

Title	ユーザ用件に基づく情報統合環境に関する研究
Author(s)	林, 正治
Citation	
Issue Date	2009-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/8011
Rights	
Description	Supervisor: 吉田武稔, 知識科学研究科, 博士

博士論文

ユーザ要件に基づく情報統合環境に関する研究

指導教官 吉田 武稔 教授

北陸先端科学技術大学院大学
知識科学研究科 システム知識領域専攻
林 正治

2009 年 3 月

Copyright © 2009 by Masaharu Hayashi

要旨

膨大なデータを有効活用する目的でつくられるセマンティック Web は RDF コンテンツにより構成され、問い合わせ言語を利用することでその中のデータを効率的に取り出すことができる。ところで、問い合わせ文を記述するには、問い合わせ対象の RDF コンテンツに含まれる語彙についての知識が必要である。標準化された語彙であれば、その語彙の意味関係は自明であるため、問い合わせ文を記述するのは容易い。しかしながら、多くの語彙は独自に定義されるため、RDF コンテンツを熟知していなければ、問い合わせ文を作成することは困難である。

本研究では、セマンティック Web にビューという概念を導入することにより、この課題を解決した。まず、RDF コンテンツへの問い合わせ方法をビューとして定義し、そのビューを共有可能にする。そして、そのビューを介してセマンティック Web データへのアクセス手段を提供する。これにより、RDF コンテンツを熟知していなくとも、セマンティック Web のデータの利用が可能となる。

この提案した技術の有用性を実証するために、本研究では、認知症早期診断法開発現場のファイル管理支援システムの構築にこの技術を適用した。

まず、認知症早期診断法開発現場の情報システム環境に保存されたファイルに含まれるデータを有効活用するために、ファイルから RDF コンテンツを作成した。しかしながら、全てのファイル形式から RDF コンテンツを作成するのは難しい。そこで、RDF コンテンツ作成のための拡張性を考慮した情報システムを開発した。

つぎに、ビューの概念を実装した情報システムを開発した。この情報システムは問い合わせ文のテンプレートをビューとして管理する。それと同時に、ビューを実行する Web サービスを公開する。この Web サービスはリクエストパラメータとビューを併合して、問い合わせ文を作成する。そして、その問い合わせ文を実行後、問い合わせ結果を Web サービスのレスポンスとして返す。

最後に、その Web サービスを利用してファイル管理支援システムを構築した。つまり、このファイル管理システムは RDF コンテンツのための問い合わせ文を発行しない。全ての RDF コンテンツのための問い合わせ処理は、Web サービスを通じて実施される。これにより、問い合わせ文を記述せずとも、セマンティック Web のデータの活用が可能なることを示した。

Abstract

In the Semantic web which has been developed to utilize data around the world effectively, data are stored as RDF contents, and data are obtained directly and easily by writing query statements. However, it demands knowledge of each RDF contents' vocabularies, which are often defined by various styles, various places, and unknown users. Thus, we should have knowledge about each RDF contents, to create a query statement.

In this research, this issue has been solved by adopting the concept of "View" into the Semantic web. To share a query method, a query is defined as a "View", and through this defined "View" a data access method is provided for this defined "View". That is, by using this approach, data of the Semantic web are available without and knowledge about RDF contents.

This paper describes an application of this approach to a file management supporting system for the research and development of an early diagnosis method of dementia. First, RDF contents are created from files on an information system environment of the research and development of an early diagnosis method of dementia. However, it is difficult to create RDF contents from all file formats around the world. For this reason, an RDF extraction system is developed in consideration of scalability. Second, the concept of "View" is implemented by developing information system. This system manages the templates of query statements as "View", and publishes Web services to perform "View". The Web services combine request parameters and "View", and create a query statement. Then, they execute the query statement, and respond the result of the query statement execution. Third, a file management supporting system is developed by using these Web services. In other words, being built without any query statement for RDF contents, this system points out the possibility of reuse of data without creating any query statements in the Semantic web.

目次

第 1 章	序論	1
1.1	研究の背景	1
1.1.1	セマンティック Web	1
1.1.2	セマンティック Web の課題	3
1.1.3	セマンティック Web と三層スキーマ・アーキテクチャ	5
1.2	本研究の目的	6
1.3	本論文の構成	7
第 2 章	ユーザ要件に基づく情報統合環境の位置づけ	8
2.1	セマンティック Web 研究の諸相と本研究の位置づけ	8
2.1.1	情報統合環境としてみたセマンティック Web 技術	9
2.1.2	セマンティック Web のビュー	12
2.1.3	RDF コンテンツ活用の可能性	12
2.1.4	セマンティックデスクトップと本研究の位置づけ	13
2.2	ファイル管理の諸相と本研究の位置づけ	15
2.2.1	ファイルコンテンツを利用したメタデータ	16
2.2.2	キーワードを利用したメタデータ	17
2.2.3	ファイルの関係を利用したメタデータ	18
2.2.4	ファイル管理研究における本研究の位置づけ	18
2.3	データ中心の視点とソフトウェア中心の視点	21
第 3 章	ユーザ要件に基づく情報統合環境のための準備	23
3.1	はじめに	23
3.1.1	DICOM 画像ファイルについて	25
3.1.2	Microsoft Excel ファイルについて	27
3.1.3	PDF(Portable Document Format)ファイルについて	27
3.1.4	関係データベースについて	28
3.1.5	XML(Extensible Markup Language)ファイルについて	29
3.1.6	疫学調査における質問票調査支援システムについて	30
3.2	RDF コンテンツの作成方法	32
3.2.1	DICOM 画像からの RDF コンテンツの作成方法	32
3.2.2	Excel ファイルからの RDF コンテンツの作成方法	34
3.2.3	XMP を利用した RDF コンテンツの作成方法	36
3.2.4	関係データベースからの RDF コンテンツの作成方法	37
3.2.5	GRDDL を利用した RDF コンテンツの作成方法	38

3.2.6 質問票調査支援システムの RDF コンテンツの作成方法	40
3.4 実験と結果.....	42
3.4.1 DICOM2RDF	42
3.4.2 Excel2RDF	44
3.4.3 ReadXMPFromFile.....	47
3.4.4 RDBToRDFContents	49
3.4.5 GRDDLPlugin.....	54
3.4.6 QuestionnaireEditor.....	56
3.5 考察とまとめ	58
3.5.1 DICOM2RDF の考察とまとめ	58
3.5.2 Excel2RDF の考察とまとめ	59
3.5.3 ReadXMPFromFile の考察とまとめ.....	60
3.5.4 RDBToRDFContents の考察とまとめ	60
3.5.5 GRDDLPlugin の考察とまとめ	61
3.5.6 QuestionnaireEditor の考察とまとめ.....	61
3.6 おわりに	63
第 4 章 ユーザ要件に基づく 情報統合環境の構築.....	65
4.1 はじめに	65
4.2 ユーザ要件に基づく情報統合環境の設計.....	66
4.2.1 Pluggable Metadata Extractor の設計.....	66
4.2.2 RDFView の設計	67
4.3 ユーザ要件に基づく情報統合環境の実装.....	70
4.3.1 Pluggable Metadata Extractor の実装.....	70
4.3.2 RDFView の実装	72
4.4. 評価.....	75
4.4.1 評価方法.....	75
4.5. 考察とまとめ	78
4.5.1 Pluggable Metadata Extractor の考察とまとめ	78
4.5.2 RDFView の考察とまとめ.....	78
4.6. おわりに	80
第 5 章 ユーザ要件に基づく情報統合環境を利用した	
ファイル管理支援システム	82
5.1. はじめに	82
5.2 ファイル管理支援システムのメタデータ	84
5.3 ファイル管理支援システム.....	85
5.3.1 基礎メタデータと検査メタデータの作成及び編集	88

5.3.2	メタデータを利用したファイルの検索	91
5.3.3	メタデータを利用したファイルブラウジング機能	92
5.4	評価.....	95
5.5	考察とまとめ	95
5.5.1	関連研究.....	96
第 6 章	結論	98
6.1.	本研究のまとめ.....	98
6.1.1	RDF の利用について	99
6.2.	今後の展望.....	99
謝辞		101
参考文献		102
学位論文に係る発表論文		111

目次

図 1.1 セマンティック Web レイヤーケーキ図 (文献[4]の Figure 7: The Semantic Web layers より引用).....	2
図 1.2 セマンティック Web レイヤーケーキ図 (W3C Semantic Web Activity[2] Latest “layercake” diagram より引用).....	3
図 3.1 認知症診断法開発研究現場の情報システム環境	24
図 3.2 DICOM データ要素の構造	26
図 3.3 ファイルに埋め込まれた XMP パケット	28
図 3.4 GRDDL の概要.....	30
図 3.5 疫学調査の作業工程と質問票調査支援システム	31
図 3.6 PACS による DICOM 画像の検索.....	32
図 3.7 RDF トリプル作成の概念図	33
図 3.8 Excel のモデル	34
図 3.9 XMP パケットの例（一部抜粋）	36
図 3.10 関係データベースのスキーマ情報の例.....	38
図 3.11 名前空間として XSL ファイルを指定する場合	38
図 3.12 メタデータプロファイルに XSL ファイルを指定する場合	39
図 3.13 EntryItem クラス	41
図 3.14 NumericFormat RDF クラス定義	41
図 3.15 DICOM2RDF クラス図.....	42
図 3.16 DICOM 画像から生成した RDF コンテンツ（抜粋）	43
図 3.17 SPARQL による DICOM 画像の問い合わせ例	44
図 3.18 Excel2RDF のクラス図	45
図 3.19 レコードデータの変換例	46
図 3.20 属性の変換例	46
図 3.21 全てのワークシートを表示する問い合わせ文例	46
図 3.22 複雑な問い合わせ例	47
図 3.23 ReadXMPFromFile クラス図	47
図 3.24 XMP に含まれる DublinCore メタデータ	48
図 3.25 SPARQL 文例.....	49
図 3.26 RDBToRDFContents のクラス図	49
図 3.27 RDF スキーマを作成する.....	50
図 3.28 RDF コンテンツを作成する	50
図 3.29 症状措置機序マスタから作成した RDF コンテンツ例.....	52

図 3.30 症状措置機序マスタから作成した RDF スキーマ	53
図 3.31 ある薬剤の併用禁忌薬を検索する SPARQL 文例	53
図 3.32 GRDDLPlugin のクラス図	54
図 3.33 GRDDLPlugin が抽出した RDF コンテンツ	55
図 3.34 QuestionnaireEditor の画面写真	56
図 3.35 質問票の定義例.....	57
図 3.36 数字回答形式の定義例.....	57
図 3.37 質問票スキーマの検索.....	58
図 4.1 Pluggable Metadata Extractor システム概要.....	67
図 4.2 問い合わせ文テンプレート.....	68
図 4.3 RDFView の概要.....	69
図 4.4 Pluggable Metadata Extractor のクラス図	71
図 4.5 各 Plug-in のクラス図	72
図 4.6 RDFView システム概要	73
図 4.7 ビューの登録画面	74
図 4.8 ビューの実行結果例	74
図 4.9 プログラム行数（メソッド行数）の比較.....	76
図 4.10 セマンティックマッシュアップの例（RDFView と Yahoo! Pipe）	80
図 5.1 ディレクトリ構造抜粋.....	83
図 5.2 基礎メタデータ	85
図 5.3 検査メタデータ	85
図 5.4 システムの概要図.....	86
図 5.5 システム環境	87
図 5.6 ログイン画面	88
図 5.7 基礎メタデータの RDF スキーマ（一部抜粋）	89
図 5.8 基礎情報の登録.....	90
図 5.9 検査情報の登録.....	90
図 5.10 検査情報を表示するためのビュー	92
図 5.11 Excel ファイルのブラウジング画面	93
図 5.12 DICOM のブラウジング画面	94
図 5.13 Excel ファイルのブラウジング	94

表目次

表 3.1 データ要素形式(Data Element Format)の定義例	26
表 3.2 医薬品マスタのスキーマ構造（一部省略）	51
表 3.3 症状措置機序マスタのスキーマ構造	51
表 3.4 相互作用テーブルのスキーマ構造	51
表 3.5 GRDDL Test Cases の実験結果	55
表 4.1 RDFView 共通で利用できる変数	73
表 4.2 C&K メトリクスの計測結果	77

第1章

序論

1.1 研究の背景

1.1.1 セマンティック Web

Tim Berners-Lee 氏により提唱されたセマンティック Web は、情報システムと人間との協調活動を実現する Web 環境である[1]. また、W3C (World Wide Web Consortium) の Semantic Web Activity によると、セマンティック Web はデータの共有と再利用のための共通の枠組みを提供する概念であり、一種のデータの Web(web of data)であると説明している[2].

一方、これまでの Web は URI(Uniform Resource Identifier)で識別できるリソースとそれらリソースを繋ぎ合わせるハイパーリンクにより構成されたリソースのネットワークである。ただし、この従来の Web で人間と情報システムの協調活動を実現するには、多大な努力を必要とした。なぜなら、情報システムによる処理を考慮した形式でデータが記述されていないため、有用なデータを発見したり、処理したりすることが困難であるからである。

そのような Web に対してセマンティック Web は、情報システムが処理可能なデータの Web を構築する。このデータの Web は、リソースのメタデータとそれらメタデータの意味関係から構成されるデータのネットワークである。つまり、情報システムはリソースの意味が記述されたメタデータを処理することで、リソースの内容を踏まえた高度な処理を実現できる。

セマンティック Web では、リソースのメタデータを RDF(Resource Description Framework)[3]で表現する。RDF のデータモデルではリソースのメタデータを主語(subject)、述語(predicate)、目的語(object)の三つの要素で表現する。この三つの要素をトリプルと呼び、このトリプルの集合がデータの Web を形成する。したがって、この RDF はセマンティック Web の基盤技術として位置づけられる。

RDF などのセマンティック Web の実現に必要とされる基盤技術の開発や標準化は進み、セマンティック Web はその実践の段階にあると言われている[5]. セマンティック Web の

基盤技術はレイヤーケーキ図として表現される[4]。 2001 年のレイヤーケーキ図を図 1.1 に、 2008 年 10 月時点のレイヤーケーキ図を図 1.2 に示す。 2001 年から 2008 年の間にセマンティック Web の基盤技術の標準化および開発も進み、問い合わせ言語 SPARQL とルール記述言語 RIF の層が追加されている。

さらに近年ではセマンティック Web 技術を Web 環境以外で利用することも検討されている。 その一つに個人のデスクトップ環境における情報統合技術として利用したセマンティックデスクトップに関する研究がある[6,7,8,9]。 これらのことから、セマンティック Web は広くデータの共有と再利用のための共通の枠組みとして認識され、その成果と実績を積み重ねていることが解る。

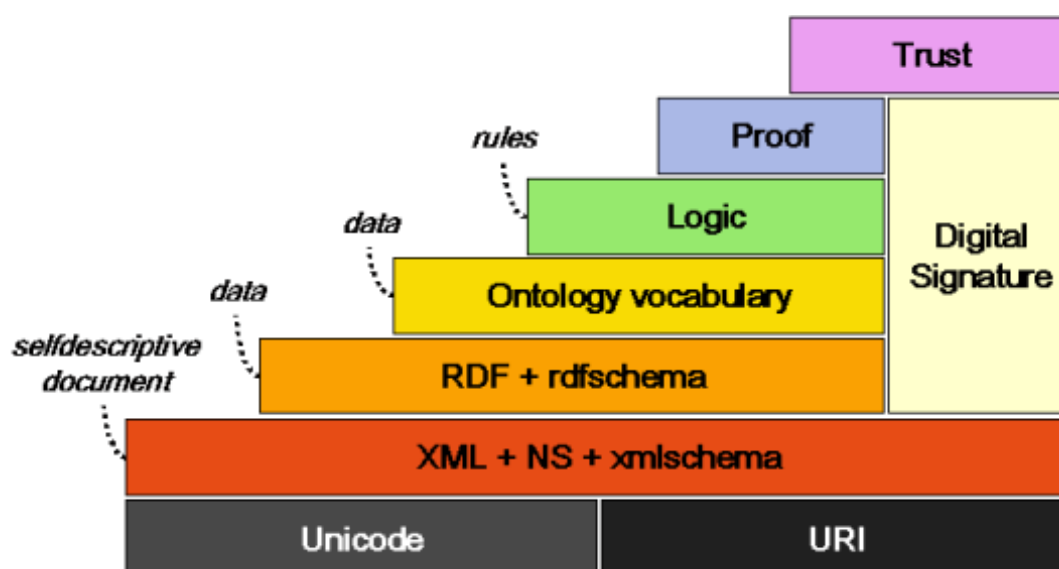


図 1.1 セマンティック Web レイヤーケーキ図
(文献[4]の Figure 7: The Semantic Web layers より引用)

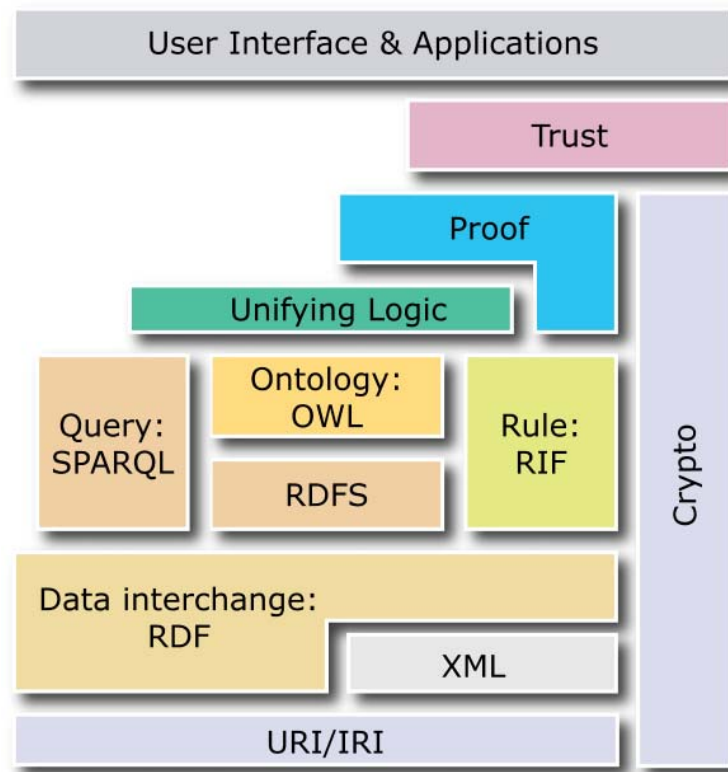


図 1.2 セマンティック Web レイヤーケーキ図

(W3C Semantic Web Activity[2] Latest “layercake” diagram1より引用)

1.1.2 セマンティック Web の課題

セマンティック Web はデータの Web である。つまり、RDF で表現されたデータ（以降、RDF コンテンツと記す）ありきの世界である。

これまで RDF コンテンツ作成手法について、多くの議論がされてきた[11~16]。また、多くの研究プロジェクトにおいて、様々な RDF コンテンツが作成されてきた。ところが、実際に利用できる RDF コンテンツの数は少ない[10]。

なぜなら、ほとんどの RDF コンテンツは独立して存在しており、互いに接続されていないからである。つまり、それらの RDF コンテンツはデータの Web を形成していないのである。さらに RDF コンテンツの多くは一般に公開されておらず、利用可能な状態になっていない[10]。これらの理由から、利用可能な RDF コンテンツの数が少なくなるのである。近年は DBpedia[17]を始めとした Linked Data[10]の試みなどで、利用可能な RDF コンテン

ツの量は増加の傾向にある。しかしながら、Web のリソースの量と比較すると、RDF コンテンツの量はごく僅かである。

このような状況の中で、Web2.0 のパラダイム[18]をセマンティック Web の実現に結びつけようという議論がある[19,20]。Web2.0 のパラダイムでは、データ中心のアプリケーション開発が実施されているのが特徴[18]で、セマンティック Web のデータ指向の考え方は Web2.0 のパラダイムにも適用可能である[20]。また、Web2.0 アプリケーションの開発手法、たとえば集合知の利用やデータサービスの緩やかなデータサービスの統合については、セマンティック Web の利用を促進させる重要な要素になると考えられる。Web2.0 のパラダイムはセマンティック Web のデータを増加させる鍵である。たとえば Web2.0 アプリケーションのデータを RDF コンテンツとして蓄積していけばセマンティック Web の実現が大きく近づく。また、RDF コンテンツの相互運用性の恩恵を Web2.0 アプリケーションも受けることができる。

その一方、RDF コンテンツの処理にも課題がある。一般的に RDF コンテンツを取得するには問い合わせが必要である。そのためセマンティック Web アプリケーションは問い合わせ文を大量に発行する。ところが、この RDF コンテンツを取得するための問い合わせ文の作成が難しい。セマンティック Web では RDF のための問い合わせ言語 SPARQL(SPARQL Protocol and RDF Query Language)[21]を利用して問い合わせ文を記述する。この SPARQL はトリプルストア[22]に保存された RDF コンテンツのトリプルのパターンを特定することで、問い合わせ処理を実現する。

ここに問い合わせ文の作成を困難にする理由が隠れている。SPARQL による問い合わせは、トリプルのパターンを特定することで実現される。つまり、問い合わせ文を記述するには、あらかじめ問い合わせに必要なトリプルパターンを把握しておく必要である。すなわち、RDF コンテンツの中で利用されている語彙とその語彙の意味(使い方)を知らなければ、その RDF コンテンツのための問い合わせ文を記述することは難しい。

セマンティック Web のオントロジは独自に定義された語彙や概念を統合し、それら語彙や概念の相互運用を可能にする[4]。ただし、汎用的なオントロジを構築することは難しい[23]。オントロジ構築支援ツール[24,25]や複数のオントロジを統合するためのオントロジアライメント研究[26]は行われてきているが、まだ汎用的なオントロジを構築するための有効な手立ての開発には至っていない。したがって、このような状況が発生した場合、個別にオントロジを用意するか、問い合わせ文で吸収するか、またはその他の手法、たとえばプログラムで吸収する必要がある。このように、利用可能なデータが存在するのに、利用する術がないという状況はセマンティック Web のデータを利用する上で大きな問題で

ある。この問題を打開するためにも、セマンティック Web には Web2.0 のパラダイムと同じように、誰もがデータを利用できて、誰もが開発者として参加可能な環境が必要である。

1.1.3 セマンティック Web と三層スキーマ・アーキテクチャ

セマンティック Web は RDF コンテンツの集合で構成されたデータの Web である。さらに問い合わせ言語 SPARQL とその処理機構を利用すれば、セマンティック Web をあたかもデータベースのように利用できる。このデータベースには概念スキーマ、外部スキーマ、内部スキーマから構成される三層スキーマ・アーキテクチャと呼ばれる基本概念がある[27]。この三層スキーマ・アーキテクチャはデータの独立性を維持するための枠組みである。それぞれの役割について説明する。

概念スキーマ。 データベース化したデータ、つまりデータモデルの構造を記述する。この概念スキーマにしたがって、データは保管される。

外部スキーマ。 個々の業務で利用するユーザの立場でデータの関係を記述する。アプリケーションは外部スキーマにしたがって、データを取得する。

内部スキーマ。 概念スキーマをコンピュータ環境で実現するための物理構造を記述する。

この三層スキーマ・アーキテクチャの概念をセマンティック Web に当てはめてみると、次のようになる。

概念スキーマ。 RDF コンテンツの構造を記述する。たとえば RDF スキーマや RDF モデル、オントロジが概念スキーマに含まれる。

外部スキーマ。 アプリケーションの処理対象としての RDF コンテンツの構造を記述する。SPARQL による問い合わせ文が外部スキーマに相当する。

内部スキーマ。 RDF コンテンツを情報システムで保持するために必要な構造を記述する。RDF モデル表現手法の RDF/XML[63], Notation3[119], N-Triples[120],そして Turtle[121]が内部スキーマに相当する。

先ほど、セマンティック Web において問い合わせ文の記述が難しいことを述べたが、この三層スキーマ・アーキテクチャの概念からみても、問い合わせ文の記述、すなわち外部スキーマの記述が難しいことが解る。つまり、そもそも外部スキーマを記述するには、概念スキーマを把握する必要があるが、セマンティック Web ではこの概念スキーマの特定

は困難であるからである。セマンティック Web では、この概念スキーマの記述を誰でも自由に記述することができる。ところが、必ずしも概念スキーマを記述した人間と外部スキーマを記述する人間が一致するわけではない。このことが、問い合わせ文の記述を難しくする。

一方、関係データベースでは外部スキーマをあらかじめ宣言しておくことができる。関係データベースでは、この機能をビューと呼び、ユーザの利便性の向上を目的に開発された[28]。特にビューは応用プログラマの視点でデータの関係性を定義可能なデータベースのマクロ的機能としての役割とデータベースの更新の際のアプリケーションの変更を最小限にする仮想のテーブルとしての役割を担うことでユーザの利便性の向上を図る。また、このビューは実テーブルを見せたくない場合など、データ保護の仕組みとしても使われることもある。

セマンティック Web は関係データベースの場合と異なり、不特定多数の人間が概念スキーマを設計し、その概念スキーマにしたがってデータが作成される。そして、その概念スキーマはいつ変更されても不思議ではない。そのため、応用プログラマは関係データベースの場合よりも、スキーマの更新に注意する必要がある。さらに、セマンティック Web には応用プログラマにとって不必要な概念スキーマも多く存在する。これらのことから、応用プログラマの視点で外部スキーマを記述し、セマンティック Web のサブセットを用意することが必要となる。つまり、セマンティック Web には関係データベースのビューの概念が必要である。

セマンティック Web のための問い合わせ文 SPARQL には、CONSTRUCT 文と呼ばれる、問い合わせ結果から RDF コンテンツを作成する機能がある。しかし、一時的な RDF コンテンツを作成するもので、関係データベースにおけるビューの定義とは大きく異なる。つまり、SPARQL ではビューを定義することができない。したがって、セマンティック Web にビューの概念を導入する必要がある。

1.2 本研究の目的

セマンティック Web は、データの共有と再利用のための枠組みである。このセマンティック Web のデータを利用するには、問い合わせ文を記述する必要がある。ところが、その問い合わせ文を記述するには、問い合わせ対象の RDF コンテンツに含まれる語彙についての知識が必要である。標準化された語彙であれば、その語彙の意味関係は自明であるため、問い合わせ文を記述するのは容易い。しかしながら、多くの場合、それらの語彙は

独自に定義される．そのため、問い合わせ対象の **RDF** コンテンツを熟知していなければ、問い合わせ文を作成することは困難である．

一方、セマンティック **Web** の実現には **Web2.0** のパラダイムとの協力が必要である．そのためには、セマンティック **Web** のデータを、応用プログラマのために提供しなければならない．なぜなら、アプリケーションは特定の目的のために開発され、そのアプリケーションが利用するデータも、特定の目的のもとに集められるからである．つまり、セマンティック **Web** のデータを応用プログラマの視点で定義しなおす必要がある．

本研究では、これらの課題を解決するために関係データベースのビューの概念をセマンティック **Web** に導入する．そのために本論文ではセマンティック **Web** のためのビューとそのビューを管理する情報システムを提案する．このセマンティック **Web** のためのビューは、応用プログラマ（以降、単にユーザと記す）の利便性の向上を目的に開発された．特に、以下の項目を実現することでユーザの利便性の向上を目指す．

1. セマンティック **Web** のマクロ的視点としてのビュー
2. データの独立性を維持するためのビュー

本論文では、このセマンティック **Web** のビューの機能を実証するために、ビューを利用した認知症早期診断法開発研究におけるファイル管理支援システムの開発も行う．そのために、本論文では **RDF** コンテンツの作成および抽出手法についても議論する．

1.3 本論文の構成

本論文は全部で 5 章から構成される．第 1 章では研究の背景と目的を述べた．第 2 章では、本研究の関連研究について述べ、本研究の位置づけを明確にする．そして、第 3 章からは、認知症早期診断法開発現場でのファイル管理支援システムでの実例をもとに説明する．第 3 章では、ユーザ要件に基づく情報統合環境の準備と題して、ファイルコンテンツからの **RDF** コンテンツ作成手法について説明する．

第 4 章では、本研究の重要なアイデアであるユーザ要件に基づく情報統合環境の構築手法について述べる．つづく、第 5 章では、ユーザ要件に基づく情報統合環境の実証のため構築したファイル管理支援システムについて説明する．そして、最後の第 6 章では結論として本研究の成果と今後の課題について述べる．

第2章

ユーザ要件に基づく情報統合環境の位置づけ

本章では、本研究の基礎となる情報統合環境としてみたセマンティック Web 研究、セマンティックデスクトップの概観、そしてファイル管理に関する研究について述べ、本研究の位置づけを行う。

2.1 セマンティック Web 研究の諸相と本研究の位置づけ

Web には個人のコンピュータ環境とは比較にならない量のデータが記録されている。それらのデータは URI で識別され、リソースと呼ばれる。Web はそのリソースとそのリソースに関係する別のリソースを HTML(Hyper Text Markup Language)のハイパーリンクで相互に接続した、巨大なハイパーテキストシステムである。

初期の Web はハイパーリンクをブラウジングすることで、目的のリソースを発見することができた。ところが Web の発展に伴いリソースの量が劇的に増加した。また、ハイパーリンクの数も膨大になった。そのため、ブラウジングにより目的のリソースを発見することが困難になっていた。

このような背景の中、開発されたのが Google[38], Yahoo!検索[39]そして Live Search[40]に代表される Web 検索エンジンである。Web 検索エンジンはリソースに含まれるテキストデータから統計的手法によりキーワードを抽出し、そのキーワードを索引付けする。ところが、自動的にキーワードを抽出するため、リソースの作者が意図しないキーワードやリソースの文脈と異なるキーワードなどが抽出される場合があった。さらに同じキーワードで索引付けられるリソースも数多くあるという問題がある。したがって、Web 検索エンジンは、キーワード抽出アルゴリズムの工夫やランキングアルゴリズムの導入し、これらの問題へ取り組んでいる。

一方、リソースの意味を処理することで、より正確な検索の実現を目指す試みが始まっ

ている。たとえば意味処理を実現した Powerset[41]では、Wikipedia[42]の意味検索を実現している。Powerset に “Which country won the first world baseball classic?” と入力すると、その検索文の意味を解析して日本チームに関する記事が提示される。このようにリソースに含まれるテキストデータの意味処理を統計的処理や自然言語処理により実現することをトップダウン型と呼ぶ。トップダウン型では情報システムがリソースの意味を特定するため、リソースを大量に処理できるという利点がある。しかしながら、キーワード検索を提供する Web 検索エンジンと同様、処理結果の精度に課題が残る。

また W3C ではメタデータによる意味処理を目指したセマンティック Web に関する技術の標準化を実施している。セマンティック Web では、リソースの意味をメタデータに記述する。そのメタデータをコンピュータが処理することで、意味処理を実現する。具体的には、リソースの意味関係を RDF として表現し、コンピュータが処理可能なデータの Web を構築する。ただし、メタデータ作成に必要な人的・時間的コスト、語彙の標準化、オントロジ構築などに課題が残る。

セマンティック Web はデータの有効活用を目指してデータの Web の開発を進めている。セマンティック Web に関する研究ではデータの Web の構築に関わる、データ構造、問い合わせ、オントロジといったあらゆる課題に取り組んでいる。本研究では、このセマンティック Web を情報統合環境として捉え、本研究で開発したユーザ要件に基づく情報統合環境の位置づけを明確にする。

2.1.1 情報統合環境としてみたセマンティック Web 技術

Web 検索エンジンを利用することで、我々は URI で識別される膨大な量のリソース利用することができる。その Web 検索エンジンは、リソースの URI とそこから抽出したキーワードから索引を作成する。そして、その索引を利用したキーワード検索を提供する。Web 検索エンジンはユーザが入力したキーワードに関係したリソースの URI をリストとして表示する。

ところが、入力されるキーワードに依存するものの、検索結果に表示される URI の数は膨大になることが多い。そのため、検索エンジンはランキングアルゴリズムを適用し、ユーザの要求に適した URI から表示するなどの工夫を行っている。それでも検索結果から必要なリソースを探し出すことは難しい。したがって、リソースに含まれる情報を効率的に統合するための情報統合手法が求められている。

情報統合とは、単一の情報源にアクセスするだけでは解決できない問題を複数の異なる

情報源を統合することで解決することを目指した技術の総称である。情報統合では、複数の異なる情報源(Source)に対し、統一的なインタフェース(Wrapper)を提供する。そして、それらの情報源を統合する(Mediator)ための仕組みを提供する。情報統合に関する研究は Web が普及する以前より行われてきたが、Web には特有の情報統合を困難にする要因がある。以下に、文献[43]により指摘された要因をまとめる。

情報の量の問題。 情報量が膨大で、必要な情報にたどり着くのが困難である。

情報の鮮度の問題。 Web ページは分散環境で独立に更新されるため、ある時点では意味があったリンク関係が、別の時点では成り立たない場合がある。

情報の質の問題。 情報の質が多様であり格差が大きい。信頼できる組織、個人の情報もあるが、正確でない情報も多い。また、対象も専門家だけでなく、さまざまなレベルのユーザが対象となる。

情報の表現の問題。 HTML を基本とした Web ページ作成のための統一的な構文は存在するが、記述内容に関する制約が存在しない。また、使用されている言語もさまざまである。

セマンティック Web では、Web に存在する情報源を巨大な知識システムとして考える。そして、その巨大な知識システムを利用した人間と情報システム(エージェント)間の高度な協調作業の実現を目指している[1]。そのために、Web 情報源に対してコンピュータが処理可能なメタデータを付与する。あらかじめ、Web 情報源に対してメタデータにより情報の質や情報の意味を関係付けることで、その統合を容易にする。文献[43]では情報統合技術の観点からセマンティック Web 技術をまとめている。文献[43]で示されたセマンティック Web 技術の位置づけに、近年のセマンティック Web 技術を加えたものを表 2.1 に示す。

これまで情報統合における Mediator 及び Planning engine にあたる機能がセマンティック Web では不明瞭であった[43]。しかしながら、この不明瞭な点は、近年の技術開発により解決しつつある。たとえば、W3C により標準化された RDF のための問い合わせ言語 SPARQL[21]は、セマンティック Web への統一したアクセス手段を提供する。その SPARQL は Mediator の役割を担うことが期待できる。

また文献[43]では Wrapper の役割を担う RDF の課題として、メタデータのオーサリング・自動生成ツール、メタデータ再利用のための仕組みの必要性が述べられている。しかしながら、W3C が標準化した GRDDL[44]はメタデータの自動生成及び再利用のための解決策の一つである。これらのことから、セマンティック Web 技術を利用した情報統合のための環境は整いつつあると言える。

本研究では、このセマンティック Web 技術を利用して、ファイルコンテンツを統合する。そのために、コンピュータ環境に保存されたファイルコンテンツからセマンティック Web を構築する。

ただしファイルコンテンツはリソースとは異なりハイパーリンクの仕組みを持たない。そのため、Web 情報統合を困難にする要因の一つ、リンク関係による情報の鮮度の問題は無視される。しかしながら、その他の情報の量の問題、情報の質の問題、そして情報の表現の問題はファイルコンテンツを統合する際の課題となる。

最後に本研究で提案する情報システムを情報統合技術の Source, Wrapper, そして Mediator に当てはめてみると、次のようになる。Source はコンピュータ環境に保存されたファイルコンテンツである。Wrapper の役割は RDF コンテンツを作成する Pluggable Metadata Extractor が担う。そして、ビューの概念を適用した RDFView が Mediator の役割を果たす。

表 2.1 情報統合の観点からみたセマンティック Web
(浦本[43], p712 の表 1 を改変転載)

構成要素	セマンティック Web	セマンティック Web 技術
Source	URI で識別される Web 資源	
Wrapper	人手で Web 視点に対するメタデータを作成する。あるいは半自動的にメタデータを構築する。	RDF(Recommended), GRDDL(Recommended), RDFa, microformats
Mediator/Planning engine	論理に基づく推論(第一階述語論理, 記述論理, etc.) ,問い合わせ	SPARQL(Recommended)
オントロジ	必要とする	OWL(Recommended)
ディレクトリ	未知のメタデータを検索する場合, サーチエンジンを用いる	
データモデル	RDF,OWL	RDF,OWL
プロトコル	HTTP	

2.1.2 セマンティック Web のビュー

セマンティック Web にビューの概念を導入した研究に文献[45,46]がある. 文献[45]では, 仮想のクラスをセマンティック Web のビューとして定義することを提案している. 一方, 文献[46]では SPARQL の名前付けグラフを拡張することでセマンティック Web のビューを定義している.

仮想のクラスを使って RDF コンテンツの統合を行うことは, これまでのオントロジによる RDF コンテンツの統合と同じである. 現在の W3C のセマンティック Web 技術を使った一番現実的なセマンティック Web のビューである. しかしながら, このビューにはオントロジ記述の課題が潜んでいる. 前にも述べたようにオントロジの記述は困難である. さらに, 仮想のクラスを定義することで, 余計な語彙が増える可能性がある. このことは, RDF コンテンツ利用を阻害する大きな要因となる.

名前付けグラフの拡張によるビューの定義は, 関係データベースにおけるビューの定義と同じ感覚で使うことができる. しかしながら, 問い合わせ言語 SPARQL の拡張が必要であるという問題がある. また, 新たなグラフの作成は問い合わせ文の記述をさらに難しくする可能性がある.

本研究が提案するユーザ要件に基づく情報統合環境では, 問い合わせ言語 SPARQL の構文拡張は行わずに, ビューの機能を実現する. また, そのビューの利用を既存の Web 技術で実現できるようにする. 本研究のビューの特徴は, 既存のセマンティック Web 技術の拡張を行わずにビューの機能を実現し, さらにそのビューは既存の Web 技術で利用可能である点である.

2.1.3 RDF コンテンツ活用の可能性

セマンティック Web は RDF コンテンツの集合から構成される. それら RDF コンテンツの活用には Web2.0 とセマンティック Web の統合が必要である[19,20]. 特に, Web 2.0 のマッシュアップ (Mash-up) と呼ばれる, 新たなパラダイムからは, RDF コンテンツ活用の可能性を垣間見ることができる. このマッシュアップは一般に公開された Web API を組み合わせて, 新たなサービスを構築する手法を表す言葉である. 一般に, 公開されている Web API は組織が保有するデータへのアクセス手法を提供する. たとえば, オンラインショップとして著名な Amazon の Web API[47]では, 書籍や CD/DVD など Amazon が所有する膨大な商品データベースへのアクセスを可能にする. また, Web 検索エンジン

の大手 Google では、検索機能や地図情報を Web API[48,49]で利用可能にしている。これらの Web API を利用することで新しい情報システム構築の機会が生まれている。

セマンティック Web は共通のデータ形式としての RDF コンテンツを利用することでマッシュアップの構築を容易にすることが可能となる[20]。文献[20]では、そのようなマッシュアップをセマンティックマッシュアップ (Semantic Mash-up) と定義している。そして、セマンティックマッシュアップに近いアプリケーションとして、Yahoo! Pipe[50]を紹介している。Yahoo! Pipe はマッシュアップを作成するための Web サービスであり、RSS や XML などの構造化されたデータの統合を支援する。

しかしながら、セマンティックマッシュアップの実現にはまだ課題がある。たとえば Dublin Core[51]は Amazon Web API が提供する書籍データベースとして見るができる。しかしながら、セマンティック Web ではまず Dublin Core が記述された RDF コンテンツを探し出す必要がある。そして探し出した RDF コンテンツのための SPARQL による問い合わせ文を記述する必要がある。Dublin Core の場合は語彙自体が標準化されており、広く世間に知れ渡っているため、問い合わせ文を記述するのは容易である。しかし、標準化された語彙は数少ない。

一方、Web API では、Web API 提供元のデータベースの構造が解らなくても、利用方法があらかじめ定められた API によりデータにアクセスできる。セマンティックマッシュアップの実現には、Web API と同様に、容易に RDF コンテンツにアクセスできる環境が求められる。本研究はパーソナルコンピュータに保存されているファイル群から RDF コンテンツを作成し、ユーザの要求に応じて文献[20]に示されたセマンティックマッシュアップの実現を目指すものと位置づけることができる。

2.1.4 セマンティックデスクトップと本研究の位置づけ

これまでリソースを発見するために、様々な取り組みが行われてきた。ところが、殆どのリソースは URI で識別可能なファイルであるため、リソース発見のための技術は、コンピュータ環境におけるファイル検索にも適用可能であるにも関わらず、Web の検索機能と比べファイルシステムが提供するファイル検索機能は低い。

近年、個人のコンピュータ環境で Web と同等の検索機能を実現するデスクトップ検索システムに注目がされている。このデスクトップ検索システムはファイルシステムの検索機能の強化を目的としている [52~54]。これらのデスクトップ検索システムでは、Web 検索エンジンと同じようにファイルに含まれるテキストデータからキーワードを抽出し、そ

のキーワードを索引付けすることで、ファイル検索機能を実現する。しかしながら、ファイルシステムにはハイパーリンクと同様の仕組みがないため、キーワード検索には課題が残る[35].

一方、個人のコンピュータに保存された個人情報の有効活用を目的とした、セマンティックデスクトップに関する研究も行われている[6~9].

個人のコンピュータは保存されているファイルの内容に関して、多くの情報を扱うことができない。たとえば、保存されたファイルに著者や題目、そして発行日などの情報が含まれていても、コンピュータはそれらの情報を処理できない。また、コンピュータは保存されたファイルのフォーマットに応じた方法でしか情報にアクセスできない。そのため、ファイル形式に応じてファイルを整理することができない。

たとえば、仕事の予定とメールは別々のファイルで保管され、内容自体が関係していても、コンピュータはその関係を考慮した処理を実行できない。なによりもコンピュータに保存されるデータの量が増大しているため、目的のデータが多くのデータに埋もれてしまい、目的のデータにアクセスすることが難しくなりつつある。

セマンティックデスクトップはセマンティック Web 技術により統一されたインタフェースにより、保存された個人情報を自由にアクセス可能なデスクトップ環境を提供することを目指した概念である。

文献[7]によるセマンティックデスクトップの定義を以下に示す。

“セマンティックデスクトップは個人の文書、マルチメディア、メッセージなどの全ての電子情報を保存するデバイスである。これらはセマンティックリソースとして解釈され、各リソースは *URI* によって識別され、全てのデータは *RDF* グラフとしてアクセスかつ問い合わせできる。Web のリソースは保存可能で、そして作成されたコンテンツは他者と共有できる。オントロジはユーザに対して、個人的なメンタルモデルの表現と、意味が付着した相互接続可能な情報とシステム形式の表現を可能にする。アプリケーションはそれを尊敬し、保存、読み込み、そしてオントロジ及びセマンティック Web プロトコルを介して通信を行う。セマンティックデスクトップはユーザの記憶を大幅に増強させるデバイスである。”

セマンティックデスクトップの中には、ピアツーピア(Peer to Peer)、ソーシャルネットワークサービス (Social Network Service) などのネットワークリソースを組み合わせ、コンピュータ環境に保存された個人情報の操作・統合するものもある[8]。セマンティック

デスクトップは、コンピュータ環境及びネットワーク環境に保存された個人の情報をセマンティック Web 技術で関連付けることで、それらの個人情報の統合を可能にする。そのことにより、ユーザの作業と情報システムの協調作業を可能にする。

本研究が提案するユーザ要件に基づく情報統合環境においても、コンピュータ環境に保存されたファイル群から RDF コンテンツを作成することでそれらファイルコンテンツの操作を可能にする。また、RDF コンテンツをユーザの要求に応じて統合し、その情報を表示する。これらはセマンティックデスクトップを実現するための一手法と捉えることができる。

セマンティックデスクトップに保存されるすべてのデータはセマンティックリソースとして、つまり RDF コンテンツとして解釈されなければならない。ただし、前述したように、セマンティック Web の RDF コンテンツの利用方法には課題が残る。このことはセマンティックデスクトップにおいて同様である。本研究によるユーザ要件に基づく情報統合環境はその課題の一つの解決法を示すことでセマンティックデスクトップの実現を目指すものと位置づけられる。

2.2 ファイル管理の諸相と本研究の位置づけ

オペレーションシステムの一機能であるファイルシステムは、コンピュータに記録されたデータをファイルとして管理する。ファイル管理のために、このファイルシステムはファイルの名前やファイルの作成・更新・アクセス日時などのファイルに関する情報をファイルのメタデータとして持つ。とくにファイルの名前はディレクトリとして束ねられ、ファイルシステムにより階層的に管理される。またファイルシステムはファイルやディレクトリの作成や削除といった操作機能やファイルに関する情報を利用したファイルの検索機能を備える。

一方、ファイルシステムを利用するユーザにとって重要なのはファイルに格納されたデータ（以降、ファイルコンテンツと記す）である。ファイルコンテンツを管理する上で、ファイルやディレクトリの名前、そしてディレクトリによる階層構造は重要である。ところが、これらはファイルシステムを利用するユーザにより作られる。ファイルコンテンツを効率的に管理するためにも、ファイルシステムのユーザには、ファイルやディレクトリの命名規則や分類規則を定めることが求められる。

しかしながら、このような規則を定めることは難しい。たとえば、論文データを含むファイルの分類規則を定めるにしても、著者別、テーマ別、学会誌別、年代別と多様な分類

方法が存在する。したがって、どの分類手法が適切であるかを決定することは困難である。また、分類規則を決定できたとしても、ファイルの種類により、その変更を余儀なくされる場合もある。

さらに、ファイルシステムの検索機能を利用して検索したファイルが目的のファイルであるかどうかを確認するためには、アプリケーションソフトウェアを起動する必要がある。一部のファイルシステムには画像などのファイルコンテンツに対応したビューを備える場合もある。しかしながら、一般的にファイルシステムはファイルコンテンツの内容を確認するための手段を用意していないため、アプリケーションソフトウェアを起動して、ユーザ自身でファイルコンテンツを確認する必要がある。たとえば、ある論文の筆頭著者を調べるためにも、論文を検索し、アプリケーションソフトウェアを起動して作者の名前を確認する必要がある。アプリケーションソフトウェアを起動する必要があるため、確認には時間がかかる。

これらの課題に対して、古くからメタデータを利用したファイル管理を手法が提案されてきた[29~34]。本論文では、これらの研究をメタデータの種類により分類する。これにより、ファイル管理研究の概観を述べ、本研究で開発したユーザ要件に基づく情報統合環境の位置づけを明確にする。

2.2.1 ファイルコンテンツを利用したメタデータ

ファイルコンテンツを利用したメタデータとはファイルコンテンツ特有の情報を利用したメタデータである。多くの場合、ファイルコンテンツにはファイルを識別可能な情報が含まれている。たとえば、メールファイルには、宛先情報、送信元情報、送信日時、そして件名などの情報がファイルコンテンツとして含まれる。これらの情報はメールファイルの識別に使うことができる。

文献[29]の SFS(Semantic File System) は UNIX のファイルシステムを拡張したファイル管理支援システムである。SFS では、ファイルコンテンツからメタデータを作成するために、メタデータ抽出機構 **Transducer** を備える。**Transducer** はファイルの種類に対応して作成され、ファイルコンテンツを解析することでメタデータを作成する。SFS はその **Transducer** が作成したメタデータを利用したファイル検索機能を提供する。

一方、文献[35]ではファイル管理の中でも検索機能に特化したデスクトップ検索システムを提案している。このデスクトップ検索システムはセマンティック Web 技術を利用する。具体的には、電子メールファイルや Web キャッシュファイル、そしてディレクトリ

階層を解析することで、ファイルの関係情報を RDF として表現する。そして、デスクトップ検索ソフトウェアの Beagle[54]に RDF の検索機能の実装を行うことで、高度な検索を実現する。

ファイルコンテンツからメタデータを作成する際の利点と欠点は次の通りである。利点は、ファイルコンテンツを一度でも解析すれば、自動的にメタデータを作成できる点である。欠点は、文献[29]の SFS の Transducer や文献[35]のように個別のファイルに対応した解析方法を用意することが必要な点である。

2.2.2 キーワードを利用したメタデータ

キーワードはファイルの内容を表す単語の集まりであり、古くからファイルの分類手法として利用されてきた。キーワードを入力することで、そのキーワードに関連付けられた、ファイルを探し出すことができる。適切なキーワードをファイルに関連付ければ、ファイル発見の効率は向上する。しかしながら、ファイルの内容とかけ離れたキーワードなど不適切なキーワードは、ファイル発見の効率を逆に下げる。

文献[30]では、ファイル自身の名前やそのファイルの名前やディレクトリの名前を含んだファイルのパス情報から、ファイルのキーワードを自動的に作成している。ただし、ユーザアカウントの名前やオペレーションシステムが作成したディレクトリの名前などファイルのキーワードに不要なものは削除する。また、ユーザが不要なキーワードを指定することもできる。

また、文献[31]では複数ユーザによるファイル管理支援手法を提案している。ファイルのキーワードは各ユーザが独立して作成する。つまり、ユーザが個別にファイルのキーワードリストを所有する。一方の文献[32]では、del.icio.us[59]に代表される folksonomy[60]を利用したファイル管理支援システムである TagFS を提案している。TagFS は各ユーザがファイルのキーワードを作成する。ただし、文献[31]とは異なり、それらのキーワードは全てのユーザで共有される。また、文献[33]の Database File System (DBFS) もユーザがファイルのキーワードを作成する。ただし、ファイル自身がキーワードのリストを所有する。

ユーザによるキーワード作成には、ファイルの内容を正確に表現できる可能性が高いという利点がある。

ただし、ユーザ毎にファイルのキーワードリストにぶれが生じることを考慮しなければならない。文献[31]では、ユーザ個別にキーワードリストを保存し、必要に応じて統合す

る手法を示している。また、文献[32]では、同じようにユーザ毎にキーワードリストを所有する。ただし、ファイルのキーワード作成時にそのリストを共有させることで、作成されるキーワードを集約させている。一方、文献[36]では、ファイルがキーワードリストを保有し、誰でも編集することができる。

一方、情報システムによるキーワード作成には、手作業でキーワードを作成する必要がある点と、キーワードにぶれが無い点である。ただし、自動的にキーワードを作成した場合には、不要なキーワードが含まれる可能性がある。そのため、文献[30]のように不要なキーワードを削除するための工夫が必要である。

2.2.3 ファイルの関係を利用したメタデータ

文献[33]はファイルの関係を表現するリンクとファイルの属性をメタデータとして利用したファイル管理支援システム Linking File System(LiFS)を提案している。LiFS では、ユーザがツールを通じて、ファイルの関係とファイルの属性を自由に定義することができる。

一方、文献[33]及び文献[37]ではミドルウェアによるファイル管理機能を提案している。アプリケーションソフトウェアがこのミドルウェアを通じてファイルにアクセスすることで、ファイルの関係を抽出し、ファイルのメタデータとして保存する。

文献[33]の手法のように、ユーザがファイルの関係やファイルの属性を作成することで、ファイルの正確なメタデータと作ることができる。また、単なるキーワードとは異なり、よりファイルの内容を詳細に表すことができる。ただし、メタデータを作成するユーザの間で、作成する属性の名前を統一し、その属性の意味を共有する必要がある。

一方、文献[34,37]のように、アプリケーションソフトウェアの操作やファイルのアクセスなど、ユーザの行動からファイルの関係を抽出する利点は、自動的にメタデータを作成できる点にある。たとえば、プログラムで利用されるライブラリは、明らかに相互に関係があり、かつ有用な情報を抽出できる。ただし、必ずミドルウェアを通過するように、アプリケーションソフトウェアを修正する必要がある。

2.2.4 ファイル管理研究における本研究の位置づけ

本研究では、ファイルのメタデータをファイルのコンテンツから自動生成するのと平行して、ユーザによるファイルメタデータの作成支援も行う。メタデータの表現形式には

RDF を利用する.

文献[29]が作成するメタデータはファイルの属性とその値の組み合わせ（属性値の組）である. しかしながら, 属性値の組では, 文献[33,34]で示されたファイルとファイルの関係を表現するには不十分である. つまり, 属性値は必ずしも定数ではない.

一方, RDF は主語, 述語, 目的語の三つ組により, メタデータを表現する. RDF の目的語は定数 (リテラル) またはリソースのどちらかの形式を取ることができる. RDF はリソースのメタデータを表現するためのデータモデルとして設計されたため, 当然であるがファイルのメタデータを表現するのに適している.

また, ユーザによるメタデータの作成には, 属性の名前やその意味の統一が必要である. 本研究では, あらかじめ属性の名前や属性の意味を統一し, RDF スキーマにメタデータの語彙を定義する. そして, その RDF スキーマをユーザがメタデータを作成する際に利用する.

また, ファイルのコンテンツを統合するという点では, 文献[33,36]のカスタマイズ可能なファイルビューの考えが本研究に近い. ただし, 本研究とはそのコンテンツの表示過程が大きく異なる. 文献[33]のファイルビューでは, 属性値の組からなるメタデータを単純にフィルタリングすることで, 目的のファイルを表示する.

一方, 文献[36]では, 複数のキーワードを指定することで, ファイルの絞り込みを行う. しかしながら, 本研究ではファイルコンテンツを RDF コンテンツとして表現することで, 全てのファイルコンテンツに対して統一した問い合わせ手段を提供する. ファイルコンテンツに対しての問い合わせ結果を利用することで, 高度なファイル表示が実現できる. ファイルの関係情報を RDF で表現することで高度なファイル管理の実現を目指した研究はこれまでも行われてきている[34,35,37]. これらの研究はファイルの発見に注目している. しかしながら, 本研究では, ファイルのコンテンツの発見に注目している. 最後に本論文で紹介したファイル管理支援研究と本研究におけるメタデータの種類, メタデータの付与方法, そしてその統合方法を表 2.2 にまとめる.

表 2.2 ファイル管理研究の特徴と本研究の位置づけ

	メタデータの種類	メタデータ付与方法	統合手法
本研究	ファイルコンテンツ，ファイル関係	手作業及び自動抽出	RDF コンテンツへの問い合わせ
文献[29]	ファイルコンテンツ	自動抽出	属性値の組のフィルタリング
文献[33]	ファイル関係	ソフトウェアが対応すれば自動抽出	属性値の組のフィルタリング
文献[30]	キーワード	手作業，自動抽出，辞書の利用	キーワードによる統合
文献[31]	キーワード	手作業，キーワードの読み替え，キーワードの統合	キーワードによる統合
文献[32]	キーワード	手作業，キーワードの共有	キーワードによる統合
文献[34,37]	ファイル関係	ソフトウェアが対応すれば自動抽出	問い合わせ
文献[36]	キーワード，ファイルコンテンツ	手作業及び自動抽出	キーワードによる統合
文献[35]	ファイルコンテンツ，ファイル関係	自動抽出	RDF コンテンツへの問い合わせ

2.3 データ中心の視点とソフトウェア中心の視点

個人のコンピュータのファイルシステムに保存されているファイルはアプリケーションソフトウェアが作成したデータの集合である。本来、データ自体はアプリケーションソフトウェアに依存しない。しかしながら、アプリケーションソフトウェアがそのデータをファイルとして保存した瞬間に、そのデータはアプリケーションソフトウェアに依存した存在となる。したがって、ファイルとして保存されたデータはアプリケーションソフトウェア中心の視点で処理されることになる。

たとえば、多くのリソースとしての HTML ファイルもアプリケーションソフトウェア中心の視点で処理される。HTML は W3C により標準化されたハイパーテキストを構成するためのマークアップ言語である。したがって、その HTML 形式で記述されたテキストデータはハイパーテキストシステムというアプリケーションソフトウェアで処理するために作られる。

一方、我々はそれらのファイルをデータ中心の視点で捉える。たとえば、ある小説家の文書が HTML ファイルで保存されていれば、我々は Web ブラウザを使って HTML ファイルを表示する。同じように、その小説家の文書が Microsoft Word ファイルで保存されていれば、Microsoft Word を使ってそのファイルを表示する。

つまり、我々はデータの為にアプリケーションソフトウェアを選択するのではなく、ファイル形式の為にアプリケーションソフトウェアを選択する。我々にとってファイルとは、データのための単なる器である。それらのデータを活用するためにも、ファイルに格納されたデータをデータ中心の視点から捉えるべきである。

セマンティック Web はデータ中心の視点で捉えるための考え方やそのための技術と捉えることができる。セマンティック Web の基盤技術である RDF はデータの構造を定義する。ただし、RDF で記述されたデータはその RDF のルールに従えば処理することができる。そのため RDF コンテンツはデータの相互運用性が高い。

これまでのファイルはソフトウェア中心の視点で処理されてきた。そのため、ファイルに保存されたデータの相互運用性は低かった。しかし、それらのデータを活用するためにも、相互運用性の向上が求められる。

本研究では、コンピュータ環境にセマンティック Web の構築を試みる。つまり、ファイルシステムに保存されてきたファイルのコンテンツをデータ中心の視点で捉える。そのために、ファイルのメタデータを RDF で表現する。

本論文では，ユーザはファイルではなく，そのファイルのコンテンツを必要とするという点に注目し，ファイルをデータ中心の視点で捉える．そして，そのデータ中心の視点で捉えられたファイルコンテンツに対して，その利用方法をユーザの要求に応じて定義する方法を示す．これらのことにより，ユーザ要件に基づく情報統合環境が実現できる．

第3章

ユーザ要件に基づく情報統合環境のための準備

3.1 はじめに

本章では、ユーザ要件に基づく情報統合環境を構築するために必要となる RDF コンテンツの作成手法について説明する。先にも述べたように、RDF コンテンツの作成は、セマンティック Web の基盤構築に必要不可欠な工程である。特に、セマンティックデスクトップの実現には、全てのリソースを RDF コンテンツ化することが求められる[7]。しかしながら、現実的に全てのファイル形式から RDF コンテンツを作成するのは困難である。なぜなら、アプリケーションソフトウェア毎に独自のファイル形式が存在するし、新たなファイル形式が今後も登場することも予想される。したがって、全てのファイル形式から RDF コンテンツを作成するのは困難である。そこで、本研究では認知症早期診断法開発研究現場で利用されているファイル形式を対象を絞り、RDF コンテンツを作成する。

本研究が対象とする認知症早期診断法開発研究の現場では、MRI(Magnetic Resonance Imaging) 検査，MEG(Magnetoencephalography) 検査，健康診断・血液検査，MMSE(Mini-Mental State Examination)やタッチパネル式簡易認知機能検査などの各種検査を実施している。そして、認知症早期診断法開発研究の現場の研究者達は各種検査結果を蓄積し、それらを分析する。各種検査結果データは各データに対応したアプリケーションソフトウェアで加工される。そして、各アプリケーションソフトウェアは、それぞれ独自のファイル形式で検査結果データを格納する。そのようにして作成されたファイル群は、認知症早期診断法開発研究を実施する研究者のパーソナルコンピュータからアクセス可能なネットワーク対応ハードディスクドライブ保存される。パーソナルコンピュータに保存されたそれらのファイル群は、認知症早期診断法開発研究を実施する研究者により、第一次データ群(未加工の検査結果データ)、第二次データ群(グラフデータや図データ)、第三次データ群(表形式データ)に分類される。研究者達は主に第三次データ群を使って、研究を実施する。第一次データ群と第二次データ群は彼らの研究の中で必要なときにだけ利用される。

3.1.1 DICOM 画像ファイルについて

DICOM は ACR(American College of Radiology) と NEMA(National Electrical Manufacturers Association)により定められた医療用画像機器のためのネットワーク規格である[55]. DICOM 画像形式は DICOM 規格の中で定められた医療用画像データのためのファイル形式で、広く医療用画像データのファイル形式として利用されている.

一般に CT(Computerized Tomography),MRI,そして PET(Positron Emission Tomography)で撮影された医療用画像は DICOM 画像形式に変換され、PACS(Picture Archiving and Communication System)で管理される.

しかしながら、PACS は一般に高価な機器であるため、導入できる医療機関は限られてくる. たとえば、K-PACS(無料で利用できる DICOM ビューワ)[56]と Conquest DICOM Software(パブリックドメインの DICOM サーバ)[57]等の無料のソフトウェアを組み合わせることで、PACS 機能の実現は可能であるが、一般的ではない. そのため、撮影された医療用画像データは、光ディスクや光学磁気ディスク、あるいはハードディスクドライブに保存されることが多い. それらの環境では、DICOM 画像ファイルはディレクトリで分類され、保存されている. これらの DICOM 画像ファイルから RDF コンテンツを作成することで、PACS と同等の医療用画像検索を実現できる.

その DICOM 画像ファイルのファイル形式は画像データが格納されたデータ部分と患者情報と撮影機器情報からなるメタデータが格納されたヘッダ部分に分けられる. DICOM 画像のメタデータは、大量に撮影される医用画像の管理に不可欠な情報である. DICOM 規格ではこれらのメタデータをデータ要素という形式で定義している[58]. この定義をデータ辞書と呼ぶ. データ要素は、要素の識別に用いられるグループ番号、要素番号からなるデータ要素タグ(Data Element Tag)、値領域のデータ形式を定義する値表現(VR; Value Representation)、値領域の個数を表す値複数度(VM; Value Multiplicity)、そして実際のデータ要素には含まれないが、データ要素の意味を表す名前 (Name) から構成される (図 3.2 参照).

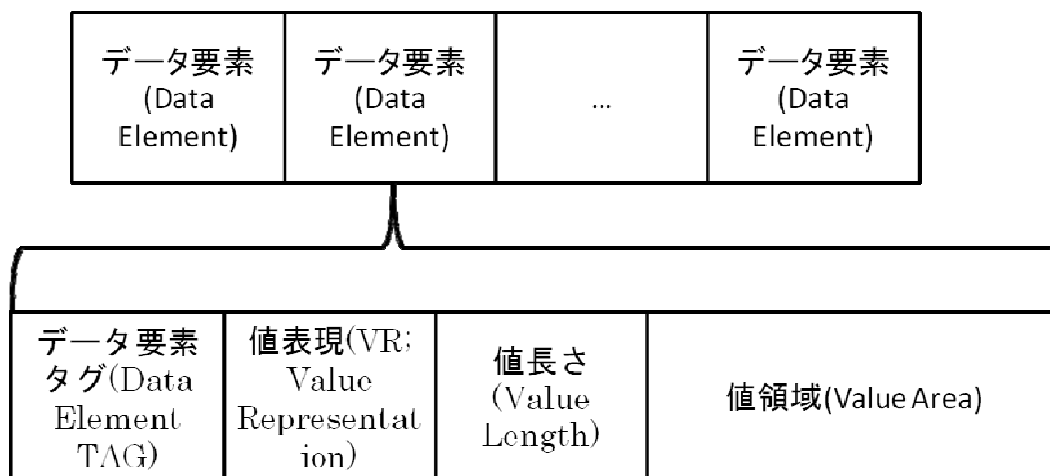


図 3.2 DICOM データ要素の構造

表 3.1 はこのデータ辞書で定義された「患者の年齢(Patient's Age)」を表現するデータ要素である。値表現「AS」は Age String の略称であり、AS で識別されるデータ形式は年齢を表す数字と期間を示す D,M,Y の記号からなる文字列であることを意味する (28 歳なら 28Y)。また、値複数度に 1 が設定されているため、このデータ要素の値は一つである。本研究では、DICOM 画像ファイルに含まれるデータ要素を利用して RDF コンテンツを作成する。

表 3.1 データ要素形式(Data Element Format)の定義例

データ要素タグ (Data element tag)		値表現 (VR;Value Representation)	値複数度 (VM;Value Multiplicity)	名前 (Name)
グループ番号 (Group Number)	要素番号 (Element Number)			
0010	1010	AS	1	Patient's Age

3.1.2 Microsoft Excel ファイルについて

Microsoft 社の製品 Excel は広く知られた表計算ソフトウェアの一つである(以降, Excel と記す). 表計算ソフトウェアは, 家計簿の管理や実験データの管理, そして医療用統計データの管理など幅広く利用されている.

この Excel のファイルには, ワークブックという単位でデータが格納される. そのワークブックは複数の行と列から構成されるワークシートと呼ばれる表形式データの集合である. ワークブックは, 一種の関係データベースとしてみることができる. 実際に, ODBC(Open Database Connectivity)を利用することで, ワークブックを関係データベースとして扱われる場合もある.

本研究ではこの Excel ファイルから RDF コンテンツを作成することで, ワークブックの検索を実現する. Excel ファイルから RDF コンテンツを作成することで, 複数の Excel ファイルを横断したデータ検索を実現できる.

3.1.3 PDF(Portable Document Format)ファイルについて

Adobe 社が開発した PDF は文書や図表を始めとしたドキュメントファイルの配布に広く利用される. PDF は情報システム環境に依存せずにドキュメントファイルを提供することができる. それを背景に, PDF で保存される文書データは増加の傾向にある.

一方, Adobe 社を中心に増加するメディアデータの管理を効率的に行うことを目的にメタデータ埋め込み型コンテンツ管理環境 XMP(Extensible Metadata Platform) [61]の開発が行われている. XMP ではメタデータを XMP パケットと呼ばれる形式でファイルのヘッダ部分に埋め込む[62]. XMP のメタデータには RDF が採用されている. 実際のメタデータは RDF/XML 形式[63]で記述する.

XMP は PDF を始めとした Adobe 社のソフトウェア製品での実装が進められている. また, XMP は JPEG(Joint Photographic Experts Group),PNG(Portable Network Graphics),TIFF(Tagged Image File Format)などの複数の画像ファイルにも対応している. 図 3.3 に実際に PDF ファイルに埋め込まれた XMP パケット例を示す. 表示にはバイナリエディタ BZ[64]を用いた. 本研究では XMP の仕組みを利用することで RDF コンテンツの抽出を行う.

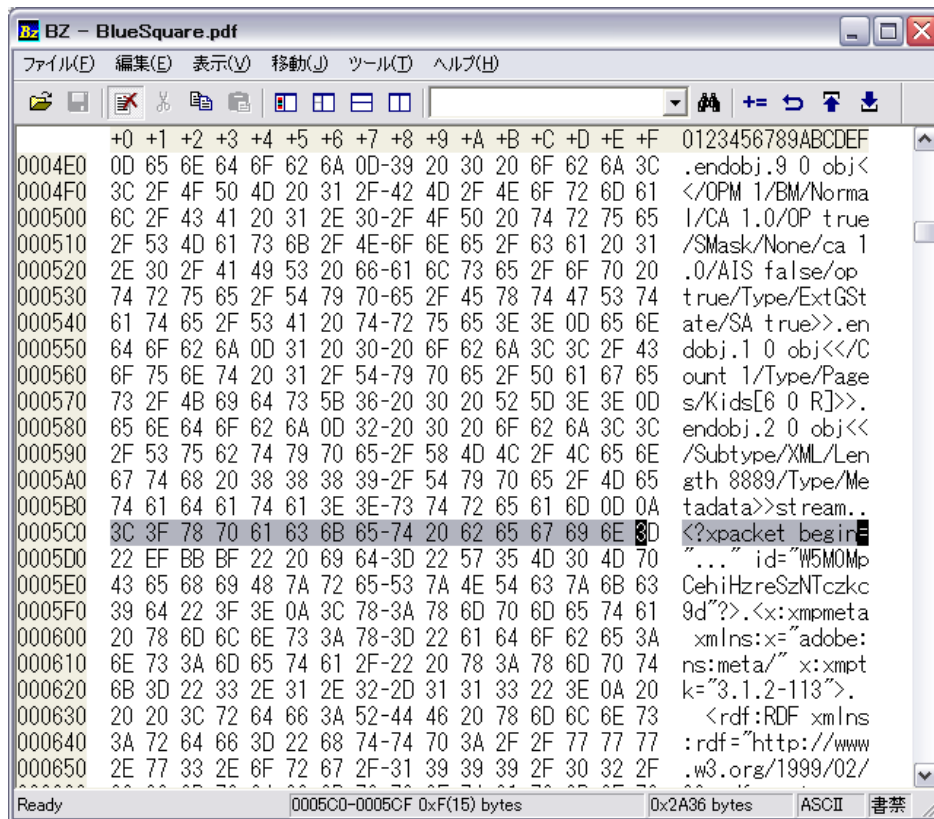


図 3.3 ファイルに埋め込まれた XMP パケット

3.1.4 関係データベースについて

健康診断データや検査データ、そして医用画像データなどの医療データの多くは、医療機関ごとに保存されている。その大量に保存された医療データを活用するために、医療データを管理する情報システムには単なるパターンマッチングによるデータ検索だけでなく、医療用語シソーラスを用いた類似検索や意味的情報を基にした情報検索が求められている[65]。これらのことを背景に、医薬品の規制に関する医学用語集からオントロジ（階層概念定義とプロパティ属性定義）の構築が行われている[66]。このような医学用語集に代表されるデータベースやシソーラスのデータは、完全でないにしても、ある程度整理された用語間の関係や属性情報を有している。したがって、これらから RDF コンテンツを作成できれば効率がよい[66]。

本研究では医療情報コードが格納された関係データベースから、RDF コンテンツを作成する。本論文の提案手法は関係データベースのテーブル構造に依存しない。これまでにも

医療情報コードは医療データに意味付けに利用されている[67,68]。ただし、これらの研究ではメタデータの属性値としてそのコードの値を利用しているだけである。

3.1.5 XML(Extensible Markup Language)ファイルについて

XML ファイル形式は拡張可能なマークアップ言語である。XML ファイルは、その名の通り拡張性が高いため、広くデータ交換用ファイル形式として利用されている。たとえば、Excel はデータの保存形式として XML 形式を選択することができる。このように、XML ファイルはその拡張性の高さから、多くのアプリケーションソフトウェアのファイル形式として採用されている。XML ファイルから RDF コンテンツを作成できれば、XML を利用した数多くのファイル形式に保存されたデータを利用できる。

このような背景のもと、W3C では XML ファイル形式から RDF コンテンツを取り出すための方法を GRDDL としてまとめ、その標準仕様の公開を実施している[44]。

GRDDL は XSL(Extensible Style Language)変換を利用して、XML ファイルから RDF コンテンツを作成する。GRDDL は最小限の手数で XML ファイルから RDF コンテンツを作成することを可能にする。その GRDDL の概要を図 3.4 に示す。

GRDDL に対応した XML ファイルを作るのは容易である。まず、RDF コンテンツを作成するための変換アルゴリズムを XSL ファイルに記述する。そして、その変換アルゴリズムを XML ファイルに関係付ける。具体的には作成した XSL ファイルを XML ファイルの名前空間、または XHTML(Extensible Hypertext Markup Language)のメタデータプロファイル[69]に指定する。以上の手順で、GRDDL に対応した XML ファイルを作ることができる。

GRDDL に対応した XML ファイルから RDF コンテンツを作成する手順はつぎの通りである。まず、対象の XML ファイルが GRDDL に対応しているかを確認する。つぎに、XML ファイルに関連付けられた XSL ファイルを取得する。そして、その XSL ファイルを使って、対象の XML ファイルの XSL 変換を実行する。その XSL 変換処理後、RDF コンテンツが作成される。

本研究では GRDDL の機能を実装することで、XML ファイル形式からの RDF コンテンツの作成に対応する。

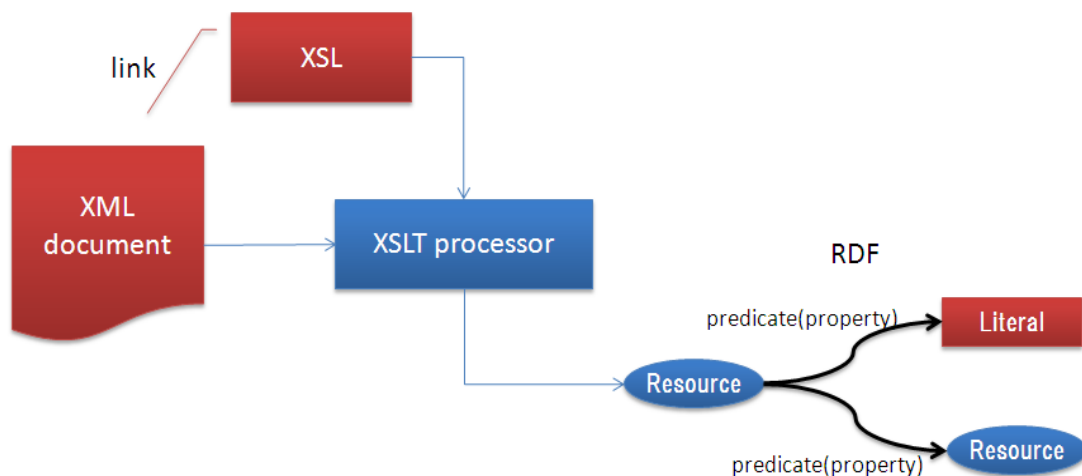


図 3.4GRDDL の概要

3.1.6 疫学調査における質問票調査支援システムについて

疫学調査では、地域を限定するなど特定の集団に対して長期間にわたって質問票調査などを実施する。この質問票調査には、主に質問項目の設計、調査の実施、回収データの入力・分析、調査報告書作成の作業工程がある（図 3.5 参照）。そして調査実施者が調査開始から調査目的と調査項目の因果関係を十分に納得できるまでには、質問項目の追加、修正、削除が繰り返されることが多い。

このような事象に効率的に対応できる情報システムの構築は、質問票調査のデータ入力や分析作業などにおける時間と費用面での効率化という観点から望まれている。

疫学調査は目的に応じて種々実施されている。このような調査は、単独の調査として意味を持つばかりではなく、いくつかの調査結果が関連を持つことも考えられる。それにも関わらず、そのような調査を支援する情報システムは、個々の調査だけを支援する観点から構築される。それゆえに、今後の情報システムには、関連を持った調査間のデータを柔軟に取り扱うことができるような観点も要求される。

本研究では、このような観点を踏まえて、主に質問票調査の効率化を目指した情報システムを構築した。なおここで提案するアプローチは、いろいろな目的に応じた疫学調査へ適用可能であり、そのような疫学調査間の連携を支援する技術への発展性も考慮している。疫学調査の作業工程と質問票調査支援システムの概略図を図 3.5 に示す。

質問票調査支援システムは、疫学調査の一環として実施される質問票調査において、質問項目の追加、修正、削除に対して耐性を持ち、これらの事象の発生に際し可能な限り柔軟に対処できる情報システムを目指して構築された。このような情報システムでは、調査項目の変更に対する情報システムの耐性や、調査項目の追加、修正、削除を考慮したデータ分析の整合性の確保に配慮することが必要となる。

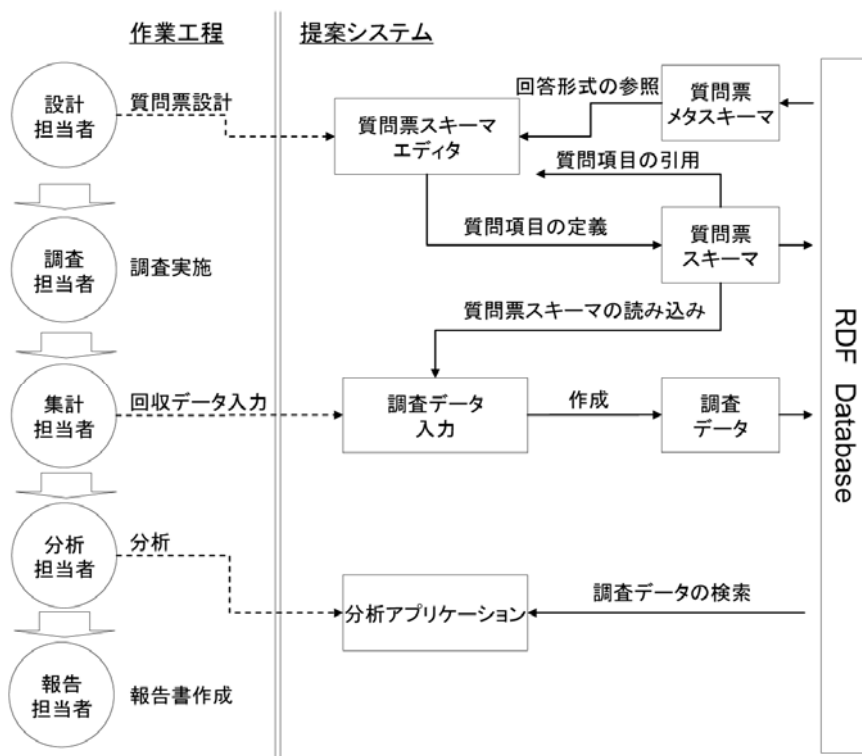


図 3.5 疫学調査の作業工程と質問票調査支援システム

本研究では、これらの要件を満たすために、RDF を用いて、質問票で用いられる用語体系とデータ構造を定義するアプローチをとる。この質問票調査支援システムは、対象とする質問票調査を一つのドメインとして捉え、その調査で用いられる質問項目を RDF スキーマとして定義する。この RDF スキーマを質問票スキーマと呼ぶ。また質問票スキーマを記述する際に利用される回答形式は RDF コンテンツとして保存される。この RDF コンテンツを質問票メタスキーマと呼ぶ。

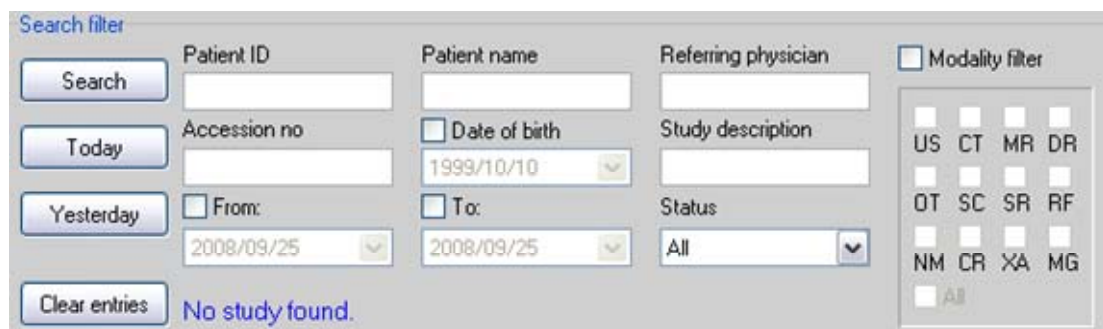
これら質問票スキーマと質問票メタスキーマを利用することで、質問票調査を支援する情報システムに耐性と柔軟性を持たせることが可能となる。このような情報システムは質

問票調査を効率化させ、さらにいろいろな質問票調査を支援し、連携させる基盤情報システムとして利用できるものと考えられる。

3.2 RDF コンテンツの作成方法

3.2.1 DICOM 画像からの RDF コンテンツの作成方法

一般的な PACS は、DICOM 画像ファイルに含まれるデータ要素を用いて DICOM 画像の検索を行う。たとえば PACS では DICOM データ要素に含まれる患者 ID や患者の氏名、そして生年月日などの DICOM データ要素の問い合わせを行う（図 3.6）。DICOM 画像を検索する上で、DICOM データ要素は不可欠な情報である。



The screenshot shows a web-based search interface for a PACS system. It features a 'Search filter' section on the left with buttons for 'Search', 'Today', 'Yesterday', and 'Clear entries'. The main area contains several input fields for search criteria: 'Patient ID', 'Patient name', 'Referring physician', 'Accession no', 'Date of birth' (with a calendar icon), 'Study description', 'Status', 'From' (with a date dropdown), and 'To' (with a date dropdown). A 'Modality filter' section on the right includes a grid of checkboxes for different imaging modalities: US, CT, MR, DR, OT, SC, SR, RF, NM, CR, XA, MG, and an 'All' option. The search results area at the bottom displays the message 'No study found.'

図 3.6 PACS による DICOM 画像の検索

本研究ではこの DICOM データ要素を利用して RDF コンテンツを作成する。DICOM 画像ファイルに含まれる各データ要素から RDF トリプルを作成する（図 3.7 参照）。以下に、DICOM 画像から RDF を作成する際の手順を示す。

1. Web またはディスク上の DICOM 画像の位置情報から URI を作成する。
2. 手順 1. で作成した URI を用いて RDF リソースを作成する。
3. DICOM 画像からヘッダ情報を抽出する。
4. 手順 3. で抽出したヘッダ情報から RDF プロパティと RDF リテラルを作成する。
5. 手順 2. 及び手順 4. で作成した RDF リソース、RDF プロパティ、RDF リテラルを用いて RDF トリプルを作成する。
6. 手順 3. から手順 5. をヘッダ部の最後まで繰り返す。

本来, RDF はリソースのメタデータを表現するための手法として定義されたものである。そのため扱われるファイルは URI で識別できなければならない。まず DICOM 画像ファイルのリソース URI を作成する。本研究ではファイルを URL(Uniform Resource Locator) で識別できれば, その URL をリソース URI として利用する。しかし, URL で識別できない場合, つまり Web で公開されていないファイルの場合には, File URL[70] を作成して, リソース URI とする (手順 1,2)。

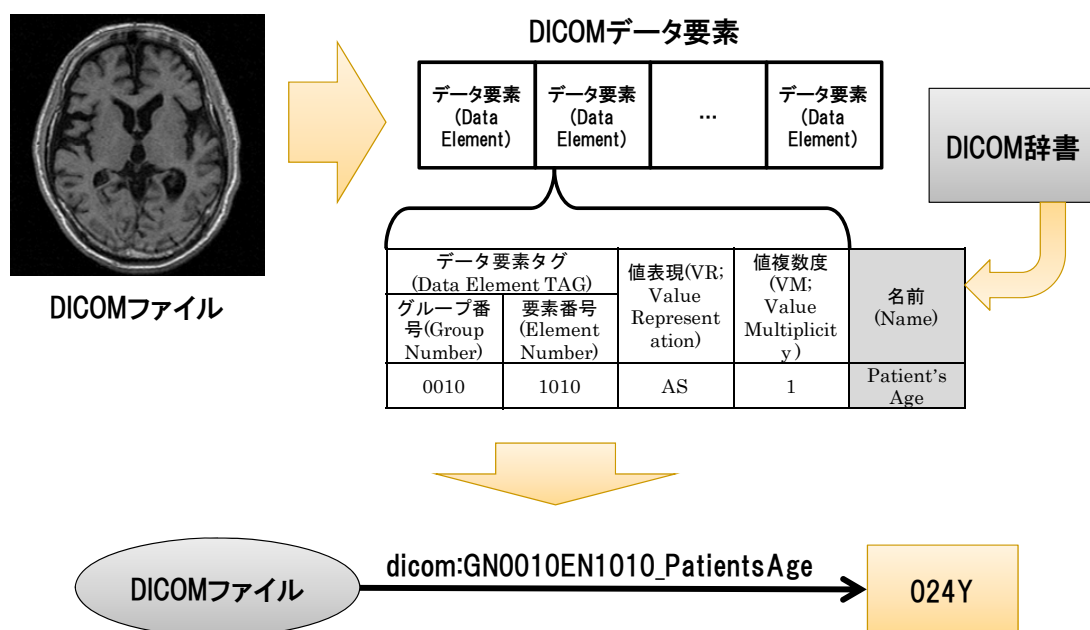


図 3.7 RDF トリプル作成の概念図

つぎに DICOM 画像からファイルヘッダー部を読み込み, そこから DICOM データ要素を取り出す (手順 3)。その取り出した DICOM 要素を各データ要素タグに切り分ける。つぎに各データ要素のタグを検索キーとして DICOM 規格のデータ辞書からデータ要素の名前を検索する。そして, データ要素タグと検索したデータ要素の名前から RDF プロパティのための URI を作成する。データ要素タグはグループ番号と要素番号の組であるため, それを識別するために各番号の前にグループ番号の場合は GN, そして要素番号の場合は EN を付ける。さらに, データ要素タグとデータ要素の名前の間にはアンダーラインを挟む。そして, データ要素の値領域から RDF リテラルを作成する (手順 4)。

最後に, 先ほど作成した RDF リソース, RDF プロパティ, そして RDF リテラルから RDF トリプルを作成する (手順 5)。これらの処理を DICOM 画像のヘッダー部に含まれる全てのデータ要素を処理するまで繰り返す (手順 6)。

3.2.2 Excel ファイルからの RDF コンテンツの作成方法

本研究では Excel ファイルから RDF コンテンツを作成するために、Excel のワークブックやワークシートの構造と経験則から Excel ファイルの RDF モデルを開発した。図 3.8 の上部に、Excel ファイルの RDF モデルを示す。そこで新たに `WorkBook`, `WorkSheet`, `Attributes`, `Attribute`,そして `Record` の五つのクラスと `hasWorkSheet`, `hasAttributes`, `hasAttribute`,そして `hasRecord` の四つのプロパティを定義した。

これら五つのクラスはそれぞれ Excel のワークブック、ワークシート、属性、行（レコード）を表す。また、四つのプロパティは各クラスの関係を表す。なお、図中の接頭辞 `xls` はこれらのクラスとプロパティの名前空間を示す(その他の接頭辞については図 3.8 の右下部参照)。

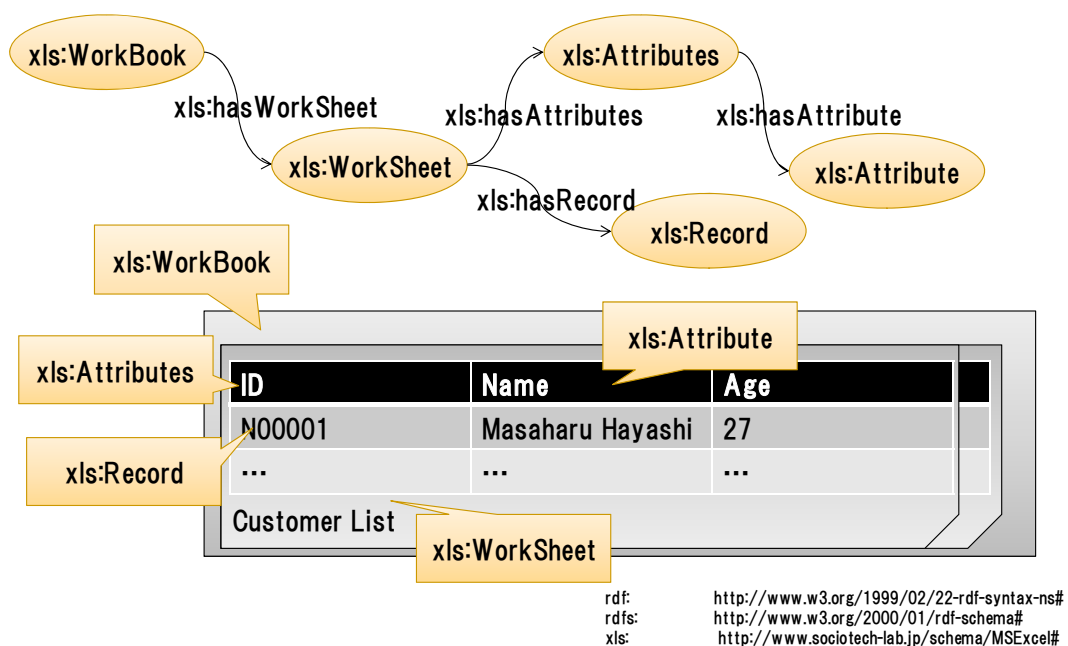


図 3.8 Excel のモデル

図 3.8 の下部にそれら五つのクラスと Excel ファイルとの関係を図示する。この五つのクラスの中でも特に **Attribute** クラスと **Record** クラスは Excel に格納されたデータを表す重要なクラスである。前者の **Attribute** クラスはワークシートの第一行目の値から作成される。Excel を利用する場合、ワークシートの第一行目を属性の名前として用いる場合が多い。本論文では、その値を属性の名前と判断して、**Attribute** クラスを作成する。後者の **Record** クラスは第一行目以外の行を表現する。Excel は表計算ソフトウェアという特性上、一つの行は一つのレコードとして扱うことができる。つまり、**Record** クラスは一つのレコードを表す。これら **Attribute** クラスと **Record** クラスを使って、Excel のセルに格納されたデータを表現する。具体的には、次のようなトリプルとして表現される。

<Record のインスタンス> <Attribute のインスタンス> “セルの値”。

このトリプル表現により、ワークシートの属性名を使って、レコードを特定することが可能となる。

3.2.3 XMP を利用した RDF コンテンツの作成方法

XMP ではファイルのヘッダ部分に RDF コンテンツを XMP パケットとして埋め込む。図 3.9 に XMP パケットの例を示す。ファイルのヘッダ部分に埋め込まれた XMP パケットは XML 形式の文字列である。XMP の仕様書には、XMP のデータモデル、XMP スキーマの定義情報、そして PDF を始めとしたファイル形式への埋め込み手法の実例が示されている[62]。

本研究では XMP の仕様書[62]を参考に XMP パケットに抽出手法の実装を行う。また、XMP の特徴上、埋め込まれた RDF コンテンツには明示的なリソース URI が指定されていない。したがって、本研究では RDF コンテンツの抽出時にリソース URI を設定する。以下に、XMP の仕様書[62]に示された XMP の抽出手法を示す。

1. XMP パケットの開始を示す文字列 “<?xpacket begin=” を探索する。
2. バイト配列からエンコーディング形式を判定する。
3. XMP パケットの終了を示す文字列”<?xpacket end=’w’?>”までの全文字列を取り出す。
4. 取り出した文字列から RDF コンテンツを取り出す。

```
1. <?xpacket begin="" id="W5M0MpCehiHzreSzNTczkc9d"?>
2. <x:xmpmeta xmlns:x="adobe:ns:meta/" x:xmptk="3.1.2-113">
3.   <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
4.     <rdf:Description rdf:about=""
5.       xmlns:xapMM="http://ns.adobe.com/xap/1.0/mm/">
6.       <xapMM:InstanceID>uuid:ce7a0ad8-c8d1-11da-9cb6-000d936b79c4</xapMM:InstanceID>
7.       <xapMM:DocumentID>adobe:docid:indd:af62ddc6-213f-11da-ac18-
         fe020baf4f13</xapMM:DocumentID>
8.       <xapMM:RenditionClass>proof:pdf</xapMM:RenditionClass>
9.     </rdf:Description>
10.    <rdf:Description rdf:about=""
11.      xmlns:xap="http://ns.adobe.com/xap/1.0/"
12.      xmlns:xapGImg="http://ns.adobe.com/xap/1.0/g/img/">
13.      <xap:CreateDate>2005-09-07T15:00:22-07:00</xap:CreateDate>
14.      <xap:ModifyDate>2006-04-10T13:37:19-07:00</xap:ModifyDate>
15.      <xap:MetadataDate>2006-04-10T13:37:19-07:00</xap:MetadataDate>
16.      <xap:CreatorTool>Adobe InDesign CS2 (4.0)</xap:CreatorTool>
```

図 3.9 XMP パケットの例（一部抜粋）

3.2.4 関係データベースからの RDF コンテンツの作成方法

関係データベースはデータベースの構造情報をスキーマとして保持する。そのスキーマには、データベースを構成するテーブルの構造、テーブルに含まれる属性、そしてそれら属性の定義域の情報が含まれる（図 3.10）。

本研究では関係データベースのスキーマに含まれるテーブル名や属性名、そして定義域を利用して、RDF コンテンツを作成する。以下に関係データベースのスキーマ情報を利用した RDF コンテンツ作成手法を示す。

1. RDF コンテンツを作成する対象テーブルを定める。その対象テーブルを RDF のクラスとして定義する。
2. 対象テーブルのスキーマ情報を取得する。そのスキーマ情報からテーブルの属性を取り出し、RDF のプロパティとして定義する。また、先に定義した RDF のクラスをプロパティの定義域として設定する。
3. それら定義した RDF のプロパティとクラスを用いて、テーブルのデータを RDF コンテンツに変換する。まず、データベースの URI とテーブル名、そして主キー項目を組み合わせて URI を構築する。主キーが存在しないテーブルの場合は列番号を利用する。そのリソース URI と先に定義しておいた属性と対応したプロパティ、そしてデータをリテラル値として、RDF のトリプルを作成する。

主キーがある場合のリソース URI の例：

```
jdbc:postgresql://localhost/normaldb?symptom#S000548
```

主キーがない場合のリソース URI の例：

```
jdbc:postgresql://localhost/normaldb?interaction#RowNum32683
```

4. 3 の処理を全てのデータに対して行う。

以上の手順により、関係データベースのデータを RDF コンテンツ化できる。本手法は、関係データベースのスキーマ情報を利用して RDF コンテンツを作成している。したがって、データベースの構造が変わっても、スキーマ情報がある限り対応可能である。

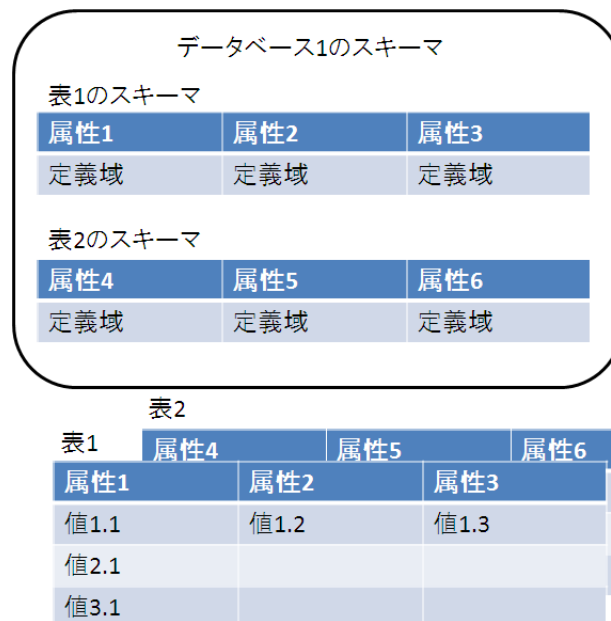


図 3.10 関係データベースのスキーマ情報の例

3.2.5 GRDDL を利用した RDF コンテンツの作成方法

GRDDL[44]では、XML ファイルから RDF コンテンツを作成するために、XSL ファイルを XML ファイルの名前空間またはメタデータプロファイルに指定する。

図 3.11 に XSL ファイルを名前空間に指定した XHTML ファイルの実例を示す。この例では名前空間 `grddl` に定義された `transformation` 属性の値として XSL ファイル(`test.xsl`)を指定している。GRDDL 処理機構はこれらの記述から、変換アルゴリズムとして `test.xsl` を取得し、その変換アルゴリズムを利用して XHTML ファイルの XSL 変換を実施する。

1. `<html xmlns="http://www.w3.org/1999/xhtml">`
2. `xmlns:grddl='http://www.w3.org/2003/g/data-view#'`
3. `grddl:transformation="test.xsl">`

図 3.11 名前空間として XSL ファイルを指定する場合

1. `<head profile="http://www.w3.org/2003/g/data-view">`
2. `<link rel="transformation" href="http://www.test.org/test.xsl" />`
3. `</head>`

図 3.12 メタデータプロファイルに XSL ファイルを指定する場合

XHTML 文書のメタデータプロファイルに変換アルゴリズムを指定することもできる。メタデータプロファイルを使った変換アルゴリズムの指定方法では link 要素を利用する。Link 要素の rel 属性の値に transformation と設定することで、その link 要素が GRDDL の変換アルゴリズムを指し示していることを表現する(図 3.12)。

さらに GRDDL では名前空間またはメタデータプロファイルに直接変換アルゴリズムを指定するのではなく、参照先のファイルに変換方法を含めることも許可している。このことにより、共通の RDF コンテンツ作成方法を共通の名前空間またはメタデータプロファイルを持った GRDDL 対応 XML ファイルを作成することができる。以下に、文献[44]を参考にした GRDDL 処理手法を示す。

1. 対象文書の名前空間またはメタデータプロファイルに `http://www.w3.org/2003/g/data-view` が宣言されているかを確認する。
2. 名前空間にその URL が確認できれば、transformation 属性を探し出し、XSL ファイル名を取得する。
3. メタデータプロファイルにその URL が確認できれば、transformation が設定された rel 属性を持つ link 要素を探し出し、XSL ファイル名を取得する。
4. その URL を確認できない場合、名前空間やメタデータプロファイルに指定された他のコンテンツを取得する。XSL ファイルを発見するまで、そのコンテンツに対してこれまでと同じ処理を繰り返す。
5. 取得した XSL ファイルを用いて、対象文書から RDF コンテンツを作成する。

3.2.6 質問票調査支援システムの RDF コンテンツの作成方法

質問票調査支援システムは質問の回答形式として次の六つの形式を持っている[71]。これらは質問票メタスキーマとして RDF スキーマに定義される。ここで定義された回答形式は枝分かれ質問として利用することができる。以下にそれらの回答形式を示す。

単一回答形式 性別（男・女）や年号（大正・昭和・平成）などの複数の選択項目から単一の回答を選択する形式である。

複数回答形式 選択項目から複数の回答を選択する形式である。

順位回答形式 何らかの達成度を問う設問を想定している。達成度に応じて回答者が選択肢に順位を付ける形式である。

段階回答形式 何らかの達成度を問う設問を想定している。達成度を測る段階（1 から 5 など）が提示されており、そのどれかを回答者が選択する形式である。

自由回答形式 回答者が自由に回答を記述することが可能な形式である。

数字回答形式 年齢や身長など、回答として数字を求める形式である。

質問票調査支援システムでは、これら六つの回答形式を RDF クラスとして定義する。具体的には、単一回答形式は `SelectFormat RDF`、複数回答形式は `SelectMultipleFormat RDF`、順位回答形式は `OrderFormat RDF`、段階回答形式は `NStageFormat RDF`、自由回答形式は `FreeFormat RDF`、そして最後の数字回答形式は `NumericFormat RDF` と定義した。

図 3.13 に `EntryItem` クラスの定義を示す。このクラスは六つの回答形式クラスの基底クラスである。そして `title`, `description`, `hasEntry` の三つのプロパティを持つ。それぞれ `title` プロパティは回答項目の題目を `description` プロパティは回答項目の詳細を表す。そして最後のプロパティ `hasEntry` は回答項目の上下関係を表現する。

1. <rdfs:Class rdf:ID="EntryItem">
2. <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
3. </rdfs:Class>
4. <rdf:Property rdf:ID="hasEntry">
5. <rdfs:domain rdf:resource="#EntryItem"/>
6. </rdf:Property>
7. <rdf:Property rdf:about="http://purl.org/dc/elements/1.1/creator">
8. <rdfs:domain rdf:resource="#EntryItem "/>
9. </rdf:Property>
10. <rdf:Property rdf:about="http://purl.org/dc/elements/1.1/date">
11. <rdfs:domain rdf:resource="#EntryItem "/>
12. </rdf:Property>

図 3.13 EntryItem クラス

1. <rdfs:Class rdf:ID="NumericFormat">
2. <rdfs:subClassOf rdf:resource="#EntryItem"/>
3. </rdfs:Class>
4. <rdfs:Class rdf:ID="FreeFormat">
5. <rdfs:subClassOf rdf:resource="#EntryItem"/>
6. </rdfs:Class>
7. <rdf:Property rdf:about="hasValueType">
8. <rdfs:domain rdf:resource="#FreeFormat"/>
9. </rdf:Property>
10. <rdfs:Class rdf:ID="SelectFormat">
11. <rdfs:subClassOf rdf:resource="#EntryItem"/>
12. </rdfs:Class>
13. <rdfs:Class rdf:ID="NStageFormat">
14. <rdfs:subClassOf rdf:resource="#EntryItem"/>
15. </rdfs:Class>
16. <rdf:Property rdf:ID="hasStage">
17. <rdfs:domain rdf:resource="#NStageFormat"/>
18. </rdf:Property>

図 3.14 NumericFormat RDF クラス定義

六つの回答形式クラスは **EntryItem** クラスを継承して定義される。たとえば段階回答クラス **NStageFormat** の定義を図 3.14 に示す。図 3.14 の二行目にて **EntryItem** RDF クラスのサブクラスであることが定義されている。

質問票支援システムでは、質問票の語彙を事前に定義している質問票メタスキーマを参照し、質問票スキーマを定義する。質問票スキーマは **Questionnaire** クラスを継承した質問票クラスを中心に定義される。各質問項目は六つの回答形式に対応するクラスを参照して定義される。

3.4 実験と結果

本節では前述した RDF コンテンツ作成方法の実装と、その機能について説明する。また、結果として実際に作成した RDF コンテンツとその問い合わせ文を示す。

3.4.1 DICOM2RDF

DICOM2RDF は前述した DICOM 画像からの RDF コンテンツ作成手法を実装したプログラムである。セマンティック Web ライブラリ **Jena**[72] と画像処理ライブラリ **ImageJ**[73] を用い Java 言語による実装を行った。

DICOM2RDF は指定された URL から DICOM 画像を読み込み、その DICOM 画像のメタデータを RDF 形式で出力する。

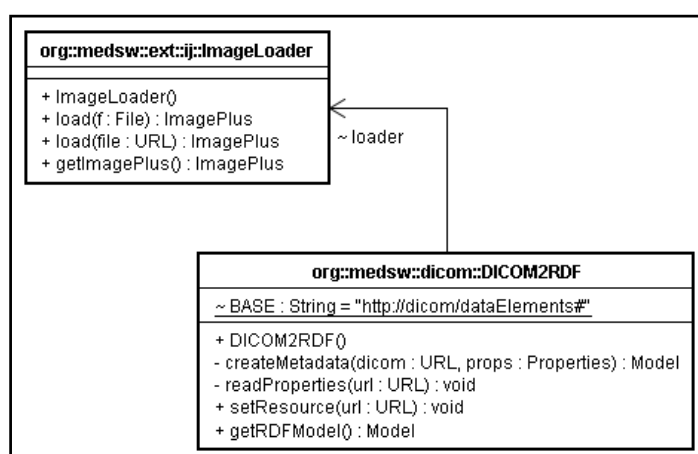


図 3.15 DICOM2RDF クラス図

図 3.15 に DICOM2RDF のクラス図を示す。それぞれのクラスの説明を行う。まず ImageLoader クラスは DICOM 画像からヘッダ情報を読み込む。そして DICOM2RDF クラスは読み込んだヘッダ情報に含まれる DICOM データ要素を RDF 形式で出力する。具体的には DICOM2RDF の setResource(URL)メソッドにセットされた URL を使って、readProperties(URL)メソッドを呼び出し、DICOM データ要素を取得する。次に、createMetadata(URL,properties)メソッドを呼び出して、その DICOM データ要素（引数 properties）とリソース URI（引数 URL）から RDF コンテンツを作成する。

図 3.16 に実際に DICOM2RDF を使って、DICOM 画像から作成した RDF コンテンツの一部を示す。

```

1. <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:dicom="http://dicom/dataElements#">
2. <dicom:Dicom rdf:about="http://localhost/Masa.dcm">
3. <dicom:GN0008EN0022_AcquisitionDate
   rdf:datatype="http://www.w3.org/2001/XMLSchema#int">20020828</dicom:GN0008EN0022_AcquisitionDate>
4. <dicom:GN0021EN1082_unknown
   rdf:datatype="http://www.w3.org/2001/XMLSchema#double">0.0</dicom:GN0021EN1082_unknown>
5. <dicom:GN0021EN0010_unknown>GEMS_RELA_01</dicom:GN0021EN0010_unknown>
6. <dicom:GN0020EN000E_SeriesInstanceUID>1.2.840.113619.2.5.1762369908.1411.1030507379.586
   </dicom:GN0020EN000E_SeriesInstanceUID>
7. <dicom:GN0019EN10AC_unknown
   rdf:datatype="http://www.w3.org/2001/XMLSchema#double">0.0</dicom:GN0019EN10AC_unknown>
8. <dicom:GN0008EN0090_ReferringPhysiciansName></dicom:GN0008EN0090_ReferringPhysiciansName>
9. <dicom:GN0018EN0050_SliceThickness
   rdf:datatype="http://www.w3.org/2001/XMLSchema#double">1.4</dicom:GN0018EN0050_SliceThickness>
10. <dicom:GN0010EN0030_PatientsBirthDate
   rdf:datatype="http://www.w3.org/2001/XMLSchema#int">19800112</dicom:GN0010EN0030_PatientsBirthDate>
11. <dicom:GN0020EN0052_FrameofReferenceUID>1.2.840.113619.2.5.1762369908.1435.1030507340.980
   </dicom:GN0020EN0052_FrameofReferenceUID>
12. <dicom:GN0021EN105B_unknown
   rdf:datatype="http://www.w3.org/2001/XMLSchema#double">180.0</dicom:GN0021EN105B_unknown>

```

図 3.16 DICOM 画像から生成した RDF コンテンツ（抜粋）

PACS は DICOM のデータ要素を利用して DICOM 画像の検索を行う。本手法で変換された RDF コンテンツにたいしても、PACS と同様に DICOM 画像の検索が可能である。

たとえばデータ要素名 PatientID で検索する SPARQL 問い合わせ文を図 3.17 に示す。問い合わせ文の五行目でデータ要素のグループ番号、六行目でデータ要素の要素番号、七行目でデータ要素の名前を指定することで、データ要素の選択を行っている。八行目は選択されたデータ要素の値を指定している。SPARQL 処理機構はこの問い合わせ文を処理し、これら問い合わせ文の条件に適合した DICOM 画像のリソース URL を結果として返す。

```
1. PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2. PREFIX dcm:<http://dicom/dataElements#>
3. SELECT ?uri
4. WHERE {
5.   ?prop dcm:GroupNumber "0010" .
6.   ?prop dcm:ElementBumer "0020" .
7.   ?prop dcm:Name "PatientID" .
8.   ?uri ?prop "XXX00049" .
9. }
```

図 3.17 SPARQL による DICOM 画像の問い合わせ例

3.4.2 Excel2RDF

Excel2RDF は前述した Excel RDF データモデル（図 3.8 参照）を実装した、Excel から RDF コンテンツを作成するプログラムである。Excel2RDF の構築には、Excel を解析するために POI ライブラリ[74]と Jena ライブラリ[72]を利用した。図 3.18 に Excel2RDF のクラス図を示す。

Excel2RDF は readXLS メソッドにセットされた Excel ファイルの URL から、RDF コンテンツを作成する。ところで readXLS メソッドは二つの URL を引数にとるが、Excel ファイルの URL と RDF コンテンツのリソース URI が異なる場合があるためである。readXLS メソッドは、MS Excel のワークシートを順に捜査する。ワークシートを発見した場合は、createProperty メソッドを呼び出し、そのワークシートの第一行目からプロパティを作成する。第二行目以降は、createTypedLiteral メソッドがそれらの行の各セルからリテラルを作成する。そして、先に作成しておいたプロパティとそのリテラル、そしてその行の URI から、RDF トリプルを作成する。これらの処理を全てのセルを処理するまで続ける。

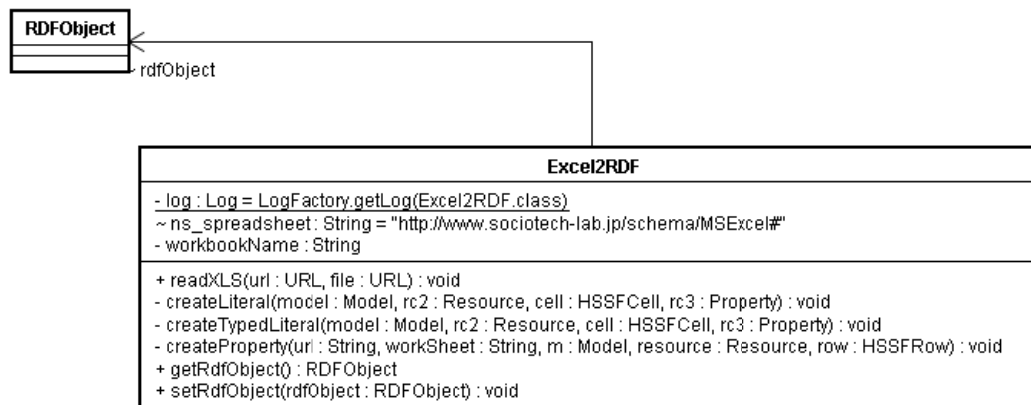


図 3.18 Excel2RDF のクラス図

Excel2RDF が作成した RDF コンテンツの一部を図 3.19 と図 3.20 に示す。図 3.19 は MS Excel ファイルのレコードを表現したものである。このレコードには合計七つのプロパティが指定されている。そのプロパティの定義の例を図 3.20 に示す。Excel2RDF では各 Excel のセルは一つのトリプルとして表現される。

このように Excel ファイルから RDF コンテンツを作成することで、SPARQL による問い合わせが可能となる。たとえば、全てのワークシートを表示する場合は図 3.21 のように問い合わせ文を記述する。SPARQL 処理機構はこの問い合わせ文の条件に適合した Excel ワークブックのリソース URL とそのラベルを検索結果として返す。

また、SPARQL による問い合わせ文を工夫することで、さらに複雑な検索ができる。図 3.22 は MS Excel ファイルの項目 MMSE の値が 21 点以下の被験者の点数と被験者 ID、そしてこれらの値が含まれたレコード URI を表示するための問い合わせ文である。

このように Excel ファイルを RDF コンテンツ化することで、あたかもデータベースとして Excel ファイルを検索することができる。このことにより、たとえば複数の Excel ファイルを横断した問い合わせを実現できる。

```

1. <xls:Record rdf:about="http://localhost/nouanq.xls#ws000001.row000007">
2. <j.O:ws000001.attribute000038
   rdf:datatype="http://www.w3.org/2001/XMLSchema#double">1.0</j.O:ws000001.attribute000038>
3. <j.O:ws000001.attribute000029
   rdf:datatype="http://www.w3.org/2001/XMLSchema#double">1.0</j.O:ws000001.attribute000029>
4. <j.O:ws000001.attribute000020
   rdf:datatype="http://www.w3.org/2001/XMLSchema#double">1.0</j.O:ws000001.attribute000020>
5. <j.O:ws000001.attribute000011
   rdf:datatype="http://www.w3.org/2001/XMLSchema#double">1.0</j.O:ws000001.attribute000011>
6. <j.O:ws000001.attribute000002
   rdf:datatype="http://www.w3.org/2001/XMLSchema#double">1.0</j.O:ws000001.attribute000002>
7. <j.O:ws000001.attribute000001
   rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">2006-08-
   10T15:00:00Z</j.O:ws000001.attribute000001>
8. <j.O:ws000001.attribute000000
   rdf:datatype="http://www.w3.org/2001/XMLSchema#string">XXX000025</j.O:ws000001.attribute
   000000>
9. </xls:Record>

```

図 3.19 レコードデータの変換例

```

1. <xls:Attribute rdf:about="http://localhost/nouanq.xls#ws000001.attribute000000">
2. <rdfs:label>No</rdfs:label>
3. </xls:Attribute>

```

図 3.20 属性の変換例

```

1. PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
2. PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
3. PREFIX xls: <http://www.sociotech-lab.jp/schema/MSExcel#>
4. select *
5. where {
6. ?worksheet rdf:type xls:WorkSheet .
7. ?worksheet rdfs:label ?worksheetLabel .
8. }

```

図 3.21 全てのワークシートを表示する問い合わせ文例

```

1. PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2. PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3. PREFIX xls: <http://www.sociotech-lab.jp/schema/MSExcel#>
4. select distinct ?val_id ?val_mmse ?uri
5. where {
6.   ?prop_mmse rdfs:label "MMSE" .
7.   ?label_row xls:hasLabel ?prop_mmse .
8.   ?row ?prop_mmse ?val_mmse .
9.   filter (?val_mmse <= 21 ) .
10.  ?prop_id rdfs:label "ID" .
11.  ?label_row xls:hasLabel ?prop_id .
12.  ?row ?prop_id ?val_id .
13.  ?worksheet xls:hasRow ?row .
14.  ?uri xls:hasWorkSheet ?worksheet .
15. }

```

図 3.22 複雑な問い合わせ例

3.4.3 ReadXMPFromFile

ReadXMPFromFile は前述した XMP から RDF コンテンツを抽出する手法を実装したプログラムである。図 3.23 に ReadXMPFromFile のクラス図を示す。

ReadXMPFromFile
- XPACKET_END : String = "<?xpacket end='w'/?>" - XPACKET_END_BYTES : byte[] = XPACKET_END.getBytes() - XPACKET_BEGIN : String = "<?xpacket begin" - XPACKET_BEGIN_BYTES : byte[] = XPACKET_BEGIN.getBytes()
+ extractRDF(xmpfile : String, uri : String) : RDFObject + extract(url : URL) : String

図 3.23 ReadXMPFromFile クラス図

extractRDF メソッドでは、XMP に対応したファイルから RDF コンテンツを作成する。実際に XMP パケットの抽出処理を行うのは extract メソッドである。なお、extractRDF メソッドの二つ目の引数 uri は抽出した RDF コンテンツにリソース URI を付与するため

に設けてある。XMP ではファイルに埋め込まれた RDF コンテンツにはリソース URI が指定されていない。したがって、本メソッドにより抽出する際に明示的にリソース URI を指定している。

PDF に埋め込まれた XMP には、Dublin Core メタデータ[51]も含まれる(図 3.24 参照)。この Dublin Core メタデータにより、タイトルやキーワードといったプロパティを指定した PDF ファイルの検索が可能となる。

図 3.25 は抽出された RDF コンテンツに対して、キーワード (Dublin Core メタデータの subject) に Blue が含まれるファイルを問い合わせするための問い合わせ文例である。条件文は図 3.25 の四行目から八行目までである。条件は複数個想定されるキーワード (図 3.25 の五行目と六行目) の中で文字列 Blue が含まれることである。

SPARQL 処理機構は、この問い合わせ文の条件に適合する RDF コンテンツのリソース URI とキーワードを結果として返す。

```
1. <rdf:Description rdf:about=""  
2.     xmlns:dc="http://purl.org/dc/elements/1.1/">  
3.     <dc:format>application/pdf</dc:format>  
4.     <dc:title><rdf:Alt><rdf:li xml:lang="x-default">Blue Square Test File -  
       .pdf</rdf:li></rdf:Alt></dc:title>  
5.     <dc:description>  
6.     <rdf:Alt>  
7.         <rdf:li xml:lang="x-default">XMPFilesBlueSquare test file, created in  
       InDesign CS2, saved as .indd and .pdf.</rdf:li>  
8.     </rdf:Alt>  
9. </dc:description>  
10. <dc:subject><rdf:Bag><rdf:li>XMP</rdf:li><rdf:li>Blue  
       Square</rdf:li><rdf:li>test  
       file</rdf:li><rdf:li>InDesign</rdf:li><rdf:li>.pdf</rdf:li></rdf:Bag></dc:subject>  
11. </rdf:Description>
```

図 3.24 XMP に含まれる DublinCore メタデータ

1. prefix dc: <http://purl.org/dc/elements/1.1/>
2. PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3. select ?uri ?keyword
4. where {
5. ?uri dc:subject ?bag .
6. ?bag rdfs:member ?keyword .
7. filter regex(?keyword,'Blue','i').
8. }

図 3.25 SPARQL 文例

3.4.4 RDBToRDFContents

RDBToRDFContents は、前述した関係データベースから RDF コンテンツを作成する手法を実装したプログラムである。クラス図を図 3.26 に示す。RDBToRDFContents には二つのメソッドがある。一つ目の createRDFS メソッドは、関係データベースのスキーマ情報から、クラスやプロパティを抽出し、RDF スキーマを作成する。二つ目の createRDF メソッドは、作成した RDF スキーマを利用して、関係データベースのデータを RDF コンテンツに変換する。

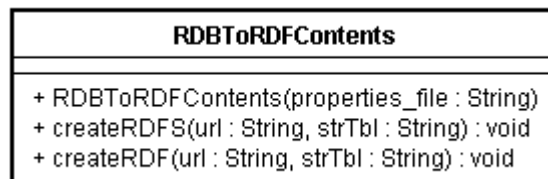


図 3.26 RDBToRDFContents のクラス図

createRDFS メソッドの概要を図 3.27 に示す。createRDFS メソッドは、スキーマ情報を取り出すために、JDBC が提供する Connection クラスの getMetadata メソッドを利用する。つぎに取り出したスキーマ情報（DatabaseMetaData クラスのオブジェクト）の getColumnns メソッドを利用して、属性の名前と定義域を取得する。テーブルの名前から

クラスを定義し、取り出した属性の名前と属性の定義域からプロパティを定義する。それから定義したクラスとプロパティを RDF スキーマに出力する。

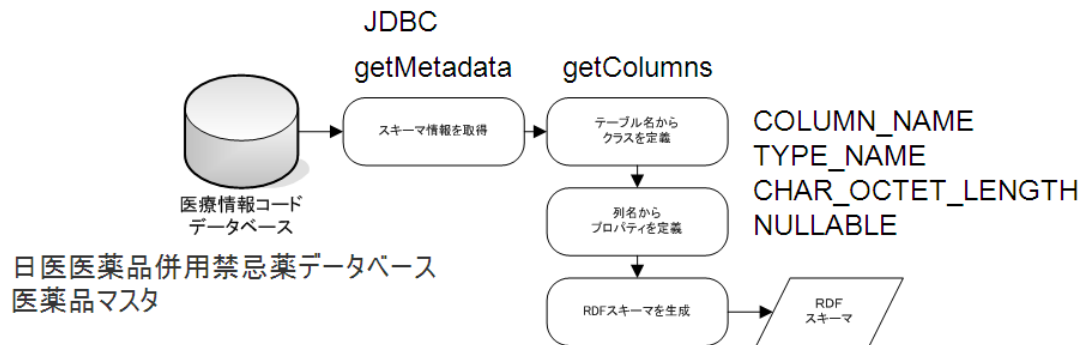


図 3.27 RDF スキーマを作成する

createRDF メソッドはデータベースのデータから RDF コンテンツを作成する(図 3.28). createRDF はデータベースに対して SELECT 文を発行しレコードを取得する。そして取得したレコードに対して URI を割り振る。そして、そのリソース URI と、先に定義した RDF プロパティとそして取得したレコードのデータから RDF コンテンツ を作成する。

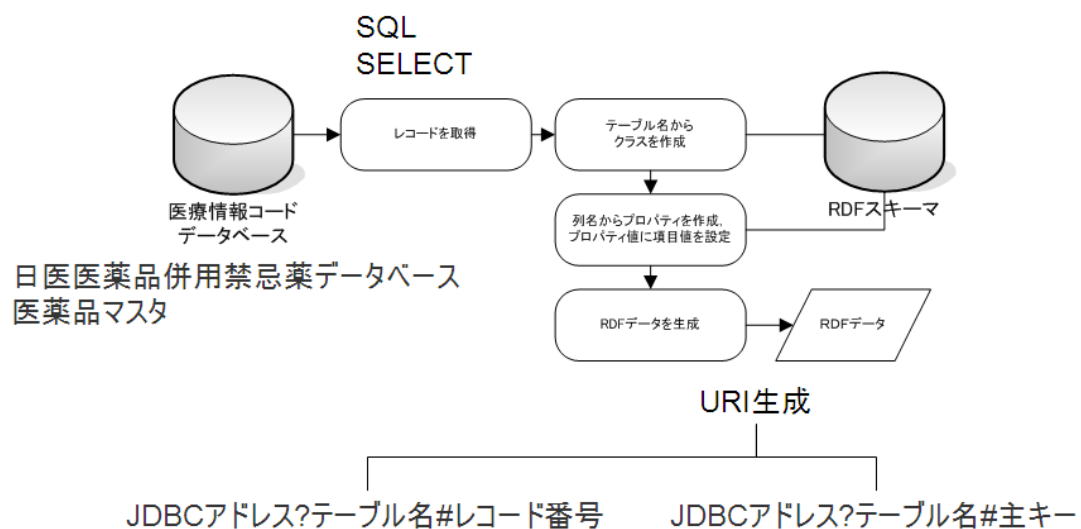


図 3.28 RDF コンテンツを作成する

本研究では、RDF コンテンツ化するデータベースとして、医薬品コードや医薬品名など

が格納された医薬品マスタ[75]，医薬品の併用禁忌情報を格納した併用禁忌データベース[76]を選択する．

医薬品マスタと併用禁忌データベースから RDF コンテンツを作成することで，医薬データを含んだ医療データに対して意味情報の付与に寄与することができる．

医薬品マスタのスキーマ構造を表 3.2，症状措置マスタと相互作用テーブルのスキーマ構造をそれぞれ表 3.3，表 3.4 に示す．これら医薬品マスタと併用禁忌データベースの二つは独立して構築されたが，併用禁忌データベースでは医薬品マスタの薬剤コードを参照している（表 3.2 を参照）．

表 3.2 医薬品マスタのスキーマ構造（一部省略）

属性名	型	主キー
modification_category	varchar(200)	
master_category	varchar(200)	
yakuzai_code	varchar(200)	○
省略		
interim_measure_date	varchar(200)	

表 3.3 症状措置機序マスタのスキーマ構造

属性名	型	主キー
symptom_code	varchar(200)	○
symptom	varchar(1000)	
action_mechanism	varchar(1000)	

表 3.4 相互作用テーブルのスキーマ構造

属性名	型	主キー
agent_code	varchar(200)	
object_agent_code	varchar(200)	
symptom_code	varchar(200)	

RDBToRDFContentsにより実際に作成した RDF コンテンツとその RDF スキーマを示す。図 3.29 は症状措置機序マスタから作成した RDF コンテンツである。症状措置機序マスタでは、`symptom_code` が主キーとなっている(表 3.3 参照)。したがって、図 3.29 の RDF コンテンツのリソース URI はその主キーの値とデータベース接続用 URI とテーブル名を利用した形になっている。この RDF コンテンツで利用している語彙は RDF スキーマに定義されている (図 3.30 参照)。

1. `<?xml version="1.0"?>`
2. `<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:symptom="jdbc:postgresql://localhost/normaldb?symptom#">`
3. `<rdf:Description
rdf:about="jdbc:postgresql://localhost/normaldb?symptom#S000193">`
4. `<symptom:action_mechanism>塩化スキサメニウムの血中カリウム増加
作用又はカテコールアミン放出が原因と考えられている。
</symptom:action_mechanism>`
5. `<symptom:symptom>両剤の併用により重篤な不整脈を起こすおそれがある。</symptom:symptom>`
6. `<symptom:symptom_code>S000193</symptom:symptom_code>`
7. `</rdf:Description>`
8. `</rdf:RDF>`

図 3.29 症状措置機序マスタから作成した RDF コンテンツ例

作成した RDF コンテンツは、関係データベースでの検索と同様に検索することができる。図 3.31 は、薬剤名「サクシン注射液 2% 2mL」の併用禁忌薬の薬剤名とその症状、そして発生メカニズムを検索する問い合わせ文例である。この問い合わせ文では三つのテーブルを統合して検索している。医薬品マスタと相互作用テーブル統合のための指示は図 3.31 の八行目と九行目、そして十行目と十一行目で記述されている。相互作用テーブルと症状措置機序マスタ統合のための指示は図 3.31 の十三行目と十四行目に記述されている。

```

1. <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
2. <rdfs:Class rdf:about="jdbc:postgresql://localhost/normaldb?symptom#SYMPTOM">
3. <rdfs:comment>symptom</rdfs:comment>
4. </rdfs:Class>
5. <rdf:Property rdf:about="jdbc:postgresql://localhost/normaldb?symptom#symptom">
6. <rdfs:domain
   rdf:resource="jdbc:postgresql://localhost/normaldb?symptom#SYMPTOM"/>
7. <rdfs:comment>symptom varchar(1000)</rdfs:comment>
8. </rdf:Property>
9. <rdf:Property
   rdf:about="jdbc:postgresql://localhost/normaldb?symptom#symptom_code">
10. <rdfs:domain
   rdf:resource="jdbc:postgresql://localhost/normaldb?symptom#SYMPTOM"/>
11. <rdfs:comment>symptom_code varchar(200)</rdfs:comment>
12. </rdf:Property>
13. <rdf:Property
   rdf:about="jdbc:postgresql://localhost/normaldb?symptom#action_mechanism">
14. <rdfs:domain
   rdf:resource="jdbc:postgresql://localhost/normaldb?symptom#SYMPTOM"/>
15. <rdfs:comment>action_mechanism varchar(1000)</rdfs:comment>
16. </rdf:Property>
17. </rdf:RDF>

```

図 3.30 症状措置機序マスタから作成した RDF スキーマ

```

1. PREFIX yakuzaiMaster:<jdbc:postgresql://localhost/normaldb?yakuzaimaster#>
2. PREFIX interaction:<jdbc:postgresql://localhost/normaldb?interaction#>
3. PREFIX symptom: <jdbc:postgresql://localhost/normaldb?symptom#>
4. select ?name ?object_agent_kanji ?symptom ?action_mechanism
5. WHERE {
6.   ?y yakuzaiMaster:kanji ?name .
7.   filter regex(?name,'サクシン注射液2% 2mL','i').
8.   ?y yakuzaiMaster:yakuzai_code ?yakuzai_code .
9.   ?i interaction:agent_code ?yakuzai_code .
10.  ?i interaction:object_agent_code ?object_agent_code .
11.  ?y2 yakuzaiMaster:yakuzai_code ?object_agent_code .
12.  ?y2 yakuzaiMaster:kanji ?object_agent_kanji .
13.  ?i interaction:symptom_code ?symptom_code .
14.  ?s symptom:symptom_code ?symptom_code .
15.  ?s symptom:symptom ?symptom .
16.  ?s symptom:action_mechanism ?action_mechanism .
17. }

```

図 3.31 ある薬剤の併用禁忌薬を検索する SPARQL 文例

3.4.5 GRDDLPlugin

GRDDLPlugin は前述した GRDDL を実装したプログラムである．クラス図を図 3.22 に示す．

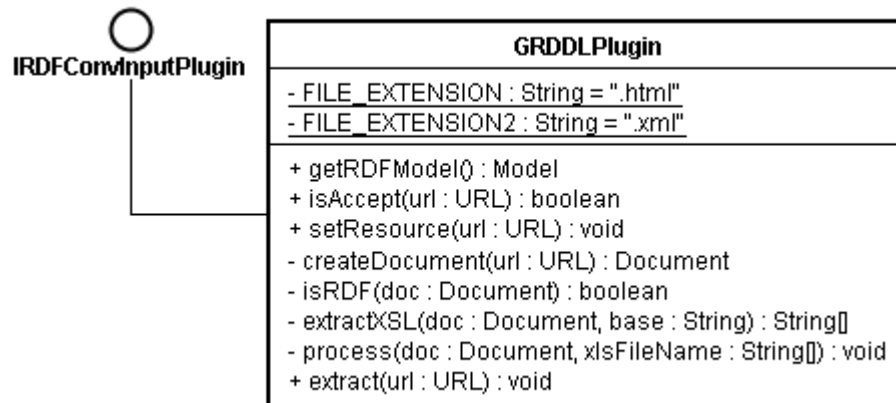


図 3.32 GRDDLPlug-in のクラス図

GRDDLPlugin クラスは **extract** メソッドに指定された XML ファイルの URL から，RDF コンテンツの抽出を行う．**extract** メソッドは，**createDocument** メソッド，**extractXSL** メソッド，**process** メソッドを呼び出す．**createDocument** メソッドは，XML ファイルの解析を行う．**extractXSL** メソッドは GRDDL の処理の中核となる XSL ファイルの抽出処理を担う．最後の **process** メソッドは抽出した XSL ファイルを用いて XML ファイルの変換処理を行う．

なお，GRDDLPlugin クラスは IRDFConvInputPlugin インタフェースを実装しているため，拡張可能なメタデータ抽出機構 **Pluggable Metadata Extractor** のプラグインとしても利用可能である．**Pluggable Metadata Extractor** については後で説明する．

本研究では，GRDDL Test Cases[77] が提供するテストケースを利用して GRDDLPlugin の性能評価を実施した．表 3.5 に GRDDLPlugin のテスト結果を示す．

表 3.5 GRDDL Test Cases の実験結果

Normative Tests	テスト結果
Localized Tests	12/12
Namespace Documents and Absolute Locations	10/12
Library Tests	4/13
Ambiguous Infosets, Representations, and Traversals	4/30

各テストの内容を説明する。Localized Tests は名前空間と絶対 URI を使わない単純なテストケースである。Namespace Documents and Absolute Locations は名前空間と絶対 URI を含めたテストケースでネットワークへの接続が必須となる。Library Tests は GRDDL Standard Transform Library[78]を利用したテストケースである。最後に Ambiguous Infosets, Representations, and Traversals は特殊なケースを集めた堅牢性を測るためのテストケースである。最後に P3P work-alike テスト[79]の結果を図 3.33 に示す。

```

1. <rdf:RDF
2.   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
3.   xmlns:j.0="http://www.w3.org/2002/01/">
4. <rdf:Description rdf:about="">
5. <j.0:P3Pv1expiry rdf:parseType="Resource">
6. <j.0:P3Pv1max-age>604800</j.0:P3Pv1max-age>
7. </j.0:P3Pv1expiry>
8. </rdf:Description>
9. <j.0:P3Pv1Policy rdf:ID="public">
10. <j.0:P3Pv1discuri rdf:resource="http://www.w3.org/Consortium/Legal/privacy-
    statement#Public"/>
11. </j.0:P3Pv1Policy>
12. </rdf:RDF>

```

図 3.33 GRDDLPlug-in が抽出した RDF コンテンツ

3.4.6 QuestionnaireEditor

QuestionnaireEditor は、前述した質問票メタスキーマに定義された回答形式を利用した質問票スキーマ作成用プログラムである。図 3.34 に **QuestionnaireEditor** の画面写真を示す。**QuestionnaireEditor** では、六つの回答形式をベースにした質問票スキーマの設計ができる。図 3.34 の左側は質問票の階層構造を表したツリービュー、画面の右側は設計している質問票のプレビュー画面である。

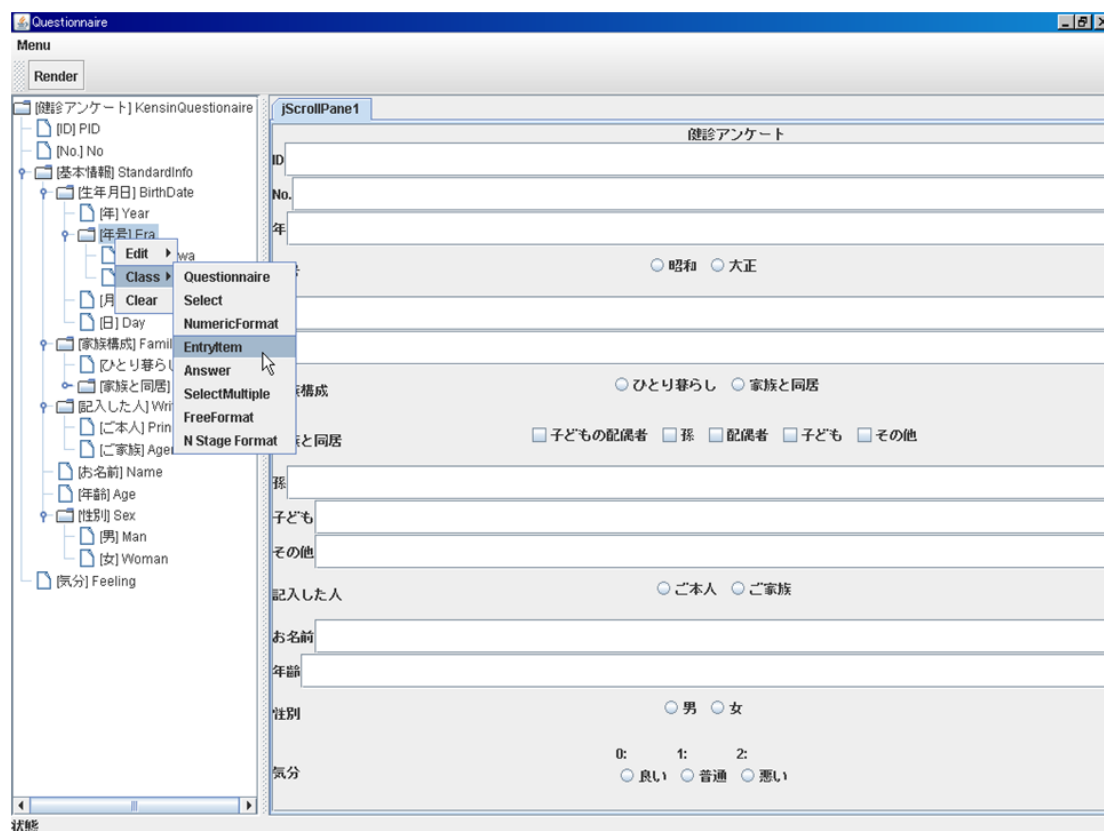


図 3.34 QuestionnaireEditor の画面写真

図 3.35 は **QuestionnaireEditor** により作成された質問票スキーマ例を示している. この質問票スキーマには「健康診断アンケート」が定義されており, 質問項目として「**StandardInfo**», 「**No**», 「**PID**」が定義されている. 質問項目「**No**」の定義例を図 3.36 に示す. この「**No**」は **NumericFormat** クラスを参照しており, 数字回答形式として定義されている. **QuestionnaireEditor** により作成された質問票スキーマは **RDF** コンテンツとして保存される. 保存された質問票スキーマは, 次の質問票作成時や同種の質問票を作

成するときに再利用することができる。図 3.37 は単一回答形式の質問項目で性別のタイトルをもつ質問項目を検索するための問い合わせ文である。このようにして RDF コンテンツとして保存された質問票スキーマは自由に検索可能である。

また、QuestionnaireEditor は作成した質問票データの入力画面としても利用することができる。入力された質問票データは QuestionnaireEditor で作成した質問票スキーマを使用した RDF コンテンツとして保存できる。

```
1. <rdf:Description rdf:about="urn:QuestionnaireTest">
2. <qem:hasEntry rdf:resource="urn:StandardInfo"/>
3. <qem:hasEntry rdf:resource="urn:No"/>
4. <qem:hasEntry rdf:resource="urn:PID"/>
5. <dc:creator>山田 太郎</dc:creator>
6. <dc:date>1167996171712</dc:date>
7. <dc:description>健康診断アンケート</dc:description>
8. <dc:title>健康診断アンケート</dc:title>
9. <rdf:type rdf:resource="http://www.sociotech-
   lab.jp/schema/Questionnaire.rdfs#Questionnaire"/>
10. <rdf:Description>
```

図 3.35 質問票の定義例

```
1. <rdf:Description rdf:about="urn:No">
2. <dc:description>アンケート毎に振り分けられる番号
   </dc:description>
3. <dc:title>No,</dc:title>
4. <rdf:type rdf:resource="http://www.sociotech-
   lab.jp/schema/Questionnaire.rdfs#NumericFormat"/>
5. <rdf:Description>
```

図 3.36 数字回答形式の定義例

```

1. prefix dc:<http://purl.org/dc/elements/1.1/>
2. prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
3. prefix qem:<http://www.sociotech-lab.jp/schema/Questionnaire.rdfs#>
4. prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>
5. select *
6. where {
7.   ?entry rdf:type qem:SelectFormat .
8.   ?entry dc:title ?title .
9.   ?entry dc:description ?desc .
10. filter regex(?title,'性別','i').
11. ?entry qem:hasEntry ?childentry .
12. ?childentry rdfs:member ?entries .
13. ?entries dc:title ?entriesTitle .
14. }

```

図 3.37 質問票スキーマの検索

3.5 考察とまとめ

3.5.1 DICOM2RDF の考察とまとめ

一般的な PACS では DICOM 画像のデータ要素をキーに画像を検索する。DICOM2RDF はその DICOM 画像のデータ要素から RDF コンテンツを作成する。その RDF コンテンツを利用することで、PACS と同様の画像検索を実現できる。

これまでも医療用画像データの検索に RDF コンテンツは利用されてきた[67,80]。文献[67]では、インターネットでの利用を想定した TCP/IP ネットワーク上の医療用画像データ検索システム、MediSeek を提案している。MediSeek は JPEG 画像形式のヘッダ部分に RDF コンテンツを埋め込むことで、医療用画像の検索精度を向上させている。埋め込まれる RDF コンテンツは、医療従事者が Descriptor と呼ばれるシステムを用いて作成する。しかしながら、JPEG 画像形式のヘッダ部分に RDF コンテンツを直接埋め込むため、ファイル形式に大きく依存するという問題がある。そのため、医療用画像形式の標準である DICOM 画像形式に対応できていない。

一方、文献[80]では RDF コンテンツと医療用画像データを関係付けることで、医師が記述した症例データの共有支援を行う情報システムの提案を行っている。この研究では、読影レポート文書を RDF コンテンツとして表現し、DICOM 画像形式データと関係付ける。RDF コンテンツ化された読影レポートを検索することで、医療用画像に関係した症例の共

有支援を実現している．文献[80]では RDF コンテンツの作成に DICOM のデータ要素の一部を利用しているが，主に医学知識の蓄積に注目している．また，医用画像が保存された環境として PACS が導入されていることを想定している．

本研究の DICOM2RDF ではハードディスクドライブに保存された DICOM 画像群を検索可能にすることを目的としている．そして DICOM2RDF が作成した RDF コンテンツは，DICOM 画像の基本情報を表現している．したがって，作成した RDF コンテンツに対して文献[80]で示された医学知識を関係付ける手法を適用することも可能である．

3.5.2 Excel2RDF の考察とまとめ

Excel2RDF はバイナリ形式のワークブックから日本語と複数ワークシートに対応した RDF コンテンツを作成する．作成した RDF コンテンツは問い合わせ言語 SPARQL を用いることでデータベースと同様の検索が可能である．

これまでも Excel ファイルから RDF コンテンツを作成する取り組みは行われてきた [11~13]．

文献[11]では CVS 形式やワークブックに対してマッピングファイルを作成することで，RDF コンテンツの作成を行っている．

文献[12]では，Babel[81]と呼ばれる Web サービスを開発している．Babel はワークブックを含む複数のファイル形式から自動的に RDF コンテンツを作成する．しかしながら，これらの情報システムには日本語が扱えないことや，複数のワークシートに対応していないため，機能面に不備がある．

文献[13]では GRDDL[44]と Microformats[82]を利用した RDF コンテンツの作成を提案している．MS Excel は XML 形式でワークブックを出力することができる．あらかじめワークブックに Microformats によるアノテーションを付与することで，GRDDL による RDF コンテンツの作成を可能にする．しかしながら，XML 形式でワークブックを出力する必要があることと，あらかじめアノテーションを付与しなければならない．

3.5.3 ReadXMPFromFile の考察とまとめ

ReadXMPFromFile は XMP が埋め込まれたファイル形式から RDF コンテンツを抽出する. 本研究では文献[62]を参考にして ReadXMPFromFile の XMP 抽出機能を実装した. また, ReadXMPFromFile により抽出された RDF コンテンツの検索例を示した.

XMP はファイルのヘッダ部に RDF コンテンツを埋め込むため, 対応可能なファイル形式は限られる. しかしながら, XMP の標準化が進めば, ファイルに対して自由にメタデータを付与することが可能となり, それらメタデータを利用したアプリケーションの登場が期待される.

3.5.4 RDBToRDFContents の考察とまとめ

RDBToRDFContents は関係データベースのスキーマ情報から, RDF コンテンツを作成する. これまでにも, セマンティック Web からデータベースを処理可能にする取り組みは行われてきた[83,84].

文献[83]では RDF と関係データベースの構造をマッピングすることで, セマンティック Web からのデータアクセスを可能にしている.

文献[84]では, RDF コンテンツを検索する独自の言語を SQL 言語に変換することで, データアクセスを可能にしている.

文献[83]の手法では, あらかじめ RDF と関係データベースの関係を記述したマッピング言語を予め定義する必要がある, 手間がかかる. また文献[84]の手法では, 独自の検索言語を用いているため汎用的ではない. その点, 本手法はデータベースの内容を RDF コンテンツに変換する. したがって, RDF コンテンツに変換した後は, SPARQL に代表されるセマンティック Web 技術を利用してデータアクセスが可能である.

ただし, 本手法には RDF コンテンツ変換後にデータの更新要求が発生した場合は, RDF コンテンツを直接更新する必要性が生じるという課題が残る.

現在の SPARQL には, SELECT, CONSTRUCT, DESCRIBE,そして ASK 構文しか用意されておらず, データの操作ができない. したがって, 現状ではデータの更新するプログラムを用意する必要がある. しかしながら, SPARQL にデータ操作機能を追加する SPARQL Update[85]の提案が行われてきており, 今後この課題は解決されることが期待される.

3.5.5 GRDDLPlugin の考察とまとめ

GRDDL は XML ファイルから RDF コンテンツを作成するための標準化された手法である。W3C による標準化により、今後 GRDDL を利用した XML ファイルが増えてくることが想定される。

GRDDLPlugin はこの GRDDL の機能を実装したプログラムである。GRDDLPlugin は GRDDL の基本機能を実装することができた。

しかし、GRDDLPlugin には W3C が提供する GRDDL Test Cases[77]の全てのテストをまだパスできていないという課題が残る。特に、Library Tests と Ambiguous Infosets, Representations, and Traversals のテスト結果は悪い。この結果から GRDDLPlugin はまだ実用に耐えうる性能を持っていないことが示された。

ただし、基本的なテストケースである Localized Tests と Namespace Documents and Absolute Locations はクリアしているため、実験システムとしては利用可能である。

3.5.6 QuestionnaireEditor の考察とまとめ

質問票調査では、その目的に応じた設問を工夫することになる。一方、継続的な調査研究や他の調査との比較研究を考慮した場合、調査項目などの標準化が望まれる。

一般に医療情報システムにおける用語体系とデータ構造の重要性は以前から議論されており、標準化活動が続けられている。このような標準化された用語体系として ICD[86] や SNOMED-CT[87]がある。そして標準化されたデータ構造として HL7[88], MML[89], そして DICOM [55]がある。さらに UMLS[90]では標準化された医学用語を統一的に扱うために、ICD や SNOMED-CT などの用語集の統合作業を進めている。

このような用語体系やデータ構造の標準化は統一された概念表記やデータ表記を可能にし、医療情報の交換や共有を促す。しかしながら標準化された用語体系およびデータ構造も利用分野や利用形態によっては表現力が不足することがある。たとえば、利用形態による表現力不足の問題が指摘され、その問題に対して UMLS で提供される標準と病院などの施設独自の用語体系を組み合わせた医学用語交換手法が提案されている[91]。また健診情報のみに適用分野を限定したデータ共有のための標準化の試みも行われているが、医療機関ごとに用語の意味や表現の違いが存在し、標準化の妨げとなっている[92]。また保険医療統計データを XML で定義し、Web 上で公開することにより共有しようという提案もある[93]。

疫学調査を情報システムによって支援する試みとして米国疾患予防管理センターが開発し、提供している Epi Info[94]がある。このシステムではデータ入力画面の開発から、データ入力、簡易言語によるデータ分析、グラフ描画、レポート作成を支援する。そしてデータは関係データベースで管理される。

一方、質問票調査を支援する情報システムの試みとして、坂本らは大学の授業評価を対象に階層的な管理機能を持った質問票調査実施支援システムを提案している[95]。そこでは質問票データの所有権を階層化して管理することで、質問票のきめ細かい管理を容易にしている。Tornqvist らは質問票調査のデータ構造を XML と DTD (Document TypeDefinition) で表現し、入力データの妥当性の確認や、スクリプトを利用した枝分かれ質問項目のナビゲーションを考慮したシステムを提案している[96]。また楯は、選択式と記述式の質問票構造を DTD として定義し、この DTD に従った妥当な XML データを質問票調査に利用することで、質問票のデータ構造の柔軟性と汎用性を実現した[97]。ただし楯は DTD の汎用性とデータの表現能力にはトレードオフがあることを指摘している。すなわち汎用的な DTD はデータの仕様変更に対して耐性がある反面、データの表現能力が低い。逆にデータを詳細に表現した DTD は、データの仕様変更に対する耐性が低い。

質問票調査を支援する情報システムでは、データ構造の柔軟性も必要であるが、データ処理の柔軟性が求められる。このような観点から田村らは、質問票調査ごとに異なる集計方法を可能とするために、集計機能をプラグインとして追加できる質問票調査支援システムを提案している[98]。三船らは、質問票の収集、分析の処理能力を向上するために、P2P モデルに則り分散化したアンケート収集モデルを提案している[99]。久保らは、調査項目の共有を考慮した光学式マーク読み取りシステムと XForm 技術[100]を利用した分散型質問票調査支援システム SQS (Shared Questionnaire System) を提案している[101]。そこでは、XForm と Dublin Core を組み合わせて質問票調査項目を定義し、意味に基づく質問票調査項目の再利用と共有を実現している。またオープンソフトウェアを利用することにより、調査費用の抑制に成功している。

質問票調査を支援する情報システムに対して、プライバシーの保護も重要な機能として要求される[102~104]。横川らは質問票の回答の改ざん、捏造を防止しながら、匿名通信を提供する情報システムを提案している。そこでは認証処理と集計処理を分離し、集計処理では個人が特定できない仕組みを提供することでプライバシーの保護を実現している[102]。中里らは、電子投票プロトコルを質問票調査支援システムに適用することで、暗号化したデータを収集し、集計する方式を実現している[103]。また北川らも同様に、電子投票プロトコルを適用した大学の授業評価システムを構築している[104]。

QuestionnaireEditor では、データの意味表現を得意とする RDF スキーマと RDF を用いて質問票の調査項目を定義している。ここで RDF スキーマにはオブジェクト指向におけるクラス概念が取り込まれており、データ構造の汎用性と表現能力の両立が可能である。

3.6 おわりに

我々が普段から利用している情報源の殆どはまだ RDF に対応していない。セマンティック Web は”Web of Data“の構築を目指した取り組みであることは、先に述べた。セマンティック Web では、URI で識別可能なリソースのメタデータを RDF で記述する。RDF は W3C のセマンティック Web 技術における基礎技術として位置づけられている[3]。 ”Web of Data” は RDF コンテンツの集合といえる。つまり、セマンティック Web を実現するには、はじめに、この RDF コンテンツを作成する必要がある。

これまでに、RDF コンテンツを作成するために様々な取り組みが行われてきた[11~16]。

文献[14]では XHTML に埋め込まれた microformats や eRDF[106]を RDFa[107]に変換することで、XHTML に埋め込まれた RDF コンテンツの作成を容易にする hGRDDL を提案している。

文献[15]では Amazon や Yahoo!などが公開している REST サービスのデータを SPARQL から操作可能にしている。言い換えると、REST サービスのデータを RDF コンテンツ化している。

文献[16]では Web ブラウザ用セマンティック Web プラグイン Piggy Bank を提案している。Piggy Bank はページにタグを付与したり、XHTML に埋め込まれた RDF コンテンツを収集したり、編集したりすることで、RDF コンテンツの作成を支援する。

一方、SIMILE プロジェクトの一つ RDFizer[108]では Web サービスによる RDF 形式への変換ツールを公開している。また、ESW Wiki では RDF 形式への変換ツールのリンク集を公開している[109]。

本章ではユーザ要件に基づく情報統合環境のための準備として、パーソナルコンピュータに保存された各種ファイル形式やデータソースから RDF コンテンツを作成する手法を示した。特に DICOM 画像ファイル、Microsoft Excel ファイル、Adobe PDF ファイル、そして XML 形式ファイルから RDF コンテンツを作成できることを示した。

また、関係データベースから RDF コンテンツを作成する手段についても述べた。これ

らは RDF に対応していない情報源から RDF コンテンツを作成するための手法であった。

質問票調査を支援する情報システムでは、設計当初から RDF に対応した情報システムの例を示し、そこで作成された RDF コンテンツを示した。RDF コンテンツの作成はセマンティック Web の成立に関わる大きな課題である。

本研究が示した RDF コンテンツ作成手法はその課題の解決の一助になりうる。また、XMP と GRDDL 自体は本研究のアイデアではないが、標準的な RDF コンテンツ作成手法はユーザ要件に基づく情報統合環境に必要である。

第4章

ユーザ要件に基づく 情報統合環境の構築

4.1 はじめに

本研究の目的は、パーソナルコンピュータ上にユーザ要件に基づく情報統合環境を構築することである。パーソナルコンピュータ環境に保存されていたファイルは、そのファイルをアプリケーションで読み込むまで中身のデータを確認することができない。情報システムがこのファイルの中身を把握する事ができるのなら、ファイル一つを提示するのにも様々な手段を取ることができる。また、保存されたファイルの関係性を利用した情報処理も可能になる。

第3章では、これらの環境を実現するユーザ要件に基づく情報統合環境の準備として、電子海産期間教に保存されていたファイルから RDF コンテンツを作成した。本章では、それら RDF コンテンツ作成手法を統合する拡張可能なメタデータ抽出機構 **Pluggable Metadata Extractor** と、ユーザ要件に基づく情報統合環境の基礎を成す **RDFView** について述べる。

Pluggable Metadata Extractor は第3章で述べ RDF コンテンツ作成手法を統合することを目的に設計された。ユーザ要件に基づく情報統合環境はパーソナルコンピュータ上に構築されたセマンティック Web の上に成り立つ。

すでに第2章でも述べたがセマンティック Web において RDF コンテンツはその基礎を成す重要な要素である。理想はすべてのファイル群が RDF コンテンツ化されていることである。しかしながら、現実的にそれを実現するのは難しい。**Pluggable Metadata Extractor** はプラグインにより RDF コンテンツ作成機能の拡張が可能な情報システムである。

RDFView は本研究において重要な役割を担う情報システムである。ユーザ要件に基づく情報統合環境では、コンピュータ上に保存されたファイルコンテンツをユーザがどのように表示して欲しいのかをユーザ要件とし、そのユーザ要件に基づいてそれらファイルコンテンツの統合を行う。**RDFView** はこの関係データベースのビューテーブルに相当する

機能を実現する。

RDFView では問い合わせ文をテンプレートとして保管しておく。そして、RDFView はそれらのテンプレートを REST(Representational State Transfer)形式の Web サービスに変換し、公開する。REST 形式は文献[109]の HTTP 本来の原則に従った REST full と XML と HTTP を利用した簡易な RPC としての REST に分けられる。RDFView ではテンプレートの管理を REST full, RDFView が公開する Web サービスには簡易な RPC としての REST を使用する。

4.2 ユーザ要件に基づく情報統合環境の設計

本節ではユーザ要件に基づく情報統合環境の設計手法について述べる。ユーザ要件に基づく情報統合環境では、ユーザの要求に応じて情報の統合を行う。本研究では、ユーザの幅広い要求に耐えうるよう情報システムの設計を行った。

4.2.1 Pluggable Metadata Extractor の設計

ユーザ要件に基づく情報統合環境を実現する上で、パーソナルコンピュータ環境に保存されているファイル群に対して、統一したアクセス手法を提供することが求められる。情報統合では、複数の異なる情報源(Source)に対し、統一的なアクセス手法(Wrapper)を提供することで、それらの情報源を統合する(Mediator)。情報統合において、統一的なアクセス手法は重要な要素である。

第 3 章で述べた RDF コンテンツの作成手法は、情報統合では Wrapper が担当する機能である。本研究における Pluggable Metadata Extractor は、それら Wrapper としての RDF コンテンツ作成手法を統合する。さらに、Wrapper 部分はプラグイン機構により拡張可能である。図 4.1 に Pluggable Metadata Extractor のシステム概要図を示す。

Pluggable Metadata Extractor では URL またはファイルをアップロードすることで RDF コンテンツを作成する。アップロード機能を用意している理由は、URL が利用できない環境も想定したからである。そして URL またはアップロードされたファイルの名前から、RDF コンテンツに変換するプラグインを検索する。

Pluggable Metadata Extractor はプラグインを発見後、そのプラグインの RDF コンテンツ変換機構を実行する。適合するプラグインが無い場合は、ファイルの名前、登録時間情報を含んだ基本的な RDF コンテンツを作成する。

最終的に作成された RDF コンテンツをクライアントに送信する。その際、クライアントから明示的に RDF コンテンツのリソース URI の指定がある場合は、その URI を RDF コンテンツに埋め込む。

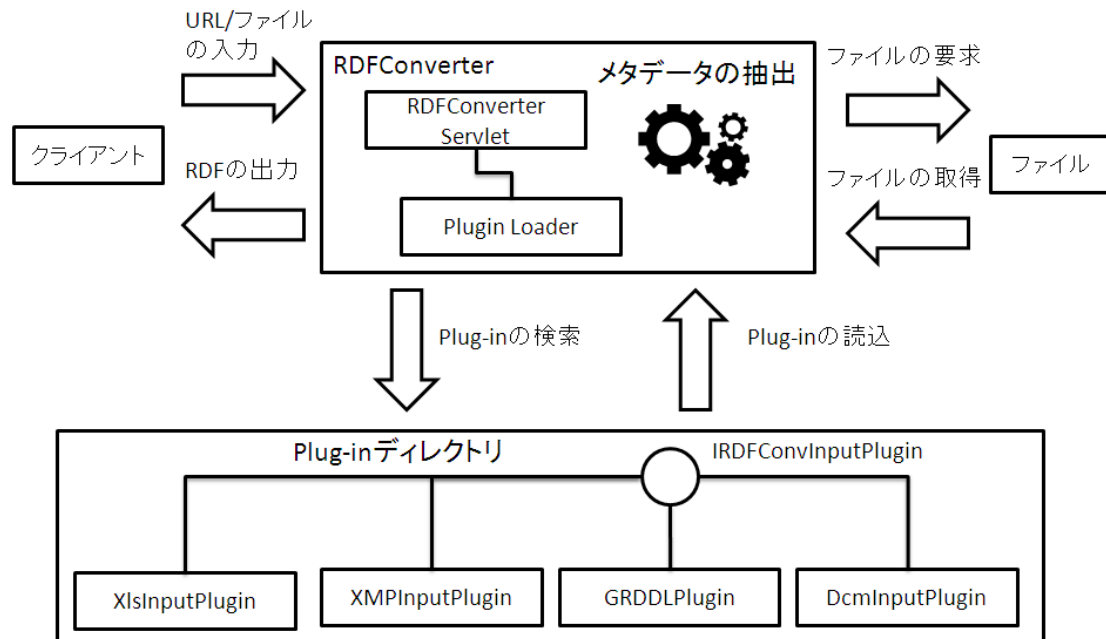


図 4.1 Pluggable Metadata Extractor システム概要

4.2.2 RDFView の設計

セマンティック Web では RDF コンテンツにアクセスする手法として問い合わせ言語 SPARQL が提供されている。この SPARQL はトリプルの条件、つまり、どの語彙でトリプルが構成されているかを指定することで、この条件に適合した RDF コンテンツの検索を行う。このため SPARQL による問い合わせ文を記述するには、あらかじめ検索対象の RDF コンテンツにどの語彙が用いられているかを知らなければならない。つまり、RDF コンテンツの内容を把握している人でない限り、問い合わせ文を記述するのは難しい。

RDFView は SPARQL による問い合わせ文を記述することなく、セマンティック Web の検索を実現する情報システムである。RDFView では、SPARQL により記述された問い合わせ文をテンプレートとして保管する。RDFView はそのテンプレートから REST インタフェースを生成する。つまり、その生成された REST インタフェースを利用すれば、SPARQL による問い合わせ文を記述する必要なく、RDF コンテンツの検索が実現できる。

REST インタフェース生成に用いられるテンプレートは, RDF コンテンツの内容を把握している人が作成する. 図 4.2 は単純なテンプレートの例である. テンプレートは SPARQL による問い合わせ文に外部変数定義を加えたものである. このテンプレートにタイトルとラベルを付与する. RDFView ではこれをビューと呼ぶ. ビューは RDF コンテンツを取得するための関数と考えると理解しやすい.

```
1. PREFIX rdfs: <http://www.w3.org/2000/01/rdf-  
   schema#>  
2. select *  
3. where {  
4.   ?s rdfs:label"$o" .  
5. }
```

図 4.2 問い合わせ文テンプレート

関数を呼び出すには, リクエスト URL を作成して RDFView に GET メソッドを発行する. Z 図 4.2 のテンプレートが保存されたビューを取得するリクエスト URL 例を以下に示す. このビューには sample というラベルが付与されている.

`http://.../MedSW/servlet/RDFView?label=sample&o=サンプル`

上記のリクエスト URL を受け取った RDFView はラベル sample で識別されるテンプレートを呼び出し, 外部変数に値を代入する. ここでは文字列「サンプル」が代入される. この時点で問い合わせ文が完成する. RDFView はこの問い合わせ文を実行し, その結果を返す.

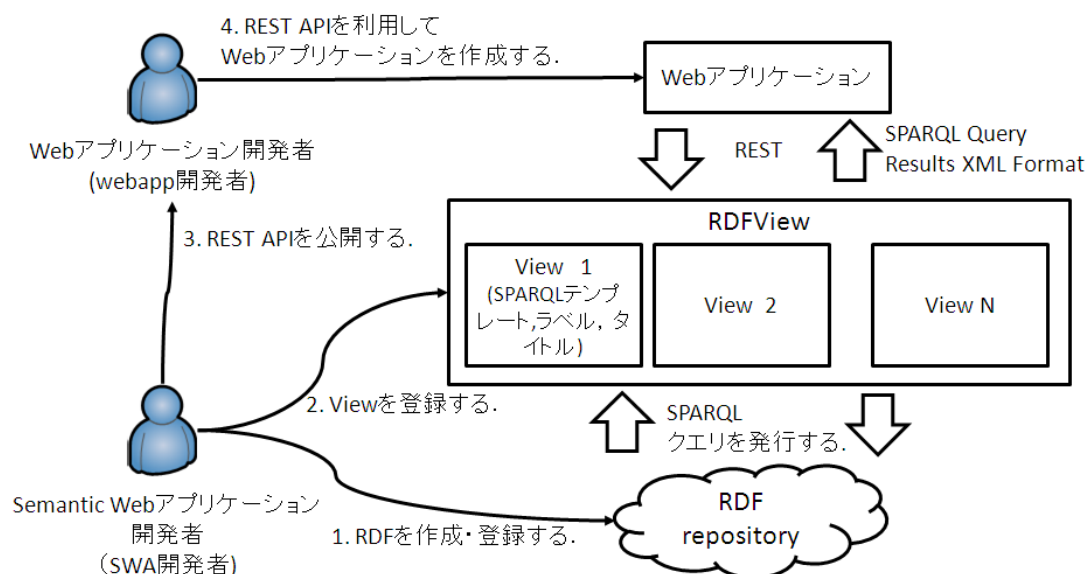


図 4.3 RDFView の概要

図 4.3 に RDFView の概要図を示す。図では RDF コンテンツの内容を把握している人間を SWA(Semantic Web Application)開発者と表記し、REST インタフェースの利用者を Webapp(Web Application)開発者と表記する。

RDFView はビューの取得機能,ビューの登録機能,ビューの削除機能を REST full インタフェースで提供する。図 4.3 を用いて RDFView の利用形態を説明する。

まず、SWA 開発者は RDF コンテンツを RDF repository に登録する(図 4.3 の 1)。つぎに、登録された RDF データのための問い合わせ文をテンプレートとして記述する。そのテンプレートをビューとして RDFView に登録する(図 4.3 の 2)。ビューには問い合わせ文のテンプレートとそのタイトル、そして識別用のラベルが付与される。

つぎに RDFView は登録されたビューにしたがって REST インタフェースを作成する。具体的には、テンプレートには外部変数が定義されており、その外部変数から REST インタフェースを作成する。SWA 開発者は生成された REST インタフェースを公開する(図 4.3 の 3)。

つぎに webapp 開発者はそのビューを実行するために、リクエスト URL を組み立て、HTTP メソッドを発行する(図 4.3 の 4)。

HTTP メソッドを受信した RDFView は指定されたビューからテンプレートを読み込む。そしてリクエストパラメータ変数をテンプレートに埋め込み、問い合わせを実行する。

RDFView は RDF コンテンツを検索後、検索結果を XML 形式で出力する。

最後に、Webapp 開発者は受け取った検索結果を用いて新たなアプリケーションを構築する。

このように RDFView ではセマンティック Web を利用できないユーザの要求に応じて、セマンティック Web を利用できるユーザがセマンティック Web の統合条件をビューとして定式化する。そして、RDFView はそのビューを REST 形式の Web サービスとしてビューを公開する。その REST 形式の Web サービスはセマンティック Web を利用できないユーザでも利用することができる。

4.3 ユーザ要件に基づく情報統合環境の実装

4.3.1 Pluggable Metadata Extractor の実装

Pluggable Metadata Extractor は RDF コンテンツ作成機能をプラグイン機構により拡張可能な Web サービスである。本研究では Pluggable Metadata Extractor を Java サーブレットとして実装した。

Pluggable Metadata Extractor には二通りの RDF コンテンツ作成手法がある。変換対象ファイルの URL から RDF コンテンツを作成する手法とアップロードされた変換対象ファイルから RDF コンテンツを作成する手法である。

前者は既に Web 環境上に公開されており、URL による識別可能なファイルから RDF コンテンツを作成する場合に用いる。この場合、Pluggable Metadata Extractor は指定された URL からファイルをダウンロードしてから RDF コンテンツを作成する。後者は、URL で識別不能なファイルから RDF コンテンツを作成する場合に用いる。図 4.4 に Pluggable Metadata Extractor のクラス図を示す。

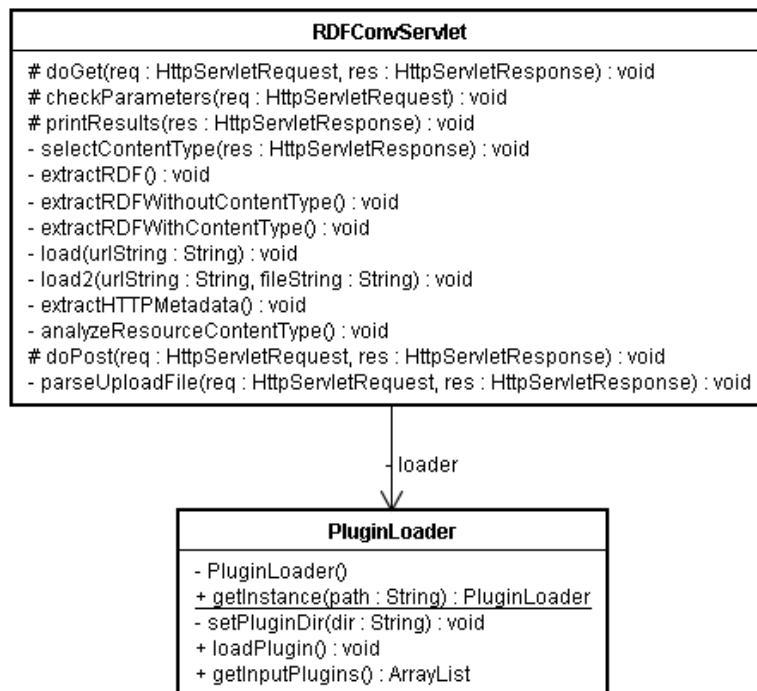


図 4.4 Pluggable Metadata Extractor のクラス図

まず始めに、Pluggable Metadata Extractor はファイルの種類に応じて RDF コンテンツを作成するプラグインを選択する。ファイルの種類は HTTP の Content-Type ヘッダとファイルの拡張子の両者を用いて特定する。ファイルの種類を特定後、対応したプラグインを Plug-in Loader で呼び出す。次に、呼び出したプラグインに対象ファイルの URL を渡す。プラグインは受け取った URL で識別されるファイルから RDF コンテンツを作成する。最後に、プラグインが作成した RDF コンテンツを表示する。

Pluggable Metadata Extractor に対応したプラグインを作成するには、インタフェース IRDFConvInputPlugin を実装した Java クラスを作成する。

このインタフェースには四つのメソッド、extract, setResource, isAccept そして getRDFModel が定義されている。extract メソッドは指定された URL からファイルを取得し、RDF コンテンツを作成する。次の setResource メソッドは RDF コンテンツにリソース URI を指定する場合に用いる。isAccept メソッドは指定された URL が変換対象であるかを判定する。最後の getRDFModel は変換した RDF コンテンツを取得するためのメ

ソッドである．図 4.5 に Pluggable Metadata Extractor に対応した RDF コンテンツ作成するクラス図を示す．

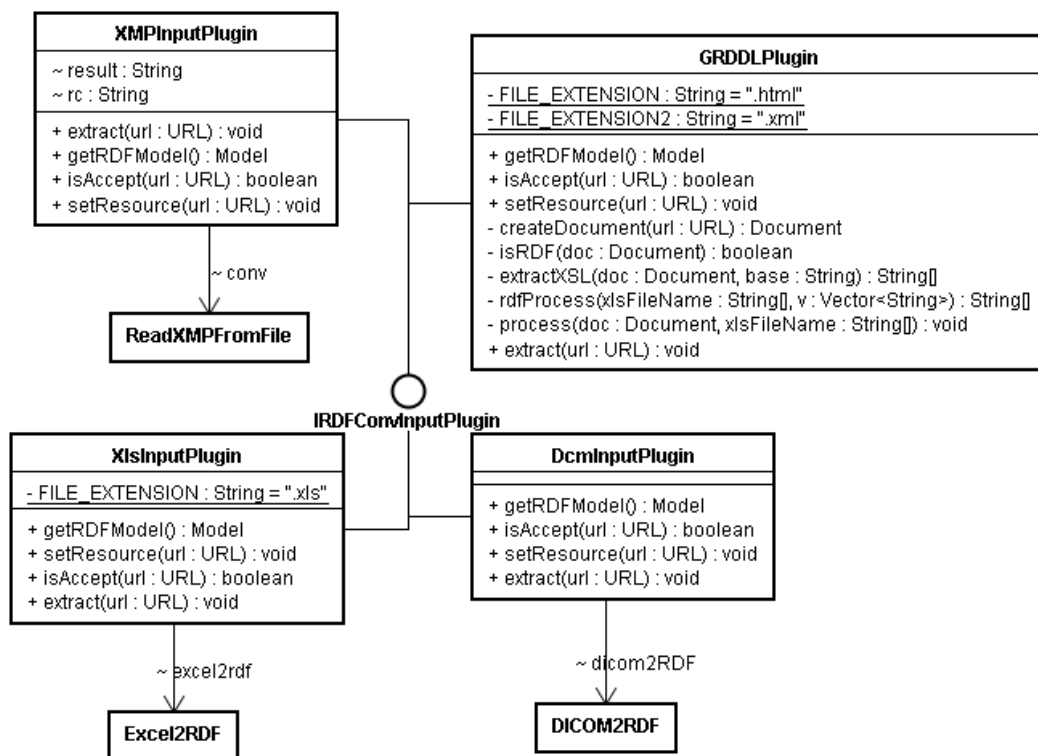


図 4.5 各 Plug-in のクラス図

4.3.2 RDFView の実装

本研究では，RDFView は Java サーブレットとして実装した．図 4.6 に RDFView のシステム概要図を示す．RDFView はセマンティック Web エンジンに Jena ライブラリをテンプレートエンジンに Apache Velocity[110]を利用する．RDF コンテンツを保存するデータベースには MySQL[111]を選択した．

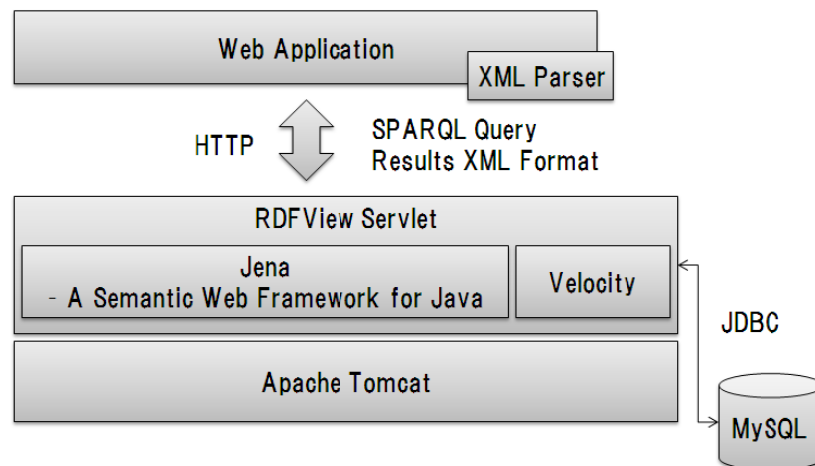


図 4.6 RDFView システム概要

RDFView は REST full なインタフェースを介してビューの管理を行う。したがって、RDFView は HTTP の各メソッドに対応した機能呼び出す。GET メソッドの場合はビューの取得機能、POST メソッドの場合はビューの登録機能、DELETE メソッドの場合はビューの削除機能呼び出す。表 4.1 に RDFView で利用できる URL 変数を示す。

表 4.1 RDFView 共通で利用できる変数

変数名	説明	GET	POST	DELETE
label	ビューの名前	必須	必須	必須
query	SPARQL テンプレート	-	必須	-
title	タイトル	-	○	-
stylesheet	スタイルシート URL	○	-	-
reasoner	推論エンジン	○	-	-

RDFView はまたビューを操作できる Web 画面も用意している。図 4.7 に RDFView におけるビューの登録画面である。登録画面で送信ボタンを押すと、Web 画面はリクエスト URL を組み立てて、RDFView に対して POST メソッドを発行する。

RDFView Label
medblog.testSubjectPage

View識別用ラベル

title:
被験者に関連するデータの一覧を表示する

Viewのタイトル・説明

SPARQL Query

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>;
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>;
PREFIX med: <http://www.sociotech-lab.jp/schema/medblog#>;
select distinct ?uri ?url ?dateOfSurvey ?owner ?surveyType ?surveyContent
where{
  ?uri med:testSubjectID '$ssid' .
  ?url med:isSurveyResultOf ?uri .
  ?url rdf:type med:SurveyData .
  ?url med:dateOfSurvey ?dateOfSurvey .
  ?url med:owner ?owner .
  ?url med:surveyType ?surveyType .
  optional{?url med:surveyContent ?surveyContent .}
} order by ASC(?dateOfSurvey)
limit $order offset $offset
```

SPARQLテンプレート
\$記号から始まる英字を外部変数とする。

モデル名:
medblog

送信

図 4.7 ビューの登録画面

```
<?xml version="1.0" ?>
- <sparql xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:xs="http://www.w3.org/2001/XMLSchema#"
  xmlns="http://www.w3.org/2005/sparql-results#">
- <head>
  <variable name="uri" />
  <variable name="url" />
  <variable name="dateOfSurvey" />
  <variable name="owner" />
  <variable name="surveyType" />
  <variable name="surveyContent" />
- </head>
- <results>
- <result>
  - <binding name="uri">
    <uri>http://localhost/medblog/wp-content/themes/medblog/testSubjectPage.php?sid=C000001</uri>
  - <binding>
  - <binding name="url">
    <uri>file:///localhost/C:/Documents%20and%20Settings/m-haya/%E3%83%87%E3%82%B9%E3%82%AF%E3%83%88%E3%83%83%E3%83%97/%E7%A8%B8%E5%BA%A6%E3%82%AA%E3%83%B3%E3%83%88%E3%83%AD%E3%82%B8/dex200803.pdf</uri>
  - <binding>
  - <binding name="dateOfSurvey">
    <literal>2008-01-1</literal>
  - <binding>
  - <binding name="owner">
    <literal>admin</literal>
  - <binding>
  - <binding name="surveyType">
    <literal>MEGデータ</literal>
  - <binding>
- </result>
- </results>
- </sparql>
```

ヘッダ部:変数名のリスト
<variable name="surveyType" />

問い合わせ結果部:データリスト
<binding name="surveyType">
<literal>MEGデータ</literal>
</binding>

図 4.8 ビューの実行結果例

RDFView はリクエスト URI を処理して、ビューをデータベースに登録する。図 4.7 で登録されたビューの実行結果例を図 4.8 に示す。実行結果は XML 形式で返されるので XSLT を使い自由に変換可できる。

4.4. 評価

RDFView はセマンティック Web に関係データベースのビューの概念を導入する。RDFView によるビュー機能の提供は、関係データベースのビューと同様、ユーザの利便性の向上が目的である。ビューを導入することでの利点は、データ独立性の確保とデータベースのマクロ的利用を支援できる点である。

本研究では RDFView の利便性を評価するために、データ独立性とプログラミングの容易性の評価を行った。評価項目には、データの独立性とプログラムの可読性である。ここでのプログラムの可読性は直接プログラミングの容易性に繋がるものではない。しかしながら、可読性を向上させることで、学習効率やプログラムの保全性を高めることができる。その結果、プログラミングの容易性にも関係してくる。

4.4.1 評価方法

まず、データ独立性の評価方法について説明する。本研究では、問い合わせ文の変更により、プログラムの変更がどのくらい必要なかを調べることでデータの独立性の評価を行う。そのために、三つの問い合わせ文を用意し、問い合わせ処理を行うプログラムをそれぞれの問い合わせ文に対して三種類用意した。

問い合わせ文の内訳は、(1) 基準となる問い合わせ文、(2) 前の文に条件文を追加した問い合わせ文、(3) そしてプログラムから変更される問い合わせ文である。

同じく準備したプログラムの内訳は、(1) RDFView 利用プログラム、(2) SPARQL Endpoint 利用プログラム、(3) Jena データベース利用プログラムである。これらのプログラムは全て Jena ライブラリを利用して作成され、問い合わせ文に対応して作られた。したがって、合計九つのプログラムを用意したことになる。本研究では、これらのプログラムの行数を計測することで、データ独立性の評価を行った。プログラム行数の計測には、Eclipse Metrics Plug-in[115]を利用した。

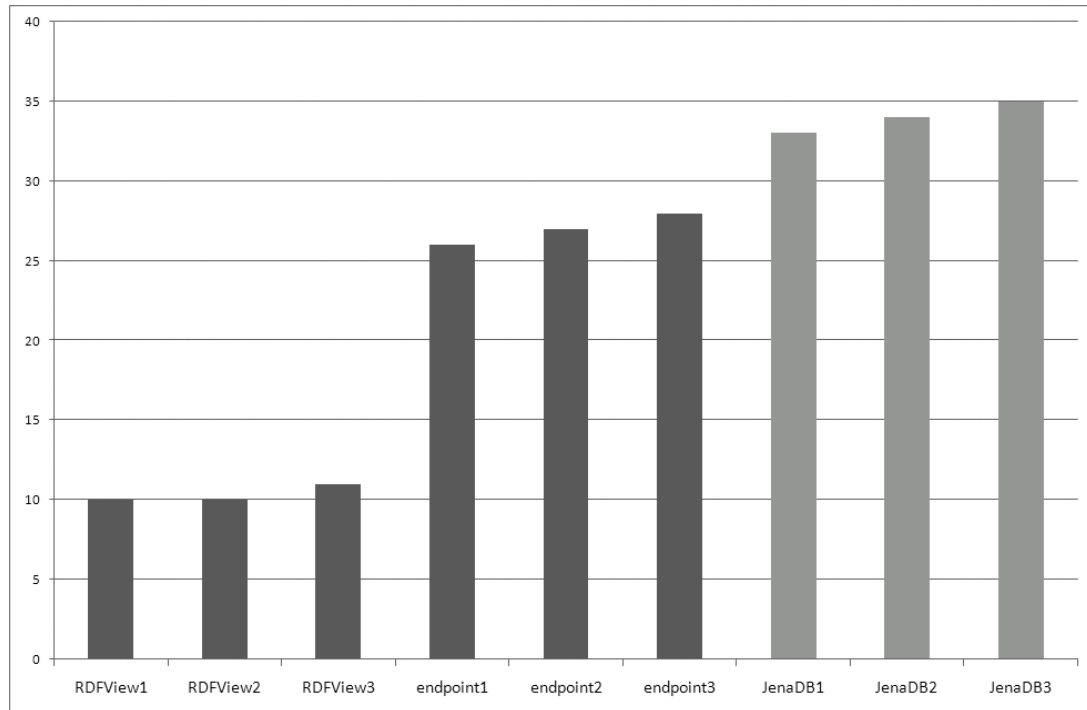


図 4.9 プログラム行数（メソッド行数）の比較

棒グラフによる計測結果を図 4.9 に示す．図中のラベル **RDFView** が **RDFView** を利用したプログラム, **endpoint** が **SPARQL Endpoint** を利用したプログラム, **JenaDB** が **Jena** データベースを利用したプログラムを表している．また，ラベルの後ろの数字は問い合わせ文の種類を表す．

計測結果から **RDFView** を利用したプログラムでは基準となる問い合わせ文と条件を追加した問い合わせ文によるプログラム行数の変化はみられない．一方, **SPARQL Endpoint** と **Jena** データベースを利用したプログラムでは問い合わせ文の変更によるプログラム行数の変化がみられる．一方，三番目の問い合わせ文を利用したプログラムではプログラム行数が全ての手法で増加している．

次にプログラムの可読性の評価を行った．本研究ではプログラム可読性の評価項目としてメトリクスを利用する．メトリクスはプログラムの可読性を評価する尺度として一般的に利用されている．メトリクスには, **McCabe** の循環複雑度[117]と **Chidamber and Kemerer** のオブジェクト指向メトリクス(**C&K** メトリクス)[118]を利用する．

これらメトリクスの計測には **McCabe** の循環複雑度を **Eclipse Metrics Plug-in** で, **C&K** メトリクスの計測には **ckjm**[116]を利用した．

計測の結果, McCabe の循環複雑度は全てのプログラムで同値で循環複雑度 3 であった.

C&K メトリクスの計測結果を表 4.2 に示す. 表中の WMC (Weighted methods per class) はクラスにおけるメソッドの割合, DIT (Depth of Inheritance Tree) は継承の深さ, NOC (Number of Children) はクラスの子の数, CBO (Coupling between object classes) は他クラスとの関連度, RFC (Response for a Class) は他クラスのメソッドの関連度, LCOM (Lack of cohesion in methods) はクラス内のメソッド数を表す. C&K メトリクスの計測結果から, CBO と RFC に値に各手法の変化が現れた. なお, 実験に利用した問い合わせ文, プログラムは付録 A に示す.

表 4.2 C&K メトリクスの計測結果

Program	WMC	DIT	NOC	CBO	RFC	LCOM
RDFView1	3	1	0	4	11	3
RDFView2	3	1	0	4	11	3
RDFView3	3	1	0	4	15	3
endpoint1	3	1	0	8	13	3
endpoint2	3	1	0	8	13	3
endpoint3	3	1	0	8	16	3
JenaDB1	3	1	0	15	18	3
JenaDB2	3	1	0	15	18	3
JenaDB3	3	1	0	15	21	3

4.5. 考察とまとめ

4.5.1 Pluggable Metadata Extractor の考察とまとめ

Pluggable Metadata Extractor はプラグイン機構を備えた拡張可能な RDF コンテンツ作成 Web サービスである。Pluggable Metadata Extractor は非常に Babel[112]と似たシステムである。Babel も同様に Web サービスとして RDF コンテンツを作成する。異なる点として、Babel ではユーザが変換対象であるファイルを指示した後に RDF コンテンツを作成するが、Pluggable Metadata Extractor は自動的に対象ファイルを判断する。そして、もう一つはプラグイン機構により拡張可能な点である。

ユーザ要件に基づく情報統合環境ではこの拡張可能な点が大きな利点となる。本研究の情報統合環境ではパーソナルコンピュータ上に保管されたファイルから RDF コンテンツを作成し、ファイルの検索を可能にする。始めから全てのファイル形式に対応することは難しいが、プラグインで拡張可能になったことで、必要なときに機能を加えることが可能である。

4.5.2 RDFView の考察とまとめ

データ独立性の評価実験の結果から、RDFView はこれまで利用されてきた SPARQL Endpoint や Jena データベースを利用していた場合よりも、データの独立性の向上がみられた。この結果は、RDFView の場合問い合わせ文を外部定義するため自明である。しかしながら、セマンティック Web という不安定なデータ群の集合を利用したアプリケーションを構築する場合には、データ独立性は高いほど良い。

一方、プログラムの可読性評価実験の結果からは、RDFView を導入した事による可読性の向上は見られなかった。しかしながら、逆にこの結果は RDFView の導入がこれまでのプログラミングスタイルを阻害することは無いことを示している。C&K メトリクスの CBO と RFC から RDFView は外部クラスとの依存度が低いことが解る。このこともまた、RDFView はデータベースの接続や問い合わせ文のオブジェクト化を Web サービスとして実行していることから当然の結果である。ただし、RDFView の外部クラスとの依存度の低さは、他のプログラミング言語での利用も容易であることを示す。実際、RDFView を利用するには、HTTP の GET メソッドの発行と XML 解析処理が可能なプログラミング言語であれば環境を選ぶことはない。

RDFView は本研究において重要な役割を担う情報システムである。ユーザ要件に基づく情報統合環境では、パーソナルコンピュータ上に保存されたファイルコンテンツをユーザがどのように表示して欲しいのかというユーザの要件を、この RDFView のビューとして実装する。

パーソナルコンピュータ環境上に保管されたファイル群は先に述べた **Pluggable Metadata Extractor** により、RDF コンテンツに変換される。しかしながら、ユーザはそれら RDF コンテンツを統合する術を知らない。ユーザからどの情報が欲しいのか、必要ないのかを要件として引き出し、それをテンプレートとして表現し、ビューに保管する。ユーザ要件に基づく情報統合環境では、そのようなビューをいくつも用意する。それらのビューを必要なときに呼び出し、ユーザが求める形で情報を提供する。これが RDFView の役割である。

また RDFView はセマンティックマッシュアップ[20]の実現も支援する。Web から RDF コンテンツを収集し、目的に応じたビューを定義する。RDFView はそれらのビューを Web サービスとして公開する。それらの Web サービスを使ってセマンティックマッシュアップを実現する。たとえばその FOAF メタデータの名前、GEO メタデータの位置情報を検索するビューを定義する。そして RDFView と Yahoo! Pipe を使ってそれらの情報を地図上にマッピングする。図 4.10 はこの例を実現した Yahoo! Pipe の画面写真である。

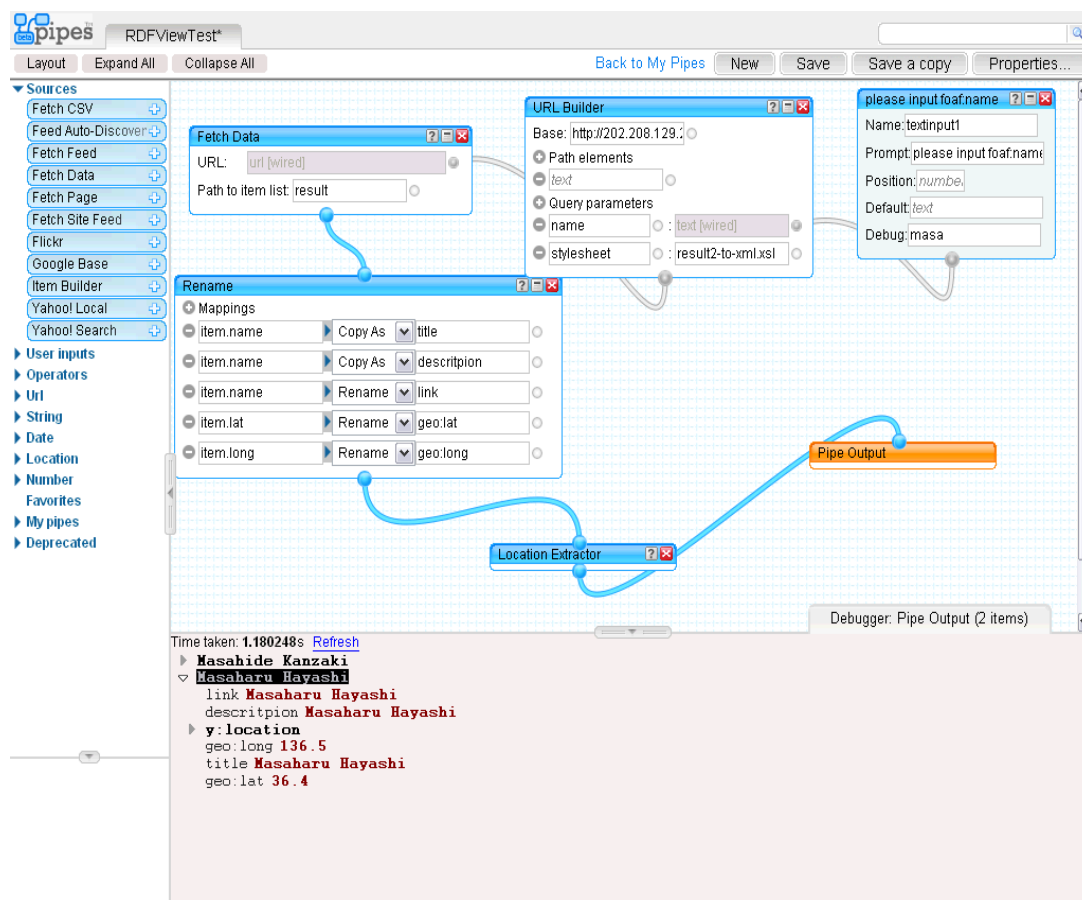


図 4.10 セマンティックマッシュアップの例 (RDFView と Yahoo! Pipe)

4.6. おわりに

本章ではユーザ要件に基づく情報統合環境の中心機能となる Pluggable Metadata Extractor と RDFView について述べた。

Pluggable Metadata Extractor はパーソナルコンピュータ環境上に保存されたファイル群を RDF コンテンツ化する。Pluggable Metadata Extractor は RDF コンテンツ作成機能の拡張性を確保するために、プラグイン機構を備える。本章では、そのプラグイン機構を利用して、DICOM 画像ファイル形式、Microsoft Excel ファイル形式、XML ファイル形式そして Adobe XMP ファイル形式から RDF コンテンツを作成するプラグインを示した。Pluggable Metadata Extractor と類似する SIMILE プロジェクトの Babel もまた、RDF コンテンツを作成する Web サービスを提供する。しかしながら、プラグイン機構は

備えていない。Web には RDF コンテンツ化されていないファイルが多く存在する。Pluggable Metadata Extractor が備えるプラグイン機構は将来の拡張性を有するという点で有用である。

一方、RDFView は Pluggable Metadata Extractor が作成した RDF コンテンツや、トリプルストアに保存された RDF コンテンツを予め用意しておいたビューを利用して統合する。

Web に存在する RDF コンテンツを活用できる人間は少ない。なぜなら、RDF コンテンツを活用するには、その RDF コンテンツが何を表しているかを知っている必要があるからである。つまり、その RDF コンテンツで利用されている語彙についての知識が求められる。つまり、RDF コンテンツの中で Dublin Core や FOAF のように広く知られた語彙が使われていない限り、その RDF コンテンツは利用可能な状態にない。

RDFView は利用可能な状態にない RDF コンテンツに対して、あらかじめ定式化した SPARQL 問い合わせ文より Web サービスを提供することで、その RDF コンテンツの利用手段を提供する。RDFView は RDF コンテンツを作成した人間が、その RDF コンテンツの活用方法をその他の人間に提供するための一手法である。また、RDF コンテンツを利用したい人間の要求にしたがって、その RDF コンテンツに対するインタフェースを提供する役割を担うこともできる。

また、Yahoo! Pipe の例で示したように、RDFView のこの仕組みはセマンティックマッシュアップ実現の一つの手法になりうる。RDFView は現在の Web とセマンティック Web との橋渡しを実現するという点においても有用な仕組みである。

Pluggable Metadata Extractor はプラグイン機構を備えた RDF コンテンツ作成機構である。Pluggable Metadata Extractor はファイルのコンテンツに対して、統一されたアクセス手段を提供するために、ファイルから RDF コンテンツを作成する。一方、RDFView はそれら RDF コンテンツをユーザの要件に合わせて事前に用意したビューによる統合手段を提供する。これら二つの仕組みはユーザの要求に応じてファイル内の情報を自由に統合するための環境を提供する。次章では、このユーザ要件に基づく情報統合環境を利用して構築した認知症早期診断法開発研究におけるファイル管理を行う情報システムについて説明する。

第5章

ユーザ要件に基づく情報統合環境を利用したファイル管理支援システム

5.1. はじめに

認知症診断法の研究開発では膨大な量の検査データを分析し、その評価を行う。それら検査データは、コンピュータに保管され、管理されている。しかし被験者の増加に伴い、検査データの量やそれらを保持しているファイル数が増える。またこれらの検査データから加工生成されるファイルや、その他、加工途中で生成される一時ファイルなどが混在することになる。また研究開発の特徴として、取り扱うデータ項目が研究の進展に伴い変化するため、それらを管理する定型的なアプリケーションソフトウェアの開発が困難である。このようなことを背景として、これらのファイルは、ファイル名及びそれらの保管場所としてのディレクトリ名に一種の命名規則を設定し、管理されることになる。そして検査チームの各人がそれらの命名規則を理解することで対応する。しかしながらこのデータ管理方法は、検査等データの増加やディレクトリの追加や削除などに伴い、検査チームが特定のデータをアクセスする作業に混乱を引き起こすことになる。

なお、第3章でも述べたが、本研究が対象とする認知症早期診断法開発研究の現場では、MRI 検査、MEG 検査、健康診断・血液検査、MMSE やタッチパネル式簡易認知機能検査などの各種検査を実施し、それら検査の結果のファイルの蓄積を行っている。それらのファイルは研究者により、第一次データ群（未加工の検査結果データ）、第二次データ群（グラフデータや図データ）、第三次データ群（表形式データ）に分類される。認知症診断法開発研究現場の情報システム環境で（図 3.1 参照）で利用されているファイル形式は、MRI 検査結果を格納する DICOM 画像形式、健康診断・血液検査結果や各種認知機能検査結果、各種検査データの分析結果を格納する Microsoft Excel 形式、紙媒体の検査結果を格納する Adobe Acrobat 形式、MEG 検査の結果を格納する Meg Laboratory 形式、そして被験者・患者への説明資料及び発表資料に使われるデータを格納した PowerPoint 形式である。

これらのファイルは被験者 ID や検査方法によりディレクトリに分類され、管理されて

いた（図 5.1 参照）。また、各検査データは一つのファイルにまとめられ（表データ.xls），通常はその一つのファイルを加工しながら，作業が行われていた。ただ，各検査値の元データを確認する場合も多く，ディレクトリに分類されたファイルを探し出し，元データの確認を行う作業が頻繁に生じていた。

本研究の目的はこれらの作業を支援することである。その為に解決すべき課題が二つある。まず，認知症早期診断法開発研究現場に存在するファイル群を研究者達が発見しやすい形にすること，そして，それら発見しやすい形にしたファイル群を研究者達が要求する形で提示することである。

本研究では，これらの課題を解決するために，ユーザ要件に基づく情報統合環境を応用する。具体的には，ファイルの発見を容易にするために，ファイルのメタデータを作成する。そして，そのメタデータを統合するビューを **RDFView** に定義する。ファイル管理支援システムは **RDFView** の処理結果を利用した。研究者達はそのファイル管理支援システムを介してファイルにアクセスできる。

- JB
 - etc
 - ankt
 - edin
 - memo
 - mmse
 - c99_mmse.PDF
 - MEG_data
 - c99
 - raw_data
 - c99_eye_close_1.con
 - work
 - data_work(c99).xls
 - 磁場強度分布(c99).ppt
 - c99_eye_close_miss_oddの加算平均.sqd
 - 患者用お土産(c99).ppt
 - MRI_data
 - C99
 - oc001-835.290380.1183352430
- PT
- 表データ.xls

図 5.1 ディレクトリ構造抜粋

5.2 ファイル管理支援システムのメタデータ

ファイル管理支援システムでは、研究者達のファイル発見を支援するために、メタデータを活用する。メタデータに研究者達がファイルを識別する際に用いる情報を含めることで、ファイル発見の機会を増加させる。そこで、研究者達と打ち合わせを行い、ファイルの識別に必要な情報を明らかにし、メタデータの設計を行った。

認知症早期診断法研究開発現場では、ID により管理された被験者毎に検査が実施される。そして、検査毎にファイルが作成される。ファイルは研究者達が作成したディレクトリ構造(図 5.1 参照)に分類され保管される。研究者達は、そのディレクトリ構造から被験者の ID と検査方法をキーにファイルを探す。これらのことから、ファイルの発見には被験者 ID と検査方法に関する情報が必要である。また、研究者達は、被験者 ID 以外に、年齢や利き腕、そして病歴などの被験者に関する基礎情報と、検査日時や検査の概要などの検査情報も重要であることが明らかとなった。また、これらの情報以外に必要な情報が発生する可能性も明らかとなった。

そこで本研究では明らかになった情報を基に、被験者の基礎情報を表現する「基礎メタデータ」と検査情報を表現する「検査メタデータ」の設計を行った。まず、被験者を表現する被験者クラスと、検査データを表現する検査データクラスを定義した。つぎに各クラスに対して、打ち合わせの中で必要とされた項目をプロパティとして定義した。本研究では、これらの定義情報は RDF スキーマとして記述した。

被験者クラスは被験者 ID、性別、年齢、計測日、利き手、病歴、被験者タイプ、HDS-R(改定長谷川式簡易知能評価スケール)、MMSE という被験者を識別するためのプロパティを所有する。これらのプロパティを利用して、被験者の基礎情報を表す基礎メタデータを作成する。図 5.2 に基礎メタデータの例を RDF グラフで示す。

検査データクラスは検査日時、担当者、検査方法、ラベル、概要、検査対象という検査データを識別するためのプロパティを所有する。ここでの検査対象プロパティは被験者クラスを参照することで、被験者と検査データの関係を表現する。これらのプロパティを利用して、検査情報を表す検査メタデータを作成する。図 5.3 に検査メタデータの例を RDF グラフで示す。

研究者達は被験者毎に基礎メタデータを作成し、ファイル管理支援システムに登録する。同じように、検査メタデータは、ファイルをファイル管理支援システムに登録するときに、研究者達が作成する。研究者達は被験者 ID や性別、そして検査方法などの条件をファイル管理支援システムに設定することで、ファイルを検索することが可能となる。

また、ファイル管理支援システムでは、ファイルから抽出したメタデータも同じように、ファイル検索に利用する。このことにより、さらに詳細な情報を利用したファイル検索を提供することができる。

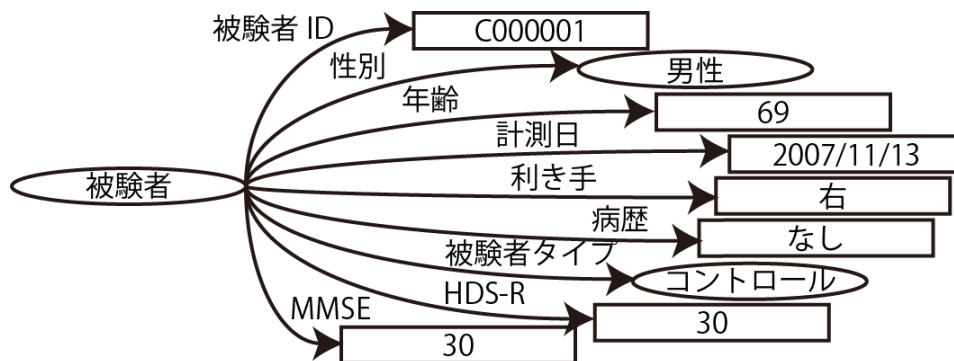


図 5.2 基礎メタデータ

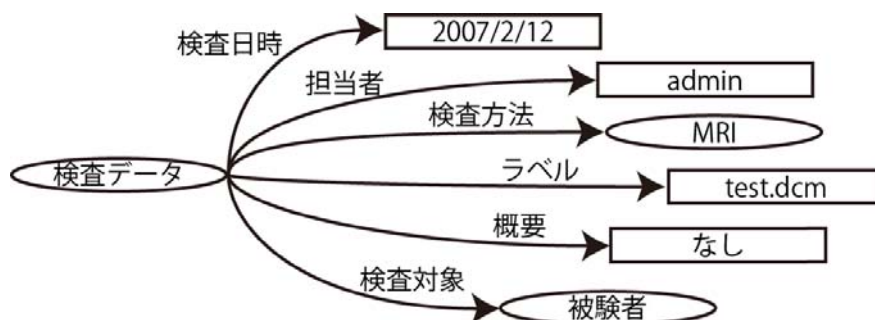


図 5.3 検査メタデータ

5.3 ファイル管理支援システム

ファイル管理支援システムでは、ファイルにメタデータを関係付けることで、詳細なファイル検索を実現する。ファイル管理支援システムは Web アプリケーションとして構築した。Web ブラウザがインストールされているパーソナルコンピュータであればシステムを動作させることが可能な点と、ファイル管理に必要な情報をサーバで一元管理できるからである。また、近年の Web アプリケーションは AJAX(Asynchronous JavaScript and

XML)の登場により、その操作系も改善されてきている。さらに、Web ブラウザは研究者達が普段から慣れ親しんでいるインタフェースの一つであり、ファイル管理支援システムの操作時の負担を下げることも期待できる。

ファイル管理支援システムは Google Web Toolkit (GWT) [113]を利用して作成した。GWT では、AJAX ベースの Web アプリケーションを Java 言語で記述することができる。GWT の利点に、Web ブラウザの違いを GWT 自体が吸収することがあげられる。したがって、Web ブラウザ毎に異なるコードを書かなくても、GWT が対応する複数の Web ブラウザで Web アプリケーションを動作させることが可能となる。

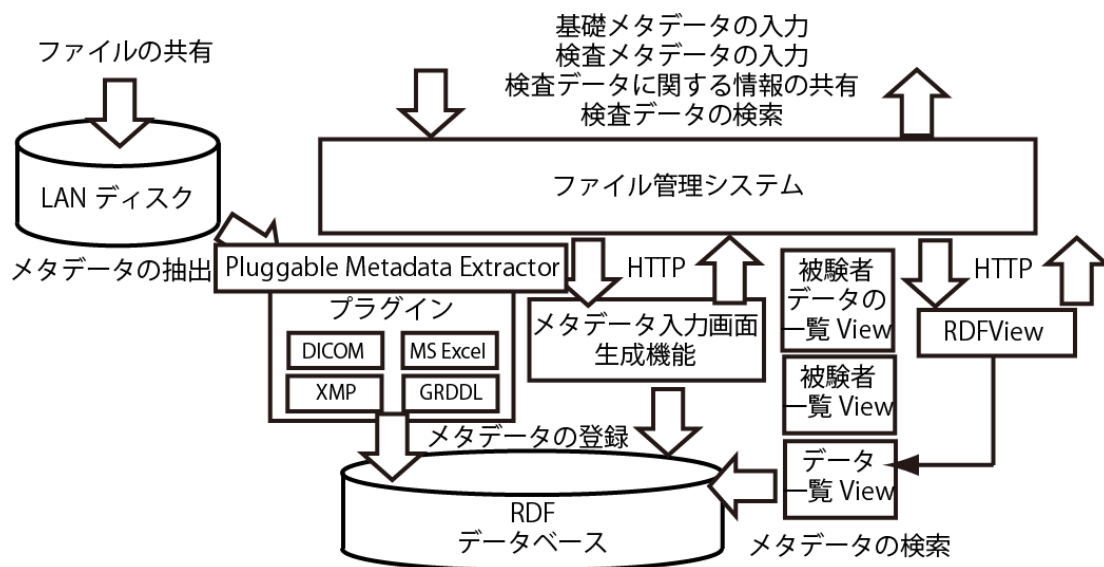


図 5.4 システムの概要図

図 5.4 にファイル管理支援システムの概要を示す。ファイル管理支援システムは、Web アプリケーションとして構築したファイル管理支援システム本体と、ファイルからメタデータの抽出を行う Pluggable Metadata Extractor,そしてメタデータが格納された RDF データベースからの情報統合を担う RDFView から構成される。このファイル管理支援システムを認知症早期診断法開発現場のシステム環境に導入した(図 5.5 参照)。具体的には、図 5.5 の MIMS Server と名付けられたパーソナルコンピュータにファイル管理支援システムを導入した。このパーソナルコンピュータには Pluggable Metadata Extractor と RDFView も導入されている。

ファイル管理支援システムは、認知症早期診断法開発研究現場の情報システム環境に保

存されたファイル群にたいしてメタデータを関係付けることで、研究者達のファイル管理を支援する。ファイル管理支援システムは、基礎メタデータと検査メタデータの作成及び編集、メタデータを利用したファイル検索、そしてメタデータを利用したファイルブラウジング機能を提供する。これらの機能はファイル管理支援システムにログインすることで利用可能となる(図 5.6 参照)。

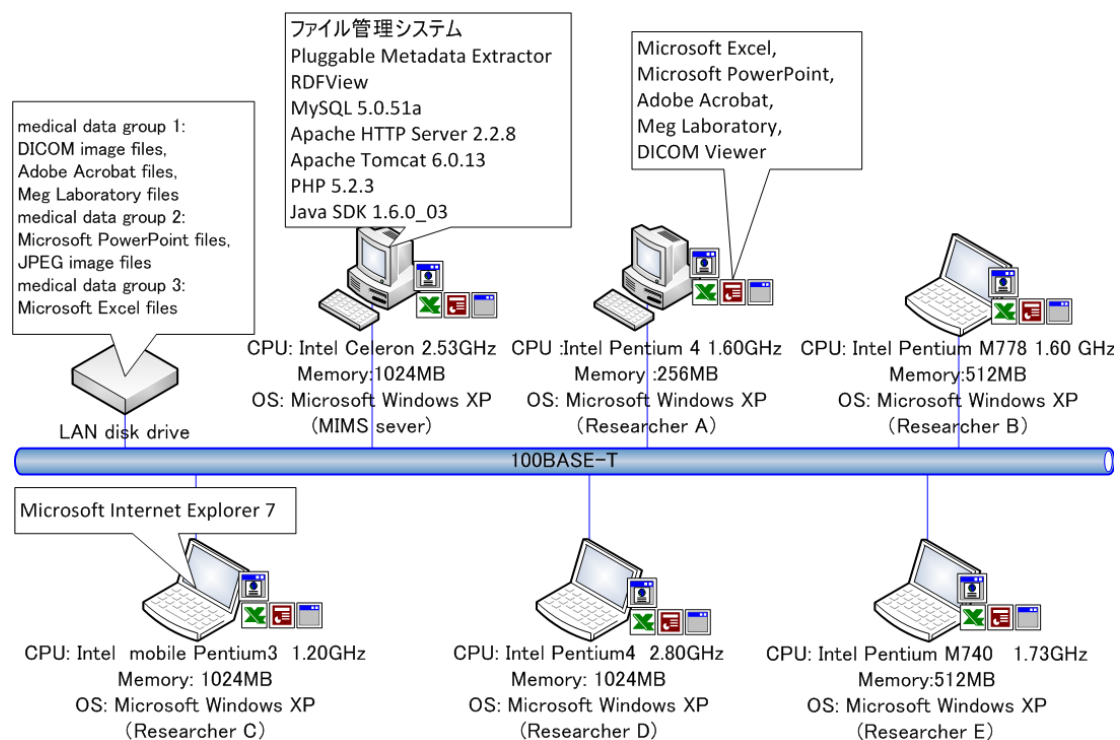


図 5.5 システム環境

url	被験者ID	性別	年齢	計測日	利き手	病歴	MMSE	HDSR	検査者タイプ
☐	C01	男性	87.0	2007-11-13	右	なし	23.0	22.0	コントロール
☐	P01	男性	78.0	2008-07-27	右	なし	23.0	23.0	単室
☐	P02	女性	89.0	2008-07-23	右	なし	24.0	22.0	コントロール
☐	P05	男性	89.0	2008-07-28					ケース
☐	P06	女性	86.0	2008-07-29					単室
☐	P07	男性	78.0	2008-07-29					単室
☐	c02	男性	88.0	2007-11-13					コントロール
☐	c03	男性	90.0	2007-11-13	右	なし	22.0	23.0	コントロール
☐	c04	女性	88.0	2007-11-13	右	なし	22.0	24.0	ケース
☐	c05	女性	92.0	2007-11-13	右	なし	20.0	23.0	ケース

図 5.6 ログイン画面

5.3.1 基礎メタデータと検査メタデータの作成及び編集

ファイル管理支援システムを利用するには、ファイルのメタデータを登録する必要がある。そのために、まず被験者の基本情報をファイル管理支援システムに入力する。その後に、その被験者の基本情報と検査データを関係付ける。ファイル管理支援システムはそのための基礎メタデータと検査メタデータの作成及び返球機能を提供する。

基礎メタデータと検査メタデータの作成及び編集機能は、それらの定義情報を記述された RDF スキーマから作成されるメタデータ入力画面を介して提供される。具体的には、RDF スキーマに定義されたクラスとそのクラスを `rdfs:domain` に持つプロパティを利用して Web 画面を作成する。メタデータ入力画面作成手順を図 5.7 の RDF スキーマを例に説明する。まず、RDF スキーマに定義されたクラスを読み込みクラスのプロパティと制約条件を列挙する。図 5.7 の例では `TestSubject` クラスを読み込み、`rdfs:domain` に `TestSubject` クラスが指定された `testSubjectID` プロパティを列挙する。また、制約条件として `testSubjectID` プロパティに出現回数(`owl:cardinality` の値)が指定されている(図 5.7 の六行目から九行目まで)。つぎに列挙されたプロパティの `rdfs:range` の値を調べる。

rdfs:range の値がリテラルであればテキストフィールドを作成する。rdfs:range の値がリソースであれば、そのリソースのインスタンスを選択項目としたセレクトボックスを作成する。図 5.7 では、testSubjectID プロパティの rdfs:range にはリテラルが指定されているため、テキストフィールドを作成する。

最後に制約条件がある場合はスクリプト言語を作成し入力の制御を行う。図 5.7 では、testSubjectID プロパティの出現回数に関する制約条件が設定されている。そのため、testSubjectID プロパティの入力を強制するスクリプトを作成する。

以上の手順を踏まえて作成された入力画面を介して入力されたメタデータをファイル管理支援システムはデータベースに保管する。また、メタデータの編集作業も同じ入力画面で行う。図 5.8 に基礎情報の入力画面写真を、図 5.9 に検査情報とファイルの入力画面を示す。

なお、ファイル管理支援システムはファイル登録の際に、Pluggable Metadata Extractor を呼び出し、登録されたファイルからメタデータの抽出を行う。抽出したメタデータは検査メタデータと同様、RDF データベースに保管される。

```
1. <rdfs:Class rdf:about="http://www.sociotech-lab.jp/schema/medblog#TestSubject">
2. <rdfs:isDefinedBy rdf:resource="http://www.sociotech-lab.jp/schema/medblog#" />
3. <rdfs:label>被験者</rdfs:label>
4. <rdfs:subClassOf rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
5. <rdfs:subClassOf>
6. <owl:Restriction>
7. <owl:onProperty rdf:resource="http://www.sociotech-lab.jp/schema/medblog#testSubjectID"/>
8. <owl:cardinality>1</owl:cardinality>
9. </owl:Restriction>
10. </rdfs:subClassOf>
11. <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
12. </rdfs:Class>
13. <rdf:Property rdf:about="http://www.sociotech-lab.jp/schema/medblog#testSubjectID">
14. <rdfs:isDefinedBy rdf:resource="http://www.sociotech-lab.jp/schema/medblog#" />
15. <rdfs:label>ID</rdfs:label>
16. <rdfs:domain rdf:resource="http://www.sociotech-lab.jp/schema/medblog#TestSubject"/>
17. <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
18. <rdfs:subPropertyOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#label"/>
19. </rdf:Property>
```

図 5.7 基礎メタデータの RDF スキーマ（一部抜粋）

A Medical Information Management System

ホーム 基礎データの管理 調査データの管理 Logout

Next Back Add Select All UnSelect All Delete reload

ID C01 検索 戻る

url	被験者ID	性別	年齢	計測日	利き手	病歴	MMSE	HDSR	被験者タイプ	Edit	Add
☒	C01	男性	87.0	2007-11-13					コントロール	Edit	Add
☒	P01	男性	78.0	2008-07-27					未定	Edit	Add
☒	P02	女性	89.0	2008-07-23					コントロール	Edit	Add
☒	P05	男性	89.0	2008-07-28					ケース	Edit	Add
☒	P06	女性	86.0	2008-07-29					未定	Edit	Add
☒	P07	男性	78.0	2008-07-29					未定	Edit	Add
☒	c02	男性	88.0	2007-11-13					コントロール	Edit	Add
☒	c03	男性	90.0	2007-11-13					コントロール	Edit	Add
☒	c04	女性	88.0	2007-11-13	右	なし	22.0	24.0	ケース	Edit	Add
☒	c05	女性	92.0	2007-11-13	右	なし	20.0	23.0	ケース	Edit	Add

Next Back

Footer

図 5.8 基礎情報の登録

A Medical Information Management System

ホーム 基礎データの管理 調査データの管理 Logout

Next Back Add Select All UnSelect All Delete reload

ID C01 検索 戻る

url	被験者ID	性別	年齢	計測日	利き手	病歴	MMSE	HDSR	被験者タイプ	Edit	Add
☒	C01	男性	87.0	2007-11-13					コントロール	Edit	Add
☒	P01	男性	78.0	2008-07-27					未定	Edit	Add
☒	P02	女性	89.0	2008-07-23					コントロール	Edit	Add
☒	P05	男性	89.0	2008-07-28					ケース	Edit	Add
☒	P06	女性	86.0	2008-07-29					未定	Edit	Add
☒	P07	男性	78.0	2008-07-29					未定	Edit	Add
☒	c02	男性	88.0	2007-11-13					コントロール	Edit	Add
☒	c03	男性	90.0	2007-11-13					コントロール	Edit	Add
☒	c04	女性	88.0	2007-11-13	右	なし	22.0	24.0	ケース	Edit	Add
☒	c05	女性	92.0	2007-11-13	右	なし	20.0	23.0	ケース	Edit	Add

Next Back

Footer

図 5.9 検査情報の登録

5.3.2 メタデータを利用したファイルの検索

RDF データベースに保存されたファイルのメタデータはファイル管理支援システムの提供する検索機能を介して利用できる。ファイル管理支援システムの検索機能は、RDFView を利用して提供される。

本研究では、認知症早期診断法開発研究現場の研究者達の意見を聞きながら RDFView のビューの定義を行った。ファイル管理支援システムで利用するビューは被験者一覧を表示するためのビュー、検査データの一覧を表示するためのビューである。図 5.10 に検査データを表示するためのビューを示す。

RDFView はこれらのビューに定義された外部変数を利用した Web サービスインタフェースを作成する。ファイル管理支援システムはこれらの Web サービスインタフェースを呼び出すことで、条件に沿った検査情報の表示を実現する。たとえば、被験者 ID を指定して検査データの一覧を表示するビューを呼び出せば、その被験者 ID の検査データの一覧が表示される。

ファイル管理支援システムは RDFView のビューと変数の組み合わせでファイルのメタデータ検索機能を提供する。ファイルのメタデータの検索結果にはそのファイルの検査メタデータのそのファイルへのリンクが含まれている。研究者はそのリンクを開くことで、目的のファイルへアクセスすることができる。

```

1. <j.0:RDFView
   rdf:about="http://localhost/MedSW/servlet/SPARQLComponentServlet?label=medblog.testSubjectData">
2. <rdfs:label>medblog.testSubjectData</rdfs:label>
3. <j.0:title>medblog.testSubjectData</j.0:title>
4. <j.0:label>medblog.testSubjectData</j.0:label>
5. <j.0:query>select ?url ?調査日時 ?担当者 ?調査方法 ?ラベル ?概要
6.   where {
7.     ?url &lt;http://www.w3.org/1999/02/22-rdf-syntax-ns#type&gt; &lt;http://www.sociotech-
       lab.jp/schema/medblog#SurveyData&gt; .
8.     ?url &lt;http://www.w3.org/2000/01/rdf-schema#label&gt; ?ラベル .
9.     optional{?url &lt;http://www.sociotech-lab.jp/schema/medblog#dateOfSurvey&gt; ?調査日時 }
10.    optional{?url &lt;http://www.sociotech-lab.jp/schema/medblog#owner&gt; ?担当者 }
11.    optional{?url &lt;http://www.sociotech-lab.jp/schema/medblog#surveyContent&gt; ?概要 }
12.    optional{?url &lt;http://www.sociotech-lab.jp/schema/medblog#surveyType&gt; ?surveyType .
13.      ?surveyType &lt;http://www.w3.org/2000/01/rdf-schema#label&gt; ?調査方法 .}
14.    $filter
15.  } order by desc(?調査日時) limit $limit offset $offset</j.0:query>
16. </j.0:RDFView>

```

図 5.10 検査情報を表示するためのビュー

5.3.3 メタデータを利用したファイルブラウジング機能

ファイル管理支援システムでは、ファイルメタデータの検索の他に、ファイルのメタデータのブラウジング機能を提供する。研究者達は、ファイルのメタデータをブラウジングすることで、目的のファイルを発見できる。図 5.11 と図 5.12 にファイルブラウジング機能の画面写真を示す。ファイルブラウジング機能では、画面右側に表示されたメタデータの構造を選択しながら、ファイルを特定していく。条件に合ったファイルは画面左側にアイコンとして表示され、そのアイコンに関係付けられたリンクを開くことで、そのファイルにアクセスすることができる。図 5.11 では Excel ファイルから抽出されたメタデータを利用して、ファイルの特定を行っている。たとえば、ワークシートの名前から Excel ファイルを特定することや、Excel ファイルの属性名前から目的の Excel ファイルを特定することができる。図 5.12 は DICOM 画像ファイルのメタデータを利用して、ファイルの特定を行っている。図 5.12 では DICOM 画像ファイルのデータ要素をブラウジングして、DICOM 画像ファイルを特定することができる。

また、認知症早期診断法開発研究では Excel ファイルの利用頻度が多い。そのため、ファイル管理支援システムでは、Excel ファイル専用のブラウジング画面も用意している。

図5.13にそのExcelファイル専用のブラウジング画面を示す. このブラウジング画面では, ブラウジング結果を利用して, 研究者達が必要なデータだけを新たな表データとして表示することができる.

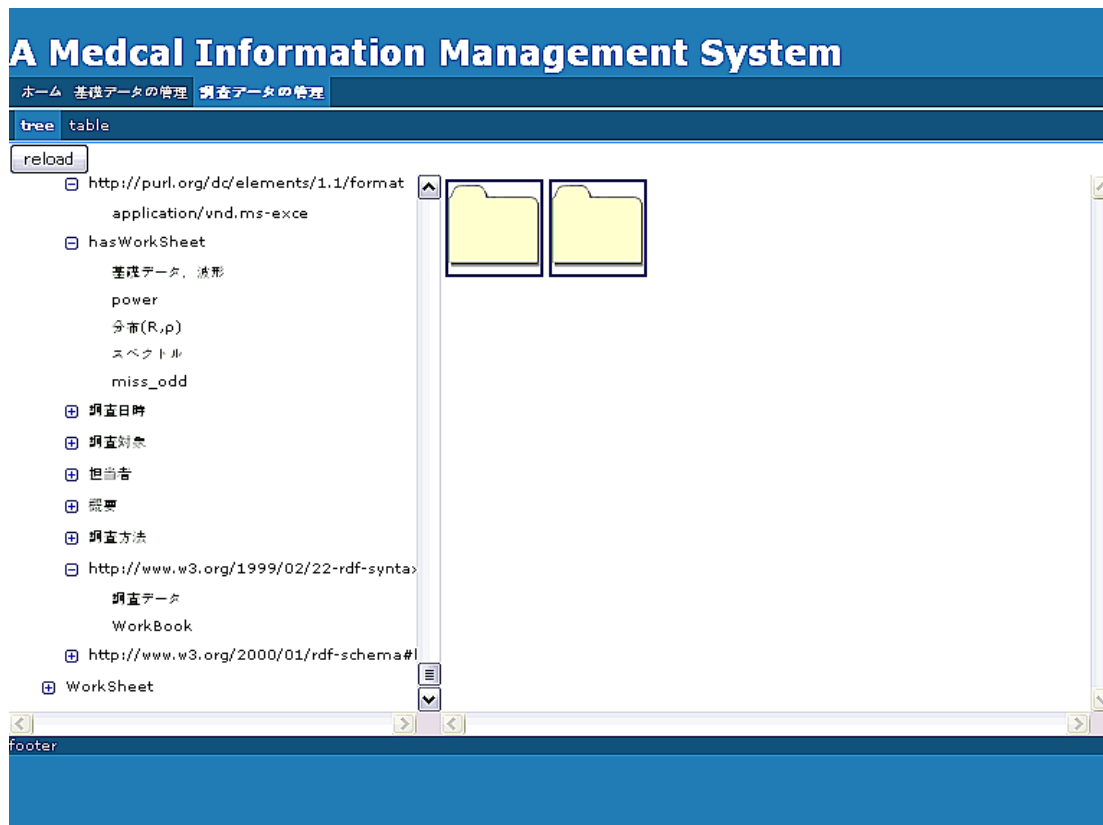


図 5.11 Excel ファイルのブラウジング画面

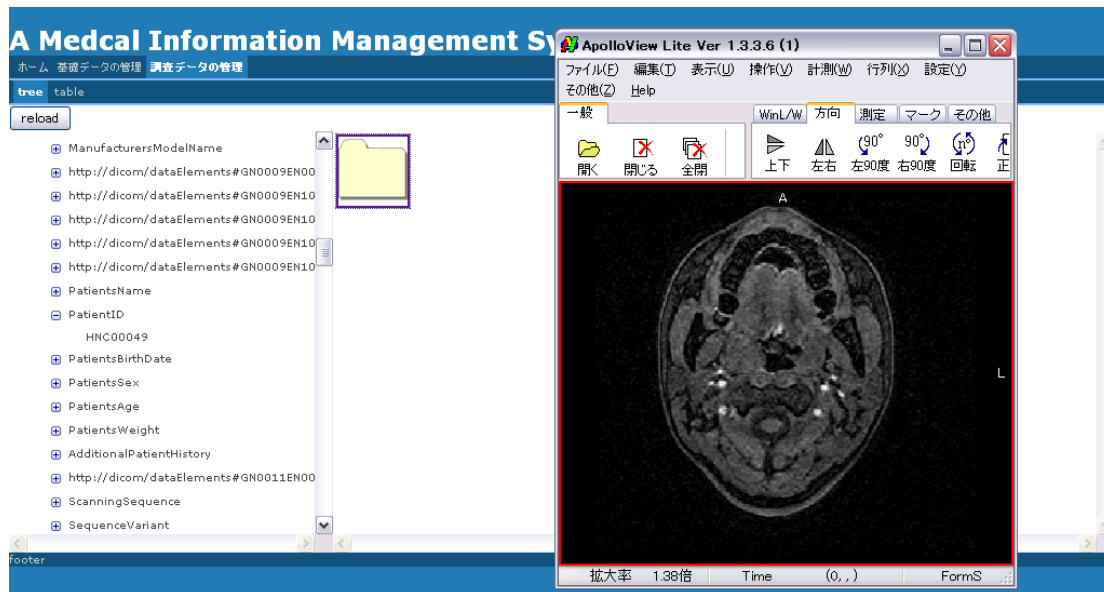


図 5.12 DICOM のブラウジング画面

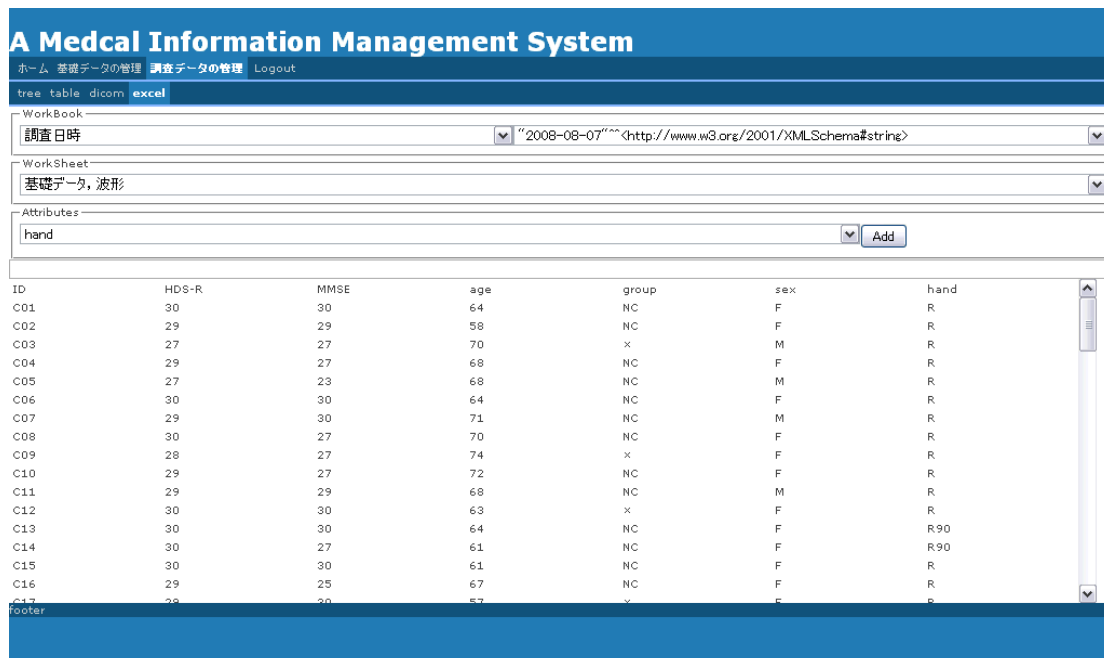


図 5.13 Excel ファイルのブラウジング

5.4 評価

本研究では **RDFView** の機能実証とともに、ファイル管理支援システムをファイル探索の効率化を目的に開発した。ここではファイル探索の効率化が実現できたかの評価を行う。評価方法は、作業工程の複雑度と作業工程の数をファイル管理支援システム導入前と導入後の **UML** のアクティビティ図を記述することで比較した。認知症早期診断法開発研究現場でよく発生する作業シナリオをアクティビティ図で表現した。その際、ファイル管理支援システムを利用するために必ず必要な作業工程以上のシナリオを作成し、アクティビティ図を記述した。そして、そのアクティビティ図を利用して **McCabe** の循環複雑度の計算を行った。**McCabe** の循環複雑度はグラフの複雑度の尺度として利用される。その計算式は次の通りである。

$$G(v) = e - n + p.$$

ここで、 $G(v)$ が循環複雑度、 e は辺の数、 n は頂点の数、 p はグラフの中のパスの数を意味する。計算したアクティビティ図は「**MMSE** 認知機能テストの点数が 21 点以下の被験者の **DICOM** ファイルを開く作業」である。計算に利用したアクティビティ図は付録の **B** に示す。これらのアクティビティ図の循環複雑度計算の結果、ファイル管理支援システムの導入前の複雑度が 8 ポインに対し、導入後の複雑度は 3 ポイントであった。ファイル管理支援システム導入後の方が作業工程の複雑度が減っていることが示された。

5.5 考察とまとめ

本章では、認知症早期診断法開発研究現場におけるファイルの管理を支援するファイル管理支援システムについて述べた。このファイル管理支援システムは、ファイルに関係付けられたメタデータを利用したファイル検索及びブラウジング機能を提供する。ファイル管理支援システムによる被験者情報を表現する基礎メタデータと検査情報を表現する検査メタデータ、そして **Pluggable Metadata Extractor** によるファイルのメタデータは **RDFView** によるビューにより統合され、ファイル管理支援システムはその統合結果を表示する。

本研究ではファイル管理支援システムをアクティビティ図の複雑度により評価した。卓上ベースの評価方法であり、作業工程の選定やアクティビティ図の妥当性に課題が残るが、

ファイル管理支援システムにより作業工程の複雑度が低くなったことを示すことで、ファイル探索における作業効率の向上が可能であることを示した。

ファイル管理システムは **RDFView** の機能の有効性を検証するために構築された情報システムである。本研究では、セマンティック **Web** を一つのデータベースとして捉え、**RDFView** は外部スキーマをビューとして定義した。そして、そのビューが作成する **Web** サービスを利用して本システムは作られた。これにより、**SPARQL** の課題を解決し、セマンティック **Web** アプリケーションの構築例を示した。

本ファイル管理支援システムは金沢工業大学先端電子技術応用研究所の樋口正法助教授の研究室の情報システム環境に導入された。これまでハードディスクに保存されたファイルを探し出す場合は、ディレクトリを探索する必要があったが、ファイル管理支援システムにより、ハードディスクに保存されたファイル群と患者情報や検査情報を関連付けて検索することが可能とになったとの評価を頂いた。

5.5.1 関連研究

文献[29]の **SFS** は本研究と同様にファイル種類特有 **Transducer** によりメタデータの自動抽出を行っているが、抽出したメタデータは独自形式で記録されており、**SFS** 以外から利用することができない。このことは将来のデータ構造の変更時に対応することや、他システムとの連携を困難とし、機能拡張が困難になることが考えられる。その点、**Pluggable Metadata Extractor** では汎用的なメタデータ抽出手法と **RDF** 形式によるメタデータの蓄積を行っているため、他のシステムから蓄積されたメタデータを利用することが容易である。また **RDFView** が提供する **REST** によるシステム間の接続も将来のデータ構造やシステムの変更時に対応するための工夫点である。

文献[31]の手法はディレクトリ構造で管理されたファイルに対して、一つ以上のキーワードを付加し、ファイルの発見の効率化を実現している。ファイルに対して複数の属性を付けるという点では本研究も同様である。しかし本研究では被験者の **ID**、調査日時、調査方法といった明確な属性を付与し、検索時に有効利用している点で異なる。

文献[30]はファイル名やディレクトリ名を利用して検索可能性を高めている。しかし本研究の適用分野のように、研究活動で作成されるファイルに対して的確にファイル名を付与し、適切なディレクトリ構造で分類することは困難である。

文献[34]は情報システム側からみたファイル間の関連に注目している。一方、本研究ではユーザからみたファイルに関する情報に注目して研究を行っている点で異なる。本研究

では、被験者 ID をもとにした調査データ管理を行っていた脳磁計を用いた検査チームに対して、被験者の基本情報と調査データを関係付けたデータ管理システムを提供した。本研究では問題に特化するのではなく、RDF スキーマから入力画面を作成し、ファイルからメタデータを抽出することで、汎用性を考慮した設計及び実装を行っている。

文献[12]でもファイルのコンテンツからメタデータを抽出することで、ファイル検索の効率を向上させている。本研究の被験者の基本情報と検査データの基本情報の関係のように、ファイルのコンテンツには含まれない情報もパーソナルコンピュータにおけるファイル検索には必要である。本研究では、ファイルを登録する際にそれらのメタデータを入力してもらうことで、ユーザの文脈をファイルに関係付けた。

第6章

結論

6.1. 本研究のまとめ

本論文では、新たにユーザ要件に基づく情報統合環境を提案し、コンピュータ環境上に保存されたファイルのコンテンツをユーザの要件に基づいて統合する手法について述べた。

第1章では、本研究の目的と背景について述べるとともに、論文の構成を示した。

第2章では、メタデータを用いたファイル管理研究、セマンティック Web 研究、そしてセマンティックデスクトップ研究と比較し、ユーザ要件に基づく情報統合環境の研究としての位置づけを示した。

第3章では、ユーザ要件に基づく情報統合環境の準備として、RDF コンテンツの作成手法について述べた。特に、認知症早期診断法開発現場におけるファイル群に注目した RDF コンテンツ作成手法を述べた。

第4章では、本研究における重要なアイデアであるユーザ要件に基づく情報統合環境を開発した。ユーザ要件に基づく情報統合環境は Pluggable Metadata Extractor と RDFView から構成される。Pluggable Metadata Extractor はプラグイン機構により拡張性を確保した RDF コンテンツ作成する Web サービスである。RDFView はこれまでの Web 環境からセマンティック Web 検索を実現する Web サービスである。また、RDFView を導入することでデータの独立性の確保が可能であることを確認した。

第5章では、ユーザ要件に基づく情報統合環境を実証するために、認知症早期診断法開発研究におけるファイル管理支援システムを構築した。アクティビティ図を利用した評価の結果、作業効率の向上の可能性を確認した。また、ファイル管理支援システムは実際の認知症早期診断法開発研究を実施している金沢工業大学先端電子技術応用研究所の樋口正法助教授の研究室への導入された。

本論文の成果は、以下の3つである。

- (1) これまでの Web 環境からセマンティック Web 検索を実施することの難しさを示し、セマンティック Web にビューの概念を導入したこと。具体的には、SPARQL のテンプレートを利用したセマンティック Web のビュー定義手法とビューの利用手法を提案した。
- (2) これまで単一であった RDF の作成方法をまとめて、コンピュータに保存されたファイルからセマンティック Web を構築したこと。
- (3) 上記の二つの成果を組み合わせたファイル管理支援システムを構築することで、認知症早期診断法開発現場におけるファイルの詳細な検索を可能にしたこと。

6.1.1 RDF の利用について

RDF を用いなくともメタデータを表現することは可能である。たとえば、DICOM 画像にはデータ要素というメタデータが存在し、画像情報や撮影機器に関するデータが記録されている。DICOM 画像だけを考えた検索システムであれば、わざわざ DICOM 画像のデータ要素を RDF に保存しなくても、データ要素を読み込むプログラムを用意すればよい。

しかしながらメタデータを RDF として表現することのメリットはその再利用性にある。RDF を処理可能な処理系であれば、DICOM 画像を処理できないプログラムでも RDF で記述されたデータ要素を利用したデータ処理が可能となる。汎用的な RDF を利用することは、他のメタデータと関連付けることができ、またデータを検索可能にするだけでなく、将来にわたってメタデータへの接続を実現するといった意味でも有用である。

6.2. 今後の展望

今後の展望として、以下の3点がある。

- (1) セマンティック Web のビューのあり方をより深く考察したい。名前付けグラフを利用したビューと本研究のビューの比較検討を行いたい。
- (2) メタデータの付与方法を工夫したい。たとえば OS の機能にメタデータの付与手段を組み込む、インタフェースを改良するなど、より容易に付与できる仕組みが必要

である.

- (3) ファイルから抽出した RDF の利用方法を工夫したい. それらの RDF は処理可能であり, SPARQL を用いた検索もできる. たとえば, OS の検索機能の一つに RDF を利用した検索機能を組み込む.
- (4) 語彙間の関係性を自動的に構築したい. たとえば, オントロジアライメント研究を詳しく調査して, それを実践する. 膨大な数の RDF で作られたネットワークを自動的に繋げることは重要である.

謝辞

本研究を博士論文としてまとめるにあたり，多くの方に多大なご支援を賜りました．この場を借りてお世話になった方々にお礼を申し上げます．

まず主指導教員である北陸先端科学技術大学院大学知識科学研究科の吉田武稔教授には，博士前期課程および後期課程において，研究に関して様々なご指導，ご鞭撻を賜りました．また研究環境をはじめとして様々なご支援を頂きました．深く感謝致します．

北陸先端科学技術大学院大学知識科学研究科の中森義輝教授，小坂満隆教授，金井秀明准教授，そして関西大学総合情報学部堀雅洋教授には本論文を審査して頂き，研究に関して種々の懇切なるご助言を賜りました．厚く感謝いたします．

また，金沢工業大学先端電子技術応用研究所の樋口正法教授には，認知症早期診断法開発現場におけるファイル管理支援システムの研究開発に際して，多大な協力を賜りました．心より感謝致します．

北陸先端科学技術大学院大学知識科学研究科の堀井洋助教には常日頃から研究に対する助言や議論を頂きました．また，研究活動以外の面に関しまして，大変お世話になりましたことを感謝しております．

北九州工業高等専門学校電子制御工学科の松久保潤助教には研究生活に関しての助言を頂きました．深く感謝しております．

また，いろいろと支えて頂いた友人や吉田研究室の皆様に心から感謝致します．そして，これまでの多くの困難の中，私を辛抱強く支えてくださった祖父母や母，そして妹や親戚の皆様に深く感謝します．

最後に，学部で先生であり，北陸先端科学技術大学院大学への進学のお機を作って頂いた，四日市大学環境情報学部の井岡幹博教授に感謝致します．そして，これまでお世話になった多くの人に深く感謝します．

参考文献

- [1] Tim Berners-Lee, James Hendler and Ora Lassila, The Semantic Web, Scientific American, vol.284, no.5, pp.34-43, 2001.
- [2] World Wide Consortium, W3C Semantic Web Activity, <http://www.w3.org/2001/sw/>, accessed on Oct. 16, 2008.
- [3] Frank Manola and Eric Miller, RDF Primer W3C Recommendation 10 February 2004, <http://www.w3.org/TR/rdf-primer/>, accessed on Oct. 12, 2008.
- [4] Marja-Riitta Koivunen and Eric Miller, W3C Semantic Web Activity, Proceedings of the Semantic Web Kick-off Seminar in Finland, 2001, <http://www.w3.org/2001/12/semweb-fin/w3csw>, accessed on Nov. 10, 2008.
- [5] Feigenbaum, Lee, Ivan Herman, Tonya Hongsermeier, Eric Neumann, and Susie Stephens, The Semantic Web in Action, Scientific American, vol. 297, no.6, pp. 90-97, 2007.
- [6] David Huynh, Dennis Quan, and David Karger Huynh, Haystack: A Platform for Creating, Organizing and Visualizing Information Using RDF, In Proceedings of the Semantic Web Workshop, The Eleventh World Wide Web Conference 2002 (WWW2002), Honolulu, Hawaii, USA, May 7-11, 2002.
- [7] Leo Sauermann, Ansgar Bernardi, Andreas Dengel, Overview and Outlook on the Semantic Desktop, Proceedings of the 1st Workshop on The Semantic Desktop at the ISWC 2005 Conference, Galway, Ireland, November 6-1, 2005, available at <http://www.dfki.uni-kl.de/~sauermann/papers/Sauermann+2005d.pdf>.
- [8] Stefan Decker and Martin Frank, the Social Semantic Desktop, DERI Technical Report 2004-05-02, 2004.
- [9] Jörg Richter, Max Völkel and Heiko Haller, DeepaMehta - A Semantic Desktop, Proceedings of the 1st Workshop on The Semantic Desktop at the ISWC 2005 Conference, Galway, Ireland, November 6-1, 2005, available at <http://www.deepamehta.de/ISWC-2005/deepamehta-paper-iswc2005.pdf>.
- [10] Tim Berners-Lee, Design issue: Linked Data, <http://www.w3.org/DesignIssues/LinkedData.html>, accessed on Oct. 7, 2008.

- [11] Jennifer Golbeck, Michael Grove, Bijan Parsia, Aditya Kalyanpur and James A. Hendler, New Tools for the Semantic Web, In Proceedings of the 13th international Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web (October 01 - 04, 2002), Lecture Notes in Computer Science, vol. 2473, Springer-Verlag, London, UK, pp.392-400, 2002.
- [12] David F. Huynh, David R. Karger and Robert C. Miller, Exhibit: lightweight structured data publishing, In Proceedings of the 16th international Conference on World Wide Web (Banff, Alberta, Canada, May 08 - 12, 2007), WWW '07, ACM, New York, NY, US, pp.737-746, 2007.
- [13] Charles Eames, Adding Semantics to Excel with Microformats and GRDDL, http://www.mnot.net/blog/2005/08/13/excel_microformats, accessed on Oct. 3, 2008.
- [14] Ben Adida, hGRDDL: Bridging microformats and RDFa, Web Semantics: Science, Services and Agents on the World Wide Web, vol.6, pp.54-60, 2008.
- [15] Robert Battle and Edward Benson, Bridging the semantic Web and Web 2.0 with Representational State Transfer (REST), Web Semantics: Science, Services and Agents on the World Wide Web, vol.6, pp.61-69, 2008.
- [16] David Huynh, Stefano Mazzocchi and David Karger, Piggy Bank: Experience the Semantic Web inside your web browser, Web Semantics: Science, Services and Agents on the World Wide Web vol.5, pp.16-27, 2007.
- [17] DBpedia, <http://dbpedia.org/>, accessed on Oct. 3, 2008.
- [18] Tim O'Reilly, What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software, <http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html>, accessed on Dec. 30, 2008.
- [19] Robert Battle and Edward Benson, Bridging the semantic Web and Web 2.0 with Representational State Transfer (REST), Web Semantics: Science, Services and Agents on the World Wide Web, vol.6, pp.61-69, 2008.
- [20] Anupriya Ankolekar, Markus Krotzsch, Thanh Tran, Denny Vrandećis, The two cultures: Mashing up Web 2.0 and the Semantic Web, Web Semantics: Science, Services and Agents on the World Wide Web, vol.6, pp.70-75, 2008.

- [21] Eric Prud'hommeaux and Andy Seaborne, SPARQL Query Language for RDF W3C Recommendation 15 January 2008, <http://www.w3.org/TR/rdf-sparql-query/>, accessed on Oct.19, 2008.
- [22] Nigel Shadbolt, Tim Berners-Lee and Wendy Hall, The Semantic Web Revisited, IEEE Intelligent Systems, vol.21, no.3, pp.96-101, 2006.
- [23] 武田 英明, 上位オントロジー, 人工知能学会誌, vol.19, no.2, pp.172-186, 2004.
- [24] Aditya Kalyanpur, Bijan Parsia, Evren Sirin, Bernardo Cuenca Grau and James Hendler, Swoop: A Web Ontology Editing Browser, Web Semantics: Science, Services and Agents on the World Wide Web, vol.4, pp.144-153, 2006.
- [25] Protégé Ontology Editor and Knowledge Acquisition System, <http://protege.stanford.edu/>, accessed on Nov. 10, 2008.
- [26] Pavel Shvaiko, Jerome Euzenat, Ten Challenges for Ontology Matching, Ten Challenges for Ontology Matching. Technical Report DISI-08-042, Ingegneria e Scienza dell'Informazione, University of Trento, 2008.
- [27] 都司達夫, 宝珍輝尚: IT TEXT データベース技術教科書, CQ出版社.
- [28] Stonebraker, M: Implementation of Integrity Constraints and Views by Query Modification, Proc. ACM SIGMOD, pp.65-78, 1975.
- [29] David K. Gifford, Pierre Jouvelot, Mark A. Sheldon and James W. O'Toole, Jr., Semantic File Systems, In Proceedings of the thirteenth ACM symposium on Operating systems principles (Pacific Grove, California, United States), ACM, New York, NY, USA, pp.16-25, 1991.
- [30] 阿部 淳也, 出石 大志, 杉上 裕一, 堀 幸雄, 今井 慈郎, タグ情報に基づくファイル管理システム, 情報処理学会研究報告「データベースシステム」, vol..2007, no.6, pp.97-102, 2007.
- [31] 轟木 伸俊, 多田 知正, 樋口 昌宏, 谷口 健一, 階層的キーワードに基づく名前管理手法とそれに基づくファイル共有手法, 情報処理学会研究報告「グループウェアとネットワークサービス」, vol..2000, no.97, pp.49-54, 2000.
- [32] Stephan Bloehdorn, Olaf Goerlitz, Simon Schenk and Max Voelkel, TagFS - Tag Semantics for Hierarchical File Systems, In Proceedings of the 6th International Conference on Knowledge Management (I-KNOW 06), Graz, Austria, September 6-8, 2006.

- [33] Alexander Ames, Nikhil Bobb, Scott A. Brandt, Adam Hiatt, Carlos Maltzahn, Ethan L. Miller, Alisa Neeman and Deepa Tuteja, Richer File System Metadata Using Links and Attributes, In Proceedings of the 22nd IEEE / 13th NASA Goddard Conference on Mass Storage Systems and Technologies, IEEE Computer Society, Washington, DC, USA, pp. 49-60, 2005.
- [34] 三森 裕一郎, 森嶋 厚行, メタデータを利用した高度ファイル操作のためのミドルウェアの提案, 情報処理学会研究報告「データベースシステム」vol.2007, no. 65, pp. 189-194, 2007.
- [35] Paul - Alexandru Chirita, Rita Gavriloaie, Stefania Ghita, Wolfgang Nejdl, and Raluca Paiu, Activity Based Metadata for Semantic Desktop Search, In Proceedings of the 2nd European Semantic Web Conference, Heraklion, Greece, May 29- June 1, 2005.
- [36] Onne Gorter, Database File System An Alternative to Hierarchy Based File Systems, Master's thesis, University of Twente, 2004.
- [37] Zhichen Xu, Magnus Karlsson, Chunqiang Tang, Christos Karamanolis, Towards a Semantic-Aware File Store, In Proceedings of the 9th Conference on Hot Topics in Operating Systems - Volume 9 (Lihue, Hawaii, May 18 - 21, 2003), USENIX Association, Berkeley, CA, USA, pp.31-31, 2003.
- [38] Google, <http://www.google.co.jp>, accessed on Oct.3, 2008.
- [39] Yahoo!検索, <http://search.yahoo.co.jp/>, accessed on Oct.3, 2008.
- [40] Live Search, <http://search.live.com/>, accessed on Oct. 3, 2008.
- [41] Powerset, <http://www.powerset.com/>, accessed on Oct. 7, 2008.
- [42] Wikipedia, <http://en.wikipedia.org>, accessed on Oct. 7, 2008.
- [43] 浦本 直彦, Webにおける情報統合-セマンティックWebとWebサービス, IPSJ Magazine, vol.44, no.7, pp.707-712, 2003.
- [44] Dan Connolly, Gleaning Resource Descriptions from Dialects of Languages (GRDDL) W3C Recommendation 11 September 2007, <http://www.w3.org/TR/2007/REC-grddl-20070911/>, accessed on Nov. 10, 2008.
- [45] Aimilia Magkanaraki, Val Tannen, Vassilis Christophides and Dimitris Plexousakis, Viewing the semantic web through RVL lenses, Web Semantics: Science, Services and Agents on the World Wide Web, vol.1, no.4, pp.359-375, 2004.

- [46] Simon Schenk and Steffen Staab, Networked Graphs: A Declarative Mechanism for SPARQL Rules, SPARQL Views and RDF Data Integration on the Web, WWW '08: Proceeding of the 17th international conference on World Wide Web(Beijing, China, April 21-25, 2008), pp.585-594, 2008.
- [47] Amazon web services, <http://aws.amazon.com/>, accessed on Oct. 8, 2008.
- [48] Google AJAX Search API, <http://code.google.com/intl/ja/apis/ajaxsearch/>, accessed on Oct. 8, 2008.
- [49] Google Maps API ,<http://code.google.com/intl/ja/apis/maps/>, accessed on Oct. 8, 2008.
- [50] Pipes: Rewrite the web, <http://pipes.yahoo.com/pipes/>, accessed on Oct. 8, 2008.
- [51] Dublin Core Metadata Initiative,Dublin Core Metadata Element Set, Version 1.1, <http://dublincore.org/documents/dces/>, accessed on Nov. 10, 2008.
- [52] Googleデスクトップ, <http://desktop.google.com/ja/>, accessed on Oct. 3, 2008.
- [53] Windowsデスクトップサーチ,
<http://www.microsoft.com/japan/windows/desktopsearch/default.mspx>, accessed on Oct.3, 2008.
- [54] Beagle, http://beagle-project.org/Main_Page, accessed on Oct.3, 2008.
- [55] National Electrical Manufacturers Association, Digital Imaging and Communications inMedicine (DICOM) , <http://medical.nema.org/>, accessed on Nov.10, 2008.
- [56] K-PACS, <http://www.k-pacs.de/index.html>, accessed on Oct.3, 2008.
- [57] Conquest DICOM software, <http://www.xs4all.nl/~ingenium/dicom.html>, accessed on Oct.3, 2008.
- [58] National Electrical Manufacturers Association. Digital Imaging and Communications in Medicine (DICOM) Part 6: Data Dictionary. PS 3.6-2007 edition, 2007.
- [59] Delicious, <http://delicious.com>, accessed on Oct.3, 2008.
- [60] フリー百科事典『ウィキペディア (Wikipedia)』:フォークソノミー,
<http://ja.wikipedia.org/wiki/フォークソノミー>, accessed on Oct.3, 2008.
- [61] Adobe Extensible Metadata Platform (XMP), <http://www.adobe.com/xmp>, accessed on Oct. 8, 2008.
- [62] ADOBE SYSTEMS INCORPORATED, XMP Specification September 2005, 2005.

- [63] RDF/XML Syntax Specification (Revised) W3C Recommendation 10 February 2004,
<http://www.w3.org/TR/rdf-syntax-grammar/>, accessed on Nov. 9, 2008.
- [64] バイナリエディタ Bz ver 1.62 ,<http://www.zob.ne.jp/~c.mos/soft/bz.html>, accessed on
Nov. 9, 2008.
- [65] 独立行政法人 情報処理推進機構, 医療分野における情報技術ロードマップ, 2005,
available at <http://www.ipa.go.jp/SPC/data/200401/200401bb.pdf>.
- [66] 鈴木 博道, 清水 昇, 足立 和夫, セマンティックWeb を用いたMedDRA オントロ
ジの開発とその利用方法の研究, 医療情報学, vol.25, no.6, pp.447-455, 2005.
- [67] Silvio Carro Antonio and Jacob Scharcanski, A framework for medical visual
information exchange on the WEB, Computers in Biology and Medicine,
vol.36,no.4,pp. 327-338, 2006.
- [68] Maged N. Kamel Boulos, AbdulV. Roudsari, Ewart R. Carson, Towards a semantic
medical Web: HealthCyberMap's tool for building an RDF metadata base of health
information resources based on the qualified Dublin Core metadata set, Med Sci
Monit, vol.8, no.7, pp.24-36, 2002.
- [69] HTML 4.01 Specification: metadata profiles,
<http://www.w3.org/TR/1999/REC-html401-19991224/struct/global.html#profiles>, acce
ssed on Nov. 8, 2008.
- [70] RFC 1738: Uniform Resource Locators (URL), <http://www.ietf.org/rfc/rfc1738.txt>,
accessed on Nov. 8, 2008.
- [71] 内田 治, 醍醐 朝美, 実践 アンケート調査入門, 日本経済新聞社, 2003.
- [72] Jena – A Semantic Web Framework for Java, <http://jena.sourceforge.net/>, accessed
on Nov. 10, 2008.
- [73] ImageJ: Imaging Processing and Analyzing in Java, <http://rsb.info.nih.gov/ij/>,
accessed on Oct.3, 2008.
- [74] Apache POI - Java API To Access Microsoft Format Files ,<http://poi.apache.org/>,
accessed on Oct.3,2008.
- [75] 厚生労働保険局, 診療報酬情報提供サービス : 医薬品マスター,2006.
- [76] 日本医師会, 併用禁忌データベース,
<http://www.orca.med.or.jp/orca/tec/heiyoukinkidb/heiyoukinkidb.rhtml>, accessed on
Nov. 10, 2008.
- [77] Chimezie Ogbuji, GRDDL Test Cases W3C Recommendation 11 September 2007,
<http://www.w3.org/TR/grddl-tests/>, accessed on Nov. 10, 2008.

- [78] GRDDL Standard Transform Library ,<http://www.w3.org/2001/sw/grddl-wg/library>, accessed on Oct. 19, 2008.
- [79] GRDDL Test Cases ,P3P
work-alike ,<http://www.w3.org/TR/grddl-tests/#xmlWithGrddlAttribute>, accessed on Oct. 19, 2008.
- [80] 川上 洋一, 松村 泰志, 笹井 浩介, 安永 晋, 稲田 紘, 木内 貴弘, 黒田 知宏, 坂本 憲広, 竹村 匡正, 田中 博, 玉川 裕夫, 仲野 俊成, 朴 勤植, 平松 治彦, 宮本 正喜, レポーティングシステムにおけるRDFの応用, 医療情報学 , vol.25, no.6, pp. 421-429, 2005.
- [81] Babel,<http://simile.mit.edu/babel/>, accessed on Oct. 8, 2008.
- [82] Microformats, <http://microformats.org/>, accessed on Oct. 8, 2008.
- [83] Christian Bizer, Andy Seaborne : D2RQ – Treating Non-RDF Databases as Virtual RDF Graphs , Third International Semantic Web Conference, 2004.
- [84] Johan Petrini and Tore Rich, Processing queries over RDF views of wrapped relational databases , In Proceedings of the 1st International Workshop on Wrapper Techniques for Legacy Systems (WRAP2004), Delft, Holland, 2004.
- [85] Andy Seaborne, Geetha Manjunath, Chris Bizer, John Breslin, Souripriya Das, Ian Davis, Steve Harris, Kingsley Idehen, Olivier Corby, Kjetil Kjernsmo and Benjamin Nowack. SPARQL Update A language for updating RDF graphs W3C Member Submission 15 July 2008,<http://www.w3.org/Submission/2008/SUBM-SPARQL-Update-20080715/>, accessed on Nov. 10, 2008.
- [86] World Health Organization, International Classification of Diseases(ICD), <http://www.who.int/classifications/icd/en/>, accessed on Oct. 13, 2008.
- [87] SNOMED, SNOMED Clinical Terms, <http://www.snomed.org/snomedct/>, accessed on Oct. 13, 2008.
- [88] Health Layer 7, <http://www.hl7.org/>, accessed on Nov. 10, 2008.
- [89] MedXML Consortium, Medical Markup Language(MML), <http://www.medxml.net/>, accessed on Oct. 13, 2008.
- [90] National Library of Medicine, Unified Medical Language System(UMLS), <http://www.nlm.nih.gov/research/umls/>, accessed on Oct. 13, 2008.

- [91] 田中 昌昭, 施設間での医療情報の交換と共有を実現するための用語の標準化に対する一提案, 医療情報学, vol.21, no.1, pp.3-11, 2001.
- [92] 原 正一郎, 杉森 裕樹, 古海 勝彦, 東福寺 鰯夫, 窪寺 健, 河合 正樹, 吉田 勝美, 健診情報のための電子的交換規約 Health Data Markup Language (HDML), 情報知識学会誌, vol.12, no.4, pp.32-52, 2002.
- [93] 橋本 英明, 岡田 美保子, 土光 智子, 大井田 隆, 保健医療統計データ要素の登録管理と共有化の方法に関する研究, 医療情報学, vol.21, no.1, 2001.
- [94] Centers for Disease Control and Prevention (CDC), Epi Info, <http://www.cdc.gov/epiinfo/>, accessed on Nov. 10, 2008.
- [95] 坂本 尚子, 森 康真, 北上 始, 階層的管理機能を持つアンケート実施支援システムの構築, 情報処理学会研究報告「コンピュータと教育」, vol.75, no.6, 2004.
- [96] Anders Tornqvist, Chris Nelson, and Mats Johnsson, XML and Objects-The Future for E-Forms on the Web, In Proceedings of the 8th Workshop on Enabling Technologies on infrastructure For Collaborative Enterprises, IEEE Computer Society, pp.303-308, 1999.
- [97] 楯 武士, XML を用いたアンケートシステム, 情報処理学会研究報告「デジタル・ドキュメント」, vol.25, no.2, pp.9-16, 2000.
- [98] 田村 直之, 重村 哲至, プラグイン可能なアンケートサーバの作成, 徳山工業高等専門学校研究紀要2004, vol.28, pp.63-68, 2004.
- [99] 三船 真哲, 山田 嶺, 澤津 健吾, 小坂 慶和, 綿貫 理明, 飯田 周作, P2P 技術を利用したアンケート集計システムの設計と開発, 情報処理学会研究報告「情報システムと社会環境」 vol.87, no.6, pp.39-46, 2004.
- [100] John M. Boyer, XForms 1.0 (Third Edition) W3C Recommendation 29 October 2007, <http://www.w3.org/TR/xforms/>, accessed on Nov. 10, 2008.
- [101] 久保 裕也, 玉村 雅敏, 木幡 敬史, 金子 郁容, カスタマイズ可能な調査スキーマの共有による学校評価支援, 情報処理学会論文誌, vol.46, no.1, pp.172-186, 2005.
- [102] 横川 典子, 菊池 浩明, 村井 順, 電子匿名アンケート機構の設計と実装, 情報処理学会研究報告「マルチメディア通信と分散処理」, vol.75, no.13, pp.73-78, 1996.
- [103] 中里 純二, 藤本 賢次, 菊池 浩明, 個人情報漏洩を防止するWeb アンケートのセキュリティ強化, 情報処理学会論文誌, vol.26, no.8, pp.2068-2077, 2005.
- [104] 北川 隆, 岡 博文, 楫 勇一, 大学における講義評価のための匿名アンケートプロトコルとその試作, 情報処理学会論文誌, vol.44, no.9, pp.2353-2363, 2003.

- [105] 相川 勇之, 伊藤 山彦, 高山 泰博, 鈴木 克志, 今村 誠, 概念抽出型テキストマイニングによるアンケート分析手法の提案, 情報処理学会研究報告「デジタル・ドキュメント」, vol.38,no.1,pp.1-6, 2003.
- [106] Embedded RDF Wiki, <http://research.talis.com/2005/erdf/wiki>, accessed on Oct. 14, 2008.
- [107] RDFa in XHTML, <http://www.w3.org/TR/rdfa-syntax/>, accessed on Oct. 14, 2008.
- [108] RDFizers, <http://simile.mit.edu/wiki/RDFizers>, accessed on Oct. 14, 2008.
- [109] ConverterToRDF, <http://esw.w3.org/topic/ConverterToRdf>, accessed on Oct. 14, 2008.
- [110] Roy Thomas Fielding , Architectural Styles and the Design of Network-Based Software Architectures, doctoral dissertation, School of Information & Computer Science, University of California, Irvine, 2000.
- [111] Apache Velocity, <http://velocity.apache.org/>, accessed on Oct. 20, 2008.
- [112] MySQL, <http://www-jp.mysql.com/>, accessed on Oct. 20, 2008.
- [113] Babel, <http://simile.mit.edu/babel/>, accessed on Oct. 8, 2008.
- [114] Google Web Toolkit, <http://code.google.com/webtoolkit/>, accessed on Nov. 11, 2008.
- [115] Eclipse Metrics Plug-in, <http://www.stateofflow.com/projects/16/eclipsemetrics>, accessed on Dec. 29, 2008.
- [116] ckjm — Chidamber and Kemerer Java Metrics, <http://www.spinellis.gr/sw/ckjm/>, accessed on Dec. 29, 2008.
- [117] Thomas J. McCabe, A complexity measure, IEEE Trans. on Software Engineering, Vol. SE-8, no.4, pp.308-320, 1976.
- [118] Shyam R. Chidamber and Chris F. Kemerer, A Metrics Suite for Object Oriented Design, IEEE Transactions on Software Engineering, Vol.20, No.6, pp.476-493, June. 1994.
- [119] Tim Berners-Lee, Notation3 (N3) a readable RDF syntax, <http://www.w3.org/DesignIssues/Notation3.html>, accessed on Dec. 30, 2008.
- [120] Dave Beckett, N-Triples, <http://www.dajobe.org/2001/06/ntriples/>, accessed on Dec. 30, 2008.
- [121] Dave Beckett, Turtle - Terse RDF Triple Language, <http://www.dajobe.org/2004/01/turtle/>, accessed on Dec. 30, 2008.

学位論文に関する発表論文

[査読付き論文]

- 林 正治, 堀井 洋, 権 仁洙, 吉田 武稔, Semantic Web技術を応用した質問票調査支援情報システムの構築, 医療情報学, vol.27, no.1, pp.109-116, 2007.
- 堀井 洋, 林 正治, 権 仁洙, 吉田 武稔, メタデータ照合型ネットワーク解析システム'MANACO'を用いた医療情報通信観測に関する提案, 医療情報学, vol.27, no.3, pp.321-328, 2007.
- 堀井 洋, 林 正治, 沢田 史子, 堀井 美里, 吉田 武俊, デジタルコンテンツ化による歴史資料活用の提案～歴史資料「梅田日記」を事例として～, 観光と情報, vol.4, no.1, pp.27-34, 2008.
- 林 正治, 堀井 洋, 権 仁洙, 吉田 武稔, 認知症診断のためのメタデータを用いたデータ管理システムの提案, 医療情報学 (投稿中).

[一般論文・口頭発表]

- 林 正治, 堀井 洋, 権 仁洙, 吉田 武稔, CMS を用いた地域医療ポータルサイトの提案, 第4回情報科学技術フォーラム (FIT2005) 講演論文集, 2005.
- 林 正治, 中川 好貴, 堀井 洋, 権 仁洙, 吉田 武稔, 意味に基づく情報統合システムの提案, 平成17年度電気関係学会北陸支部連合大会 講演論文集, 2005.
- 中川 好貴, 林 正治, 堀井 洋, 権 仁洙, 吉田 武稔, 地域医療連携コミュニケーションを支援する情報システム構築, 平成17年度電気関係学会北陸支部連合大会 講演論文集, 2005.
- Y. Wang, H. Xia, M. Hayashi, T. Yoshida, Adding semantic support to Web services in distribution logistics, 2nd IEEE Conference on Service Systems and Service Management, Chongqing, China, 13-15 June, 2005. (査読あり)
- 林 正治, 堀井 洋, 権 仁洙, 吉田 武稔, 医療情報コードデータベースを用いたメタデータ生成システム, 第5回情報科学技術フォーラム (FIT2006) 講演論文集, 2006.

- 堀井 洋, 林 正治, 権 仁洙, 吉田 武稔, 地域医療情報ネットワークを対象にしたメタデータ照合型医療情報通信監視システムの構築, 第5回情報科学技術フォーラム(FIT2006)講演論文集, 2006.
- 堀井 洋, 林 正治, DONG JINGE, 宮田 諭, 権 仁洙, 吉田 武稔, メタデータ照合型医療情報通信可視化技術に関する提案, 日本バーチャルリアリティ学会 第11回大会講演論文集, 2006.
- 林 正治, 堀井 洋, 権 仁洙, 吉田 武稔, Semantic Web の概念を適用した疫学調査支援情報システムの構築, 第26回医療情報学連合大会講演論文集, 2006. (査読あり)
- 堀井 洋, 林 正治, 権 仁洙, 吉田 武稔, メタデータ照合型医療情報通信監視システムの構築, 第26回医療情報学連合大会講演論文集, 2006. (査読あり)
- 堀井 洋, 林 正治, 権 仁洙, 吉田 武稔, メタデータ照合型ネットワーク解析システム MANACO を用いた医療情報通信観測手法の提案, 情報処理学会第69回全国大会, 2007.
- 林 正治, 堀井 洋, 権 仁洙, 吉田 武稔, Semantic Web 技術を応用した質問票調査支援情報システムの構築, 第11回日本医療情報学会春季学術大会, 2007. (査読あり)
- 堀井 洋, 林 正治, 権 仁洙, 吉田 武稔, メタデータ照合型ネットワーク解析システム'MANACO'を用いた医療情報通信観測に関する提案, 第11回日本医療情報学会春季学術大会, 2007. (査読あり)
- 林 正治, 堀井 洋, 権 仁洙, 吉田 武稔, 認知症のためのメタデータを用いたデータ管理システムの提案, 第12回医療情報学会春期学術大会, 2008. (査読あり)
- 林 正治, 堀井 洋, 権 仁洙, 吉田 武稔, RDFView:メタデータ活用を目的とした情報システム, 第70回情報処理学会全国大会, 2008.
- Masaharu Hayashi, Hiroshi Horii, Insoo Kweon, Taketoshi Yoshida, A Medical Information Management System using The Semantic Web Technology, The 4th International Conference on Networked Computing and Advances in Information Management (NCM2008), Gyeongju, Korea, 2-4 Sep., 2008. (査読あり)

[その他の論文]

- T. Yoshida, H. Horii, M. Hayashi, I. Kweon, T. Inuzuka, A Study of the Relations Between Soft Systems Methodology and Organizational Knowledge Creation Theory. Int. J. of Knowledge Systems Sciences, vol.1, no.1, pp. 56-62, 2004.

付録 A

実験には Berlin SPARQL Benchmark (BSBM)のデータジェネレータを利用して作成した RDF コンテンツを用いた。データジェネレータのパラメータは次の通りである。

➤ `generator -s nt -pc 100`

BSBM は <http://www4.wiwiss.fu-berlin.de/bizer/BerlinSPARQLBenchmark/>から入手できる。

A.1 基準問い合わせ文

```
PREFIX bsbm-inst: <http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/>
PREFIX bsbm: <http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/vocabulary/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

select ?product_label ?producer_label ?country
where {
  ?product rdf:type <http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/ProductType6> .
  ?product rdfs:label ?product_label .
  ?product bsbm:producer ?producer .
  ?producer rdfs:label ?producer_label .
  ?producer bsbm:country ?country .
}
```

A.2 条件を追加した問い合わせ文

```
PREFIX bsbm-inst: <http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/>
PREFIX bsbm: <http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/vocabulary/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

select ?product_label ?producer_label ?country
where {
  ?product rdf:type bsbm-inst:ProductType6 .
  ?product rdfs:label ?product_label .
  ?product bsbm:producer ?producer .
  ?producer rdfs:label ?producer_label .
  ?producer bsbm:country ?country .
  filter (?country =<http://download.org/rdf/iso-3166/countries#US>).
}
```

A.3 入力値を必要とする問い合わせ文

```
PREFIX bsbm-inst: <http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/instances/>
PREFIX bsbm: <http://www4.wiwiss.fu-berlin.de/bizer/bsbm/v01/vocabulary/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

select ?product_label ?producer_label ?country
where {
  ?product rdf:type bsbm-inst:ProductType6 .
  ?product rdfs:label ?product_label .
  ?product bsbm:producer ?producer .
  ?producer rdfs:label ?producer_label .
  ?producer bsbm:country ?country .
  filter (?country =<入力値>).
}
```

A.4 実験プログラム RDFView(処理部分のみ)

```
ResultSet results = ResultSetFactory.load("http://localhost/MedSW/RDFView/berlin/query1");
while (results.hasNext()) {
    QuerySolution sol = results.nextSolution();
    System.out.println(sol.get("product_label"));
    System.out.println(sol.get("producer_label"));
    System.out.println(sol.get("country"));
}
```

A.5 実験プログラム SPARQL Endpoint(処理部分のみ)

```
String strQuery = "";
Query query = QueryFactory.create(strQuery, "", Syntax.syntaxSPARQL);
QueryExecution queryExe = QueryExecutionFactory
    .sparqlService("http://localhost/MedSW/servlet/SparqlSrv", query);
ResultSet res = queryExe.execSelect();
while (res.hasNext()) {
    QuerySolution sol = res.nextSolution();
    System.out.println(sol.get("product_label"));
    System.out.println(sol.get("producer_label"));
    System.out.println(sol.get("country"));
}
```

A.6 実験プログラム JenaDB (処理部分のみ)

```
String strQuery = "";

Class.forName("com.mysql.jdbc.Driver");

DBConnection conn = new DBConnection(

    "jdbc:mysql://localhost/medsw?useUnicode=true&characterEncoding=UTF-8",

    "jena_user", "jena_user!", "MySQL");

Model model = ModelRDB.open(conn);

Query query = QueryFactory.create(strQuery, "", Syntax.syntaxSPARQL);

DataSource ds = DatasetFactory.create(model);

QueryExecution queryExe = QueryExecutionFactory .create(query, ds);

ResultSet res = queryExe.execSelect();

while (res.hasNext()) {

    QuerySolution sol = res.nextSolution();

    System.out.println(sol.get("product_label"));

    System.out.println(sol.get("producer_label"));

    System.out.println(sol.get("country"));

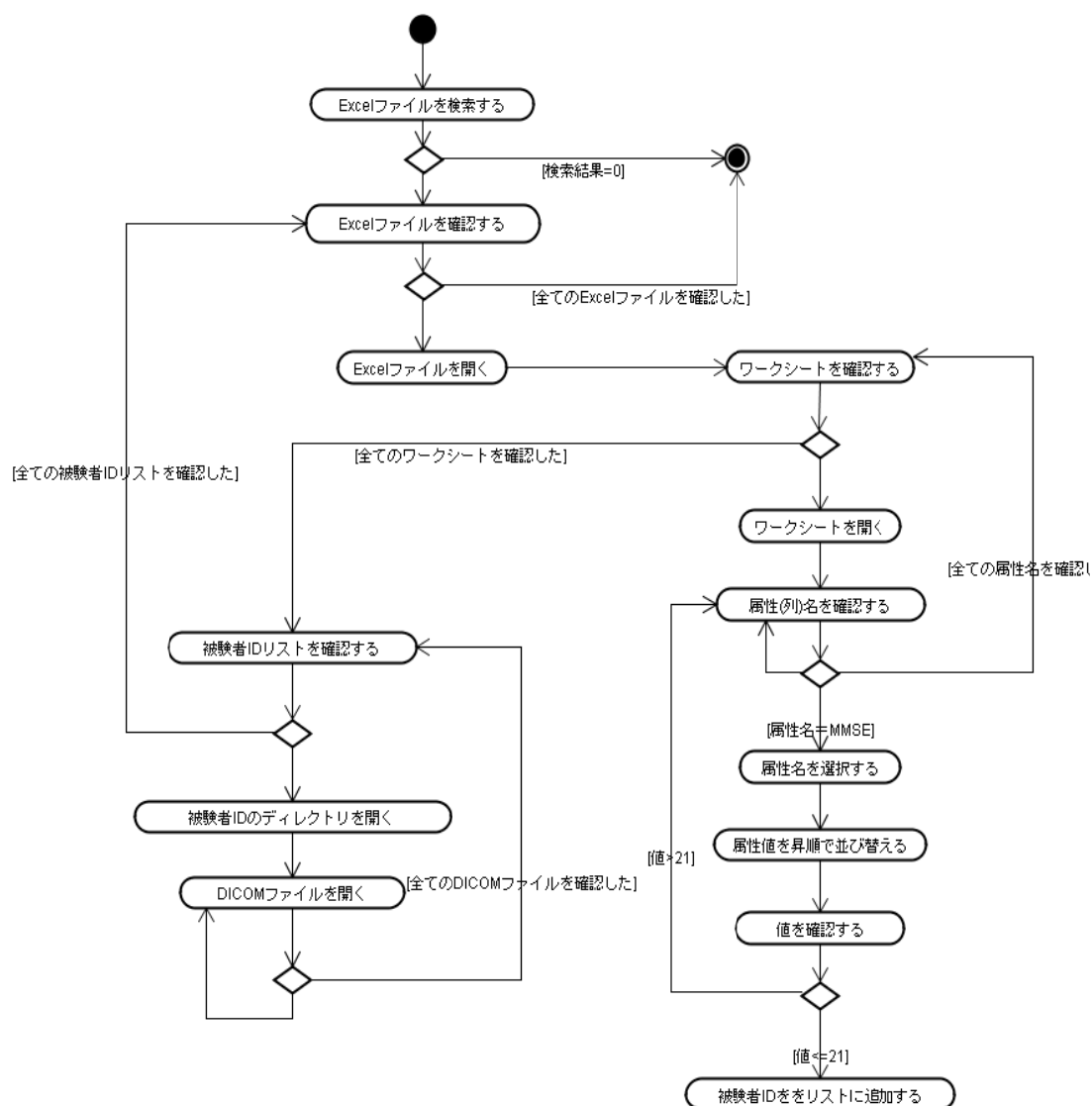
}

conn.close();
```

付録 B

ここでは、評価に利用した「MMSE スコア 21 以下の被験者の DICOM ファイルを開く」のアクティビティ図を示す。

B.1 ファイル管理支援システム導入前



B.2 ファイル管理支援システム導入後

