



Java is a trademark of Sun Microsystems, Inc.



JavaOneSM

Developing Smart JavaTM Code with Semantic Web Technology

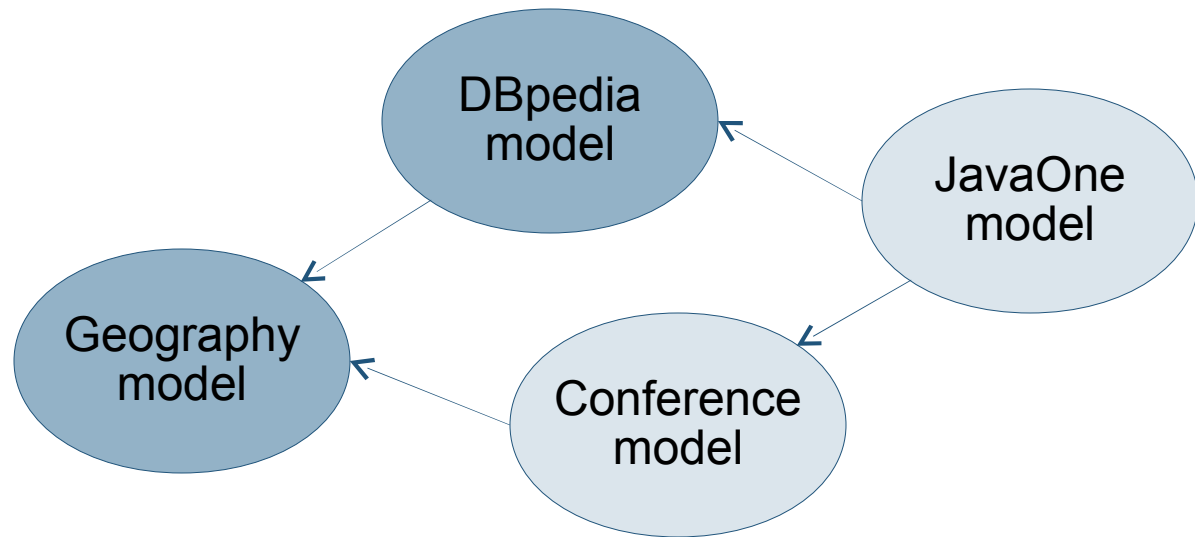
Holger Knublauch

TopQuadrant, Inc.

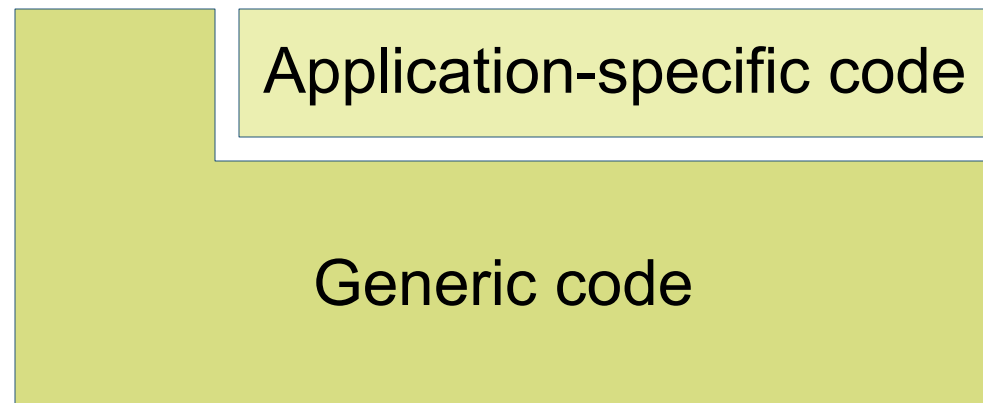
holger@topquadrant.com

Semantic Web Elevator Pitch

Linked
Smart
Data



Smart
Code



Overview of this talk

- > Introduction to RDF
 - Resources, literals and triples
- > Linked Data
- > RDF Schema
 - Classes, properties, and instances
- > Java APIs and Tools
- > Data Binding and software architecture
- > SPARQL
 - Queries and rules
- > Model-driven applications

Semantic Web Technology

- > Infrastructure to link data on the Web
- > Global database
- > Borrows some ideas of object-oriented languages
- > Data is self-describing
- > W3C standards
 - RDF
 - RDF Schema and OWL
 - SPARQL
 - RDFa, GRDDL

RDF: Resources

- > Semantic Web entities have a unique identifier

<http://examples.topquadrant.com/conferences/javaone#ScottMcNealy>

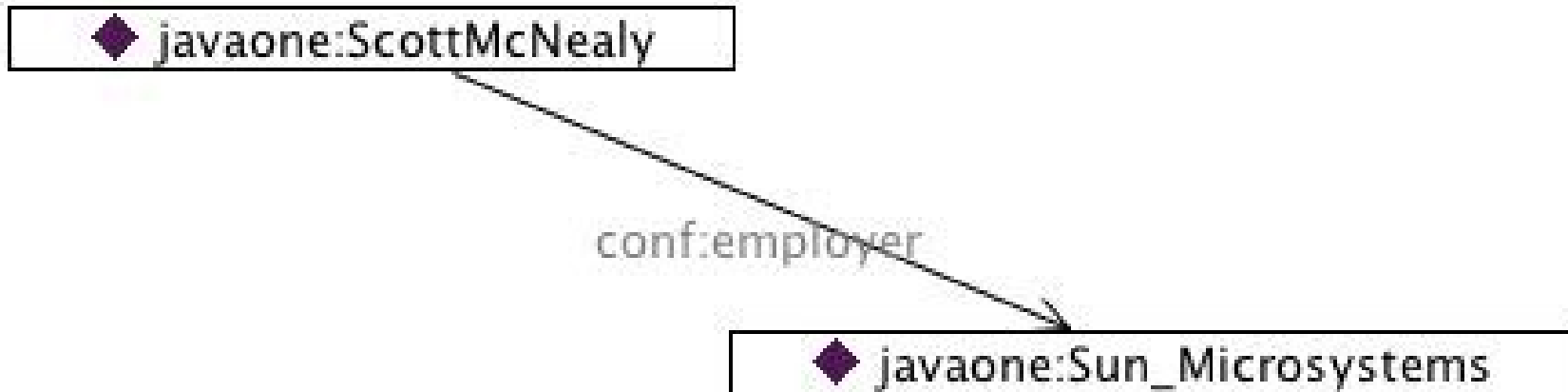
Namespace

Local name

abbreviated as

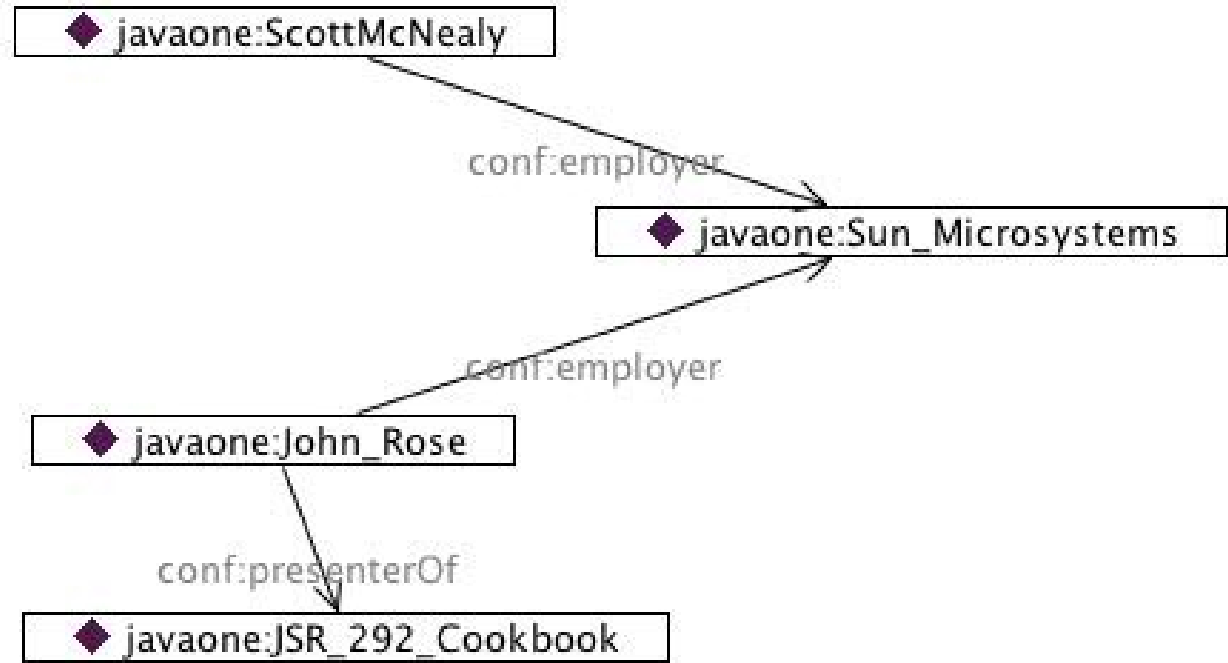
[javaone:ScottMcNealy](#)

RDF: Triples







Subject	Predicate	Object
<code>javaone:ScottMcNealy</code>	<code>conf:employer</code>	<code>javaone:Sun_Microsystems</code>

RDF: Graphs



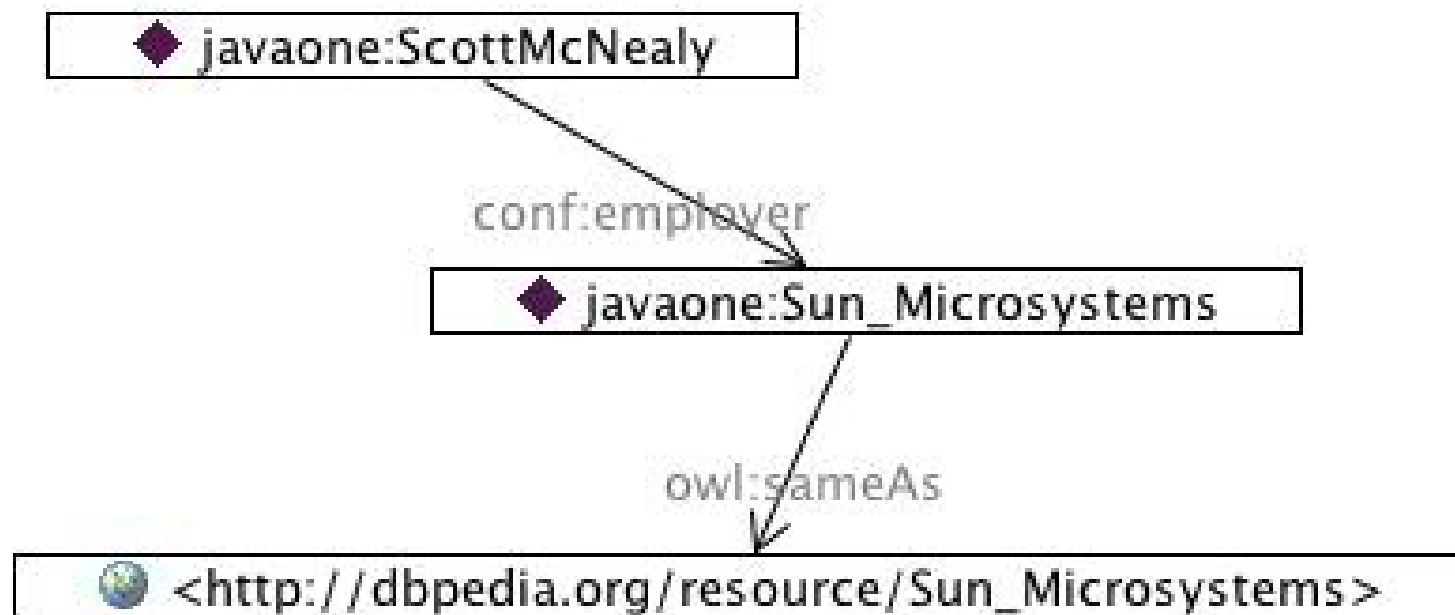
Subject	Predicate	Object
javaone:ScottMcNealy	conf:employer	javaone:Sun_Microsystems
javaone:John_Rose	conf:employer	javaone:Sun_Microsystems
javaone:John_Rose	conf:presenterOf	javaone:JSR_292_Cookbook

RDF: Datatype property values

 javaone:ScottMcNealy
 conf:firstName = Scott
 conf:lastName = McNealy
 rdfs:label = Scott McNealy

Subject	Predicate	Object
javaone:ScottMcNealy	conf:firstName	"Scott"^^xsd:string
javaone:ScottMcNealy	conf:lastName	"McNealy"^^xsd:string
javaone:ScottMcNealy	rdfs:label	"Scott McNealy"

Linked Data



Linked Data: Wikipedia

Sun Microsystems - Wikipedia, the free encyclopedia - Windows Internet Explorer

W http://en.wikipedia.org/wiki/Sun_microsystems


W Sun Microsystems - Wikipedia, the free encyclopedia

Log in / create account

article discussion edit this page history

Sun Microsystems

From Wikipedia, the free encyclopedia
(Redirected from Sun microsystems)

 This article documents a **current event**. Information may change rapidly as the event progresses.

Sun Microsystems, Inc. (NASDAQ: [JAVA](#))^[3] is a multinational vendor of computers, computer components, computer software, and information technology services, founded on February 24, 1982.^[4] The company is headquartered in Santa Clara, California (part of Silicon Valley), on the former west campus of the Agnews Developmental Center.

Products include computer servers and workstations based on its own SPARC processors as well as AMD's Opteron and Intel's Xeon processors; storage systems; and, a suite of software products including the Solaris Operating System, developer tools, Web infrastructure software, and identity management applications. Other technologies of note include the Java platform, MySQL and NFS.

Sun is a proponent of open systems in general and Unix in particular, and a major contributor to open source software.^[5]


On April 20, 2009, Sun and Oracle Corporation announced that they entered into a definitive agreement under which Oracle will acquire Sun for \$7.4 billion.^{[6][7]}

Sun's manufacturing facilities are located in Hillsboro, Oregon, USA and Linlithgow, Scotland.

Contents [\[hide\]](#)

- 1 History
 - 1.1 The "Bubble" and its aftermath
 - 1.2 Present focus

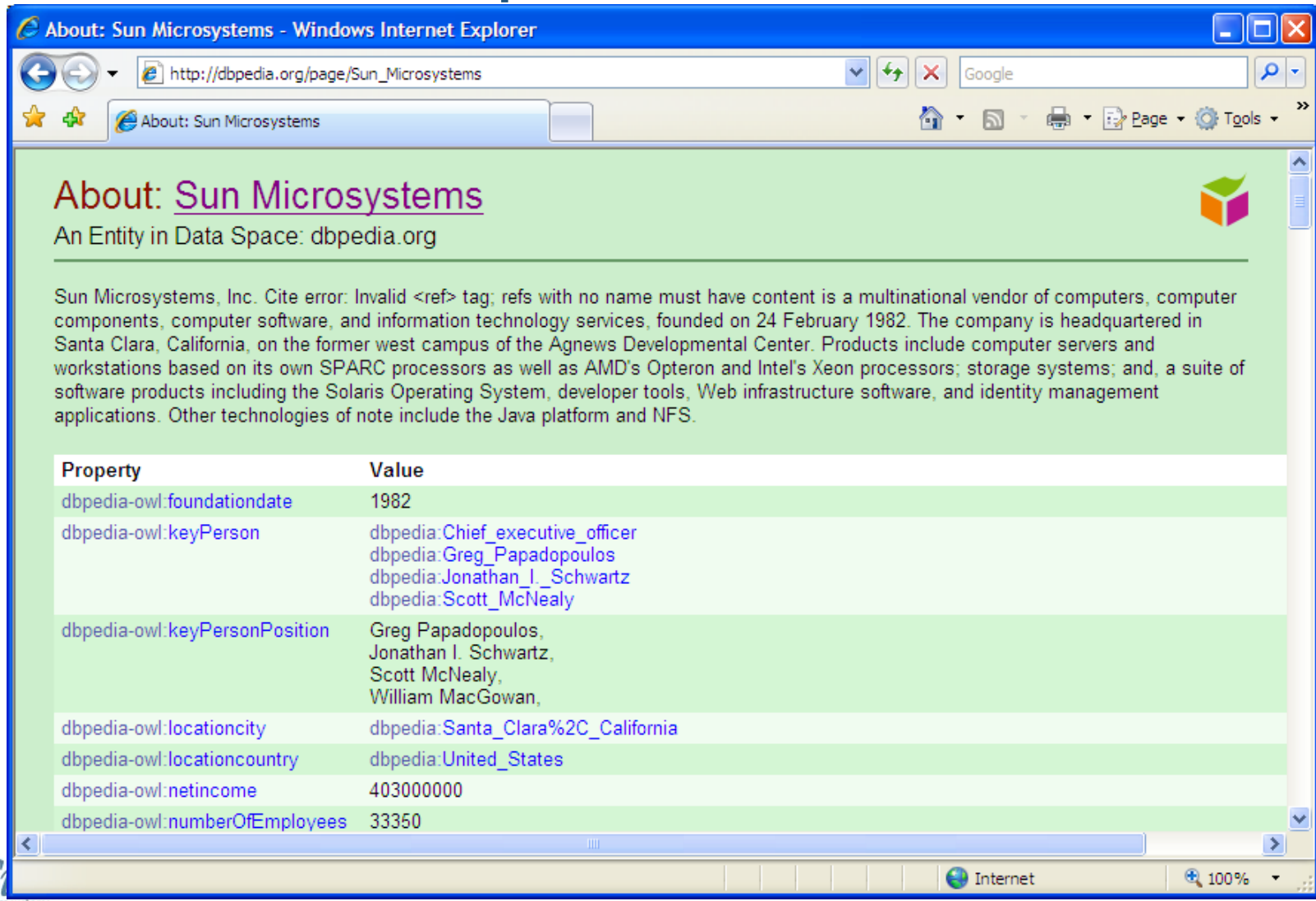
Sun Microsystems



Type	Public (NASDAQ: JAVA)
Founded	1982
Headquarters	Santa Clara, California, United States
Key people	Scott McNealy , Chairman Jonathan I. Schwartz , President and CEO William MacGowan , Executive Vice President, People and Places, and CHRO Greg Papadopoulos , Executive

Done

Linked Data: DBpedia Browser

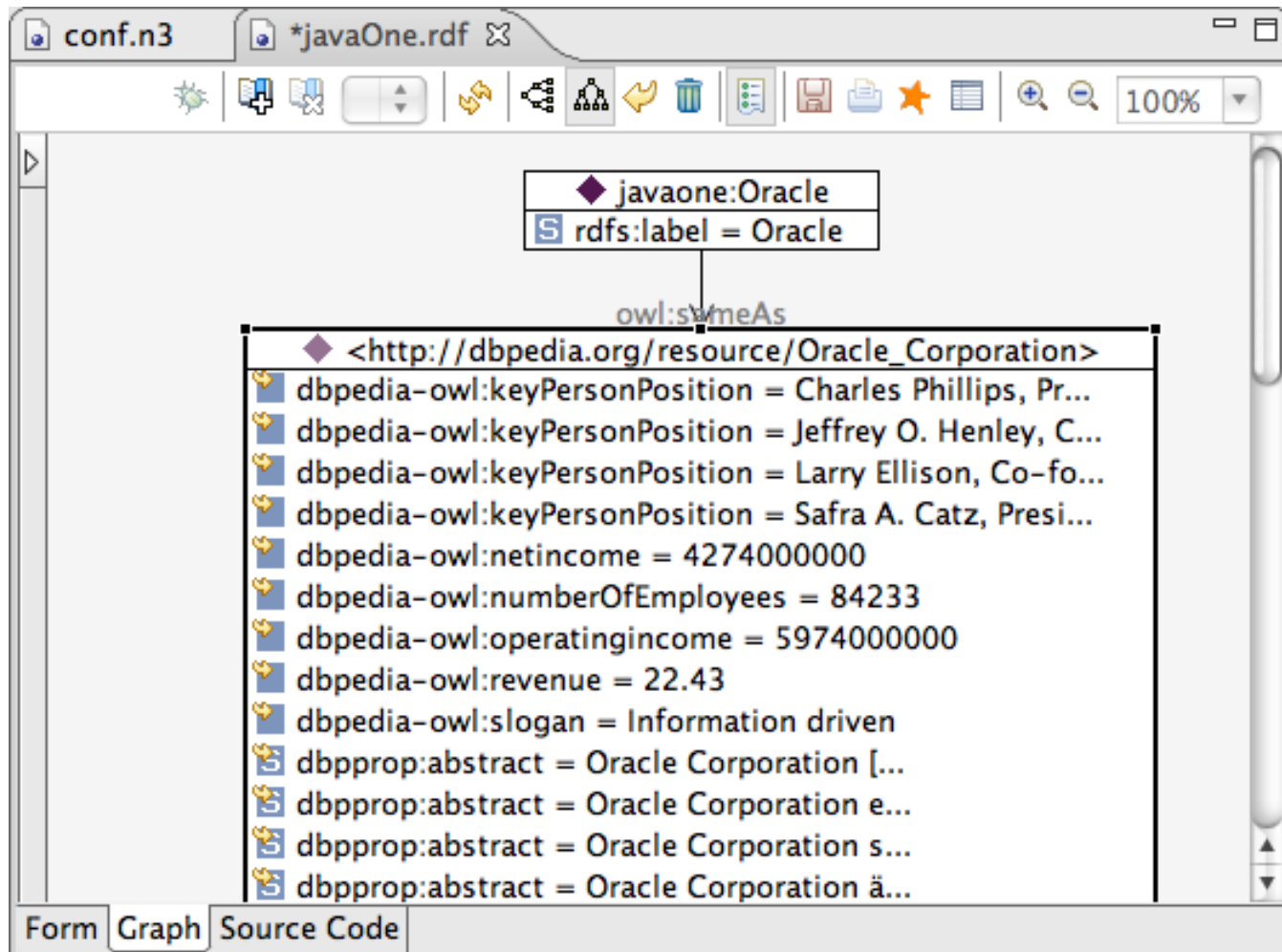


About: Sun Microsystems
An Entity in Data Space: dbpedia.org

Sun Microsystems, Inc. Cite error: Invalid <ref> tag; refs with no name must have content is a multinational vendor of computers, computer components, computer software, and information technology services, founded on 24 February 1982. The company is headquartered in Santa Clara, California, on the former west campus of the Agnews Developmental Center. Products include computer servers and workstations based on its own SPARC processors as well as AMD's Opteron and Intel's Xeon processors; storage systems; and, a suite of software products including the Solaris Operating System, developer tools, Web infrastructure software, and identity management applications. Other technologies of note include the Java platform and NFS.

Property	Value
dbpedia-owl:foundationdate	1982
dbpedia-owl:keyPerson	dbpedia:Chief_executive_officer dbpedia:Greg_Papadopoulos dbpedia:Jonathan_I._Schwartz dbpedia:Scott_McNealy
dbpedia-owl:keyPersonPosition	Greg Papadopoulos, Jonathan I. Schwartz, Scott McNealy, William MacGowan,
dbpedia-owl:locationcity	dbpedia:Santa_Clara%2C_California
dbpedia-owl:locationcountry	dbpedia:United_States
dbpedia-owl:netincome	403000000
dbpedia-owl:numberOfEmployees	33350

Linked Data: more RDF triples, for free!

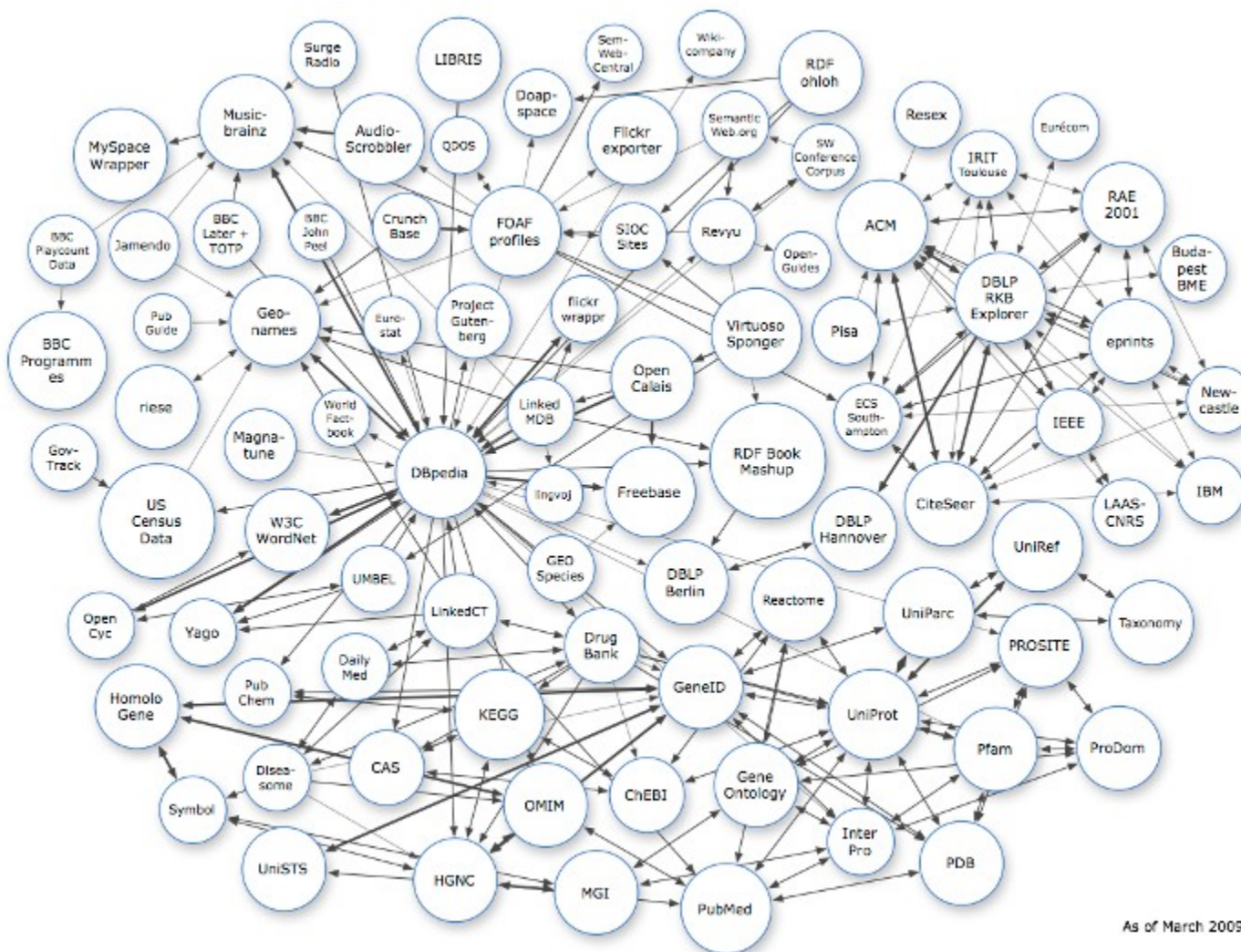


The screenshot shows a web browser window with two tabs: 'conf.n3' and '*javaOne.rdf'. The browser displays an RDF graph. At the top, a class box for 'javaone:Oracle' is shown with the property 'rdfs:label = Oracle'. Below this, a vertical line labeled 'owl:sameAs' connects to a resource box for '<http://dbpedia.org/resource/Oracle_Corporation>'. This resource box contains a list of properties and their values, each preceded by a small icon (either a blue square with a white 'S' or a yellow star). The properties include:

- dbpedia-owl:keyPersonPosition = Charles Phillips, Pr...
- dbpedia-owl:keyPersonPosition = Jeffrey O. Henley, C...
- dbpedia-owl:keyPersonPosition = Larry Ellison, Co-fo...
- dbpedia-owl:keyPersonPosition = Safra A. Catz, Presi...
- dbpedia-owl:netincome = 4274000000
- dbpedia-owl:numberOfEmployees = 84233
- dbpedia-owl:operatingincome = 5974000000
- dbpedia-owl:revenue = 22.43
- dbpedia-owl:slogan = Information driven
- dbpprop:abstract = Oracle Corporation [...
- dbpprop:abstract = Oracle Corporation e...
- dbpprop:abstract = Oracle Corporation s...
- dbpprop:abstract = Oracle Corporation ä...

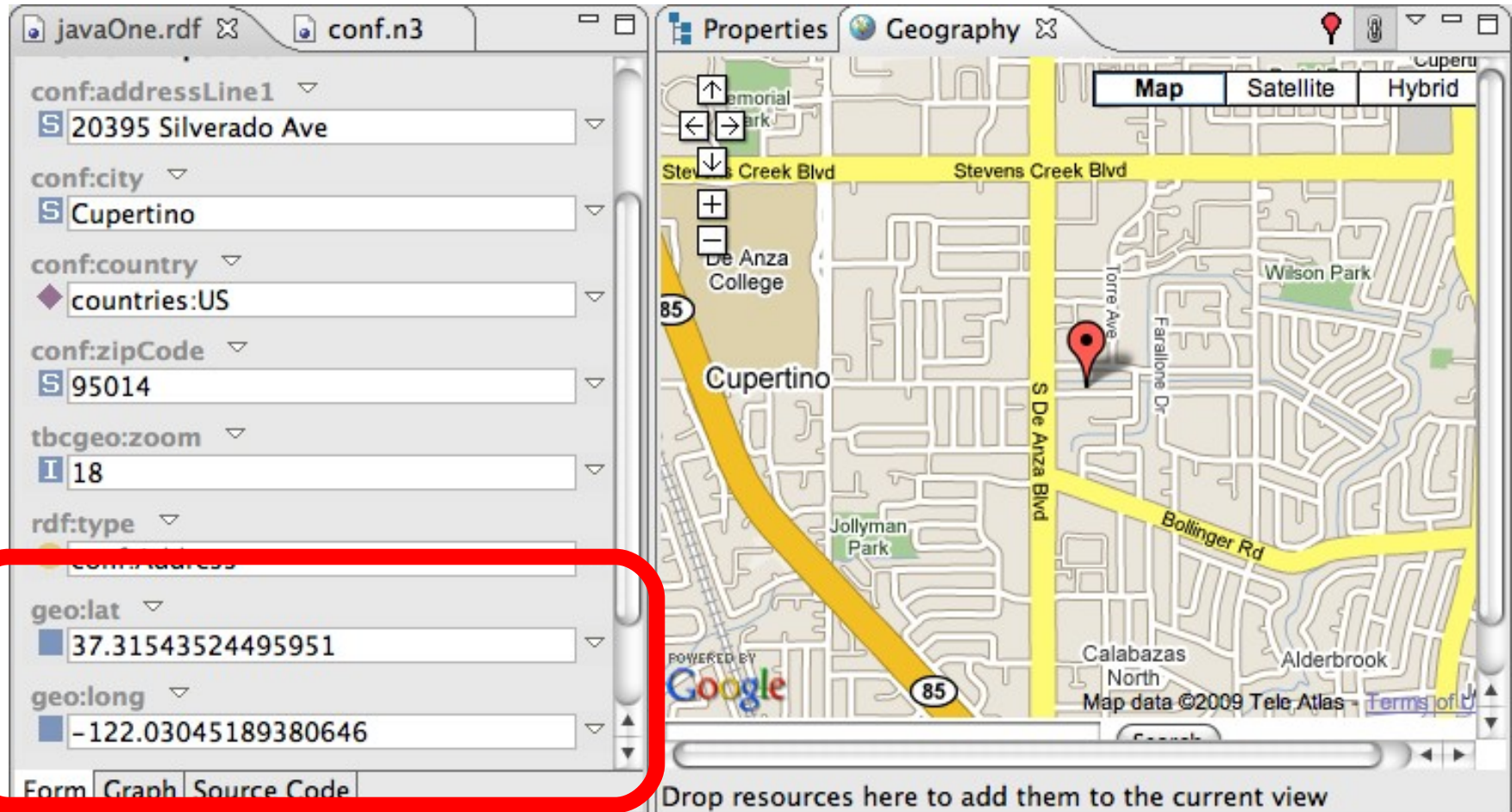
At the bottom of the browser window, there are three tabs: 'Form', 'Graph', and 'Source Code', with 'Graph' currently selected.

Linked Data: join the cloud



As of March 2005

Linked Data: Geography ontology



The screenshot displays a web application interface for managing address data. On the left, a form titled 'javaOne.rdf' and 'conf.n3' contains several fields. The 'geo:lat' and 'geo:long' fields are highlighted with a red rectangle. On the right, a map titled 'Geography' shows a street view of Cupertino, CA, with a red pin indicating the location. The map includes labels for 'Stevens Creek Blvd', 'S De Anza Blvd', 'Cupertino', 'Wilson Park', 'Jollyman Park', 'Calabazas North', and 'Alderbrook'. The map is powered by Google and includes a search bar.

Form fields:

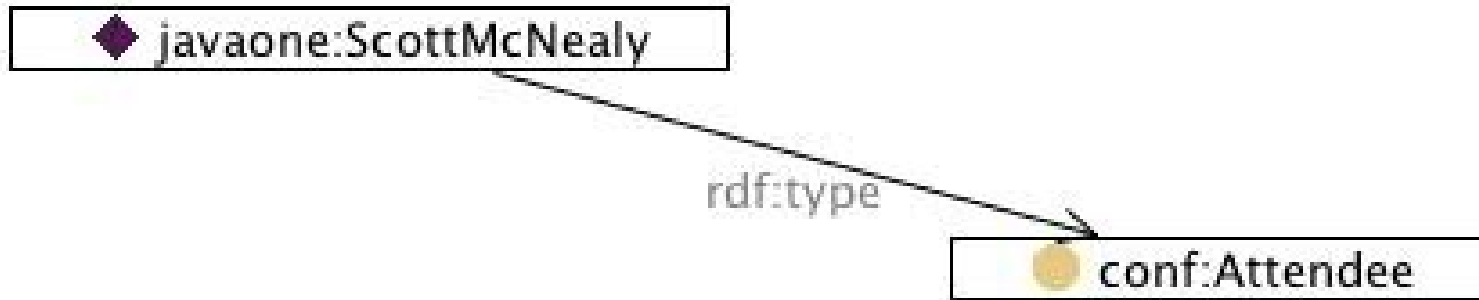
- conf:addressLine1: 20395 Silverado Ave
- conf:city: Cupertino
- conf:country: countries:US
- conf:zipCode: 95014
- tbcgeo:zoom: 18
- rdf:type: conf:address
- geo:lat: 37.31543524495951
- geo:long: -122.03045189380646

Map features:

- Map, Satellite, Hybrid tabs
- Stevens Creek Blvd
- S De Anza Blvd
- Cupertino
- Wilson Park
- Jollyman Park
- Calabazas North
- Alderbrook
- Map data ©2009 Tele Atlas
- Search bar

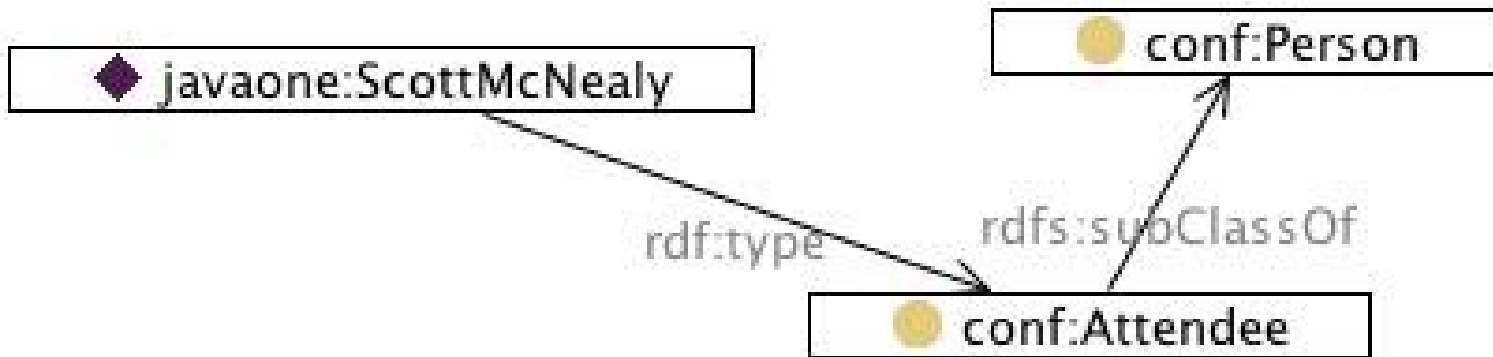
Drop resources here to add them to the current view

RDF Schema: Type relationships



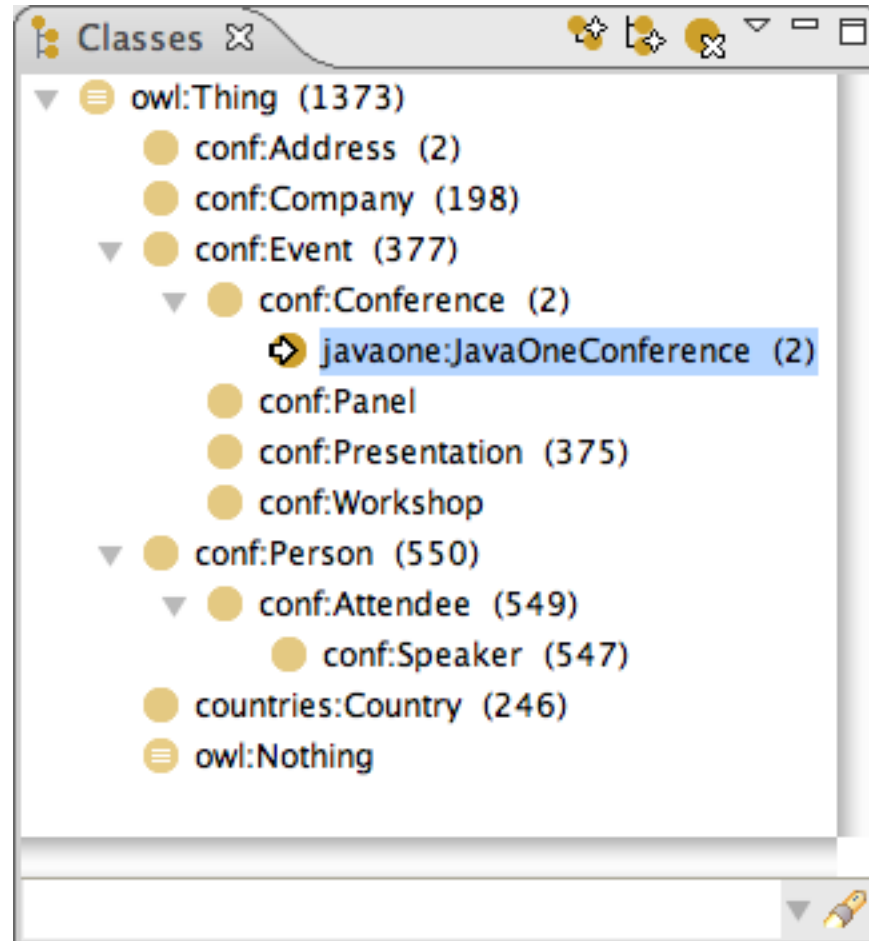
Subject	Predicate	Object
<code>javaone:ScottMcNealy</code>	<code>rdf:type</code>	<code>conf:Attendee</code>

RDF Schema: Subclass relationships

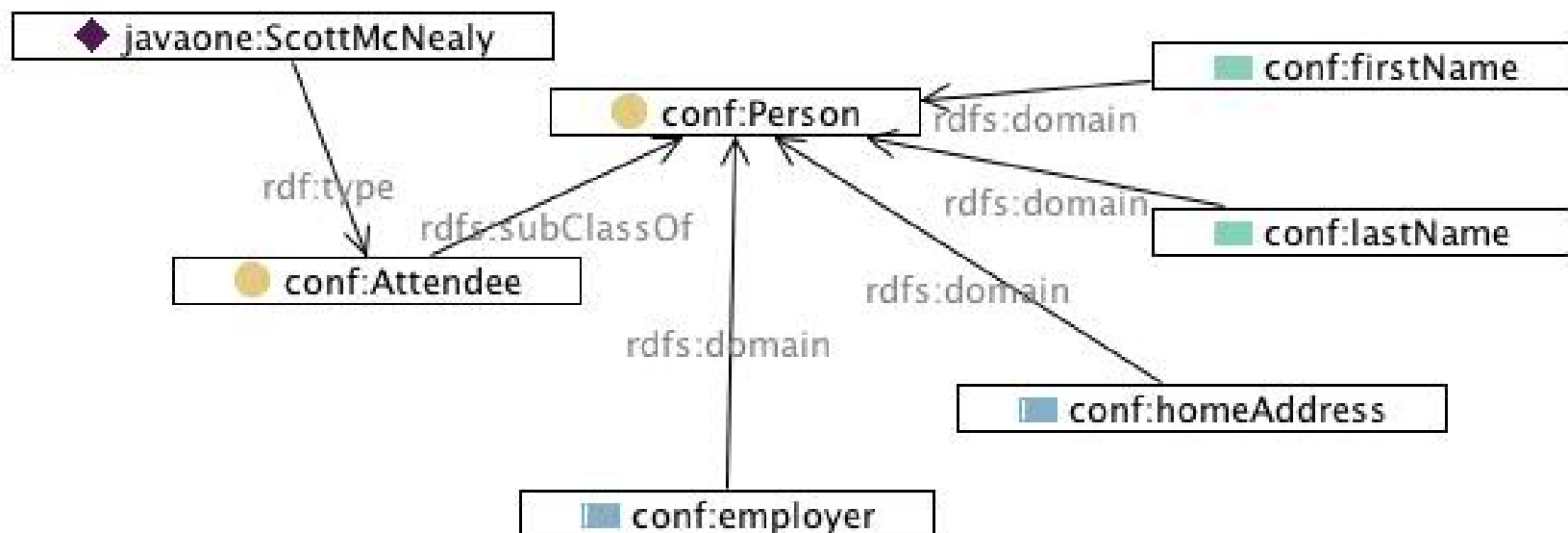


Subject	Predicate	Object
javaone:ScottMcNealy	rdf:type	conf:Attendee
conf:Attendee	rdfs:subClassOf	conf:Person

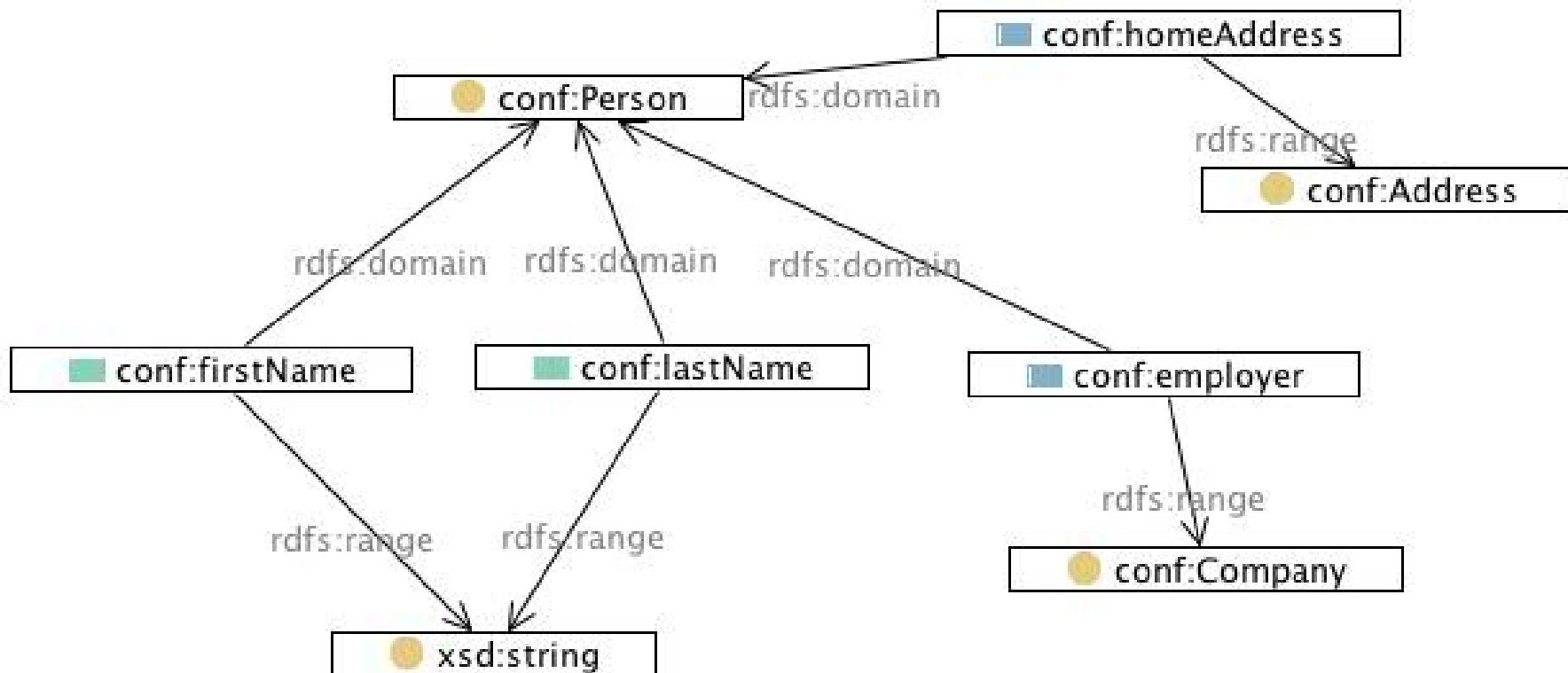
RDF Schema: Class Hierarchy



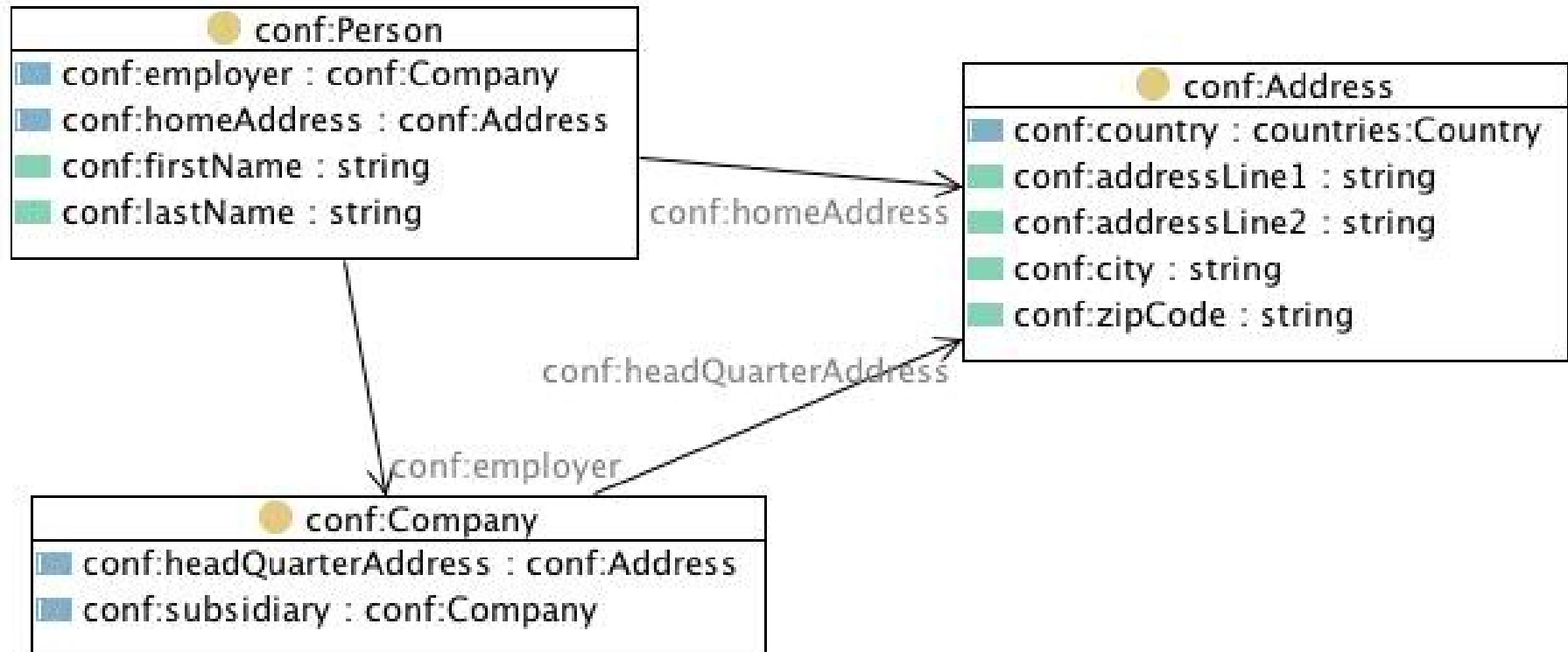
RDF Schema: Property Domains



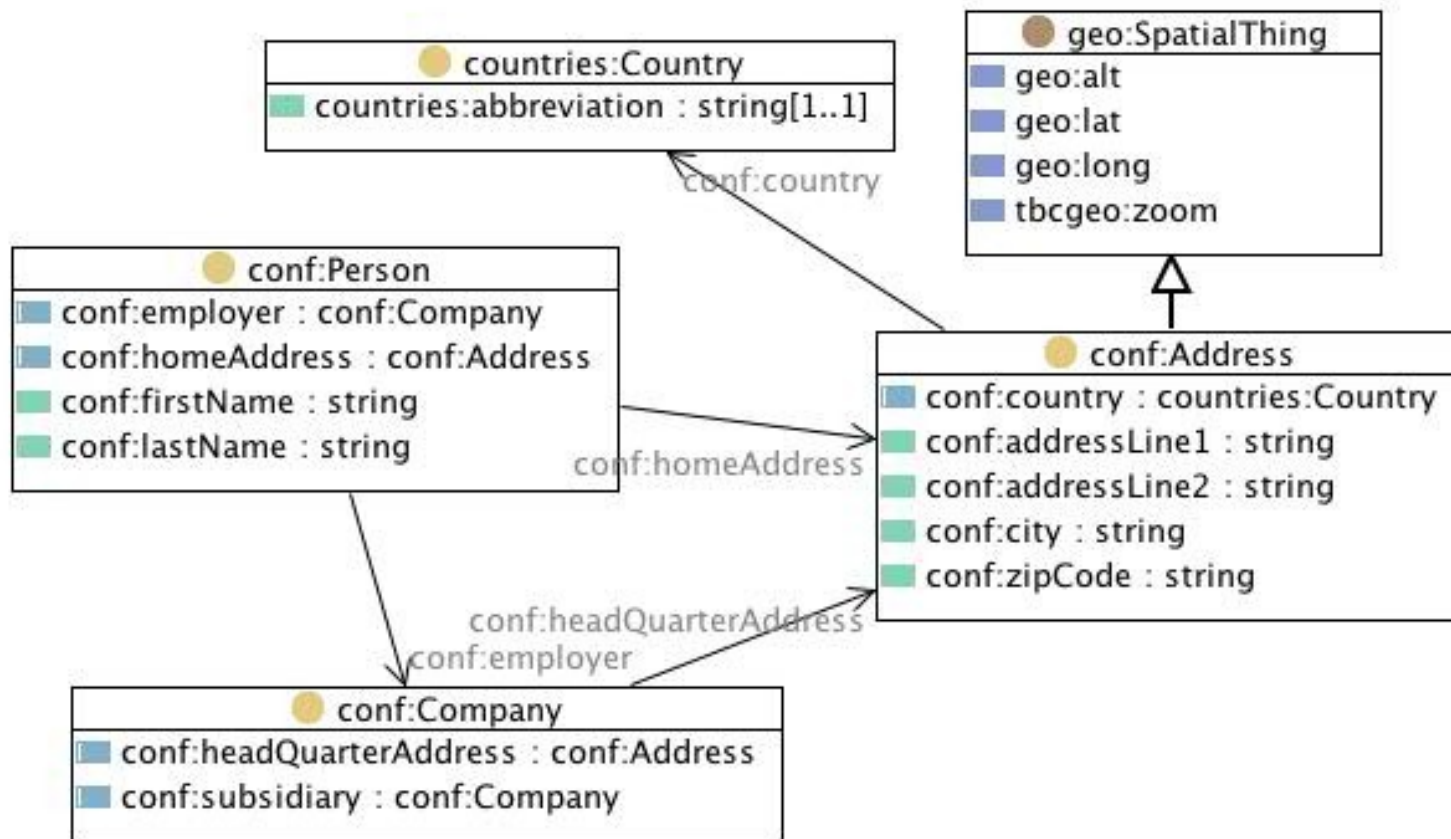
RDF Schema: Property Ranges



RDF Schema: from a UML point of view



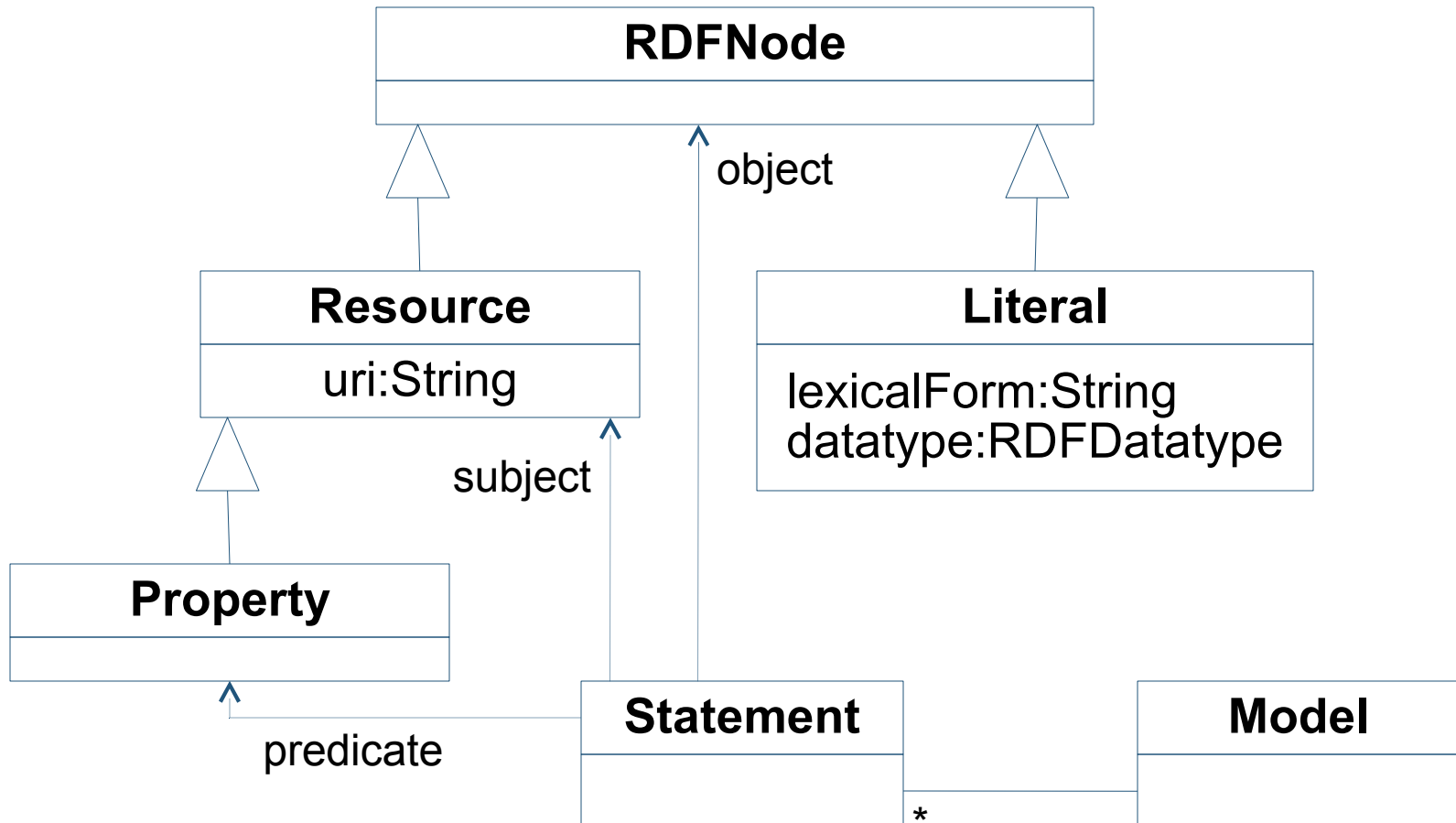
RDF Schema: linking namespaces



General Java APIs for RDF

- > Jena
 - <http://jena.sourceforge.net/>
- > Sesame
 - <http://www.openrdf.org/>

Jena API: Core interfaces (simplified)



Jena API: “Hello, World”

// Create a Jena Model and init some namespace prefixes

```
String NS = "http://examples.topquadrant.com/conferences/conf#";
```

```
Model model = ModelFactory.createDefaultModel();
```

```
model.setNsPrefix("conf", NS);
```

```
model.setNsPrefix("rdfs", RDFS.getURI());
```

```
model.setNsPrefix("xsd", XSD.getURI());
```

// Create an example triple

```
Resource javaOne = model.createResource(NS + "JavaOne2009");
```

```
Literal label = model.createTypedLiteral("JavaOne 2009");
```

```
model.add(javaOne, RDFS.label, label);
```

// Write the Model in N3 format to the screen

```
model.write(System.out, FileUtils.langN3);
```


Jena API: “Hello, World” Output

@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

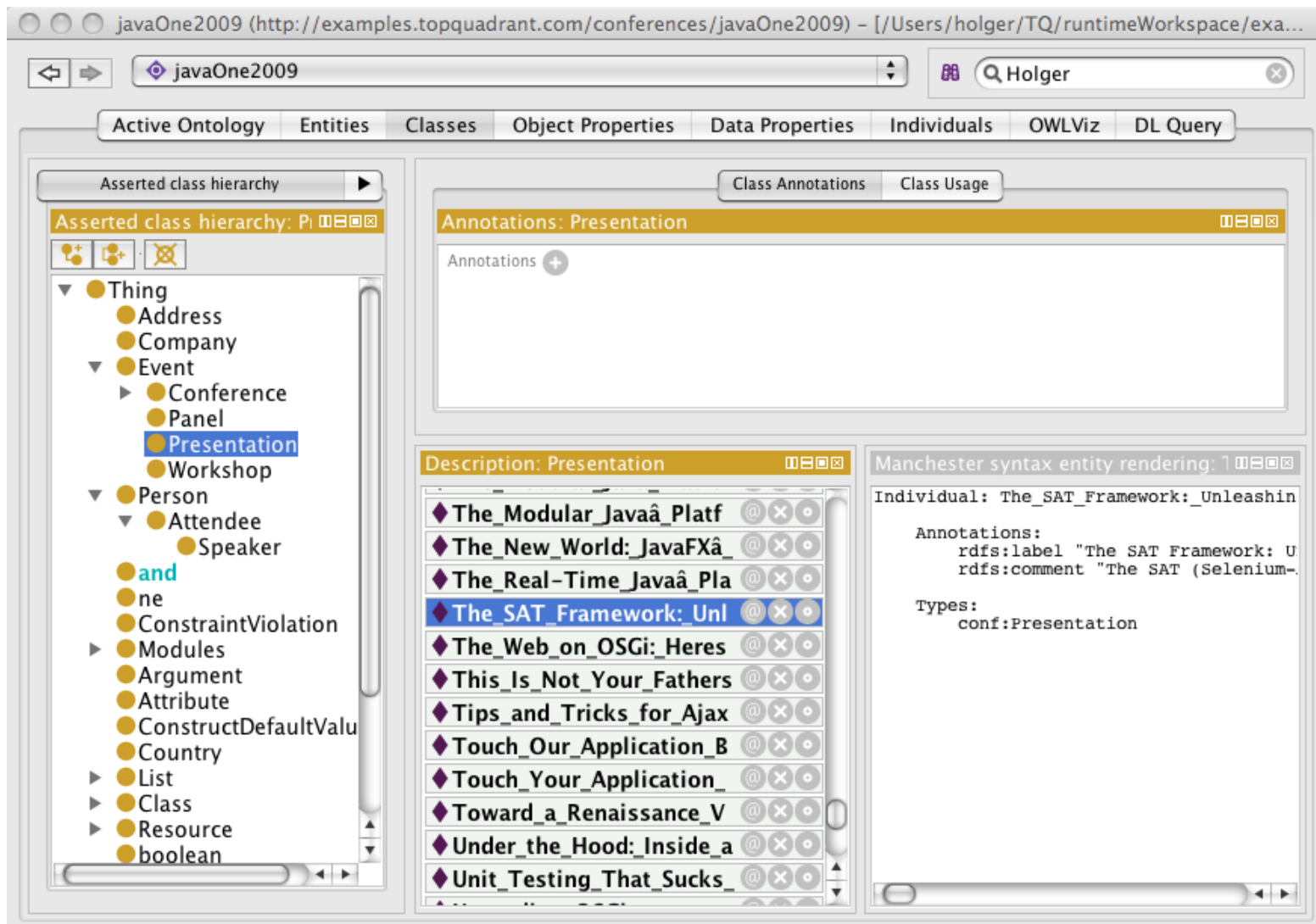
@prefix conf: <http://examples.topquadrant.com/conferences/conf#> .

conf:JavaOne2009

rdfs:label "JavaOne 2009"^^xsd:string .

RDF file in N3 format
Options include RDF/XML

Tools: OWL ontology editor Protege



javaOne2009 (http://examples.topquadrant.com/conferences/javaOne2009) - [/Users/holger/TQ/runtimeWorkspace/exa...]

javaOne2009

Active Ontology | Entities | Classes | Object Properties | Data Properties | Individuals | OWLViz | DL Query

Asserted class hierarchy

Asserted class hierarchy: Pi

- Thing
 - Address
 - Company
 - Event
 - Conference
 - Panel
 - Presentation**
 - Workshop
 - Person
 - Attendee
 - Speaker
 - and
 - ne
 - ConstraintViolation
 - Modules
 - Argument
 - Attribute
 - ConstructDefaultValu
 - Country
 - List
 - Class
 - Resource
 - boolean

Class Annotations | Class Usage

Annotations: Presentation

Annotations +

Description: Presentation

- The_Modular_Javaâ_Platf
- The_New_World:_JavaFXâ
- The_Real-Time_Javaâ_Pla
- The SAT Framework: Unl**
- The_Web_on_OSGi:_Heres
- This_Is_Not_Your_Fathers
- Tips_and_Tricks_for_Ajax
- Touch_Our_Application_B
- Touch_Your_Application_
- Toward_a_Renaissance_V
- Under_the_Hood:_Inside_a
- Unit_Testing_That_Sucks

Manchester syntax entity rendering: 1

Individual: The_SAT_Framework:_Unleashin

Annotations:

- rdfs:label "The SAT Framework: U
- rdfs:comment "The SAT (Selenium-

Types:

- conf:Presentation

Tools: TopBraid Composer

TopBraid – examples.topquadrant.com/conferences/javaOne.rdf – Eclipse SDK

conf:Speaker

Classes

- owl:Thing (1373)
 - conf:Address (2)
 - conf:Company (198)
 - conf:Event (377)
 - conf:Conference (2)
 - javaone:JavaOneConference
 - conf:Panel
 - conf:Presentation (375)
 - conf:Workshop
 - conf:Person (550)
 - conf:Attendee (549)
 - conf:Speaker (547)**
 - countries:Country (246)
 - owl:Nothing

Diagram

conf.n3

javaOne.rdf

conf:Person

- conf:employer : conf:Company
- conf:homeAddress : conf:Address
- conf:firstName : string
- conf:lastName : string

conf:Presentation

conf:Attendee

- conf:attendeeOf : conf:Event

conf:presenterOf

Properties

- conf:attendeeOf
- conf:country
- conf:employer
- conf:headQuarterAddress
- conf:homeAddress
- conf:part
- conf:sponsorOf
- conf:subsidiary
- conf:addressLine1
- conf:addressLine2
- conf:city
- conf:endTime

Navigator

- conferences 864
 - examples.topquadrant.com.confe
 - conf.n3 873 [http://examples.to
 - createJavaOne2009.n3 868 [http
 - javaOne.rdf 873** [http://exampl
 - javaOne.rdf.tbc 873

Form **Diagram** **Graph** **Form Layout** **Source Code**

Import **Instan** **Domain** **Releva** **SPARQ** **Rules**

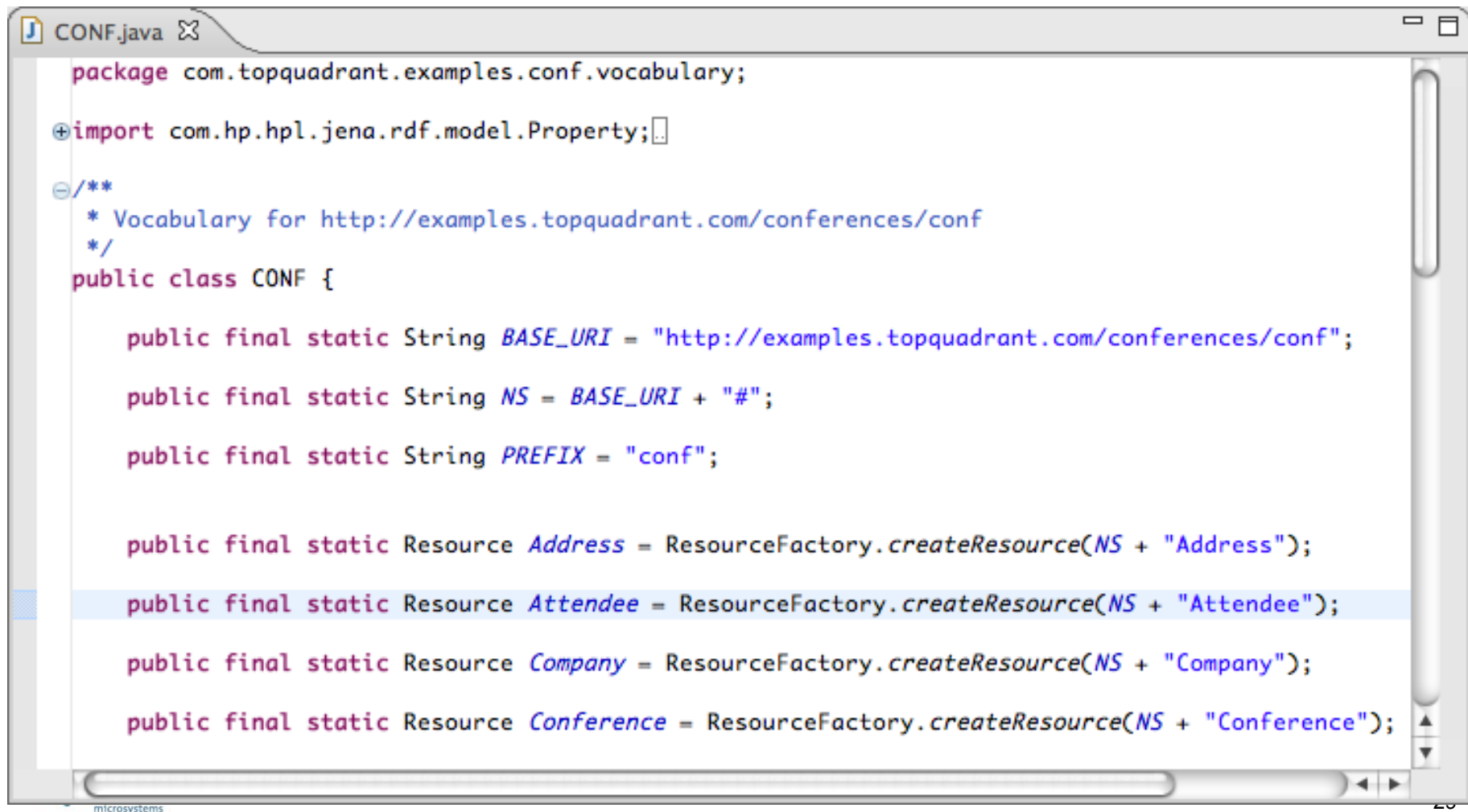
[Resource]	rdfs:label	rdfs:comment
javaone:Aaron_Mulder	Aaron Mulder	
javaone:Aaron_Walker	Aaron Walker	
javaone:Adam_Bien	Adam Bien	
javaone:Adam_Gross	Adam Gross	
javaone:Adam_Mollenkopf	Adam Mollenkopf	
javaone:Adam_Sotona	Adam Sotona	

Basket

Development Methodology

- > Define/re-use schema
- > Play with example instances, queries etc
- > Take a data-centric approach!
- > Link key classes of the schema to your Java code (Data Binding)
- > Assume schema evolution
- > Develop smart, generic code instead of domain-specific code

Data Binding: Generated schema class



```
CONF.java
package com.topquadrant.examples.conf.vocabulary;

import com.hp.hpl.jena.rdf.model.Property;

/**
 * Vocabulary for http://examples.topquadrant.com/conferences/conf
 */
public class CONF {

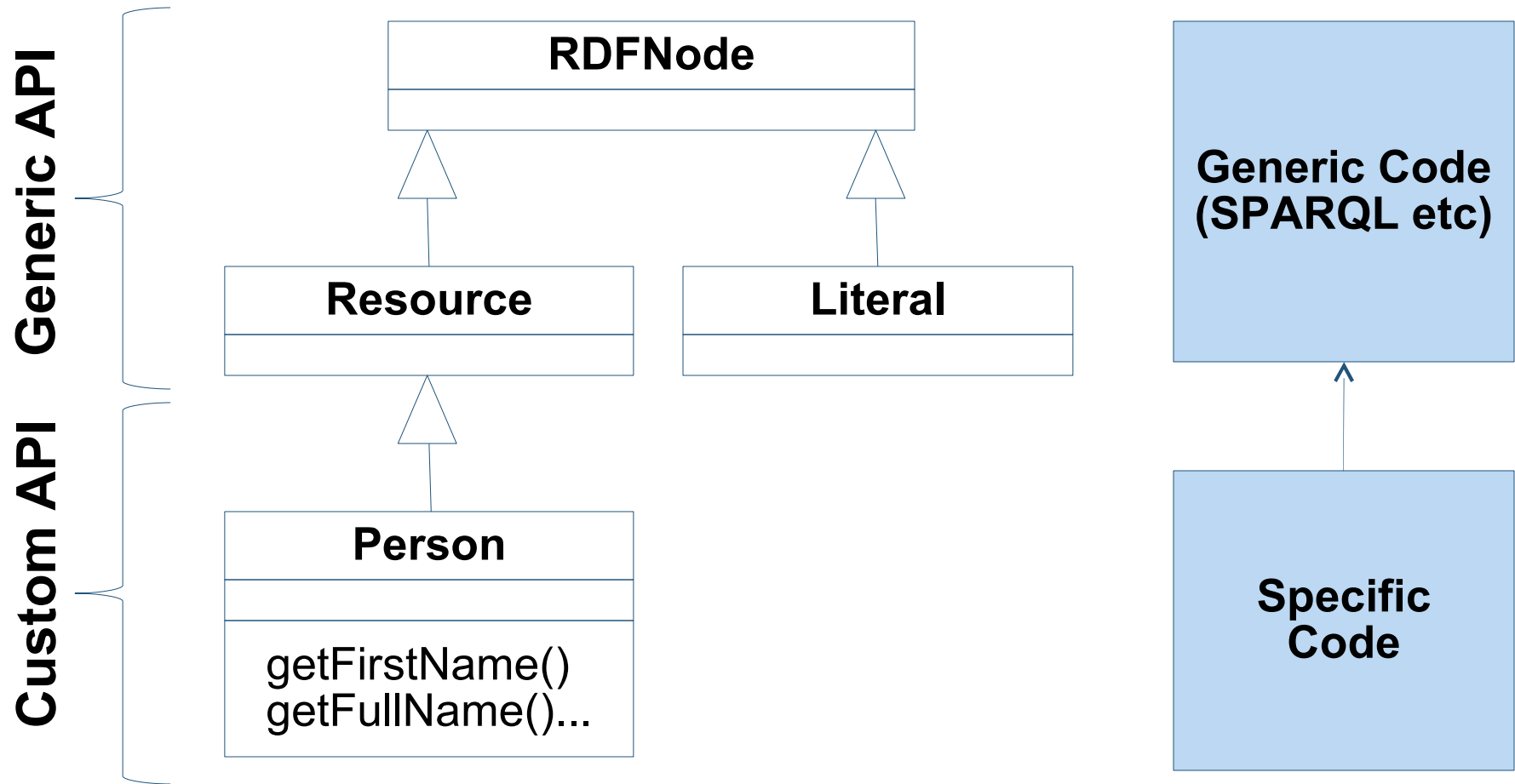
    public final static String BASE_URI = "http://examples.topquadrant.com/conferences/conf";

    public final static String NS = BASE_URI + "#";

    public final static String PREFIX = "conf";

    public final static Resource Address = ResourceFactory.createResource(NS + "Address");
    public final static Resource Attendee = ResourceFactory.createResource(NS + "Attendee");
    public final static Resource Company = ResourceFactory.createResource(NS + "Company");
    public final static Resource Conference = ResourceFactory.createResource(NS + "Conference");
}
```

Data Binding: Basic Architecture



Data Binding: Example use

```
Model model = ModelFactory.createDefaultModel();
```

```
// Create a new Person instance
```

```
String steveURI = CONF.NS + "SteveJobs";
```

```
Person steve = ConfFactory.createPerson(model, steveURI);
```

```
// Set some property values
```

```
steve.setFirstName("Steve");
```

```
steve.setLastName("Jobs");
```

```
// Query some property values
```

```
String fullName = steve.getFullName();
```

```
System.out.println("Person: " + fullName);
```

Data Binding: Example interface

public interface Person **extends** Resource {

String getFirstName();

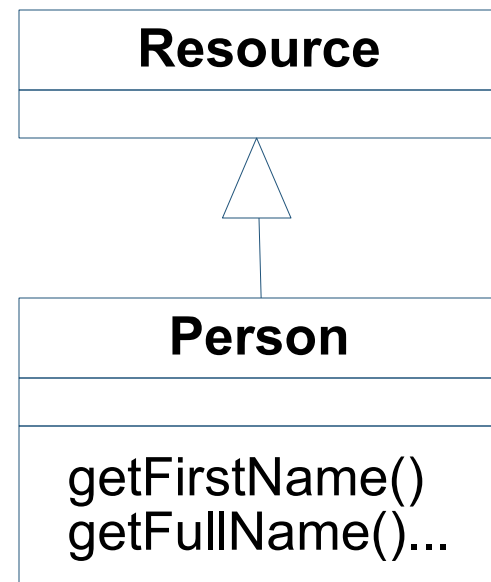
String getFullName();

String getLastName();

void setFirstName(String value);

void setLastName(String value);

}



Data Binding: Example implementation

```
public class PersonImpl extends ResourceImpl implements Person {  
  
    public PersonImpl(Node node, EnhGraph eg) {  
        super(node, eg);  
    }  
  
    public String getFirstName() {  
        return getStringProperty(CONF.firstName);  
    }  
  
    public String getFullName() {  
        return getFirstName() + " " + getLastName();  
    }  
}
```

Data Binding: Factory class

```
public class ConfFactory {  
    static {  
        register(Person.class, PersonImpl.class);  
    }  
    public static Person createPerson(Model model, String uri) {  
        return model.createResource(uri, CONF.Person).as(Person.class);  
    }  
    public static Person getPerson(Model model, String uri) {  
        return model.createResource(uri).as(Person.class);  
    }  
    ...  
}
```

Data Binding: Frameworks

> ELMO

- <http://www.openrdf.org/doc/elmo/1.4/>

> Sommer

- <https://sommer.dev.java.net/sommer/index.html>

> JenaBean

- <http://code.google.com/p/jenabean/>
- <http://www.incunabulum.de/projects/it/owl2java/>

> Kazuki

- <http://projects.semwebcentral.org/projects/kazuki/>

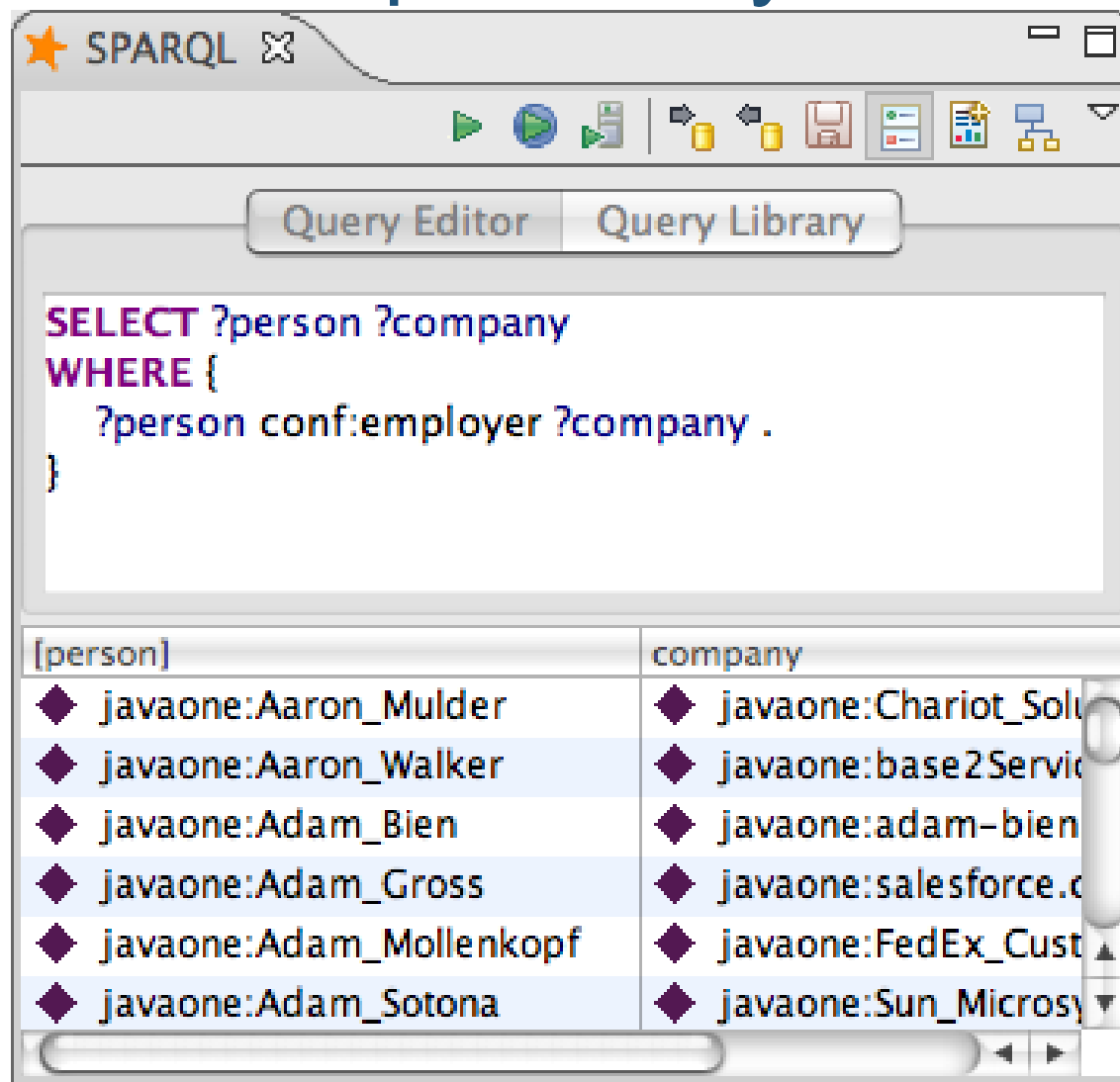
Data Binding: Java code generators

- > RDF Reactor
 - <http://semanticweb.org/wiki/RDFReactor>
- > ELMO
 - <http://www.openrdf.org/doc/elmo/1.4/>
- > Jastor
 - <http://jastor.sourceforge.net/>
- > OWL2Java

Intermediate Summary

- > Data model in RDF and RDF Schema
- > Data binding to link RDF and Java worlds
- > Next: let the data work for us with SPARQL

SPARQL: Example Query



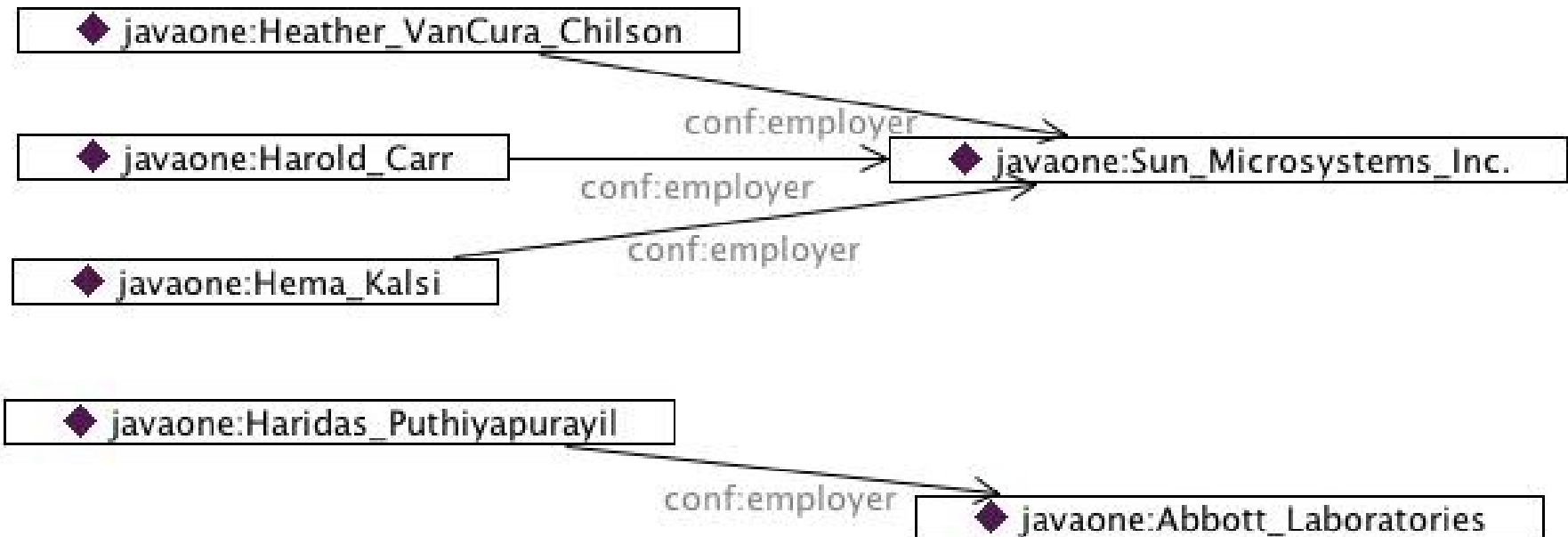
The screenshot shows a web application window titled "SPARQL". It has a toolbar with icons for running queries, saving, and other functions. Below the toolbar are two tabs: "Query Editor" and "Query Library". The "Query Editor" tab is active, displaying a SPARQL query in a text area:

```
SELECT ?person ?company
WHERE {
  ?person conf:employer ?company .
}
```

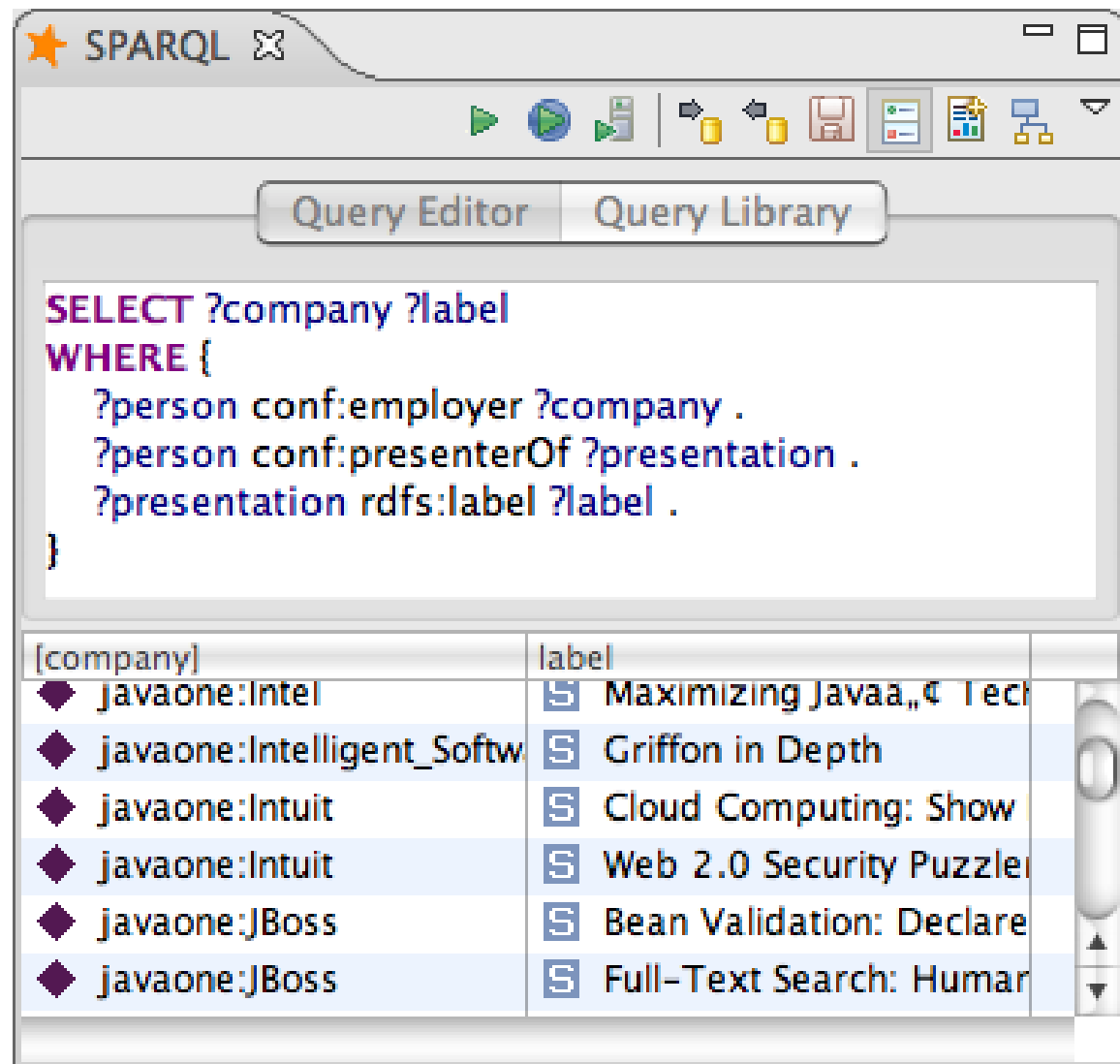
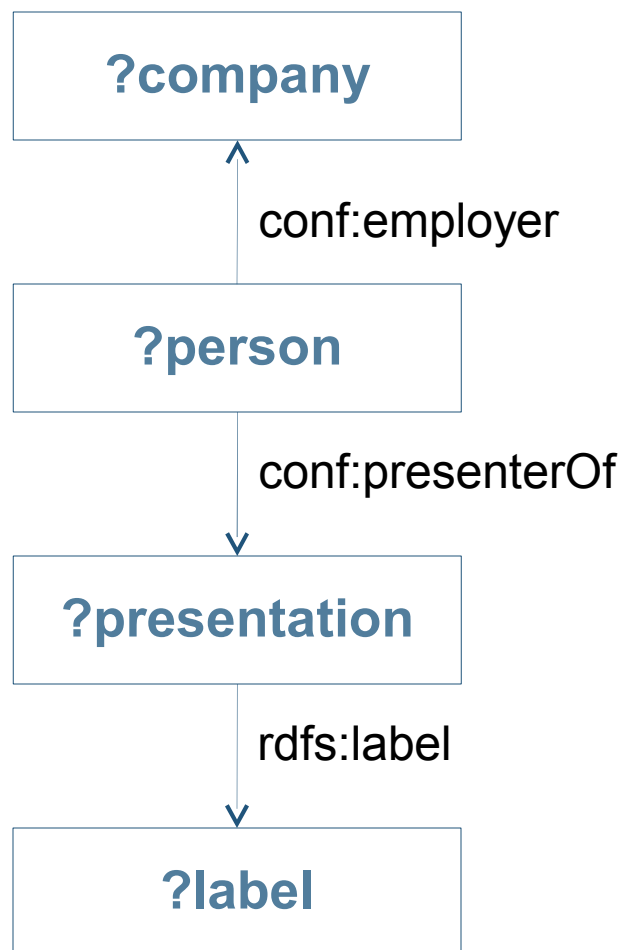
Below the query editor, the results are displayed in a table with two columns: "[person]" and "company". Each row contains a diamond icon followed by a URI. The results are as follows:

[person]	company
◆ javaone:Aaron_Mulder	◆ javaone:Chariot_Solutions
◆ javaone:Aaron_Walker	◆ javaone:base2Service
◆ javaone:Adam_Bien	◆ javaone:adam-bien
◆ javaone:Adam_Gross	◆ javaone:salesforce.com
◆ javaone:Adam_Mollenkopf	◆ javaone:FedEx_Customer
◆ javaone:Adam_Sotona	◆ javaone:Sun_Microsystems

SPARQL: Query results as subgraphs



SPARQL: Example 2



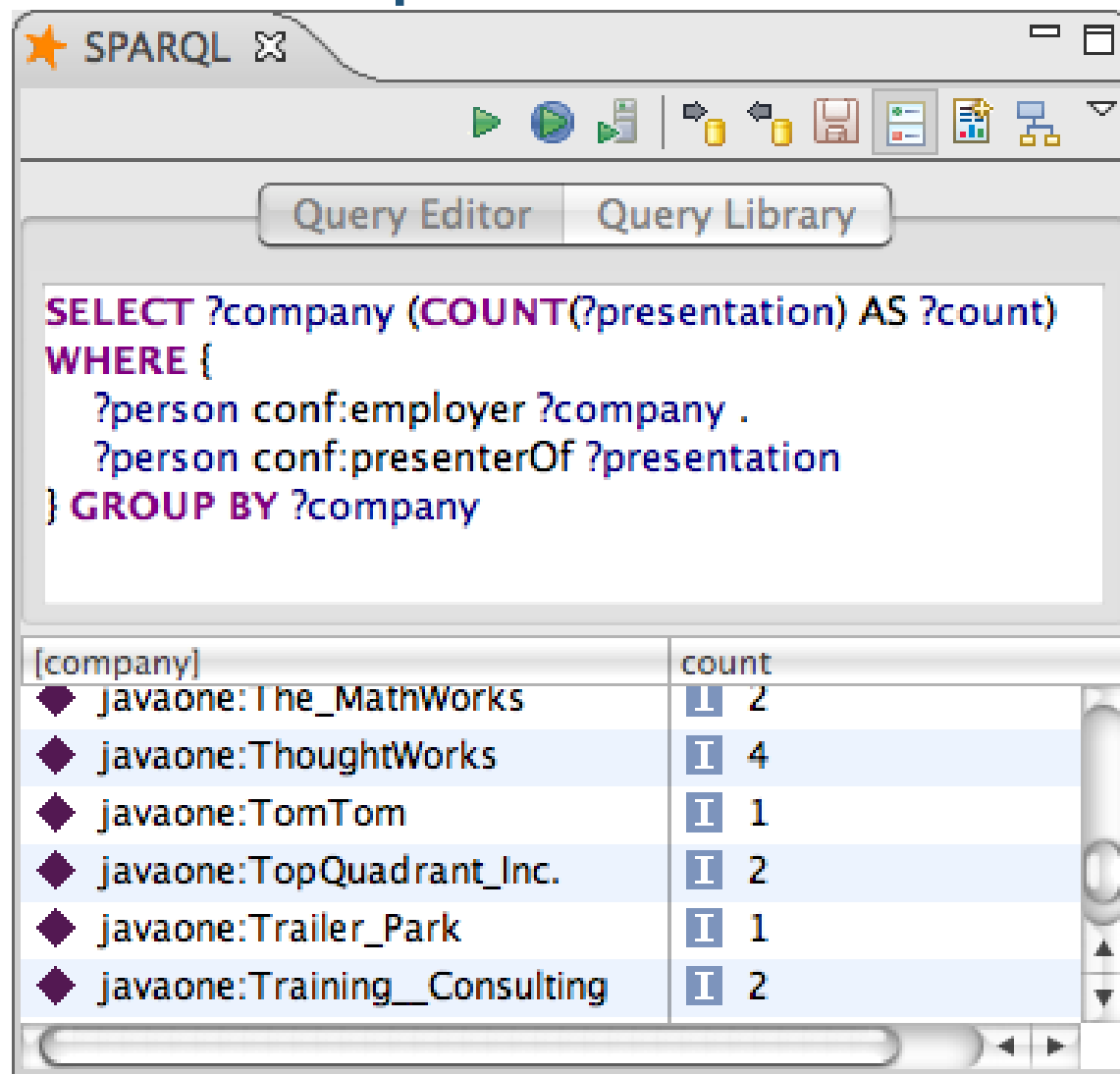
A screenshot of a SPARQL query editor and results window. The window title is "SPARQL". It contains a "Query Editor" tab with the following query:

```
SELECT ?company ?label
WHERE {
  ?person conf:employer ?company .
  ?person conf:presenterOf ?presentation .
  ?presentation rdfs:label ?label .
}
```

Below the query editor, the results are displayed in a table with two columns: **[company]** and **label**.

[company]	label
javaone:Intel	Maximizing Javaa,,C Tech
javaone:Intelligent_Softw	Griffon in Depth
javaone:Intuit	Cloud Computing: Show
javaone:Intuit	Web 2.0 Security Puzzler
javaone:JBoss	Bean Validation: Declare
javaone:JBoss	Full-Text Search: Human

SPARQL: Example 3



The screenshot shows a SPARQL query editor window. The title bar is labeled "SPARQL". Below the title bar is a toolbar with various icons. The main area is divided into two tabs: "Query Editor" and "Query Library". The "Query Editor" tab is active, displaying the following SPARQL query:

```
SELECT ?company (COUNT(?presentation) AS ?count)
WHERE {
  ?person conf:employer ?company .
  ?person conf:presenterOf ?presentation
} GROUP BY ?company
```

Below the query editor, the results are displayed in a table with two columns: "[company]" and "count". The results are as follows:

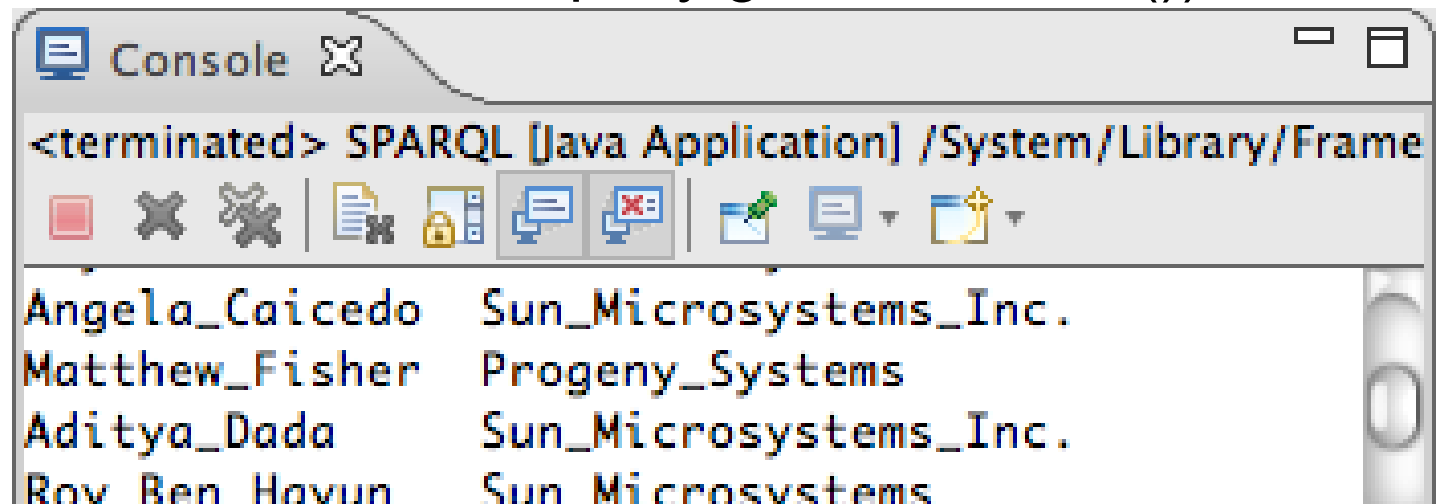
[company]	count
javaone:The_MathWorks	2
javaone:ThoughtWorks	4
javaone:TomTom	1
javaone:TopQuadrant_Inc.	2
javaone:Trailer_Park	1
javaone:Training__Consulting	2

Jena API: SPARQL

```
Model m = ModelFactory.createDefaultModel();
m.read(new FileReader("conf.n3"), null, FileUtils.langN3);
m.read(new FileReader("javaOne.rdf"), null);
String queryString =
    "PREFIX " + CONF.PREFIX + ": <" + CONF.NS + "> \n" +
    "SELECT ?person ?company \n" +
    "WHERE { \n" +
    "    ?person conf:employer ?company . \n" +
    "}";
Query query = QueryFactory.create(queryString);
QueryExecution qx = QueryExecutionFactory.create(query, m);
```

Jena API: SPARQL (cont'd)

```
ResultSet rs = qx.execSelect();  
while(rs.hasNext()) {  
    QuerySolution s = rs.nextSolution();  
    Resource person = (Resource) s.get("person");  
    Resource company = (Resource) s.get("company");  
    System.out.println(person.getLocalName() + "\t" +  
                        company.getLocalName());  
}
```

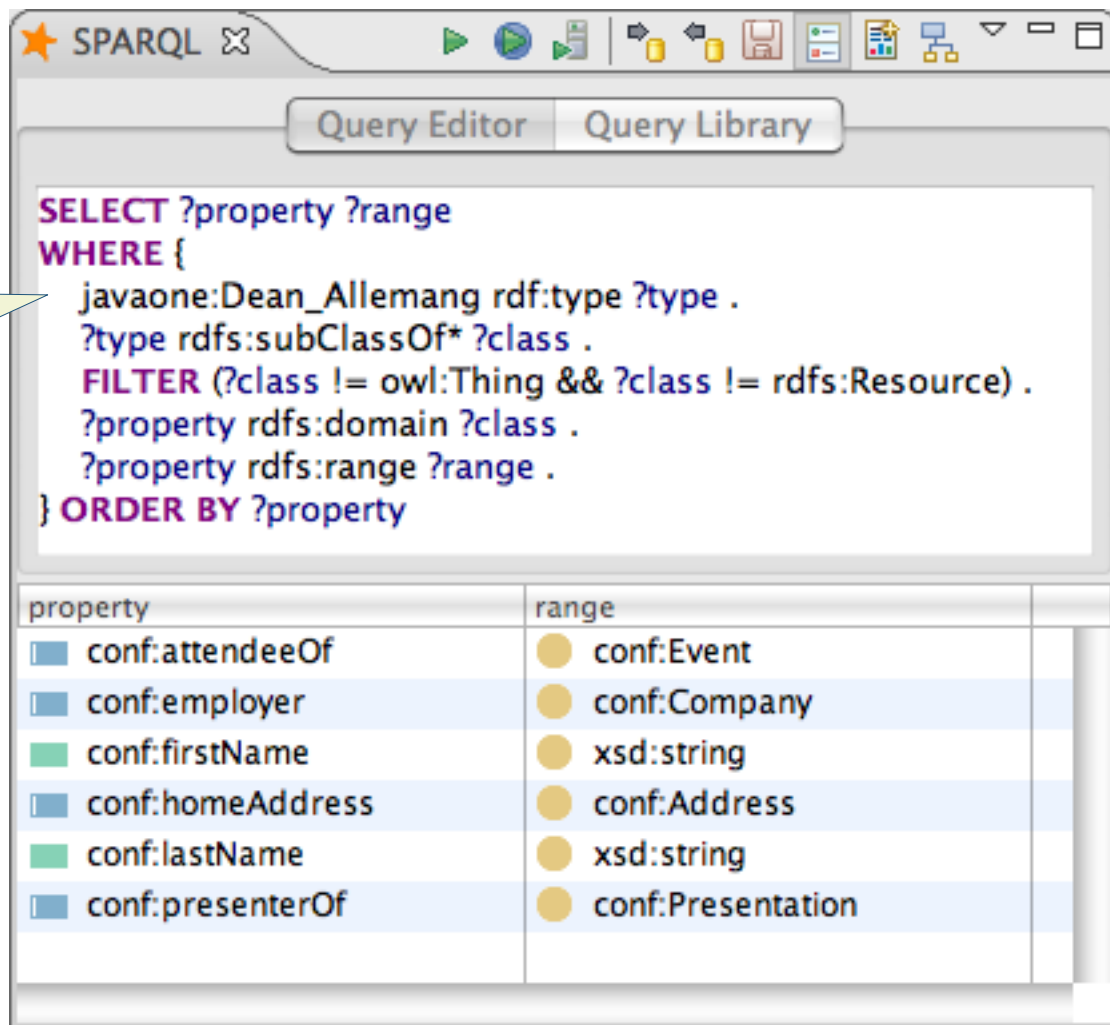


The screenshot shows a Java IDE console window titled "Console". The output displays the results of a SPARQL query, showing four rows of data. Each row contains a person's name and their associated company name, separated by a tab character. The results are: Angela_Caicedo Sun_Microsystems_Inc., Matthew_Fisher Progeny_Systems, Aditya_Dada Sun_Microsystems_Inc., and Roy Ben Havun Sun Microsystems.

Person	Company
Angela_Caicedo	Sun_Microsystems_Inc.
Matthew_Fisher	Progeny_Systems
Aditya_Dada	Sun_Microsystems_Inc.
Roy Ben Havun	Sun Microsystems

SPARQL: Querying the schema

“Which properties are relevant for a given resource?”



The screenshot shows a SPARQL Query Editor window with a toolbar and two tabs: "Query Editor" and "Query Library". The query editor contains the following SPARQL query:

```
SELECT ?property ?range
WHERE {
  javaone:Dean_Allemang rdf:type ?type .
  ?type rdfs:subClassOf* ?class .
  FILTER (?class != owl:Thing && ?class != rdfs:Resource) .
  ?property rdfs:domain ?class .
  ?property rdfs:range ?range .
} ORDER BY ?property
```

Below the query editor, a table displays the results of the query. The table has two columns: "property" and "range".

property	range
conf:attendeeOf	conf:Event
conf:employer	conf:Company
conf:firstName	xsd:string
conf:homeAddress	conf:Address
conf:lastName	xsd:string
conf:presenterOf	conf:Presentation

SPARQL: Model-driven forms

javaOne.rdf x conf.n3

Resource Form

Name: OK

▼ Annotations

rdfs:label ▼

▼ Other Properties

conf:attendeeOf ▼

conf:employer ▼

conf:firstName ▼

conf:homeAddress ▼

conf:lastName ▼

conf:presenterOf ▼

rdf:type ▼

▼ Incoming References

Form Graph Source Code

javaone:HolgerKnublau...

▼ Annotations

comment:

label:

▼ Other Properties

address line... 20395 Silverado Ave

address line...

altitude:

city: Cupertino

country: [United States](#)

latitude: 37.31543524495951

longitude: -122.03045189380646

tbcgeo:zoom: 18

zip code: 95014

▼ Incoming References

SPARQL: Model-driven applications

http://localhost:8083/tbl/app/server.topbraidlive.org/dynamic/applications/Default%20Applicatio - Windows Internet Explorer

http://localhost:8083/tbl/app/server.topbraidlive.org/dynamic/applications/Default%20Application%20(configurable).n3/-/examples.topquadrant... composing semantic web

Composing the Semantic We... TopQuadrant | Resources | V... http://localhost:8083/tbl/...

Save App Save Data Add Component

powered by: TopBraid Suite

TopBraid Ensemble

Tree

- Address (2)
- Company (198)
- Country (246)
- Event
 - Conference
 - JavaOne Conference (2)
 - Panel
 - Workshop

Search Company

comment:

free passes:

head quarter...

label:

sponsor of:

Results Grid

Name
Sprint Nextel
SRA International, Inc.
SugarCRM
Sun Microsystems
Sun Microsystems India Pvt Ltd

Sun Microsystems

Visualization Yahoo Map

Annotations

comment:

label: Sun Microsystems

Other Properties

free passes:

head quarter address:

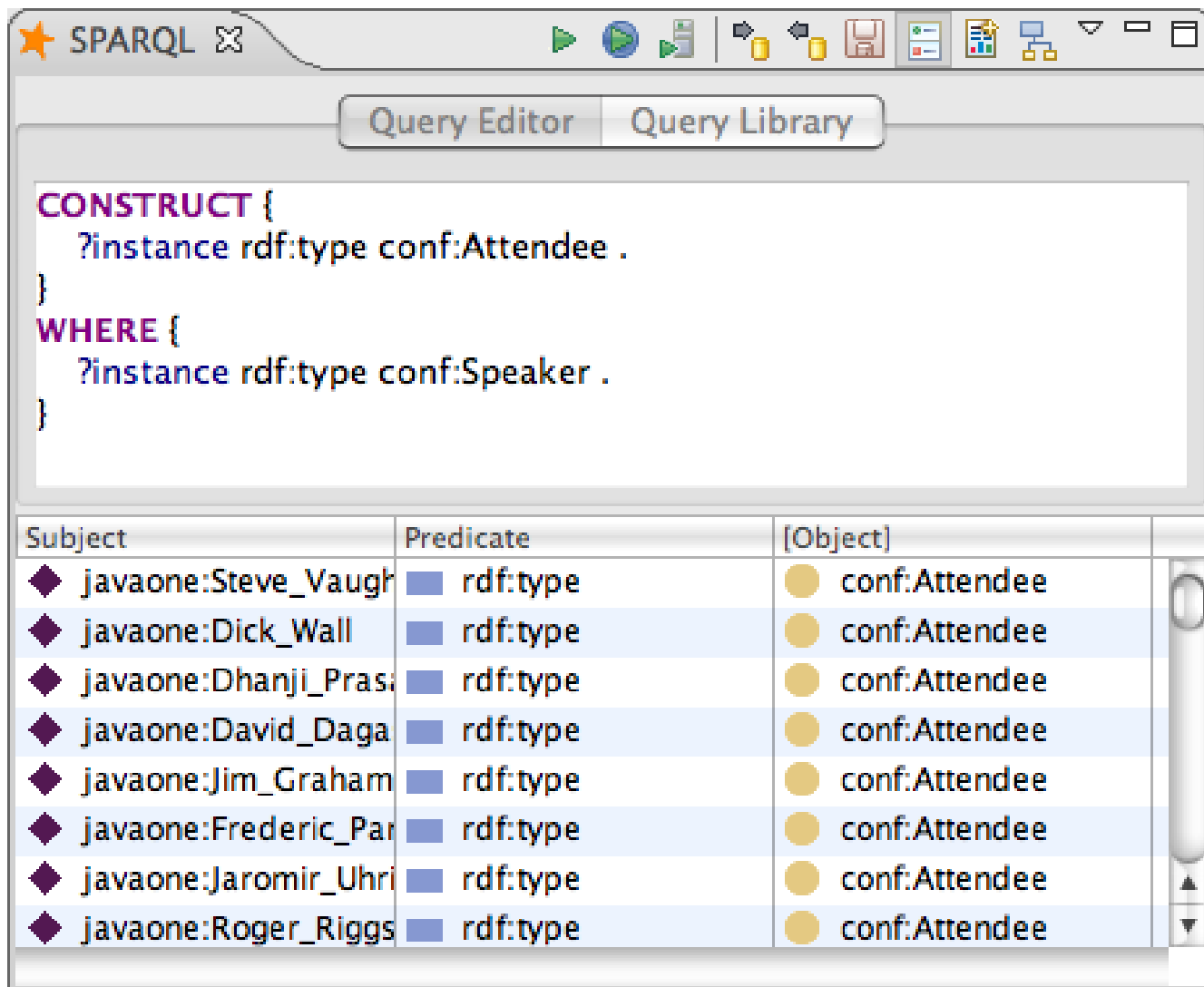
sameAs:

sponsor of:

subsidiary: [Sun Microsystems India Pvt Ltd](#)

Done Internet 100%

SPARQL: CONSTRUCT rule



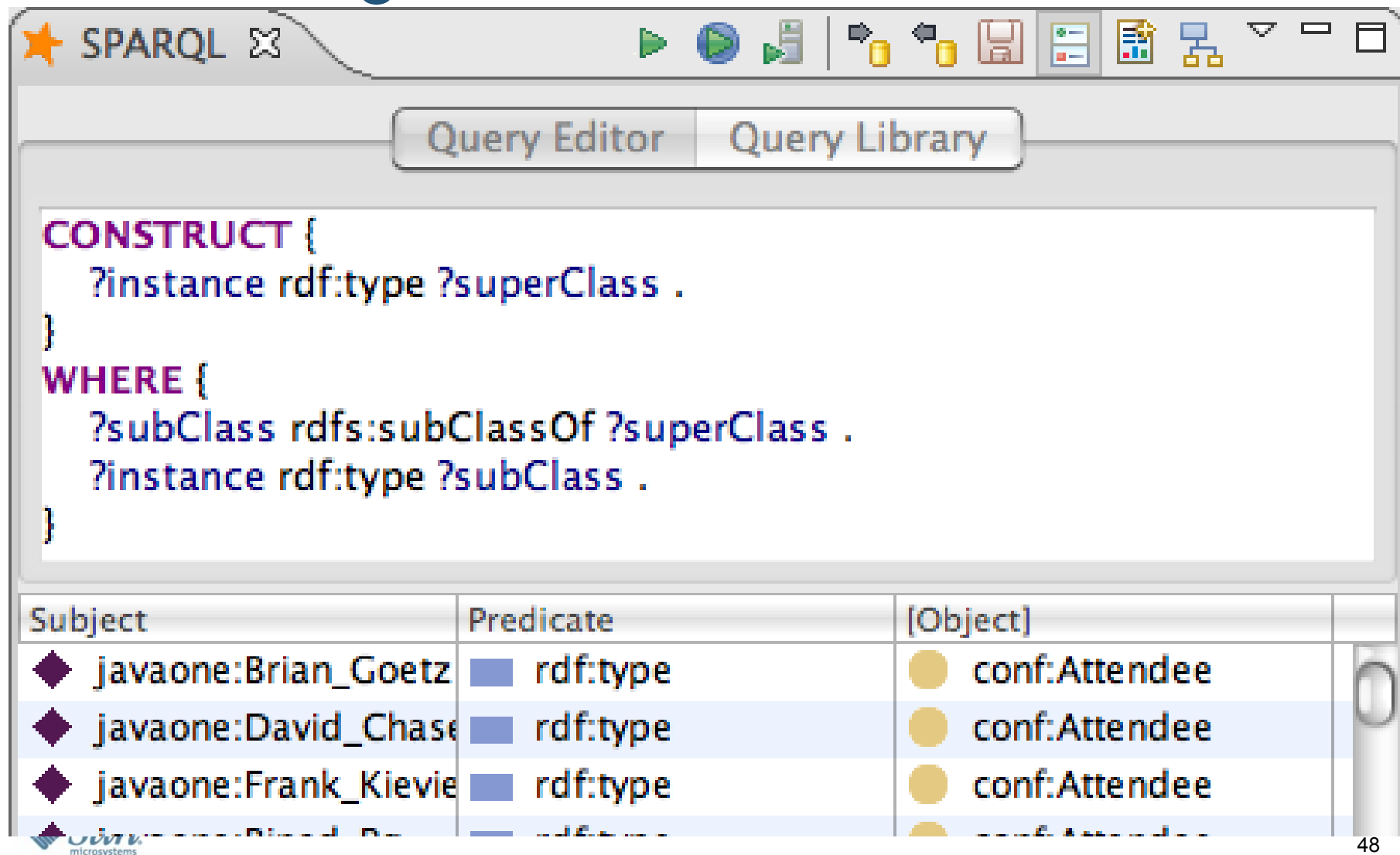
The screenshot shows a SPARQL Query Editor window. The title bar includes a star icon, the text "SPARQL", and a close button. The toolbar contains icons for running queries, saving, and other standard functions. The main area is divided into two tabs: "Query Editor" (active) and "Query Library". The "Query Editor" tab displays a SPARQL query:

```
CONSTRUCT {  
  ?instance rdf:type conf:Attendee .  
}  
WHERE {  
  ?instance rdf:type conf:Speaker .  
}
```

Below the query editor is a table showing the results of the query. The table has three columns: "Subject", "Predicate", and "[Object]". The results are as follows:

Subject	Predicate	[Object]
javaone:Steve_Vaugh	rdf:type	conf:Attendee
javaone:Dick_Wall	rdf:type	conf:Attendee
javaone:Dhanji_Pras	rdf:type	conf:Attendee
javaone:David_Daga	rdf:type	conf:Attendee
javaone:Jim_Graham	rdf:type	conf:Attendee
javaone:Frederic_Par	rdf:type	conf:Attendee
javaone:Jaromir_Uhri	rdf:type	conf:Attendee
javaone:Roger_Riggs	rdf:type	conf:Attendee

SPARQL: generalized rule



The screenshot shows a SPARQL query editor window. The title bar includes a star icon, the text "SPARQL", and a close button. The toolbar contains icons for running queries, saving, and other functions. The main area has two tabs: "Query Editor" and "Query Library". The "Query Editor" tab is active, displaying the following SPARQL query:




```
CONSTRUCT {  
  ?instance rdf:type ?superClass .  
}  
WHERE {  
  ?subClass rdfs:subClassOf ?superClass .  
  ?instance rdf:type ?subClass .  
}
```

Below the query editor is a table showing the results of the query. The table has three columns: "Subject", "Predicate", and "[Object]". The results are as follows:

Subject	Predicate	[Object]
javaone:Brian_Goetz	rdf:type	conf:Attendee
javaone:David_Chase	rdf:type	conf:Attendee
javaone:Frank_Kievie	rdf:type	conf:Attendee
javaone:David_Chase	rdf:type	conf:Attendee

The bottom left corner of the window shows the "JVM" logo and the text "microsystems". The bottom right corner of the window shows the number "48".

Rules: Extending the core schema

- ▼  `conf:sponsorOf`
 - ▼  `javaone:cosponsorOf`
 -  `javaone:platinumCosponsorOf`

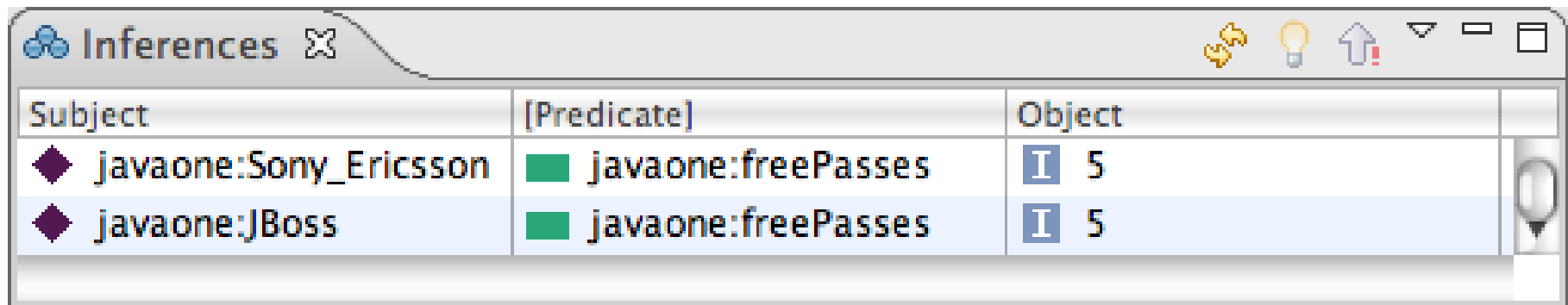
Adding domain-specific properties;
the application's code knows nothing
about them at compile-time.

Rules: Conference Sponsorship

“Companies that are co-sponsors get 5 free conference passes”

Condition
(triple pattern in the graph)

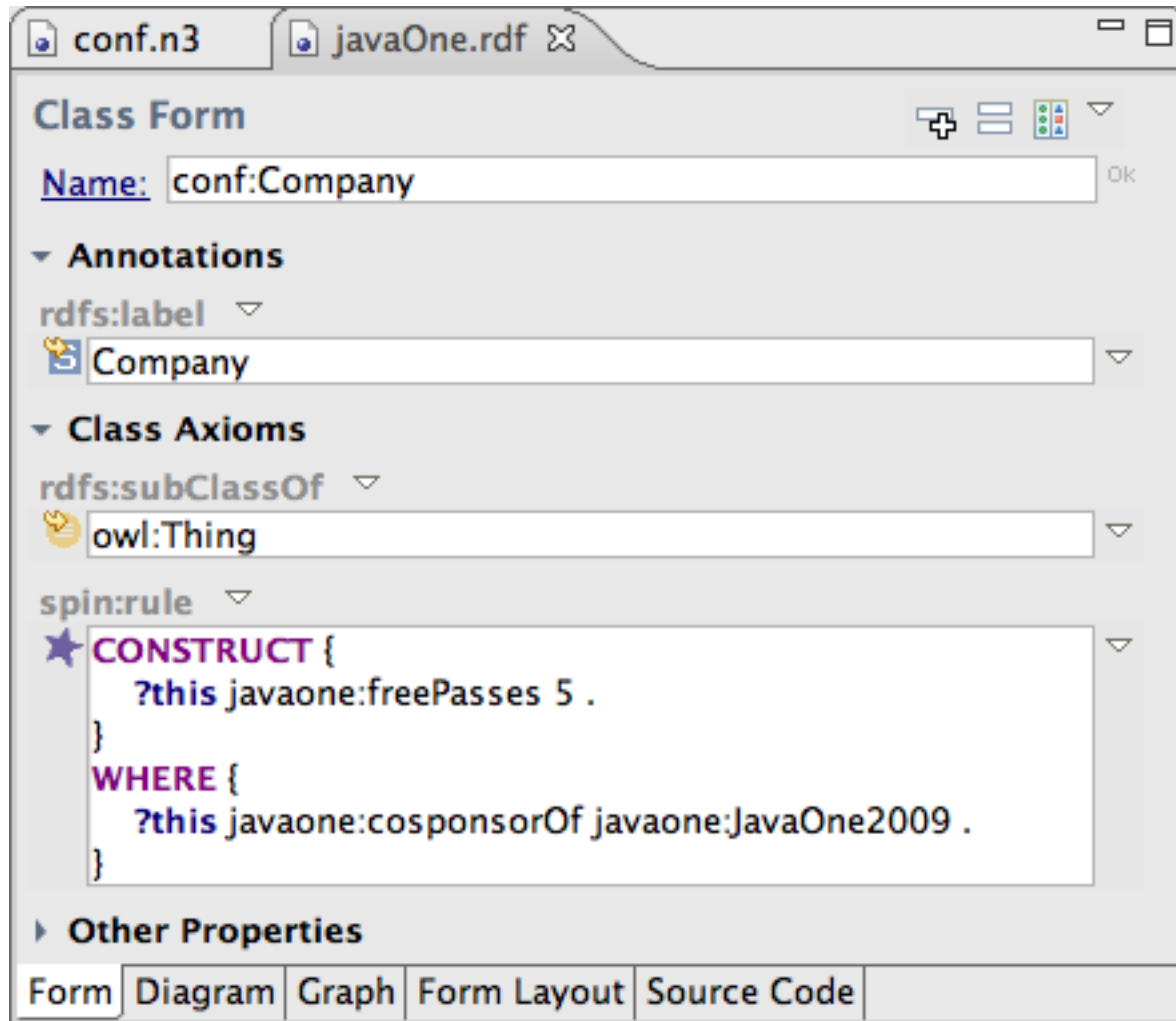
Action
(triples to add/infer)



The image shows a software window titled "Inferences" with a close button (X) and several icons (dollar signs, lightbulb, upward arrow, and window controls) in the top right corner. The window contains a table with three columns: "Subject", "[Predicate]", and "Object". There are two rows of data, each preceded by a purple diamond icon. The first row shows "javaone:Sony_Ericsson" as the subject, "javaone:freePasses" as the predicate, and "I 5" as the object. The second row shows "javaone:JBoss" as the subject, "javaone:freePasses" as the predicate, and "I 5" as the object.

Subject	[Predicate]	Object
◆ javaone:Sony_Ericsson	■ javaone:freePasses	I 5
◆ javaone:JBoss	■ javaone:freePasses	I 5

Rules: SPARQL Inferencing Notation



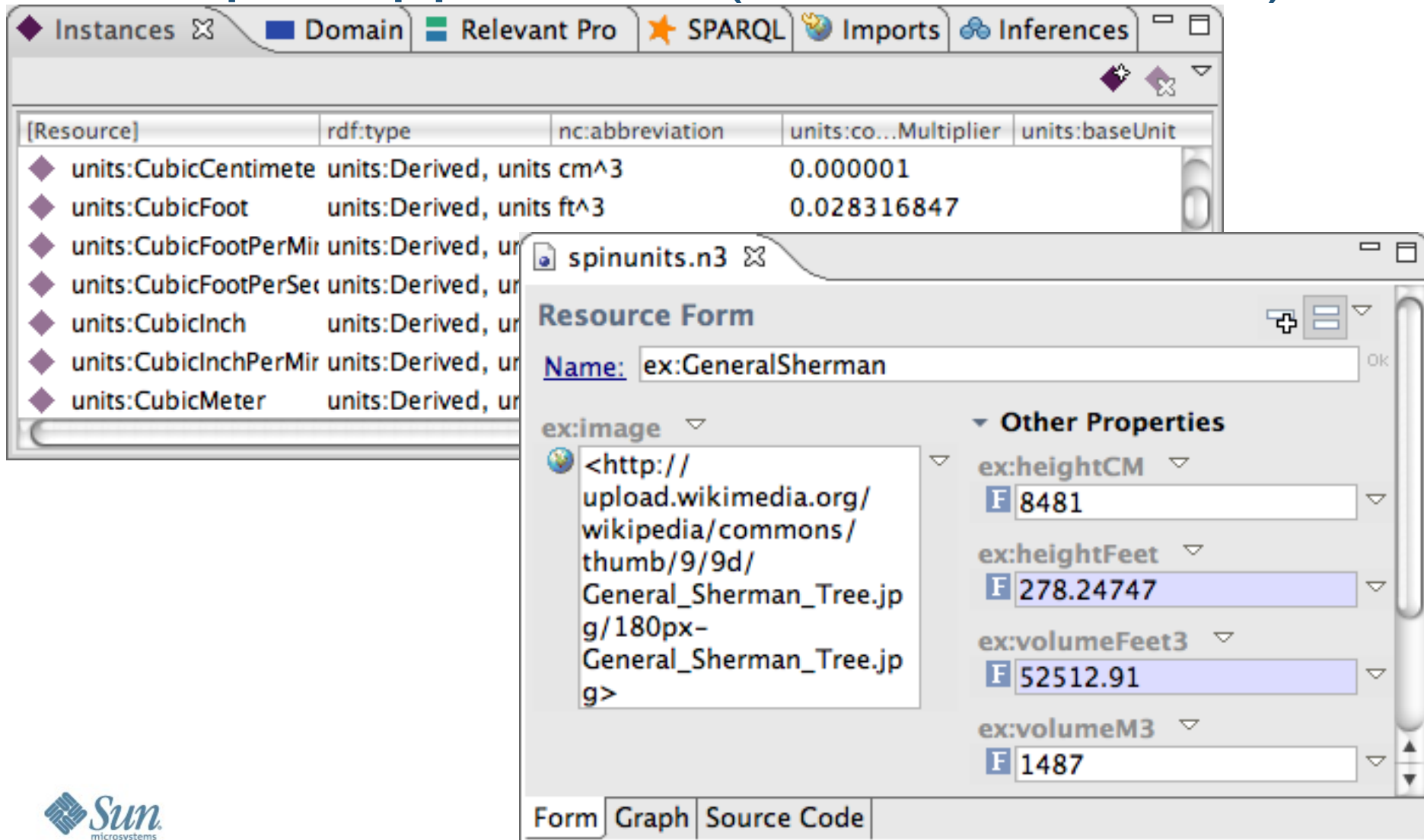
The screenshot shows a web-based IDE window with two tabs: `conf.n3` and `javaOne.rdf`. The `conf.n3` tab is active, displaying the "Class Form" for the class `conf:Company`. The form includes several sections:

- Name:** `conf:Company`
- Annotations:** `rdfs:label` with the value `Company`.
- Class Axioms:**
 - `rdfs:subClassOf` with the value `owl:Thing`.
 - `spin:rule` with a SPARQL query:

```
CONSTRUCT {  
  ?this javaone:freePasses 5 .  
}  
WHERE {  
  ?this javaone:cosponsorOf javaone:JavaOne2009 .  
}
```
- Other Properties:** (collapsed)

At the bottom, there are five tabs: `Form`, `Diagram`, `Graph`, `Form Layout`, and `Source Code`. The `Form` tab is currently selected.

Example Application (Unit Conversion)



The screenshot displays a Java application interface with a table of units and a 'Resource Form' dialog box.

Units Table:

[Resource]	rdf:type	nc:abbreviation	units:co...Multiplier	units:baseUnit
units:CubicCentimete	units:Derived, units	cm^3	0.000001	
units:CubicFoot	units:Derived, units	ft^3	0.028316847	
units:CubicFootPerMir	units:Derived, ur			
units:CubicFootPerSec	units:Derived, ur			
units:CubicInch	units:Derived, ur			
units:CubicInchPerMir	units:Derived, ur			
units:CubicMeter	units:Derived, ur			

Resource Form Dialog:

Name: ex:GeneralSherman

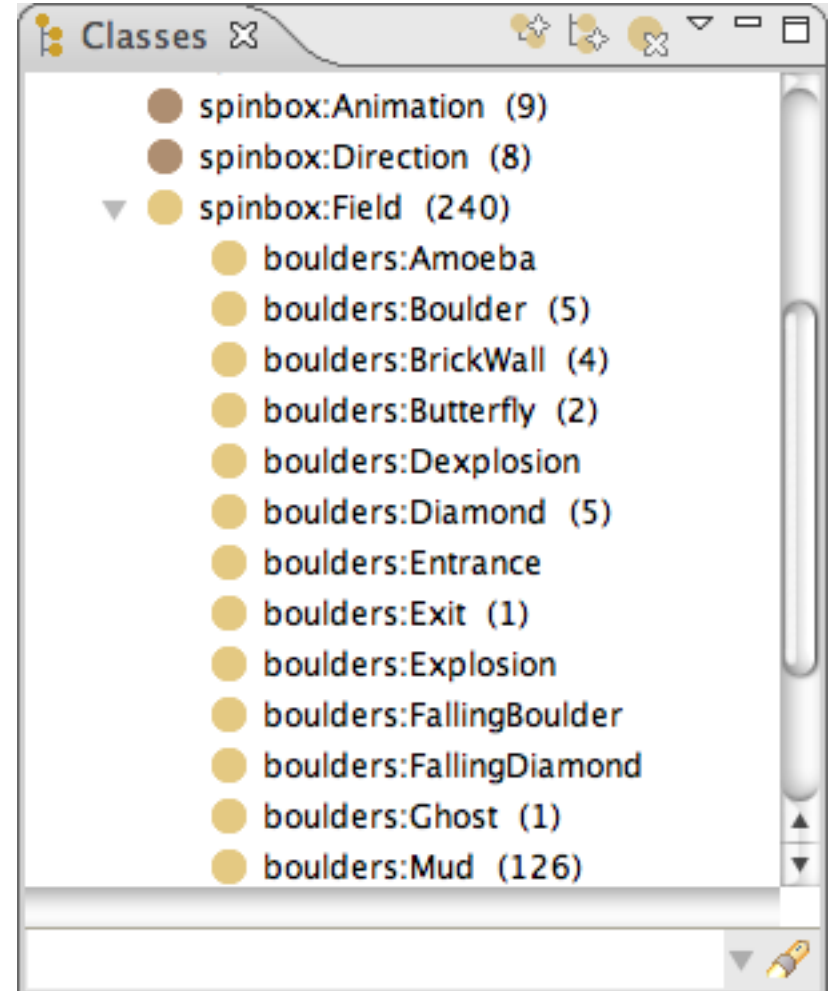
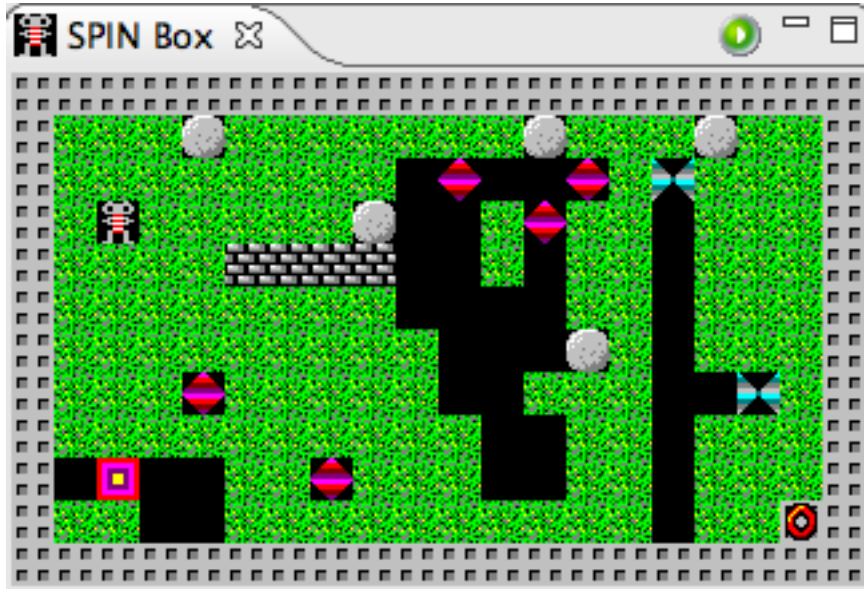
ex:image

http://upload.wikimedia.org/wikipedia/commons/thumb/9/9d/General_Sherman_Tree.jpg/180px-General_Sherman_Tree.jpg

Other Properties

- ex:heightCM: 8481
- ex:heightFeet: 278.24747
- ex:volumeFeet3: 52512.91
- ex:volumeM3: 1487

Example Application (Computer Game)



composing-the-semantic-web.blogspot.com

Summary

- > New development paradigm based on Linked Data
- > Smart code walks the data graph at run-time
- > Very little hard-coding of behavior
- > Benefits
 - flexible architecture
 - faster turn-around times
- > Challenges
 - learning curve (this is not OO!)
 - integration with established technologies



JavaOneSM

Thank You

Holger Knublauch
holger@topquadrant.com

<http://www.topquadrant.com>

Backup Slides

RDF Databases

- > AllegroGraph
- > Jena (RDB, SDB, TDB)
- > Mulgara
- > Oracle 11g
- > Sesame

Inferencing APIs

- > Often integrated into RDF stores:
 - Jena Rules
 - Sesame
 - Oracle 11g RDF
- > OWLIM (Rule engine with RDFS/OWL support)
 - <http://ontotext.com/owlim/>
- > Pellet (OWL DL)
 - <http://clarkparsia.com/pellet/>
- > TopSPIN (Rules via SPARQL)
 - <http://spinrdf.org/api>