Sistemas Basados en Conocimiento

Mauricio Espinoza Mejía

Contenido

- Resumen: herramientas
- Sistemas basados en conocimiento
- Cómo crear una ontología programáticamente
- Apache Jena
- OWL API

Resumen: herramientas

Para gestionar un sistema basado en conocimiento se requieren herramientas

- **Editores** (http://semanticweb.org/wiki/Editors)
  - Editor mas común: **Protégé 5**
  - Otras herramientas: TopBraid Composer ($),
  - NeOn toolkit –para propósitos especiales, especialmente para ontologías livianas (por ejemplo, editores FOAF)

- **Razonadores** (http://semanticweb.org/wiki/Reasoners)
  - OWL DL: **Pellet 2.0***, **HermiT**, FaCT++, RacerPro ($)
  - OWL EL: CEL, SHER, snorocket ($), ELLY
  - OWL RL: OWLIM, Jena, Oracle OWL Reasoner ($)
  - OWL QL: Owlgres, QuOnto, Quill

Sistemas basados en conocimiento

- **Definición:** Sistemas que utilizan conocimientos explícitamente representados para resolver problemas complejos en lugar de depender únicamente de datos o algoritmos.

- **Componentes clave**:
  - Base de conocimiento (ontologías, reglas, hechos).
  - Motor de inferencia.
  - Interfaz de usuario.

- **Aplicaciones comunes**: Diagnóstico médico, planificación, comercio electrónico, sistemas de tutoría inteligente

Sistema basado en conocimiento: Ontologías como Pilar Fundamental

- **¿Qué es una ontología?**
  - Definición: Representación formal de un conjunto de conceptos dentro de un dominio y las relaciones entre ellos.
  - Ejemplo: En medicina, conceptos como "Paciente", "Enfermedad", "Tratamiento" y sus interrelaciones.

- **Rol de las ontologías en los sistemas basados en conocimiento**
  - Estandarización del vocabulario.
  - Interoperabilidad entre sistemas.
  - Facilitación del razonamiento lógico y la inferencia

Ejemplos Prácticos de Ontologías en Sistemas Basados en Conocimiento

- **Medicina**: Ontologías como SNOMED CT para estandarizar términos médicos.
- **Energía**: Ontologías para modelar redes inteligentes y optimizar el consumo energético.
- **Educación**: Ontologías para crear sistemas de tutoría personalizados que entienden las competencias del estudiante.
- **E-commerce**: Ontologías para personalizar recomendaciones de productos basadas en el comportamiento del usuario.

Desarrollo de Ontologías

- **Herramientas populares**:
  - Protégé: Creación, edición y gestión de ontologías.
- **Lenguajes de representación**:
  - OWL (Web Ontology Language): Ideal para sistemas semánticos y razonamiento.
  - RDF (Resource Description Framework): Para representar datos vinculados.

Cómo crear una ontología programáticamente

- Eb el curso ha creado una ontología OWL…
- … ha consultado algunos SPARQL Endpoint…
- ¿Cómo hacer esto mediante programación?
  - p. ej., desde una aplicación de software

- **Buenas noticias**: ¡existen los frameworks!
  - Los más populares están escritos en Java…
  - Apache Jena (RDF/OWL)
  - OWL API (OWL2)

Investigación

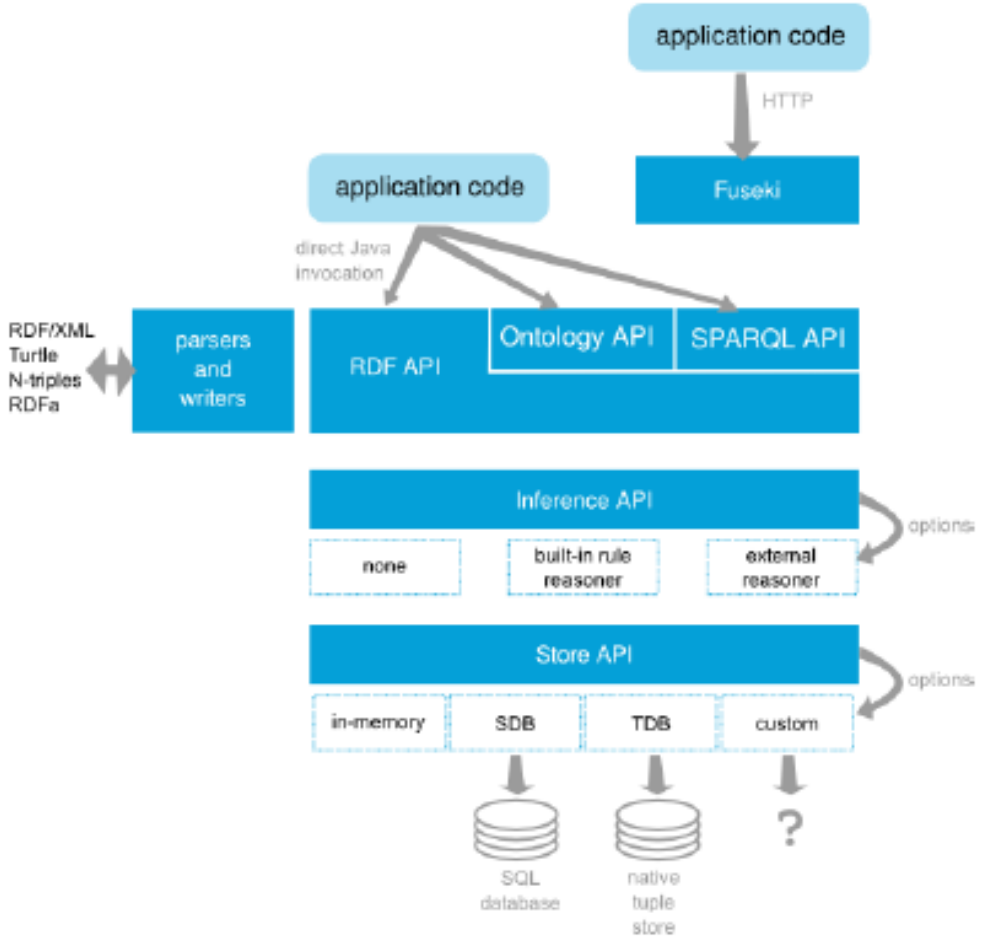
- Qué otros frameworks existen para gestionar una ontología?

- En Python?
- En C?

Apache Jena

- Marco Java gratuito y de código abierto para crear aplicaciones basadas en conocimiento y de datos vinculados
  - https://jena.apache.org

- Está compuesto por varias API, así como herramientas de línea de comandos

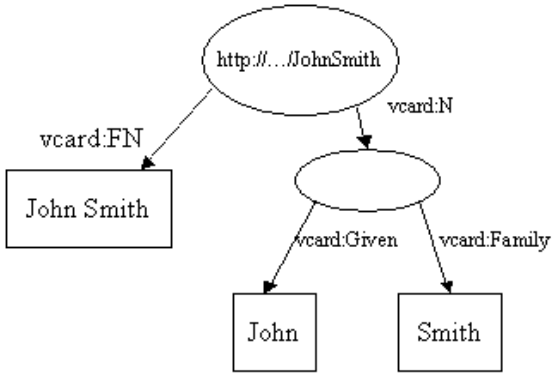


Apache Jena

- Tutoriales
  - https://jena.apache.org/tutorials/index.html
- Código de muestra:
  - https://github.com/apache/jena/tree/master/jena-core/src-examples/jena/examples/rdf

- Especialmente para modelos en RDF

Creando RDF




```
String personURI= "http://somewhere/JohnSmith";
String givenName= "John";
String familyName= "Smith";
String fullName= givenName+ " " + familyName;
// create an empty model
Model model= ModelFactory.createDefaultModel();
// create the resource and add the properties cascading style
Resource johnSmith= model.createResource(personURI)
        .addProperty(VCARD.FN, fullName)
        .addProperty(VCARD.N, model.createResource()
        .addProperty(VCARD.Given, givenName)
        .addProperty(VCARD.Family, familyName));
```

Escribir RDF…

- Escribir el modelo anterior en un OutputStream

```
// now write the model in XML form to a file
model.write(System.out);
```

- También puede especificar el formato

```
// you can also specify the format, e.g.,
model.write(System.out,"TURTLE");
```

Leyendo RDF…

- Leer desde un flujo de entrada

```
String inputFileName= "vc-db-1.rdf";
InputStreamin = FileManager.get().open(inputFileName);
// read the RDF/XML file
model.read(in, "");
```

The base URI to be used when converting relative URI's to absolute URI's

SPARQL

- Jena admite consultas SPARQL a través del motor ARQ
  - SPARQL estándar
  - Búsqueda de texto libre a través de Lucene
  - Acceso y extensión del álgebra SPARQL
  - Funciones de propiedad para procesamiento personalizado de relaciones semánticas
  - Agregación, GROUP BY y asignación como extensiones SPARQL
  - Compatibilidad para acceso remoto a cualquier punto final SPARQL
  - …

SPARQL con ARQ: Ejemplo

```
[...]
// Create a new query
String queryString=
"PREFIX foaf: <http://xmlns.com/foaf/0.1/> " +
"SELECT ?url" +
"WHERE {" +
"      ?contributor foaf:name\"Luigi De Russis\" . " +
"      ?contributor foaf:weblog?url. " +
"      }";
```

SPARQL con ARQ: Ejemplo

```
Query query= QueryFactory.create(queryString);
// Execute the query and obtain results
QueryExecution qe= QueryExecutionFactory.create(query, model);
ResultSetresults = qe.execSelect();
// Output query results
ResultSetFormatter.out(System.out, results, query);
// Free up resources used running the query
qe.close();
```

OWL API

- Una API de Java y una implementación de referencia
  - para crear, manipular y serializar ontologías de OWL 2
  - http://owlcs.github.io/owlapi

- Libre y de código abierto
  - Creada y mantenida por la Universidad de Manchester
  - http://owl.cs.manchester.ac.uk

OWL API

Incluye los siguientes componentes

- API para OWL 2 y una implementación eficiente de referencia en memoria
- Analizador y escritor RDF/XML
- Analizador y escritor OWL/XML
- Analizador y escritor OWL Functional Syntax
- Analizador y escritor Turtle
- SWRL
- Interfaces de razonadores
  - por ejemplo, FaCT++, HermiT, Pellet y Racer

OWL API

- Documentación y Javadocs
  - https://github.com/owlcs/owlapi
- Actualmente utilizado por Protégé

Fundamentos de la API de OWL

OWLOntology

- una interfaz
- que modela un conjunto de axiomas OWL lógicos y no lógicos, con un nombre (un IRI) y métodos convenientes para recuperar dichos axiomas

OWLEntity

- cualquier cosa que pueda identificarse con un IRI, es decir, nombres de clase, datos y propiedades de objeto, individuos nombrados, …

Fundamentos de la API de OWL

OWLAxiom

- la unidad básica
- Los axiomas de TBox describen relaciones entre clases y expresiones de clase (equivalencia, subsunción, disyunción)
- Los axiomas de ABox (aserciones) describen relaciones entre individuos y entre individuos y clases/expresiones de clase
- Los axiomas de RBox describen relaciones entre propiedades

Cargar (o crear) una ontología…

- Creación de una ontología

```
OWLOntologyManagerm = OWLManager.createOWLOntologyManager();
OWLOntologyo = m.createOntology(example_iri);
```

- Carga de una ontologia

```
OWLOntologyManagerm = OWLManager.createOWLOntologyManager();
OWLOntologyo = m.loadOntologyFromOntologyDocument(ont_iri);
```


Un archivo o una IRI

Guardar una ontología…

- Guardar en formato OWL/XML

```
m.saveOntology(ontology, new
                                    OWLXMLOntologyFormat (), file);
```

- Guardar en un formato RDF/XML

```
m.saveOntology(ontology, file);
```