

Chapitre IV

RDF-Schema

- 1 Présentation
- 2 Classes
- 3 Propriétés
- 4 Compléments
- 5 Exemple : Dublin Core
- 6 Exemple : FOAF

Insuffisances de RDF

RDF permet de représenter des déclarations de propriétés sur des ressources. . .

Mais ne permet pas n'exprimer des connaissances sur les propriétés ou sur les types de ressources :

- Quelles sont les propriétés autorisées sur un type de ressources ?
- Quelles sont les valeurs autorisées pour une propriété ?
- Quels sont les liens entre les types de ressources (généralisation / spécialisation) ?
- . . .

→ Définir le *vocabulaire*.

Chapitre IV

RDF-Schema

- 1 Présentation
- 2 Classes
- 3 Propriétés
- 4 Compléments
- 5 Exemple : Dublin Core
- 6 Exemple : FOAF

RDF-Schema : Présentation

Exemple

Un Livre est une sorte de Publication, et un Magazine aussi.
Toute Publication a un titre, un Livre a un ou plusieurs
auteur, ...

RDF Schema permet de décrire des classes et des propriétés.

Attention.

RDF-Schema ne fournit pas un vocabulaire (Publication, auteur ...), il permet de *définir* un vocabulaire.

Recommandation W3C : <http://www.w3.org/TR/rdf-schema/>
« *RDF Vocabulary Description Language* »

RDF-Schema : Présentation

RDF-Schema est un « système de typage » pour RDF, comparable à l'approche orientée objet :

- Classes (types) et instances (ressources)
- Relation hiérarchique (héritage)
- Propriétés (attributs)
- Mais pas de méthodes : description uniquement.
- Pas de structure fixée : d'autres propriétés peuvent être ajoutées.

RDF-Schema : Présentation

- Une description de vocabulaire est appelée *schema*.
- Un RDF-Schema est représenté en RDF sous la forme de triplets.
 - Donc toute application qui gère le RDF peut gérer le RDF-Schema...
 - ... mais une application doit être étendue pour prendre en compte le « sens » du schema.
- Les ressources et prédicats de RDF-Schema sont définis à l'URI <http://www.w3.org/2000/01/rdf-schema#> (identifié par `rdfs`).

Chapitre IV

RDF-Schema

- 1 Présentation
- 2 **Classes**
- 3 Propriétés
- 4 Compléments
- 5 Exemple : Dublin Core
- 6 Exemple : FOAF

Classes

Une *classe* est un type (ou une catégorie) qui regroupe plusieurs *instances* (ressources) partageant des caractéristiques communes.

- Une classe est identifiée par une URI.
- Pour préciser qu'une URI est une classe, il faut écrire que cette ressource a pour `rdf:type rdfs:Class`.

Exemple

```
exvoc:Vehicule rdf:type rdfs:Class .
```

- Pour préciser qu'une URI est une instance d'une classe, il faut écrire que cette ressource a pour `rdf:type` la classe.

Exemple

```
exvoit:v1234AB49 rdf:type exvoc:Vehicule .
```

- Convention : Un nom de classe commence par une majuscule, une instance par une minuscule.

Classes : Multi-instanciation

Multi-instanciation

- **approche objets.** Un objet peut être instance de plusieurs classes (mais la plupart des langages objets ne proposent pas la multi-instanciation).
- **RDF-Schema.** Une ressource peut avoir plusieurs types classes.

Exemple

```
exvoit:v1234AB49 rdf:type exvoc:Vehicule .
exvoit:v1234AB49 rdf:type exvoc:PeutEtreLoue .
```

Évite de créer une sous-classe (de Vehicule et PeutEtreLoue) si une seule instance est concernée.

Classes : Généralisation / Spécialisation

Une classe peut être sous-classe (spécialisation) d'une ou plusieurs classes.

Exemple

```
exvoc:Voiture rdf:type rdfs:Class .
exvoc:Voiture rdfs:subClassOf exvoc:Vehicule .
exvoc:Camion rdf:type rdfs:Class .
exvoc:Camion rdfs:subClassOf exvoc:Vehicule .
exvoc:VoitureALouer rdf:type rdfs:Class .
exvoc:VoitureALouer rdfs:subClassOf exvoc:Voiture,
exvoc:PeutEtreLoue .
```

Toute instance de la sous-classe (VoitureALouer) est instance de toutes les super-classes (Voiture, PeutEtreLoue, Vehicule). (mais l'application doit gérer le RDF-Schema pour inférer ceci)

Classes : Généralisation / Spécialisation

Syntaxe RDF/XML : Déclaration de classe

Exemple (Syntaxe standard)

```
<rdf:Description rdf:ID="Voiture">
  <rdf:type rdf:resource=
    "http://www.w3.org/2000/01/rdf-schema#Class"/>
  <rdfs:subClassOf rdf:resource="#Vehicule"/>
</rdf:Description>
```

Abréviation : utilisation du type de la ressource comme nom de l'élément...

Ici, le type est rdfs:Class.

Exemple (Syntaxe abrégée)

```
<rdfs:Class rdf:ID="Voiture">
  <rdfs:subClassOf rdf:resource="#Vehicule"/>
</rdfs:Class>
```

Classes : Généralisation / Spécialisation

Syntaxe RDF/XML : Instanciation

Exemple (Syntaxe standard)

```
<rdf:RDF
  xmlns:rdf=... xmlns:rdfs=...
  xmlns:exvoc="http://www.exemple.org/voc#">
  <rdf:Description rdf:ID="v1234AB49">
    <rdf:type
      rdf:resource="http://www.exemple.org/voc#Voiture"/>
  </rdf:Description>
</rdf:RDF>
```

Exemple (Syntaxe abrégée)

```
...
<exvoc:Voiture rdf:ID="v1234AB49"/>
...
```

Chapitre IV

RDF-Schema

- 1 Présentation
- 2 Classes
- 3 **Propriétés**
- 4 Compléments
- 5 Exemple : Dublin Core
- 6 Exemple : FOAF

Propriétés

Une *propriété* est définie par :

- un nom ;
- un domaine (types de ressources sur lesquels peut porter la propriété) ;
- un co-domaine (valeurs autorisées pour la propriété).

⇒ permet de poser des contraintes sur les ressources (ou littéraux) qui peuvent être utilisés dans une déclaration utilisant la propriété :

- Permet de mieux définir le sens d'une propriété.
- Contrôles.
- Traitements particuliers.

Propriété

- Une propriété est identifiée par une URI.
- Convention : Un nom de propriété commence par une minuscule.
- Pour préciser qu'une URI est une propriété, il faut écrire que cette ressource a pour `rdf:type rdf:Property`.

Exemple

```
exvoc:vitesseMax rdf:type rdf:Property .
exvoc:aPourMarque rdf:type rdf:Property .
```

- Le co-domaine d'une propriété définit le type des objets autorisés pour la propriété.

Exemple

```
exvoc:vitesseMax rdfs:range xsd:integer .
exvoc:aPourMarque rdfs:range exvoc:MarqueVehicule .
```

Propriété

Co-domaine

Une propriété peut avoir :

- 0 propriété range. « tout » (ressources et littéraux) est autorisé comme objet.
- 1 propriété range. Seules les ressources de ce type (ou les littéraux typés de ce type) sont autorisés comme objets de la propriété.
- Plusieurs propriétés range. Les seules ressources autorisées comme objet de la propriété doivent être instances de **tous** les types figurant comme propriété range.

Exemple

```
exvoc:aPourMarque rdfs:range exvoc:MarqueVehicule .
exvoc:aPourMarque rdfs:range exvoc:MarqueEuropeenne .
```

Les objets de aPourMarque doivent être des instances de MarqueVehicule et de MarqueEuropeenne.

Propriété

Domaine

- Le domaine d'une propriété définit le type des sujets autorisés pour la propriété.

Exemple

```
exvoc:vitesseMax rdfs:domain exvoc:Vehicule .
exvoc:aPourMarque rdfs:domain exvoc:Vehicule .
```

Propriété

Domaine

Une propriété peut avoir :

- 0 propriété domain. Toute ressource est autorisée comme sujet.
- 1 propriété domain. Seules les ressources de ce type sont autorisées comme sujets de la propriété.
- Plusieurs propriétés domain. Les seules ressources autorisées comme sujet de la propriété doivent être instances de **tous** les types figurant comme propriété domain.

Exemple

```
exvoc:kilometrageMaxi rdf:type rdf:Property .
exvoc:kilometrageMaxi rdfs:range xsd:integer .
exvoc:kilometrageMaxi rdfs:domain exvoc:Vehicule,
exvoc:PeutEtreLoue .
```

Les ressources sujets de kilometrageMaxi doivent être des instances de Vehicule et de PeutEtreLoue.

Propriété

Spécialisation de propriétés

RDF-Schema permet d'exprimer qu'une propriété est une spécialisation d'une autre propriété.

Exemple

```

exvoc:assurance rdf:type rdf:Property .
exvoc:assurance rdf:domain exvoc:Vehicule,
exvoc:PeutEtreLoue .
exvoc:assurance rdf:range exvoc:Client .
exvoc:assuranceConducteur rdf:type rdf:Property .
exvoc:assuranceConducteur rdf:domain exvoc:Vehicule,
exvoc:PeutEtreLoue .
exvoc:assuranceConducteur rdf:range exvoc:Client,
exvoc:PossedePermis .
assuranceConducteur est une spécialisation d'assurance : Un
client qui est « assuré comme conducteur » est aussi « assuré ».

```

Propriété

Spécialisation de propriétés

La spécialisation de propriétés est représentée par une propriété `rdfs:subPropertyOf` ayant pour sujet la spécialisation et pour objet la généralisation.

Exemple

```

exvoc:assuranceConducteur rdfs:subPropertyOf
exvoc:assurance .

```

Si « A a pour P_2 B » et « P_2 est une spécialisation de la propriété P_1 » alors on peut inférer « A a pour P_1 B ».

Exemple

```

si exvoit:v1234AB49 rdf:assuranceConducteur
exclient:c12 .
alors on peut inférer exvoit:v1234AB49 rdf:assurance
exclient:c12 .

```

Propriété

Spécialisation de propriétés

Une propriété P peut être une spécialisation de :

- 0 propriété.
- 1 propriété P' .
En plus des propriétés `range` et `domain` de P , les sujets et objets de P doivent respecter le domaine et co-domaine de P' .
- plusieurs propriétés P_1, \dots, P_n .
En plus des propriétés `range` et `domain` de P , les sujets et objets de P doivent respecter les domaines et co-domaines de tous les P_i .

Propriété

Différences RDF-Schema / POO

- **Programmation OO**

Une classe *contient* des attributs (typés).

Toutes les instances de la classe ont *une* valeur pour ces attributs.

Si deux classes ont un attribut de même nom, ces attributs n'ont rien en commun : Les types peuvent être différents, aucun lien entre les deux.

- **RDF-Schema**

Des classes sont définies ; Des propriétés sont définies ; (de façon séparée).

Les instances de la classe peuvent avoir (ou pas) une valeur pour les propriétés.

Par défaut une propriété est « globale » : peut être appliquée à toute classe sujet, peut recevoir tout objet.

Propriété

Différences RDF-Schema / POO

Particularités de RDF-Schema

- Une propriété peut être définie sans domaine : applicable sur toute ressource. . .
- Elle peut être spécialisée pour certains domaines.

Exemple

La propriété `createur` n'a pas de domaine et pour range `Personne`.

Elle est spécialisée en `auteur` avec domaine `Livre` et range `Ecrivain`.

Elle est spécialisée en `concepteur` avec domaine `Logiciel` et range `Informaticien`.

...

Propriété

Limites

La souplesse de la définition des propriétés n'est pas toujours suffisante.

Exemple

`aPourParent` pour `Humain`, `Chien`, `Chat` etc.

`Humain`, `Chien`, `Chat` sont des spécialisations de `EtreVivant`.

Quel est le domaine et co-domaine de `aPourParent` ?

- `Humain - Humain`. `aPourParent` ne peut être utilisé sur `Chien`.
- `EtreVivant - EtreVivant`. Un `Chien` peut avoir pour parent un `Chat`.
- `EtreVivant - EtreVivant + Spécialisations de aPourParent` : `aPourParentHumain : Humain - Humain`. Seule solution qui permet d'exprimer les contraintes, mais nouvelles propriétés.

D'autres langages de définition du vocabulaire fournissent une meilleure représentation.

Propriété

Limites

En POO, quand une classe possède *une* propriété, elle est

- unique. C'est la seule propriété associée aux instances de la classe.
- obligatoire. Toute instance de la classe a une valeur pour la propriété.

Dans RDF-Schema, quand une classe possède *une* propriété. . .

- les instances de la classe peuvent ne pas avoir de valeur pour la propriété.
- les instances de la classe peuvent avoir plusieurs valeurs pour la propriété (du bon type?).
- les instances de la classe peuvent avoir d'autres propriétés.

RDF-Schema n'offre pas de mécanisme permettant de poser plus de contraintes. . .

Propriété

Limites

. . . sauf si certaines applications le décident.

Exemple

`aPourParent : Humain - Humain`

- Contrainte.

Exemple

- Un Humain n'a pas de parent : l'application détecte une erreur.
- Un Humain a un parent p , mais son type n'est pas connu : erreur.
- Un Humain a un parent p , d'un type autre que Humain : erreur.

Propriété

Limites

- Inférence.

Exemple

- Un Humain n'a pas de parent : l'application lui associe un parent (Humain) inconnu.
- Un Humain a un parent p , mais son type n'est pas connu : l'application infère que p est de type Humain.
- Un Humain a un parent p , d'un autre type que Humain : l'application infère que p est de type Humain (en plus du type connu).
Ce qui peut amener à des incohérences. Car il est certain que certains couples de classes n'auront jamais d'instances communes.

Un RDF-Schema peut être interprété de différentes façons.

Chapitre IV

RDF-Schema

- 1 Présentation
- 2 Classes
- 3 Propriétés
- 4 Compléments**
- 5 Exemple : Dublin Core
- 6 Exemple : FOAF

Propriétés supplémentaires pour Classes et propriétés

- `rdfs:comment`
Commentaire. (littéral)
- `rdfs:label`
Intitulé « affichable », lisible par un humain. (littéral)
- `rdfs:seeAlso`
Lien vers une autre ressource. (ressource)
- `rdfs:isDefinedBy`
Lien vers une autre ressource qui définit la ressource.
(ressource)

...

<http://www.w3.org/TR/rdf-schema/>

Chapitre IV

RDF-Schema

- 1 Présentation
- 2 Classes
- 3 Propriétés
- 4 Compléments
- 5 Exemple : Dublin Core**
- 6 Exemple : FOAF

Présentation

Dublin Core est une norme visant à simplifier l'exploitation de documents.

- Description
- Recherche
- Localisation
- etc.

de

- Livres
- Images
- Vidéos
- Documents électroniques
- etc.

<http://www.dublincore.org>

Présentation

Pour faciliter l'exploitation :

Représenter des données sur les données du document

→ **Metadonnées.**

- Fiche carton dans une bibliothèque.
Pas facile à utiliser, pas d'automatisation.
- Entrée dans un système de recherche documentaire.
Uniquement en bibliothèque. Base locale uniquement.

⇒ Standardisation de l'expression des métadonnées.

⇒ Définir un vocabulaire simple et général pour décrire les métadonnées.

Ce vocabulaire peut être défini en RDF-Schema. Les métadonnées peuvent être exprimées en RDF.

Les termes Dublin Core

Dublin Core définit 15 *termes* qui permettent de préciser 15 types de métadonnées reliées à un document. (+ des raffinements)
 Dans le RDF-Schema, ces termes sont des propriétés.

Objectifs :

- Simple à utiliser pour un humain qui « indexe ».
- Simple à exploiter par des moteurs

Le RDF-Schema des termes Dublin Core est défini à l'URI :
<http://purl.org/dc/elements/1.1/>

Les termes Dublin Core

- title** Titre de la ressource.
- creator** Entité responsable de la création de la ressource.
- subject** Sujet de la ressource sous la forme de mots-clefs, phrases clés, codes de classification.
 Recommandation : vocabulaire contrôlé ou schéma de classification.
 Dublin Core n'impose pas un langage pour la représentation du sujet.
- description** Une description de la ressource : résumé, table des matières, extrait, etc.
- publisher** Entité responsable de la publication de la ressource.
- contributor** Entité ayant contribué à la création.
- date** Une date associée à un événement dans le cycle de vie de la ressource. Habituellement date de création ou de publication au format AAAA-MM-JJ.
 Raffinements : *created*, *available*, etc.

Les termes Dublin Core

- type** Nature de la ressource. Il est recommandé d'utiliser un vocabulaire contrôlé, *DCMI Type Vocabulary* (RDF-Schema : <http://purl.org/dc/dcmitype/>) Sound, Image, Text, etc.
- format** Format de la ressource. Type de fichier, etc.
- identifier** Référence non ambiguë à la ressource.
- source** Référence à une ressource dont la ressource décrite est dérivée.
- language** Langage de rédaction de la ressource.
- relation** Relation avec une autre ressource. Raffinements : replaces, isPartOf, etc.
- coverage** Domaine de la ressource : localisation géographique ou temporelle.
- rights** Droits d'utilisation de la ressource.

Extrait du RDF-Schema

```
<rdf:Property rdf:about=
  "http://purl.org/dc/elements/1.1/creator">
  <rdfs:label xml:lang="en-US">Creator</rdfs:label>
  <rdfs:comment xml:lang="en-US">
    An entity primarily responsible for making the
    content of the resource.
  </rdfs:comment>
  <dc:description xml:lang="en-US">
    Examples of a Creator include a person, an
    organisation, or a service. Typically, the name
    of a Creator should be used to indicate the entity.
  </dc:description>
  <rdfs:isDefinedBy rdf:resource="http://purl.org/dc/elements/1.1
  <dcterms:issued>1999-07-02</dcterms:issued>
  <dcterms:modified>2002-10-04</dcterms:modified>
  <dc:type rdf:resource=
    "http://dublincore.org/usage/documents/principles/#element"/>
</rdf:Property>
```

Exemple

```
<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.exemple.com/websem.pdf">
    <dc:creator>David Genest</dc:creator>
    <dc:title>Cours de web semantique</dc:title>
    <dc:description>Les transparents du cours de web
    semantique pour le M2 de l'universite d'Angers
    </dc:description>
    <dc:subject>
      <rdf:Bag><rdf:li>Web semantique</rdf:li>
        <rdf:li>RDF</rdf:li><rdf:li>RDF-Schema</rdf:li>
      </rdf:Bag>
    </dc:subject>
    <dc:date>2006-09-27</dc:date>
  </rdf:Description>
</rdf:RDF>
```

Avantages

- Langage simple mais complet
 - Interrogations plus précises qu'un moteur de recherche texte intégral.
 - ex : « copyright » : recherche de documents à propos du copyright, ou ayant un copyright.
 - ex : « Victor Hugo » : recherche de documents écrits par Hugo, ou sur Hugo, ou citant Hugo.
 - Possibilité d'incorporer facilement des métadonnées dans du HTML, XHTML, RDF, et des formats basés sur XML tels que SVG.
- Standardisation
 - Possibilité d'échange de bases de données (sous réserve que le même vocabulaire contrôlé soit utilisé dans Subject et d'autres termes)
 - Possibilité d'interroger plusieurs bases à partir d'un seul outil.

Chapitre IV

RDF-Schema

- 1 Présentation
- 2 Classes
- 3 Propriétés
- 4 Compléments
- 5 Exemple : Dublin Core
- 6 Exemple : FOAF

FOAF : Présentation

FOAF (<http://www.foaf-project.org>) (Friend Of A Friend) permet d'exprimer des informations personnelles et des relations (avec d'autres personnes, des groupes, etc.) afin de rechercher de quelles communautés « virtuelles » une personne fait partie : « Réseaux sociaux ».

Comment trouver des personnes qui font partie d'une même communauté (sans FOAF) ?

Moteur de recherche, en espérant que les personnes cherchées possèdent une page web décrivant des centres d'intérêts.

Principe. Créer un document FOAF (RDF défini par un RDF-Schema (+ OWL)), hébergé sur son site web pour « se décrire ».

FOAF : Présentation

Des outils peuvent « agréger » les données contenues dans plusieurs documents FOAF pour faire des traitements :

- Filtrer les messages de collègues faisant partie d'une communauté, classer automatiquement des messages.
- Trouver des personnes qui ont des intérêts communs.
- Donner plus d'importance aux avis laissés sur des sites par des personnes ayant le même profil.
- etc.

→ Une adaptation au web sémantique de la page personnelle.

Vocabulaire

Vocabulaire FOAF : <http://xmlns.com/foaf/0.1>

Exemple

```
<rdf:RDF xmlns:rdf=
  "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:foaf="http://xmlns.com/foaf/0.1/">
  <foaf:Person>
    <foaf:name>Serge Karamazov</foaf:name>
    <foaf:gender>Male</foaf:gender>
    <foaf:title>Mr</foaf:title>
    <foaf:givenname>Serge</foaf:givenname>
    <foaf:family_name>Karamazov</foaf:family_name>
    <foaf:mbox rdf:resource="mailto:s.kara@free.fr"/>
    <foaf:homepage rdf:resource=
      "http://www.karamazov.fr"/>
  </foaf:Person>
</rdf:RDF>
```

Quelle est l'URI de la ressource Person ?

Vocabulaire

FOAF a fait le choix de ne pas attribuer l'identificateur aux personnes.

... mais utilise mbox, homepage, name pour faire des traitements.

Exemple

```
<foaf:Person>
  <foaf:mbox rdf:resource="mailto:s.kara@free.fr"/>
  <foaf:nick>Kara</foaf:nick>
</foaf:Person>
```

Ajout d'une donnée sur la même personne.

Liens

Exemple

```
<foaf:Person>
  <foaf:mbox rdf:resource="mailto:s.kara@free.fr"/>
  <foaf:depicts rdf:resource=
    "http://www.karamazov.fr/moi.png"/>
  <foaf:knows>
    <foaf:Person>
      <foaf:mbox rdf:resource="oderay@wanadoo.fr"/>
      <foaf:name>Odile Deray</foaf:name>
    </foaf:Person>
  </foaf:knows>
</foaf:Person>
```

D'autres propriétés

- Appartenance (member) à un groupe (Group).
- Localisation géographique (based_near).
- Centre d'intérêt (interest)
- Création (made / maker)
- École (schoolHomepage)
- ...

Extrait du schema

```
<rdfs:Class rdf:about="http://xmlns.com/foaf/0.1/Person"
  rdfs:label="Person" rdfs:comment="A person.">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
  <rdfs:subClassOf>
    <owl:Class rdf:about="http://xmlns.com/wordnet/1.6/Person"/>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Class rdf:about="http://xmlns.com/wordnet/1.6/Agent"/>
  </rdfs:subClassOf>
  <rdfs:isDefinedBy rdf:resource="http://xmlns.com/foaf/0.1/" />
</rdfs:Class>
```

Extrait du schema

```

<rdf:Property rdf:about="http://xmlns.com/foaf/0.1/firstName"
  vs:term_status="testing" rdfs:label="firstName"
  rdfs:comment="The first name of a person.">
  <rdf:type
    rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/
  <rdfs:domain rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
  <rdfs:range
    rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
  <rdfs:isDefinedBy rdf:resource="http://xmlns.com/foaf/0.1/">
</rdf:Property>

```

Exemple

```

<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:foaf="http://xmlns.com/foaf/0.1/">
  <foaf:Person>
    <foaf:name>Leigh Dodds</foaf:name>
    <foaf:firstName>Leigh</foaf:firstName>
    <foaf:surname>Dodds</foaf:surname>
    <foaf:mbox_sha1sum>71b88e951cb5f07518d69e5bb49a45100fbc3ca5
    </foaf:mbox_sha1sum>
    <foaf:knows rdf:resource="#dan">
  </foaf:Person>
  <foaf:Person rdf:ID="dan">
    <foaf:name>Dan Brickley</foaf:name>
    <foaf:mbox_sha1sum>241021fb0e6289f92815fc210f9e9137262c252e
    </foaf:mbox_sha1sum>
    <rdfs:seeAlso
      rdf:resource="http://rdfweb.org/people/danbri/foaf.rdf"/>
  </foaf:Person>
</rdf:RDF>

```