SWRL

Semantic Web Rule Language (SWRL)

- Document W3C : SWRL : A Semantic Web Rule Language Combining OWL and RuleML, W3C Member Submission, 21 May 2004
- Limitations de OWL2 : raisonnement LD -> nécessité de règles allant au-delà
- Permet d'écrire des règles exprimées en termes d'entités OWL2 (classes, propriétés), de raisonner sur des individus et d'inférer de nouvelles connaissances
- Supporté par certains outils (dont Protégé)

SWRL : syntaxe

- Règles SWRL de la forme : A1 ^ A2 ... ^An -> B1 ^ B2 ... ^ Bm
- Ai et Bj axiomes (prédicats) n-aires
- Variables : commencent par un ?, exemple : ?x, ?X
  - quantifiées universellement
  - portée d'une variable : règle dans laquelle elle apparaît
- Littéraux (constantes) : typés
- Exemples
  - `Person(?p) ^ hasCar(?p, true) ->  Driver(?p)`
  - `Publication(?a) ^ hasAuthor(?x,?y) ^ hasAuthor(?x,?z) ^ differentFrom(?y,?z) -> cooperatedWith( ?y, ?z)`
  - `Person( ?p)^ hasAge(?p,?age)^ swrlb :greaterThan(?age, 17) -> Adult( ?p)`
  - `Publication(?p), (hasAuthor = 1)(?p) → SingleAuthorPublication( ?p)`

SWRL : axiomes

- Classe(RefIndividu)
  Classe : soit une classe nommée soit une description de classe
  RefIndividu : soit variable soit individu nommé
  Ex : Personne(jocaste), (hasChild >= 1)(?x)
- ObjectProperty(RefIndividu1, RefIndividu2)
  ObjectProperty : nom de propriété
  RefIndividu1, RefIndividu1: soit variable soit individu nommé
  Ex : hasBrother(?x, ?y)hasSibling(Fred, ?y)
- DataProperty(RefIndividu, Valeur)
  RefIndividu : soit variable soit individu nommé
  Valeur : valeur d'un type de données
  Ex : hasAge(?x, ?age), hasHeight(Fred, ?h), hasAge(?x, 232), hasName(?x, « Fred″)
- Intervalles de données (Data Range atom)
  xsd:int(?x) : ?x est un entier
  [3, 4, 5](?x) : ?x peut prendre l'une des 3 VALEURS
  xsd:int[>= 18 , <= 65](?age) : ?x est dans l'intervalle spécifié
- sameAs(RefIndividu1, RefIndividu2)
- differentFrom(RefIndividu1, RefIndividu2)

SWRL : prédicats prédéfinis

- Espace de nom swrlb
- Pour réaliser des comparaisons : equal, notEqual, lessThan, lessThanOrEqual, greaterThan, greaterThanOrEqual
  Ex : swrlb:greaterThan(?age, 17)
- Pour réaliser des opérations mathématiques : add, subtract, multiply, divide, integerDivide, mod, powSatisfied...
  vrai si ?A1 est le résultat de l'opération sur les arguments ?A2 et ?A3
  Ex : swrlb:multiply(?aire, ?long, ?larg)
- Pour travailler sur des chaînes de caractères : upperCase, lowerCase, contains, containsIgnoreCase, startWith, endWith...
  Ex :s swrlb:tartsWith(?telephone, '+')
- Pour travailler sur les booléens, les dates (« Date, Time, Duration »), les URIS, les listes

SWRL : remarques

- Impossible de modifier et de supprimer des informations
  Ex : Driver(?d) ^ hasAge(?d,?age) ^ swrlb :greaterThan(?age, 25)
  -> isInsurable(?d, true)
- Hypothèse du monde ouvert
- Pas d'hypothèse de nom unique
- Pas de négation mais usage de négation dans définition des classes
  (owl :complementOf)
  Ex : (not A) (?x)
- Pas de disjonction mais usage de la disjonction dans la définition des classes
  Ex : (A or B) (?x)

SWRL : example (chaînage avant) (1)

```
CurryThai ⊑ ContientArachide
T ⊑ ∀commande.Plat
AllergiqueArachide(sebastien)
CurryThai(curryThai1)
commande(sebastian,curryThai1)
AllergiqueArachide(?x) ^ ContientArachide(?y) -> aimePas(?x,?y)
commande(x,y) ^ aimePas(x,y) ->  Malheureux(x)
```

```
CurryThai ⊑ ContientArachide
T ⊑ ∀commande.Plat
AllergiqueArachide(sebastien)
CurryThai(curryThai1)
commande(sebastian,curryThai1)
AllergiqueArachide(?x) ^ ContientArachide(?y) -> aimePas(?x,?y)
commande(x,y) ^ aimePas(x,y) ->  Malheureux(x)
_________________
Plat(curryThai1)
```


SWRL : example (chaînage avant) (2)

```
CurryThai ⊑ ContientArachide
T ⊑ ∀commande.Plat
AllergiqueArachide(sebastien)
CurryThai(curryThai1)
commande(sebastian,curryThai1)
AllergiqueArachide(?x) ^ ContientArachide(?y) -> aimePas(?x,?y)
commande(x,y) ^ aimePas(x,y) ->  Malheureux(x)
_________________
Plat(curryThai1)
aimePas(sebastien, curryThai1)
```

```
CurryThai ⊑ ContientArachide
T ⊑ ∀commande.Plat
AllergiqueArachide(sebastien)
CurryThai(curryThai1)
commande(sebastian,curryThai1)
AllergiqueArachide(?x) ^ ContientArachide(?y) -> aimePas(?x,?y)
commande(x,y) ^ aimePas(x,y) ->  Malheureux(x)
_________________
Plat(curryThai1)
aimePas(sebastien, curryThai1)
Malheureux(sebastien)
```

SQWRL : SWRL et requêtes

- SWRL est un langage de règle pas d'interrogation
- Cependant, le corps d'une règle peur-être vu comme une spécification de « pattern matching » (donc une requête)
- Développement de SQWRL
  - Corps de la règle : identique à SWRL
  - Tête de la règle : contient select, order by
- Ex : Personne(?p) ^ estAgéDe(?p,?a) ^ swrl:greaterThan(?a,17) -> swrl:select(?p) ^sqwrl:orderBy(?a)

Conclusion

- SWRL est plus expressif que OWL DL mais ceci aux dépens de la décidabilité des raisonneurs
  - transformation des règles en règles DL-safe (restrictions sur les structures des règles, les variables...)
- OWL vs SWRL
  - OWL : raisonnement dit « taxonomique »
  - SWRL : raisonnements, sur les données, pouvant être plus complexes
- Si possible, privilégier l'expression en OWL plutôt qu'en SWRL
- RIF : Rule Interchange Format