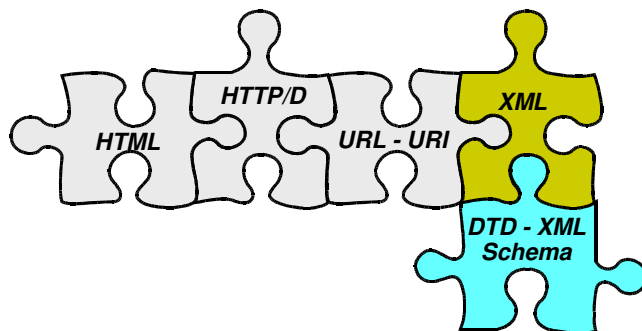


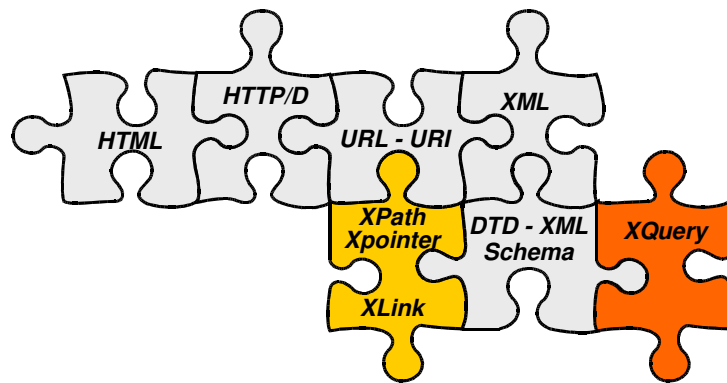
- 1998 – XML 1.0 (Extensible Markup Language)
  - représenter et non présenter (séparation entre contenu et présentation)
  - un **format textuel d'échange** de **données structurées**
  - standard pour définir des langages balisés
- Web structuré accessible comme Base de Données
- Famille de langages : MathML, CML, SVG, XMI, XHTML, XFRAMES, ...

- **DTD** (Document Type Definition)
  - une DTD définit les balises autorisées, leurs attributs et leur enchaînement
- 2004 – **XML Schéma**
  - contraintes sur structure (namespaces, éléments, attributs, cardinalité ) et contenu (datatypes et types prédéfinis, entités, notations, expressions régulières)
  - notion de type et héritage pour définitions éléments, attributs, et datatypes (extension & restriction)
  - valeurs par défaut des éléments et des attributs
  - Spécification partielle (élément / attribut quelconque)

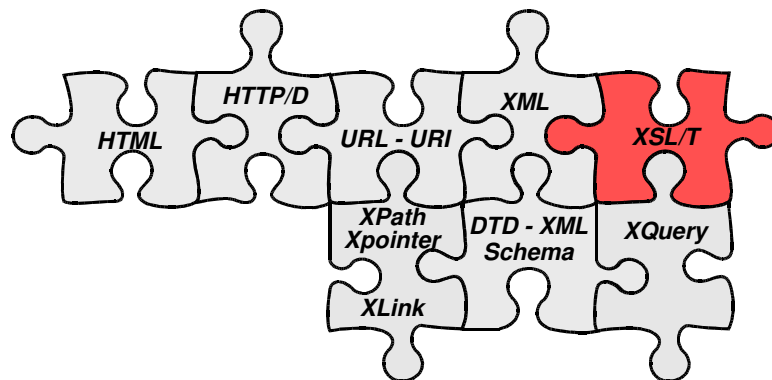


- 1999 – **XPath** 1.0 (XML Path Language)
  - description des **chemins** dans un document XML
- 2001 – **Xlink** 1.0 (XML Linking Language)
  - la **généralisation XML du concept de lien** du HTML
  - plus expressif (destinations multiples, contrôle des déclencheurs, adaptation du comportement, ...)
- 2003 – **XPointer** 1.0 (XML Pointer Language)
  - extension des URL pour pointer sur des éléments d'un document XML même si le fragment est sans ID
  - fait une connexion entre **XPath et les URL**

- 2006 – **XQuery** (XML Query Language)
  - langage de requête sur les structures XML, inspiré de SQL
  - S'appuie sur les systèmes d'adressage XPath, XLink, XPointer



- 2001 – **XSL** 1.0 (XML Stylesheet Language)
  - feuilles de style pour XML
- 1999 – **XSLT** 1.0 (XSL Transformation)
  - transformation récursive d'un arbre XML source en un arbre XML cible: tester et filtrer, modifier des valeurs, ajouter des éléments, réorganiser (trier), etc.



- Le Web est donc
  - un ensemble de ressources (données et pages)
  - bien outillé pour
    - la représentation de données structurées
    - leur restitution sous forme de page Web
    - l'adressage des ressources
    - la recherche de ressources
    - La recherche d'éléments à l'intérieur des ressources
- Mais les ressources stockées et manipulées dans le cadre du Web sont purement syntaxiques

http://www.google.fr/

Google™ France

Web Images Groupes Annuaire Actualités plus »

**Bruit ≠ Précision** ↓

**Manqué ≠ Rappel** ↓

**Agences l'RAM**

La Galère  
148, rue Victor **Hugo**  
76600 Le Havre

L'Agence de la Presse et des Livres  
38, rue Saint Dizier BP 445  
54001 Nancy Cédex

RESUME DU **ROMAN** DE  
VICTOR HUGO

*NOTRE DAME DE PARIS*  
(1831) - 5 parties

L'enlèvement . Livres 1-2 : 6 janvier  
1482. L'effrayant bossu Quasimodo

Exc

Mozilla

Fichier Edition Affichage Aller à Marque-pages Outils Fenêtre Aide

http://www.reazymread.com/sacks.html

**The Man Who Mistook His Wife for a Hat :  
And Other Clinical Tales** by Oliver W. Sacks

In his most extraordinary book, "one of the great clinical writers of the 20th century" (The New York Times) recounts the case histories of patients lost in the bizarre, apparently inescapable world of neurological disorders. Oliver Sacks's The Man Who Mistook His Wife for a Hat tells the stories of individuals afflicted with disorders that strip away their memories and distort their sense of self and their relationship to the world around them. These are people and communities whose lives are marked by involuntary obsessions and compulsions. They are gifted with unique talents and insights, but they are also suffering from a disease that is often incurable. If inconceivably strange, these brilliant tales are studies of life struggling against increasingly impaired, to imagine with our hearts what medicine's ultimate responsibility: "the suffering of the individual."

Our rating : ★★★★★

Find other books in :  Neurology

Search books by terms :

Google - Mozilla

Fichier Edition Affichage Aller à Marque-pages Outils Fenêtre Aide

http://www.google.com/

Google™

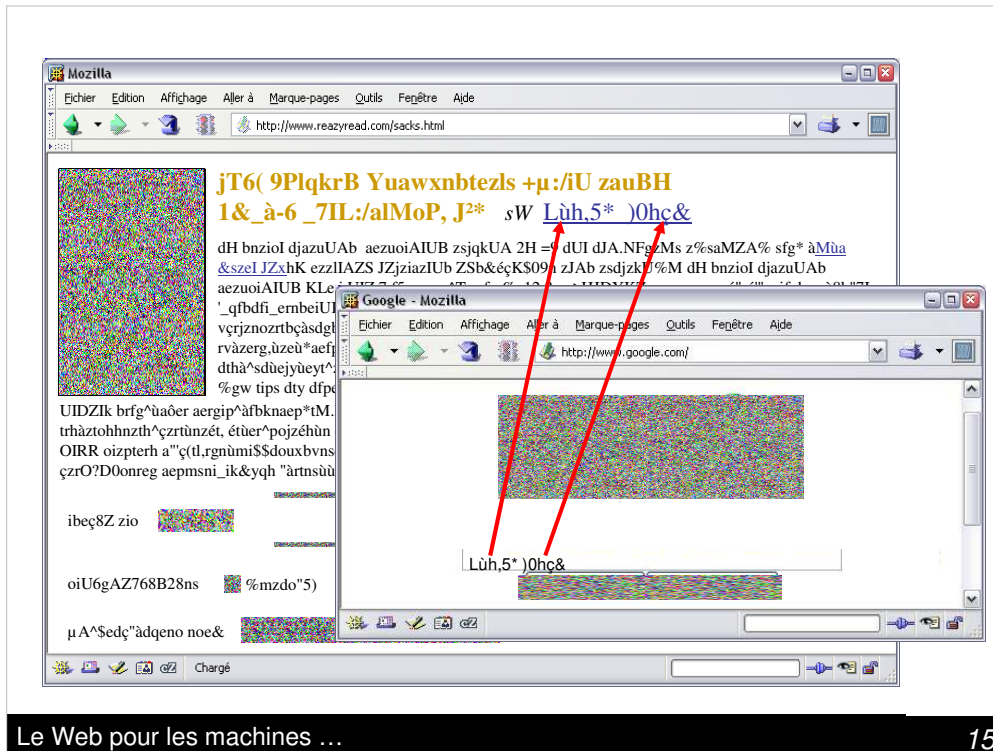
Oliver Sacks

Google Search I'm Feeling Lucky

Chargé

Le Web pour nous ...

14



- Dans le Web classique, pas d'interprétation
  - manipulation formelle, syntaxique des données
- Dans les SI « traditionnels », la sémantique n'est pas forcément nécessaire à l'interaction homme-machine
  - les informations relèvent d'un domaine borné et souvent technique
  - pas ou peu d'ambiguïté
  - système fermé à destination de spécialistes
- Le Web est un **système ouvert** à tous, couvrant **tous les domaines**

- La **masse des informations** stockées sur le Web augmente à vitesse exponentielle
  - $5 \cdot 10^6$  To (1 To =  $10^{12}$  octets) en 2005 (la Library of Congress stocke 20 To de données texte)
  - en 2005, le nombre de sites Web a augmenté de 17,5 millions
  - la production mondiale de données mémorisées double chaque année
- Le problème du Web n'est plus d'augmenter la taille des « autoroutes de l'information », mais de concevoir et réaliser des systèmes permettant de filtrer les informations et de les délivrer de façon « intelligente »

- « *The **Semantic Web** is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation* »

Tim Berners-Lee, James Hendler, Ora Lassila

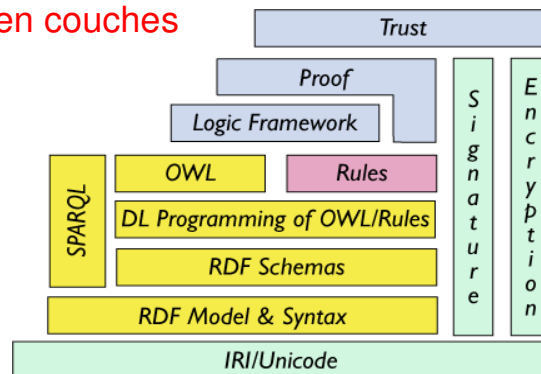
in *The Semantic Web*, Scientific American, May 2001

- Ajout d'une **couche de connaissances** au dessus du Web pour que :
  - les applications Web interprètent les données comme les utilisateurs afin d'améliorer l'interaction
  - l'interopérabilité et le dialogue entre applications soit possible



- La couche connaissance du Web sémantique contient
  - des annotations sémantiques (RDF)
  - des ressources sémantiques (ontologies en RDFS ou OWL)
  - des outils de manipulations des annotations et ontologies (SPARQL, ...)
- D'autres éléments sont en cours d'ajout
  - Des règles pour augmenter l'expressivité des ontologies OWL
  - ...

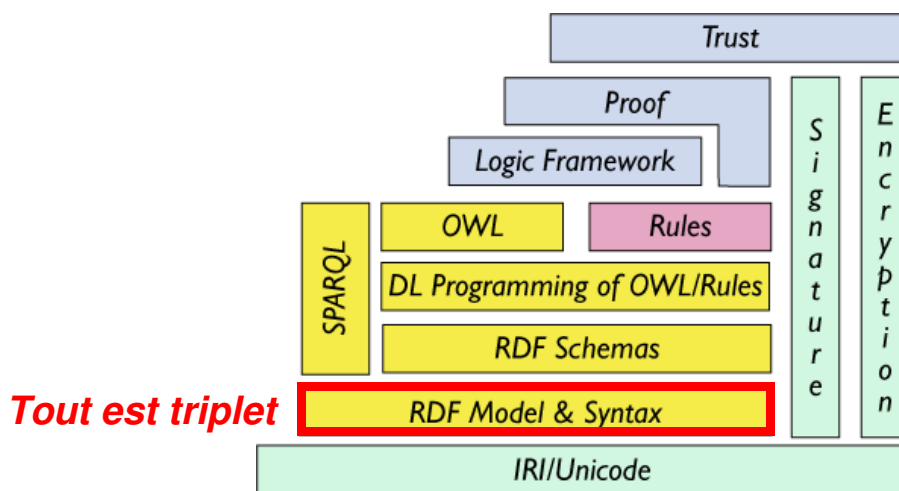
- Les 4 principaux standards du Web sémantique
  - RDF: un modèle de **triplets** pour décrire et connecter des ressources anonymes ou identifiées par un URI
  - SPARQL: un langage de requête sur les graphes RDF
  - RDFS est un langage de **descriptions légères**
  - OWL: 3 couches d'extension de l'expressivité (logique)
  - **modèle en couches**



# Resource Description Framework



21



*Tout est triplet*

W3C, T Berners-Lee, Ivan Herman

22

- 1999 – RDF (Ressource Description Framework)
  - Annotation sémantique des ressources
  - Assertion de liens entre ressources (donner du sens, c'est lier des informations ou connaissances)
  - Plus riche qu'une annotation syntaxique par mot-clé
- Triplet RDF *<entité, propriété, valeur>*
  - Décrit une entité (identifié par une URI)
  - Associe au sujet une propriété (identifiée par une URI)
  - Donne une valeur à la propriété. Une valeur est soit une ressource identifiée par une URI, soit une valeur primitive



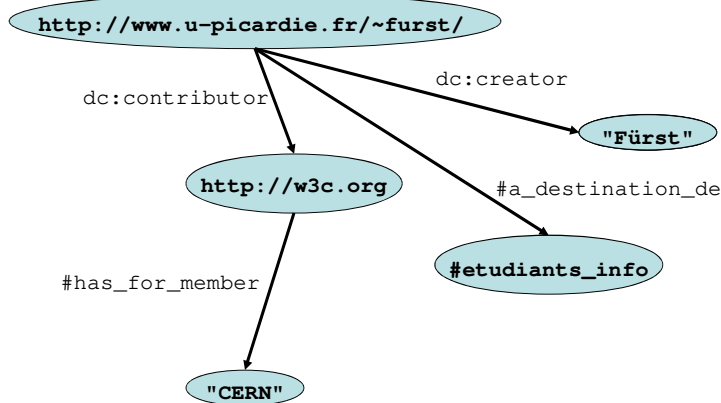
- Exemples :

```
<http://www.u-picardie.fr/~furst/ , dc:creator , "Fürst">
<http://www.u-picardie.fr/~furst/ , dc:contributor , http://w3c.org>
<http://www.u-picardie.fr/~furst/ , #a_destination_de , #etudiants_info>
<http://w3c.org , #has_for_member , "CERN">
```

- Les URL introduites par # sont définies localement
- Des prédicats sont définis dans des vocabulaires existants :
  - **dc** (Dublin Core) : schéma de métadonnées pour les documents (*dublincore.org*)
  - Il existe d'autres vocabulaires (ex. **foaf** Friend of a Friend, *www.foaf-project.org*)
  - rdf définit aussi ses propres prédicats

## Les triplets RDF forment des graphes d'entités

```
<http://www.u-picardie.fr/~furst/ , dc:creator , "Fürst">  
<http://www.u-picardie.fr/~furst/ , dc:contributor , http://w3c.org>  
<http://www.u-picardie.fr/~furst/ , #a_destination_de , #etudiants_info>  
<http://w3c.org , #has_for_member , "CERN">
```



- **Syntaxe XML** pour RDF : RDF possède une syntaxe XML (mais ne s'y réduit pas!)
- Un document RDF est une liste de descriptions incluses dans des balises **rdf:RDF**
- Une description est introduite par la balise **rdf:Description** et porte sur une entité désignée par une URI introduite par l'attribut **rdf:about**
- A la place de l'attribut **rdf:about**, on peut utiliser **rdf:ID** pour donner une URL relative

```

<http://www.u-picardie.fr/~furst/ , dc:creator , "Fürst">
<http://www.u-picardie.fr/~furst/ , dc:contributor , http://w3c.org>
<http://www.u-picardie.fr/~furst/ , #a_destination_de , #etudiants_info>
<http://w3c.org , #has_for_member , "CERN">

```

```

<?xml version="1.0"?>
<rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc = "http://dublincore.org/2008/01/14/dcelements.rdf#"
  xmlns:ex = "http://www.u-picardie.fr/~furst/exemples.rdf#">

  <rdf:Description rdf:about="http://www.u-picardie.fr/~furst/">
    <dc:creator>Fürst</dc:creator>
    <dc:contributor>
      <rdf:Description rdf:about="http://w3c.org">
        <ex:hasformember>CERN</ex:hasformember>
      </rdf:Description>
    </dc:contributor>
    <ex:a_destination_de rdf:resource="ex:etudiants_info"/>
  </rdf:Description>
</rdf:RDF>

```

## Contenu du fichier <http://www.u-picardie.fr/~furst/exemples.rdf>

```

<?xml version="1.0"?>
<rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc = "http://dublincore.org/2008/01/14/dcelements.rdf#"
  xmlns:ex = "http://www.u-picardie.fr/~furst/exemples.rdf#">
  <rdf:Property rdf:about="ex:hasformember"/>
  <rdf:Property rdf:about="ex:a_destination_de"/>
  <rdf:Description rdf:about="ex:etudiants_info">
    <dc:date>2009</dc:date>
    <dc:description>Liste des étudiants en informatique
      <rdf:Bag>
        <rdf:li>Duchnok Mathieu</rdf:li>
        <rdf:li>Tartempion Robert</rdf:li>
      </rdf:Bag>
    </dc:description>
  </rdf:Description>
</rdf:RDF>

```

- Utilisation de l'attribut **rdf:ID**
  - **rdf:about="truc#blabla"** est équivalent à **rdf:ID="blabla"** si l'URI de référence est **truc**

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc = "http://dublincore.org/2008/01/14/dcelements.rdf#"
  xmlns:ex = "http://www.u-picardie.fr/~furst/exemples.rdf#"
  xml:base = "http://www.u-picardie.fr/~furst/exemples.rdf">

  <rdf:Description rdf:ID="etudiants_info">
    <dc:creator>Furst</dc:creator>
  </rdf:Description>
</rdf:RDF>
```

## Vocabulaire XML propre à RDF

- Mots réservés

**RDF, Description, ID, about, parseType, resource, li, nodeID, datatype**

- Noms de classes

**Seq, Bag, Alt, Statement, Property, XMLLiteral, List**

- Noms de propriétés

**subject, predicate, object, type, value, first, rest, \_n** (où n est un entier plus grand que 0 et ne commençant pas par 0)

- Noms de ressources

**nil**

- Les ressources sont identifiables par des URI
  - Si l'URI est un URL alors ressource du Web
  - Si non, URI d'une ressource physique, abstraite, etc.
  - Jointure entre les assertions **même si elles sont distribuées**
  - Liant entre les couches réseau (Internet, Web, Web sémantique)
- RDF est un modèle de métadonnées ouvert
  - Vocabulaire **extensible** basé sur les URI et XML schema datatypes
  - Autorise quiconque à faire des déclarations sur n'importe quelle ressource

- Il existe un autre langage pour exprimer plus simplement des triplets RDF : le **langage N3** développé par T. Berners Lee
  - Syntaxe non XML
  - Vise une manipulation humaine

```
<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.u-picardie.fr/~furst/">
    <dc:title>Cours d'informatique</dc:title>
    <dc:creator>Furst</dc:creator>
  </rdf:Description>
</rdf:RDF>
```

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

<http://www.u-picardie.fr/~furst/>
  dc:title "Cours d'informatique";
  dc:creator "Furst".
```

- Chaque triplet représente un prédicat binaire en logique

```
<http://www.u-picardie.fr/~furst/ , dc:creator , "Fürst">
<http://www.u-picardie.fr/~furst/ , dc:contributor , http://w3c.org>
<http://www.u-picardie.fr/~furst/ , #a_destination_de , #etudiants_info>

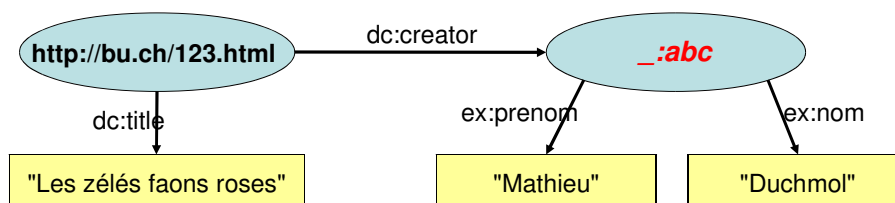
dc:creator(http://www.u-picardie.fr/~furst/, "Fürst")
=> dc:contributor(http://www.u-picardie.fr/~furst/, http://w3c.org)
#a_destination_de(http://www.u-picardie.fr/~furst/, #etudiants_info)
```

- Sémantique formelle : RDF est un sous-ensemble de la logique du premier ordre
  - Avec : prédicats binaires, quantification existentielle, conjonction
  - Sans : disjonction, négation, quantification universelle
- Tout énoncé RDF est considéré comme vrai et RDF est **monotone** i.e. ce qui est vrai et ce que l'on peut déduire reste vrai si l'on rajoute de nouveaux énoncés.

- Une ressource peut ne pas être identifiée

```
<rdf:Description rdf:about="http://bu.ch/123.html ">
  <dc:creator>
    <rdf:Description>
      <ex:nom>Duchmol</ex:nom>
      <ex:prenom>Mathieu</ex:prenom>
    </rdf:Description>
  </dc:creator>
  <dc:title>Les zélés faons roses</dc:title>
</rdf:Description>
```

- Quantification existentielle :  $\exists x \text{ dc:author}(\text{http://bu.ch/123.html}, x) \wedge \text{ex:nom}(x, \text{"Duchmol"}) \wedge \text{ex:prenom}(x, \text{"Mathieu"})$





- En cas de **noeud anonyme**, les parseurs RDF créent des ID
- On peut également introduire un ID local pour identifier les noeuds anonymes (vital pour les sérialisations)

```

<rdf:Description rdf:about="http://bu.ch/123.html ">
  <dc:creator rdf:nodeID="abc123"/>
  <dc:title>Les zélés faons roses</dc:title>
</rdf:Description>

<rdf:Description rdf:nodeID="abc123">
  <ex:nom>Duchmol</ex:nom>
  <ex:prenom>Mathieu</ex:prenom>
</rdf:Description>

```

- Il est possible d'omettre les balises **rdf:Description** pour un noeud anonyme, en utilisant l'attribut **rdf:parseType**

```

<rdf:Description rdf:about="http://bu.ch/123.html ">
  <dc:creator rdf:parseType="Resource">
    <ex:nom>Duchmol</ex:nom>
    <ex:prenom>Mathieu</ex:prenom>
  </dc:creator>
  <dc:title>Les zélés faons roses</dc:title>
</rdf:Description>

```

- Autre utilisation de **rdf:parseType** : obliger le parseur à ignorer la structure du contenu

```

<rdf:Description rdf:ID="reportR-25">
  <dc:title rdf:parseType="Literal">
    The world <i>wild</i> web
  </dc:title>
</rdf:Description>

```

## Exercice

- Récupérer le fichier *human.rdf* situé sur la page <http://www.u-picardie.fr/~furst/onto.html>
- Question 1 : Quel est le namespace utilisé pour les instances / ressources créés dans ce fichier? Par quel mécanisme l'association entre instances et namespace est-elle faite?
- Question 2 : Quel est le namespace du schéma RDF utilisé et comment est-il associé aux balises?
- Question 3 : Trouver toutes les informations sur John.

- **Réification d'un triplet** : rendre un triplet explicite pour pouvoir en parler i.e. l'utiliser comme le sujet ou l'objet d'une propriété
  - Un triplet est réifié par un statement (balise **rdf:Statement**)
  - Le statement fait du triplet une ressource
  - Cette ressource peut être décrite à son tour

```
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
...
<rdf:RDF>
<rdf:Statement rdf:nodeID="declaration1">
  <rdf:subject rdf:resource="http://www.u-picardie.fr/~furst/" />
  <rdf:predicate rdf:resource="dc:creator"/>
  <rdf:object rdf:datatype="&xsd:string">Fürst</rdf:object>
</rdf:Statement>

<rdf:Description rdf:about="declaration1">
  <dc:creator rdf:resource="http://mis.u-picardie.fr/" />
</rdf:Description>
</rdf:RDF>
```

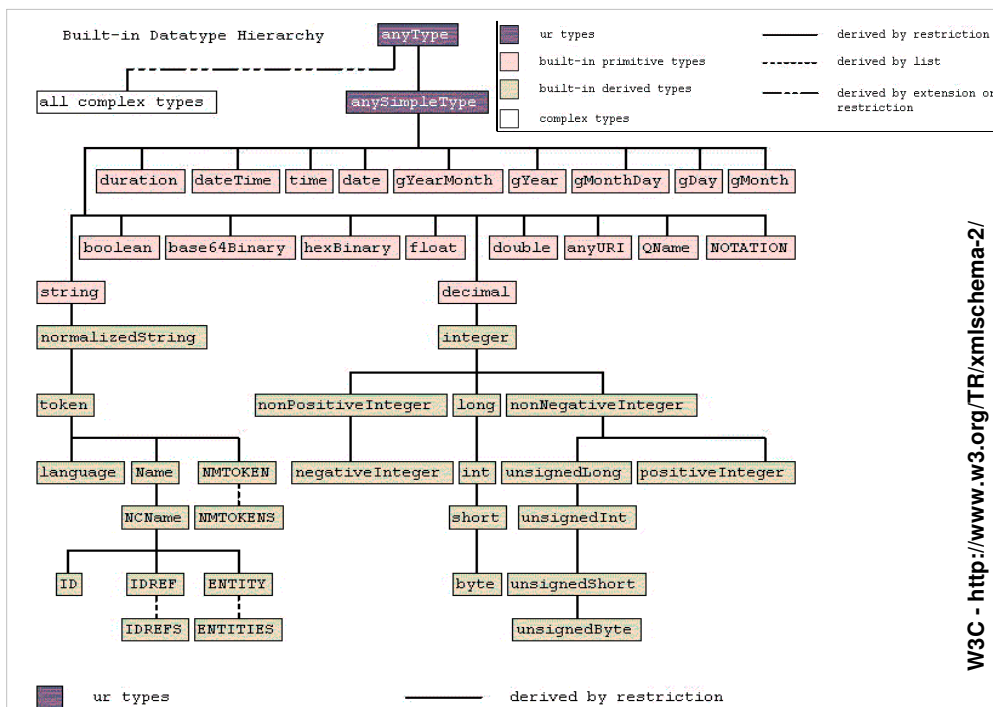
- XML schema datatypes

- Les littéraux standards sont des chaînes de caractères
- Pour typer les valeurs littérales, RDF repose sur les **datatypes de XML Schema**

```
<rdf:Description>
  ...
  <ns:age rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">12</ns:age>
  ...
</rdf:Description>
```

- Notation en N3

```
ns:id1 ns:age "22"^^xsd:integer
```



W3C - <http://www.w3.org/TR/xmlschema-2/>

- On peut  **typer les ressources**  par l'attribut `rdf:type`
  - Le type peut être une `rdf:Resource`, une `rdf:Property` ou un `rdf:Statement`
  - Une ressource peut avoir plusieurs types

```
<rdf:Description rdf:about="urn://~furst">
  <rdf:type rdf:Resource="ex:Personne"/>
  <rdf:type rdf:Resource="ex:Enseignant"/>
  <rdf:type rdf:Statement="ex:Chercheur"/>
  ...
</rdf:Description>
```

- Autre syntaxe possible :

```
<ex:Personne rdf:about="urn://~furst">
  ...
</ex:Personne>
```

- L'instanciation d'un type suffit à faire exister une ressource

```
_:x rdf:type http://www.ugb.sn/schema#Personne
```

- On peut spécifier une langue avec **xml:lang**

```
<ex:Personne rdf:about="urn://~furst">
  <ex:job xml:lang="fr">Maitre de Conférences</ex:job>
  <ex:job xml:lang="en">Associate professor</ex:job>
</ex:Personne>
```

- En N3 :

```
<urn://~furst>
  ex:job "Maitre de Conférences"@fr
  ex:job "Associate professor"@en
```

- Attention : les littéraux avec langue et sans langue sont différents

"Truc" ≠ "Truc"@en ≠ "Truc"@fr

- Un triplet peut avoir pour objet un **groupe non ordonné** de ressources ou de littéraux

```
<rdf:Description rdf:about="http://www.truc.fr/mapage.html">
  <dc:creator>
    <rdf:Bag>
      <rdf:li>Duchmol Robert</rdf:li>
      <rdf:li>Tartempion Mathieu</rdf:li>
    </rdf:Bag>
  </dc:creator>
</rdf:Description>
```

- L'accès aux valeurs se fait par l'attribut **rdf:\_n** (n entier) qui peut également être utilisé dans la déclaration

```
<rdf:Description rdf:about="http://www.truc.fr/mapage.html">
  <dc:creator>
    <rdf:Bag>
      <rdf:_1>Duchmol Robert</rdf:li>
      <rdf:_2>Tartempion Mathieu</rdf:li>
    </rdf:Bag>
  </dc:creator>
</rdf:Description>
```

- Un triplet peut avoir pour objet un **groupe ordonné** de ressources ou de littéraux

```
<rdf:Description rdf:about="http://www.truc.fr/mapage.html">
  <dc:creator>
    <rdf:Seq>
      <rdf:li>Duchmol Robert</rdf:li>
      <rdf:li>Tartempion Mathieu</rdf:li>
    </rdf:Seq>
  </dc:creator>
</rdf:Description>
```

- L'accès aux valeurs se fait de la même manière

- Un triplet peut avoir pour objet un **groupe** de ressources ou de littéraux alternatifs (le premier est le plus usité)

```
<rdf:Description rdf:about="urn://:La_vie_devant_soi">
  <dc:creator>
    <rdf:Alt>
      <rdf:li>Emile Ajar</rdf:li>
      <rdf:li>Romain Gary</rdf:li>
    </rdf:Alt>
  </dc:creator>
</rdf:Description>
```

- Une ressource peut être décrite comme liste **exhaustive** et ordonnée de constituants

```
<rdf:Description rdf:about="#Semaine">
  <ex:jours rdf:parseType="Collection">
    <rdf:Description rdf:about="#Lundi"/>
    <rdf:Description rdf:about="#Mardi"/>
    <rdf:Description rdf:about="#Mercredi"/>
    <rdf:Description rdf:about="#Jeudi"/>
    <rdf:Description rdf:about="#Vendredi"/>
    <rdf:Description rdf:about="#Samedi"/>
    <rdf:Description rdf:about="#Dimanche"/>
  </ex:jours>
</rdf:Description>
```

- Les propriétés **rdf:first** et **rdf:rest** permettent de parcourir une collection

- Description équivalente :

```
<rdf:Description rdf:about="#Semaine">
  <ex:jours rdf:nodeID="node1"/>
</rdf:Description>

<rdf:Description rdf:about="node1">
  <rdf:first rdf:resource="#Lundi"/>
  <rdf:rest rdf:nodeID="node2"/>
</rdf:Description>

<rdf:Description rdf:about="node2">
  <rdf:first rdf:resource="#Mardi"/>
  <rdf:rest rdf:nodeID="node3"/>
</rdf:Description>

...

<rdf:Description rdf:about="node7">
  <rdf:first rdf:resource="#Dimanche"/>
  <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
</rdf:Description>
```

- RDF : standard permettant l'échange de métadonnées sur le web et leur traitement automatique
- Modèle & syntaxe d'annotation :
  - Modèle simple avec une **sémantique formelle** (Graphes RDF au même niveau que l'arbre XML)
  - Format d'échange basé sur une syntaxe XML (RDF/XML)
- Le sens donné aux annotations provient des **liens entre ressources** (et entre métadonnées par réification)

- **Limitations de RDF**

- Pas de restriction sur les objets et sujets d'une propriété

```
<rdf:Description rdf:about="#souris">
  <ex:mange rdf:resource="#chat"/>
</rdf:Description>
```

- Pas de lien de généralisation/spécialisation entre types ou entre propriétés (**rdf:type** lie uniquement une instance et un type)

```
<rdf:Description rdf:about="#chat">
  <ex:mange rdf:resource="#souris"/>
</rdf:Description>

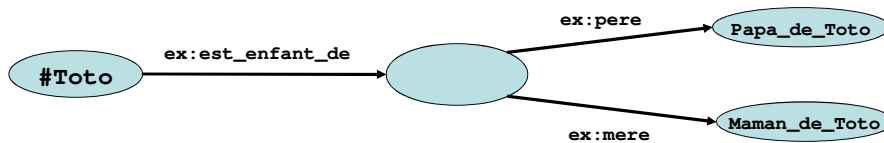
<rdf:Description rdf:about="#chat_angora">
  <rdf:type rdf:Resource="#chat"/>
  ...
</rdf:Description>
```

- Expressivité limitée : pas de quantification universelle, pas de négation, pas de disjonction

- Représenter les **relations n-aires** avec plusieurs objets

- Choisir un sujet principal
- Réifier la relation par une ressource anonyme
- Lier la relation réifiée aux différents objets

```
<rdf:Description rdf:about="#Toto">
  <ex:est_enfant_de rdf:parseType="Resource">
    <ex:pere>Papa_de_Toto</ex:pere>
    <ex:mere>Maman_de_Toto</ex:mere>
  </ex:est_enfant_de>
</rdf:Description>
```





- Parseurs RDF existants :
  - <http://www.w3.org/RDF/Validator/>
  - Jena (HP lab mais opensource) : <http://jena.sourceforge.net/>
  - Sesame (opensource) : <http://www.openrdf.org/>
  - ICS-Forth RDF suite
  - CWM
  - CORESE (INRIA)
  - Redland RDF Framework (en C)
  - ....

### Exercice

- Question 1 : en utilisant le vocabulaire défini sur la page <http://xmlns.com/foaf/0.1/>, écrire un fichier RDF exprimant les informations suivantes : Robert Duchmol est une personne surnommée Bob qui connaît Marie Tartempion qui est membre de l'association des Amis de RDF.

Valider votre fichier sur <http://www.w3.org/RDF/Validator/>

- Question 2 : sans modifier les déclarations précédentes, ajouter l'information suivante : Bob a pour fratrie, dans l'ordre, Justine et Marcel

Valider votre fichier sur <http://www.w3.org/RDF/Validator/>