用いた大規模オントロジーの構築および評価を行った．

精度の高い検索を実現するために，検索キーワードによる検索ではなく，文書の意味を考慮した意味検索に関する研究が多数行われている．オントロジーを用いて意味検索をするための研究も行われているが，文書検索に用いるオントロジーは，網羅性が必要となるため規模が大きくなる．そのため，オントロジー構築コストが高いという問題がある．本ケーススタディでは，大規模オントロジーの構築コストを DODDLE-OWL が削減できるかどうかを検証し，構築したオントロジーを文書検索に適用した際に適切に機能するかどうかを検証することにより，DODDLE-OWL のスケーラビリティの評価とする．

図 4.42 に DODDLE-OWL を初期オントロジー構築に用いた検索システムを示す．検索システムは，(株）ギャラクシーエクスプレス社（GX 社）で開発された社内文書検索システム GXFinder [66] を用いる．GXFinder は OWL 形式の領域オントロジーを搭載し，オ

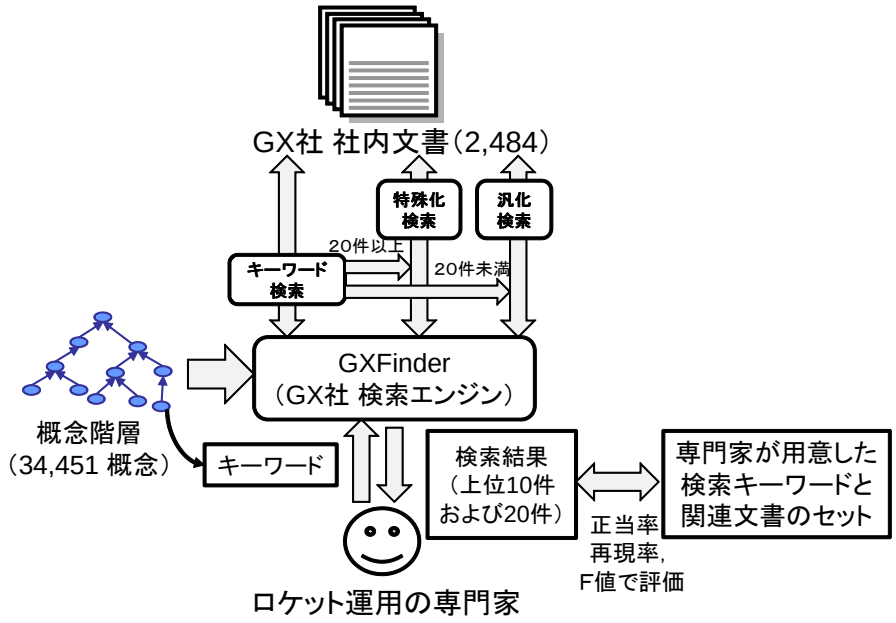


図 4.43: 検索実験の方法

ントロジーを用いた検索を行うことができる．オントロジーベースの検索システムでは，検索結果に対するユーザからのフィードバックを得ることによって，領域オントロジーを洗練しながら検索に特化した領域オントロジーを構築することが必要となる．しかし，検索結果に対するユーザからのフィードバックを領域オントロジーに反映させる部分については，DODDLE-OWL の研究の範囲外であるため，本ケーススタディでは扱わないこととする．

4.13.3 項で述べる手順で構築したロケット運用オントロジーを GXFinder に搭載し，検索実験を行った．図 4.43 に検索実験の方法を示す．検索はロケット運用に詳しい専門家が行った．検索コンテクストを仮定せずに検索対象となる文書を専門家が思い浮かべることは困難であるため，領域オントロジーの概念階層を専門家に見てもらいながら検索対象となる文書（正解文書）を想定してもらった．検索対象となる文書を専門家が選択後，はじめにキーワード検索を行い，次に検索結果数が多すぎる（20 件以上）場合には絞込検索を，少なすぎる場合（20 件未満）には拡大検索を行った．検索結果，上位 10 件および 20 件について，適合率 (Precision)，再現率 (Recall)，F 値 (F-measure) [67] を用いて評価を行った．ここで，適合率は，上位 10 件および 20 件の検索結果に含まれる，正解文書の割合を表す．再現率は，正解文書を上位 10 件および 20 件の検索結果が網羅する割合を表す．F 値は，適合率と再現率の調和平均であり，両者を総合的に評価するための指標である．適合率 P，再現率 R，F 値 F は，式 4.10 から式 4.12 で表される．式 4.10 から式 4.12 では，ある検索キーワードについて，キーワード検索またはオントロジー検索を用いた検索結果，上位 10 件または 20 件に含まれる文書集合を D，正解文書集合を SD としている[14]．

[14]検索結果が 10 件以下の場合は，検索結果に含まれる全文書集合が D となる．

$$P := \frac{|D \cap SD|}{|D|} \tag{4.10}$$

$$R := \frac{|D \cap SD|}{|SD|} \tag{4.11}$$

$$F = \frac{2PR}{P+R} \tag{4.12}$$

以下では，領域オントロジーを用いた文書検索，DODDLE-OWL を用いたロケット運用オントロジーの構築，キーワード検索および領域オントロジーを用いた検索の比較実験について述べる．また，実験の考察について述べる．

4.13.2 オントロジーを用いた文書検索

GXFinder は，DODDLE-OWL により構築された OWL 形式の領域オントロジーを搭載し，概念階層を用いた特殊化検索および汎化検索を行うことができる．特殊化検索と汎化検索の定義は以下のとおりである．

特殊化検索

特殊化検索は，検索結果数が膨大な場合に検索結果数を絞込み，ユーザの目的に合った文書を見つけやすくする．検索キーワードを見出しとして持つ概念の下位概念について，それぞれの概念見出しを OR で結合し，検索を行う．例えば,「油」をキーワードとして絞込検索を行うとする.「油」の下位概念には「残油」「既存油」「禁油」「作動油」「防油」が存在するとする．GXFinder で絞込検索を実行すると，上記のキーワードを OR で結合し，検索を行う．何階層下まで展開するかはユーザが指定することができる．

汎化検索

汎化検索は，検索結果数が少なすぎる場合や検索したいキーワードをユーザが漠然としか思い浮かばない場合に，関連する文書を多くユーザに提示し，目的に合った文書を見つけやすくする．汎化検索は，検索キーワードを見出しとして持つ概念の上位概念および兄弟概念について，それぞれの概念見出しを OR で結合し検索を行う．例えば,「油」の上位概念は,「油脂」であり，兄弟概念には「灯油」,「軽油」などが存在するとする．拡大検索を行うと上記のキーワードが展開され，検索が実行される.「油」のみで検索する場合に比べ，油に類似するキーワードを含む文書を検索することができ，目的の文書が見つからない場合でも，目的と類似する文書を見つけることが可能であると考えられる．特殊化検索と同様に，何階層上まで展開するかはユーザが指定することが可能である．

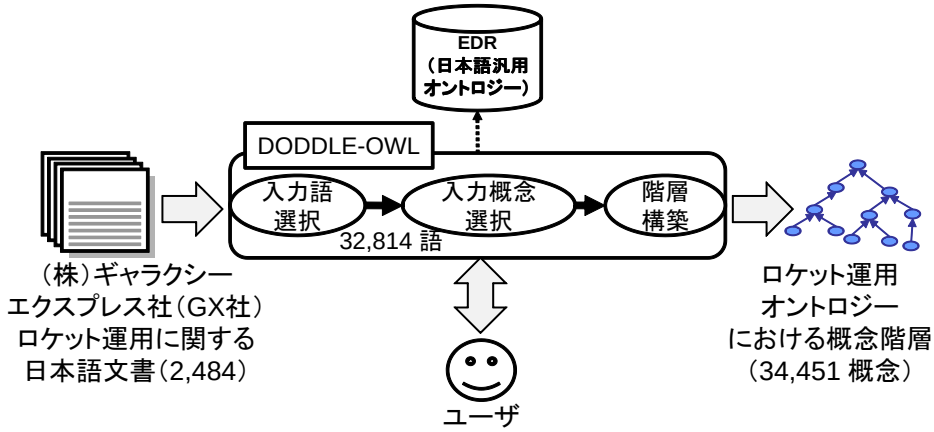


図 4.44: ロケット運用オントロジー構築手順

表 4.23: ロケット運用オントロジー構築における自動抽出語数，入力語数，完全照合語数，部分照合語数，未定義語数，全概念数

自動抽出語数	41,806
入力語数	32,814
完全照合語数	4,982
部分照合語数	26,835
未定義語数	997
全概念数	34,451

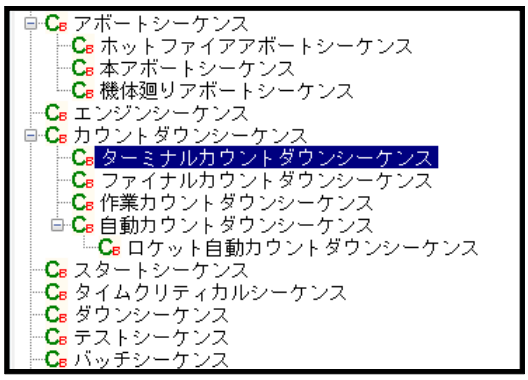
4.13.3 ロケット運用オントロジーの構築

ロケット運用オントロジーの構築は，DODDLE-OWL を用いて，1 人のユーザが約 30 時間かけて行った．ユーザは，GX 社豊洲分室で作成されたロケット運用に関する 2,484 の日本語文書を入力文書としてオントロジーの構築を行った．図 4.44 にロケット運用オントロジー構築手順を示す．

第 1 に，ユーザは入力文書選択モジュールにより，入力文書から名詞，動詞，複合語を自動抽出した．第 2 に，ユーザは入力語選択モジュールにより，自動抽出された用語の中から不要語の削除および入力語の選択を行った．第 3 に，ユーザは入力概念選択モジュールにより，入力概念の選択を行った．部分照合語数が膨大であったことから，部分照合語の入力概念選択については，同様の照合概念を持つ部分照合語の入力概念を統一し，個々の部分照合語についての入力概念選択は行わなかった．同様に，部分照合語については，照合概念の下位概念とするか，照合概念の別見出しとするかをユーザは選択可能であるが，本ケーススタディでは，部分照合語数が膨大なことから，すべて照合した概念の下位概念として階層構築を行った．

表 4.23 に，ロケット運用オントロジー構築における自動抽出語数，入力語数，完全照合語数，部分照合語数，未定義語数，全概念数を示す．

「管制卓」周辺の概念階層



「ターミナルカウントダウンシーケンス」周辺の概念階層

図 4.45: 「管制卓」および「カウントダウンシーケンス」周辺の概念階層

本来は，概念階層構築後に，階層洗練モジュールを用いて概念階層を領域に特化した構造に洗練をすべきである．しかし，今回は，概念数が膨大（約 34,000 概念）であったために，洗練手法が示唆する概念変動が生じていると想定される箇所も膨大（約 2,000）となったため，オントロジーの洗練を行っていない．よって，上述のオントロジー構築時間（約 30 時間）は，入力語選択時間および入力概念選択時間を表しており，初期概念階層の洗練時間は含まれていない．

4.12 節で述べた DODDLE-OWL と手動構築との比較実験より，数百概念規模のオントロジー構築経験が何度かあるユーザが，約 200 概念を含む概念階層を手動構築するのに約 2 時間程度の時間がかかる．約 34,000 概念を含む概念階層構築時間を見積もることは困難であるが，手動構築により概念階層を構築する場合には少なくとも数百時間はかかると考えられる．よって，本ケーススタディでは，階層洗練を十分に行うことはできていないが，約 30 時間で約 34,000 概念を含む概念階層を構築することができたということは，非常に短時間で概念階層を構築できたといえる．

4.13.4 結果

本ケーススタディでは，二つの実験を行った．実験 1 は,「発射管制卓」に関する文書を探したいが,「発射」が思い浮かばず,「管制卓」で検索を行うという想定で行った．実験 2 は,「ターミナルカウントダウンシーケンス」に関する文書を探したいが,「ターミナル」が思い浮かばず,「カウントダウンシーケンス」で検索を行うという想定で行った．図 4.45 に「管制卓」および「カウントダウンシーケンス」周辺の概念階層を示す．表 4.24 に実験 1 および実験 2 におけるキーワード検索ヒット数，特殊化検索ヒット数，上位 10 件および 20 件のキーワード検索および特殊化検索における正解文書数，全文書中の正解文書数を示す．また，表 4.25 および表 4.26 に表 4.24 より求めた，検索結果上位 10 件および 20 件におけるキーワード検索と特殊化検索における適合率，再現率，F 値を示す．

表 4.24: 実験 1 および 2 の検索結果

	実験 1	実験 2
正解文書数	4	22
キーワード検索ヒット数	86	82
キーワード検索上位 10 件の正解文書数	1	3
キーワード検索上位 20 件の正解文書数	3	3
特殊化検索ヒット数	44	46
特殊化検索上位 10 件の正解文書数	3	8
特殊化検索上位 20 件の正解文書数	4	11

表 4.25: 実験 1 の検索結果

	再現率	適合率	F 値
キーワード検索 上位 10 件	0.250	0.100	0.143
特殊化検索 上位 10 件	0.750	0.300	0.429
キーワード検索 上位 20 件	0.750	0.150	0.250
特殊化検索 上位 20 件	1.000	0.200	0.333

4.13.5 考察

表 4.25 および表 4.26 より，特殊化検索はキーワード検索に比べ適合率，再現率，F 値のすべてにおいて高い検索精度を得ることができた．特殊化検索では，領域オントロジーの概念階層構造を用いて，より詳しいキーワードに絞り込んで検索をすることにより，ユーザが検索キーワードを明確に思い浮かべることができない状況でも，目的の文書を探すことを容易にできると考えられる．汎化検索については，うまく機能する例題が得られなかった．原因の一つとして，概念階層が洗練されていなかったために，兄弟概念数が膨大となっており，汎化検索を行うと幅広くキーワードが展開されてしまい，検索結果も膨大となったことがあげらえる．汎化検索を有効に機能させるためには，概念階層の洗練（特に兄弟概念数の削減）が必要である．本ケーススタディでは，DODDLE-OWL が提供する階層洗練モジュールが，概念数が膨大であったためにうまく機能しなかった．概念数が膨大な場合にも適切に機能する新たな階層洗練戦略の考案は今後の課題である．

本ケーススタディでは，表 4.23 に示した未定義語，約 1,000 語を概念階層中に定義していない．領域オントロジーを構築したユーザが専門家ではなかったことと，現状では，DODDLE-OWL には，未定義語を概念階層中の適切な位置に定義することを支援する機能がないためである．専門家は 2 時間程度の検索実験の間に，未定義語の概念階層中に定義すべき位置などを多く指摘していた．未定義語は領域に特化した概念となる可能性が高いため，今後，専門家が未定義語を概念階層中の適切な位置に定義することを支援する機能を考える必要がある．

表 4.26: 実験 2 の検索結果

	再現率	適合率	F 値
キーワード検索 上位 10 件	0.136	0.300	0.188
特殊化検索 上位 10 件	0.364	0.800	0.500
キーワード検索 上位 20 件	0.136	0.150	0.143
特殊化検索 上位 20 件	0.500	0.550	0.524

また，DODDLE-OWL の階層構築方法では対応できない例が本ケーススタディを通していくつか得ることができた．例えば、「中部ガス」というガス会社をあらわす概念の階層関係を定義する場合に，参照オントロジー中に「ガス会社」に関する階層関係の定義がなく，気体としての「ガス」概念の階層関係の定義しかない場合を考える．その場合には,「中部ガス」は気体としての「ガス」概念と部分照合し,「可燃ガス」概念や「ヒドラジンガス」概念などの兄弟概念として誤って階層化される．また，本ケーススタディでは,「ガス」概念の下位概念に「有毒ガス」が定義されていたが,「有毒ガス」概念の下位概念は適切に階層化できていなかった.「有毒ガス」は,「〜有毒ガス」という見出しではなく,「一酸化炭素」のように見出し上は「有毒ガス」との関連はわからない．よって，ロケット運用における有毒ガスを自動的に階層化することは，現在の DODDLE-OWL の階層構築方法では困難である．さらに，本ケーススタディにおいて「管制卓」概念は「卓」概念の下位概念として概念化されている．しかしながら,「管制卓」は管制を行う場所を意味しているため,「机」と同義である「卓」の下位概念に定義されることは誤っている．このような専門領域に特化した概念階層の定義については，手動による洗練が不可欠である．

本ケーススタディを通して，専門家はロケット運用オントロジーを見て，いくつかの階層関係の誤りを指摘していた．本ケーススタディで構築されたロケット運用オントロジーは，汎化検索には十分な精度を発揮できないなど，洗練の余地は多くある．しかしながら，手動構築では数百時間かかると考えられる約 34,000 概念を含む概念階層を約 30 時間で構築することができた．また，専門家と概念階層のどこを修正すべきかを議論することができた．このことから，概念数が膨大な領域オントロジー構築にも DODDLE-OWL の有用性を示すことができたといえる．

4.14 既存オントロジーの獲得およびランキング実験

4.14.1 概要

4.4 節で述べた手法を用いて，Swoogle を用いた既存オントロジーの獲得およびランキングの有用性を評価するための実験を行った．本実験では，入力文書として法律分野の専門文書である国際売買法 [68] 第二部の英文テキスト（以下，CISG）を用いた．CISG に関する入力語選択，入力概念選択，階層構築，その他の関係定義は， [19] において，法

表 4.27: CISG Part-II から専門家が選択した 46 の入力語

acceptance	act	addition	address	assent	circumstance	communication system
conduct	contract	counteroffer	day	delay	delivery	discrepancy
dispatch	effect	envelope	goods	holiday	indication	intention
invitation	letter	modification	offer	offeree	offerer	party
payment	person	place of business	price	proposal	quality	quantity
rejection	reply	residence	revocation	silence	speech act	telephone
telex	time	transmission	withdrawal			

表 4.28: 獲得オントロジー数，入力語の網羅率，入力概念に関連するクラス数およびプロパティ数

獲得オントロジー数	120
入力語の網羅率	0.717 (33/46)
クラス数	331
プロパティ数	558

律の専門家によってなされている．よって，本実験では，専門家によって選択された 46 の入力語（表 4.27）をオントロジー選択モジュールの入力として，既存オントロジーの獲得およびランキングを行った．また，獲得された既存オントロジーが対象領域のオントロジー構築に再利用可能であるかどうかは，専門家が定義した階層関係と比較することにより確認した．本実験は図 4.2 に示す手順で行った．

4.14.2 結果

表 4.28 に獲得オントロジー数，入力語の網羅率，入力概念に関連するクラス数およびプロパティ数を示す．表 4.29 に，Swoogle を用いて獲得した，オントロジーに含まれる入力概念数でソートした上位 10 個の既存オントロジーを示す．なお，本実験は，2007 年 6 月に行った．

4.14.3 考察

表 4.28 より，CISG の約 7 割の入力語については，獲得したオントロジーで網羅することができる．しかし，表 4.29 より，入力概念を含む数が最も多い既存オントロジーで 3 割程度であり，入力概念をすべて含む領域オントロジーを構築するためには，複数の既存オントロジーを参照する必要があることがわかる．また，入力概念を多く含むオントロジーの大部分は，WordNet [17] や Cyc [15]などの汎用オントロジーであり，領域オントロ

[15]http://www.cyc.com/

表 4.29: Swoogle を用いて獲得した既存オントロジー（オントロジーに含まれる入力概念数でソートした上位 10 個）

順位	入力概念数 (OntoRank)	オントロジーの URL
1	15 (0.881)	http://www.loa-cnr.it/ontologies/OWN/OWN.owl
2	15 (0.881)	http://www.w3.org/2001/sw/BestPractices/WNET/wnNounsyn_v7.owl
3	12 (642.815)	http://morpheus.cs.umbc.edu/aks1/ontosem.owl
4	12 (0.860)	http://www.cs.umbc.edu/ aks1/ontosem.owl
5	11 (0.735)	http://rhizomik.net/ontologies/2005/07/FrameNet_1.1_inferred.owl
6	11 (0.726)	http://rhizomik.upf.edu/ontologies/2005/07/FrameNet_1.1.owl
7	11 (0.726)	http://rhizomik.net/ontologies/2005/07/FrameNet_1.1.owl
8	10 (0.875)	http://athena.ics.forth.gr:9090/RDF/VRP/Examples/DCD100.rdf
9	10 (0.875)	http://139.91.183.30:9090/RDF/VRP/Examples/DCD100.rdf
10	9 (1.701)	http://www.cyc.com/2004/06/04/cyc

ジーではなかった．図 4.2 の手順 1 から 3 で，TermRank の高いクラスおよびプロパティを獲得し，それを基に既存オントロジーを獲得したため，最終的に汎用的なオントロジーを多く獲得してしまったと考えられる．しかしながら，8 位および 9 位にランキングされたオントロジーは，電子商取引に関するオントロジーであり，法律概念（contract，offeree，party，proposal など）をいくつか含んでいた．これらの概念は，法律に特化した概念関係の定義がされていたため，国際売買に関するオントロジー構築に利用可能だと考えられる．

オントロジー選択モジュールでは，入力語と完全に一致する見出しを含むクラスおよびプロパティを獲得し，それを基に既存オントロジーを獲得していた．しかし，8 位および 9 位にランキングされたオントロジーの内容を確認してみると，HomeAddress，OrderQuantity，CreditAcceptances，HolidayCalendar などのクラスが定義されていた．これらは，CISG から専門家が選択した入力語である address，quantity，acceptance，holiday とほぼ同様の意味としてとらえることができる．よって，入力語とそれに関連するクラスおよびプロパティの獲得方法を工夫することで，より適切な既存オントロジーを獲得できる可能性があるといえる．例えば，今回の例では，入力語と見出しが部分照合するクラスおよびプロパティを獲得する方法が考えられる．また，入力語の同義語を見出しとして持つクラスおよびプロパティの獲得も領域に関連する既存オントロジー獲得に有効であるかもしれない．

表 4.29 より，内容は等しいが URL のみが異なる既存オントロジーが多く獲得された．Swoogle を用いて獲得した既存オントロジーの中から，ユーザが領域に関連するオントロジーを選択する際には，内容の等しい既存オントロジーを一つにまとめて提示できるとよいと思われる．本実験では，owl:imports プロパティ以外の既存オントロジーにおける OWL メタデータを利用していない．バージョン情報などがあれば，バージョン違いのオントロジーを同定することができると考えられる．また，OWL メタデータがない場合には，URL の類似度やオントロジーに定義されているクラスやプロパティの類似度などか

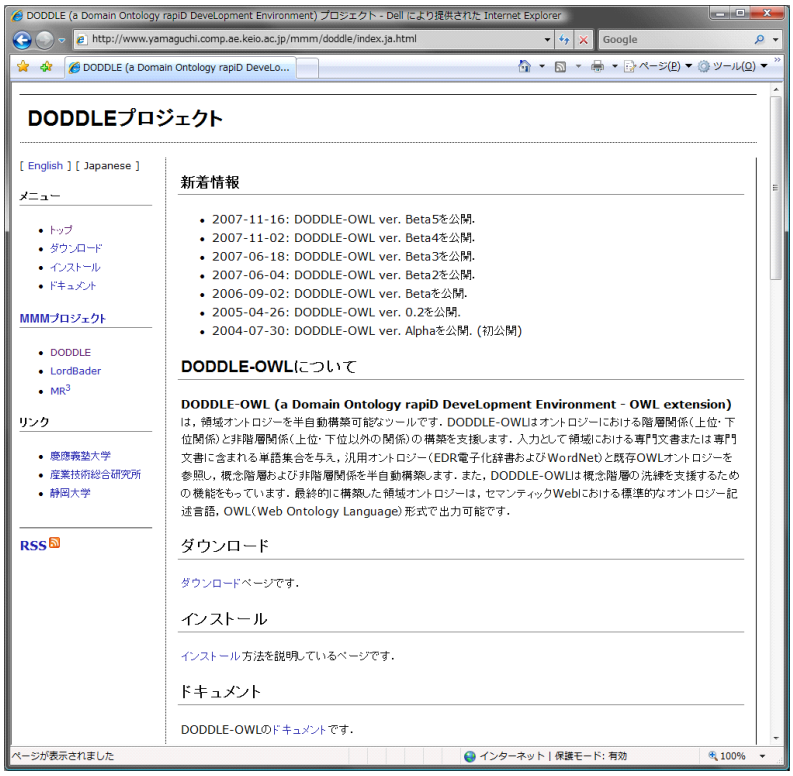


図 4.46: DODDLE プロジェクト Web サイト

ら，内容の等しいオントロジーを同定できると考えられる．

4.15 オープンソースソフトウェアとしての領域オントロジー構築支援環境

本章で提案する領域オントロジー構築支援環境を，知識システムやセマンティック Web のためのオントロジー構築を行うコミュニティに広く利用してもらえるように，DODDLE-OWL のシステムおよびソースコードを Web 上で公開している（図 4.46）[16]．2007 年 12 月現在，20 カ国，約 150 人のユーザ登録がなされており，国内外の企業や研究所で利用されつつある．利用者からのコメントは，実用的な領域オントロジー構築支援環境を実現するためには重要であり，将来的に DODDLE-OWL の実装に反映していきたいと考えている．

4.16 まとめ

本章では，インタラクティブな領域オントロジー構築支援環境 DODDLE-OWL について述べた．DODDLE-OWL は，領域専門文書を入力として，WordNet や EDR 電子化辞

[16] http://www.yamaguti.comp.ae.keio.ac.jp/mmm/doddle/

書などの既存オントロジーを参照しながら，領域オントロジーにおける階層関係およびその他の関係の半自動構築を支援する．DODDLE-OWL は Java 言語で実装され，オープンソースソフトウェアとして Web で公開されている．2007 年 12 月現在，20 カ国，約 150 人のユーザ登録がなされており，国内外の企業や研究所で利用されつつある．

DODDLE-OWL の有用性を評価するために，DODDLE-OWL と手動構築の比較実験を行った．被験者 A と B が財務会計と人事ドメインそれぞれについて，DODDLE-OWL および手動構築により 200 から 400 概念程度の概念階層を構築した．その結果，被験者 A と B 共に，手動構築と比べて DODDLE-OWL を用いたほうが約 1 時間早く概念階層を構築することができた．この実験からは，DODDLE-OWL を利用することによる，オントロジーの規模とオントロジー構築時間の相関までは知ることはできない．しかしながら，数百概念程度の概念階層構築には DODDLE-OWL は有効であるといえる．

DODDLE-OWL のスケーラビリティを評価するために，DODDLE-OWL を用いて，ロケット運用に関する約 34,000 概念を含む大規模オントロジーの構築および評価を行った．大規模オントロジーを評価するために，構築したオントロジーをオントロジーベースの検索システムに搭載し，ギャラクシーエクスプレス社の協力の元，文書検索実験を行った．文書検索実験では，キーワード検索とオントロジーを用いた検索（特殊化検索と汎化検索）の比較を行い，正答率，再現率，F 値で評価を行った．その結果，特殊化検索ではキーワード検索と比べて高い精度で検索を行うことができた．しかしながら，汎化検索については，大規模オントロジーの洗練を十分に行うことができなかったため，有効に機能しなかった．この実験では，数百時間はかかると考えられる約 34,000 概念を含む概念階層構築を 1 人のユーザが約 30 時間で構築できた．このことより，DODDLE-OWL を用いることで，非常に低いコストで大規模オントロジーを構築することができたといえる．また，専門家と構築したオントロジーの洗練についての議論を行うことができた．以上より，DODDLE-OWL は大規模オントロジー構築にも有用であると考えられる．

最後に，オントロジー選択モジュールの評価として，既存オントロジーの獲得およびランキングの実験を行った．国際売買法に関する領域オントロジー構築に再利用可能な既存オントロジーの獲得を試みた．結果として，上位にランキングされたオントロジーの大部分は汎用オントロジーであった．しかしながら，8 位および 9 位にランキングされた電子商取引に関するオントロジーは，法律に関するクラスがいくつか定義されており，国際売買法に関するオントロジー構築に再利用可能であった．

上記の評価実験より，DODDLE-OWL の有用性を確認することができた．

今後の課題として，大規模オントロジーにおける階層洗練手法があげられる．また，複数の参照オントロジーから関連する概念を抽出，合成して概念階層を構築する場合には，上位概念階層の構造の違いにより単純に合成することは困難であると考えられる．その際には，オントロジーアライメントにより類似概念を同定する必要がある．オントロジーアライメントを用いた階層構築支援も今後の課題である．

第5章　結論

本章では，本論文のまとめと今後の課題および展望について述べる．

本論文では，セマンティック Web コンテンツ構築支援ツールを実装し，評価することを目的として，RDF(S) コンテンツ構築支援ツールおよび領域オントロジー構築支援環境の提案を行った．

RDF(S) コンテンツ構築支援ツールは，RDF(S) コンテンツ管理機能と RDF(S) コンテンツの視覚的編集機能を持つ実用的な RDF(S) エディタである．本論文では，RDF(S) コンテンツ管理機能を「RDF コンテンツおよび RDFS コンテンツをモデルとオントロジーの関係として捉え，RDF コンテンツと RDFS コンテンツを分離し，両者の間の整合性を半自動的に管理する機能」と定義した．本論文では，RDF(S) コンテンツ管理機能をオントロジーの変更をモデルに反映させる機能である **O→M** とモデルの変更をオントロジーに反映させる機能である **M→O** の大きく分けて 2 種類に分類した．また，RDF(S) コンテンツ構築支援ツールのプロトタイプ実装である MR^3 を Java 言語を用いて実装した．MR^3 を評価するために，関連ツールとの機能比較，RDF(S) コンテンツ管理機能および RDF(S) コンテンツの視覚的編集機能の評価実験を行った．関連ツールとの機能比較の結果，MR^3 の RDF(S) コンテンツ管理機能における **M→O** は，MR^3 独自の機能であることを示すことができた．また，RDF(S) コンテンツ管理機能および RDF(S) コンテンツの視覚的編集機能の評価実験の結果，MR^3 の有用性を示すことができた．

領域オントロジー構築支援環境 DODDLE-OWL は，領域専門文書を入力として，WordNet や EDR 電子化辞書などの既存オントロジーを参照しながら，領域オントロジーにおける階層関係およびその他（階層以外）の関係の半自動構築を支援する．DODDLE-OWL は，様々なモジュールを統合することにより，インタラクティブで実用的な領域オントロジー構築支援環境となっている．DODDLE-OWL を評価するために，関連研究との比較，DODDLE-OWL と手動構築の比較実験，大規模オントロジーの構築，既存オントロジーの獲得およびランキングの実験を行った．

DODDLE-OWL と手動構築の比較実験では，被験者 A および B が財務会計と人事ドメインそれぞれについて，DODDLE-OWL および手動構築により 200 から 400 概念程度の概念階層を構築した．その結果，被験者 A および B 共に，手動構築と比べて DODDLE-OWL を用いたほうが約 1 時間早く概念階層を構築することができた．この実験からは，DODDLE-OWL を利用することによる，オントロジーの規模とオントロジー構築時間の相関までは知ることはできない．しかしながら，数百概念程度の概念階層構築には DODDLE-OWL は有効であるといえる．

DODDLE-OWL のスケーラビリティを評価するために，DODDLE-OWL を用いて，ロ

ケット運用に関する約 34,000 概念を含む大規模オントロジーの構築を行った．大規模オントロジーを評価するために，構築したオントロジーをオントロジーベースの検索システムに搭載し，ギャラクシーエクスプレス社の協力の元，文書検索実験を行った．文書検索実験では，キーワード検索とオントロジーを用いた検索 (特殊化検索および汎化検索) の比較を行い，正答率，再現率，F 値で評価を行った．その結果，特殊化検索ではキーワード検索と比べて高い精度で検索を行うことができた．しかしながら，汎化検索については，大規模オントロジーの洗練を十分に行うことができなかったため，有効に機能しなかった．この実験では，数百時間はかかると考えられる約 34,000 概念を含む概念階層構築を 1 人のユーザが約 30 時間で構築できた．このことより，DODDLE-OWL を用いることで，非常に低いコストで大規模オントロジーを構築することができたといえる．また，専門家と構築したオントロジーの洗練についての議論を行うことができた．以上より，DODDLE-OWL は大規模オントロジー構築にも有用であるといえる．

最後に，オントロジー選択モジュールの評価として，既存オントロジーの獲得およびランキングの実験を行った．国際売買法に関する領域オントロジー構築に再利用可能な既存オントロジーの獲得を試みた．結果として，上位にランキングされたオントロジーの大部分は汎用オントロジーであった．しかしながら，8 位および 9 位にランキングされた電子商取引に関するオントロジーは，法律に関するクラスがいくつか定義されており，国際売買法に関するオントロジー構築に再利用可能であった．

上記の評価実験より，DODDLE-OWL の有用性およびスケーラビリティを確認することができた．

MR^3 と DODDLE-OWL は共に，オープンソースソフトウェアとして Web で公開されている．2007 年 12 月現在，両方のツールを合わせて，約 45 カ国，約 1,000 人のユーザ登録がなされており，国内外の企業や研究所で利用されつつある．利用者からのコメントは，実用的なセマンティック Web コンテンツ構築支援ツールを実現するためには不可欠であり，将来的に両ツールの実装に反映していきたいと考えている．また，現状では，ツールは公開しているが，セマンティック Web コンテンツの公開は行っていない．今後は，本論文で提案したセマンティック Web コンテンツ構築支援ツールを用いて構築したセマンティック Web コンテンツも公開していきたい．

RDF(S) コンテンツ構築支援ツールの今後の課題として，OWL Lite レベルのオントロジー構築支援と OWL コンテンツと RDF コンテンツに関する整合性管理機能の整理がある．MR^3 は，プラグイン機構により，DODDLE-OWL と連携可能となっている．DODDLE-OWL により，半自動的に構築された OWL 形式のオントロジーをインポートし，視覚的な編集を行うことができる．しかしながら，OWL については現状では MR^3 は部分的にしか対応していない． MR^3 の今後の課題として，OWL Lite レベルのオントロジーの視覚的編集を支援することがあげられる．また，RDF(S) コンテンツ管理機能についても，現状では RDFS レベルの支援にとどまっている．OWL のクラス公理，プロパティ制約などが定義されているオントロジーとそのモデルの間に発生する種々の整合性管理機能を整理することも，今後の課題である．

領域オントロジー構築支援環境の今後の課題として，大規模オントロジーにおける洗練

手法の考案と複数の既存オントロジーからの領域オントロジー構築支援がある．本論文において大規模オントロジーの構築実験を行ったが，既存のオントロジー洗練手法は大規模オントロジーの洗練には十分に機能しなかった．概念数が膨大なオントロジーにも機能するオントロジー洗練手法を考案する必要がある．また，本論文では対象領域に関連する既存オントロジーを獲得する手法を提案した．しかしながら，汎用オントロジーを参照オントロジーとする場合とは異なり，複数の既存オントロジーを参照オントロジーとする場合には，様々な問題がある．例えば，複数の既存オントロジーから関連する概念を抽出，合成して概念階層を構築する場合には，上位概念階層の構造の違いにより単純に合成することは困難であると考えられる．その際には，オントロジーアライメントにより類似概念を同定する必要がある．オントロジーアライメントを用いた階層構築支援も今後の課題である．

また，ツールと方法論は一体となって意味がある．本論文で提案したセマンティック Web コンテンツ構築支援ツールを用いたセマンティック Web コンテンツ構築方法論を考えることも今後の課題である．

最後に，今後の展望として，Wikipedia とフォークソノミーからのセマンティック Web コンテンツ構築支援が考えられる．本論文では，セマンティック Web コンテンツ構築支援のために，既存情報リソースとして，主に専門文書および汎用オントロジーを用いた．近年，第 3 のリソースとして，Wikipedia とフォークソノミーが注目を集めている．フォークソノミーとは，多数のユーザが Web リソースへ各ユーザの指向に基づきタグ（キーワード）を付与することにより，Web リソースの分類や検索に役立てる仕組みである．Wikipedia は，多数のユーザが自由に執筆できる Web 上の百科事典である．Wikipedia もフォークソノミーも共に，多数の人からの知識を自動的に集約していることから，集合知と呼ばれる．Wikipedia もフォークソノミーも，現状では，人間を情報参照の対象としており，セマンティック Web が目指すソフトウェアによる理解は困難である．しかしながら，Wikipedia もフォークソノミーも一定の構造に従って記述されていることから，半構造データとみなすことができる．半構造データは，専門文書などの非構造データと比べると，より形式的なセマンティック Web コンテンツに近いといえる．現在，Wikipedia からセマンティック Web コンテンツを構築する研究がはじまっている [69, 70]．Wikipedia やフォークソノミーの利点は，コンテンツの網羅性と更新の即時性にある．Wikipedia やフォークソノミーからセマンティック Web コンテンツを構築することができれば，急速にセマンティック Web コンテンツが普及し，セマンティック Web の実現がより現実的になると考えられる．現状では，Wikipedia から汎用オントロジーの構築を支援する研究はなされているが，本論文で提案しているような各領域ごとのオントロジー構築支援に関する研究はなされていない．今後は，Wikipedia やフォークソノミーを第 3 の既存情報リソースとして，各領域に特化したオントロジー構築支援にも取り組んでいきたい．

参考文献

[1] T. Berners-Lee, J. Hendler, and O. Lassila, “The Semantic Web,” Scientific American, pp.34-43, 2001.

[2] N. Shadbolt, T. Berners-Lee, and W. Hall, “The Semantic Web Revisited,” IEEE Intelligent Systems, vol.21, no.3, pp.96-101, 2006.

[3] T. Berners-Lee, R. Fielding, and L. Masinter, “RFC 3986: Uniform Resource Identifier (URI): Generic Syntax,” IETF, 2005, http://www.rfc-editor.org/rfc/rfc3986.txt.

[4] T. Bray, J. Paoli, C.M. Sperberg-McQueen, E. Maler, and F. Yergeau, “Extensible Markup Language (XML) 1.0 (Fourth Edition),” W3C Recommendation, 2006, http://www.w3.org/TR/REC-xml/.

[5] F. Manola, and E. Miller, “RDF Primer,” W3C Recommendation, 2004, http://www.w3.org/TR/rdf-primer/.

[6] D. Brickley, and R. Guha, “RDF Vocabulary Description Language 1.0: RDF Schema,” W3C Recommendation, 2004, http://www.w3.org/TR/rdf-schema/.

[7] M.K. Smith, C. Welty, and D.L. McGuinness, “OWL Web Ontology Language Guide,” W3C Recommendation, 2004, http://www.w3.org/TR/owl-guide/.

[8] E. Prud’hommeaux, and A. Seaborne, “SPARQL Query Language for RDF,” W3C Recommendation, 2008, http://www.w3.org/TR/rdf-sparql-query/.

[9] T. Berners-Lee, “Notation 3: Ideas about Web Architecture - yet another notation,” W3C, 2001, http://www.w3.org/DesignIssues/Notation3.

[10] D. Beckett, and A. Barstow, “N-Triples,” W3C RDF Core WG Internal Working Draft, 2001, http://www.w3.org/2001/sw/RDFCore/ntriples/.

[11] D. Beckett, “Turtle - Terse RDF Triple Language,” —, 2004, http://www.dajobe.org/2004/01/turtle/.

[12] Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer, and D. Wenke, “OntoEdit: Collaborative Ontology Engineering for the Semantic Web,” Proceedings of the First Internation Semantic Web Conference, no.2342, pp.221–235, Springer Verlag, 2002.

[13] H. Knublauch, R.W. Fergerson, N.F. Noy, and M.A. Musen, “The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications,” Proceedings of the Third International Semantic Web Conference, pp.229–243, 2004, http://protege.stanford.edu/.

[14] Y. Ding, and S. Foo, “Ontology Research and Development, Part 1 – A Review of Onlotogy,” Journal of Information Science, pp.123–136, 2002.

[15] A. Maedche, and S. Staab, “Discovering Conceptual Relations from Text,” Proceedings of the ECAI2000, pp.321–325, 2000.

[16] D. Faure, and C. Nedellec, “Knowledge Acquisition of Predicate Argument Structures from Technical Texts Using Machine Learning: The System ASIUM,” Proceedings of the EKAW, vol.1621, pp.329-334, Springer, 1999.

[17] G.A.Miller, “WordNet: A Lexical Database for English,” Commun. ACM, vol.38, no.11, pp.39–41, 1995.

[18] T. Yokoi, “The EDR Electronic Dictionary,” Commun. ACM, vol.38, no.11, pp.42–44, 1995, http://www2.nict.go.jp/r/r312/EDR/.

[19] M. Kurematsu, T. Iwade, and T. Yamaguchi, “DODDLE II : A Domain Ontology Development Environment Using a MRD and Text Corpus,” IEICE transactions on information and systems, vol.87, no.4, pp.908-916, 2004.

[20] 神崎正英，セマンティック・ウェブのための RDF/OWL 入門，森北出版株式会社，2005.

[21] T. Bray, D. Hollander, A. Layman, and R. Tobin, “Namespaces in XML 1.0 (Second Edition),” W3C Recommendation, 2006, http://www.w3.org/TR/REC-xml-names/.

[22] D. Beckett, “RDF/XML Syntax Specification (Revised),” W3C Recommendation, 2004, http://www.w3.org/TR/rdf-syntax-grammar/.

[23] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P.P.S. eds, The Description Logic Handbook - Theory, Implementation and Applications, Cambridge University Press, 2003.

[24] オントロジー工学研究委員会編，“大規模知識ベースに関する調査研究－オントロジー工学に関する調査研究－報告書,” 財団法人日本情報処理開発協会，1998.

[25] J. Bateman, “Upper Modeling: Organizing Knowledge for Natural Language Processing,” Proceedings of the Fifth International Workshop on Natural Language Generation, 1990.

[26] L. Carlson, and S. Nirenburg, “World Modeling for NLP,” TR CMU-CMT-90-121, 1990.

[27] M.Uschold, and M.Gruninger, “ONTOLOGIES : Principles, Methods and Applications,” AAAI96 TUTORIAL, pp.SA1-69 - SA1-137, 1997.

[28] 関内律恵子，小森聡，樗松理樹，青木千鶴，山口高平，“DODDLE：計算機可読型辞書を利用した領域オントロジー構築支援環境 (2) －概念階層構築支援－,” 第 55 回情報処理学会全国大会 3AF-7，1997.

[29] 古崎晃司，來村徳信，池田満，溝口理一郎，“「ロール」および「関係」に関する基礎的考察に基づくオントロジー記述環境の開発,” 人工知能学会論文誌，vol.17，no.3，pp.196–208，2002.

[30] W.Swartout, R.Patil, K.Knight, and T.Russ, “Toward Distributed Use of Large-Scale Ontologies,” Proceedings of the 10th Knowledge Acquisition Workshop (KAW’96), pp.32.1–32.19, 1996.

[31] R. Navigli, and P. Velardi, “Automatic Adaptation of WordNet to Domains,” Proceedings of the International Workshop on Ontologies and Lexical Knowledge Bases (OntoLex2002), 2002.

[32] G. van Heijst, The Role of Ontologies in Knowledge Engineering, Dr.thesis, University of Amsterdam, 1995.

[33] M. Uschold, M. King, S. Moralee, and Y. Zorgios, “The Enterprise Ontology,” Knowledge Engineering Review, vol.13, no.1, pp.31–89, 1998.

[34] T. Morita, N. Izumi, N. Fukuta, and T. Yamaguchi, “A Graphical RDF-based Meta-Model Management Tool,” IEICE Transactions on Information and Systems, Special Issue on Knowledge-Based Software Engineering, vol.E89-D, no.4, pp.1368–1377, 2006.

[35] D. Rager, J. Lerner, and T. Self, “vOWLidator,” , 2005, http://projects.semwebcentral.org/projects/vowlidator/.

[36] G. Alder, “JGraph,” , 2007, http://www.jgraph.com.

[37] HP Labs, “Jena Semantic Web Framework,” , 2007, http://jena.sourceforge.net/.

[38] M. Denny, “Ontology Tools Survey, Revisited,” , 2004, http://www.xml.com/pub/a/2004/07/14/onto.html.

[39] E. Pietriga, “IsaViz, a Visual Environment for Browsing and Authoring RDF Models,” Proceedings of the Eleventh International World Wide Web Conference Developers Day, 2002, http://www.w3.org/2001/11/IsaViz/.

[40] D. Steer, L. Miller, and D. Brickley, “RDFAuthor: Enabling everyone to author RDF,” Proceedings of the Eleventh International World Wide Web Conference Developers Day, 2002, http://rdfweb.org/people/damian/RDFAuthor/.

[41] A. Maedche, B. Motik, L. Stojanovic, R. Studer, and R. Volz, “An Infrastructure for Searching, Reusing and Evolving Distributed Ontologies,” Proceedings of the 12th International World Wide Web Conference (WWW’03), pp.439–448, 2003.

[42] RDF Site Summary 1.0 Specification Working Group, “RDF Site Summary (RSS) 1.0,” , 2001, http://web.resource.org/rss/1.0/spec.

[43] N.F. Noy, and D.L. McGuinness, “Ontology Development 101: A Guide to Creating Your First Ontology,” Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, 2001.

[44] J. Broekstra, A. Kampman, and F. van Harmelen, “Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema,” Proceedings of the First Internation Semantic Web Conference, no.2342, pp.54–68, Springer Verlag, 2002, http://www.openrdf.org/.

[45] T. Morita, N. Fukuta, N. Izumi, and T. Yamaguchi, “DODDLE-OWL: Interactive Domain Ontology Development with Open Source Software in Java,” IEICE Transactions on Information and Systems, Special Issue on Knowledge-Based Software Engineering, 2008, (to appear).

[46] P. Buitelaar, P. Cimiano, and B. Magnini, Ontology Learning from Text: Methods, Evaluation and Applications, vol.123 of Frontiers in Artificial Intelligence and Applications, IOS Press, 2005.

[47] 山口高平，樽松理樹，青木千鶴，関内律恵子，加賀山茂，吉野一，“計算機可読型辞書を利用した領域オントロジー構築支援環境,” 人工知能学会誌，vol.14，no.6，pp.1080–1087，1999.

[48] P. Buitelaar, “OntoSelect: Towards the Integration of an Ontology Library, Ontology Selection and Knowledge Markup,” Proceedings of the Workshop on Knowledge Markup and Semantic Annotation (Semannot2004), 2004, http://olp.dfki.de/ontoselect.

[49] L. Ding, R. Pan, T. Finin, A. Joshi, Y. Peng, and P. Kolari, “Finding and Ranking Knowledge on the Semantic Web,” Proceedings of the 4th International Semantic Web Conference, LNCS 3729, pp.156–170, 2005, http://swoogle.umbc.edu/.

[50] L. Page, S. Brin, R. Motwani, and T. Winograd, “The PageRank Citation Ranking: Bringing Order to the Web,” Technical report, Stanford Digital Library Technologies Project, 1998, http://citeseer.ist.psu.edu/page98pagerank.html.

[51] 小出誠二，森田武史，山口高平，ムリアディヘンドリー，武田英明，“WordNet と EDR の OWL 表現,” 人工知能学会セマンティック Web とオントロジー研究会 SIG-SWO-A601-03，2006.

[52] A. Miles, and D. Brickley, “SKOS Core Guide,” , 2005, http://www.w3.org/TR/swbp-skos-core-guide/.

[53] 中山浩太郎，原隆浩，西尾章治郎，“Wikipedia マイニングによるシソーラス辞書構築手法,” 情報処理学会論文誌，vol.47，no.10，pp.2917–2928，2006，http://wikipedia-lab.org/.

[54] 中川裕志，森辰則，湯本紘彰，“出現頻度と連接頻度に基づく専門用語抽出,” 自然言語処理，vol.10，no.1，pp.29–35，2003，http://gensen.dl.itc.u-tokyo.ac.jp/.

[55] M.A. Hearst, and H. Schütze, “Customizing a Lexicon to Better Suit a Computational Task,” Corpus Processing for Lexical Acquisition, pp.77–96, 1996.

[56] 元田浩，津本周作，山口高平，沼尾政行，データマイニングの基礎，オーム社，2006.

[57] R. Agrawal, and R. Srikant, “Fast Algorithms for Mining Association Rules,” Processing of the 20th International Conference Very Large Data Bases, VLDB, pp.487–499, Morgan Kaufmann, 1994.

[58] Y. Tsuruoka, and J. Tsujii, “Bidirectional Inference with the Easiest-First Strategy for Tagging Sequence Data,” Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, pp.467–474, Association for Computational Linguistics, 2005.

[59] A. Montoyo, R. Munoz, and E. Metais, “Text2Onto - A Framework for Ontology Learning and Data-driven Change Discovery,” Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems, vol.3513, pp.227-238, Springer Verlag, 2005.

[60] S. M., and A.A. Barforoush, “An Introduction to HASTI: An Ontology Learning System,” Proceedings of 6th Conference on Artificial Intelligence and Soft Computing, 2002.

[61] P. Buitelaar, D. Olejnik, and M. Sintek, "A Protégé Plug-In for Ontology Extraction from Text Based on Linguistic Analysis," Proceedings of the 1st European Semantic Web Symposium, pp.31–44, 2004.

[62] P. Spyns, and M.L. Reinberger, "Lexically Evaluating Ontology Triples Generated Automatically from Texts," Proceedings of the Second European Conference on the Semantic Web, vol.3532, pp.563–577, Springer Verlag, 2005.

[63] J. Volker, D. Vrandecic, and Y. Sure, "Automatic Evaluation of Ontologies," Proceedings of the 4th International Semantic Web Conference, pp.716–731, 2005.

[64] D. Lin, and P. Pantel, "DIRT - Discovery of Inference Rules from Text," Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp.323–328, 2001.

[65] M. Shamsfard, and A.A. Barforoush, "The State of the Art in Ontology Learning: a Framework for Comparison," Knowl. Eng. Rev., vol.18, no.4, pp.293–316, 2003.

[66] S. Koide, M. Kawamura, T. Morita, T. Yamaguchi, and H. Takeda, "Semantic Search: An Implementation, Deployments, and Lessons Learned," Proceedings of the 1st Asian Semantic Web Conference Workshop on Web Search Technology, 2006.

[67] J.D.M. Rennie, "Derivation of the F-Measure," , 2004, http://people.csail.mit.edu/jrennie/writing.

[68] K. Sono, and M. Yamate, United Nations Convention on Contracts for the International Sale of Goods, Seirin Shoin, 1993, http://cisgw3.law.pace.edu/.

[69] F.M. Suchanek, G. Kasneci, and G. Weikum, "Yago: A Core of Semantic Knowledge," Proceedings of the 16th International Conference on World Wide Web, pp.697–706, ACM, 2007.

[70] S. Auer, C. Bizer, J. Lehmann, G. Kobilarov, R. Cyganiak, and Z. Ives, "DBpedia: A Nucleus for a Web of Open Data," Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference, vol.4825, pp.715–728, Springer Verlag, 2007.

学位論文に関連する論文および口頭発表

3章 RDF(S) コンテンツ構築支援ツールに関連する論文

1. Takeshi Morita, Noriaki Izumi, Naoki Fukuta, Takahira Yamaguchi, “A Graphical RDF-based Meta-Model Management Tool,” IEICE Transactions on Information and Systems, Special Issue on Knowledge-Based Software Engineering, Vol.E89-D, No.4, pp.1368-1377, 2006

2. Takeshi Morita, Noriaki Izumi, Naoki Fukuta, Takahira Yamaguchi, “ MR^3 : Meta-Model Management Based on RDFs Revision Reflection,” Proceedings of the Joint Conference on Knowledge Based Software Engineering, Frontiers in Artificial Intelligence and Applications, Vol.108, pp.228-236, 2004

3. Takeshi Morita, Noriaki Izumi, Naoki Fukuta, Takahira Yamaguchi, “Meta-Model Management Environment for RDF Contents,” Proceedings of the 2nd International Conference on Global Research and Education, Vol.2, pp.307-314, 2003

3章 RDF(S) コンテンツ構築支援ツールに関連する口頭発表

1. Takeshi Morita, Noriaki Izumi, Naoki Fukuta, Takahira Yamaguchi, “Building up RDF and RDFS Descriptions with Meta-Model Management,” Proceedings of the 4th International Semantic Web Conference, Poster, pp.28, 2005

2. 森田 武史, 和泉 憲明, 福田 直樹, 山口 高平, “セマンティック Web のためのメタモデル管理ツール,” 第 17 回 人工知能学会全国大会論文集, 1G2-05, 2003

3. Noriaki Izumi, Takeshi Morita, Naoki Fukuta, Takahira Yamaguchi, “RDF-based Meta-Model Management Environment,” Proceedings of the 6th SANKEN (I.S.I.R) International Symposium on New Trends in Knowledge Processing, Data mining, Seamntic Web and Computational Science, pp.87-88, 2003

4章 領域オントロジー構築支援環境に関連する論文

1. Takeshi Morita, Naoki Fukuta, Noriaki Izumi, Takahira Yamaguchi, “DODDLE-OWL: Interactive Domain Ontology Development with Open Source Software in Java,” IEICE Transactions on Information and Systems, Special Issue on Knowledge-Based Software Engineering, 2008 (to appear)

2. Takeshi Morita, Naoki Fukuta, Noriaki Izumi, Takahira Yamaguchi, “DODDLE-OWL: A Domain Ontology Construction Tool with OWL,” Proceedings of the 1st Asian Semantic Web Conference, Lecture Notes in Computer Science, Vol.4185, pp.537-551, 2006

3. Takeshi Morita, Yoshihiro Shigeta, Naoki Sugiura, Naoki Fukuta, Noriaki Izumi, Takahira Yamaguchi, “DODDLE-OWL: OWL-based Semi-Automatic Ontology Development Environment,” Proceedings of the 3rd International Workshop on Evaluation of Ontology-based Tools, 2004

4. Takeshi Morita, Yoshihiro Shigeta, Naoki Sugiura, Naoki Fukuta, Noriaki Izumi, Takahira Yamaguchi, “Domain Ontology Constrution with Quality Refinement,” Proceedings of the Pacific Rim Knowledge Acquisition Workshop 2004, pp.116-126, 2004

4章 領域オントロジー構築支援環境に関連する口頭発表

1. Takeshi Morita, Takuya Tejima, Noriaki Izumi, Takahira Yamaguchi, “A Domain Ontology Construction Method Supported by an Ontology Search Engine,” Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference, Poster & Demonstration, pp.73-74, 2007

2. 森田 武史, 和泉 憲明, 山口 高平, “オントロジー検索エンジンと領域オントロジー構築支援環境 DODDLE-OWL の統合,” 第 21 回 人工知能学会全国大会論文集, 1D2-4, 2007

3. 森田 武史, 和泉 憲明, 山口 高平, “オントロジー検索エンジンを用いた領域オントロジー構築支援環境 DODDLE-OWL の拡張,” 人工知能学会 セマンティック Web とオントロジー研究会 第 15 回, SIG-SWO-A603-07, 2007

4. 森田 武史, 手島 拓也, 和泉 憲明, 山口 高平, “既存情報資源を活用したオントロジー構築支援環境,” 電子情報通信学会, 信学技報（AI, KBSE）, KBSE2006-73, Vol.106, No.618, pp.21-24, 2007

5. 森田 武史, 小野 穣, 洪 潤基, 川村 正則, 小出 誠二, 山口 高平, “領域オントロジー構築支援環境を用いたオントロジー搭載型検索システムの提案と評価,” 情報システム学会 第 2 回研究発表大会, 2006

6. 森田 武史, 山口 高平, “日本語概念を対象にした領域オントロジー構築支援環境 DODDLE の機能拡張,” 電子情報通信学会, 信学技報（知能ソフトウェア工学研究会）, KBSE2006-1, Vol.106, No.36, pp.1-6, 2006

7. 森田 武史, 山口 高平, “オントロジーエディタ DODDLE ファミリーの開発と利用,” 人工知能学会 セマンティック Web とオントロジー研究会 第 11 回, SIG-SWO-A502-07, 2005

8. Takeshi Morita, Yoshihiro Shigeta, Naoki Sugiura, Naoki Fukuta, Noriaki Izumi, Takahira Yamaguchi, “DODDLE-OWL: On-the-fly Ontology Construction with Ontology Quality Management,” Proceedings of the 3rd International Semantic Web Conference, Poster, pp.47, 2004

9. 森田 武史, 繁田 佳宏, 杉浦 直樹, 福田 直樹, 和泉 憲明, 山口 高平, “ セマンティック Web のためのオントロジー構築支援環境 DODDLE-R の実装,” 第 18 回 人工知能学会全国大会論文集, 1G3-01, 2004

謝辞

本論文を執筆するにあたり，多くの方々から多大なるご指導およびご助言を賜りました．

本研究を行う契機と環境を与えて下さり，研究の全過程を通じて，常に温かく適切な御指導を頂いた，山口高平 先生に心から感謝いたします．

本論文の副査を快諾していただき，本論文の執筆にあたり，有益な御助言および御指導を頂いた，櫻井彰人 先生，大門樹 先生，高田眞吾 先生に厚く御礼申し上げます．

本研究を進めるにあたって，適切な御指導および御助言を頂いた，産業技術総合研究所 和泉憲明 先生，静岡大学情報学部 福田直樹 先生に深く感謝いたします．

（株）ジャストシステム 上田俊夫 氏には，RDF(S) コンテンツ構築支援ツール開発の初期の段階から，ツールを使用していただき，多くの有益なコメントをいただきました．ここに感謝いたします．

国立情報学研究所 小出誠二 氏,（株）IHI エスキューブ 川村正則 氏,（株）IHI 塚本信行 氏，山口研究室 小野穣 氏，洪潤基 氏には，領域オントロジー構築支援環境について多くの有益なコメントをいただくと共に，大規模オントロジー構築実験およびその評価に協力していただきました．ここに感謝いたします．

山口研究室 近藤恵一 氏には，RDF(S) コンテンツ構築支援ツールおよび領域オントロジー構築支援環境を使用していただき，多くの有益なコメントをいただきました．また，両ツールの評価実験に協力していただきました．ここに感謝いたします．

研究活動全般に渡って，御援助下さった山口研究室の学生諸氏に感謝いたします．

最後に，快く大学院への進学を許可してくれたことをはじめ，博士課程在学中において経済的，精神的に支えてくれた両親へ感謝の言葉を送りたいと思います．

2008 年 2 月

森田 武史