2005－DPS－123（18）
2005／6／3

健康領域上のオントロジを用いた推論システムの開発

湯本 純也 † 富樫 敦 ‡

† 宮城大学 事業構想学研究科 ‡ 宮城大学 事業構想学部
E-mail: {p0552016, togashi}@myu.ac.jp

概要:
「健康福祉のための先進的エージェント・ネットワークに関する研究」を推進している．プロジェクトの目標は，市民から発せられる多岐にわたる要求や質問に対して，その背景と意味を十分に理解し適切に応答できる高度な健康福祉サービスを実現することである．この目的を達成するためには，健康に関するオントロジを構築し，その上の推論機構を確立する必要がある．
本論文では，オントロジ記述言語 OWL で表現されたオントロジと，OWL と Description Logic レベルのルールから生まれたルール記述言語 SWRL を用いた推論システムを提案する．オントロジの読み込みには Jena を，ルール推論エンジンとして jDREW を，オントロジ構築に Protégé など既存のツールを利用した．
キーワード: 推論システム，オントロジ，健康福祉領域, セマンティックウェブ

Inference System based on Health Welfare Ontology

Junya YUMOTO† Atsushi TOGASHI‡

† Graduate School of Project Desing, Miyagi University
‡ School of Project Design, Miyagi University
E-mail: {p0552016, togashi}@myu.ac.jp

Abstract: We are promoting the research project ”research on the advanced agent network for health welfare”. One of the main targets of the project is to achieve the intelligent social welfare services which fully understand the meaning of queries and answer them properly. For this end, it is necessary to build ontology on health welfare and to establish reasoning over it.
This paper introduces an inference system based on ontology. The ontology is written by OWL (Web Ontology Language) and background knowledge is written by SWRL (Semantic Web Rule Language) as rules over ontology. The resulting system consists of the tools, Jena for loading necessary ontology, jDREW as a rule inference engine, and Proté gé as an editor. Several examples illustrate the usefulness of the system.
Keyword: Inference System, Ontology, Health Welfare Domain, Semantic Web

1. はじめに

個人のプライバシーとセキュリティ上の安全性を十分に確保したネットワークを構築する「健康福祉のための先進的エージェント・ネットワークに関する研究」(以下「健康福祉プロジェクト」) が総務省により採択された．この上で，市民から発せられる多岐にわたる要求や質問に対して，その背景と意味を十分に理解し適切に応答できる高度な健康福祉サービスを実現するため，先進的・ネットワーク技術を確立する．この目的を達成するため，(1) 高次セキュリティ機構，(2) エージェント間コミュニケーション機構，(3) 高次指紋認証機構付きモバイル GUI 端末，(4) 先進的エージェント技術，に関する基幹技術の研究開発を行う．

健康福祉サービスを提供するシステムの全体像は図 1 の通りである．ユーザが発したクエリに対し，ユーザエージェントはオントロジに格納されているユーザの情報とともにオントロジエージェントにクエリを渡す．オントロジエージェントはメタオントロジを用いてクエリを分割し，それぞれ分割されたクエリを解くにふさわしい，ドメインエージェントに渡す．

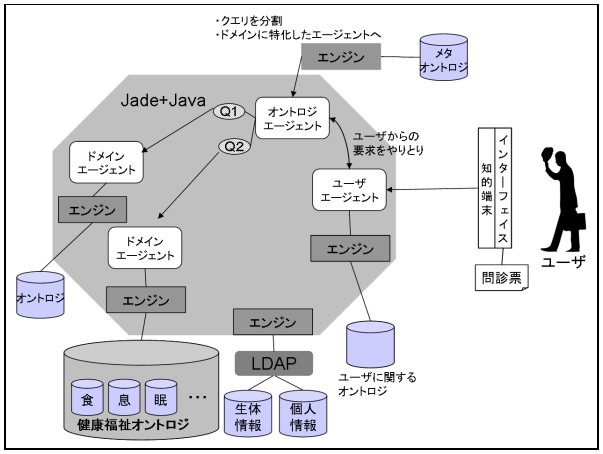

図 1　健康福祉プロジェクトの全体像

本論文ではオントロジを用いた推論システムに焦

点を当て，その推論機構と推論エンジンの設計開発を行う．

2. Semantic Web と知識

2.1 Semantic Web

2.1.1 ウェブのセマンティクス

コンピュータがウェブ上でデータを交換するのに適した文書記述言語として，まず XML（Extensible Markup Language）が 1998 年に W3C（World Wide Web Consortium）から勧告された．そして，リンクの関係を表現するモデルとして，1999 年に RDF[4]（Resource Description Framework）が勧告された．RDF では，ノードは文書だけに限らず，人，プロジェクト，コンセプトなどあらゆるリソースを表す．すなわち, RDF をもちいてウェブは人間が読む文書ネットワークから，意味ネットワークに近いものに進んでいく．リソースを URI で名前付けし，その関係を RDF のモデルで表現し，XML の基礎の上でコンピュータが自動的に処理できるようにする．こうして，セマンティクスをもつウェブ「Semantic Web[3](以下「SW」)」がスタートすることになる．

2.1.2 基本 5 原則

SW は 5 つの基本的な考え方に基づいて設計されている．

URI 識別　すべてのものが URI によってグローバルに識別される．

部分的情報　SW は部分的な情報しか知り得ない実世界を対象とし，そこから有益な結論を引き出すことを目指している．不完全な情報，断片的な情報の存在を前提とし，それを生かすことの重要性を示している．

発展性　分散型のウェブでは，様々なコミュニティや個人が，独自に情報を発信したり，知識の体系を整えたりする．情報を統合することや，新しい知識を加えるときに，古いものを犠牲にしなくても上手く統合できること，別々のコミュニティが同一性や矛盾をきちんとチェックできることは，分散型の世界が自立的に発展していくために欠かせない要件である．

最小デザイン　できるだけ制約を課さず，必要以上の標準化を求めない，複雑なものは細かくモジュール化する，という方針で使用を設計する．こうすることで，個別仕様の実装が容易になり，またそれに基づいた応用を柔軟に考えることが可能になる．

信頼のウェブ　ウェブには多種多様な情報が散在し，その信頼度もまた様々である．SW では，存在する情報全てが信頼できると保証するのではなく，アプリケーションが文脈から信頼度を評価する仕組みを考えていく．

2.1.3 技術階層

SW 実現のためのアーキテクチャを，いくつかの技術の組み合わせとして表 1 のような階層構造を示し，一般的に 9 階層の構造からなっている．現在は，Ontology 層までは標準化が進み，今後 Rules 層以降が標準化の対象となっている．

表 1　階層構造と各層における目的

レイヤー		役割
Trust		文脈や Proof，暗号化と電子署名により，エージェントが示した結果の信頼性を判断
Proof	暗号化 電子署名	エージェントの処理の履歴，処理理由など，結果を導いた根拠を示す
Logic		一階述語論理などを用いた知識の記述と，それに基づくエージェントの処理
Rules		問い合わせ，フィルタリングを可能にする共通基盤としての論理の定義
Ontology		より精密な語彙の定義と，複数のスキーマの関係づけ・融合を可能にする推論
RDF Schema		語彙 (クラス，プロパティ) を定義する手段の提供
RDF MS		機械処理可能なメタデータの表現 (データモデル)
XML/Namespace		処理が容易な記述言語（XML）と複数語彙の区別・混在を可能にするメカニズム (名前空間)
URI/Unicode		リソースのグローバルな識別 (URI) とグローバルなデータ表現 (Unicode)

今回焦点を当てるエンジンは，SW における Rules 層・Logic 層を考慮して設計開発を行う．

2.2 知識とルール

2.2.1 RDF Model and Syntax

ウェブの目標である「マシンに理解可能な情報」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 は，主語（リソース）と述語（プロパティ），そしてその目的語（オブジェクト，プロパティの値）の三者関係によって，関係の連鎖を辿ることができるようなデータモデルを記述する．1999 年 2 月に W3C によって正式な勧告となった．

適切に記述された HTML 文書は，その文法を理解しているユーザエージェント（ブラウザ）によって，見出しを抽出したり用語集を作ったり，あるいは検索のデータベースに収録したりといった自動処理は可能である．しかし，これはエージェントが HTML という特定の語彙を知っていることが前提であり，ここで定義されていないこと（リソース）について，意味を推論する手段はない．RDF は，特定のアプリケーションや知識領域を前提とせずに，相互運用可能な形で「リソースを記述する」ための標準的なメカニズムを提供する．

RDF は，リソースの関係を主語・述語・目的語という 3 つの要素（トリプル）で表現する．トリプルの集合は RDF のグラフと呼ばれ，トリプルは「主語・目的語間の関係のステートメント」を表す．また，RDF ではこれらの関係を有向ラベル付きグラフで表現する．一般的には，リソースを円，プロパティを矢印，リテラルを四角で表現する．図 2 に示した例は，リソース「http://somewhere」とリテラル「someone」はプロパティ「dc:creator」の関係で結ばれていて，「リソー

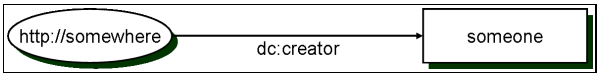

図 2　RDF モデル例

```
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xml:lang="ja">
 <rdf:Description rdf:about="http://somewhere">
  <dc:creator>someone</dc:creator>
 </rdf:Description>
</rdf:RDF>
```

図 3　RDF の XML 表現

ス http://somewhere の作者（dc:creator）は someone である」ことを意味している．

図 2 に示したモデルを，XML 形式で表現すると図 3 のようになる．

2.2.2　RDF Schema

RDF はリソースのプロパティやリソース間の関係を記述するモデルを提供するが，そこで使われるプロパティそのものについては直接定義しない．creator, title, publisher といったプロパティ（リソース記述の語彙）は，RDF のスキーマ（Schema）を用いて別途定義し，それを URI で参照する．

RDF スキーマは，これらのプロパティや一般的なリソースのカテゴリを定義するための基本的なメカニズムを提供する．仕様で用意される基本クラス，基本プロパティなどを用いて，必要なクラスを定義し，そのインスタンスやサブクラスを導いていく方法は，オブジェクト指向言語になぞらえて理解することもできる．

RDF スキーマは, 語彙を定義するもっとも基本的な仕組みで，知識を表現するためのより詳細な語彙や関係性の定義のためには，このスキーマのツールをもとに，オントロジ言語などを設計し，利用していくことになる．RDF スキーマで用いられる, 基本クラスと基本プロパティを表 2 に示す．名前空間接頭辞「rdfs:」は「http://www.w3.org/2000/01/rdf-schema#」を，同じく「rdf:」は「http://www.w3.org/1999/02/22-rdf-syntax-ns#」を表している．

またプロパティを定義する場合,「domain（定義域）」と「range（値域）」が必要になってくる．この制約を理解するために簡単な例を挙げてみる．ここでは「食べる（eat）」というプロパティが定義されていて,「食べる」プロパティを用いて新たなトリプルを定義する．一般的には「食べる」の主語は「動物」で目的語には「食べ物，食べられる物」が来るはずだ．しかしマシンは，そのような背景に存在する知識を理解する術は持ち合わせていないので，主語を「食器」，目的語を「建築物」などのような，好ましくない（事実上考えられない）トリプルを生成してしまう可能性がある．このような事態を避けるために，ドメインとして主語の，レンジとして目的語の取り得るクラスを定義する．

表 2　RDF スキーマの基本クラスと基本プロパティ

Class	概要
rdfs:Resource	リソース一般を表す基本クラス
rdfs:Class	クラスという概念を示すクラス
rdfs:Literal	文字列などのリテラル値を示すクラス
rdfs:Datatype	データ型を定義するクラス
rdf:XMLLiteral	XML としてマーク付けしたリテラルを示すクラス
rdf:Property	プロパティという概念を示すクラス

Property	概要	domain	range
rdfs:range	プロパティの目的語のタイプ	Property	Class
rdfs:domain	プロパティの主語のタイプ	Property	Class
rdf:type	どんなクラスに属するかを示す	Resource	Class
rdfs:subClassOf	サブクラスであることを示す	Class	Class
rdfs:subPropertyOf	プロパティの精密化であることを示す	Property	Property
rdfs:label	人間が読みやすい形のリソース名	Resource	Literal
rdfs:comment	人間のための説明	Resource	Literal

2.2.3　Ontology(OWL)

RDF，RDF スキーマによるリソースの叙述という基本ツールを使って，ウェブに存在するものごとの分類体系やその関係，さらにはそれを推論していくためのルールを定義するオントロジ言語 Web Ontology Language OWL[8] が 2004 年 2 月に W3C 勧告となった．SW では，各地で独自に定義される語彙を関連づけ，相互運用できるようにするためにオントロジが重要になる．オントロジで表現された知識を利用し，エージェントが高度な検索などを行うことが期待されている．

オントロジは，アリストテレス以来の「存在論」を指す哲学用語として知られているが，その意味が転じて（拡張されて），対象とする世界に存在するものごとを体系的に分類し，その関係を記述するものとして，言語学や人工知能研究に取り込まれてきた．ウェブでのオントロジとは，ティム・バーナーズ・リーらの言葉を借りれば,「分類体系」と「推論ルール集」である

RDF スキーマは基本的なクラスとプロパティ定義の手段を提供している．しかし，様々な領域でコンピュータ（エージェント）が自動的に相互動作・運用できるようにするには，より詳細で厳密な意味と関係の記述が必要になる．特にウェブにおいては，あちこちで独自に語彙や知識ベースが構築されうる．そこで，これらの相互の関係を示し，統合や相互利用ができるような仕組みが重要となってくる．オントロジは，SW をグローバルで誰もが利用可能なものにするための要となる部分である．

ウェブ・オントロジー言語に求められる共有性，発展性，相互運用性，矛盾の検出といった要件を満たす

```
<owl:Ontology rdf:about="">
 <owl:versionInfo>webont.html, v.0.9; 2002-08-25 Exp</owl:versionInfo>
 <owl:imports rdf:resource="http://www.w3.org/2002/07/owl"/>
 <dc:creator>someone</dc:creator>
</owl:Ontology>
```

図 4　OWL ヘッダ要素

```
<owl:Class rdf:ID="Male">
 <rdfs:subClassOf rdf:resource="#Animal"/>
 <owl:disjointWith rdf:resource="#Female"/>
</owl:Class>
```

図 5　OWL クラス要素

ものとして，W3C のワーキンググループで開発されているのが，OWL という言語である．これは「用語・語彙とそこに含まれる各要素の関連の明確な表現」を目的としており，これまでに開発されてきたオントロジ言語の改良版となっている．

フル規格の OWL には，記述論理をベースにタイプの区別を厳密にして決定可能性を確保する OWL DL と，クラスを個体とみなすことができるなど実用的なオントロジ構築を念頭においた OWL Full がある．また，より実装しやすい OWL DL のサブセットである OWL Lite も合わせて提供される．

OWL のオントロジは，RDF のトリプルの集合で構成される．OWL 言語仕様では，どんな RDF トリプルが OWL の語彙を構成し，それによって何を意味するかを定義する．OWL は一般に RDF の XML 構文によって記述され，以下の 4 つの構成要素からなる．

1. バージョン情報と他のオントロジのインポートを記述するヘッダ
2. クラスを定義するクラス公理
3. プロパティを定義するプロパティ公理
4. 個体 (Individual)：クラスのインスタンスによる事実の記述

ヘッダは「owl:Ontology」要素として記述し，バージョン情報と他のオントロジのインポートを示すことができる．図 4 では OWL ファイルの版，「http://www.w3.org/2002/07/owl」のインポート，OWL ファイルの作者を定義している．

「ウェブに存在するもの」の概念であるクラスは「rdfs:Class」のサブクラスである「owl:Class」要素によって表現し，次の要素でクラス公理を構成する．RDF のトリプルにあてはめれば，定義されるクラスが主語，要素名 URI が述語，参照クラスが目的語となる．例えば，「雄」というクラスは「動物」のサブクラスで「雌」とは互いに疎の関係にある，という定義をする場合は，図 5 のように記述できる．

プロパティの制約は，「owl:Restriction」要素を用い，その中に対象となるプロパティを示す「owl:onProperty」要素と制約要素を 1 つだけ記述する．「owl:Restriction」は「owl:Class」のサブクラスで，この制約は匿名のクラスを定義する．「Person」というクラスは「Animal」のサブクラスで，「RealName」というプロパティの値は 1 つだけである，と定義するなら図 6 のようになる．

```
<owl:Class rdf:ID="Person">
 <rdfs:subClassOf rdf:resource="#Animal"/>
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#hasRealName"/>
   <owl:cardinality>1</owl:cardinality>
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

図 6　OWL プロパティ要素

```
<ex:Novelist rdf:ID="Lewis_Carroll">
 <owl:sameAs rdf:resource="#Charles_Lutwidge_Dodgson"/>
 <ex:isAuthorOf rdf:resource="#Alice_in_Wonderland"/>
</ex:Novelist>
```

図 7　OWL での個体定義

OWL では，インスタンスすなわち「個体 (Individual)」は，必ず何かのクラスに属する．通常はそのクラス名による型付ノード要素の内容に，「owl:sameAs」などのプロパティを記述して表現する．ウェブ上では，異なる URI が同じリソースを表すことは珍しくない．異なる名前が異なる個体を指すという前提を持てないシステムでは，同一性や違いを明示することは論理的な推論のために重要になってくる．個体を記述する例を図 7 に示す．

2.2.4　Rules

推論規則を XML 形式で表現する言語の 1 つに，Rule Markup Language RuleML[9] がある．この RuleML をベースとして，OWL DL を拡張する形で推論規則を記述できるように手を加え，W3C によって提案されているのが Semantic Web Rule Language SWRL[10] である．さっそく，SWRL の例を見ながら，その構文について触れてみたい．その際，名前空間接頭辞「swrlx:」は「http://www.w3.org/2003/11/swrl#」の，「ruleml:」は「http://www.w3.org/2003/11/ruleml#」の名前空間 URI を示すものとする．

SWRL では，含意規則全体を「imp」要素のノードとし，前件を「body」，後件を「head」というプロパティで表す．これらは，RuleML の要素を引き継いでいるので，「ruleml:」名前空間の語彙として表現されている．それぞれのアトムの記述は，SWRL の語彙を使う．クラス表現，オブジェクト値型プロパティ，データ値型プロパティといった OWL の要素に応じて，「ClassAtom」，「IndividualPropertyAtom」，「DatavaluedPropertyAtom」などのノードで示され，述語が「propertyPredicate」，引数が「argument1」，「argument2」などのプロパティで表される．また変

```
<swrl:Variable rdf:ID="x"/>
<swrl:Variable rdf:ID="y"/>
<swrl:Variable rdf:ID="z"/>
<ruleml:Imp>
  <ruleml:body rdf:parseType="Collection">
    <swrl:IndividualPropertyAtom>
      <swrl:propertyPredicate rdf:resource="&eg;hasParent"/>
      <swrl:argument1 rdf:resource="#x" />
      <swrl:argument2 rdf:resource="#y" />
    </swrl:IndividualPropertyAtom>
    <swrl:IndividualPropertyAtom>
      <swrl:propertyPredicate rdf:resource="&eg;hasBrother"/>
      <swrl:argument1 rdf:resource="#y" />
      <swrl:argument2 rdf:resource="#z" />
    </swrl:IndividualPropertyAtom>
  </ruleml:body>
  <ruleml:head rdf:parseType="Collection">
    <swrl:IndividualPropertyAtom>
      <swrl:propertyPredicate rdf:resource="&eg;hasUncle"/>
      <swrl:argument1 rdf:resource="#x" />
      <swrl:argument2 rdf:resource="#z" />
    </swrl:IndividualPropertyAtom>
  </ruleml:head>
</ruleml:Imp>
```

図 8　SWRL の記述例

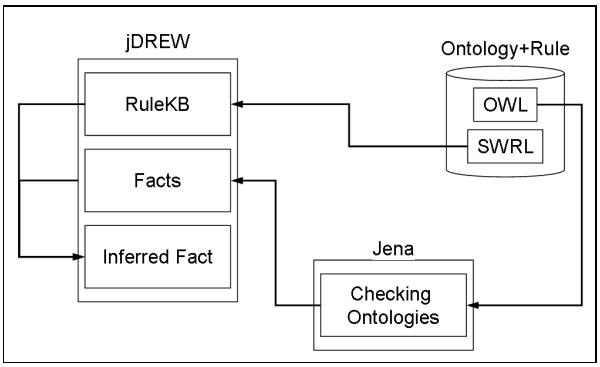

図 9　推論システムの概要

数は「swrl:Variable」クラスのインスタンスとして記述する方法が提案されている．この方法は RDF のモデルを逸脱する形になってしまうが，OWL などの言語との親和性が重要視されているようだ．

最後に，図 8 に示した例が何を表しているのか見てみる．まず，1〜3 行で「x」「y」「z」なる変数を定義している．これらの変数を用いて以下の規則を定義している．

$$hasUncle(?x, ?y) \leftarrow$$
$$hasParent(?y, ?z) \wedge hasBrother(?x, ?z)$$

この式は，x が y を親として持ち，y が z を兄弟として持つなら，x は z を叔父として持つ，という内容を表している．

3.　システムの実装

3.1　システムの構成

3.1.1　全体像

実装したシステムの全体像を図 9 に示した．推論プロセスについて詳しくて見ていくことにする．

まず，構築済みのオントロジを SW のフレームワークである Jena[11] によって解析する．Jena で提供されている機能を用いて，読み込んだオントロジに衝突・矛盾が存在しないかを確認するとともに，明示されていない関係を補足する．オントロジの点検が正常に終了した場合，OWL ファイルに記述されている情報を推論システムである jDREW[12] の Fact として扱える状態に変換し，読み込ませる．

次に，SWRL ファイルに記述されているルールを jDREW の RuleKB に読み込ませる．この時にルールに記述されている，クラス・プロパティ等が OWL ファイルで定義されているものか点検し，正常に終了した場合，SWRL で記述されたルールを jDREW で扱える形に変換しながら RuleKB に保存する．

最後に，jDREW が内部的に保持しているルール (RuleKB) とファクト (Facts) を用い jDREW の機能を利用し推論を行い，結果をエージェントに提供する．

```
X@prefix swrlbImport:  <http://www.daml.org/rules/proposal/swrlb.owl#> .
X@prefix swrl:    <http://www.w3.org/2003/11/swrl#> .
X@prefix swrlb:   <http://www.w3.org/2003/11/swrlb#> .
X@prefix swrlImport:  <http://www.daml.org/rules/proposal/swrl.owl#> .
X@prefix rdfs:    <http://www.w3.org/2000/01/rdf-schema#> .
X@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
X@prefix :        <http://www.myu.ac.jp/~p0122099/inference#> .
X@prefix owl:     <http://www.w3.org/2002/07/owl#> .
'owl:inverseOf'(':hasChild',':hasParent').
':hasParent'(':M01',':M02').
'owl:onProperty'('_:b1',':hasGender').
'a'('_:b1','owl:Restriction').
'a'(':Woman','owl:Class').
':hasChild'(':M02',':M01').
'rdfs:domain'(':hasUncle',':Human').
'a'(':M03',':Man').
'a'(':M02',':Man').
'rdfs:range'(':hasBrother',':Man').
'rdfs:subClassOf'(':Woman',':Human').
'owl:unionOf'('_:b2','_:b3').
'rdf:first'('_:b4',':Male').
```

図 10　OWL の Prolog 形式表記

3.1.2　N-Triple を利用した変換

jDREW は RuleML 形式のルールを扱うことができるが，Fact は Prolog 形式で与えてやる必要がある．Prolog の基本的な構文は P(S,O) からなっていて，それぞれ，P は述語，S は主語，O は目的語を表している．Prolog の基本的構文を形成している 3 要素は RDF や OWL のトリプルと一致する．相違点は，主語・述語・目的語をどのように表現しているかだけである．そこで，N-Triple 形式で出力させた OWL ファイルを Prolog 形式に変換し jDREW の「Facts」に読み込ませる．図 10 に Prolog 形式で表現した OWL の一部を示した．また，ルールに関しても，Prolog 形式に変換し，jDREW で扱っている．

3.2　動作検証

構築したシステムの動作検証を行うに当たり，Protégé[13] を用いオントロジを構築し，SWRL 形式でルールを記述した．

図 11 に構築したオントロジのクラス図を，定義した個体とそれぞれの関係を図 12 に示した．また，ルールは図 8 で示したルールを用いた．これらの定義から，個体「M01」と個体「M03」の間に，プロパティ

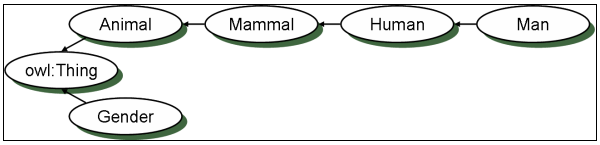

図 11　構築したオントロジのクラス図

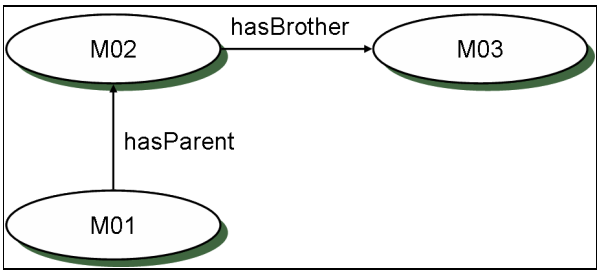

図 12　定義した個体と関係

```
'rdfs:subPropertyOf'('rdfs:isDefinedBy','rdfs:seeAlso').
'rdfs:subPropertyOf'('rdfs:isDefinedBy','rdfs:isDefinedBy').
'rdfs:subPropertyOf'('rdf:rest','rdf:rest').
'rdfs:subPropertyOf'('rdf:first','rdf:first').
':hasChild'(':M02',':M01').
':hasUncle'(':M01',':M03').
'owl:equivalentClass'(':Gender','_:b7').
'owl:equivalentClass'(':Human','_:b3').
'owl:equivalentClass'(':Woman','_:b4').
'owl:equivalentClass'(':Man','_:b6').
a('rdfs:subClassOf','rdf:Property').
a('rdf:type','rdf:Property').
a('_:b8','rdf:List').
a(':Mammal','owl:Class').
a(':Animal','owl:Class').
a('_:b7','owl:Class').
a(':Male',':Gender').
```

図 13　推論結果の一部

「hasUncle」が追加されなければならない．推論結果の一部を図 13 に示す．図 13 の四角で囲まれた部分に注目すると「hasUncle(M01, M03).」が期待通り追加されている．

4.　まとめ

オントロジ記述言語 OWL で表現されたオントロジと，OWL と Description Logic レベルのルールから生まれたルール記述言語 SWRL を用いた推論システムを提案した．オントロジの読み込みには Jena を，ルール推論エンジンとして jDREW を，オントロジ構築に Protégé など既存のツールを利用した．

しかし今日では，オントロジはより大きなものになり，巨大なオントロジを効率良く扱う推論エンジンの必要性が高まっている．今後の課題として，効率的且つ効果的な推論機構の開発が必須である．また，個々人の「健康」の概念は当然異なってくる．その差を埋めるようなオントロジの利用法も健康福祉プロジェクトにおける推論システムには必要になってくる．

謝辞

本研究は総務省「戦略的情報通信研究開発推進制度」において，「健康福祉のための先進的エージェント・ネットワークに関する研究」が採択されその一部として実施したものである．研究の機会を与えて頂いたことに感謝する．

参考文献

[1] 健康福祉のための先進的エージェント・ネットワークに関する研究, http://www.myu.ac.jp/~togashi/scope.
[2] Tim Berners-Lee,Information Management : A Proposal, http://www.w3.org/History/1989/proposal.html, 1989.
[3] Semantic Web, World Wide Web Consortium, http://www.w3.org/2001/sw/.
[4] Resource Description Framework RDF, World Wide Web Consortium, http://www.w3.org/RDF/.
[5] Dublin Core, Dublin Core Metadata Initiative, http://dublincore.org/.
[6] vCard, Internet Mail Consortium , http://www.imc.org/pdi/.
[7] Representing vCard Objects in RDF/XML, World Wide Web Consortium, http://www.w3.org/TR/vcard-rdf/, 2001.
[8] Web Ontology Language OWL, World Wide Web Consortium, http://www.w3.org/TR/owl-features.
[9] The Rule Markup Initiative, Harold Boley, Said Tabet, http://www.ruleml.org/.
[10] A Semantic Web Rule Language Combining OWL and RuleML, World Wide Web Consortium, http://www.w3.org/Submission/SWRL/.
[11] Jena Semantic Web Framework, Hewlett-Packard Development Company, http://jena.sourceforge.net/.
[12] jDREW Java Deductive Reasoning Engine for the Web, http://www.jdrew.org/jDREWebsite/jDREW.html.
[13] The Protégé Ontology Editor and Knowledge Acquisition System, http://protege.stanford.edu/.
[14] The Web KANZAKI, Masahide Kanzaki, http://kanzaki.com/.