Tecnológico
Fundación Deusto
Teknologikoa
Deustu Fundazioa

Universidad de Deusto
Deustuko Unibertsitatea

Lower-s Semantic Web

or the "Easier and More Usable Semantic Web"

Dr. Diego Lz. de Ipiña Gz. de Artaza
dipina@eside.deusto.es
http://paginaspersonales.deusto.es/dipina

http://www.tecnologico.deusto.es
http://www.smartlab.deusto.es
http://www.morelab.deusto.es

17 July 2007

Universidad de Deusto
Deustuko Unibertsitatea

Introduction

- Review of "upper-s" Semantic Web :
  - RDF and OWL
    - OWL reasoning
  - Jena Semantic Web toolkit:
    - Merging and querying: SPARQL
- Lower-s Semantic Web
  - Microformats, RDFa & GRDDL
- Web 2.0 and semantic web:
  - Semantic Mash-ups

Tecnológico
Fundación Deusto
Teknologikoa
Deustu Fundazioa

Web Semántica

- Problema de la Web Actual:
  - El significado de la web no es accesible a máquinas
- **Web Semántica** → crea un medio universal de intercambio de información, aportando semántica a los documentos en la web
  - Añade significado comprensible por ordenadores a la Web
  - Usa técnicas inteligentes que explotan esa semántica
  - Liderada por Tim Berners-Lee del W3C
- **Mission: turning existing web content into machine-readable content**

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Web vs. Web Semántica

- La Web permite acceder a todo tipo de información fácilmente
  - Los motores de búsqueda nos ayudan a encontrar información
    - Pero, los resultados devueltos no son siempre correctos
- Web Actual:
  - Colección de documentos ligados por hipervínculos
    - El texto de un enlace es una palabra clave que hace referencia a otros documentos
  - Útil para describir, con un énfasis en presentación visual, bloques de texto, imágenes y formularios
    - Pero, una máquina no puede extraer semántica de listado de productos en una página web

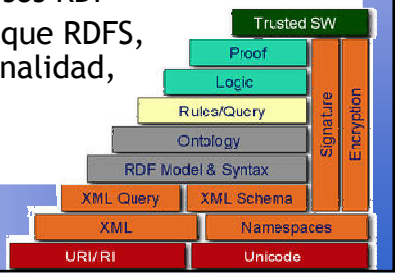
Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Web vs. Web Semántica

- La Web Semántica pretende crear un medio universal para intercambiar información y relacionar conceptos
- Web Semántica:
  - Conjunto de conceptos ligados a otros conceptos
    - RDF y OWL permiten indicar cómo un concepto se relaciona con otro
      - Añaden significado al contenido, facilitando el uso autónomo de la web por ordenadores

Semantic Web Stack

- La Web Semántica está compuesta de:
  - XML, sintaxis para documentos estructurados
  - XML Schema, restringe la estructura de documentos XML
  - RDF es un modelo de datos que hace referencia a objetos y sus relaciones
  - RDF Schema, vocabulario para definir propiedades y clases de recursos RDF
  - OWL, añade más vocabulario que RDFS, relaciones entre clases, cardinalidad, igualdad …



Propósito Web Semántica

- Mejorar la usabilidad y utilidad de la Web y sus recursos interconectados, mediante:
  - **Anotación semántica** → documentos mejorados con metadatos semánticos leíbles por máquinas o metadatos representando hechos sobre cualquier concepto (lugar, persona, etc.)
  - **Ontologías** → vocabularios de metadatos comunes y mapas entre ellos que guían marcado de documentos para que los agentes puedan utilizar la semántica suministrada
    - Autor de la página o autor del libro
  - **Agentes** → realizan tareas para usuarios utilizando estos metadatos (shopbot)
  - **Infrastructura** → Servicios Web que suministren información a agentes (Trust Service – informa calidad información)
- Los principales facilitadores de la Web Semántica son URIs, XML, XML NameSpaces y RDF

Universidad de Deusto
Deustuko Unibertsitatea

Tecnológico Fundación Deusto
Teknologikoa Deustu Fundazioa

RDF: Introduction

- RDF stands for Resource Description Framework
- RDF is a framework for describing resources on the web
- RDF provides a model for data, and a syntax so that independent parties can exchange and use it
- RDF is designed to be read and understood by computers
- RDF is not designed for being displayed to people
- RDF is written in XML
- Enables merging and querying functionality
- RDF is a part of the W3C's Semantic Web Activity
- RDF is a W3C Recommendation

Universidad de Deusto
Deustuko Unibertsitatea

Tecnológico Fundación Deusto
Teknologikoa Deustu Fundazioa

Designed to be read by computers

- RDF was designed to provide a common way to describe information so it can be read and understood by computer applications.
- RDF descriptions are not designed to be displayed on the web.
  - Well, look and see RDFa!!!

RDF Resource, Property, and Property Value

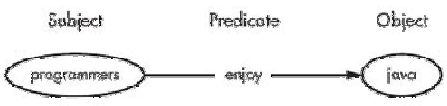
- RDF identifies things using Web identifiers (URIs), and describes resources with properties and property values.
- Explanation of Resource, Property, and Property value:
  - A **Resource** is anything that can have a URI, such as "http://www.w3schools.com/RDF"
  - A **Property** is a Resource that has a name, such as "author" or "homepage"
  - A **Property value** is the value of a Property, such as "Jan Egil Refsnes" or "http://www.w3schools.com" (note that a property value can be another resource)

Resource Description Framework (RDF)

- Modelo basado en la definición de sentencias acerca de recursos en formato:
  - Sujeto-predicado-objeto ⇔ RDF Triple
    - Sujeto: recurso descrito
    - Predicado: relación entre sujeto y objeto
    - Objeto: el valor asociado al sujeto

Resource Description Framework (RDF)

- Modelo para describir pseudo-grafos dirigidos etiquetados:
  - Dirigido → cada arco tiene una dirección
  - Etiquetado → cada arco tiene una etiqueta
  - Pseudo-grafo → puede haber más de un arco entre nodos
- Un modelo RDF es una colección no ordenada de sentencias o ternas, con:
  - Sujeto (nodo)
  - Predicado (arco)
  - Objeto (nodo)



RDF Statements

- The combination of a Resource, a Property, and a Property value forms a Statement
  - Known as the subject, predicate and object of a Statement.
- Some example statements:
  - "The author of http://www.w3schools.com/RDF is Jan Egil Refsnes".
    - The subject of the statement above is: http://www.w3schools.com/RDF
    - The predicate is: author
    - The object is: Jan Egil Refsnes
  - "The homepage of http://www.w3schools.com/RDF is http://www.w3schools.com".
    - The subject of the statement above is: http://www.w3schools.com/RDF
    - The predicate is: homepage
    - The object is: http://www.w3schools.com

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Resource Description Framework (RDF)

- Un grafo RDF crea una web de conceptos
  - Realiza aserciones sobre relaciones lógicas entre entidades
- Información en RDF puede ligarse con grafos en otros lugares
  - Mediante software se pueden realizar inferencias
    - Lenguajes de consulta sobre triple stores como SPARQL
- Mediante RDF hacemos que la información sea procesable por máquinas
  - Agentes software pueden guardar, intercambiar y utilizar metadatos sobre recursos en la web
- **Ontología** → jerarquía de términos a utilizar en etiquetado de recursos

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Conceptos Fundamentales RDF

- URIs → mecanismo utilizado por RDF para identificar unívocamente conceptos
- Literals → objetos con contenido real en vez de URIs
  - `this line --- (was written on) --> "20060603"[date]`
- Reification → uso de sentencias como sujeto de otras sentencias

```
this news ---(has category)---> "semantic web"
[this news ---(has category)---> "semantic web"]
  ---(added by)---> stefano
```


Formatos RDF: rdf+xml

- El mecanismo de serialización oficial de RDF es RDF/XML
  - Tipo MIME es `application/rdf+xml`
  - No es muy legible
- Ej. Expresión en RDF de "Artículo en Wikipedia sobre Maradona"

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-
  syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description
    rdf:about="http://en.wikipedia.org/wiki/Maradona">
     <dc:title>Diego Armando Maradona</dc:title>
     <dc:publisher>Wikipedia</dc:publisher>
  </rdf:Description>
</rdf:RDF>
```

Formatos RDF: N3

- N3 es una notación que permite definir ternas o relaciones "sujeto-predicado(verbo)-objeto", de una manera más concisa
- Ej1:

```
<http://en.wikipedia.org/wiki/Maradona>
  <http://purl.org/dc/elements/1.1/title> "Diego
  Armando Maradona" .
```

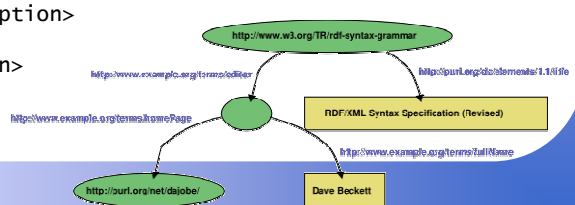
- Ej2:

```
@prefix wcj: http://example.org/wcjava/uri/ .

wcj:programmers wcj:enjoy wcj:java .
```


RDF Formats: Turtle

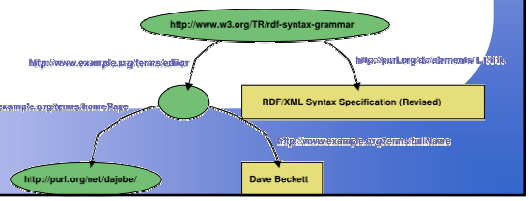
- Turtle: Terse RDF Triple Language
- Serialization format for RDF
- Its MIME type is application/x-turtle
- Example in RDF/XML:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dc="http://purl.org/dc/elements/1.1/"
         xmlns:ex="http://example.org/stuff/1.0/">
  <rdf:Description rdf:about="http://www.w3.org/TR/rdf-syntax-
  grammar"
       dc:title="RDF/XML Syntax Specification (Revised)">
    <ex:editor>
      <rdf:Description ex:fullName="Dave Beckett">
   <ex:homePage rdf:resource="http://purl.org/net/dajobe/" />
      </rdf:Description>
    </ex:editor>
  </rdf:Description>
</rdf:RDF>
```



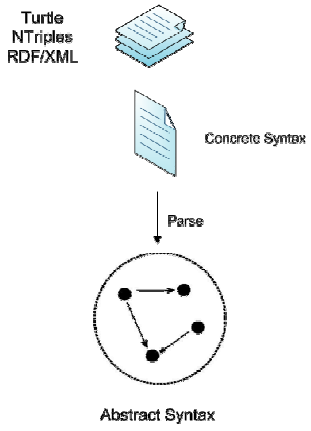
RDF Formats: Turtle

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-
  syntax-ns#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix ex: <http://example.org/stuff/1.0/> .

<http://www.w3.org/TR/rdf-syntax-grammar>
  dc:title "RDF/XML Syntax Specification
  (Revised)" ;
  ex:editor [
    ex:fullname "Dave Beckett";
    ex:homePage <http://purl.org/net/dajobe/>
  ] .
```



RDF formats

- All the different RDF representations are translated into Abstract Syntax:
  - RDF/XML
  - Turtle
  - NTriples
  - N3



RDF Example

Title	Artist	Country	Company	Price	Year
Empire Burlesque	Bob Dylan	USA	Columbia	10.90	1985
Hide your heart	Bonnie Tyler	UK	CBS Records	9.90	1988
...					

- The first line of the RDF document is the XML declaration. The XML declaration is followed by the root element of RDF documents: `<rdf:RDF>`.
- The `xmlns:rdf` namespace, specifies that elements with the rdf prefix are from the namespace "`http://www.w3.org/1999/02/22-rdf-syntax-ns#`".
- The `xmlns:cd` namespace, specifies that elements with the cd prefix are from the namespace "`http://www.recshop.fake/cd#`".
- The `<rdf:Description>` element contains the description of the resource identified by the `rdf:about` attribute.
- The elements: `<cd:artist>`, `<cd:country>`, `<cd:company>`, etc. are properties of the resource. Those elements are defined in a namespace outside RDF. RDF defines only the framework. The elements, `artist`, `country`, `company`, `price`, and `year`, must be defined by someone else.

RDF Example

```
<?xml version="1.0"?>

<rdf:RDF xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
    xmlns:cd="http://www.recshop.fake/cd#">

<rdf:Description rdf:about="http://www.recshop.fake/cd/Empire Burlesque">
  <cd:artist>Bob Dylan</cd:artist>
  <cd:country>USA</cd:country>
  <cd:company>Columbia</cd:company>
  <cd:price>10.90</cd:price>
  <cd:year>1985</cd:year>
</rdf:Description>

<rdf:Description rdf:about="http://www.recshop.fake/cd/Hide your heart">
  <cd:artist>Bonnie Tyler</cd:artist>
  <cd:country>UK</cd:country>
  <cd:company>CBS Records</cd:company>
  <cd:price>9.90</cd:price>
  <cd:year>1988</cd:year>
</rdf:Description>
.
.
.
</rdf:RDF>
```

RDF Online Validator

- W3C's RDF Validation Service is useful when learning RDF: http://www.w3.org/RDF/Validator/
- The online RDF Validator parses your RDF document, checks your syntax, and generates tabular and graphical views of your RDF document.
- Copy and paste the example below into W3C's RDF validator:

```
<?xml version="1.0"?><rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:si="http://www.recshop.fake/siteinfo#">
  <rdf:Description rdf:about="http://www.w3schools.com/RDF">
    <si:author>Jan Egil Refsnes</si:author>
    <si:homepage>http://www.w3schools.com</si:homepage>
  </rdf:Description>
</rdf:RDF>
```


RDF Containers

- Used to describe group of things. For example, to list the authors of a book or to list the members in a band.
- The following RDF elements are used to describe such groups: <Bag>, <Seq>, and <Alt>.
- Bag Element:
  - The <rdf:Bag> element is used to describe a list of values that is intended to be unordered.
  - The <rdf:Bag> element may contain duplicate values.
- Example

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.recshop.fake/cd#">
<rdf:Description
 rdf:about="http://www.recshop.fake/cd/Beatles">
 <cd:artist>
   <rdf:Bag>
     <rdf:li>John</rdf:li>
     <rdf:li>Paul</rdf:li>
     <rdf:li>George</rdf:li>
     <rdf:li>Ringo</rdf:li>
   </rdf:Bag>
 </cd:artist>
</rdf:Description>
</rdf:RDF>
```

RDF Containers

- The <rdf:Seq> Element
  - The <rdf:Seq> element is used to describe a list of values that is intended to be ordered (For example, in alphabetical order).
  - The <rdf:Seq> element may contain duplicate values.
- Example

```
<?xml version="1.0"?><rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.recshop.fake/cd#"> <rdf:Description
 rdf:about="http://www.recshop.fake/cd/Beatles">
 <cd:artist>
   <rdf:Seq>
     <rdf:li>George</rdf:li>
     <rdf:li>John</rdf:li>
     <rdf:li>Paul</rdf:li>
     <rdf:li>Ringo</rdf:li>
   </rdf:Seq>
 </cd:artist>
</rdf:Description></rdf:RDF>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

RDF Containers

- The <rdf:Alt> Element
  - The <rdf:Alt> element is used to describe a list of alternative values (the user can select only one of the values).
- Example

```
<?xml version="1.0"?><rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:cd="http://www.recshop.fake/cd#"> <rdf:Description
 rdf:about="http://www.recshop.fake/cd/Beatles">
 <cd:format>
   <rdf:Alt>
     <rdf:li>CD</rdf:li>
     <rdf:li>Record</rdf:li>
     <rdf:li>Tape</rdf:li>
   </rdf:Alt>
 </cd:format>
</rdf:Description></rdf:RDF>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

RDF Collections

- RDF collections are used to describe groups that contains ONLY the specified members.
  - You cannot close a container. A container says that the containing resources are members - it does not say that other members are not allowed.
  - RDF collections are used to describe group that contains ONLY the specified members.
  - A collection is described by the attribute `rdf:parseType="Collection"`.
- Example

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:cd="http://recshop.fake/cd#">
    <rdf:Description rdf:about="http://recshop.fake/cd/Beatles">
       <cd:artist rdf:parseType="Collection">
          <rdf:Description
   rdf:about="http://recshop.fake/cd/Beatles/George"/>
          <rdf:Description
   rdf:about="http://recshop.fake/cd/Beatles/John"/>
          <rdf:Description
   rdf:about="http://recshop.fake/cd/Beatles/Paul"/>
          <rdf:Description
   rdf:about="http://recshop.fake/cd/Beatles/Ringo"/>
       </cd:artist>
    </rdf:Description>
</rdf:RDF>
```


RDF Vocabularies

- The Dublin Core is a set of predefined properties for describing documents.

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
  ns#"
xmlns:dc= "http://purl.org/dc/elements/1.1/">
<rdf:Description rdf:about="http://www.w3schools.com">
  <dc:title>D-Lib Program</dc:title>
  <dc:description>W3Schools - Free
  tutorials</dc:description>
  <dc:publisher>Refsnes Data as</dc:publisher>
  <dc:date>1999-09-01</dc:date>
  <dc:type>Web Development</dc:type>
  <dc:format>text/html</dc:format>
  <dc:language>en</dc:language>
</rdf:Description>
</rdf:RDF>
```

RDFS

- Extensión semántica de RDF
  - Conjunto de sentencias que definen clases y propiedades
  - RDF Schema provides the framework to describe application-specific classes and properties
  - Classes in RDF Schema is much like classes in object oriented programming languages. This allows resources to be defined as instances of classes, and subclasses of classes.
- Sirve para decir cosas como:
  - Esta URI debería ser considerada como una clase (`rdfs:Class`) o propiedad (`rdfs:Property`)
  - Indicar si una etiqueta (`rfds:label`) o comentario (`rdfs:comment`) es leible por humanos
  - Esta URL está definida por (`rfds:DefinedBy`)
  - Esta clase es subclase de (`rdfs:subClassOf`)
  - Esta propiedad es subpropiedad de (`rfds:subPropertyOf`)
  - Esta propiedad conecta esta clase de sujetos (`rdfs:domain`) con esta clase de objetos (`rdfs:range`)

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

RDFS Example

- In the example below, the resource "horse" is a subclass of the class "animal".

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf= "http://www.w3.org/1999/02/22-rdf-syntax-
  ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xml:base=  "http://www.animals.fake/animals#">
   <rdf:Description rdf:ID="animal">
      <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-
  schema#Class"/>
   </rdf:Description>
   <rdf:Description rdf:ID="horse">
      <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-
  schema#Class"/>
     <rdfs:subClassOf rdf:resource="#animal"/>
   </rdf:Description>
</rdf:RDF>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

RDFS Example

- Since an RDFS class is an RDF resource we can abbreviate the example above by using `rdfs:Class` instead of `rdf:Description`, and drop the `rdf:type` information `<?xml version="1.0"?>`

```
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf= "http://www.w3.org/1999/02/22-rdf-syntax-
  ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xml:base=  "http://www.animals.fake/animals#">
   <rdfs:Class rdf:ID="animal" />
   <rdfs:Class rdf:ID="horse">
     <rdfs:subClassOf rdf:resource="#animal"/>
   </rdfs:Class>
</rdf:RDF>
```


What is an ontology?

- An ontology describes basic concepts in a domain and defines relations among them. Basic building blocks of ontology design include:
  - classes or concepts
  - properties of each concept describing various features and attributes of the concept (slots (sometimes called roles or properties))
  - restrictions on slots (facets (sometimes called role restrictions))
- An ontology together with a set of individual instances of classes constitutes a knowledge base.

Why create an ontology?

- An ontology provides a common vocabulary for researchers who need to share information in the domain. Some of the reasons to create an ontology are:
  - To share common understanding of the structure of information among people or software agents
  - To enable reuse of domain knowledge
  - To make domain assumptions explicit
  - To separate domain knowledge from operational knowledge
  - To analyze domain knowledge

Universidad de Deusto
Deustuko Unibertsitatea

Tecnológico Fundación Deusto
Teknologikoa Deustu Fundazioa

How do I create an ontology?

- There is no one correct methodology for developing ontologies, nor is there a single correct result. Developing an ontology is usually an iterative process.
- In practical terms, developing an ontology includes:
  - defining classes in the ontology
  - arranging the classes in a subclass-superclass hierarchy
  - defining slots and describing allowed values for these slots
  - filling in the values for slots for instances

Universidad de Deusto
Deustuko Unibertsitatea

Tecnológico Fundación Deusto
Teknologikoa Deustu Fundazioa

OWL

- Acrónimo de Web Ontology Language, derivado de DAML+OIL
- Extensión de vocabulario a RDF → añade más metadatos a los nodos, clases y propiedades para razonar sobre ellas
  - OWL es a RDF lo que XSL es a XML
- Algunos ejemplos:
  - Esta propiedad es transitiva (`owl:TransitiveProperty`)
  - Esta propiedad es simétrica (`owl:SymmetricProperty`)
  - Esta propiedad es inversa a esta otra (`owl:InverseOf`)
  - Equivalente a (`owl:equivalentProperty`)
  - Lo mismo que (`owl:sameAs`)
  - Esta propiedad puede aparecer sólo una vez (`owl:cardinality`)

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Ejemplo OWL Ontology Reasoning

- Supongamos el siguiente modelo RDF:

```
<http://www.betaversion.org/~stefano/> -(is author of)->
   http://www.betaversion.org/~stefano/linotype/
<http://www.apache.org/~stefano/> -(is author of)->
   http://www.apache.org/~stefano/agora/
<http://web.mit.edu/people/stefanom/> -(is author of)->
   <http://simile.mit.edu/gadget/>
```

- Aunque pertenecen al mismo autor, no están relacionadas entre ellas, con la ayuda de OWL podemos mapear estas URIs

```
<http://www.apache.org/~stefano/> -(owl:sameAs)->
   <http://www.betaversion.org/~stefano/>
<http://web.mit.edu/people/stefanom/> -(owl:sameAs)->
   <http://www.betaversion.org/~stefano/>
```

- Si mezclamos ambos modelos y ejecutamos un razonador podríamos responder a “dime todo lo que ha escrito “http://www.betaversion.org/~stefano”:

```
http://www.betaversion.org/~stefano/> -(is author of)->
   http://www.apache.org/~stefano/agora/
<http://www.betaversion.org/~stefano/> -(is author of)->
   <http://simile.mit.edu/gadget/>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

OWL Review

- An ontology differs from an XML schema in that it is a knowledge representation, not a message format
- One advantage of OWL ontologies will be the availability of tools that can reason about them
- The normative OWL exchange syntax is RDF/XML.
- OWL is a vocabulary extension of RDF
- Web ontologies are distributed
- Can be imported and augmented, creating derived ontologies.

OWL NameSpaces

```
<!DOCTYPE rdf:RDF [
    <!ENTITY vin  "http://www.w3.org/TR/2004/REC-owl-guide-
   20040210/wine#" >
    <!ENTITY food "http://www.w3.org/TR/2004/REC-owl-guide-
   20040210/food#" > ]>
<rdf:RDF
    xmlns     ="&vin;"
    xmlns:vin ="&vin;"
    xml:base  ="&vin;"
    xmlns:food="&food;"
    xmlns:owl ="http://www.w3.org/2002/07/owl#"
    xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:xsd ="http://www.w3.org/2001/XMLSchema#">
```

Universidad de Deusto
Deustuko Unibertsitatea
Tecnológico Fundación Deusto
Teknologikoa Deustu Fundazioa

Ontology Headers

```
<owl:Ontology rdf:about="">
  <rdfs:comment>An example OWL
  ontology</rdfs:comment>
  <owl:priorVersion
  rdf:resource="http://www.w3.org/TR/2003/PR-owl-
  guide-20031215/wine"/>
  <owl:imports
  rdf:resource="http://www.w3.org/TR/2004/REC-owl-
  guide-20040210/food"/>
  <rdfs:label>Wine Ontology</rdfs:label>
  ...
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Classes in OWL

- A class definition has two parts: a name introduction or reference and a list of restrictions.

```
<owl:Class rdf:ID="Wine">
  <rdfs:subClassOf rdf:resource="&food;PotableLiquid"/>
  <rdfs:label xml:lang="en">wine</rdfs:label>
  <rdfs:label xml:lang="fr">vin</rdfs:label>
  ...
</owl:Class>

<owl:Class rdf:ID="Pasta">
  <rdfs:subClassOf rdf:resource="#EdibleThing" />
  ...
</owl:Class>
```

- Individuals: describe members of classes

```
<Region rdf:ID="CentralCoastRegion" />
```

- Difference between class and individual:
  - A class is simply a name and collection of properties that describe a set of individuals. Individuals are the members of those sets.
  - Individuals should correspond to actual entities that can be grouped into these classes.
  - OWL Full allows the use of classes as instances and OWL DL does not

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

OWL Properties

- *Properties* let us assert general facts about the members of classes and specific facts about individuals.
- Two types:
  - datatype properties, relations between instances of classes and RDF literals and XML Schema datatypes

```
<owl:Class rdf:ID="VintageYear" />
<owl:DatatypeProperty rdf:ID="yearValue">
  <rdfs:domain rdf:resource="#VintageYear" />
  <rdfs:range  rdf:resource="&xsd;positiveInteger"/>
</owl:DatatypeProperty>
```

  - object properties, relations between instances of two classes.

```
<owl:ObjectProperty rdf:ID="madeFromGrape">
  <rdfs:domain rdf:resource="#Wine"/>
  <rdfs:range rdf:resource="#WineGrape"/>
</owl:ObjectProperty>
```

- Restrictions:

```
<owl:Class rdf:ID="Wine">
  <rdfs:subClassOf rdf:resource="&food;PotableLiquid"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#madeFromGrape"/>
      <owl:minCardinality
   rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  ...
</owl:Class>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Property Characteristics

- Provide a powerful mechanism for enhanced reasoning about a property
- **Transitive property:**

```
<owl:ObjectProperty rdf:ID="locatedIn">
  <rdf:type rdf:resource="&owl;TransitiveProperty" />
  <rdfs:domain rdf:resource="&owl;Thing" />
  <rdfs:range rdf:resource="#Region" />
</owl:ObjectProperty>
<Region rdf:ID="SantaCruzMountainsRegion">
  <locatedIn rdf:resource="#CaliforniaRegion" />
</Region>
<Region rdf:ID="CaliforniaRegion">
  <locatedIn rdf:resource="#USRegion" />
</Region>
```

- Because the `SantaCruzMountainsRegion` is `locatedIn` the `CaliforniaRegion`, then it must also be `locatedIn` the `USRegion`, since `locatedIn` is transitive.

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Property Characteristics

- **Symmetric property:**

```
<owl:ObjectProperty rdf:ID="adjacentRegion">
  <rdf:type rdf:resource="&owl;SymmetricProperty" />
  <rdfs:domain rdf:resource="#Region" />
  <rdfs:range rdf:resource="#Region" />
</owl:ObjectProperty>
<Region rdf:ID="MendocinoRegion">
  <locatedIn rdf:resource="#CaliforniaRegion" />
  <adjacentRegion rdf:resource="#SonomaRegion" />
</Region>
```

  - The `MendocinoRegion` is adjacent to the `SonomaRegion` and vice-versa. The `MendocinoRegion` is located in the `CaliforniaRegion` but not vice versa.

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Property Characteristics

- **functional property:**

```
<owl:Class rdf:ID="VintageYear" />
<owl:ObjectProperty rdf:ID="hasVintageYear">
  <rdf:type rdf:resource="&owl;FunctionalProperty" />
  <rdfs:domain rdf:resource="#Vintage" />
  <rdfs:range  rdf:resource="#VintageYear" />
</owl:ObjectProperty>
```

- **inverseOf property:**

```
<owl:ObjectProperty rdf:ID="hasMaker">
  <rdf:type rdf:resource="&owl;FunctionalProperty" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="producesWine">
  <owl:inverseOf rdf:resource="#hasMaker" />
</owl:ObjectProperty>
```

- **inverseFunctional property:**

```
<owl:ObjectProperty rdf:ID="hasMaker" />
<owl:ObjectProperty rdf:ID="producesWine">
  <rdf:type rdf:resource="&owl;InverseFunctionalProperty" />
  <owl:inverseOf rdf:resource="#hasMaker" />
</owl:ObjectProperty>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Property Restrictions

- Further constrain the range of a property in specific contexts in a variety of ways
- **allValuesFrom:**

```
<owl:Class rdf:ID="Wine">
  <rdfs:subClassOf rdf:resource="&food;PotableLiquid" />
  ...
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasMaker" />
      <owl:allValuesFrom rdf:resource="#Winery" />
    </owl:Restriction>
  </rdfs:subClassOf>
  ...
</owl:Class>
```

- **someValueFrom:**

```
<owl:Class rdf:ID="Wine">
  <rdfs:subClassOf rdf:resource="&food;PotableLiquid" />
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasMaker" />
      <owl:someValuesFrom rdf:resource="#Winery" />
    </owl:Restriction>
  </rdfs:subClassOf>
  ...
</owl:Class>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Property Restrictions

- **Cardinality:**

```
<owl:Class rdf:ID="Vintage">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasVintageYear"/>
      <owl:cardinality
   rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

- **hasValue:**

```
<owl:Class rdf:ID="Burgundy">
  ...
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasSugar" />
      <owl:hasValue rdf:resource="#Dry" />
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Ontology Mapping

- **equivalentClass**

```
<owl:Class rdf:ID="Wine">
<owl:equivalentClass rdf:resource="&vin;Wine"/>
</owl:Class>
```

- **sameAs**

```
<Wine rdf:ID="MikesFavoriteWine">
<owl:sameAs rdf:resource="#StGenevieveTexasWhite" />
</Wine>
```

- **differentFrom**

```
<WineSugar rdf:ID="Dry" />
<WineSugar rdf:ID="Sweet">
   <owl:differentFrom rdf:resource="#Dry"/>
</WineSugar>
<WineSugar rdf:ID="OffDry">
   <owl:differentFrom rdf:resource="#Dry"/>
   <owl:differentFrom rdf:resource="#Sweet"/>
</WineSugar>
```

- **AllDifferent**

```
<owl:AllDifferent>
  <owl:distinctMembers rdf:parseType="Collection">
    <vin:WineColor rdf:about="#Red" />
    <vin:WineColor rdf:about="#White" />
    <vin:WineColor rdf:about="#Rose" />
  </owl:distinctMembers>
</owl:AllDifferent>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Complex Classes

- Costructors to create so-called *class expressions*
  - OWL supports the basic set operations, namely union, intersection and complement.
    - `owl:unionOf`, `owl:intersectionOf`, and `owl:complementOf`
  - Classes can be *enumerated*.
    - Class extensions can be stated explicitly by means of the `oneOf` constructor.
  - And it is possible to assert that class extensions must be disjoint.

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Complex Classes Examples

- **owl:intersectionOf**

```
<owl:Class rdf:ID="WhiteWine">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Wine" />
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasColor" />
      <owl:hasValue rdf:resource="#White" />
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

- **owl:unionOf**

```
<owl:Class rdf:ID="Fruit">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#SweetFruit" />
    <owl:Class rdf:about="#NonSweetFruit" />
  </owl:unionOf>
</owl:Class>
```

- **owl:oneOf**

```
<owl:Class rdf:ID="WineColor">
  <rdfs:subClassOf rdf:resource="#WineDescriptor"/>
  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#White"/>
    <owl:Thing rdf:about="#Rose"/>
    <owl:Thing rdf:about="#Red"/>
  </owl:oneOf>
</owl:Class>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Complex Classes Examples

- **owl:disjointWith**

```
<owl:Class rdf:ID="SweetFruit">
  <rdfs:subClassOf rdf:resource="#EdibleThing" />
</owl:Class>

<owl:Class rdf:ID="NonSweetFruit">
  <rdfs:subClassOf rdf:resource="#EdibleThing" />
  <owl:disjointWith rdf:resource="#SweetFruit" />
</owl:Class>

<owl:Class rdf:ID="Fruit">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#SweetFruit" />
    <owl:Class rdf:about="#NonSweetFruit" />
  </owl:unionOf>
</owl:Class>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Lógica e Inferencia

- La lógica es la disciplina que estudia los principios de razonamiento
- Los razonadores automáticos deducen conclusiones a partir del conocimiento
- Aplicado a ontologías, puede:
  - Descubrir conocimiento ontológico implícito
  - Descubrir relaciones e inconsistencias inesperadas

Críticas a la Web Semántica

- Para funcionar requiere hacer referencia a un vocabulario gigante y centralizado
- Debido a reification, las relaciones entre conceptos pueden ser tan extensas que se incrementa la computabilidad
- Aconsejable utilizar OWL Lite (NO DL, Full) para no imponer demasiados requerimientos computacionales
- Crear documentos semánticos es difícil y muy poco popular

Herramientas Web Semántica

- Jena para Java y CWM para Python
- Permiten transformar entre sintaxis N3 y RDF/XML
- Realizan inferencias

Jena: a Framework for Semantic Web

- Jena Semantic Web Framework
  - http://jena.sourceforge.net/
- Enables among other things:
  - Create and populate RDF models
  - Persist models to a database
  - Query models programmatically with RDQL y SPARQL
  - Reasoning over ontologies
- Currently in versión 2.5.2
  - Download from: http://jena.sourceforge.net/downloads.html

Create a RDF Model

- The `ModelFactory` class enables the creation of models:
  - `ModelFactory.createDefaultModel()`, allows the creation of an in-memory model
    - Returns a `Model` instance over which you can create `Resources`
  - `Model.createProperty()` allow relationship creation
  - To add statements for a model use: `Resource.addProperty()` or `Model.createStatement()` and `Model.add()`
- In Jena a statement is compossed by:
  - A subject in the form of a `Resource`
  - A predicate represented by a `Property` class
  - An object, either a `Literal` or `Resource`
- `Resource`, `Property` and `Literal` inherit from `RDFNode`

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Create a RDF Model representing a Family

```
// URI declarations
String familyUri = "http://family/";
String relationshipUri = "http://purl.org/vocab/relationship/";

// Create an empty Model
Model model = ModelFactory.createDefaultModel();

// Create a Resource for each family member, identified by their URI
Resource adam = model.createResource(familyUri+"adam");
Resource beth = model.createResource(familyUri+"beth");
Resource chuck = model.createResource(familyUri+"chuck");
Resource dotty = model.createResource(familyUri+"dotty");
// and so on for other family members

// Create properties for the different types of relationship to represent
Property childOf = model.createProperty(relationshipUri,"childOf");
Property parentOf = model.createProperty(relationshipUri,"parentOf");
Property siblingOf = model.createProperty(relationshipUri,"siblingOf");
Property spouseOf = model.createProperty(relationshipUri,"spouseOf");

// Add properties to adam describing relationships to other family members
adam.addProperty(siblingOf,beth);
adam.addProperty(spouseOf,dotty);
adam.addProperty(parentOf,edward);

// Can also create statements directly . . .
Statement statement = model.createStatement(adam,parentOf,fran);
// but remember to add the created statement to the model
model.add(statement);
```

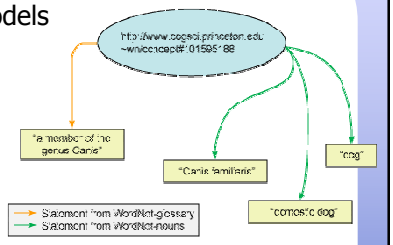
Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Interrogating an RDF Model

- By means of `listXXX()` method in `Model` and `Resource` interfaces
  - It returns specializations of `java.util.Iterator`
  - El método más genérico es `Model.listStatements(Resource s, Property p, RDFNode o)`
- Examples:
  - `ResIterator parents = model.listSubjectsWithProperty(parentOf);`
    - `Resource person = parents.nextResource();`
  - `NodeIterator moreParents = model.listObjectsOfProperty(childOf);`
  - `StmtIterator moreSiblings = edward.listProperties(siblingOf);`
  - `model.listStatements(adam,null,null);`

Importing and Persisting Models

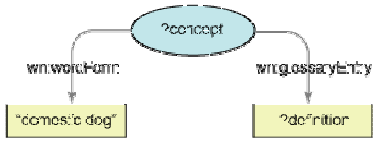
- So far we have worked with in-memory models
  - Necessary to persist and retrieve models
- Easiest solution:
  - `Model.read()`
  - `Model.write()`
- More sophisticated over RDBMS:
  - MySQL
- Example: `ImportWordnet.java`
  - Imports the following RDF models into a single Jena model:
    - WordNet-nouns
    - WordNet-glossary
    - WordNet-hyponyms



RDF Data Query Language (RDQL)

- RDQL is a query language for RDF
  - Allows complex queries to be expressed concisely
    - A query engine performing the hard work of accessing the data model
- Example:

```
SELECT ?definition
WHERE
   (?concept, <wn:wordForm>, "domestic dog"),
   (?concept, <wn:glossaryEntry>, ?definition)
USING
   wn FOR <http://www.cogsci.princeton.edu/~wn/schema/>
```



Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

RDF Data Query Language (RDQL)

- Another example:


```
SELECT
      ?wordform, ?definition

WHERE
      (?firstconcept, <wn:wordForm>, "panther"),
      (?secondconcept, <wn:wordForm>, "tiger"),

      (?firstconcept, <wn:hyponymOf>, ?hypernym),
      (?secondconcept, <wn:hyponymOf>, ?hypernym),

      (?hypernym, <wn:wordForm>, ?wordform),
      (?hypernym, <wn:glossaryEntry>, ?definition)

USING
      wn FOR <http://www.cogsci.princeton.edu/~wn/schema/>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Using RDQL

- Need to `import com.hp.hpl.jena.rdql`
- To create a query instantiate `Query` and pass as a `String` the query
- Create `QueryEngine` and invoke `QueryEngine.exec(Query)`
- Variables can be bound to values through a `ResultBinding` object
- Example:

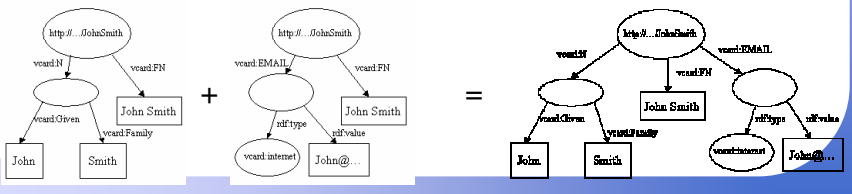
```
// Create a new query passing a String containing the RDQL to execute, containing
   variables x and y
Query query = new Query(queryString);

// Set the model to run the query against
query.setSource(model);

// A ResultBinding specifies mappings between query variables and values
ResultBindingImpl initialBinding = new ResultBindingImpl() ;

// Bind the query's first variable to a resource
Resource someResource = getSomeResource();
initialBinding.add("x", someResource);

// Bind the query's second variable to a literal value
RDFNode foo = model.createLiteral("bar");
initialBinding.add("y", foo);

// Use the query to create a query engine
QueryEngine qe = new QueryEngine(query);

// Use the query engine to execute the query
QueryResults results = qe.exec();
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Operations on Models

- The common set operations:
  - Union (`.union(Model)`), intersection (`.intersection(Model)`) and difference (`.difference(Model)`)
- Example: union

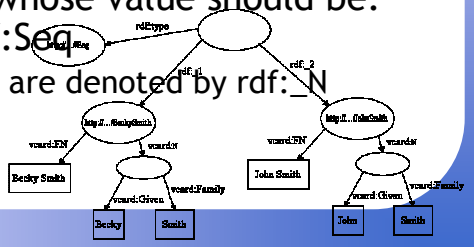
```
// read the RDF/XML files
model1.read(new InputStreamReader(in1), "");
model2.read(new InputStreamReader(in2), ""); +
// merge the Models
Model model = model1.union(model2);
// print the Model as RDF/XML
model.write(system.out, "RDF/XML-ABBREV");
```



Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

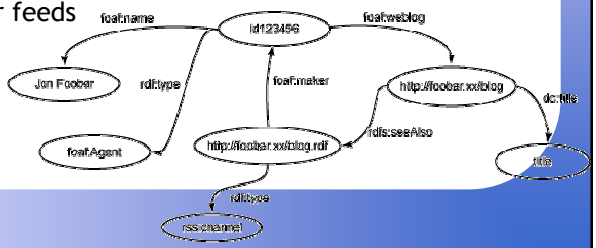
Containers in RDF

- RDF defines a special kind of resources to represent collections of things:
  - A BAG is an unordered collection
  - An ALT, unordered collection representing alternatives
  - A SEQ is an ordered collection
- A container is represented by a resource with an rdf:type property whose value should be: rdf:Bag, rdf:Alt or rdf:Seq
  - The ordinal properties are denoted by rdf:_N



SPARQL Protocol And RDF Query Language (SPARQL)

- SPARQL query over data-sets enlarged by the use of OWL reasoning
  - Enables automatically "mash-up" of data from multiple sources.
- Builds on previously existing query languages such as rdfDB, RDQL, SeRQL
- In our experimentation with SPARQL 3 RDF graphs will be used, corresponding to blogger.rdf:
  - FOAF graphs describing contributors
  - RSS 1.0 contributor feeds
  - Bloggers graph



Syntax SPARQL

- SPARQL query to find the URL of a contributor's blog (`david.rq`):

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?url
FROM      <bloggers.rdf>
WHERE  {
    ?contributor foaf:name "Dave Beckett" .
    ?contributor foaf:weblog ?url .
}
```

  - `PREFIX` indicates prefix for FOAF namespace
  - `SELECT` indicates what the query should return
  - `FROM` optional clause indicating the URI of the dataset to use
  - `WHERE` triple patterns expressed in Turtle syntax (graph pattern)
- Test queries from command line with `java jena.sparql`
  - `java jena.sparql --query david.rq`
  - `FROM` clause can be omitted and specified by `--data URL`

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Using SPARQL with JENA

- SPARQL is supported in JENA via ARQ module
  - It also understands RDQL queries
- Need to import package: `com.hp.hpl.jena.query`
- `QueryFactory.create()` returns a Query object from a file or String
- `QueryExecutionFactory.create(query, model)` returns a `QueryExecution` object
- `QueryExecution` supports varios methods:
  - `execSelect()` returns a `ResultSet`
  - Apart from `SELECT`, you can apply the following types of queries:
    - `ASK`, `DESCRIBE`, `CONSTRUCT`

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Executing a Simple Query with JENA

```
// Open the bloggers RDF graph from the filesystem
InputStream in = new FileInputStream(new File("bloggers.rdf"));

// Create an empty in-memory model and populate it from the graph
Model model = ModelFactory.createMemModelMaker().createModel();
model.read(in,null); // null base URI, since model URIs are absolute
in.close();

// Create a new query
String queryString =
    "PREFIX foaf: <http://xmlns.com/foaf/0.1/> " +
    "SELECT ?url " +
    "WHERE {" +
    "      ?contributor foaf:name \"Jon Foobar\" . " +
    "      ?contributor foaf:weblog ?url . " +
    "      }";

Query query = QueryFactory.create(queryString);

// Execute the query and obtain results
QueryExecution qe = QueryExecutionFactory.create(query, model);
ResultSet results = qe.execSelect();

// Output query results
ResultSetFormatter.out(System.out, results, query);

// Important - free up resources used running the query
qe.close();
```


Refining SPARQL Queries

- DISTINCT used with SELECT

  `SELECT DISTINCT`
- Used with SELECT clause:
  - `LIMIT n` → shows upto n results
  - `OFFSET n` → ignores first n results
  - `ORDER BY var` → sorts results by normal ordering
    - `ASC(?var)` and `DESC(?var)`

More complex queries

- RDF is often used to represent *semi-structured* data. This means that two nodes of the same type in a model may have different sets of properties.
- Optional matches

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?depiction
WHERE {
   ?person foaf:name ?name .
   OPTIONAL {
      ?person foaf:depiction ?depiction .
   } .
}
```

- Alternative matches:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?name ?mbox
WHERE {
  ?person foaf:name ?name .
  {
     { ?person foaf:mbox ?mbox } UNION { ?person
foaf:mbox_sha1sum ?mbox }
  }
}
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

More complex queries

- Using filters

```
PREFIX rss:   <http://purl.org/rss/1.0/>
PREFIX xsd:   <http://www.w3.org/2001/XMLSchema#>
PREFIX  dc:   <http://purl.org/dc/elements/1.1/>
SELECT ?item_title ?pub_date
WHERE {
   ?item rss:title ?item_title .
   ?item dc:date ?pub_date .
   FILTER xsd:dateTime(?pub_date) >= "2005-04-
01T00:00:00Z"^^xsd:dateTime &&
         xsd:dateTime(?pub_date) < "2005-05-
01T00:00:00Z"^^xsd:dateTime
}
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Working with Multiple Graphs

- The model after the FROM clause is the background graph
- Several graphs can be specified after the FROM NAMED <URI> clause
- Named graphs are used within a SPARQL query with the GRAPH keyword
- Example: find people found in two named FOAF graphs

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?name
FROM NAMED <jon-foaf.rdf>
FROM NAMED <liz-foaf.rdf>
WHERE {
  GRAPH <jon-foaf.rdf> {
    ?x rdf:type foaf:Person .
    ?x foaf:name ?name .
  } .
  GRAPH <liz-foaf.rdf> {
    ?y rdf:type foaf:Person .
    ?y foaf:name ?name .
  } .
}
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Working with Multiple Graphs

- Example: determining which graph describes different people

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-
  syntax-ns#>
SELECT ?name ?graph_uri
FROM NAMED <jon-foaf.rdf>
FROM NAMED <liz-foaf.rdf>
WHERE {
    GRAPH ?graph_uri {
      ?x rdf:type foaf:Person .
      ?x foaf:name ?name .
    }
}
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Combining Background Data and Named Graphs

- Example: getting a personalized live PlanetRDF feed

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX rss:  <http://purl.org/rss/1.0/>
PREFIX dc:   <http://purl.org/dc/elements/1.1/>
SELECT ?title ?known_name ?link
FROM <http://planetrdf.com/index.rdf>
FROM NAMED <phil-foaf.rdf>
WHERE {
   GRAPH <phil-foaf.rdf> {
     ?me foaf:name "Phil McCarthy" .
     ?me foaf:knows ?known_person .
     ?known_person foaf:name ?known_name .
   } .

   ?item dc:creator ?known_name .
   ?item rss:title ?title .
   ?item rss:link ?link .
   ?item dc:date ?date.
}
ORDER BY DESC(?date) LIMIT 10
```


Ontologies in Jena

- They are treated a special type of RDF model, `OntModel`
  - This interface allows to manipulate programmatically an ontology:
    - Create classes, property restrictions
- Alternatively:
  - Statements meaning semantic restrictions can be added to an RDF model
  - Merge an ontology model with a data model with `Model.union()`
- Examples:

```
// Make a new model to act as an OWL ontology for WordNet
OntModel wnOntology = ModelFactory.createOntologyModel();

// Use OntModel's convenience method to describe
// WordNet's hyponymOf property as transitive
wnOntology.createTransitiveProperty(WordnetVocab.hyponymOf.getURI());

// Alternatively, just add a statement to the underlying model to
   express that hyponymOf is of type TransitiveProperty
wnOntology.add(WordnetVocab.hyponymOf, RDF.type,
   OWL.TransitiveProperty);
```

Inference in Jena

- Given an ontology and a model Jena can inference statements not explicitly expressed
- `OWLReasoner` applies OWL ontologies over a model to reason
- Example:

```
// Make a new model to act as an OWL ontology for WordNet
OntModel wnOntology = ModelFactory.createOntologyModel();
...
// Get a reference to the WordNet plants model
ModelMaker maker = ModelFactory.createModelRDBMaker(connection);
Model model = maker.openModel("wordnet-plants",true);

// Create an OWL reasoner
Reasoner owlReasoner = ReasonerRegistry.getOWLReasoner();

// Bind the reasoner to the WordNet ontology model
Reasoner wnReasoner = owlReasoner.bindSchema(wnOntology);

// Use the reasoner to create an inference model
InfModel infModel = ModelFactory.createInfModel(wnReasoner, model);

// Set the inference model as the source of the query
query.setSource(infModel);

// Execute the query as normal
QueryEngine qe = new QueryEngine(query);
QueryResults results = qe.exec(initialBinding);
```

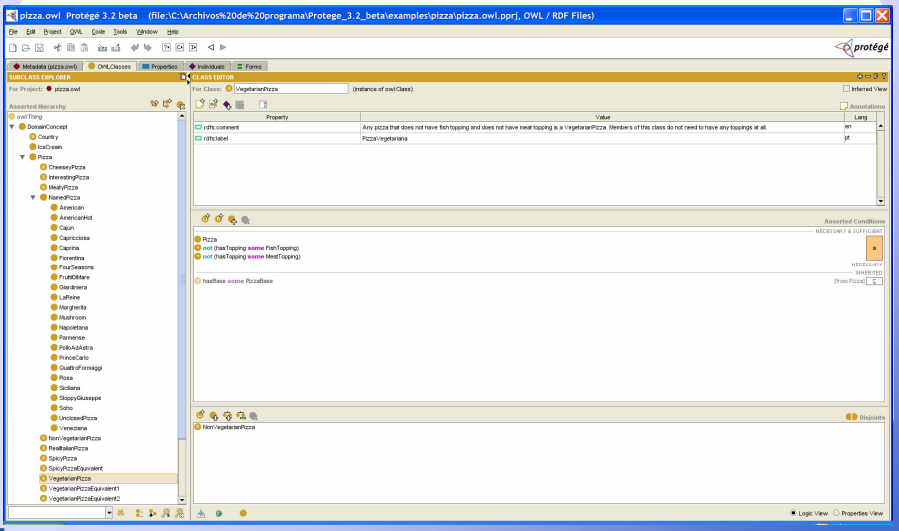

Jena Generic Rule Engine

- JENA Rule Engine supports rule-based inference over RDF graphs using:
  - Forward-chaining
  - Backward-chaining
  - Hybrid execution engine
- Implemented as class: `com.hp.hpl.jena.reasoner.rulesys.GenericRuleReasoner`
  - Requieres a `RuleSet` to define its behaviour
    - A set of `com.hp.hpl.jena.reasoner.rulesys.Rule`

Generating Ontologies with Protégé

- Protége is a free, open source ontology editor and knowledge base framework.
  - Implements a rich set of knowledge-modeling structures and actions that support the creation, visualization, and manipulation of ontologies in various representation formats.
    - An ontology describes the concepts and relationships that are important in a particular domain, providing a vocabulary for that domain as well as a computerized specification of the meaning of terms used in the vocabulary.
  - Supports two ways of modeling ontologies:
    - Protégé-Frames and Protégé-OWL
  - Downloadable from: http://protege.stanford.edu/
- W3C Ontology Definition:
  - *"An OWL ontology may include descriptions of classes, properties and their instances. Given such an ontology, the OWL formal semantics specifies how to derive its logical consequences, i.e. facts not literally present in the ontology, but entailed by the semantics. These entailments may be based on a single document or multiple distributed documents that have been combined using defined OWL mechanisms"*
- The Protégé OWL Tutorial:
  - http://www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf

Pizza Ontology in Protégé



Universidad de Deusto
Deustuko Unibertsitatea
Tecnológico Fundación Deusto
Teknologikoa Deustu Fundazioa

Microformats

- Microformats (sometimes abbreviated µF or uF) add semantics to markup to take it from being machine readable to being machine understandable.
  - Add simple semantic meaning to human-readable content which is otherwise, from a machine's point of view, just plain text.
  - Aimed to change the way we interact on the web
- Microformats allow data items (events, contact details or locations), on HTML (or XHTML) web pages, to be meaningfully detected and the information in them to be extracted by software, and indexed, searched for, saved or cross-referenced, so that it can be reused or combined.
- Example:

```
<span class="geo"><span class="latitude">52.48</span>,
    <span class="longitude">-1.89</span></span>
```

- Firefox 3 and IE8 will provide built-in support for Microformats

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

More about Microformats

- Emerged a couple of years ago as a way to tunnel richer semantics into host formats such as HTML/XHTML and Atom.
  - HTML has no direct way to express contact information, but a microformat called `hCard` allows you to augment HTML `div`s and `span`s to specify contact name, street address, postal code, and so on
- Aimed at a specific problem
  - A highly vertical approach.

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Advantages of Reusing XHTML in Microformats

- XHTML has been thoroughly debated, designed, and tested, and mostly avoids pitfalls that a fresh language might step in.
- Fragments of the microformat can be placed directly in XHTML web pages.
- Existing tools Just Work.
- Well-known element names, augmented with judicious `class` and `id` attributes.
- Additional functionality can be gracefully added within the existing semantics.

Careful with Microformats

- *Nuance microformats* or elemental microformats
  - A microformat such as rel-license provides a convention for use of an HTML attribute designed to carry such conventions.

```
<a href="http://creativecommons.org/licenses/by/2.0/"
  rel="license">cc by 2.0</a>
```

- Nuisance or compound microformats
  - XOXO seems a step backward to use a far less expressive construct just to embed the structure within HTML
    - XML was designed for that
- The idea is that the author marks up entries with hAtom, a blogroll with XOXO, the license with rel-license, inline contact information with hCard, and so on.
  - Microformats provide a simple way to publish all this data in somewhat structured form just by updating the Weblog engine templates.
  - Problems come about when the authors of microformats start to distort host formats rather than just build on them.

Most used Microformats

- XFN (http://gmpg.org/xfn/) for human relationships (using the `rel` attribute; `rel` is also a way to extend Creative Commons metadata)

```
<a href="http://jane-blog.example.org/" rel="sweetheart date met">Jane</a>
```

- Geo (http://microformats.org/wiki/geo) for location (using simple `<meta>` tags)

```
<div class="geo">GEO: <span class="latitude">37.386013</span>, <span class="longitude">-
    122.082932</span> </div>
```

- hCalendar (http://microformats.org/wiki/hcalendar) for calendar events (mapping the commonly used `iCalendar` format to XHTML).

```
<div class="vevent">
   <a class="url" href="http://www.web2con.com/">http://www.web2con.com/</a>
   <span class="summary">Web 2.0 Conference</span>:
   <abbr class="dtstart" title="2007-10-05">October 5</abbr>-
   <abbr class="dtend" title="2007-10-20">19</abbr>,
 at the <span class="location">Argent Hotel, San Francisco, CA</span>
</div>
```

- XOXO (http://microformats.org/wiki/xoxo) for outlines and blogroll-like subscriptions.
- hCard (http://microformats.org/wiki/hcard) for address books (mapping the commonly used `vCard` format to XHTML).
- Others:
  - Reviews (http://microformats.org/wiki/hreview)
  - Resumes (http://microformats.org/wiki/hresume)
  - Or even very domain specific like wines - hwine

hCard Microformat Example

- Consider the following HTML:

```
<div>
   <div>Joe Doe</div>
   <div>The Example Company</div>
   <div>604-555-1234</div>
   <a
  href="http://example.com/">http://example.com/</a>
</div>
```

- With microformat markup, that becomes:

```
<div class="vcard">
   <div class="fn">Joe Doe</div>
   <div class="org">The Example Company</div>
   <div class="tel">604-555-1234</div>
   <a class="url"
  href="http://example.com/">http://example.com/</a>
</div>
```

Microformat Wizards

- hCard Creator:
  - http://tantek.com/microformats/hcard-creator.html

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Microformats will change Web interaction

- **Aggregation sites**
  - blogging + microformats are a good combination
    - if you want to sell something, blog it with hlisting microformat (http://microformats.org/wiki/hlisting) and a site like http://www.edgeio.com/ will find it when it aggregates classified advertisements across the Web
- **Sharing Information with a Specific Community**
  - *geocast* (RSS with a payload of geo), the locations of the mountain bike trails, and other people will subscribe to the feed using Google Earth
- **Targeted Search**
  - By posting your comic with a microformat agreed upon by the web comic community, the rest of the community will be able to easily find your work using a search engine.
- **The Web Browser as an Information Broker**
  - Future Web browsers will associate semantically marked up data on the Web with specific applications
  - Examples:
    - Contact information on Web site will be associated with contacts application
    - Events will be associated with your favourite calendar application
    - Locations will be associated with your favourite mapping application
    - Phone numbers will be associated with your favorite VOIP application

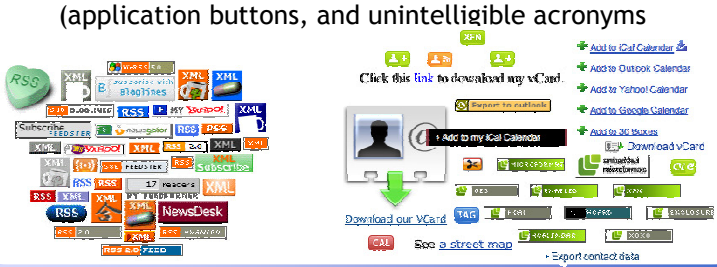
Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Websites using Microformats

- Marking up their content using microformats as a means of making the site itself an API.
  - Flickr, which lets users geotag photos
  - Yahoo! Local, which encodes each search result with an hCard
  - Upcoming.org, with encodes events with hCalendar
  - Revyu.com – allows you to review and rate things
  - http://eventful.com/
  - http://www.linkedin.com/in/harryhalpin
  - http://yedda.com/
  - http://local.yahoo.com

How to identify where Microformats are available?

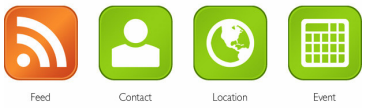
- We will move from the RSS icon to an icon for every popular type of structured data on the Web.
  - Web browsers must provide the user with a clean, consistent, and simplistic user interface.
    - the user experience is now a mixture of specific (application buttons, and unintelligible acronyms



Main Microformats

- Firefox 3 will use Microformats
  - http://wiki.mozilla.org/Firefox/Feature_Brainstorming:Microformat_Handling
- Internet Explorer 8 will include support for microformats
  - http://factoryjoe.com/blog/2006/10/29/internet-explorer-80-will-support-microformats/

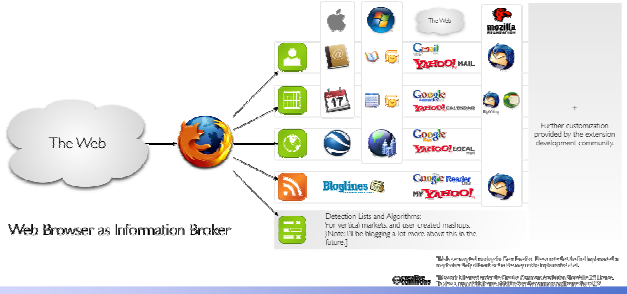
Interrogative Word	Action	Tool	Microformat
Who?	Meet a person	Contacts book	hCard
Where?	Find a location	Map	adr, geo
When?	Schedule	Calendar	hCalendar
What? How? Why?	Domain Specific		



Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

The Browser as an Information Broker

- New role for Web Browsers:
  - **Detecting information in Web pages and handing that information off to other applications**
    - From HTML renderer to being an *information broker*.
- Microformat detection should be designed as a completely open and extensible platform.
  - Contact management, calendaring, and mapping applications use browser API to integrate with browser's microformat detection system.

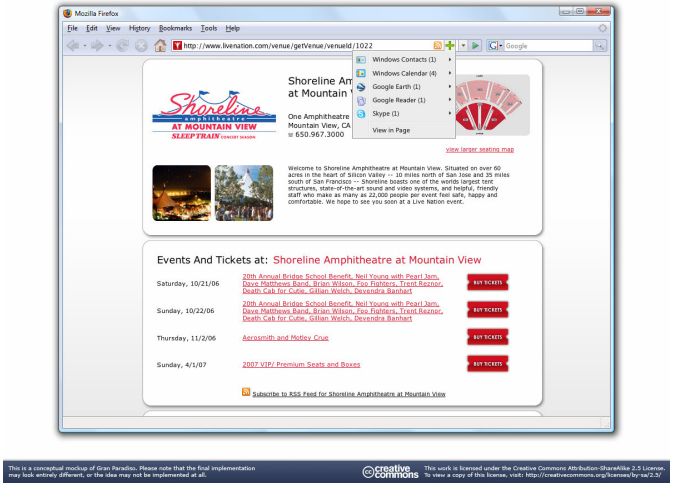


Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Operator Firefox Add-on

- Operator leverages microformats already available on many web pages to provide new ways to interact with web services.
  - Example: Flickr + GoogleMaps
    - http://www.flickr.com/photos/dbaron/299291151/
  - Example:
    - http://en.wikipedia.org/wiki/HCard + Outlook

Firefox 3: The Web Information Broker



RDFa

- Problems with RDF:
  - RDF is a flexible format for storing, aggregating, and using metadata.
  - BUT, RDF/XML syntax is messy enough to scare many people away from RDF
    - SOLUTION → RDFa
- RDFa = Making easier to embed it into XHTML and XML
  - Adds metadata to XHTML without affecting browsers display
  - Web page data is readable not only by humans but by automated processes
  - Enables data aggregation and metadata association to perform tasks more sophisticated than those enabled by screen scrapping
  - Uses some existing XHTML 1 attributes and a few new XHTML 2 attributes to store the subjects, predicates, and objects of RDF triples
    - XHTML 1 attributes `href`, `content`, `rel`, `rev`, and `datatype`
    - New `about`, `role` and `property` attributes from XHTML 2's Metainformation Attributes (http://www.w3.org/TR/xhtml2/mod-meta.html) module
- Good site: http://rdfa.info/

RDFa

- In RDF there are two basic cases:
  - Triples that have a literal string as their object and
  - Triples that have a URI as their object
    - Better since the same object can be used in many triples
- Example:

```
<span
  about="http://www.snee.com/bobdc.blog/2006/12/generating_a_single_globally_u.html" property="dc:title"
  content="Generating a Single Globally Unique ID"/>
```

  - The resource at *http://www.snee.com/bobdc.blog/2006/12/generating_a_single_globally_u.html* has a Dublin Core title value of "Generating a Single Globally Unique ID."
    - Useful whenever what the user sees and the machine sees is different

	subject	predicate	object
literal string as object	`about`	`property`	`content` attribute or PCDATA
URI as object	`about`	`rel`	`href`

More RDFa Basic Examples

- Examples:

```
<span
  about="http://www.snee.com/bobdc.blog/2006/12/generating_a_single_globally_u.html"
  property="dc:title">Generating a Single Globally Unique ID</meta>
```

  - The resource at *http://www.snee.com/bobdc.blog/2006/12/generating_a_single_globally_u.html* has a Dublin Core title value of "Generating a Single Globally Unique ID."
    - The most general and preferred case

```
<span
  about="http://www.snee.com/bobdc.blog/2006/12/generating_a_single_globally_u.html" rel="dc:subject"
  href="http://www.snee.com/bobdc.blog/neat_tricks/"/>
```

  - The resource at *http://www.snee.com/bobdc.blog/2006/12/generating_a_single_globally_u.html* has a Dublin Core subject value of http://www.snee.com/bobdc.blog/neat_tricks/.
    - A URI as an object value

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

RDFa Elements

- RDFa-aware software can pull useful metadata from span, `link` and `meta` elements with only minor modifications to these elements.
  - `span` elements are most popular for RDFa because you can insert them anywhere in the body of an HTML document, you can add the same RDFa attributes to any elements you like.
  - `link` and `metadata` are useful to insert RDFa in the `head` of an HTML document
  - The `a` linking element is also popular for storing RDFa metadata
    - It expresses a relationship between one resource (the document where it's stored) and another (the resource it links to).
      - `rel` attribute indicates the relationship in the triple

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

More Triples, Fewer Subjects

- RDFa lets you build multiple triples from the same subject without cluttering up your document too much
  - Children elements inherit the about attribute of a parent element (resource *http://www.snee.com/img/myfile.jpg*)
    - If no about attribute is found then it is the current document:

```
<img src="http://www.snee.com/img/myfile.jpg"
  about="http://www.snee.com/img/myfile.jpg"> <span
  property="dc:subject" content="Niagra Falls"/>
  <span property="dc:creator" content="Richard
  Mutt"/>
  <span property="dc:format" content="img/jpeg"/>
</img>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

3 Categories of RDFa Use Cases

- Inline metadata about document components

```
<p>This photo was taken by <span class="author"
  about="photo1.jpg" property="dc:creator">Mark
  Birbeck</span>.</p>
```

  - The span element and its attribute values let an RDFa-aware tool get the following triple out of this document, shown here in RDF/XML:

```
<rdf:Description
  rdf:about="file://C|/dat/xml/rdf/rdfa/photo1.jpg">
  <dc:creator>Mark Birbeck</dc:creator> </rdf:Description>
```

```
<p>Last revision of document: <span
  about="http://www.snee.com/docs/mydoc1.html"
  property="dc:date" content="20070315T15:32:00">March
  15, 2007, at 3:32 PM</span></p>
```

  - The resulting triple uses the content value of the date:

```
<rdf:Description
  rdf:about="http://www.snee.com/docs/mydoc1.html">
  <dc:date>20070315T15:32:00</dc:date> </rdf:Description>
```

- Metadata about the containing document
- Out-of-line metadata

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

3 Categories of RDFa Use Cases

- Inline metadata about document components
- **Metadata about the containing document**
  - Metadata about the document with no displayable content can be stored in the head element of the document

```
<html xmlns:fm="http://www.foomagazine.com/ns/prod/">
  <head>
    <title>Is Black the New Black?</title>
    <meta property="fm:newsstandDate" content="2006-04-03"/>
    <meta property="fm:copyEditor" content="RSelavy"/>
    <meta property="fm:copyEdited" content="2006-03-
28T10:33:00"/>
  </head>
  <body>
```

  - An RDFa extractor gets the following RDF/XML out of this:

```
<rdf:Description rdf:about="">
  <fm:newsstandDate>2006-04-03</fm:newsstandDate>
  <fm:copyEditor>RSelavy</fm:copyEditor>
  <fm:copyEdited>2006-03-28T10:33:00</fm:copyEdited>
</rdf:Description>
```

- Out-of-line metadata

3 Categories of RDFa Use Cases

- Inline metadata about document components
- Metadata about the containing document
- **Out-of-line metadata**

```
<html xmlns:fm="http://www.foomagazine.com/ns/prod/">
<head><meta about="#recipe13941"><meta
   property="fm:ComponentID">XZ3214</meta><meta
   property="fm:ComponentType">Recipe</meta><meta
   property="fm:RecipeID">r003423</meta></meta>
</head>
<body>
  <h>Add Some Tex Mex Sizzle to Your Kid's Lunch</h>
  <section id="recipe22143">
    <h>Amigo Corn Dogs</h>
  </section>
  <section id="recipe13941">
    <h>EZ Bean Tacos</h>
  </section>
</body>
</html>
```

  - The resulting triple uses the `content` value of the date:

```
<rdf:Description
   rdf:about="file://C|/dat/xml/rdf/rdfa/test5.html#recipe13941">
  <fm:ComponentType>Recipe</fm:ComponentType>
  <fm:RecipeID>r003423</fm:RecipeID>
  <fm:ComponentID>XZ3214</fm:ComponentID>
</rdf:Description>
```

Software to Pull RDFa from XHTML

- Usage pattern of RDFa:
  - Extract embedded RDFa triples
  - Load them into a triplestore in memory or on disk
  - Develop an application around your extracted data
- RDFa2RDFXML.xsl XSL StyleSheet
  - http://www-sop.inria.fr/acacia/soft/RDFa2RDFXML.xsl
- Web Service which receives an XHTML and returns RDFa
  - Use http://ben.adida.net/card to test service at http://torrez.us/services/rdfa/
    - If you want to use it in your semantic web application, simply point your SPARQL query to: http://torrez.us/services/rdfa/[URL of HTML page]

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Data Typing in RDFa

- Example HTML:

```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:fb="http://www.foobarco.com/ns/vocab#"
      xmlns:fbi="http://www.foobarco.com/ns/ID#"
      xmlns:xs="http://www.w3.org/2001/XMLSchema#">
<!-- head element, start of body and table...  -->
      <tr about="[fbi:x432]" >
        <td><span property="fb:shipmentID">x432</span></td>
        <td><span property="fb:date"
                  datatype="xs:date">2007-04-23</span></td>
        <td><span property="fb:amount"
                  datatype="xs:integer">34</span></td>
        <td><span property="fb:anodized"
                  datatype="xs:boolean"
   content="true">yes</span></td>
      </tr>
<!-- remaining rows of table... -->
```

Shipment ID	Date	Amount	Anodized
x432	2007-04-23	34	Yes
x921	2007-04-25	41	No
x0731	2007-04-28	17	No

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

The rev Attribute

- RDFa uses the `rel` attribute as the predicate of a triple, with the `about` attribute naming the subject and the `href` attribute naming the object
- The `rev` attribute expresses a triple in which the `href` attribute names the subject and the `about` attribute of the element
- Example:

```
<span
  about="http://caselaw.lp.findlaw.com/scripts/ge
  tcase.pl?court=US&vol=347&invol=483"
  rel="fb:overturns" rev="fb:overturnedBy"
  href="http://caselaw.lp.findlaw.com/cgi-
  bin/getcase.pl?court=us&vol=163&invol=537"/>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Reification in RDFa

- Reification is the assignment of metadata to metadata.
  - "this document was created by Richard Mutt"
    - another triple saying that the triple about the document's creator was created on 2007-04-19 would be metadata about that metadata
- Example:

```
<p>Mr. Breakfast has a nice
  <a about="link23“
  href="http://www.mrbreakfast.com/article.asp?ar
  ticleid=17">
<span property="fb:addedBy" content="BD"/>
<span property="fb:lastChecked" content="2007-03-
  15"/>
scrambled eggs recipe</a>.</p>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

The class Attribute

- XHTML fragment:

```
<tr about="[fbi:x432]" class="fb:widgetShipment">
  <td><span property="fb:shipmentID">x432</span></td>
  <td><span property="fb:date" datatype="xs:date">2007-04-
  23</span></td>
  <td><span property="fb:amount"
  datatype="xs:integer">34</span></td>
</tr>
```

- RDF/XML extracted:

```
<fb:widgetShipment
   rdf:about="http://www.foobarco.com/ns/ID#x432">
  <fb:anodized
  rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">tru
  e</fb:anodized>
  <fb:amount
  rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">34<
  /fb:amount>
  <fb:date
  rdf:datatype="http://www.w3.org/2001/XMLSchema#date">2007-
  04-23</fb:date>
</fb:widgetShipment>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Autogeneration of RDFa

- Whenever you see HTML being generated automatically, you have an opportunity to create RDFa.
  - Movie timetables, price lists, and so many other web pages from a backend database.
    - Fertile ground for easy RDFa generation
- **RDFa's ease of incorporating proper RDF triples into straightforward HTML one of the great milestones in the building of the semantic web**

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Complete RDFa Example I

- XHTML with RDFa:

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:foaf="http://xmlns.com/foaf/0.1/" >

  <head profile="http://www.w3.org/2003/g/data-view" >
    <link rel="transformation" href="RDFa2RDFXML.xsl"/>
    <title>Biblio description</title>
  </head>

  <body>
    <h2>Biblio description</h2>
    <dl about="http://www.w3.org/TR/2004/REC-rdf-mt-20040210/">

      <dt>Title</dt>
       <dd property="dc:title">RDF Semantics - W3C Recommendation 10 February 2004</dd>

      <dt>Author</dt>
       <dd rel="dc:creator" href="#a1"> <span about="#a1">

        <link rel="rdf:type" href="[foaf:Person]" />

          <span property="foaf:name">Patrick Hayes</span>
          see <a rel="foaf:homepage"
        href="http://www.ihmc.us/users/user.php?UserID=42">homepage</a>

        </span>
       </dd>
    </dl>
  </body>
</html>
```


Complete RDFa Example II

- RDF extracted:

```
PREFIX dc: <http://purl.org/dc/elements/1.1/>

<http://www.w3.org/TR/2004/REC-rdf-mt-20040210/>
dc:title "RDF Semantics - W3C Recommendation 10
  February 2004"

<http://www.w3.org/TR/2004/REC-rdf-mt-20040210/>
dc:creator <#a1>

<#a1> rdf:type
  <http://xmlns.com/foaf/0.1/Person"/>

<#a1> foaf:name "Patrick Hayes"

<#a1> foaf:homepage
  <http://www.ihmc.us/users/user.php?UserID=42/>
```

GRDDL

- **GRDDL** is a mechanism for Gleaning Resource Descriptions from Dialects of Languages
  - Defines a standard for declaring that a web page or XML can be transformed into an RDF graph, as well as the algorithms o mechanisms for performing such transformations
    - glean –verb (used with object)
      1. to learn, discover, or find out, usually little by little or slowly. –verb (used without object)
    - Joins the gap between microformats and RDFa
- A markup for
  - … declaring that an XML document includes gleanable data (PROFILE) or
  - … linking to an algorithm (typically XSLT) for gleaning the RDF data from the document (TRANSFORMATION)
- The markup includes:
  1. a namespace-qualified attribute for use in general-purpose XML documents.
     ```
     xmlns:grddl='http://www.w3.org/2003/g/data-view#'
     grddl:transformation="glean_title.xsl"
     ```
  2. a profile-qualified link relationship for use in valid XHTML documents.
     ```
     <head profile="http://www.w3.org/2003/g/data-view">
     <link rel="transformation" href="glean_title.xsl" />
     ```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

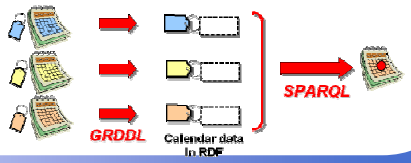
GRDDL Implementations

- List of implementations:
  - http://esw.w3.org/topic/GrddlImplementations
- GRDDL transformations currently available:
  - http://esw.w3.org/topic/MicroModels

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

GRDDL and XHTML: Scheduling a Meeting

- Example 1:
  - "Jane is trying to see if at any point next year she can schedule a meeting with all three of her friends, despite the fact that all of her friends publish their calendar data in different ways. "
- GRDDL provides a number of ways for GRDDL transformations to be associated with content, each of which is appropriate in different situations.
  - The simplest method for authors of HTML content is to embed a reference to the transformations directly using a `link` element in the `head` of the document.



Jane's Friend Robin Uses hCalendar MicroFormat

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
    <title>Robin's Schedule</title>
  </head>
  <body>
    <ol class="schedule">
      <li>2006
        <ol>
          <li class="vevent">
            <strong class="summary">Fashion Expo</strong> in
            <span class="location">Paris, France</span>:
            <abbr class="dtstart" title="2006-10-20">Oct 20</abbr> to
            <abbr class="dtend" title="2006-10-23">22</abbr>
          </li>
          ...
        </ol>
      </li>
      <li>2007
        <ol>
          <li class="vevent">
            <strong class="summary">Web Design Conference</strong> in
            <span class="location">Edinburgh, UK</span>:
            <abbr class="dtstart" title="2007-01-08">Jan 8</abbr> to
            <abbr class="dtend" title="2007-01-11">10</abbr>
          </li>
          ...
        </ol>
      </li>
    </ol>
  </body>
</html>
```

Jane's friend Robin uses hCalendar microformat

- To explicitly relate the data in this document to the RDF data model the author needs to make two changes.
  1. Add a profile attribute to the head element to denote that her document contains GRDDL metadata.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
    "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head profile="http://www.w3.org/2003/g/data-view">
    <title>Robin's Schedule</title>
  </head>
  <body>
  ...
```

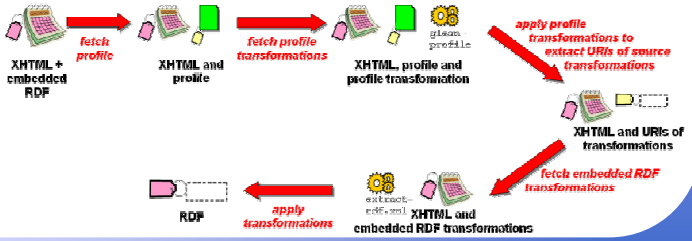
  2. Add a `link` element containing the reference to the specific GRDDL transformation for converting HTML containing `hCalendar` patterns into RDF.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
    "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
 <head profile="http://www.w3.org/2003/g/data-view">
    <title>Robin's Schedule</title>
    <link rel="transformation" href="http://www.w3.org/2002/12/cal/glean-
    hcal"/>
  </head>
  <body>
  ...
```

Jane's friend David uses embedded RDF

- Embedded RDF has a link to a GRDDL transformation in its profile document.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
   "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head profile="http://purl.org/NET/erdf/profile">
    <title>Where Am I</title>
    <link rel="schema.cal" href="http://www.w3.org/2002/12/cal#" />
  </head>
  <body>
    <p class="-cal-vevent" id="tiddlywinks">
      From <span class="cal-dtstart" title="2006-10-07">7 October, 2006</span>
      to <span class="cal-dtend"  title="2006-10-13">12 October, 2006</span>
      I will be attending the <span class="cal-summary">National Tiddlywinks
      Championship</span> in
      <span class="cal-location">Bognor Regis, UK</span>.
    </p>
    …
  </body>
</html>
```



Jane herself uses RDFa to encode her schedule

- Jane stores her calendar directly in RDFa (http://www.w3.org/TR/grddl-primer/janeschedule.html)
  - GRDDL Transformation for RDFa to convert RDFa to RDF/XML is available at:
    - http://www.w3.org/TR/2007/NOTE-grddl-primer-20070628/RDFa2RDFXML.xsl
- Example:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML+RDFa 1.0//EN"
                      "http://www.w3.org/MarkUp/DTD/xhtml+rdfa-1.dtd">
<html xmlns:cal="http://www.w3.org/2002/12/cal/icaltzd#"
xmlns:xs="http://www.w3.org/2001/XMLSchema#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns="http://www.w3.org/1999/xhtml">
 <head profile="http://www.w3.org/2003/g/data-view">
  <title>Jane's Blog</title>
    <link rel="transformation" href="RDFa2RDFXML.xsl"/>
 </head>
 <body>
  <p about="#event1" class="cal:vevent">
      <b property="cal:summary">Weekend off in Iona</b>:
      <span property="cal:dtstart" content="2006-10-21" datatype="xs:date">Oct
  21st</span>
      to <span property="cal:dtend" content="2006-10-21"  datatype="xs:date">Oct
  23rd</span>.
      See <a rel="cal:url"
  href="http://freetime.example.org/">FreeTime.Example.org</a> for
      info on <span property="cal:location">Iona, UK</span>.
  </p>
  …
 </body>
</html>
```


Merging the three friends' schedule

- One of the advantages of the RDF data model is that RDF data can be easily merged by adding it to a RDF store
  - Jane can merge and query all the calendars together once they are transformed into RDF through GRDDL
  - Jane uses SPARQL to query her data, which automatically merges the calendar data sources before running the query.
- Next slide SPARQL query explanation:
  - The SELECT line determines which variables will appear in the results, here one of the start dates, one of the stop dates, a location and a summary.
  - The FROM lines identify the data sources to use in the query, in this case the RDF/XML derived from Jane, David and Robin's original documents.
  - The WHERE section provides a pattern which can match three events.
- Results of next slide query:

start1	stop1	loc1	summ1
"2007-01-08	"2007-01-11"	Edinburgh, UK"	Web Design Conference"

Example 1 SPARQL

```
PREFIX ical: <http://www.w3.org/2002/12/cal/icaltzd#>
PREFIX xs: <http://www.w3.org/2001/XMLSchema#>
SELECT ?start1 ?stop1 ?loc1 ?summ1 ?summ2 ?summ3
FROM <http://www.w3.org/TR/grddl-primer/janeschedule.rdf>
FROM <http://www.w3.org/TR/grddl-primer/robin-hcal-grddl.rdf>
FROM <http://www.w3.org/TR/grddl-primer/david-erdf.rdf>
WHERE
  {
     ?event1 a ical:Vevent;
              ical:summary ?summ1 ;
              ical:dtstart ?start1 ;
              ical:dtend ?stop1 ;
              ical:location ?loc1.
     ?event2 a ical:Vevent;
              ical:summary ?summ2 ;
              ical:dtstart ?start2;
              ical:dtend ?stop2;
              ical:location ?loc2.
     ?event3 a ical:Vevent;
              ical:summary ?summ3 ;
              ical:dtstart ?start3;
              ical:dtend ?stop3;
              ical:location ?loc3.
      FILTER (  ?event1 != ?event2 &&  ?event2 != ?event3  &&  ?event1 != ?event3  ) .
      FILTER ( xs:string(?start1) = xs:string(?start2) ).
      FILTER ( xs:string(?stop1) = xs:string(?stop2) ).
      FILTER ( xs:string(?loc1) = xs:string(?loc2) ).
      FILTER ( xs:string(?start1) = xs:string(?start3) ).
      FILTER ( xs:string(?stop1) = xs:string(?stop3) ).
      FILTER ( xs:string(?loc1) = xs:string(?loc3) ).
      FILTER ( xs:string(?start3) = xs:string(?start2) ).
      FILTER ( xs:string(?stop3) = xs:string(?stop2) ).
      FILTER ( xs:string(?loc3) = xs:string(?loc2) ).
      FILTER ( xs:string(?summ1) <= xs:string(?summ2) ).
      FILTER ( xs:string(?summ2) <= xs:string(?summ3) ).
}
```

Mashing-Up Microformats: Booking a Hotel

- Example 2:
  - "Jane wants to book a hotel in Edinburgh"
  - Combines data dialects as different as reviews and social networks in order to guarantee the booking a hotel with a high review from a trusted friend.
- This example highlights the role of GRDDL in aggregating data from a variety of different formats and of using RDF as a common format to "mashup" all sorts of data, not just calendar data.
  - XFN file can be converted to RDF with the use of another GRDDL Transform for XFN
    - http://www.w3.org/TR/2007/NOTE-grddl-primer-20070628/grokXFN.xsl
  - The hotel review file uses `hReview` that can be converted to RDF with the following transformation:
    - http://www.w3.org/TR/2007/NOTE-grddl-primer-20070628/hreview2rdfxml.xsl
- Using GRDDL we can glean information, including the ratings about the hotels and the reviews given by Jane's friends.

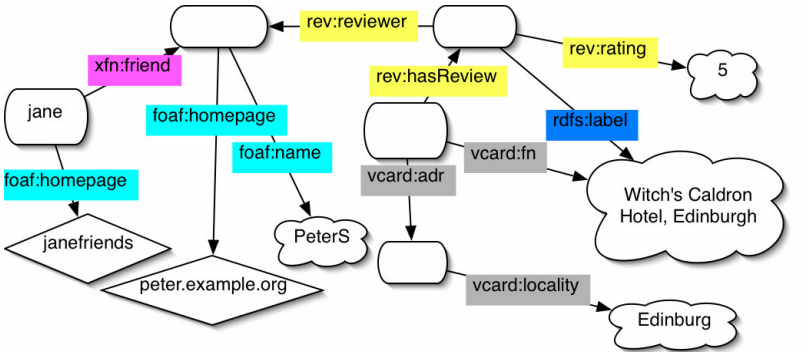
XFN Microformat

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
    "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd" >
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
   <head profile="http://www.w3.org/2003/g/data-view
        http://gmpg.org/xfn/11">
      <link rel="transformation" href="http://www.w3.org/TR/grddl-
   primer/grokXFN.xsl " />
      <title>Jane's XFN List</title>
   </head>
   <body>
      <h1>Jane's <abbr title="XHTML Friends Network">XFN</abbr> List</h1>
      <ul class="xoxo">
        <li class="vcard"><a href="http://peter.example.org/" class="url
   fn" rel="met collegue friend">Peter Smith</a></li>
        <li class="vcard"><a href="http://john.example.org/" class="url
   fn" rel="met">John Doe</a></li>
        <li class="vcard"><a href="http://paul.example.org/" class="url
   fn" rel="met">Paul Revere</a></li>
        </ul>
   </body>
</html>
```


hReview Microformat

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
  <head profile="http://www.w3.org/2003/g/data-view">
    <title>Hotel Reviews from Example.com</title>
    <link rel="transformation"
      href="http://www.w3.org/TR/grddl-primer/hreview2rdfxml.xsl"/>
</head>
<div class="hreview" id="_665">
  <div class="item vcard">
  <b class="fn org">Witch's Caldron Hotel, Edinburgh</b>
  <ul>
    <li>
      <a class="url" href="http://witches.example.com/">Homepage</a>    </li>
  </ul>
  <span><span class="rating">5</span> out of 5 stars</span>
  <ul>
    <li class="adr">
      <div class="type">intl postal parcel work</div>
      <div class="street-address">313 Cannongate</div>
      <div><span class="locality">Edinburgh</span>, <span class="postal-code">EH8 8DD
  </span> <span class="country-name">United Kingdom</span></div>
    </li>
  </ul>
  <div class="tel"><abbr class="type" title="work msg">Homepage</abbr>: <a
    class="value" href="tel:+44 1317861235">+44 1317862235</a></div>
```

Merging reviews and people



Example 2 SPARQL Query

- "Find hotels with specific ratings or higher from a group of her trusted friends"

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX rev: <http://www.purl.org/stuff/rev#>
PREFIX vcard: <http://www.w3.org/2006/vcard/ns#>
PREFIX xfn: <http://gmpg.org/xfn/11#>
SELECT DISTINCT ?rating ?name ?region ?homepage ?xfnhomepage ?hotelname
FROM <http://www.w3.org/TR/grddl-primer/janefriends.rdf>
FROM <http://www.w3.org/TR/grddl-primer/hotel-data.rdf>
WHERE {
?x rev:hasReview ?review;
   vcard:ADR ?address;
   vcard:FN ?hotelname.
?review rev:rating ?rating .
?address vcard:Locality ?region.
FILTER (?rating > "2" && ?region = "Edinburgh").
?review rev:reviewer ?reviewer.
?reviewer foaf:name ?name;
          foaf:homepage ?homepage.
?y xfn:friend ?xfnfriend.
?xfnfriend foaf:homepage ?xfnhomepage.
FILTER (?xfnhomepage = ?homepage).
}
```

rating	name	region	homepage	xfnhomepage	hotelname
"5"	"PeterS"	"Edinburgh"	<http://peter.example.org/>	<http://peter.example.org/>	"Witch's Caldron Hotel, Edinburgh"

Summary Mechanisms to Add Semantics

- **Microformats**
  - Designed for humans first, machines second
  - Very short steps to solve specific problems one at a time
  - Help putting your data into HTML, but no standard way to get the data out.
  - Cannot be validated easily, mixing `hCard` and `hCal` there's no way to guarantee you will interpret it correctly.
    - Very domain specific
- **RDFa vs. microformats**
  - when the underlying ontology/vocabulary is simply way too complicated to be re-expressed in a microformat
    - microformats are good for micro-metadata (e.g. name, address, event dates)
    - but not for complex embedded (e.g. proteins, geological data)
  - when you need to combine several ontologies/vocabularies in one page
    - handle possible conflicts between primitives e.g. class="name" and class="name" vs. rel="foaf:name" and rel="prod:name"
- **GRDDL**
  - Middleware to join microformats and RDFa
    - Glue for the different existing semantization mechanisms
- All of them together will help us rich the vision of a more meaningful without pain Web

Semantic Web vs. semantic web

	Semantic Web	semantic web
Philosophy	Build a common data format for expressing the meaning of data. Use ontologies to help machines to understand web content.	Humans first, machines second. Encode existing Web content with special tags.
Language	**RDF, RDFS, OWL**	**Microformats (based on XHTML), RDFa, GRDDL**
Format	Must be well-formed RDF documents	Anything goes, as long as its XHTML
Semantic	Defined by the underlying ontology model (e.g., OWL)	Loosely defined. No formal semantic model, unless RDFa is used
Examples	FOAF, OWL-S, OWL-Time	XFN (social network), hCard (contact), hReview (opinions), rel-tag (tagging)

Web 2.0 vs. Semantic Web

- Web 2.0 and Semantic Web are complementary
  - Challenge:
    - How to integrate Web 2.0 data on a Web scale?
    - How to enable users to create semantically rich annotations?
  - A good example of these combinations is Revyu.com

Web 2.0

- Web 2.0:
  - Elicit and reuse user-generated content, support social and collaborative interaction on the web
  - Provide engaging user interactions based on AJAX
  - Most Web 2.0 applications provide info which cannot be combined with other sources
    - My friend in Orkut is a stranger on MySpace
    - There is a need to publish data in formats easily processed by third parties
  - With mash-ups you can combine data obtained from different APIs (Amazon, Flickr) from different sources, but new mappings and XSL transformations are required
    - RDF does not suffer this limitation

Semantic Web

- Semantic Web:
  - Data published in machine readable formats
  - Gives formal semantics through the use of shared semantics
  - Publishing data in RDF lowers the barriers to its reuse by others
    - Although two organisations may describe data using the same schema, mappings can be defined between the two models
    - RDF allows one document to mix statements that use elements from any number of ontologies

Web 2.0 and Mash-up Problems

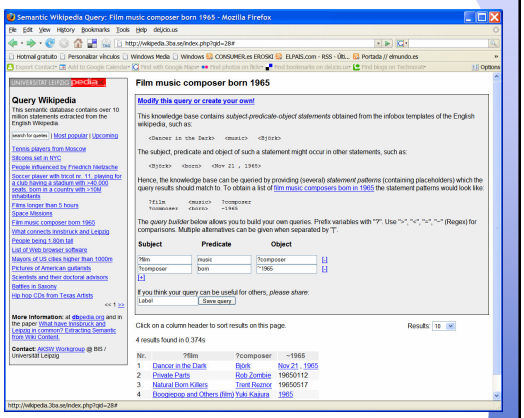
- Vanilla XML Web services require parsing XML trees to retrieve the desired data
  - Processing remains tied to underlying syntactic rather than semantic structure of the data
  - Need of writing custom handlers to interact with each API
  - No common language is available for querying and integrating such data sources
- **The challenge is to create truly flexible Web scale mash-ups**
  - SPARQL query language for the Semantic Web enables standardised access to distributed data sources.

Semantic Web Problems

- Faces similar challenges to Web 2.0 mash-ups if it is to reach widespread adoption
  - Few mechanisms exist that allow non-specialist users to contribute to the Semantic Web
  - Completing forms in a web browser content that is immediately usable on the Semantic Web
    - Without any user knowledge of RDF, ontologies or Semantic Web
  - Promote user semantic annotation with guided user input through the use of forms
    - Significant effort must be given to developing compelling interfaces able to display structured, linked data from across the Web
  - First attempts → dbpedia RDF-ising Wikipedia

Dbpedia.org

- Querying Wikipedia like a Database.
- DBpedia.org is a community effort to extract structured information from Wikipedia and to make this information available on the Web.
- DBpedia allows you to ask sophisticated queries against Wikipedia and to link other datasets on the Web to Wikipedia data.
- For querying dbpedia goto: http://dbpedia.org/sparql
- More info at: http://dbpedia.org/docs/



Semantic Mash-ups

- SO FAR …
  - Mashups are great if you're looking for one kind of thing (coffee shops, hotels, gyms) and come from one source (especially when that source is an amalgamator like Citysearch or even Google)
- Semantic Mashups will merge information from multiple feeds and organize the results
  - First step is to turn all of your data sources into feeds that can be mashed up
    - RDFa lets Web site developers make an HTML page do double duty as a presentation page and as a machine-readable source of structured data in RDF.

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Example RDFa for a Metro Station

```
<p>
  <font size="2">
    <div name="rlhv" id="rlhv">
      <link rel="rdf:type" href="[LAMetro:RedlineStation]"/>
      <meta property="rdfs:label">Hollywood/Vine</meta> <br/>
      <a
 href="../../about_us/metroart/images/pict_mrlhv.jpg">Station
 Image</a>
    <b><br/>
    </b><meta property="geo:address">6250 Hollywood Bl.<br/>
 Los Angeles 90038 <br/></meta>
 60 Park/Ride Lot Spaces (Parking Fee)<br/>
       <a
 href="../../projects_plans/bikeway_planning/images/bike_rack_m
 rhv.jpg">18 Bike Rack Spaces</a>
    </div>
  </font>
</p>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

OWL allows us to combine RDF files

```
<owl:Class rdf:about="#LALocale"/>
<owl:Class rdf:about="#Entertainment">
 <rdfs:subClassOf rdf:resource="#LALocale"/>
</owl:Class>
<owl:Class rdf:about="#LAMetro">
 <rdfs:subClassOf rdf:resource="#LALocale"/>
</owl:Class>
<owl:Class rdf:about="#Fitness">
 <rdfs:subClassOf rdf:resource="#LALocale"/>
</owl:Class>
<owl:Class rdf:ID="GoldLineStation">
 <rdfs:subClassOf rdf:resource="#LAMetro"/>
</owl:Class>
<owl:Class rdf:ID="GreenLineStation">
 <rdfs:subClassOf rdf:resource="#LAMetro"/>
</owl:Class>
<owl:Class rdf:ID="BlueLineStation">
 <rdfs:subClassOf rdf:resource="#LAMetro"/>
</owl:Class>
<owl:Class rdf:ID="RedLineStation">
 <rdfs:subClassOf rdf:resource="#LAMetro"/>
</owl:Class>
```


OWL allows us to combine RDF files

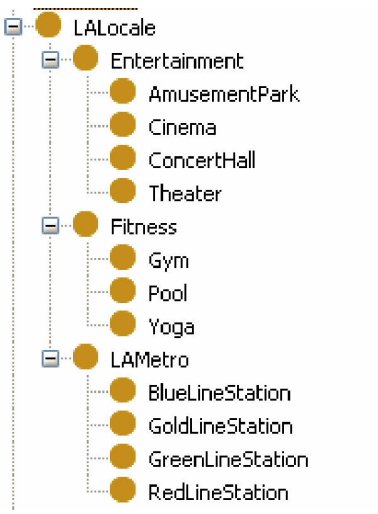
```
<owl:Class rdf:ID="Cinema">
 <rdfs:subClassOf rdf:resource="#Entertainment"/>
</owl:Class>
<owl:Class rdf:ID="Gym">
 <rdfs:subClassOf rdf:resource="#Fitness"/>
</owl:Class>
<owl:Class rdf:ID="Pool">
 <rdfs:subClassOf rdf:resource="#Fitness"/>
</owl:Class>
<owl:Class rdf:ID="Yoga">
 <rdfs:subClassOf rdf:resource="#Fitness"/>
</owl:Class>
<owl:Class rdf:ID="AmusementPark">
 <rdfs:subClassOf rdf:resource="#Entertainment"/>
</owl:Class>
<owl:Class rdf:ID="Theater">
  <rdfs:subClassOf rdf:resource="#Entertainment"/>
</owl:Class>
<owl:Class rdf:ID="ConcertHall">
 <rdfs:subClassOf rdf:resource="#Entertainment"/>
</owl:Class>
```

Specifying restrictions with OWL

```
<owl:Class rdf:about="#BlueLineStation">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:hasValue>
http://www.topquadrant.com/images/icons/bluetrain.gif
      </owl:hasValue>
      <owl:onProperty
  rdf:resource="http://www.topquadrant.com/maps/mapMo
  del.owl#hasIcon"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

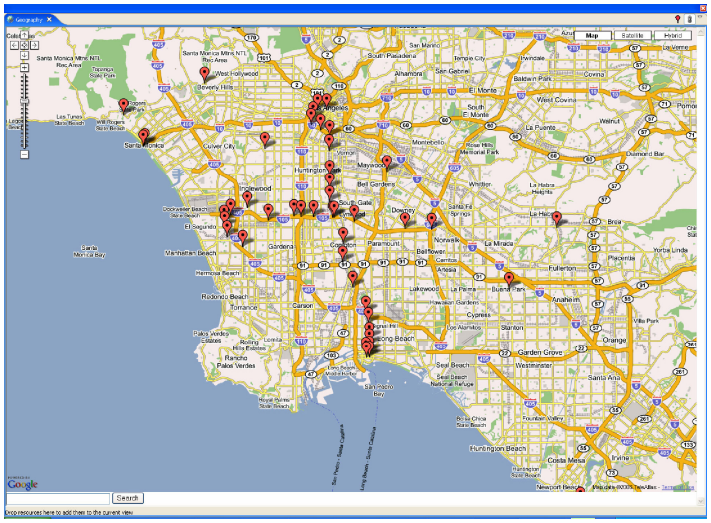
Semantic GIS Service Ontology



Tecnológico Fundación Deusto Teknologikoa Deustu Fundazioa

Universidad de Deusto
Deustuko Unibertsitatea

Semantic Mash-up Front-end



Tecnológico Fundación Deusto
Teknologikoa Deustu Fundazioa

Universidad de Deusto
Deustuko Unibertsitatea

References

- Semantic Web
  - RDF:
    - http://www.javaworld.com/javaworld/jw-12-2005/jw-1205-wicked_p.html
  - OWL:
    - A No-Nonsense Guide to Semantic Web Specs for XML People
      - http://www.betaversion.org/~stefano/linotype/news/57/
    - OWL Web Ontology Language Guide
      - http://www.w3.org/TR/owl-guide/
- Lower-s Semantic Web
  - Bootstrapping the Semantic Web with GRDDL, Microformats, and RDFa
    - http://www-sop.inria.fr/acacia/personnel/Fabien.Gandon/tmp/grddl/grddl-introduction-v3/
  - Real-world semantics
    - http://tantek.com/presentations/2004etech/realworldsemanticspres.html
  - Mashups Beyond Google Maps from a Geospatial Semantic Web Perspective
    - http://colab.cim3.net/file/work/SICoP/2006-06-20/HChen06202006.ppt

Tecnológico Fundación Deusto
Teknologikoa Deustu Fundazioa

References

- Microformats
  - http://microformats.org/
  - Alex Faaborg content on Microformats
    - http://blog.mozilla.com/faaborg/2006/12/11/microformats-part-0-introduction/
    - http://people.mozilla.com/~faaborg/files/20070416-web20Expo/faaborg-Web20Expo.pdf
- RDFa
  - Introducing RDFa
    - http://www.xml.com/lpt/a/1691
  - Introducing RDFa, Part Two
    - http://www.xml.com/lpt/a/1698
  - RDFa Primer 1.0
    - http://www.w3.org/TR/xhtml-rdfa-primer/
  - RDF/A Syntax
    - http://www.w3.org/2001/sw/BestPractices/HTML/2005-rdfa-syntax
- GRDDL:
  - Primer: Using GRDDL & Microformats to Aggregating data
    - http://suda.co.uk/sandbox/GRDDL/Primer.htm
  - GRDDL Primer
    - http://www.w3.org/TR/2007/NOTE-grddl-primer-20070628/
  - GRDDL Digital Library Example
    - http://www-sop.inria.fr/acacia/personnel/Fabien.Gandon/tmp/grddl/rdfaprimer/PrimerRDFaSection.html

References

- JENA
  - Introduction to Jena
    - http://www-128.ibm.com/developerworks/java/library/j-jena/
  - Search RDF data with SPARQL
    - http://www-128.ibm.com/developerworks/library/j-sparql

References

- Restful Semantic Web Services
  - http://blogs.sun.com/bblfish/entry/restful_semantic_web_services
- Primer: Getting into RDF & Semantic Web using N3
  - http://www.w3.org/2000/10/swap/Primer
- Web 2.0 & Semantic Web
  - How to Combine the Best of Web 2.0 and a Semantic Web
    - http://kmi.open.ac.uk/people/tom/papers/heath-motta-www2007dev-combining-web20-semantic-web.pdf
  - Mashups and Semantic Mashups, The Value Of RDF: Taking the Web to the Next Step
    - http://www.web2journal.com/read/361294.htm