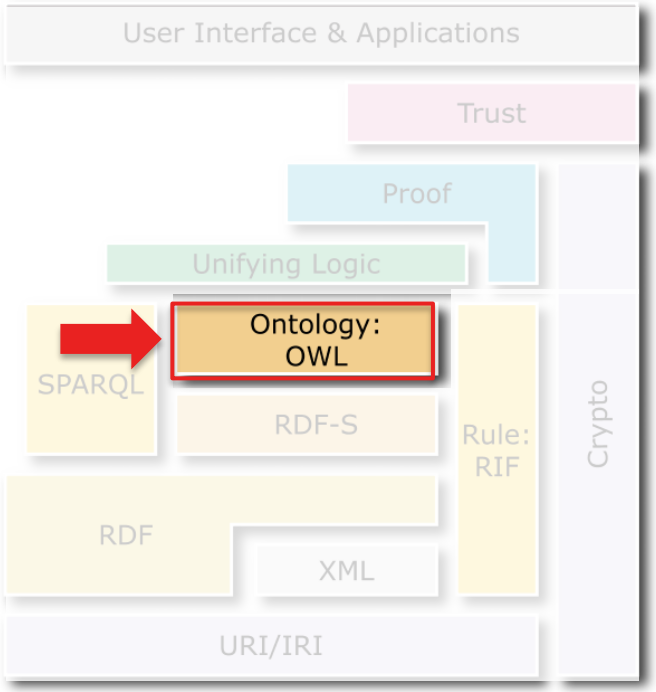
SEMAINE 05 : formalisation en OWL

1. Relations de classes
2. Caractérisation des propriétés
3. Equivalences et alignements
4. Restriction de propriétés
5. Gérer les schémas
6. Profiles OWL

SEMAINE 05 : formalisation en OWL



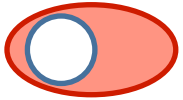
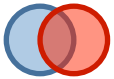
Web Ontology Language (OWL)

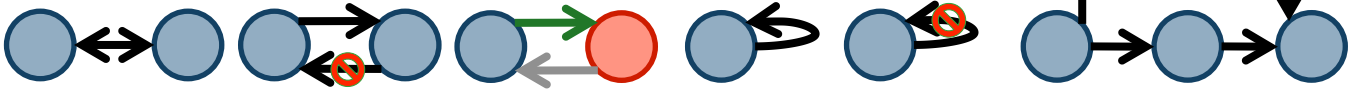
- recommandation W3C
- fournit des primitives supplémentaires pour des ontologies plus complexes.
- permet des définitions plus riches des classes et propriétés.
- permet de tirer plus de conclusions, de faire plus d’inférences.

OWL in one…

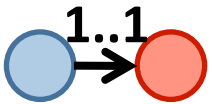
une vue graphique des constructeurs logiques offerts

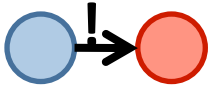
union  disjonction  union disjointe 

complément  intersection  énumération 

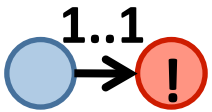
prop. algébriques 

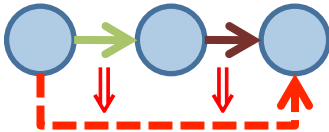
prop. disjointes 

cardinalité 

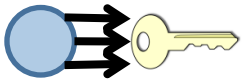
restriction 

négation prop. indiv 

cardinalité qualifiée 

chaîne 

équivalence 

clefs 

restrict. valeur 

espace de noms et préfixe pour utiliser OWL

http://www.w3.org/2002/07/owl#

- espace des primitives OWL
- même principe que pour RDFS
- préfixe `owl:` dans la suite

SEMAINE 05 : formalisation en OWL

1. **Relations de classes**
2. Caractérisation des propriétés
3. Equivalences et alignements
4. Restriction de propriétés
5. Gérer les schémas
6. Profiles OWL

Démo protégé

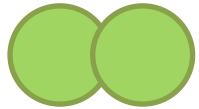
classes énumérées

définir une classe en énumérant tous ses membres

```
<owl:Class rdf:ID="EyeColor">
    <owl:oneOf rdf:parseType="Collection">
      <owl:Thing rdf:ID="Blue"/>
      <owl:Thing rdf:ID="Green"/>
      <owl:Thing rdf:ID="Brown"/>
      <owl:Thing rdf:ID="Black"/>
    </owl:oneOf>
</owl:Class>
```

```
<EyeColor> rdf:type owl:Class ;
  owl:oneOf
      ( <Blue> <Green> <Brown> <Black> )   .
```

classes définies par union d'autres classes



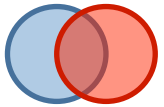
toute ressource des classes unies est aussi dans la classe union

```
<owl:Class rdf:ID="LegalAgent">
   <owl:equivalentClass>
        <owl:Class>
            <owl:unionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#Person"/>
                <owl:Class rdf:about="#Group"/>
            </owl:unionOf>
        </owl:Class>
   </owl:equivalentClass>
</owl:Class>

<LegalAgent> a owl:Class ;
     owl:equivalentClass [
        a owl:Class ;
        owl:unionOf ( <Person> <Group> )
    ] .
```

owl:equivalentClass est présenté dans la section 3

classes définies par intersection



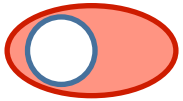
toute ressource commune aux classes est aussi dans la classe intersection

```
<owl:Class rdf:ID="Man">
    <owl:equivalentClass>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#Person"/>
                <owl:Class rdf:about="#Male"/>
            </owl:intersectionOf>
        </owl:Class>
    </owl:equivalentClass>
</owl:Class>
```

```
<Man> a owl:Class ;
     owl:equivalentClass [
        a owl:Class ;
        owl:intersectionOf ( <Person> <Male> )   .
    ]
```

classes définies par négation

les ressources qui ne font pas partie d'une classe



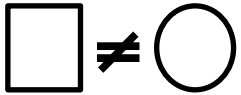
```
<owl:Class rdf:ID="Inedible">
    <owl:equivalentClass>
        <owl:Class>
            <owl:complementOf rdf:resource="#Edible"/>
        </owl:Class>
    </owl:equivalentClass>
</owl:Class>
```

```
<Inedible> a owl:Class ;
     owl:equivalentClass [
        a owl:Class ;
        owl:complementOf <Edible>
    ] .
```

disjonction de deux classes

une ressource ne peut pas appartenir aux deux classes en même temps

```
<owl:Class rdf:ID="Square">
  <owl:disjointWith rdf:resource="#Circle"/>
</owl:Class>
```

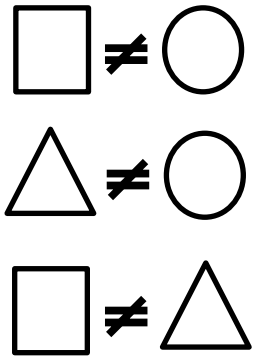


```
<Square> rdf:type owl:Class ;
  owl:disjointWith <Circle> .
```

disjonction de plusieurs classes

une ressource ne peut, au plus, appartenir qu'à une seule de ces classes

```
<owl:AllDisjointClasses>
  <owl:members rdf:parseType="Collection">
    <owl:Class rdf:about="#Square"/>
    <owl:Class rdf:about="#Circle"/>
    <owl:Class rdf:about="#Triangle"/>
  </owl:members>
</owl:AllDisjointClasses>
```



```
[] rdf:type owl:AllDisjointClasses ;
   owl:members
    ( <Square> <Circle> <Triangle> ) .
```

union disjointe

diviser une classe en une partition complète de sous classes

```
<owl:Class rdf:about="#Passenger">
  <owl:disjointUnionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Adult"/>
    <owl:Class rdf:about="#Child"/>
    <owl:Class rdf:about="#Pet"/>
  </owl:disjointUnionOf>
</owl:Class>
```

```
<Passenger> rdf:type owl:Class ;
  owl:disjointUnionOf
  ( <Adult> <Child> <Pet> ) .
```

SEMAINE 05 : formalisation en OWL

1. Relations de classes
2. **Caractérisation des propriétés**
3. Equivalences et alignements
4. Restriction de propriétés
5. Gérer les schémas
6. Profiles OWL

trois types différents de propriétés

1. `owl:ObjectProperty` sont des relations entre des ressources
2. `owl:DatatypeProperty` ont des valeurs littérales (typées)
3. `owl:AnnotationProperty` sont ignorées dans les inférences et utilisées pour documenter ou pour des extensions

propriétés symétriques

une relation qui, dès lors qu'elle existe, existe dans les deux sens (ex. marié_avec)

x R y ⇒ y R x

```
<owl:SymmetricProperty rdf:ID="hasSpouse" />
```

propriétés asymétriques

une relation qui, dès lors qu'elle existe, n'existe que dans un seul sens.

$x\ R\ y \Rightarrow \neg\ y\ R\ x$

```
<owl:AsymmetricProperty rdf:ID="hasChild" />
```

propriétés inverses

deux relations qui existent simultanément en sens inverse (ex. parent_de / enfant_de)

$x\ R_1\ y \Leftrightarrow y\ R_2\ x$

```
<rdf:Property rdf:ID="hasChild">
  <owl:inverseOf rdf:resource="#hasParent" />
</rdf:Property>
```

propriétés transitives

une relation qui se propage de proche en proche (ex. Tom ancêtre Jim ancêtre Jules)

x R y & y R z ⇒ x R z

```
<owl:TransitiveProperty rdf:ID="hasAncestor" />
```

propriétés disjointes

Des relations qui ne peuvent pas exister en même temps sur le même sujet et le même objet

```
<owl:ObjectProperty rdf:about="#hasSon">
  <owl:propertyDisjointWith rdf:resource="#hasDaughter"/>
</owl:ObjectProperty>
```

propriétés réflexives

Une relation qui relie tous les individus à eux mêmes

```
<owl:ReflexiveProperty rdf:about="#hasRelative"/>
```

propriétés irréflexives

Une relation qui ne relie aucun individu à lui même

```
<owl:IrreflexiveProperty rdf:about="#hasParent"/>
```

propriétés chaînées

des relations qui mises bout à bout impliquent une autre relation
(ex. parent + frère = oncle)

$x\ P\ y\ \&\ y\ Q\ z \Rightarrow x\ R\ z$

```
<owl:ObjectProperty rdf:ID="uncle">
  <owl:propertyChainAxiom rdf:parseType="Collection">
    <owl:ObjectProperty rdf:about="#parent"/>
    <owl:ObjectProperty rdf:about="#brother"/>
  </owl:propertyChainAxiom>
</owl:ObjectProperty>
```

propriétés fonctionnelles

une relation pour laquelle une ressource ne peut avoir qu'une valeur (ex. naissance)

$x\ R\ y\ \&\ x\ R\ z \Rightarrow y = z$

```
<owl:FunctionalProperty rdf:ID="birthDate" />
```

propriétés inverses fonctionnelles

une relation pour laquelle une même valeur implique la même ressource (ex. NSS)

$x\ R\ y\ \&\ z\ R\ y \Rightarrow x = z$

```
<owl:InverseFunctionalProperty
    rdf:ID="socialSecurityNumber" />
```

identification par des clés

deux instances qui ont même(s) valeur(s) de clé(s) sont en réalité la même instance

$x\ c_1\ v_1\ ;\ c_2\ v_2\ \&\ y\ c_1\ v_1\ ;\ c_2\ v_2 \Rightarrow x = y$

```
ex:Person owl:hasKey ( ex:hasSSN ) .
```

SEMAINE 05 : formalisation en OWL

1. Relations de classes
2. Caractérisation des propriétés
3. **Equivalences et alignements**
4. Restriction de propriétés
5. Gérer les schémas
6. Profiles OWL

équivalences de classes

ces deux classes rassemblent exactement les mêmes ressources.

```
ex:Human owl:equivalentClass foaf:Person
```

équivalences de propriétés

ces deux types de propriétés expriment exactement la même relation.

```
ex:name owl:equivalentProperty my:label
```

ressources identiques

deux URI qui identifient exactement la même chose.

```
ex:Bill owl:sameAs ex:William
```

transitivité: l'identité se propage

- URI_1 owl:sameAs URI_2
- URI_2 owl:sameAs URI_3

…

- URI_1 owl:sameAs URI_3

ressources différentes

deux URI dont on sait qu'ils identifient deux choses différentes.

```
ex:Good owl:differentFrom ex:Evil
```

SEMAINE 05 : formalisation en OWL

1. Relations de classes
2. Caractérisation des propriétés
3. Equivalences et alignements
4. **Restriction de propriétés**
5. Gérer les schémas
6. Profiles OWL

restriction de toutes les valeurs d'une propriété

classe pour laquelle toutes les valeurs d'une propriété sont issues d'une autre classe

```
<owl:Class rdf:ID="Herbivore">
  <rdfs:subClassOf rdf:resource="#Animal"/>
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:onProperty rdf:resource="#eats" />
    <owl:allValuesFrom rdf:resource="#Plant" />
   </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

restriction de certaines valeurs d'une propriété

au moins certaines valeurs de la propriété sont issues d'une classe spécifique

```
<owl:Class rdf:ID="Sportive">
 <owl:equivalentClass>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#hobby" />
   <owl:someValuesFrom rdf:resource="#Sport" />
  </owl:Restriction>
 </owl:equivalentClass>
</owl:Class>
```

restriction à une seule valeur pour une propriété

la classe définie ne peut avoir qu'une seule valeur pour la propriété visée

```
<owl:Class rdf:ID="Bicycle">
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#nbWheels" />
   <owl:hasValue>2</owl:hasValue>
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

restriction de relation avec soi-même

classe définie par les instances ayant elles-mêmes comme valeur d'une propriété

```
ex:NarcisticPerson rdfs:subClassOf

[ a owl:Restriction ;
   owl:onProperty ex:love ;
   owl:hasSelf true ]
```

restriction de cardinalité

contrainte sur le nombre de fois qu'une propriété peut être utilisée avec des valeurs différentes sur le même sujet : minimum, maximum, nombre exact

```
<owl:Class rdf:ID="Person">
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#name" />
   <owl:maxCardinality>1</owl:maxCardinality>
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

restriction de cardinalité qualifiée

contrainte sur le nombre de fois qu'une propriété peut être utilisée avec des valeurs d'un certain type sur le même sujet : minimum, maximum, nombre exact

```
<owl:Class rdf:ID="Human">
 <rdfs:subClassOf>
  <owl:Restriction>
   <owl:onProperty rdf:resource="#hasParent" />
   <owl:onClass rdf:resource="#Male" />
   <owl:qualifiedCardinality>1</owl:qualifiedCardinality>
  </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

SEMAINE 05 : formalisation en OWL

1. Relations de classes
2. Caractérisation des propriétés
3. Equivalences et alignements
4. Restriction de propriétés
5. **Gérer les schémas**
6. Profiles OWL

documenter les schémas et leurs versions

- L’ontologie est aussi une ressource
- L’ontologie a un URI qui nous permet d’en parler
- OWL fournit des primitives pour décrire cette ressource qu’est l’ontologie

description de l'ontologie

une classe (`owl:Ontology`) et plusieurs propriétés (`owl:imports, owl:versionInfo, owl:priorVersion, owl:backwardCompatibleWith, owl:incompatibleWith`)

```
<rdf:RDF xml:base="http://inria.fr/2005/humans/"
 xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl ="http://www.w3.org/2002/07/owl#">

<owl:Ontology rdf:about="http://inria.fr/2005/humans/">




</owl:Ontology>
</rdf:RDF>
```

description de l'ontologie

une classe (`owl:Ontology`) et plusieurs propriétés (`owl:imports, owl:versionInfo, owl:priorVersion, owl:backwardCompatibleWith, owl:incompatibleWith`)

```
<rdf:RDF xml:base="http://inria.fr/2005/humans/"
 xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl ="http://www.w3.org/2002/07/owl#">

<owl:Ontology rdf:about="http://inria.fr/2005/humans/">
 <rdfs:comment>An example OWL ontology</rdfs:comment>




</owl:Ontology>
</rdf:RDF>
```

description de l'ontologie

une classe (`owl:Ontology`) et plusieurs propriétés (`owl:imports, owl:versionInfo, owl:priorVersion, owl:backwardCompatibleWith, owl:incompatibleWith`)

```
<rdf:RDF xml:base="http://inria.fr/2005/humans/"
 xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl ="http://www.w3.org/2002/07/owl#">

<owl:Ontology rdf:about="http://inria.fr/2005/humans/">
 <rdfs:comment>An example OWL ontology</rdfs:comment>
 <owl:priorVersion
    rdf:resource="http://inria.fr/2004/humans/"/>


</owl:Ontology>
</rdf:RDF>
```

description de l'ontologie

une classe (`owl:Ontology`) et plusieurs propriétés (`owl:imports, owl:versionInfo, owl:priorVersion, owl:backwardCompatibleWith, owl:incompatibleWith`)

```
<rdf:RDF xml:base="http://inria.fr/2005/humans/"
 xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl ="http://www.w3.org/2002/07/owl#">

<owl:Ontology rdf:about="http://inria.fr/2005/humans/">
 <rdfs:comment>An example OWL ontology</rdfs:comment>
 <owl:priorVersion
    rdf:resource="http://inria.fr/2004/humans/"/>
 <owl:imports rdf:resource="http://cnrs.fr/animals/"/>


</owl:Ontology>
</rdf:RDF>
```

description de l'ontologie

une classe (`owl:Ontology`) et plusieurs propriétés (`owl:imports, owl:versionInfo, owl:priorVersion, owl:backwardCompatibleWith, owl:incompatibleWith`)

```
<rdf:RDF xml:base="http://inria.fr/2005/humans/"
 xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl ="http://www.w3.org/2002/07/owl#">

<owl:Ontology rdf:about="http://inria.fr/2005/humans/">
 <rdfs:comment>An example OWL ontology</rdfs:comment>
 <owl:priorVersion
    rdf:resource="http://inria.fr/2004/humans/"/>
 <owl:imports rdf:resource="http://cnrs.fr/animals/"/>
 <rdfs:label>Bio Ontology</rdfs:label>
</owl:Ontology>
</rdf:RDF>
```

changements de classes ou propriétés

marquer une classe ou une propriété comme obsolète

```
<rdf:RDF xml:base="http://inria.fr/2005/humans/"
 xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:owl="http://www.w3.org/2002/07/owl#">

 <owl:DeprecatedClass rdf:ID="mammals"/>
 <owl:DeprecatedProperty rdf:ID="age"/>

</rdf:RDF>
```

SEMAINE 05 : formalisation en OWL

1. Relations de classes
2. Caractérisation des propriétés
3. Equivalences et alignements
4. Restriction de propriétés
5. Gérer les schémas
6. **Profiles OWL**

différents profiles : différentes expressivités

- Chaque profile correspond à un sous ensemble des primitives de OWL.
- Choisir un profile c'est choisir une expressivité pour décrire une ontologie.
- Plus le degré d'expressivité est grand plus les inférences sont complexes.

les différents profiles de OWL v1

- **Lite** : essentiellement pour des hiérarchies simples.
- **DL** : plus expressive mais toujours avec un raisonnement complet.
- **Full** : expressivité maximum mais le raisonnement peut être incomplet.

les différents profiles de OWL v2

- **EL :** grand nombre de propriétés et/ou classes et temps polynomial.
- **QL :** grand volume d'instances, requêtes conjonctives avec approches BD relationnelles conventionnelles en LOGSPACE
- **RL :** raisonnement passant à l'échelle sans perdre trop d'expressivité; règles d'inférence en temps polynomial
- **DL :** la plus expressive

SEMAINE 05 : formalisation en OWL