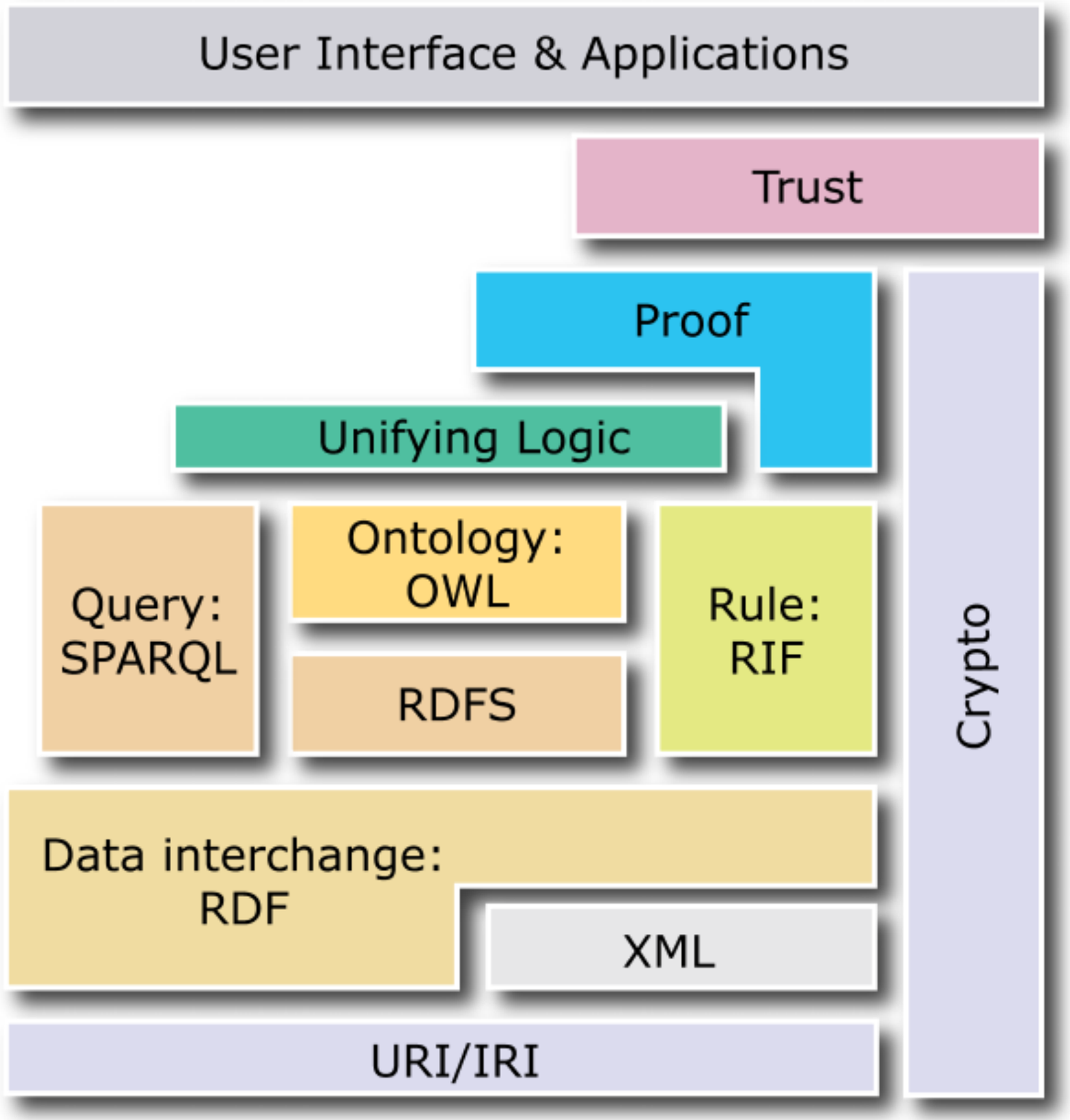
Université de Montréal

OWL 2 - Syntaxe

inspiré du OWL 2 Primer

Guy Lapalme
Université de Montréal
lapalme@iro.umontreal.ca

rali


User Interface & Applications
Trust
Proof
Unifying Logic
Query:
SPARQL
Ontology:
OWL
RDFS
Rule:
RIF
Crypto
Data interchange:
RDF
XML
URI/IRI

rali

OWL : Web Ontology Language

- Représentation de connaissances riches et complexes à propos de
  - *choses*
  - groupes de *choses*
  - relations entre *choses*
- Basé sur une logique *calculatoire* permettant
  - vérifier la consistance des connaissances
  - expliciter des connaissances implicites
- Documents OWL peuvent être liés entre eux

rali

Principes

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

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

rali

Modélisation des connaissances

- **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**

rali

Représentation de 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

rali

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

rali

Entités

NOM OWL	EXEMPLES	NOM COMMUN
individu	Jean, Marie, cette table	objet
classe	homme, femme	catégorie
propriété		relation
objet-objet	mariéÀ, pèreDe	
objet-valeur	âge	
annotation	auteur, date de création	

Expression

- Combinaison de noms d'entités avec des **constructeurs** pour de nouvelles entités
  - combine *femme* et *professeur* pour obtenir la classe des *professeures*
- Langage riche pour la combinaison de classe
- Peu de combinaison de propriétés

rali

Structure de OWL-2

- Structure de l'ontologie ⇔ graphe RDF
- Sémantique
  - sémantique directe de la structure
  - sémantique basée sur RDF
- Syntaxes
  - 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

rali

Structure de OWL-2

- Structure de l'ontologie ⇔ graphe RDF
- Sémantique
  - sémantique directe de la structure
  - sémantique basée sur RDF
- Syntaxes
  - 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

rali

Plan

Plan

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

rali

Plan

- Modélisation de base
  - Classes et instances
  - Hiérarchies de classes
  - Propriétés d'objet
  - Hiérarchies de propriétés
  - Restrictions sur les propriétés
  - Égalité des individus
  - Types des valeurs de propriétés
- Relations avancées de classes
  - Classes complexes
  - Restrictions de propriétés
  - Restrictions de cardinalité
  - Enumérations d'individus

rali

Plan

- Modélisation de base
  - Classes et instances
  - Hiérarchies de classes
  - Propriétés d'objet
  - Hiérarchies de propriétés
  - Restrictions sur les propriétés
  - Égalité des individus
  - Types des valeurs de propriétés
- Relations avancées de classes
  - Classes complexes
  - Restrictions de propriétés
  - Restrictions de cardinalité
  - Enumérations d'individus
- Utilisations avancées de propriétés
  - Caractéristiques des propriétés
  - Chaînes de propriétés
  - Clés

rali

Classes et instances

Notation Manchester (utilisée dans Protégé)
Frame-based (`.omn`)

- Marie est une personne

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

- Marie est une femme

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

Autres syntaxes

Functional-Style Syntax

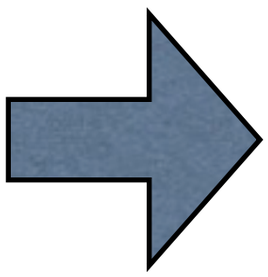
```
ClassAssertion( :Woman :Mary )
```

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>
```

rali

Hiérarchies de classes

- Sous-classes

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

- Classes disjointes

```
DisjointClasses: Woman, Man
```

rali

Autres syntaxes

Functional-Style Syntax

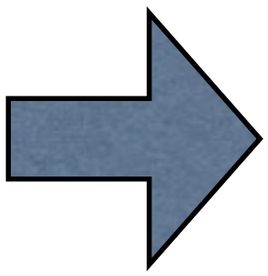
```
SubClassOf( :Woman :Person )
```

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>
```

rali

Propriétés d'objet
(relations entre individus)

- Positive

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

- Négative

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

rali

Propriétés d'objet
(relations 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!

rali

Hiérarchies de propriétés

- semblables à celle pour les classes

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

Restrictions sur les propriétés


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

Égalité et inégalité des individus

OWL ne suppose pas que deux 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
```

rali

Plan

- Modélisation de base
  - Classes et instances
  - Hiérarchies de classes
  - Propriétés d'objet
  - Hiérarchies de propriétés
  - Restrictions sur les propriétés
  - Égalité des individus
  - Types des valeurs de propriétés
- Relations avancées de classes
  - Classes complexes
  - Restrictions de propriétés
  - Restrictions de cardinalité
  - Enumérations d'individus
- Utilisations avancées de propriétés
  - Caractéristiques des propriétés
  - Chaînes de propriétés
  - Clés

rali

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
```

- Expression utilisable à la place d'un nom de classe

```
Individual: Jack
```

Restrictions de propriétés

- Quantification existentielle

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

- Quantification universelle

```
Class: HappyPerson
    EquivalentTo: hasChild only HappyPerson
```

rali

Restrictions de propriétés

- Quantification existentielle

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

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

- Quantification universelle

```
Class: HappyPerson
    EquivalentTo: hasChild only HappyPerson
```

Restrictions de propriétés

- Quantification existentielle

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

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

- Quantification universelle

Une personne heureuse si tous ses enfants sont heureux

```
Class: HappyPerson
    EquivalentTo: hasChild only HappyPerson
```

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
```

Une personne heureuse si tous ses enfants sont heureux

rali

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
```

Une personne heureuse si tous ses enfants sont heureux

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

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
```

- Cardinalité minimum

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

- Cardinalité exacte

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

rali

Restrictions de cardinalité

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

- Cardinalité maximum

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

- Cardinalité minimum

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

- Cardinalité exacte

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

rali

Restrictions de cardinalité

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

- Cardinalité maximum

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

- 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 }

Plan

- Modélisation de base
  - Classes et instances
  - Hiérarchies de classes
  - Propriétés d'objet
  - Hiérarchies de propriété
  - Restrictions sur les propriétés
  - Égalité des individus
  - Types des valeurs de propriétés
- Relations avancées de classes
  - Classes complexes
  - Restrictions de propriétés
  - Restrictions de cardinalité
  - Enumérations d'individus
- Utilisations avancées de propriétés
  - Caractéristiques des propriétés
  - Chaînes de propriétés
  - Clés

rali

Caractéristiques des propriétés - I

- Inverse

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

- Symétrie

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

- Asymétrie

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

- Disjonction

```
DisjointProperties: hasParent, hasSpouse
```

rali

Caractéristiques des propriétés - 2

- Reflexivité

```
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
```

Clé

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

```
Class: Person

    HasKey: hasSSN
```

Annotation

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

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

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

Annotation sur la définition de sous-classe

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/famili

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
  ...
```

Sémantiques des ontologies

- OWL 2 DL
  - *direct model semantics*
  - correspond au modèle en logique de description
- OWL 2 RDF
  - extension de la sémantique de RDFS
  - considère l'ontologie comme un graphe RDF
  - indécidable!!!
- différences assez mineures dans la majorité des cas

rali

Profils

- compromis entre expressivité et calculabilité
- sous-langages de OWL 2
  - propriétés computationnelles (raisonnement en LOGSPACE à PTIME)
  - possibilités d'implantation (e.g. avec un langage de requête de BD)

rali

OWL 2 - EL

- 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)

rali

OWL 2 - EL

EL: 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)

rali

OWL2 - EL (suite)

- algorithmes polynomiaux pour
  - vérifier la satisfiabilité, classification, vérification d'instances
- ne peut utiliser
  - `owl:allValuesFrom, owl:unionOf, owl:complementOf`
  - `xsd:int, xsd:byte, xsd:double`

OWL2 - EL (suite)

- algorithmes polynomiaux pour
  - vérifier la satisfiabilité, classification, vérification d'instances
- ne peut utiliser
  - `owl:allValuesFrom, owl:unionOf, owl:complementOf`
  - `xsd:int, xsd:byte, xsd:double`

car l'intersection des types doit être vide ou infinie

OWL 2 - QL

- 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 à l'aide de réécriture de requêtes SQL

rali

OWL 2 - QL

QL : 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 à l'aide de réécriture de requêtes SQL

rali

OWL 2 - RL

- *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

rali

OWL 2 - RL

RL : Rule Language

- *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

rali

Outils

- OWL-API : interface Java
- Editeurs d'ontologie
  - Protégé, SWOOP
  - TopBraid Composer (Commercial)
- Raisonneurs
  - Fact++ (Manchester)
  - Pellet
  - RacerPro

rali