3.5 LE LANGAGE OWL (WEB ONTOLOGY LANGUAGE) :

3.5.1 Naissance de OWL :

Le langage OWL est devenu une recommandation du W3C le 10 Février 2004.

3.5.2 OWL existe en 3 déclinaisons :

OWL Lite .....
.....
.....
.....

OWL DL (Description Logics) .....
.....
.....
.....
.....

OWL Full .....
.....
.....
.....
.....

3.5.3 Structure d'une ontologie :

Les ontologies OWL se présentent sous forme de fichiers texte, de « documents OWL ». (comme suffixe de nom de fichier, les extensions « .rdf » ou « .owl »).

L'espace de nommage :

Afin de pouvoir employer des termes dans une ontologie, il est nécessaire d'indiquer avec précision de quels vocabulaires ces termes proviennent. C'est la raison pour laquelle, comme tout autre document XML, une ontologie commence par une déclaration d'espace de nom (parfois appelée « de nommage ») contenue dans une balise rdf:RDF. Supposons que nous souhaitons écrire une ontologie sur une population de personnes ou, d'une manière plus générale, sur l'humanite. La figure suivante donne l'espace de nommage d'une telle ontologie.

Les deux premières déclarations identifient *l'espace de nommage propre à l'ontologie* que nous sommes en train d'écrire. La première déclaration d'espace de nom indique à quelle ontologie se rapporter en cas d'utilisation de noms sans préfixe dans la suite de l'ontologie.

La troisième déclaration identifie *l'URI de base* de l'ontologie courante.

La quatrième déclaration signifie simplement que, au cours de la rédaction de l'ontologie humanité, on va employer des concepts développés dans une ontologie vivant, qui décrit ce qu'est un être vivant.

Les quatre dernières déclaration introduisent le vocabulaire d'OWL et les objets définis dans l'espace de nommage de RDF, du schéma RDF et des types de données du Schéma XML.

```
<rdf:RDF
xmlns = "http://domain.tld/path/humanite#"
xmlns:humanite= "http://domain.tld/path/humanite#"
xmlns:base = "http://domain.tld/path/humanite#"
xmlns:vivant = "http://otherdomain.tld/otherpath/vivant#"
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#">
```

Fig. Espace de nommage d'une ontologie

Enfin , tout comme il existe une section d'en-tête <head>..</head> en haut de tout document XHTML bien formé, on peut écrire, à la suite de la déclaration d'espaces de nom, un en-tête décrivant le contenu de l'ontologie courante. C'est la balise owl:Ontology qui permet d'indiquer ces informations :

```
<owl:Ontology rdf:about="">
<rdfs:comment>Ontologie décrivant l'humanité</rdfs:comment>
<owl:imports
rdf:resource="http://otherdomain.tld/otherpath/vivant"/>

<rdfs:label>Ontologie sur l'humanité</rdfs:label>
```

Fig. Entête décrivant une ontologie.

3.5.4 Eléments du langage OWL :

La définition d'une **classe** :

Une classe définit un groupe d'individus qui sont réunis parce qu'ils ont des caractéristiques similaires. L'ensemble des individus d'une classe est désigné par le terme « extension de classe », chacun de ces individus étant alors une « instance » de la classe. Les trois versions d'OWL comportent les mêmes mécanismes de classe, à ceci près que OWL FULL est la seule version à permettre qu'une classe soit l'instance d'une autre classe (d'une métaclasse).

La déclaration d'une classe se fait de différentes manières , dont la création par le mot clé "Class" . Exemple : Une classe « humain » se déclare de la manière suivante :

```
<owl:Class rdf:ID="Humain" />
```

L'héritage :

Il est possible de dériver une classe en plusieurs sous-classes (Sub-Classe). L'exemple suivant exprime qu'un "être Humain" est une sous-classe d'un "être vivant" . Par ailleurs un "Etre humain" est dérivé en "Homme" et "Femme".

```
<owl:Class rdf:ID="Humain">
<rdfs:subClassOf rdf:resource="&etre;#EtreVivant" />
</owl:Class>
<owl:Class rdf:ID="Homme">
<rdfs:subClassOf rdf:resource="#Humain" />
</owl:Class>
<owl:Class rdf:ID="Femme">
<rdfs:subClassOf rdf:resource="#Humain" />

</owl:Class>
```

Fig. Création de classes avec Owl

Il existe dans toute ontologie OWL une superclasse, nommée Thing, dont toutes les autres classes sont des sous-classes.

L'instanciation de classes :

Définition d'un individu

La définition d'un individu consiste à énoncer un « fait », encore appelé « axiome d'individu ».

On peut distinguer deux types de faits :

1/ les faits concernant **l'appartenance à une classe**

La plupart des faits concerne généralement la déclaration de l'appartenance à une classe d'un individu et les valeurs de propriété de cet individu. Le fait écrit dans cet exemple exprime l'existence d'un Humain nommé « Ahmed » dont le père s'appelle «Omar », et qu'il a un frère nommé « Said ».

```
<Humain rdf:ID="Ahmed">

    <aPourPere rdf:resource="#Omar" />

    <aPourFrere rdf:resource="#Said" />

</Humain>
```

2/ les faits concernant **l'identité des individus**

Une difficulté qui peut éventuellement apparaître dans le nommage des individus concerne la non-unicité éventuelle des noms attribués aux individus. Par exemple, un même individu pourrait être désigné de plusieurs façons différentes. C'est la raison pour laquelle OWL propose un mécanisme permettant de lever cette ambiguité, à l'aide des propriétés *owl:sameAs*, *owl:diffrentFrom* et *owl:allDifferent*. L'exemple suivant permet de déclarer que les noms « Mohamed Boudiaf » et « Si Tayeb El Watani » désignent la même personne :

```
<rdf:Description rdf:about="#Mohamed Boudiaf">

    <owl:sameAs rdf:resource="#Si_Tayeb_El_Watani" />

</rdf:Description>
```

Une fois que l'on sait écrire une classe en OWL, la création d'une ontologie se fait par l'écriture d'instances de ces objets, et la description des relations qui lient ces instances.

Les propriétés

Les propriétés dans OWL permettent d'exprimer les liens entre les classes et leurs instances. Elles sont de deux types :

• les propriétés d'objet permettent de relier des instances à d'autres instances

• les propriétés de type de donnée permettent de relier des individus à des valeurs de données.

Une propriété d'objet est une instance de la classe owl:ObjectProperty, une propriété de type de donnée étant une instance de la classe owl:DatatypeProperty. Ces deux classes sont elles-mêmes sous-classes de la classe RDF rdf:Property.

La définition des caractéristiques d'une propriété se fait à l'aide d'un axiome de propriété qui, dans sa forme minimale, ne fait qu'affirmer l'existence de la propriété :

```
<owl:ObjectProperty rdf:ID="aPourParent" />
```

Cependant, il est possible de définir beaucoup d'autres caractéristiques d'une propriété dans un axiome de propriété.

Définition d'une propriété

Afin de spécifier une propriété, il existe différentes manières de restreindre la relation qu'elle symbolise. Par exemple, si on considère que l'existence d'une propriété pour un individu donné de l'ontologie constitue une fonction faisant correspondre à cet individu un autre individu ou une valeur de donnée, alors on peut préciser le domaine et l'image de la propriété. Une propriété peut également être définie comme la spécialisation d'une autre propriété.

```
<owl:ObjectProperty rdf:ID="habite">
    <rdfs:domain rdf:resource="#Humain" />
    <rdfs:range rdf:resource="#Pays" />
</owl:ObjectProperty>
```

Dans l'exemple ci-dessus, on apprend que la propriété habite a pour domaine la classe Humain et pour image la classe Pays : elle relie des instances de la classe Humain à des instances de la classe Pays. Dans le cas d'une propriété de type de donnée, l'image de la propriété peut être un type de donnée, comme défini dans le Schéma XML. Par exemple, on peut définir la propriété de type de données anneeDeNaissance :

```
<owl:Class rdf:ID="dateDeNaissance" />
    <owl:DatatypeProperty rdf:ID="anneeDeNaissance">
    <rdfs:domain rdf:resource="#dateDeNaissance" />
```

```
        <rdfs:range rdf:resource="&xsd;positiveInteger"/>

    </owl:DatatypeProperty>
```

Dans ce cas, anneDeNaissance fait correspondre aux instances de la classe de dateDeNaissance des entiers positifs.

On peut également employer un mécanisme de hiérarchie entre les propriétés, exactement comme il existe un mécanisme d'héritage sur les classes :

```
<owl:Class rdf:ID="Humain" />

<owl:ObjectProperty rdf:ID="estDeLaFamilleDe">

<rdfs:domain rdf:resource="#Humain" />

<rdfs:range rdf:resource="#Humain" />

</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="aPourFrere">

<rdfs:subPropertyOf rdf:resource="#estDeLaFamilleDe" />

<rdfs:range rdf:resource="#Humain" />

...

</owl:ObjectProperty>

</owl:Class>
```

La propriété aPourFrere est une sous-propriété de estDeLaFamilleDe, ce qui signifie que toute entité ayant une propriété aPourFrere d'une certaine valeur a aussi une propriété estDeLaFamilleDe de même valeur.

Caractéristiques des propriétés

En plus de ce mécanisme d'héritage et de restriction du domaine et de l'image d'une propriété, il existe divers moyens d'attacher des caractéristiques aux propriétés, ce qui permet d'affiner grandement la qualité des raisonnements liés à cette propriété.

Parmi les caractéristiques de propriétés principales, on trouve la transitivité, la symétrie, la fonctionnalité, l'inverse, etc. L'ajout d'une caractéristique à une propriété de l'ontologie se fait par l'emploi de la balise OWL type :

Exemple de propriété non symétrique :

```
<owl:ObjectProperty rdf:ID="aPourPere">

<rdfs:domain rdf:resource="#Humain" />

<rdfs:range rdf:resource="#Humain" />
```

```
</owl:ObjectProperty>
```

Exemple de propriété non symétrique :

```
<owl:ObjectProperty rdf:ID="aPourFrere">
<rdf:type rdf:resource="&owl;SymmetricProperty" />
<rdfs:domain rdf:resource="#Humain" />
<rdfs:range rdf:resource="#Humain" />
</owl:ObjectProperty>

<Humain rdf:ID="Pierre">
<aPourFrere rdf:resource="#Paul" />
<aPourPere rdf:resource="#Jacques" />
</Humain>
```

L'Humain Pierre a pour frère Paul, de même que (symétrie) l'Humain Paul a pour frère Pierre. Par contre, si Pierre a pour père Jacques, l'inverse n'est pas vrai (aPourPere n'est pas symétrique).