Introduction à RDF (Resource Description Framework)

Bernard ESPINASSE

Aix-Marseille Université
LIS UMR CNRS 7020

Septembre 2019

- **Introduction à RDF**
- **Sérialisation**
- **Agrégations : Container et collections**
- **Sémantique, inférences et vocabulaire**

Références

- **Livres, articles et rapports :**
  - O. Corby and F. Gandon and C. Faron-Zucker, Le Web sémantique : comment lier les données et les schémas sur le web ? Dunod, 2012.
  - G. Antoniou, Van Harmelen F., A Semantic Web Primer, The MIT Press Cambridge, Massachusetts London, England, 1999.
  - John Hebeler and Matthew Fisher and Ryan Blace and Andrew Perez-Lopez and Mike Dean, Semantic Web Programming, Wiley, 2009.
- **Web W3C :**
  - Page du W3C : http://www.w3.org/2004/OWL/
  - Référence : http://www.w3.org/TR/owl-ref/
  - Guide : http://www.w3.org/TR/owl-guide/
  - …
- **Cours/tutoriaux :**
  - Cours de M. Gagnon, Ecole Polytechnique de Montréal, 2007.
  - Cours de S. Staab, ISWeb – « Semantic Web », Univ. de Koblenz-Landau.
  - Cours de A.-C. Caron, Université de Lille, 2015.
  - Cours de O. Papini, Aix-Marseille Université, 2014.
  - Cours de I-M. Bilasco, Université de Lille, 2018.
  - …

Plan

1. Introduction à RDF

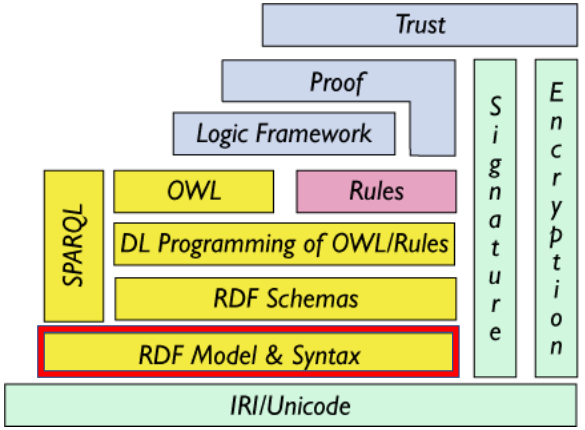
- Vers un Web plus structuré
- Un modèle pour représenter les métadonnées
- Graphe RDF
- Identification des entités/ressources – IRI
- Littéraux et nœuds blancs

Vers un Web plus structuré

- **Objectif général :**
  - **Rendre explicites les relations sémantiques** qui existent entre les différentes **ressources** qui constituent le Web
  - Définir un **mécanisme pour décrire ces ressources** par des descripteurs sémantiques (métadonnées) :
  - Pour **faciliter l'échange** et le traitement automatique de l'information sans faire d'hypothèses sur un domaine particulier d'application

⇨ **Objectif de RDF (*Ressource Description Framework*)**

Place de RDF dans le gâteau du WS …



Source : W3C, T Berners-Lee, Ivan Herman

Introduction à RDF

- **RDF** pour ***Ressource Description Framework***
  - Base du « Web des Données »
  - Proposé par W3C (World Wide Web Consortium) : RDF 1 (2004), RDF 1.1 (2014)

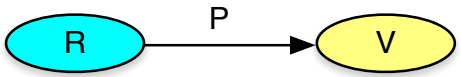
Objectif de RDF : attacher à une **ressource** un ensemble de **propriétés (métadonnées)** qui la **caractérise** au mieux et les **partager**

- Pour cela RDF propose :
  - un **modèle de données standardisé**
  - des **formats d'échanges de données** (N3, Turtle, N-Triples, …)
- **Documents de références W3C** (https://www.w3.org/standards/techs/rdf#w3c_all) :
  - *RDF Schema 1.1 (2014) : connaissances de base du RDF*
  - *Concepts and abstract syntax (2014) : syntaxe abstraite définissant le RDF*
  - *Semantics (2004) : Interprétation logique du RDF*
  - *A Datatype for RDF Plain Literals (2012) : Définition d'un type de donnée pour les chaînes de caractères (langue, taille, etc.)*
  - *Linked Data Platform 1.0 (2015) : recommandations et bonnes pratiques pour utiliser le RDF (comment nommer les URI, comment répondre à des requête HTTP, etc.)*
  - *Divers documents sur les différents formats de sérialisation (2014) : XML, Turtle, N-Triples, N-Quads, Grids, …*

Un modèle pour représenter les métadonnées

- **3 concepts de base :**
  - **Ressource** (*subject* – ***Sujet***) à décrire (ressource Web ou objet du monde réel)
    - Page Web, fragment de page, image, vidéo, tout ce qui a une URI
  - **Propriété** (*predicate* - ***Prédicat***)
    - une caractéristique, un attribut, ou
    - une relation spécifique entre ressources
  - **Valeur** (*object* - ***Objet***) de la propriété (ressources ou littéraux)
- **Un modèle de triplets (R, P, V)**
  - Une déclaration RDF est un triplet (R, P, V) :

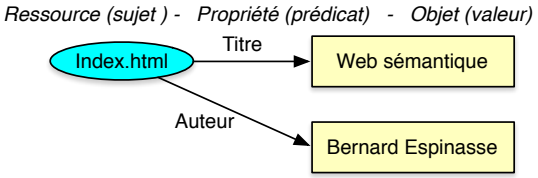
*« la ressource **R** a la valeur **V** pour la propriété **P** »*



- **Un graphe RDF** est un ensemble de déclarations RDF

Exemple 1

- **Une description** c'est un ensemble de déclarations relatives à une même ressource :



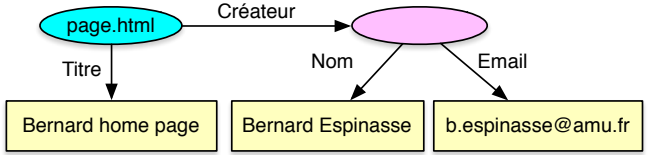
(index.html , Titre, 'Web sémantique')

(index.html , Auteur, 'Bernard Espinasse')

- La *ressource* **index.html** a comme *valeur* pour la *propriété* **titre 'Web sémantique'**
- La *ressource* **index.html** a comme *valeur* pour la *propriété* **auteur 'Bernard Espinasse'**

Exemple 2

- Dans une description, une **valeur** peut être :
  - une valeur ***littérale*** (rectangle) : une chaîne de caractères ou
  - une autre ***ressource*** (un cercle) pouvant être décrit par des littéraux



Ici la **personne** dont le **nom** Bernard Espinasse, **l'email** b.espinasse@amu.fr est le **créateur** de la page page.html dont le **titre** est Bernard home page

- Ainsi la **valeur** de la **propriété** Créateur est ***structurée***, on la représente comme une **ressource**
- **De façon générale on dira qu'un triplet est de la forme :**

(Sujet, Prédicat, Objet)

Graphe RDF

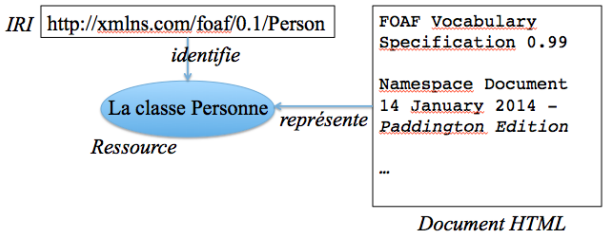
- Les **(méta)données** sont **représentées** sous forme de **graphe orienté**
- Un **graphe RDF** est un ensemble de **triplets** (Sujet, Prédicat, Objet) dans lequel :
  - Sujets et Objets = **nœuds**
  - Prédicats = **arcs**
- Les **nœuds** et **arcs** sont identifiés par des **URI** (Uniform Resource Identifier)
- Les **nœuds** peuvent aussi être :
  - des *nombres réels*
  - des *chaines de caractères*
  - des *dates*
  - …
- Quelques **problèmes à résoudre** dans les **graphes RDF** :
  - Comment gérer les *identifiants à l'échelle du Web ?*
  - Comment savoir ce que représente une *valeur*, c'est-à-dire son type, l'unité, la langue, ... ?
  - Comment savoir quelles *propriétés* (quels prédicats) utiliser ?
  - Comment *publier, échanger, interroger* les données ?

Graphe RDF et base de données RDF

- Un **graphe RDF** est un ensemble de **triplets** (Sujet, Prédicat, Objet) :
  - le **Sujet** est une *entité* représentée par un identifiant,
  - le **Prédicat** est une *propriété* de l'entité,
  - l'**Objet** est la *valeur* de la propriété pour ce sujet : cela peut être une *entité* ou un *littéral*.
- **Attention !**
  - RDF n'est pas une base de données, c'est un **format**.
  - Les données RDF peuvent être stockées dans des *BD orientées graphe* ou converties vers des *BD relationnelles*.
- **Remarques :**
  - Un document RDF qui décrit plusieurs graphes est appelé un **Dataset**
  - Une base de données capable de gérer des données RDF = **triple store.**

Identification des entités/ressources : IRI

- En RDF, les **entités** sont appelées également **ressources**, chacune est identifiée par un **IRI - International Resource Identifier** (chaine de caractères Unicode)
- Allocation d'IRI : processus d'association d'un IRI à une ressource que l'on appelle son référent
- Eviter qu'un même IRI ait 2 référents
- Référent d'un IRI fournit une représentation de ce référent sous la forme d'un document Web, auquel on accède en interprétant l'IRI comme un URL (IRI déréférençable**)**



Les littéraux

- Les **noeuds** du graphe sont en général des **entités**, identifiées par des **IRIs**
- Parfois le **nœud** à l'extrémité d'un arc est une simple **valeur** (un nombre, une date, ...), dans ce cas on utilise des **littéraux**.
- Un **littéral** consiste en 2 ou 3 éléments :
  - 1. ***une forme lexicale*** : une chaine de caractères Unicode.
  - 2. ***un IRI pour le type de données*** : cela permet de savoir comment interpréter la chaine de caractère.
  - 3. Quand le type de données est rdf:langString on associe un ***tag de langage*** au littéral
- Si on ne donne pas d'IRI de typage, alors le type par défaut est xsd:string, sauf s'il y a un tag de langage (le type est alors rdf:langString).
- Exemple :

  "1990-07-04"^^xsd:date

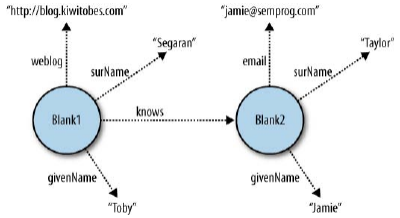
  "La Joconde"@fr
- Une fois interprétés, ces éléments **permettent d'associer une valeur** à la forme syntaxique du **littéral**

Nœuds blancs (1)

- Les **entités** (IRIs) et les **littéraux** sont suffisant pour décrire un graphe RDF
- Il se peut que l'on ne dispose pas d'IRI pour la ressource dont on souhaite parler. Par exemple, dans les réseaux sociaux, on ne dispose pas d'IRI pour les membres.
- On crée alors un **noeud blanc** (ou noeud vide, noeud anonyme, "blank node").
- Ces noeuds blancs peuvent être vus comme des variables

Parfois le nœud à l'extrémité d'un arc est une simple valeur (un nombre, une date, ...), on utilise alors des littéraux.

Exemple : Utilisation de nœuds blancs dans un réseau social :

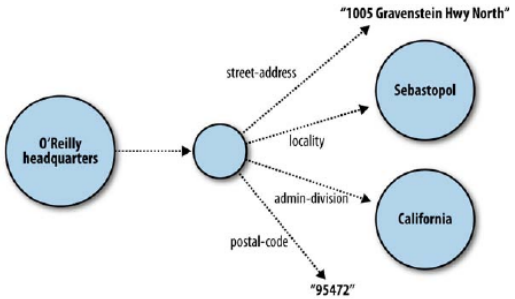


Une personne de nom Toby Segaran auteur du blog kiwitobes connait une personne nommée Jamie Taylor dont l'email est jamie@semprog.com

Nœuds blancs (2)

Les **noeuds vides** ou **nœud blanc** sont également utilisés pour grouper des informations, correspondant à l'expression d'une relation n-aire

Exemple : utilisation d'un nœud blanc pour modéliser une adresse



Réification (1)

- En informatique, la **réification** = **transformer un concept en un objet informatique**
  - Par exemple : avec un langage orienté objet avec mécanisme de réflexion, on peut **réifier une classe, qui devient instance d'une (méta-)classe**.
- En RDF, la réification permet de **considérer un triplet comme un noeud**.
- Exemple en RDF :
  - Soit le graphe G1 :

    <ex:un_sujet> <ex:une_propriete> <ex:un_objet> .
  - Le graphe G2 ci-dessous est une réification de G1 :

    _:xxx rdf:type rdf:Statement .

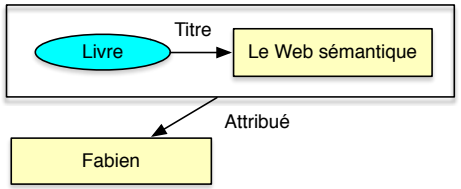
    _:xxx rdf:subject <ex:un_sujet> .

    _:xxx rdf:predicate <ex:une_propriete> .

    _:xxx rdf:object <ex:un_objet> .

Réification (2)

Graphe RDF :



Triplés associés :

- Triplet 1 : (#declaration, rdf:subjet, Livre)
- Triplet 2 : (#declaration, rdf:predicate, Titre)
- Triplet 3 : (#declaration, rdf:objet, "Le Web sémantique")
- Triplet 4 : (#declaration, Attribué, #Fabien)
- Triplet 5 : (#declaration, rdf:type, rdf:statement)

Datatypes RDF (1)

- Les littéraux standards sont des chaînes de caractères
- Pour typer les valeurs litérales, RDF repose sur les **datatypes** de XML Schema (xmlns:xsd='http://www.w3.org/2001/XMLSchema#')
- **Ces datatypes sont :**
  - xsd:integer
  - xsd:float
  - xsd:string
  - xsd:dateTime
  - xsd:boolean
  - rdf:XMLLiteral rdfs:Literal

Datatypes RDF (2)

- **Chaque littéral porte son datatype :**

  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >

  <c:Person>

  <c:age rdf:datatype='&xsd;integer'>43 </c:age>

  </c:Person>
- **Notation en triplet :**

  c:id1 c:age "43"^^xsd:integer
- **Exemple :**

  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >

  <c:Person>

  <c:age rdf:datatype='&xsd;integer'>43 </c:age>

  <c:date rdf:datatype='&xsd;dateTime'>2004-01-05 </c:date>

  <c:name rdf:datatype='&xsd;string'>Laurent </c:name>

  </c:Person>
- **Remarque :**

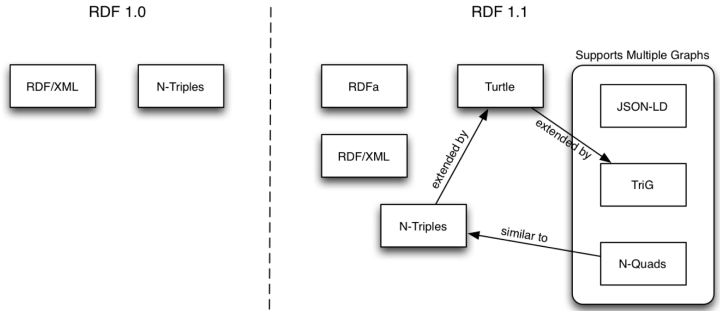
  "Bernard" ⇔ "Bernard"^^xsd:string

2. Sérialisations RDF

- Sérialisation RDF/XML
- Sérialisation N-Triples
- Sérialisation Turtle

Sérialisation de RDF (1)

- Pour échanger des données, on a besoin de **sérialiser**
- Il existe de nombreux **formats de sérialisation** :



Les plus connus :

- **RDF/XML**
- **N-Triples**
- **Turtle**

On peut passer de l'un à l'autre (http://www.easyrdf.org/converter)

Sérialisation de RDF (2)

Les plus connues :

- **RDF/XML**
  - norme RDF/XML : syntaxe XML pour représenter un graphe RDF
  - élément Description pour décrire une ressource attribut about pour le sujet,
  - sous-élément pour la propriété, contenu du sous-élément pour la propriété (qui peut être parfois simplifié en attribut)
  - on peut regrouper dans un même élément Description toutes les propriétés dont cette ressource est sujet.
    - ➢ *Difficile à lire par un humain, réservé à la machine*
- **N-Triples**
  - sérialisation sous forme de triplets
  - chaque triplet est écrit sous la forme :

    <IRI du sujet> <IRI du prédicat> <IRI de l'objet ou littéral>

    Ex : <http://example.org/bob#me> <http://xmlns.com/foaf/0.1/knows> <http://example.org/alice#me>
    - ➢ *Plus facile à lire par un humain,*
- **Turtle** (*Terse RDF Triple Language*)
  - dérivé de N-Triples, plus concis, avec des facilités syntaxiques pour rendre le code plus lisible
    - ➢ *Plus concis, plus facile encore à lire par un humain*
- ***TriG***
  - *Extension de Turtle pour utiliser plusieurs graphes*

Sérialisation RDF/XML

***Exemple** : Le cours LIS125 est assuré par un professeur dont la page web est http://www.lis-amu.fr/~espinasse et le nom est bernard espinasse*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE rdf:RDF [
   <!ENTITY udem "http://www.univ-amu.fr/">
]>
<?oxygen RNGSchema="rdfxml.rnc" type="compact"?>
<rdf:RDF 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/"
         xmlns:udem="&udem;">
<!-- Description graphe RDF -->
   <rdf:Description rdf:about="&udem;cours/LIS125">
      <udem:professeur>
         <rdf:Description>
            <foaf:homepage>
               <rdf:Description rdf:about="http://www.lis.amu.fr/~espinasse"/>
            </foaf:homepage>
         </rdf:Description>
      </udem:professeur>
   </rdf:Description>
<rdf:Description rdf:about="&udem;cours/LIS125">
      <udem:professeur>
         <rdf:Description>
            <foaf:name>bernard espinasse</foaf:name>
         </rdf:Description>
      </udem:professeur>
   </rdf:Description>
</rdf:RDF>
```

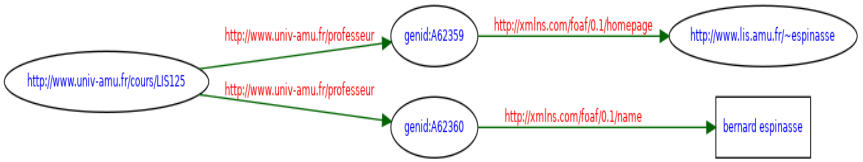
Sérialisation : Triplés et Graphe RDF

***Exemple** : Le cours LIS125 est assuré par un professeur dont la page web est http://www.lis-amu.fr/~espinasse et le nom est bernard espinasse*

Triplets (utilisation de l'outil Validator : https://www.w3.org/RDF/Validator/rdfval):

Number	Subject	Predicate	Object
1	http://www.univ-amu.fr/cours/LIS125	http://www.univ-amu.fr/professeur	genid:A62359
2	genid:A62359	http://xmlns.com/foaf/0.1/homepage	http://www.lis.amu.fr/~espinasse
3	http://www.univ-amu.fr/cours/LIS125	http://www.univ-amu.fr/professeur	genid:A62360
4	genid:A62360	http://xmlns.com/foaf/0.1/name	"bernard espinasse"

Graphe RDF (utilisation de l'outil Validator : https://www.w3.org/RDF/Validator/rdfval):



Sérialisation N-Triple

Utilisation de l'outil Easyrdf (http://www.easyrdf.org/converter):

<http://www.univ-amu.fr/cours/LIS125> <http://www.univ-amu.fr/professeur> _:genid1 .

<http://www.univ-amu.fr/cours/LIS125> http://www.univ-amu.fr/professeur _:genid2 .

_:genid1 <http://xmlns.com/foaf/0.1/homepage> <http://www.lis.amu.fr/~espinasse> .

_:genid2 <http://xmlns.com/foaf/0.1/name> "bernard espinasse" .

Sérialisation Turtle

Utilisation de l'outil Easyrdf (http://www.easyrdf.org/converter):

@prefix ns0: <http://www.univ-amu.fr/> .

@prefix foaf: <http://xmlns.com/foaf/0.1/> .

<http://www.univ-amu.fr/cours/LIS125> ns0:professeur [foaf:homepage <http://www.lis.amu.fr/~espinasse>], [foaf:name "bernard espinasse"] .

3. Agrégations en RDF

- Containers
- Collections

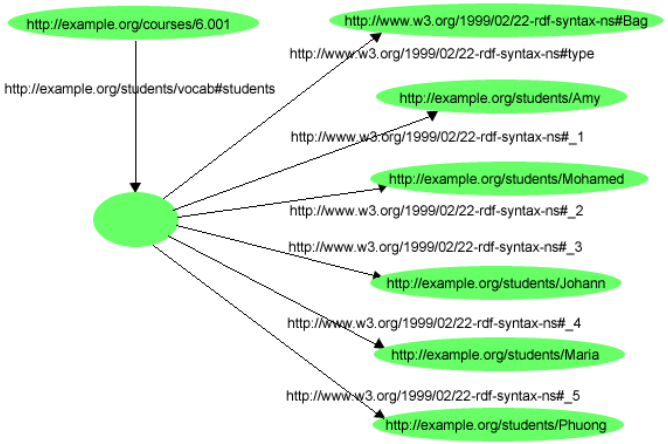
Container

- Il est fréquent de devoir faire référence à plusieurs ressources (par exemple : un livre écrit par plusieurs auteurs).
- Les **containers** permettent de décrire des groupes
- Les choses contenues dans un container sont appelées « membres du groupe »
- Il existe 3 types de containers prédéfinis :
  - **rdf:Bag** : ***liste non ordonnée*** de ressources ou de littéraux - **multi-ensemble** de ressources ou littéraux
  - **rdf:Seq** : ***liste ordonnée*** de ressources ou de littéraux
  - **rdf:Alt** : liste de ressources ou de littéraux qui ***représentent des alternatives*** pour une valeur unique
- Pour indiquer qu'une ressource est un container, on utilise la propriété **rdf:type**
- Remarques :
  - Les containers sont **ouverts**, i.e. il peut exister d'autres membres du container que ceux indiqués par la description dont on dispose (qui s'y rajoutent …)
  - Les éléments du conteneur sont listés à l'aide de l'élément ***li*** ou ***_i*** avec (i > 0)
  - Pour le container ***alternative*** il doit y avoir au moins un élément ***_1*** (élément par défaut)

Containers: Bag

(Source RDF Primer)

Soit le graphe RDF suivant précisant les étudiants du cours 6.001 :



Containers : Bag

- On peut spécifier ce graphe RDF en XML/RDF :

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
        xmlns:s="http://example.org/students/vocab#">
 <rdf:Description rdf:about="http://example.org/courses/6.001">
  <s:students>
   <rdf:Bag>
    <rdf:li rdf:resource="http://example.org/students/Amy"/>
    <rdf:li rdf:resource ="http://example.org/students/Mohamed"/>
    <rdf:li rdf:resource ="http://example.org/students/Johann"/>
    <rdf:li rdf:resource ="http://example.org/students/Maria"/>
    <rdf:li rdf:resource ="http://example.org/students/Phuong"/>
   </rdf:Bag>
  </s:students>
 </rdf:Description>
</rdf:RDF>
```

- On peut spécifier ce graphe RDF en Turtle :

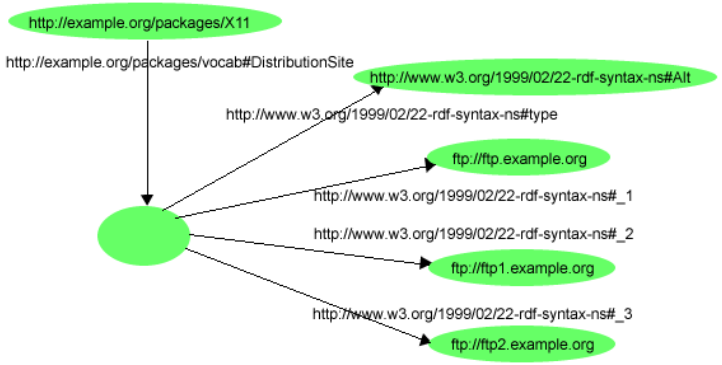
```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix s:  <http://example.org/students/vocab#>.
@prefix stu: <http://example.org/students/>.
@prefix crs: <http://example.org/courses/>.
crs:6.001
   s:students [
     a rdf:Bag;
     rdf:_1 stu:Amy;
     rdf:_2 stu:Mohamed;
     rdf:_3 stu:Johann;
     rdf:_4 stu:Maria;
     rdf:_5 stu:Phuong;
   ].
```

```
select ?student where {
          crs:6.001 s:students [
                rdfs:member ?student;
          ]
}
```


Containers : Alt

(Source RDF Primer)

Soit le graphe RDF suivant précisant les sites alternatifs pour le téléchargement de X11 :



Containers : Alt

- On peut spécifier ce graphe RDF en XML/RDF :

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:s="http://example.org/packages/vocab#">
  <rdf:Description rdf:about="http://example.org/packages/X11">
    <s:DistributionSite>
      <rdf:Alt>
        <rdf:li rdf:resource="ftp://ftp.example.org"/>
        <rdf:li rdf:resource="ftp://ftp1.example.org"/>
        <rdf:li rdf:resource="ftp://ftp2.example.org"/>
      </rdf:Alt>
    </s:DistributionSite>
  </rdf:Description>
</rdf:RDF>
```

- On peut spécifier ce graphe RDF en Turtle :

```
pkg:X11
   s:DistributionSite [
     a rdf:Alt;
     rdf:_1 <ftp://ftp.example.org>;
     rdf:_2 <ftp://ftp1.example.org>;
     rdf:_3 <ftp://ftp2.example.org>;
   ].
```

- Interrogation du Alt en SPARQL sur Turtle :

```
select ?coll ?type ?elem where {
  ?coll ?pred [a ?type;
               rdfs:member ?elem]
}
```

Containers : Seq

- Pour un conteneur rdf:**Seq**, les graphe RDF et les spécifications RDF/XML sont similaires à ceux d'un conteneur rdf:**Bag**, la seule différence réside dans le type rdf:**Seq**.
- Remarque : bien qu'un conteneur rdf:**Seq** soit destiné à décrire une séquence, c'est aux applications qui créent et traitent le graphe d'interpréter correctement la séquence des noms de propriétés à valeur entière.
- En XML/RDF :

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:s="http://example.org/packages/vocab#">
  <rdf:Description rdf:about="http://example.org/packages/X11">
    <s:DistributionSite>
      <rdf:Seq>
        <rdf:li rdf:resource="ftp://ftp.example.org"/>
        <rdf:li rdf:resource="ftp://ftp1.example.org"/>
        <rdf:li rdf:resource="ftp://ftp2.example.org"/>
      </rdf:Seq>
    </s:DistributionSite>
  </rdf:Description>
</rdf:RDF>
```

- En RDF Turtle :

```
pkg:X11
   s:DistributionSite [
      a rdf:Seq;
      rdf:_1 <ftp://ftp.example.org>;
      rdf:_2 <ftp://ftp1.example.org>;
      rdf:_3 <ftp://ftp2.example.org>;
    ].
```


Collections (1)

- En RDF, une **collection** est une **liste** (comme dans LISP) :
- Une liste de type rdf:List avec
  - un premier élément rdf:first et
  - une suite rdf:rest.
- La liste vide a la valeur rdf:nil.
- Exemple :

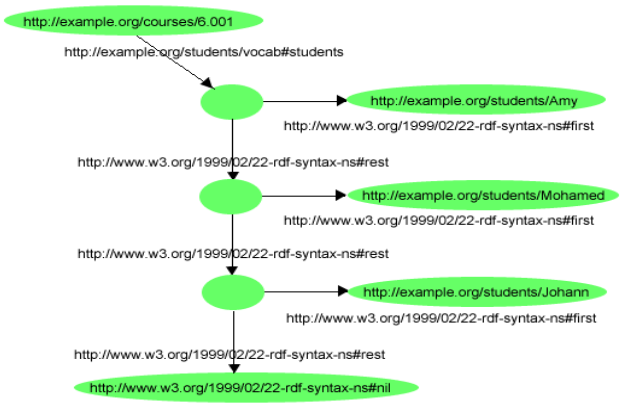
```
_:c1 rdf:first <ex:aaa> .
_:c1 rdf:rest _:c2 .
_:c2 rdf:first <ex:bbb> .
_:c2 rdf:rest rdf:nil .
```

- Une collection est une liste **fermée**
- Une collection forme un **groupe** contenant **que les membres spécifiés lors de la déclaration** de la collection.

Collections (2)

(Source RDF Primer)

Soit le graphe RDF suivant associé au cours 6.001 et ses étudiants :



Collections (3)

- On peut spécifier cette collection en XML/RDF ainsi :

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:s="http://example.org/students/vocab#">
  <rdf:Description rdf:about="http://example.org/courses/6.001">
    <s:students rdf:parseType="Collection">
        <rdf:Description rdf:about="http://example.org/students/Amy"/>
        <rdf:Description rdf:about="http://example.org/students/Mohamed"/>
        <rdf:Description rdf:about="http://example.org/students/Johann"/>
    </s:students>
  </rdf:Description>
</rdf:RDF>
```

- On peut spécifier cette collection pour le cours 6.001 (liste1-L1) en Turtle ainsi :

```
crs:6.001
   s:students (
      stu:AmyL1
      stu:MohamedL1
      stu:JohannL1
).
```

```
select ?cours ?student where {
      ?cours s:students/rdf:rest*/rdf:first ?student
}
```

Ou pour le cours 6.0002 (liste2-L2) :

```
crs:6.002 s:students _:v0.
_:v0
   rdf:first stu:AmiL2;
   rdf:rest _:v1.
_:v1
   rdf:first stu:MohamedL2;
   rdf:rest _:v2.
_:v2
   rdf:first stu:JohannL2;
   rdf:rest _:nil.
```

Collections (4): exemple

- Collection : quand la valeur d'une propriété est une collection de valeurs :
- Exemple : soit le cours LIS125 :

```
<rdf:Description rdf:about='http://www.amu.fr/cours/LIS125'>
    <ns:staff>
        <rdf:Bag>
            <rdf:li>Bernard Espinasse</rdf:li>
            <rdf:li>Sébastien Fournier</rdf:li>
            <rdf:li>Adrian Chifu</rdf:li>
        </rdf:Bag>
    </ns:staff>
</rdf:Description>
```

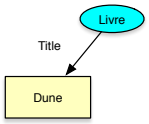

4. Exemple d'une base RDF

- **Description d'un livre et de ses auteurs**

Description RDF d'un livre et son auteur : XML

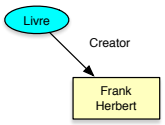
- La *ressource* « Livre » a pour *valeur* pour la *propriété* « Titre » le littéral ou valeur « Dune » ce qui donne le triplet RDF (Livre, Title, Dune), en XML :

```
<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="Livre">
        <dc:Title>Dune</dc:Title>
    </rdf:Description>
</rdf:RDF>
```



- Une autre *propriété* « Auteur » peut être associée à la *ressource* Livre, avec la valeur « Frank Herbert », on a le triplet (Livre, Creator, Frank Herbert) :

```
<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="Livre">
        <dc:Title>Dune</dc:Title>
        <dc:Creator>Frank Herbert</dc:Creator>
    </rdf:Description>
```



Description RDF d'un livre et son auteur

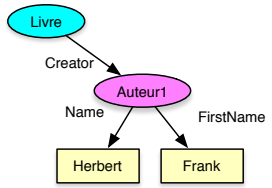
Pour distinguer dans la valeur de l'auteur « Frank Herbert » le nom du prénom :

- On définie la propriété « Name » et la propriété « FirstName »
- On créé un noeud blanc (anonyme) « #Auteur1 »
- On caractérise ce nœud anonyme par les propriétés « Name » et « FirstName »
- On a ainsi les 3 triplets RDF :

  (Livre, Creator, #Auteur1)

  (#Auteur1, Name, Herbert)

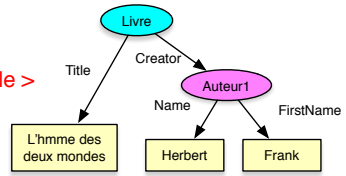
  (#Auteur1, FirstName, Frank)



Description RDF d'un livre et son auteur : XML

Soit en XML :

```
<rdf:RDF
<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://monexemple.org">
  <rdf:Description rdf:about="Livre">
     <dc:Creator rdf:resource="Auteur1"/>
     <dc:Title> L'homme des deux mondes </dc: Title >
  </rdf:Description>
  <rdf:Description rdf:ID="auteur1">
     <ex:FirstName> Frank </ex: FirstName >
     <ex:Name> Herbert </ex:Name>
  </rdf:Description>
</rdf:RDF>
```

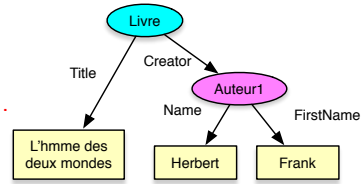


Description RDF d'un livre et son auteur : Turtle

Utilisation de EasyRDF (http://www.easyrdf.org/converter):

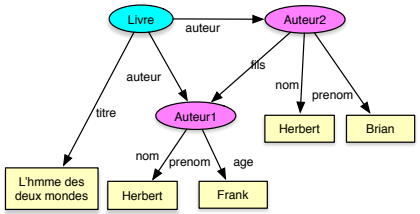
Soit en Turtle :

```
@prefix dc11: <http://purl.org/dc/elements/1.1/> .
@prefix ns0: <http://monexemple.> .

<http://njh.me/Livre>
  dc11:Creator <http://njh.me/Auteur1> ;
  dc11:Title "L'homme des deux mondes" .

<http://njh.me/#Auteur1>
  ns0:orgFirstName "Frank" ;
  ns0:orgName "Herbert" .
```



Description RDF d'un livre et ses auteurs (2) : Triplets

Considérons le livre *L'Homme de deux mondes* qui est le premier roman écrit par Frank Herbert (disparu en 1986), en collaboration avec son fils Brian :

Triplets RDF	Graphe RDF
(Livre, auteur, #Auteur1) (Livre, auteur, #Auteur2) (Livre, titre, L'homme des deux mondes) (#Auteur1, nom, Herbert) (#Auteur1, prenom, Frank) (#Auteur2, nom, Herbert) (#Auteur2, prenom, Brian) (#Auteur2, fils, #Auteur1)	

Description RDF d'un livre et ses auteurs (2) : XML

```
<?xml version="1.0" encoding="utf-8" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns:ns0="http://www.monvoc.">

 <rdf:Description rdf:about="http://#livre1">
  <ns0:frauteur rdf:resource="http://auteur1"/>
  <ns0:frtitre>Dune</ns0:frtitre>
 </rdf:Description>

 <rdf:Description rdf:about="http://#livre2">
  <ns0:frauteur rdf:resource="http://auteur1"/>
  <ns0:frauteur rdf:resource="http://auteur2"/>
  <ns0:frtitre>L'homme de deux mondes</ns0:frtitre>
 </rdf:Description>

 <rdf:Description rdf:about="http://#auteur1">
  <ns0:frprenom>Frank</ns0:frprenom>
  <ns0:frnom>Herbert</ns0:frnom>
  <ns0:frage rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">70</ns0:frage>
 </rdf:Description>

 <rdf:Description rdf:about="http://#auteur2">
  <ns0:frprenom>Brian</ns0:frprenom>
  <ns0:frnom>Herbert</ns0:frnom>
  <ns0:frage rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">30</ns0:frage>
  <ns0:frfils rdf:resource="http://auteur1"/>
 </rdf:Description>

</rdf:RDF>
```

Description RDF d'un livre et ses auteurs (2) : Turtle

Utilisation de EasyRDF (http://www.easyrdf.org/converter):

```
@prefix dc11: <http://purl.org/dc/elements/1.1/> .
@prefix ns0: <http://monexemple.> .

<http://njh.me/Livre>
  dc11:Creator <http://njh.me/Auteur1>, <http://njh.me/Auteur2> ;
  dc11:Title " L'homme des deux mondes " .

<http://njh.me/#Auteur1>
  ns0:orgFirstName " Frank " ;
  ns0:orgName " Herbert " .

<http://njh.me/#Auteur2>
  ns0:orgFirstName " Brian " ;
  ns0:orgName " Herbert " ;
  ns0:orgFils <http://njh.me/Auteur1> .
```


Description RDF d'un livre et ses auteurs (2) : Graphe

Graphe RDF associé (RDF Validator: https://www.w3.org/RDF/Validator/) :

Description RDF d'un livre et ses auteurs (2) : Triplets

Triplets RDF associés (RDF Validator: https://www.w3.org/RDF/Validator/) :

Nb	Subject	Predicate	Object
1	http://www.w3.org/RDF/Validator/run/Livre	http://purl.org/dc/elements/1.1/Creator	http://www.w3.org/RDF/Validator/run/Auteur1
2	http://www.w3.org/RDF/Validator/run/Livre	http://purl.org/dc/elements/1.1/Creator	http://www.w3.org/RDF/Validator/run/Auteur2
3	http://www.w3.org/RDF/Validator/run/Livre	http://purl.org/dc/elements/1.1/Title	"L'homme des deux mondes"
4	http://www.w3.org/RDF/Validator/run/1569169791477#Auteur1	http://monexemple.orgFirstName	"Frank"
5	http://www.w3.org/RDF/Validator/run/1569169791477#Auteur1	http://monexemple.orgName	"Herbert"
6	http://www.w3.org/RDF/Validator/run/1569169791477#Auteur2	http://monexemple.orgFirstName	"Brian"
7	http://www.w3.org/RDF/Validator/run/1569169791477#Auteur2	http://monexemple.orgName	"Herbert"
8	http://www.w3.org/RDF/Validator/run/1569169791477#Auteur2	http://monexemple.orgFils	http://www.w3.org/RDF/Validator/run/Auteur1

Description RDF d'un livre et ses auteurs (2) : simplifications

```
<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://monexemple.org.">
  <rdf:Description rdf:about="Livre">
     <dc:auteur rdf:resource="Auteur1"/>
     <dc:auteur rdf:resource="Auteur2"/>
     <dc:titre> L'homme des deux mondes </dc:titre>
  </rdf:Description>
  <rdf:Description rdf:ID="Auteur1">
     <ex:prenom> Frank </ex:prenom>
     <ex:nom> Herbert </ex:nom>
  </rdf:Description>
  <rdf:Description rdf:ID="Auteur2">
     <ex:prenom> Brian </ex:prenom>
     <ex:nom> Herbert </ex:nom>
     <ex:fils rdf:ressource= "Auteur1" />
  </rdf:Description>
</rdf:RDF>
```

Description RDF d'un livre et son auteur (2) : Turtle

Utilisation de Easy RDF (http://www.easyrdf.org/converter):

```
@prefix dc11: <http://purl.org/dc/elements/1.1/> .
@prefix ns0: <http://monexemple.org.> .

<http://njh.me/Livre>
  dc11:auteur <http://njh.me/Auteur1>, <http://njh.me/Auteur2> ;
  dc11:titre " L'homme des deux mondes " .

<http://njh.me/#Auteur1>
  ns0:prenom " Frank " ;
  ns0:nom " Herbert " .

<http://njh.me/#Auteur2>
  ns0:prenom " Brian " ;
  ns0:nom " Herbert " ;
  ns0:fils <http://njh.me/Auteur1> .
```


5. Conclusion

- Sémantique de RDF
- Inférences dans RDF
- Vocabulaire RDF
- Langages de requête autour de RDF : SPARQL
- Intérêts et limitations de RDF

Sémantique de RDF

- La sémantique d'un document RDF peut être exprimée en ***théorie des ensembles*** en se donnant des contraintes sur le monde décrites en RDF
- RDF hérite alors de la ***généricité*** et de ***l'universalité*** de la notion d'ensemble
- Cette sémantique peut être aussi traduite en formule de logique du premier ordre, positive, conjonctive et existentielle :

$$\{sujet, objet, prédicat\} \Leftrightarrow prédicat(objet, sujet)$$

- ce qui est équivalent à :

$$\exists\ objet, \exists\ sujet \text{ tel que } prédicat(objet, sujet)$$

RDF et inférences

- Le W3C a prévu un ***mécanisme d'inférence*** pour la sémantique de RDF déduisant exclusivement et intégralement les conséquences des prédicats, sans que ce mécanisme ne fasse l'objet d'une recommandation.

Vocabulaires RDF

- La structure de RDF est extrêmement générique et sert de base à un certain nombre de schémas ou vocabulaires dédiés à des applications spécifiques :
  - **des vocabulaires sont spécifiés par le W3C :**
    - les langages d'ontologie RDFS et OWL
    - le vocabulaire SKOS pour la représentation des thésaurus et autres vocabulaires structurés
    - …
  - **d'autres vocabulaires RDF** pas spécifiés par le W3C, sont néanmoins utilisés largement et constituent des standards de fait dans la communauté du Web Sémantique
    - Exemple : FOAF destiné à la représentation des personnes
    - …

Intérêts et limitation de RDF

Intérêts :

- **RDF = base du Web des données** en proposant :
  - un **modèle de données standardisé**
  - des **formats d'échanges de données** (N3, Turtle, N-Triples, …)
- **Un langage de requêtes puissant : SPARQL**

Limitations :

- **RDF** permet de définir des graphes étiquetés - **Graphes RDF**, en utilisant des ressources du web **sans vraiment de sémantique**
- Pour donner de la sémantique des étiquettes du graphe RDF, il faut des **vocabulaires mieux définis (méta-données)**
- Pour formaliser cette sémantique il faut utiliser des **ontologies**

⇨ **RDFS** (RDF-Schema), extension de RDF **permettant de construire des ontologies légères en RDF**