MI-SWE
Semantic APIs

Jan Dědek

FIT-ČVUT

2011

Outline

Semantic APIs

According to Programming Language

http://www.w3.org/2001/sw/wiki/Category:Programming_Language

- Java
  - Jena
  - openRDF (Sesame)
  - OWL API
  - Java RDF Binding (JRDF)
- JavaScript
  - RDFParser
  - Jibbering
  - Hercules
  - SPARQL JavaScript Library
- .NET
  - SemWeb
  - dotNetRDF
  - OwlDotNetApi
  - ROWLEX
- C++
  - Soprano Qt/C++ (QRDF)
- PHP
  - ARC2
  - RDF API for PHP (RAP)
  - EasyRdf
- Perl
  - CARA
  - RDF Helper
- Pyton
  - 4Suite 4RDF
  - pyrple
- Ruby
  - RDF.rb: RDF for Ruby
  - ActiveRDF
- Flex
  - TopBraid Live Flex API
- ActionScript
  - SemanticFlash

- Java
  - Jena http://jena.sourceforge.net/
  - openRDF (Sesame) http://www.openrdf.org/
  - OWL API http://owlapi.sourceforge.net/
  - Java RDF Binding (JRDF) http://jrdf.sourceforge.net/
- JavaScript
  - RDFParser http://dig.csail.mit.edu/breadcrumbs/blog/14
  - Jibbering http://www.jibbering.com/rdf-parser/
  - Hercules http://hercules.arielworks.net/
  - SPARQL JavaScript Library http://www.thefigtrees.net/lee/blog/2006/04/sparql_calendar_demo_a_sparql.html
- .NET
  - SemWeb http://razor.occams.info/code/semweb/
  - dotNetRDF http://www.dotnetrdf.org/
  - OwlDotNetApi http://users.skynet.be/bpellens/OwlDotNetApi/
  - ROWLEX http://rowlex.nc3a.nato.int/
- C++
  - Soprano Qt/C++ (QRDF) http://soprano.sourceforge.net/

- PHP
  - ARC2 http://arc.semsol.org/home
  - RDF API for PHP (RAP) http://www4.wiwiss.fu-berlin.de/bizer/rdfapi/
  - EasyRdf http://www.aelius.com/njh/easyrdf/
- Perl
  - CARA http://cara.sourceforge.net/
  - RDF Helper http://search.cpan.org/dist/RDF-Helper/
- Pyton
  - 4Suite 4RDF
  - pyrple
- Ruby
  - RDF.rb: RDF for Ruby
  - ActiveRDF
- Flex
  - TopBraid Live Flex API
- ActionScript
  - SemanticFlash

SIMILE Widgets

- http://www.simile-widgets.org/
- Exhibit
  - Faceted browser, written in JavaScript
  - http://www.simile-widgets.org/exhibit/
- Babel
  - Converter between various formats (RDF -> JSON)
  - http://simile.mit.edu/wiki/Babel
  - http://service.simile-widgets.org/babel/

Meex
RSWA 2008 - Realizing a Semantic Web Application

- http://swa.cefriel.it/Teaching/RSWA2008
- Music Event Explorer
- Used technologies
  - Jena (GRDDL, SPARQL, OWL reasoner)
  - D2RQ + Joseki
  - Exhibit

Source Codes

Online Available

http://svn.berlios.de/svnroot/repos/nswi116/trunk/MI-SWE/slides/semAPI/src/

1. Introduction

2. Code Snippets
   - Jena
   - OpenRDF
   - SIMILE Widgets: Babel and Exhibit
   - OWL API
   - RAP

Resources/Hello.rdf

```
@prefix    rdf:      <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix    rdfs:     <http://www.w3.org/2000/01/rdf-schema#> .
@prefix    owl:     <http://www.w3.org/2002/07/owl#> .
@prefix    dc:       <http://purl.org/dc/elements/1.1/> .
@prefix    springer: <http://springer.com/> .
@prefix    xsd:      <http://www.w3.org/2001/XMLSchema#> .
@prefix    ex:       <http://www.example.org/> .
@prefix    exterms:  <http://www.example.org/terms/> .
@prefix    swe:      <http://www.fit.cvut.cz/subjects/mi-swe#> .

swe:Course   rdfs:label   "Semantic Web"@en ;
  swe:hasStudent swe:Rychlonozka ;
  swe:hasStudent swe:MirekDusin ;

  swe:numberOfLectures    "13"^^xsd:integer ;
  swe:firstLectureDate    "2011-09-19T12:45:00+01:00"^^xsd:dateTime .

swe:hasStudent      rdfs:subPropertyOf  swe:hasParticipant .
swe:hasParticipant  owl:inverseOf       swe:attends .
swe:hasStudent      rdfs:range          swe:Student .
swe:attends         rdfs:domain         swe:Person .
swe:Person          rdfs:subClassOf     swe:Human .

swe:Rychlonozka swe:hasStudyYear "1"^^xsd:integer .

swe:MirekDusin  swe:hasStudyYear "3"^^xsd:integer .
```

Jena/ReadWrite.java

```java
package mi_swe.jena;

import java.io.FileOutputStream;

import com.hp.hpl.jena.rdf.model.Model;
import com.hp.hpl.jena.rdf.model.ModelFactory;

public class ReadWrite
{
   public static Model readHelloRdfFile()   {
      // create an empty model
      Model model = ModelFactory.createDefaultModel();
      // parse file (or URL) into the model
      model.read("file:hello.rdf", "TURTLE");
      return model;
   }
   public static void main(String[] args) throws Exception {
       Model model = readHelloRdfFile();
       // write the whole model to the standard output
       model.write(System.out, "RDF/XML");
       // and to a file
       model.write(new FileOutputStream("hello.nt"), "N-TRIPLE");
   }
}
```

Jena/Reasoning.java

```java
package mi_swe.jena;

import com.hp.hpl.jena.rdf.model.Model;
import com.hp.hpl.jena.rdf.model.ModelFactory;
import com.hp.hpl.jena.reasoner.Reasoner;
import com.hp.hpl.jena.reasoner.ReasonerRegistry;

public class Reasoning
{
   public static Model makeInferedHelloModel() {
      Model helloModel = ReadWrite.readHelloRdfFile();
      //create a reasoner, see also:
      //getOWLMicroReasoner(), getOWLMiniReasoner(), getOWLReasoner()
      //getRDFSReasoner(), getRDFSSimpleReasoner(), getTransitiveReasoner()
      Reasoner reasoner = ReasonerRegistry.getOWLMicroReasoner();
      //attach the model to the reasoner
      Model inferedModel = ModelFactory.createInfModel(reasoner, helloModel);
      return inferedModel;
   }
   public static void main(String[] args)   {
      makeInferedHelloModel().write(System.out, "N-TRIPLE");
   }
}
```

Jena/Explain.java

```java
package mi_swe.jena;

import java.io.PrintWriter;
import java.util.Iterator;
import com.hp.hpl.jena.rdf.model.*;
import com.hp.hpl.jena.reasoner.*;

public class Explain {
   public static void main(String[] args) {
      Model helloModel = ReadWrite.readHelloRdfFile();
      Reasoner reasoner = ReasonerRegistry.getOWLMicroReasoner();
       // Turn on derivation logging - slows down the performance!!!
      reasoner.setDerivationLogging(true);
      // Attach the model to the reasoner
      InfModel infModel = ModelFactory.createInfModel(reasoner, helloModel);
      // Create data for a query - predicate: rdf:type, object: swe:Human
      Property rdfType = infModel.createProperty("http://www.w3.org/1999/02/22-rdf-syntax-ns#type");
      RDFNode sweHuman = infModel.createResource("http://www.fit.cvut.cz/subjects/mi-swe#Human");
      PrintWriter out = new PrintWriter(System.out); // Necessary of printTrace, see bellow
      // Select matching statements (subject arbitrary) and iterate
      for (Statement s : infModel.listStatements(null, rdfType, sweHuman).toList()) {
          System.out.println("Statement is " + s);
          for (Iterator<Derivation> id = infModel.getDerivation(s); id.hasNext(); ) {
              Derivation deriv = id.next();
              // Print a deep traceback of this derivation back to axioms and source assertions.
              deriv.printTrace(out, true); //print bindings = true
          }
          out.flush();
      }
   }
}
```

Jena/SPARQL/Construct.java

```java
package mi_swe.jena.sparql;

import mi_swe.jena.Reasoning;
import com.hp.hpl.jena.query.Query;
import com.hp.hpl.jena.query.QueryExecution;
import com.hp.hpl.jena.query.QueryExecutionFactory;
import com.hp.hpl.jena.query.QueryFactory;
import com.hp.hpl.jena.rdf.model.Model;

public class Construct {
   public static String queryString =
      "PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>" +
      "PREFIX swe: <http://www.fit.cvut.cz/subjects/mi-swe#>" +
      "CONSTRUCT { ?property a swe:ConstructedProperty}" +
      "WHERE {?property a rdf:Property}";
   public static QueryExecution prepareHelloQueryExecution(String sparqlQueryString)
   {
      Model inferedHelloModel = Reasoning.makeInferedHelloModel();
      //build a query object from string
      Query query = QueryFactory.create(sparqlQueryString);
      //attach the model to the query object
      QueryExecution qexec = QueryExecutionFactory.create(query, inferedHelloModel);
      return qexec;
   }
   public static void main(String[] args) {
      QueryExecution qexec = prepareHelloQueryExecution(queryString);
      //execute the construct query !!!
      Model constructedModel = qexec.execConstruct();
      constructedModel.write(System.out, "RDF/XML-ABBREV");
   }
}
```

Jena/SPARQL/Select.java

```java
package mi_swe.jena.sparql;

import com.hp.hpl.jena.query.QueryExecution;
import com.hp.hpl.jena.query.QuerySolution;
import com.hp.hpl.jena.query.ResultSet;
import com.hp.hpl.jena.rdf.model.Literal;
import com.hp.hpl.jena.rdf.model.Resource;

public class Select {
   public static String queryString =
      "PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>" +
      "SELECT ?resource ?label " +
      "WHERE {?resource rdfs:label ?label}";
   public static void main(String[] args) {
      QueryExecution qexec = Construct.prepareHelloQueryExecution(queryString);
      ResultSet resultSet = qexec.execSelect(); //select query !!!
      //iterate over the results
      while (resultSet.hasNext())  {
         QuerySolution querySolution = resultSet.next();
         //obtain a resource form the variable name
         Resource resource = querySolution.getResource("?resource");
         //obtain a literal form the variable name
         Literal label = querySolution.getLiteral("?label");
         //print resource URI
         System.out.println(resource.getURI());
         //obtain string value and language code of the literal and print
         System.out.println(label.getString());
         System.out.println(label.getLanguage());
      }
   }
}
```

Resources/Hello-rules.jena

```
@prefix rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl:  <http://www.w3.org/2002/07/owl#> .
@prefix swe:  <http://www.fit.cvut.cz/subjects/mi-swe#> .
#http://hydrogen.informatik.tu-cottbus.de/wiki/index.php/JenaRules
#http://jena.sourceforge.net/inference/#RULEbuiltins
[senior:
    (?s rdf:type swe:Student),
    (?s swe:hasStudyYear ?y),
    greaterThan(?y, 2)
    ->
    (?s rdf:type swe:Senior)
]
[first_lecture:
    (?course swe:firstLectureDate ?date),
    (?course swe:numberOfLectures ?number)
    ->
    (?course  swe:createLecures createLecures(?date, ?number, 1))
]
[createLecures:
    (?course  swe:createLecures createLecures(?date, ?numberTotal, ?numberCurrent)),
    ge(?numberTotal, ?numberCurrent), makeTemp(?lecture),
    addOne(?numberCurrent, ?numberNew),
    addDaysToDateTime(?dateNew, ?date, 7)
    ->
    (?course swe:hasLecture ?lecture),
    (?lecture swe:hasDate ?date),
    (?lecture swe:hasNumber ?numberCurrent),
    (?course  swe:createLecures createLecures(?dateNew, ?numberTotal, ?numberNew))
]
```

Jena/Rules.java

```java
package mi_swe.jena;

import mi_swe.jena.builtin.AddDaysToDateTime;

import com.hp.hpl.jena.rdf.model.*;
import com.hp.hpl.jena.reasoner.Reasoner;
import com.hp.hpl.jena.reasoner.rulesys.BuiltinRegistry;
import com.hp.hpl.jena.reasoner.rulesys.GenericRuleReasonerFactory;
import com.hp.hpl.jena.vocabulary.ReasonerVocabulary;
//http://jena.sourceforge.net/inference/
//http://hydrogen.informatik.tu-cottbus.de/wiki/index.php/JenaRules
public class Rules {
   public static void main(String[] args) {
      Model helloModel = Reasoning.makeInferedHelloModel();
      // register our builtin AddDaysToDate
      BuiltinRegistry.theRegistry.register(new AddDaysToDateTime());
      // create a resource (empty resource)
      Resource configuration = helloModel.createResource();
      // set engine mode
      configuration.addProperty(ReasonerVocabulary.PROPruleMode, "hybrid");
      // set the rules file
      configuration.addProperty(ReasonerVocabulary.PROPruleSet, "hello-rules.jena");
      // Create an instance of such a reasoner
      Reasoner reasoner = GenericRuleReasonerFactory.theInstance().create(configuration);
      // Attach the model to the reasoner
      InfModel inf = ModelFactory.createInfModel(reasoner, helloModel );
      // Write the output
      inf.write(System.out, "N3");
   }
}
```

Jena/Builtin/AddDaysToDateTime.java

```java
package mi_swe.jena.builtin;

import java.util.Calendar;

import com.hp.hpl.jena.datatypes.xsd.XSDDatatype;
import com.hp.hpl.jena.datatypes.xsd.XSDDateTime;
import com.hp.hpl.jena.graph.Node;
import com.hp.hpl.jena.reasoner.rulesys.RuleContext;
import com.hp.hpl.jena.reasoner.rulesys.builtins.BaseBuiltin;

public class AddDaysToDateTime extends BaseBuiltin {
   @Override // Name used in rules
   public String getName() { return "addDaysToDateTime"; }
   @Override // Body call implementation
   public boolean bodyCall(Node[] args, int length, RuleContext context) {
      // Second argument contains the old date
      XSDDateTime orig_date_val = (XSDDateTime) args[1].getLiteralValue();
      // Convert the value to Calendar instance
      Calendar calendar = orig_date_val.asCalendar();
      // Add number of days (third argument)
      calendar.add(Calendar.DAY_OF_MONTH, (Integer) args[2].getLiteralValue());
      // Create a XSDDateTime value from the calendar
      XSDDateTime new_date_val = new XSDDateTime(calendar);
      // Create new node from the value
      Node new_date_literal_node =                    //lang = null
         Node.createUncachedLiteral(new_date_val, null, XSDDatatype.XSDdateTime);
      // bind the variable of the first argument with the new date node
      context.getEnv().bind(args[0], new_date_literal_node);
      return true;
   }
}
```

1. Introduction

2. Code Snippets
   - Jena
   - OpenRDF
   - SIMILE Widgets: Babel and Exhibit
   - OWL API
   - RAP

OpenRDF/ReadRdfFile.java

```java
package mi_swe.openrdf;

import java.io.FileInputStream;

import org.openrdf.model.Statement;
import org.openrdf.rio.RDFHandler;
import org.openrdf.rio.RDFParser;
import org.openrdf.rio.helpers.StatementCollector;
import org.openrdf.rio.turtle.TurtleParser;

public class ReadRdfFile {
   public static void ParseHello(RDFHandler rdfHandler) throws Exception {
      //create instance of a parser
      RDFParser parser = new TurtleParser(); //NTriplesParser, RDFXMLParser
      //attach the handler to the parser
      parser.setRDFHandler(rdfHandler);
      //parse a file, second parameter is base URI
      parser.parse(new FileInputStream("hello.rdf"), "");
   }
   public static void main(String[] args) throws Exception {
      //create a statement collector
      StatementCollector collector = new StatementCollector();
      ParseHello(collector);
      //print all statements
      for (Statement statement : collector.getStatements()) {
         System.out.println(statement);
      }
   }
}
```

OpenRDF/WriteRdfFile.java

```java
package mi_swe.openrdf;

import org.openrdf.rio.RDFWriter;
import org.openrdf.rio.rdfxml.util.RDFXMLPrettyWriter;

public class WriteRdfFile {
   public static void main(String[] args) throws Exception {
      //create a RDF writer to standard output
      //Also possible: RDFXMLWriter, N3Writer, NTriplesWriter, TurtleWriter
      RDFWriter rdfWriter = new RDFXMLPrettyWriter(System.out);
      //write the output during parsing
      ReadRdfFile.ParseHello(rdfWriter);
   }
}
```

OpenRDF/InsertIntoRepository.java

```java
package mi_swe.openrdf;

import org.openrdf.model.Statement;
import org.openrdf.repository.RepositoryConnection;
import org.openrdf.repository.RepositoryResult;
import org.openrdf.repository.sail.SailRepository;
import org.openrdf.repository.util.RDFInserter;
import org.openrdf.sail.memory.MemoryStore;

public class InsertIntoRepository {
   public static SailRepository insertHelloIntoMemoryStore() throws Exception {
      // Create in-memory repository
      // Other possibilities: MySqlStore, NativeStore, PgSqlStore
      SailRepository repository = new SailRepository(new MemoryStore());
      repository.initialize();
      // Get connection object
      RepositoryConnection connection = repository.getConnection();
      // Create an inserter and attach the connection to it
      RDFInserter inserter = new RDFInserter(connection);
      // Parse a RDF file using the inserter
      ReadRdfFile.ParseHello(inserter);
      return repository;
   }
   public static void main(String[] args) throws Exception {
      RepositoryConnection connection = insertHelloIntoMemoryStore().getConnection();
      // Get all matching statements from the connection TODO: describe arguments
      RepositoryResult<Statement> result = connection.getStatements(null, null, null, true);
      // Print all statements
      for (Statement statement : result.asList()) {
         System.out.println(statement);
      }
   }
}
```

1. Introduction

2. Code Snippets
   - Jena
   - OpenRDF
   - SIMILE Widgets: Babel and Exhibit
   - OWL API
   - RAP

Babel/ConvertToExhibitJson.java

```java
package mi_swe.babel;

import java.io.FileOutputStream;
import java.io.OutputStreamWriter;
import java.util.Locale;
import java.util.Properties;
import mi_swe.openrdf.InsertIntoRepository;
import org.openrdf.repository.sail.SailRepository;
import org.openrdf.sail.Sail;
import edu.mit.simile.babel.exhibit.ExhibitJsonWriter;

public class ConvertToExhibitJson
{
   public static void main(String[] args) throws Exception {
      SailRepository repo = InsertIntoRepository.insertHelloIntoMemoryStore();
      // Create new writer
      ExhibitJsonWriter writer = new ExhibitJsonWriter();
      // Get Sail interface
      Sail sail = repo.getSail();
      // Prepare writer properties
      Properties properties = new Properties();
      properties.setProperty("namespace", "urn:babel:");
      properties.setProperty("url", "urn:babel:/");
      // Write Json data to a file
      writer.write(
            new OutputStreamWriter(new FileOutputStream("hello.js")),
            sail, properties, Locale.getDefault());
   }
}
```

Resources/hello.js

```
{
   "items" :      [
      {
         "hasStudyYear" : 1,
         "label" :        "Rychlonozka",
         "type" :         "Student",
         "uri" :          "http:\/\/www.fit.cvut.cz\/subjects\/mi-swe#Rychlonozka"
      },
      {
         "hasStudyYear" : 3,
         "label" :        "MirekDusin",
         "type" :         "Student",
         "uri" :          "http:\/\/www.fit.cvut.cz\/subjects\/mi-swe#MirekDusin"
      }
   ],
   "types" :      {
      "Student" : {
         "uri" : "http:\/\/www.fit.cvut.cz\/subjects\/mi-swe#Student"
      }
   },
   "properties" : {
      "hasStudyYear" : {
         "uri" :       "http:\/\/www.fit.cvut.cz\/subjects\/mi-swe#hasStudyYear",
         "valueType" : "number"
      }
   }
}
```

Exhibit/Hello.html

```
<html>
 <head>
    <link href="../resources/hello.js" type="application/json" rel="exhibit/data" />

    <script src="http://static.simile.mit.edu/exhibit/api-2.0/exhibit-api.js"
            type="text/javascript"></script>

    <script src="http://api.simile-widgets.org/exhibit/2.2.0/extensions/map/map-extension.js?gmapkey=ABQ
           type="text/javascript"></script>
 </head>
 <body>
    <table width="100%">
        <tr valign="top">
            <td ex:role="viewPanel">
                <div ex:role="view"></div>
                <div ex:role="view"
                   ex:viewClass="Map"
                   ex:latlng=".hasLatLng">
                </div>
            </td>
            <td width="25%">
                browsing controls here...
            </td>
        </tr>
    </table>
 </body>
 </html>
```

1. Introduction

2. Code Snippets
   - Jena
   - OpenRDF
   - SIMILE Widgets: Babel and Exhibit
   - OWL API
   - RAP

1. Introduction

2. Code Snippets
   - Jena
   - OpenRDF
   - SIMILE Widgets: Babel and Exhibit
   - OWL API
   - RAP

RAP/ReadWrite.php

```php
<?php
  define("RDFAPI_INCLUDE_DIR", "rdfapi-php/api/");
  include(RDFAPI_INCLUDE_DIR . "RdfAPI.php");

  // Create a new MemModel
  $model = ModelFactory::getDefaultModel();

  // Load and parse document, posibilities: rdf, n3, nt
  $model->load("../../resources/hello.nt", "nt");

  // Write to outpupt
  // Also possible: $model->writeAsHtml();
  $model->writeAsHtmlTable();

  // Save to file, posibilities: rdf, n3, nt
  $model->saveAs("../../resources/hello.n3", "n3");
?>
```

RAP/Inference.php

```php
<?php
  define("RDFAPI_INCLUDE_DIR", "rdfapi-php/api/");
  include(RDFAPI_INCLUDE_DIR . "RdfAPI.php");

  //create a InfmodelF
  $infModel= ModelFactory::getInfModelF('http://InfModelF.org');

  // Load and parse document, posibilities: rdf, n3, nt
  $infModel->load("../../resources/hello.nt", "nt");

  // Print and save results
  $infModel->writeAsHtmlTable();
  $infModel->saveAs("../../resources/hello_rap_inf.xml");
?>
```

RAP/FindStatements.php

```php
<?php
  define("RDFAPI_INCLUDE_DIR", "rdfapi-php/api/");
  include(RDFAPI_INCLUDE_DIR . "RdfAPI.php");
  $infModel= ModelFactory::getInfModelF('http://InfModelF.org');
  $infModel->load("../../resources/hello.nt", "nt");

  //create parameters for find (swe:Human and rdf:type)
  $swe_human = new Resource("http://www.fit.cvut.cz/subjects/mi-swe#Human");
  $rdf_type = new Resource("http://www.w3.org/1999/02/22-rdf-syntax-ns#type");

  //find all stements subject: arbitrary, predicate: rdf:type object: swe:Human
  $result_model = $infModel->find(NULL, $rdf_type, $swe_human);

  //iterate the result statements
  $it = $result_model->getStatementIterator();
  while ($it->hasNext()) {
        $statement = $it->next();
        //print subject URI
        echo $statement->getLabelSubject() ."\n";
  }
?>
```

Number two

```
1   %[Rule 1] [Pos cover = 14 Neg cover = 0]
2   damage_root(A) :- lex_rf(B,A), has_sempos(B,'n.quant.def'), tDependency(C,B),
3      tDependency(C,D), has_t_lemma(D,'investigator').
4
5   %[Rule 2] [Pos cover = 13 Neg cover = 0]
6   damage_root(A) :- lex_rf(B,A), has_functor(B,'TOWH'), tDependency(C,B),
7      tDependency(C,D), has_t_lemma(D,'damage').
8
9
10  %[Rule 1] [Pos cover = 7 Neg cover = 0]
11  injuries(A) :- lex_rf(B,A), has_functor(B,'PAT'), has_gender(B,anim),
12     tDependency(B,C), has_t_lemma(C,'injured').
13
14  %[Rule 8] [Pos cover = 6 Neg cover = 0]
15  injuries(A) :- lex_rf(B,A), has_gender(B,anim), tDependency(C,B),
16     has_t_lemma(C,'injure'), has_negation(C,neg0).
```