Introduction à OWL-2 (Ontology Web Language)

Bernard ESPINASSE

Aix-Marseille Université
LIS UMR CNRS 7020

Novembre 2019

- **De OWL1 à OWL2**
- **Modélisation de base**
- **Relations avancées de classes**
- **Utilisation avancées de propriétés**
- **Profils OWL2: EL, QL et RL**
- **Limites de OWL2**

Réferences

- **Livres, articles et rapports :**
  - W3C, « OWL Web Ontology Language Semantics and Abstract Syntax »
  - …
- **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/
  - …
- **Course/tutorials :**
  - Cours de G. Lapalme, Université de Montréal
  - Cours de J. Dibie, AgroParisTech
  - Cours de I. Horrocks and F. Farazi
  - Tutorial de M. Kuba, Institute of Computer Science, *http://dior.ics.muni.cz/~makub/owl/*
  - …

Plan

1. De OWL1 à OWL2

- **Définition et objectifs de OWL**
- **OWL et les logiques de description *(SROIQ)***
- **De OWL1 à OWL2**
- **Sémantiques et Syntaxes de OWL2**

Introduction à OWL

- **OWL** permet une représentation de connaissances riches et complexes à propos de :
  - *choses*
  - groupes de *choses*
  - relations entre *choses*
- Basé sur une logique *calculatoire* permettant de :
  - vérifier la consistance des connaissances
  - expliciter des connaissances implicites

Principe de OWL

- Langage **déclaratif** pour exprimer des ontologies
- **Ce n'est pas :**
  - *un langage de schéma* : on ne peut pas forcer l'apparition de certaines informations
  - *un modèle de base de données* : un monde ouvert plutôt que fermé : une information manquante peut être vraie

  Mais une BD peut toutefois servir d'infrastructure pour conserver l'ontologie

Modélisation des connaissances en OWL

- S'appuie sur :
  - Axiomes : énoncés de base supposés vrais
  - Entités : référents aux objets du monde
  - Expressions : combinaisons d'entités pour former des descriptions complexes à partir de formes de base
- Le résultat de la modélisation est appelé *ontologie*

Expression en OWL

- Combinaison de noms d'entités avec des **constructeurs** pour de nouvelles entités

  Exemple : combine femme et professeur pour obtenir la classe des professeures
- Langage riche pour la combinaison de classe
- Peu de combinaison de propriétés

Représentation des connaissances

- **Énoncés de base :**
  - *il pleut*
  - *tout homme est mortel*
- **Conséquences des énoncés :**
  - un énoncé a est vrai si d'autres A le sont
  - A entraîne (*entails*) a
  - A est consistant s'il y a une situation où tous ses énoncés sont vrais
  - A est inconsistant si on ne peut trouver de situation où tous ses énoncés sont vrais
- **Sémantique formelle** : définit les états pour lesquels un ensemble d'énoncés sont vrais.

OWL: entités …

Nom OWL	Exemple	Nom commun
Individu	Pierre, Marie, une table, …	Objet
Classe	Homme, femme, meuble, …	Catégorie
Propriété		relation
objet-objet	mariéA, pèreDe, audessusDe,…	
objet-valeur	Age, …	
annotation	Auteur, date de creation, …	

Raisonnement à partir des axiomes

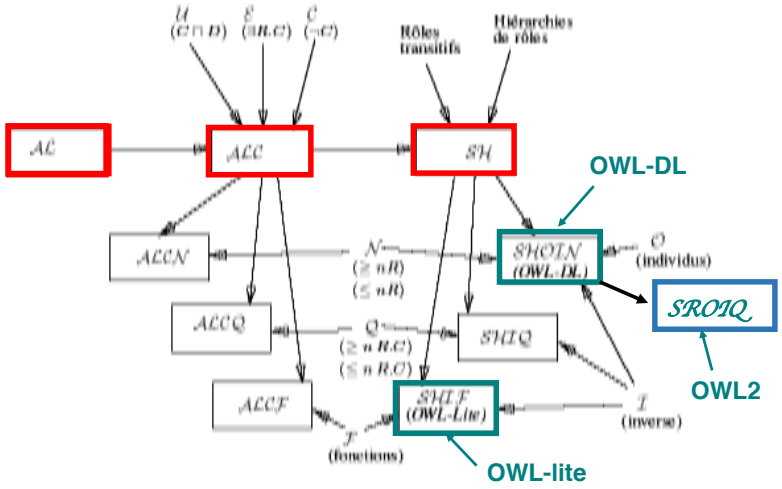
- Calcul automatique des conséquences d'un ensemble d'axiomes
- Outils sont appelés « reasoners »
- Pas toujours facile à contrôler ou à en comprendre les résultats

Définition et objectifs de OWL2

- **OWL 2** est une **extension** et une **revision** de OWL Web Ontology Language (**OWL 1**) développé par le W3C Web Ontology Working Group et publié en 2004
- **OWL 2** est développé par le W3C OWL Working Group (2009-…)
- **OWL 2** est compatible avec OWL1
- Comme OWL 1, **OWL 2** est conçu **pour faciliter le développement d'ontologies** et leur **partage** sur le Web
- Avec le but ultime de **rendre accessible le contenu du Web aux machines**.

OWL et logiques de description (1)

Photo de famille des logiques de description $\mathcal{AL}$, $\mathcal{ALC}$, $\mathcal{SH}$ … (From Gagnon) :



OWL et logiques de description (2)

OWL1 :

- **OWL1 DL** basé sur la LD *SHOIN*
- **OWL1 Lite** basé sur la LD *SHIF* DL
- **OWL1 Full** basé sur une *logique undecidable*, *sur-ensemble de SHOIN*

OWL2 :

- **OWL2 (DL)** basé sur la LD *SROIQ, et*
- **OWL2 (DL)** est orienté vers des ontologies avec un haut degré d'expressivité dans le langage.

Overview of *SROIQ* Description Logic (1)

(source : Sebastian Rudolphe)

class expressions	
class names	A, B
conjunction	$C \sqcap D$
disjunction	$C \sqcup D$
negation	$\neg C$
existential role restriction	$\exists r.C$
universal role restriciton	$\forall r.C$
Self	$\exists s.\text{Self}$
atleast restriction	$\geqslant n\, s.C$
atmost restriction	$\leqslant n\, s.C$
nominals	$\{a\}$

TBox (class axioms)	
inclusion	$C \sqsubseteq D$
equivalence	$C \equiv D$

Overview of *SROIQ* Description Logic (2)

(source : Sebastian Rudolphe)

Roles

roles	r, s, t
simple roles	s, t
universal role	u

ABox (assertions)

class membership	$C(a)$
role membership	$r(a, b)$
neg. role membership	$\neg s(a, b)$
equality	$a \approx b$
inequality	$a \not\approx b$

RBox (role axioms)

inclusion	$r_1 \sqsubseteq r_2$
complex role inclusion	$r_1 \circ \ldots \circ r_n \sqsubseteq r$
transitivity	$\mathrm{Trans}(r)$
symmetry	$\mathrm{Sym}(r)$
reflexivity	$\mathrm{Ref}(r)$
irreflexivity	$\mathrm{Irr}(s)$
disjointness	$\mathrm{Dis}(s, t)$

- Les inferences dans *SHOIN* (OWL1 DL) sont très complexes (NExpTime)
- Les inferences dans *SROIQ* (OWL2 DL) sont moins complexes : *algorithme efficace*, et la méthode des tableaux montre une *décidabilité*

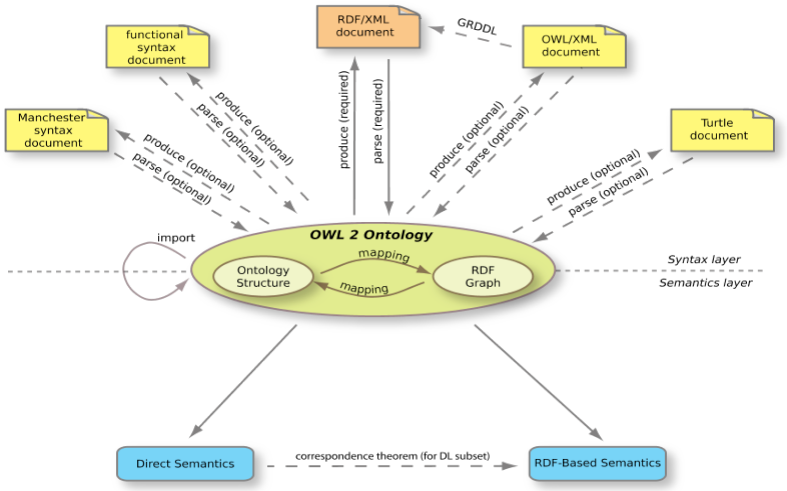
Sémantiques de OWL-2

- **Structure de l'ontologie ⇔ graphe RDF**
- **2 Voies alternatives pour donner du sens aux ontologies OWL2 :**
  - **OWL 2 DL : sémantique directe** de la structure [*OWL 2 Direct Semantics*]
    - *« direct model semantics »*
    - correspond au modèle en logique de description
  - **OWL 2 RDF : Sémantique basée sur RDF (RDF graphe)** [*OWL 2 RDF-Based Semantics*]
    - *extension de la sémantique de RDFS*
    - *considère l'ontologie comme un graphe RDF*
    - *indécidable!!!*
- différences assez mineures dans la majorité des cas

Syntaxes de OWL-2

- **Diverses syntaxes ou notations :**
  - **RDF/XML** : échange (obligatoire dans tous les outils)
  - **OWL/XML** : facilement traitable par outils XML
  - **Fonctionnelle** : fait ressortir la structure formelle
  - **Manchester** : ontologies plus faciles à lire/écrire
  - **Turtle** : triplets plus faciles à lire/écrire
- Il existe des *éditeurs graphiques d'ontologie* pour créer/lire ces syntaxes

Syntaxes et Sémantiques de OWL-2



2. Modélisation de base en OWL2

- **Classes et instances**
- **Hiérarchies de classes**
- **Propriétés d'objet**
- **Hiérarchies de propriétés**
- **Restrictions sur les propriétés**
- **Égalité des individus**
- **Types des valeurs de propriétés**

Description de classes OWL

OWL distingue 6 types de descriptions de classes :

1. Identifiant de classe (URI)
2. Enumération exhaustive de ses individus qui forment ses instances possibles
3. Intersection entre une ou plusieurs descriptions de classes
4. Union entre une ou plusieurs descriptions de classes
5. Complément d'une description de classe
6. Restriction de propriétés

Classes et instances:

- Notation **Manchester** (utilisée dans Protégé) Frame-based (.omn)
- Marie est une personne (syntaxe Manchester) :

```
Individual: Mary
   Types: Person
```

- Marie est une femme (syntaxe Manchester) :

```
Individual: Mary
   Types: Woman
```

Hiérarchies de classes

- **Sous-classes (syntaxe Manchester) :**

```
Class: Woman
   SubClassOf: Person
Class: Mother
   SubClassOf: Woman
Class: Person
   EquivalentTo: Human
```

- **Classes disjointes (syntaxe Manchester) :**

```
DisjointClasses: Woman, Man
```


Classes et instances dans diverses syntaxes

Functional-Style Syntax

```
ClassAssertion( :Woman :Mary )
```

RDF/XML Syntax

```
<Woman rdf:about="Mary"/>
```

Turtle Syntax

```
:Mary rdf:type :Woman .
```

Manchester Syntax

```
Individual: Mary
  Types: Woman
```

OWL/XML Syntax

```
<ClassAssertion>
   <Class IRI="Woman"/>
   <NamedIndividual IRI="Mary"/>
</ClassAssertion>
```

Hiérarchies de classes dans diverses syntaxes

Functional-Style Syntax

```
SubClassOf( :Woman :Person )
```

RDF/XML Syntax

```
<owl:Class rdf:about="Woman">
  <rdfs:subClassOf rdf:resource="Person"/>
</owl:Class>
```

Turtle Syntax

```
:Woman rdfs:subClassOf :Person .
```

Manchester Syntax

```
Class: Woman
  SubClassOf: Person
```

OWL/XML Syntax

```
<SubClassOf>
  <Class IRI="Woman"/>
  <Class IRI="Person"/>
</SubClassOf>
```


Propriétés d'objets (relation entre individus)

- **Positive :**

```
Individual: John
   Facts: hasWife Mary
```

- **Négative :**

```
Individual: Bill
   Facts: not hasWife Mary
```

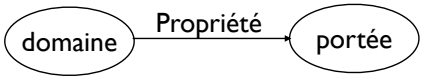
en OWL, tout est possible à moins d'affirmer le contraire!

Hiérarchies des propriétés

- Semblables à celle pour les classes :

```
ObjectProperty: hasWife
   SubPropertyOf: hasSpouse
EquivalentProperties:
   hasChild, otherOnt:child
```


Restriction sur les propriétés




```
ObjectProperty: hasWife
      Domain: Man
      Range: Woman
```

Egalité et inégalité des individus

- OWL ne suppose pas que 2 individus de noms différents sont différents
- **Différent :**

```
Individual: John
    DifferentFrom: Bill
```

- **Identique :**

```
Individual: James
    SameAs: Jim
```


Types de valeurs de propriétés

- **Positif :**

```
Individual: John
   Facts: hasAge "51"^^xsd:integer
```

- **Négatif :**

```
Individual: Jack
   Facts: not hasAge "53"^^xsd:integer
```

- **Domaine et portée des propriétés :**

```
DataProperty: hasAge
   Domain: Person
   Range: xsd:nonNegativeInteger
```

Lien: https://www.w3.org/TR/2009/WD-owl2-primer-20090611/#Property_Characteristics

3. Relations avancées de classes en OWL2

- - Classes complexes
- - Restrictions de propriétés
- - Restrictions de cardinalité
- - Enumérations d'individus

Classes complexes

- **Intersection :**

```
Class: Mother
   EquivalentTo: Woman and Parent
```

- **Union :**

```
Class: Parent
   EquivalentTo: Mother or Father
```

- **Complément**

```
Class: ChildlessPerson
   EquivalentTo: Person and not Parent
```

- **Définition de sous-classes :**

```
Class: Grandfather
   SubClassOf: Man and Parent
```


Restrictions de propriétés

- **Quantification existentielle :**

```
Class: Parent
   EquivalentTo: hasChild some Person
```

Les parents sont les individus qui sont liés à une personne par le lien hasChild

- **Quantification universelle :**

```
Class: HappyPerson
   EquivalentTo: hasChild only HappyPerson
                 and hasChild some HappyPerson
```

Une personne heureuse si tous ses enfants sont heureux

Mais alors, une personne sans enfant est heureuse...

Restrictions de cardinalité

- **Cardinalité maximum :**

```
Individual: John
   Types: hasChild max 4 Parent
```

John fait partie de la classe des gens ayant au plus 4 enfants qui sont parents, mais il pourrait avoir d'autres enfants qui ne sont pas parents

- **Cardinalité minimum :**

```
Individual: John
   Types: hasChild min 2 Parent
```

- **Cardinalité exacte :**

```
Individual: John
   Types: hasChild exactly 3 Parent
```

La spécification de la classe est optionnelle

Énumération d'individus

```
Class: MyBirthdayGuests
   EquivalentTo: { Bill, John, Mary }
```

Lien: https://www.w3.org/TR/2009/WD-owl2-primer-20090611/#Property_Characteristics

4. Utilisation avancées de propriétés en OWL2

- **Caractéristiques des propriétés**
- **Chaînes de propriétés**
- **Clés**

Caractéristiques des propriétés en OWL1/OWL2

- **OWL-1 :**
  - **Fonctionnelle** (au plus une valeur, e.g. ***hasAge***)

    $P(x,y) \wedge P(x,z) \rightarrow y = z$
  - **Inverse fonctionnelle** (l'inverse de la propriété est fonctionnelle, e.g. ***IsAgeOf***)

    $P(y,x) \wedge P(z,x) \rightarrow y = z$
  - **Transitive** (e.g. ***is-part-of***)

    $P(x,y) \wedge P(y,z) \rightarrow P(x,z)$
  - **Symétrique** (e.g. ***aLeMemeAge***)

    $P(x, y) \Leftrightarrow P(y, x)$
- **Spécificités OWL-2 :**
  - **Asymétrique** (OWL2) (e.g. ***estPlusGrand***)
  - **Réflexive** (OWL2) (e.g. ***aLeMemeAge***)

    $P(x, x)$
  - **Irréflexive** (OWL2) (e.g. ***estPlusGrand***)

Autres spécificités de OWL-2

- Les **chaines de propriétés**
- Les **clés**
- Définitions de **types de données plus riches**

  `owl:NegativePropertyAssertion`

  Exemple : Ernestine ne peut pas avoir 12 ans
- Possibilité de définir des **restrictions sur les cardinalités plus avancées** :

  `owl:maxQualifiedCardinality,`

  `owl:minQualifiedCardinality`

  Exemple : Léon a au moins 2 copines qui sont des copines d'Ernestines
- Amélioration des **possibilités d'annotations**

Caractéristiques des propriétés (1)

- **Inverse :**

```
ObjectProperty: hasParent
InverseOf: hasChild
```

- **Symétrie :**

```
ObjectProperty: hasSpouse
Characteristics: Symmetric
```

- **Asymétrie :**

```
ObjectProperty: hasChild
Characteristics: Asymmetric
```

- **Disjonction :**

```
DisjointProperties: hasParent, hasSpouse
```

Caractéristiques des propriétés (2)

- **Réflexivité :**

```
ObjectProperty: hasRelative
   Characteristics: Reflexive
ObjectProperty: parentOf
   Characteristics: Irreflexive
```

- **Unicité de valeur :**

```
ObjectProperty: hasHusband
   Characteristics: Functional
ObjectProperty: hasHusband
   Characteristics: InverseFunctional
```

- **Transitivité :**

```
ObjectProperty: hasAncestor
   Characteristics: Transitive
```


Chaîne de propriétés

- **Permet de limiter la transitivité :**

```
ObjectProperty: hasGrandparent
   SubPropertyChain: hasParent o hasParent

ObjectProperty: hasUncle
   SubPropertyChain: hasParent o hasBrother
```

- **Exemple syntaxe XML :**

```
<rdf:Description rdf:about="hasGrandparent">
<owl:propertyChainAxiom rdf:parseType="Collection">
      <owl:ObjectProperty rdf:about="hasParent"/>
      <owl:ObjectProperty rdf:about="hasParent"/>
   </owl:propertyChainAxiom>
 </rdf:Description>
```

- **Syntaxe Turtle :**

```
:hasGrandparent owl:propertyChainAxiom (
        :hasParent :hasParent ) .
```


Clé

- Chaque **instance** de la **classe** est **identifiée uniquement par la valeur de la classe clé** :

```
Class: Person
   HasKey: hasSSN
```

- **Exemple syntaxe XML :**

```
<owl:Class rdf:about="Person">
   <owl:hasKey rdf:parseType="Collection">
     <owl:ObjectProperty rdf:about="hasSSN"/>
    </owl:hasKey> </owl:Class>
```

- **Syntaxe Turtle :**

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


Annotation (1)

- Elle décrit pas le domaine, mais parle de la **description**
- Elle associe une **paire propriété-valeur** à une partie de l'ontologie
- Elle ne fait **pas partie de la logique de l'ontologie**

```
Class: Person
  Annotations:
    rdfs:comment "Represents the set of all people."
```

Annotation sur la définition de sous-classe :

```
Class: Man
   SubClassOf:
    Annotations:
      rdfs:comment "States that every man is a person."
    Person
```

Annotation (2)

Source : https://www.w3.org/TR/2009/WD-owl2-primer-20090611/

- **Exemple syntaxe XML :**

```
<owl:Class rdf:about="Man">
   <rdfs:subClassOf rdf:resource="Person"/>
</owl:Class>
<owl:Axiom>
   <owl:annotatedSource rdf:resource="Man"/>
   <owl:annotatedProperty
      rdf:resource="&rdfs;subClassOf"/>
   <owl:annotatedTarget rdf:resource="Person"/>
   <rdfs:comment>States that every man is a person.
   </rdfs:comment>
</owl:Axiom>
```

- **Syntaxe Turtle :**

```
:Man rdfs:subClassOf :Person .
 [] rdf:type       owl:Axiom ;
    owl:annotatedSource    :Man ;
    owl:annotatedProperty rdfs:subClassOf ;
    owl:annotatedTarget    :Person ;
    rdfs:comment "States that every man is a person."^^xsd:string .
```


Gestion de l'ontologie

```
Ontology: <http://example.com/owl/families>
Prefix: : <http://example.com/owl/families/>
Prefix: xsd: <http://www.w3.org/2001/XMLSchema#>
Prefix: owl: <http://www.w3.org/2002/07/owl#>
Prefix: otherOnt:
<http://example.org/otherOntologies/families/>
Import: <http://example.org/otherOntologies/families.owl>

SameIndividual: John, otherOnt:JohnBrown
SameIndividual: Mary, otherOnt:MaryBrown
EquivalentClasses: Adult, otherOnt:Grownup
EquivalentProperties: hasChild, otherOnt:child
EquivalentProperties: hasAge, otherOnt:age
```


5. Profils de OWL2

- **OWL 2-EL**
- **OWL 2-QR**
- **OWL 2-RL**

OWL et les logiques de description (LD)

Chaque sous-langage ou profil OWL est basé sur une logique de description (LD) particulière, orientée vers un objectif particulier:

- **OWL 1 et OWL2 languages et sous-languages:**
  - **OWL1 DL** est basé sur la LD *SHOIN*.
  - **OWL1 Lite** est basé sur la LD *SHIF*.
  - **OWL1 Full** est basé sur une logique *indecidable*, *un sur ensemble of SHOIN*
  - **OWL2 (DL)** est basé sur la LD *SROIQ*, *et est orienté vers l'élaboration d'ontologies avec un haut niveau d'expressivité*

Profils de OWL 2

- Compromis entre expressivité et calculabilité
- Sous-langages de OWL 2 se différenciant par :
  - **Des propriétés computationnelles** (raisonnement en Logspace à Ptime)
  - **Des possibilités d'implantation** (e.g. avec un langage de requête de BD)
- **Profils OWL 2:**
  - **OWL2 EL** est basé sur la LD EL++
  - **OWL2-QL** est basé sur la LD DL-Lite
  - **OWL2 RL** est basé sur Description Logic Programs (DLP)

OWL 2 - EL

- **EL pour « Existential Languages »**
- Vise les **grandes ontologies** (biomédicales)
  - descriptions structurales complexes
    - configurations de systèmes
    - inventaires de produits
    - domaines scientifiques (SNOMED)
  - grand nombre de classes
  - traite de grandes quantités de données
- Correspond à la famille de logique de description EL (ne permet que la quantification existentielle)
- algorithmes polynomiaux pour vérifier la satisfiabilité, classification, vérification d'instances
- ne peut utiliser (car l'intersection des types doit être vide ou infinie) :
  `owl:allValuesFrom, owl:unionOf, owl:complementOf`
  `xsd:int, xsd:byte, xsd:double`

OWL 2 - QL

- **QL pour « Query Language »**
- Vise les **ontologies simples** avec un grand nombre d'entités (p.e. thesaurus)
- Pouvoir d'expression similaire à celle des schémas entités-relation ou UML
- Intégration possible avec les BD relationnelles
- Raisonnement possible à l'aide de réécriture de requêtes SQL

OWL 2 - RL

- **RL pour « Rule Language »**
- Equivalent à RDF *enrichi* avec des règles
- Le plus expressif des profils existants
- Quelques limites sur l'expressivité pour espérer garder une certaine efficacité
- Raisonnement à l'aide de systèmes de règles

Outils

OWL-API : interface Java

Éditeur d'ontologie :

- **Protégé** (Univ. Stanford) et plug-ins associés : Visualisation avancée (Graphviz), Jambalaya, OWL-S plugin pour la modélisation et l'exécution de processus, Interface DIG pour moteur de raisonnement …
- **SWOOP** (repris par Google en 2007)
- **POWL** et **Ontowiki** (en PHP sur le WEB – manque de maturité mais en progression)
- TopBraid Composer (Commercial)
- ..

Moteurs d'inférences (ou de raisonnement) :

- **Pellet**
- Fact++ (Manchester)
- KAON2
- **RacerPro** (propriétaire et très cher)
- **Jess** (propriétaire mais licences académiques)
- Bossam
- …

6. Limites de OWL1 & OWL2

Limits of OWL1 & OWL2

- **OWL1 (DL)** :
  - ne peut pas exprimer la relation "*uncle*" (chaîne des relations *parent* et de *frères* et *sœurs - sibling)*
- **OWL2 (DL)** :
  - = fragment decidable de la logique des prédicats du 1er ordre,
  - + quelques extensions décidables qui vont au-delà du 1er ordre
  - peut exprimer la relation "*uncle*" avec des chaînes de propriétés (property chains),
  - ne peut pas exprimer les relations entre les individus référencés par les propriétés.

    Ex: pas possible d'exprimer le concept "*enfant de parents mariés*" (*child of married parents)*, car OWL2 ne peut pas exprimer la relation entre les parents de l'individu (cf: A better uncle for OWL).