人工知能学会研究会資料
SIG-SWO-A1103-01

コレクション型ウェブサイトからLinked Dataを構築するプラットフォームの開発

Platform for creating Linked Data from web sites including collections

加藤 文彦[1*]　嘉村哲郎[3,4]　大向一輝[2,3]　武田英明[2,3]
Fumihiro Kato,[1]　Tetsuro Kamura,[3,4]　Ikki Ohmukai,[2,3]　Hideaki Takeda[2,3]

[1] 情報・システム研究機構 新領域融合研究センター
[1] Research Organization of Information and Systems
[2] 国立情報学研究所
[2] National Institute of Informatics
[3] 総合研究大学院大学
[3] The Graduate University for Advanced Studies
[4] 東京藝術大学
[4] Tokyo University of the Arts

Abstract: LODAC Distiller is a platform to transform HTML files of web sites into RDF files to be able to provide Linked Data. LODAC Distiller extracts pairs of a key and a value such as a collection's title or an author's name from HTML files, then it maps the pairs to appropriate RDF properties and objects by using original mapping rules. LODAC Museum uses LODAC Distiller for generating museum collection datasets from various museum web sites.

1　はじめに

ウェブ上で様々な情報源のデータを型付リンクによって結びつける，Linked Data [1, 2] というコミュニティ活動が注目されている．Linked Data は DBpedia [3] と呼ばれる Wikipedia から変換した情報を中心として，地理情報，政府情報，学術情報，音楽情報，写真情報など，2011 年 9 月の時点で 295 のデータセットによるネットワークが形成されている．セマンティックウェブが普及しない原因の一つとして基盤となるデータの欠如があったが，Linked Data はその問題を解決する有望な手段の一つであると考えられている．

しかしながら，現在広く利用されているウェブサイトの大半は Linked Data として扱えない現状がある．そのため，ウェブサイト上にあるデータをどのように Linked Data として取り扱えるようにするかを検討する必要がある．最初のきっかけとなった DBpedia 自体も Wikipedia というウェブサイトからデータを抽出して Linked Data にしたものである．しかし，ウェブサイト毎に独自の変換プログラムを書いて Linked Data を作成するのではスケールしないので，汎用的な方法が必要である．

そこで博物館の収蔵品データベースやメーカーの商品カタログのように，ある‘モノ’に対する説明がキーとバリューの対として記述されているウェブサイトに注目することにした．本稿では，この類のウェブサイトをコレクション型ウェブサイトと呼ぶことにする．コレクション型ウェブサイトで扱っているデータは，‘モノ’-キー-バリューで表現できるため，主語-述語-目的語の 3 つ組で表現する RDF モデルに適用することが，他のウェブサイトを対象とするよりも容易であると考えられる．

著者らは上記の考えを基に，コレクション型ウェブサイトから Linked Data を作成するためのシステム LODAC Distiller を構築している．本稿では LODAC Distiller の概要について説明を行う．現在 LODAC Dis-

*連絡先：国立情報学研究所
〒 101-8430 東京都千代田区一ツ橋 2-1-2
E-mail: fumi@nii.ac.jp

tiller は，主に LODAC Museum [4] で博物館の収蔵品データ作成のために使用している．

2 関連研究

ウェブ上のデータを Linked Data として扱えるようにするためのツールやシステムは多数存在する．CSV, Excel といったデータを RDF に変換することを目的としているツールから，Linked Data として公開を視野にいれているツールなど様々である．

CSV や Excel のようなデータを変換するツールは RDF Extension for Google Refine [1] や irON [2]などがある．これらについての調査は以前 [5] にて行ったので参照されたい．

データ格納用のソフトウェアが変換ツールを提供している場合もある．例えば Virtuoso Universal Server [3]は最も人気のある RDF Store の一つだが，一部のウェブサイトを RDF に変換して Virtuoso に取り込む XSLT 群を Virtuoso Sponger [6] として提供している．

その他に，ウェブ上で公開されている RDF データを統合するためのフレームワーク例としては，LDIF (Linked Data Integration Framework) [7] がある．LDIF は Linked Data となっているサイトや RDFa が埋め込まれているウェブサイトのように，元々RDF を提供しているウェブサイトのみを対象としている．それらから集めてきたデータを共通のボキャブラリにマッピングしたり，同定作業をしてデータ統合を自動化することを主な目的としている．

3 LODAC Distiller

本節では，コレクション型ウェブサイトから Linked Data を作成するためのシステムである LODAC Distiller について述べる．

3.1 処理の流れ

コレクション型ウェブサイトから Linked Data を作成するまでの一連の流れを図 1 に示す．

ウェブサイトからの HTML 収集と HTML からのキー・バリューの抽出には，現在 Apache Nutch [4]と Ruby を併用している．抽出されたキー・バリューは Apache Solr [5]に ‘モノ’ 単位で保存されるようになっている．Nutch は Solr との相性が良いため，段階的に

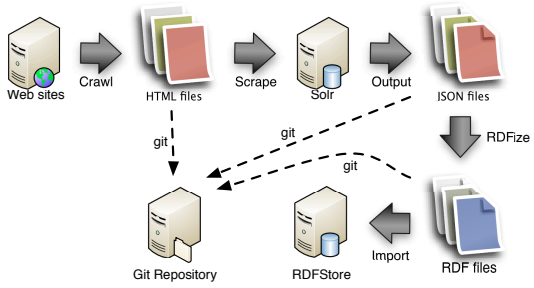

図 1: 処理の流れ

Ruby から Nutch に移行している．Solr に保存されているデータには，ウェブサイトから取得されたデータ以外に対象ウェブサイトの情報や取得時刻，サイト内での ‘モノ’ の ID など，直接抽出したキー・バリューデータ以外のデータも含んでいる．

Solr に蓄積されているデータは単純な JSON 形式で出力されるようになっている．出力される JSON の例をソースコード 1 に示す．Solr 上のデータを API 経由で直接扱うことも可能であるが，3.2 節で述べる通り，データから RDF へのマッピング作業を Solr から切り離して行う必要があった．

```
{
    "institution": "akita_museum",
    "segment": "20110512142450",
    "digest": "30d5aaf43964c64315f832f617a2e2d1",
    "tstamp": "20110512052500660",
    "作品名": "秋田萬古緑釉蓮湯ざまし",
    "anchor": "0.html",
    "title": "秋田市立千秋美術館 - 収蔵品検索",
    "institutionalId": "0",
    "寄贈・購入年": "昭63年度寄贈",
    "作者名と生没年": "鈴木緑園
        1856(安政03)-1910(明治43)",
    "作者名（かな）": "すずき　りょくえん",
    "imgURL": "http://www.city.akita.akita.jp/city/
        ed/ss/senshu-art/data/jpeg/1465.jpg",
.......
```

ソースコード 1: 抽出データ例

出力された JSON に含まれているキー・バリューのデータから必要な項目を RDF の 3 つ組に対応付けるために，独自のマッピングルールを定義している．マッピングルールは対象のウェブサイト毎に記述する．RDF 変換プロセッサにマッピングルールを渡すことで，JSON ファイルから RDF ファイルへと変換を行う．マッピングルールについては 3.3 節で解説する．

最後に，上記の手順で生成された RDF ファイルを RDF Store に格納する．これにより，RDF Store が提供する SPARQL Endpoint を通して，生成されたデータを扱えるようになる．LODAC Museum では RDF Store として OWLIM[6]を用いており，Linked Data の提供は内部で OWLIM に SPARQL で問い合わせることで実現している．

[1] http://lab.linkeddata.deri.ie/2010/grefine-rdf-extension/
[2] http://openstructs.org/iron
[3] http://virtuoso.openlinksw.com
[4] http://nutch.apache.org/
[5] http://lucene.apache.org/solr/
[6] http://www.ontotext.com/owlim

3.2 データ抽出と RDF マッピングの分離

Virtuoso Sponger のように HTML から直接 RDF へマッピングする方法とは異なり，LODAC Distiller では HTML からのキー・バリュー抽出と RDF へのマッピングのプロセスを分けて，独立して行えるようにしている．理由は，各々の準備作業を異なる人が行うことを想定しているためである．

HTML からのデータ抽出の準備は，HTML からキー・バリューとなるようなデータを探しだして，その場所に対応する xpath を nutch の設定に書くという手順になる．この作業はウェブ開発者が行うことを想定しており，キーやバリューに含まれている言葉や文章自体に深い知識を必要としない．また将来的には機械処理でデータ抽出を行うことも視野にいれている

一方，抽出したキー・バリューのペアから RDF へのマッピングについては，単語や文章がそのドメインの専門家にしか正確な意味がわからないような，難しいものの場合がある．更にそれがキーである場合，対応する RDF 述語が何であるかの判断もしなければならないが，それは専門家のほうが正しく行えると考えられる．そのため，対象のドメインを良く知る専門家がマッピング作業だけを担当できるようにすべきである．

データ抽出と RDF へのマッピングを同時に行うにはウェブ開発と専門領域双方に精通している必要があり，実際に作業を行える人が限られていた．しかしこの処理を分離した結果，LODAC Museum においてはキー・バリュー抽出をウェブ開発者，RDF へのマッピングを博物館メタデータの専門家にまかせることで，並行して作業を進めることができるようになった．

3.3 RDF マッピングルール

キー・バリュー形式で格納されているデータを RDF に変換するために，独自のマッピングルールを定義している．秋田市立千秋美術館の収蔵品データに対するマッピングルール例をソースコード 2 に示す．

マッピングルールは，JSON 形式で RDF 述語とキーの対応関係を記述する．それにより，JSON に含まれているキー・バリューのペアを RDF 述語と目的語の関係に変換する．複数のキーを単一のプロパティに対応付けることも可能である．また，正規表現によってバリューの一部分だけを目的語として割り当てることも可能となっている．

RDF で用いる主語の URI は，マッピングルールのプロセッサが，元データに含まれているユニークな ID と対象ウェブサイト名の組み合わせで自動的に生成される．これにより，元データの ID が変わらない限りは URI の一意性を保証することができる．もし元データの ID が全部入れ替わるということが発生した場合は現在は対応できず，新規の URI を発行することになる．

```
{
  "meta": {
    "institutionalURI": "http://lod.ac/id/3193",
    "institutionalName": "秋田市立千秋美術館",
    "dc:source": "http://www.city.akita.akita.jp/
        city/ed/ss/senshu-art/",
    "dc:rights": "秋田市立千秋美術館"
  },
  "rules": {
    "dc:title": [
      {
        "name": "作品名",
        "lang": "ja"
      }
    ],
    "skos:prefLabel": [
      {
        "name": "作品名",
        "lang": "ja"
      }
    ],
    "rdfs:label": [
      {
        "name": "作品名",
        "lang": "ja"
      }
    ],
    "dc:created": [
      {
        "name": "制作年",
        "regexp": "(\\d+)\\("
      }
    ],
    "dc:medium": [
      {
        "name": "材質・技法",
        "lang": "ja"
      }
    ],
......
```

ソースコード 2: マッピングルール例

以上の手順で，Solr から出力された JSON ファイルから主語-述語-目的語の 3 つ組の関係を作成し，それを RDF として保存する．RDF の例をソースコード 3 に示す．

```
<http://lod.ac/ref/811510>
    rda2:dateOfBirth "1856" ;
    rda2:dateOfDeath "1910" ;
    crm:P100I_died_in "1910" ;
    crm:P32_used_general_technique "陶器"@ja ;
    crm:P45_consists_of "陶器"@ja ;
    crm:P98I_was_born "1856" ;
    dc11:creator "すずきりょくえん"@ja-hrkt, "鈴木緑園
        "@ja ;
    dc:dateAccepted "昭63年度寄贈"@ja ;
    dc:extent "高4.0長径11.0短径7.0" ;
    dc:isReferencedBy <http://lod.ac/id/811510> ;
    dc:medium "陶器"@ja ;
    dc:rights "秋田市立千秋美術館" ;
    dc:source <http://www.city.akita.akita.jp/city/
        ed/ss/senshu-art/> ;
    dc:title "秋田萬古緑釉蓮湯ざまし"@ja ;
    rdfs:label "秋田萬古緑釉蓮湯ざまし"@ja ;
    skos:prefLabel "秋田萬古緑釉蓮湯ざまし"@ja ;
    foaf:depiction <http://www.city.akita.akita.jp/
        city/ed/ss/senshu-art/data/jpeg/1465.jpg> .
```

ソースコード 3: RDF の例

3.4 データの履歴

各段階で取得・生成されているファイルは git レポジトリでバージョン管理を行なっている．具体的にはウェ

ブサイトからクロールしてきたHTMLファイル，Solrから出力されたJSONファイル，マッピングルールを通して生成されたRDFファイルがそれに該当する．これによって，変換前のページや変換後のRDFファイルの変遷が記録されるようになっている．突然ページのフォーマットが変更になってデータがうまく抽出できなくなったりすることがよくあるため，原因を追求するためにも過去の履歴を持っておくほうが都合が良い．

一方，RDFでデータの履歴を記述しておくという考え方もある．例えばある‘モノ’について1年前のデータと半年前のデータを取得して比較したいという要求がある．しかし，RDFでデータ自体の時系列を表現しようとすると煩雑になってしまう場合が多いため，現在は採用していない．将来的にはgitの履歴データをRDFとして出力することができれば良いかもしれない．

4 今後の課題

今後の課題は以下の通りである．

- 処理手順の見直し
- 同定作業
- マッピングルール用UI
- 統合データベースへの対応

現在の処理手順ではSolrに取り込んだキー・バリューの対をJSONの形で一旦出力してからルールを適用してRDFを生成している．これはルールを作成するユーザがデータをシステムから切り離して，まとめて見ながらルール作成を行えるという点で有効であった．しかし，このプロセスはSolrのAPIを使って直接Solrのデータを使うようにすれば，本来は必要がないはずである．そのため，この部分を省略してSolrから直接ルールを通してRDFを出力できるようにしたいと考えている．また，もう少し進めてRDFの出力と同時にRDF StoreにUPDATEを発行して差分更新するようにもしたいと考えている．

同定作業についてはLODAC Distillerの中に組み込まれておらず，RDF Storeに追加された後に独立した作業として行われている．しかし，マッピングルールを通してRDFを生成する際に著者の名寄せを行うなど，いくつかの同定作業はLODAC Distillerの中に組み込んで，半自動的に処理が行えるほうが望ましい．

マッピングルールは現在手作業で作成をしているが，専門家の方々に行なってもらうことを考えると不十分であり，マッピングルールを簡単に生成できるUIを用意すべきである．

現在のデータ抽出やマッピングルールなどは，各ウェブサイトがデータの管理者であるという前提で行なっている．しかし，SAGAデジタルミュージアムのように複数組織からデータを収集して統合データベースとして提供しているサイトにはまだ完全には処理できないため，対応したい．

5 おわりに

本稿では博物館の収蔵品データベースや家電メーカーの商品カタログサイトのような，キー・バリューの対で表現できるウェブサイトからRDFを生成して，SPARQL Endpointとして公開できるようにしているLODAC Distillerシステムについて紹介した．現在は博物館の収蔵品データに対してのみ適用しているが，他のコレクション型ウェブサイトにも適用することで，その有用性を確かめていく予定である．

謝辞

本システムの開発にご協力頂いた株式会社アットウェアに感謝する．

参考文献

[1] Berners-Lee, T.: Design Issues: Linked Data, `http://www.w3.org/DesignIssues/LinkedData.html`, (2006)

[2] Linked Data - Connect Distributed Data across the Web, `http://linkeddata.org/`

[3] Auer, S., Bizer, C., Lehmann, J., Kobilarov, G., Cyganiak, R., Ives, Z.: DBpedia: A Nucleus for a Web of Open Data, *In Proceedings of the 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference*, (2007)

[4] Kamura, T., Takeda, H., Ohmukai, I., Kato, F., Takahashi, T., and Ueda, H.: Study Support and Integration of Cultural Information Resources with Linked Data, *In Proceedings of the 2nd International Conference on Culture and Computing*, pp. 177–178 (2011)

[5] 加藤文彦: Linked Data作成支援ツールの現状と課題, 第*24*回セマンティックウェブとオントロジー研究会, 人工知能学会, (2011)

[6] OpenLink Software: Virtuoso Sponger, `http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/VirtSponger`

[7] Schultz, A., Matteini, A., Isele, R., Bizer C., Becker, C.: LDIF - Linked Data Integration Framework, *2nd International Workshop on Consuming Linked Data*, (2011)