

B-017

“meiseki” セマンティック Web サーバーの開発 A Development of “meiseki” Semantic Web Server

青木 祐香里[†] 塚本 享治[‡]
Yukari Aoki Michiharu Tsukamoto

1. はじめに

筆者らは3年次生を対象に1学期間の Jena[1]をベースにしたセマンティック Web の演習を行っている。その最後にはセマンティック Web 技術を集大成したサーバ構築を行う。問題は Jena が提供しているサーバフレームワーク joseki[3]が扱いにくいことである。本稿では、このために開発したフレームワーク meiseki について報告する。

2. joseki の問題点

joseki では扱うファイル群や RDB の指定を N3 形式の設定ファイルで記述しなければいけない。ところが、その記述文法が定義されていないし、アプリケーションの記述方法も明確でなく、初心者には大変変わりにくい。

3. 解決方法

3.1 記述文法の明確化

記述文法がない N3 形式に代わって XML 形式で設定ファイルを記述することにした。XML では簡単な DTD 定義によって記述形式が定義でき、事前に記述の妥当性がチェックできるからである。

3.1 Java による開発

サーバの開発には、Jena と pellet を用いることとした。ユーザのアプリケーション開発には、JSP/JSTL/Servlet を用いることにした。またユーザのプログラミングをより簡潔に書けるように JSTL と互換性のよいタグライブラリを準備した。

3.3 追加機能

セマンティック Web で最初に苦労するのは、種々の形式のデータの RDF 化である。そこで、事前変換のためにスクリーン・スクレイピングパッケージを準備した。さらに、設定ファイルに記述して実行時変換するために、GRDDL、RDFa、eRDF の機能をサーバに準備した。

4. meiseki サーバーの実現

4.1 システム構成

システムは、事前あるいは実行時に情報を収集・変換するアプリケーション、meiseki サーバ、Web アプリケーション記述のためのタグライブラリ、RDB に格納された三つ組 RDF の入れ替えを簡単に行う管理サブシステムで構成されている。

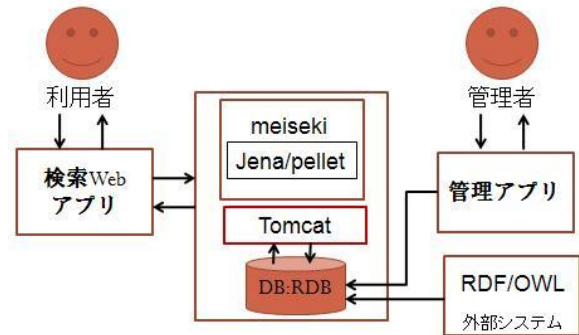


図1 meiseki システムの構成

4.2 情報収集変換アプリケーション

演習で最初に苦労するのは、ネットワークに公開されているデータの収集と RDF への変換である。このために、httpClient による情報収集、nekoHtml による XML 化を行うパッケージを準備し、ant で使えるようにした。収集する情報が複雑な Excel ファイルであったり、認証やプロキシへの対応ができる。

4.3 設定ファイルによる Assemble 機能

サーバは起動時に web.xml で指定された meiseki 設定ファイル meiseki.config (図2)を解析する。Meiseki.config はデータベース定義部と、モデル定義部からなる。

```
<definition>
<database>
  <param name="db.driver">org.h2.Driver</param>
  <param name="db.url">
    jdbc:h2:tcp://localhost:3031/...
  </param>
  <param name="db.user">sa</param>
  <param name="db.password"></param>
  <param name="db.tcpServer">
    -tcpAllowOthers -tcpPort 3031
  </param>
  <param name="db.type">HSQL</param>
</database>
<model name="kamoku" reasoning="owl">
  <data format="rdf/xml">
    &docBase: model/kamoku/kamoku.owl
  </data>
</model>
<model name="recipe" reasoning="rdf">
  <data format="rdf/xml">
    &docBase: model/recipe/data.rdf
  </data>
</model>
```

図2 config.xml の記述例

データベース定義にしたがってデータベースを起動し、モデル定義をメモリに読み込む。

モデル定義には、モデル名、推論レベル (RDF, RDFS, OWL)、読み込みソースとその形式

[†] 東京工科大学大学院 バイオ・情報メディア研究科
Tokyo University of Technology Graduate School

(RDF/XML, GRDDL, RDFa, eRDF)が書かれており、アプリケーションからの要求に応じてソースを読み込んで推論レベルに対応した RDF 展開して RDB に記憶する。すでに展開してある場合は RDF を読み込むだけである。

4.4 Sparql タグライブラリ

演習で実際に Web アプリを記述してみると、Sparql 文をアプリに対応して準備し検索結果を HTML に変換するのに大変多くの労力を費やす。そこで、これを簡単に行うための Sparql タグライブラリを参考資料[4]を書き直して meiseki サーバとのインタフェースを単純化した。

このタグライブラリでは無用に多くの Sparql 文を書かず、SQL 類似のパラメータ指定ができるようにした。パラメータの実現には、JSP タグライブラリ定義で使えるダイナミックプロパティを使った。タグライブラリを使う JSP/JSTL ページの記述例を図3に示す。

```
<%@ page contentType="text/html;charset=utf-8"%>
:
<%! //初期化 %>
<sparql:sparql>
<sparql:select var="results" modelName="recipe"
  zairyou1="${param.zairyou1}"
:
  PREFIX recipe: <http://media.teu.ac.jp/recipe#>
:
  SELECT DISTINCT ?recipename ?zairyou1 ?zairyou2
  WHERE {
    ?recipe recipe:名称 ?recipename;
      recipe:食材 ?z1;
      recipe:食材 ?z2;
      recipe:調理法 ?c.
    ?z1 rdf:type ?zz1.
    ?zz1 shokuzai:名前 ?zairyou1.
    ?z2 rdf:type ?zz2.
    ?zz2 shokuzai:名前 ?zairyou2.
    ?c rdf:type ?cc.
    ?cc chorihou:名前 ?chorihou.
    FILTER (?zairyou1 != ?zairyou2)
  }
</sparql:select>
<html><body>
<h2>セマンティック Web 技術を用いた料理レシピ検索結果</h2>
<table border="1">
<tr><th>No</th><th>レシピ名</th><th>食材 1</th><th>食材 2</th></tr>
<c:forEach var="result" items="${results.rows}" varStatus="st">
  <tr>
    <td>${st.index}</td>
    <td>${result.recipename}</td>
    <td>${result.zairyou1}</td>
    <td>${result.zairyou2}</td>
  </tr>
</c:forEach>
</table>
</body></html>
</sparql:sparql>
```

図3 Sparql タグライブラリによる JSP/JSTL 記述例

4.5 管理 Web アプリケーション

色々な Web アプリケーションを作成してみると、RDB に RDF 展開されたモデルを作り直したくなることが多い。そこで、簡単な管理 Web アプリケーション

を作成して RDB 上のモデル操作を可能にした。

5. 記述実験

5.1 セマンティック Web 作成実験

3 年次演習のほか研究室の色々な研究でセマンティック Web アプリケーションを作成してきた。この meiseki に搭載したものを次にあげる。

● 大学教務情報検索システム

当方の学部では、カリキュラム、シラバス、時間割、教員情報などの教務情報が Web ページ、Excel など別々の箇所で公開されてきた。演習ではセマンティック Web によるこれらの情報統合を題材としてきた。

● 料理レシピ検索システム

料理レシピは、RDFS を使ったカテゴリ検索の有効性を試すことができる。つまり、「肉野菜炒め」という質問によって、「牛肉レバーのもやし炒め」が検索結果として得られる。

● PC パーツ検索システム

このシステムは joseki のサンプル向けに作成されていたものを改良し、meiseki 用として直したものである。スキーマの互換性定義、部品互換性などを扱うサンプルである。

これらのアプリケーションを実際に稼働させることでセマンティック Web アプリケーション開発における問題点とその解決を meiseki サーバの改善に役立ててきた。

5.2 考察

meiseki は学生が演習で使用することを目的として作成したため、わかりやすさを第一に考えた。そのため、現在は、SOAP インタフェースは設けず、REST インタフェースにとどめている。

現在の問題点としては、①スピードが遅いこと、②RDB ではキーの長さが 200 で制限されていることである。あらかじめ良く設計された情報の RDF 化では問題ないが、見栄え重視の情報の場合、見出しに必要な情報を含めることが多い。このとき、RDB の制約に触れることがある。オープンソース RDB の一部書き直しを検討している。

6. おわりに

meiseki は JSP/JSTL だけでなく、スタンドアロンアプリからも使用することができる。しかし、すでにあげた問題点も見つかっており、今後も改善を進めていきたい。

参考資料

- [1]Jena: <http://jena.sourceforge.net/>
- [2]Pellet: <http://clarkparsia.com/pellet/>
- [3]Joseki: <http://www.joseki.org/>
- [4]Sparqltag: <http://djpowell.net/blog/entries/sparqltag.html>