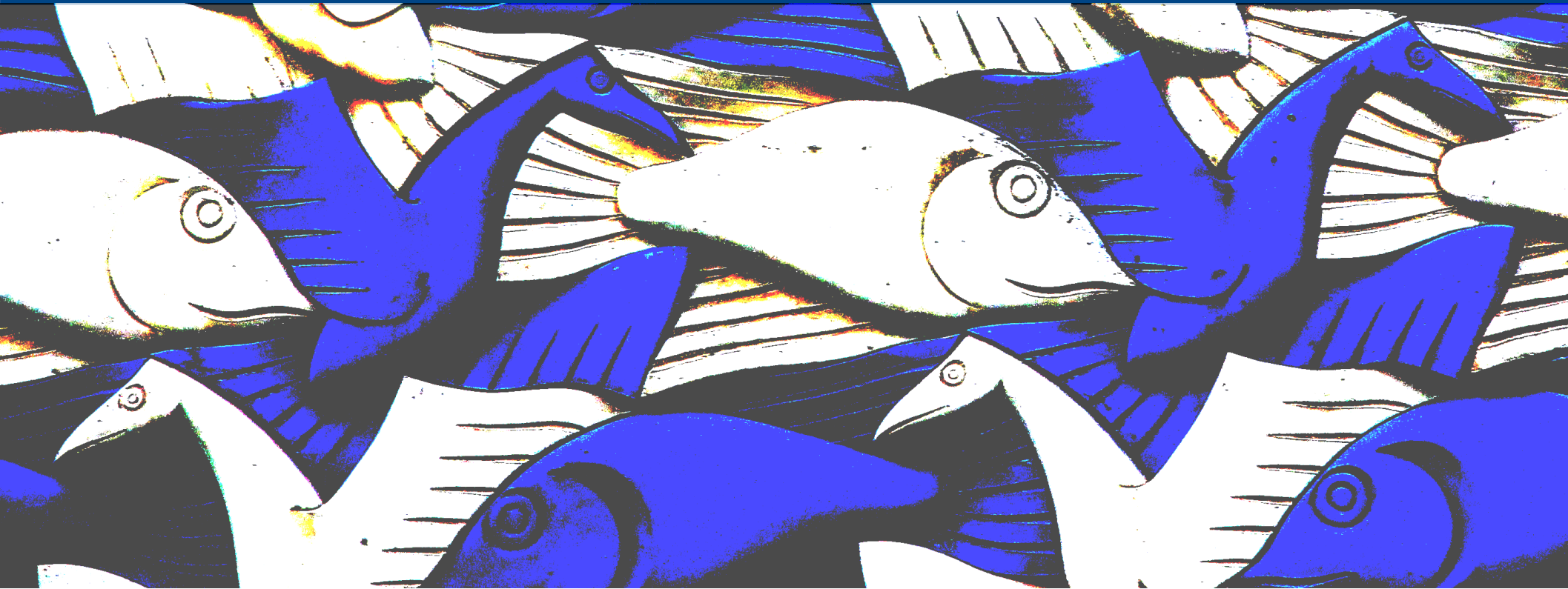


# Logique et WEB sémantique II



Raisonnement et science de la décision

Master M1 MIAGE – Ingénierie Métier

Université Toulouse 1 Capitole

**Umberto Grandi**

# Logique propositionnelle : rappel

On voulait utiliser des connaissances pour raisonner sur des données :

Une **base de connaissance** est une liste d'énoncés qui forme une représentation du monde lisible par un agent-logiciel.











La **logique propositionnelle** nous permet de décrire des faits et des propriétés avec un langage binaire de vérité/fausseté.

Vocabulaire :  $\{P, Q, R, S...\}$   
Énoncés construits avec connecteurs logiques :  $\neg \vee \wedge \rightarrow \leftrightarrow$

On veut être capables d'exprimer des **relations** entre objets et les catégoriser !  
Il nous faut **un langage plus puissant** : le premier ordre.

# Le point de départ : un site de comparaison d'offres

## GALAXY S6 SANS ABONNEMENT:

	Cdiscount - Samsung GALAXY S6 Livraison: 2.00€	387.99€ 	<a href="#">VOIR L'OFFRE</a>
	Fnac - Samsung GALAXY S6 Livraison gratuite	413.00€ 	<a href="#">VOIR L'OFFRE</a>
	Amazon - Samsung Galaxy S6 Livraison: 9.99€	392.52€ 	<a href="#">VOIR L'OFFRE</a>
	Boulanger - Samsung GALAXY S6 Livraison gratuite	599.00€ 	<a href="#">VOIR L'OFFRE</a>
	Darty - Samsung GALAXY S6 Livraison gratuite	599.00€ 	<a href="#">VOIR L'OFFRE</a>

ung GALAXY S6	1.00€ 	<a href="#">VOIR L'OFFRE</a>
GALAXY S6	1.00€ 	<a href="#">VOIR L'OFFRE</a>
GALAXY S6	549.90€ 	<a href="#">VOIR L'OFFRE</a>
<a href="#">Next</a>		

## ARTICLES PAR CATEGORIES

ACTUALITÉ

CARACTÉRISTIQUES

ACCESSOIRES

PRIX

# Comparer les prix d'un smartphone

On démarre avec la **requête** d'un utilisateur:

- Simple : smartphone Nexus 6s 2016
- Complexe : smartphone 5g moins de 300 euros



L'**environnement** de notre agent-logiciel comparateur d'offres est tout le WEB :

- Plus facile que développer une automobile autonome pour une ville
- Mais équivalent à la quantité d'information dans le monde réel

D'abord on devra créer une **liste des boutiques** et leur adresses WEB:

$Amazon \in BoutEnLigne \wedge PageAccueil(Amazon, \text{« amazon.fr »})$

$Exemple \in BoutEnLigne \wedge PageAccueil(Exemple, \text{« exemple.fr »})$

...

Plusieurs nouveaux symboles à définir :

- $BoutEnLigne$  est un ensemble
- $\in$  est le symbole d'appartenance
- $PageAccueil(x, \text{«yyyy.zz»})$  est une relation ou prédicat

Il faut expliquer à notre agent logiciel comment interpréter ces symboles !

# Le langage des ensembles

Il faudrait commencer avec le **langage des ensemble** :

- Les **objets** sont tous les éléments bien que les ensembles
- On introduit un **prédicat** unaire *Ensemble*
- Bien qu'un prédicat **binaire** d'appartenance  $\in$

On peut donc définir :

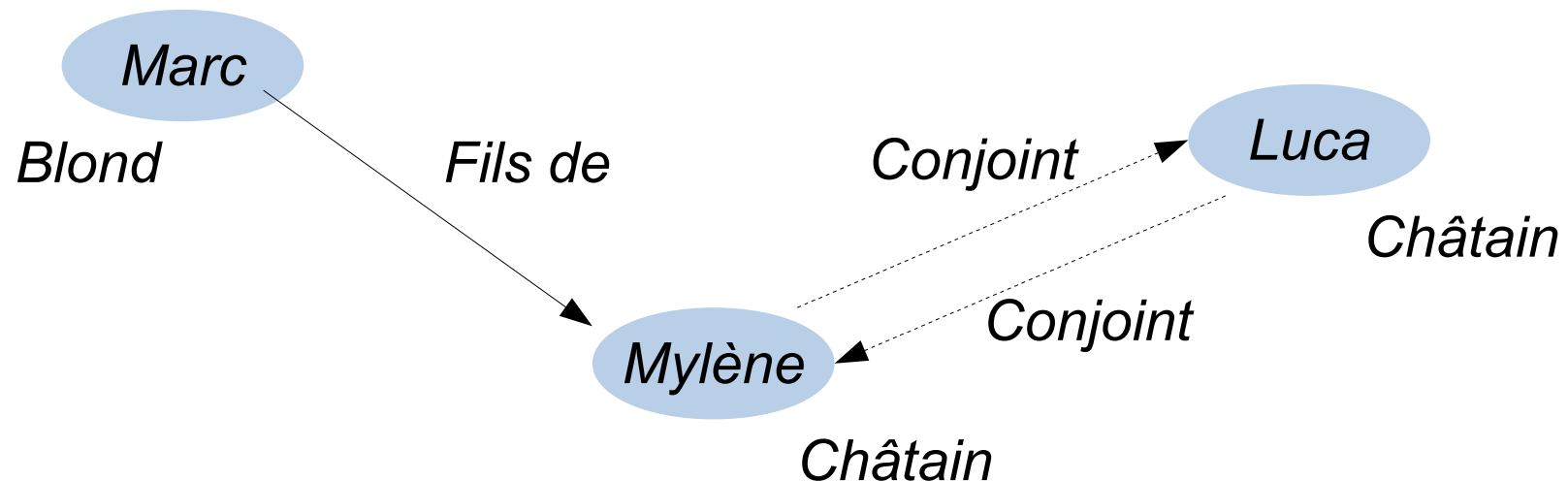
- $A \subset B := \forall x (x \in A \rightarrow x \in B)$   
 $A$  est un sous-ensemble de  $B$  si tous les éléments de  $A$  sont en  $B$ .
- $C = A \cap B := \forall x \ x \in C \rightarrow (x \in A \wedge x \in B)$   
L'intersection de  $A$  et  $B$  contient les éléments qui appartiennent aux deux ensembles au même temps.

Mais qu'est qu'un **prédicat** ?? (exemple :  $\in$ ). Et un objet ? Unaire ? Binaire ? Et les **le symbole**  $\forall$  ? *Sont-ils des concepts propositionnels ?*

# Sémantique du premier ordre

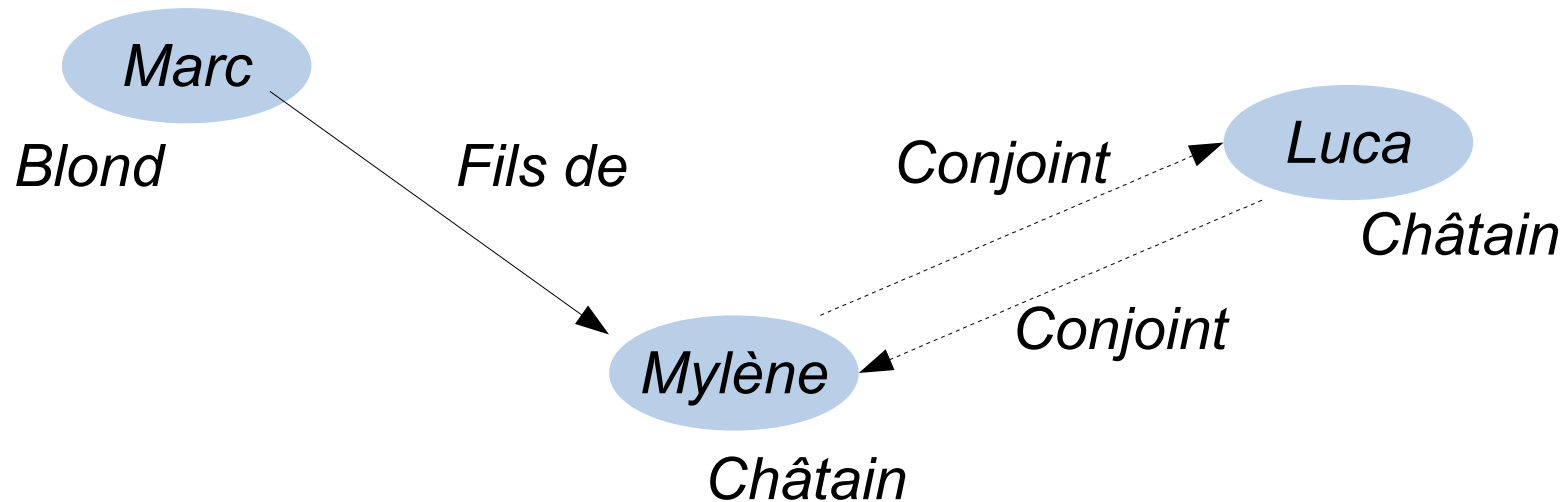
Attention ! On commence avec la sémantique,  
en partant directement des objets et des relations

Un **modèle du premier ordre** est un ensemble  
d'objets (le domaine) et des relations entre ces objets



Il y a **deux relations binaires** : « *Fils de* » et « *Conjoint* »  
et **deux relations unaires** : « *Châtain* » et « *Blond* »

# Sémantique du premier ordre



Le **domaine** est l'ensemble des objets : {Marc, Mylène, Luca}

Les **relations** sont représentés comme ensembles des tuples:

- $Fils\ de = \{(Marc, Mylène)\}$
- $Conjoint = \{(Mylène, Luca), (Luca, Mylène)\}$
- $Châtain = \{Mylène, Luca\}$
- $Blond = \{Marc\}$



# Langage du premier ordre

**Vocabulaire** de base pour construire des énoncés :

Constants  $c_1, c_2 \dots$

Prédicats avec **arité**  $(P_1, n_1), (P_2, n_2) \dots$

*Exemple :* Constants = { Mère }  
Prédicats = { (Filsde, 2), (Conjoint, 2), (Chatain, 1), (Blond, 1) }

Pour **interpréter** le vocabulaire et les énoncés il faut associer à chaque constant un objet et à chaque prédicats une relations

*Exemple :* Mère  $\leftarrow$  Mylène      Conjoint  $\leftarrow$  Conjoint  
Filsde  $\leftarrow$  Filsde      Blond  $\leftarrow$  Blond  
Chatain  $\leftarrow$  Châtain

Un **modèle** est donc un **domaine** avec une **interprétation**

# La construction des énoncé simples

Un **terme** est soit une **constante** soit une **variable** (comme  $x, y \dots$ ).

Un terme est une expression logique qui renvoie à un objet.

Il n'y a pas assez des constants pour tous objets, le langage serait trop large !

Un **énoncé atomique** ou **atome** exprime des faits concernant une ou plusieurs relations et des termes (comme dans le langage propositionnel)

*Exemples :  $Filsde(Marc, Mylène)$ ,  $Conjoint(Mylène, Marc)$ ,  $Blond(Luca)$*

Un atome est **vrai** dans un modèle si la relation associé au symbole de prédicat s'applique aux objets auxquels renvoient les termes

On peut donc contrôler la vérité/fausseté des énoncés atomiques

*Exemples : le premier énoncé ci-dessus est vrai, les autres faux*

# Énoncés complexes et quantificateurs

On peut utiliser tous **connecteurs logiques** pour combiner des énoncé,  
Bien que le **symbole d'égalité** =

Exemple :  $\neg \text{Conjoint}(\text{Marc}, \text{Mylène})$   
 $\text{Filsde}(\text{Marc}, \text{Mylène}) \wedge \text{Conjoint}(\text{Marc}, \text{Mylène})$

Mais aussi le **quantificateur universel**  $\forall$  « pour tout ... »

$\forall x, y \text{ Conjoint}(x, y) \rightarrow \text{Conjoint}(y, x)$   
« Pour tous x et y, si x est conjoint de y alors y est conjoint de x »

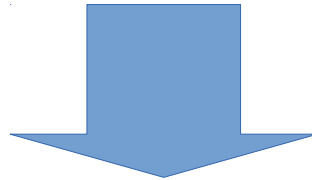
Et le **quantificateur existentiel**  $\exists$  « il existe ... tel que ... »

$\exists x \text{ Filsde}(x, \text{Mylène})$   
« Il existe un x tel que x est fils de Mylène »

# Sémantique du quantificateur universel

Pour décider la vérité d'un énoncé avec **quantificateur universel** il faut tester la vérité de l'énoncé sur tous éléments du domaine

$$\forall x \forall y [Conjoint(x,y) \rightarrow Conjoint(y,x)]$$



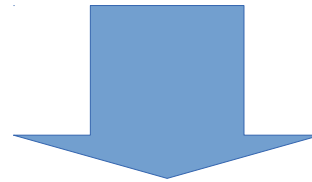
- $Conjoint(Marc, Mylène) \rightarrow Conjoint(Mylène, Marc)$  Vrai ?
- $Conjoint(Marc, Luca) \rightarrow Conjoint(Luca, Marc)$  Vrai ?
- $Conjoint(Luca, Mylène) \rightarrow Conjoint(Mylène, Luca)$  Vrai ?
- $Conjoint(Mylène, Marc) \rightarrow Conjoint(Marc, Mylène)$  Vrai ?
- $Conjoint(Mylène, Luca) \rightarrow Conjoint(Luca, Mylène)$  Vrai ?
- $Conjoint(Luca, Marc) \rightarrow Conjoint(Marc, Luca)$  Vrai ?

*Suggestion : utilisez la table de vérité de l'implication propositionnelle ( $\rightarrow$ )*

# Sémantique du quantificateur existentiel

Pour décider la vérité d'un énoncé avec **quantificateur existentiel** il faut trouver au moins un élément du domaine qui vérifie l'énoncé

$\exists x \text{ Filsde}(x, \text{Mylène})$



- |                          |        |
|--------------------------|--------|
| • Filsde(Mylène, Mylène) | Vrai ? |
| • Filsde(Luca, Mylène)   | Vrai ? |
| • Filsde(Marc, Mylène)   | Vrai ? |

# Deux observations importantes

1)  $(\forall x \exists y)$  n'est pas égal à  $(\exists x \forall y)$

*Exercice : trouvez un contre-exemple*

2)  $(\exists x \text{ Filsde}(x, \text{Mylène}))$  est équivalente à l'énoncé  
 $(\neg \forall x \neg \text{Filsde}(x, \text{Mylène}))$

*Exercice : pouvez-vous le démontrer ?*

# Suivi de liens pour un comparateur d'offres

Retournons à notre exemple initiale.

Après avoir spécifié la liste des boutiques en ligne suivante (déjà une base de connaissance !) :

$Amazon \in BoutEnLigne \wedge PageAccueil(Amazon, \text{« amazon.fr »})$

...

Nous devons être capable d'accéder aux pages spécifiques à notre requête, en définissant une notion de **pertinence**:

$Pertinence(page, requete) :=$   
     $\exists boutique, accueil$   
     $boutique \in BoutEnLigne$   
     $\wedge PageAccueil(boutique, accueil)$   
     $\wedge \exists url \textbf{ChainePertinente}(accueil, url, requete)$   
     $\wedge page = \textbf{Contenu}(url)$

# Chaîne pertinente

La chaîne de liens qu'on doit suivre pour arriver aux offres (de smartphones, de vols...) dépend fortement de la requête qu'on a reçu :

$$\begin{aligned} \text{ChainePertinente}(\text{depart}, \text{arrivee}, \text{requete}) := & (\text{depart} = \text{arrivee}) \\ & \vee (\exists \text{mid } \mathbf{Lien}(\text{depart}, \text{mid}) \wedge \mathbf{TexteLien}(\text{depart}, \text{mid}, \text{texte}) \\ & \wedge \mathbf{NomCategoriePertinente}(\text{requete}, \text{texte}) \\ & \wedge \text{ChainePertinente}(\text{mid}, \text{arrivee}, \text{requete}). \end{aligned}$$

- $\text{Lien}(\text{depart}, \text{mid})$  représente le fait qu'il y a un lien hypertexte entre la page *depart* et la page *mid*.
- $\text{TexteLien}$  nous informe que le liens entre la page *depart* et la page *mid* a comme **texte d'ancre** *texte* (voir le cours de pages WEB !).
- $\text{NomCategoriePertinente}$  nous informe si *texte* est une **expression pertinente** avec notre requête.

*Observation : êtes vous à l'aise avec des définitions récursives ?*



# Texte pertinente

On peut donc définir la pertinence du texte pour une requête :

$$\begin{aligned} \text{NomCategoriePertinente}(\text{requete}, \text{texte}) := \\ \exists c1, c2 \text{ Nom}(\text{requete}, c1) \wedge \text{Nom}(\text{texte}, c2) \\ \wedge (c1 \subset c2 \vee c1 \supset c2) \end{aligned}$$

Le prédicat  $\text{Nom}(\text{« } \text{texte} \text{ »}, c)$  qui nous dit que le *texte* est un nom pour la catégorie *c*. Par exemple,  $\text{Nom}(\text{ordiphone}, \text{smartphone})$ .

*texte* est **pertinent** pour *requete* si leurs noms sont associés à **deux catégories** dont l'une contient l'autre.

Il nous faut une ontologie pour obtenir cette information !

**Une ontologie** est composée d'énoncés dans le langage de la théorie des ensembles, qui donnent une vision complète du domaine spécifique à notre application :

*Livres  $\subset$  Produits*

*Electronique  $\subset$  Produits*

*Telephones  $\subset$  Electroniques*

*Smartphones  $\subset$  Telephones*

...

Ainsi que des énoncés qui listent les noms et les synonymes pour toutes catégories concernées :

*Nom(« livres »,livres)*

*Nom(« electronique »,electronique)*

*Nom(« telephone »,telephone)*

*Nom(« telephones »,telephone)*

*Nom(« téléphone »,telephone)*

# Comparaison des offres

Et ce n'est pas fini ! Une fois les pages pertinentes trouvées :

- Il faut extraire le contenu des pages avec la fonction **Contenu**(url) → *lien avec le cours de pages WEB*
- Extraire les informations importantes des **produits** (poids, camera, processeur...)
- Extraire les **offres** de vente et leur propriétés : le prix est une propriété d'une offre, pas d'un smartphone !
- **Comparer** les offres : pas seulement sur le prix, mais prenant en compte tous les critères → *lien avec le cours de décision*

Complicé , non ?



On a finalement complété notre projet initial :

- En partant des **données** : les pages WEB des boutiques en ligne et leur contenu
- En utilisant des **connaissances** : l'ontologie des catégories des produits
- On a obtenu des **nouvelles connaissances** : on peut trier les pages WEB des boutiques selon leurs pertinence avec la requête initiale

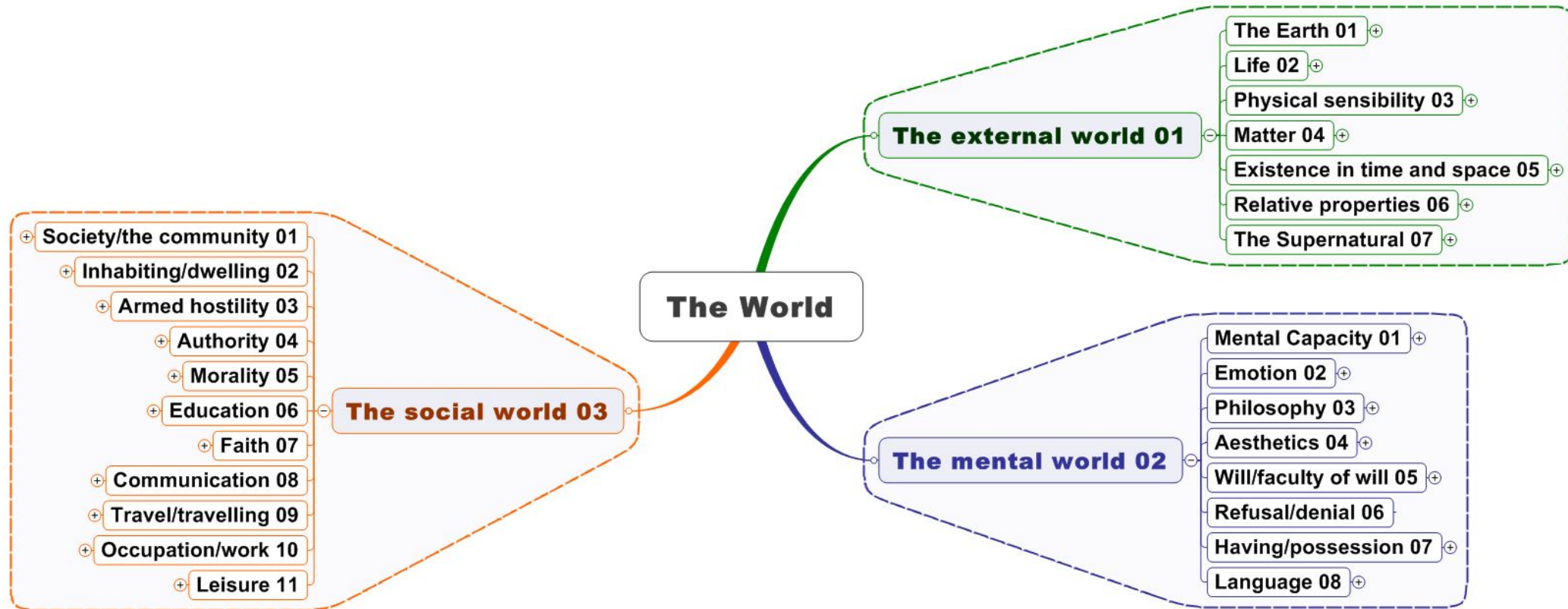
Pourquoi développer des bases des connaissances **spécifiques** pour chaque nouveau domaine d'application ?

Une **ontologie généraliste** tente de comprendre tous ce qu'il y a dans le monde et de le catégoriser. Plusieurs possibilités pour en créer une:

- 1) Appeler une équipe d'**ontologistes** et logiciens (*système CYC, 1990*)
- 2) Extraire les catégories, attributs et valeurs d'une **base de données existante** (*DBPedia, 2007, extrait de Wikipedia*)
- 3) En analysant des **documents de texte** et extrayant des informations (*TEXTRUNNER, 2008, extrait des pages web*)
- 4) Avec du **crowdsourcing**, en demandant à des amateurs de classifier des faits (*OPENMIND, 2002 et 2005*)

Deux possibilité sur quatre sont construites par des logiciels !

# Une ontologie généraliste ou supérieure

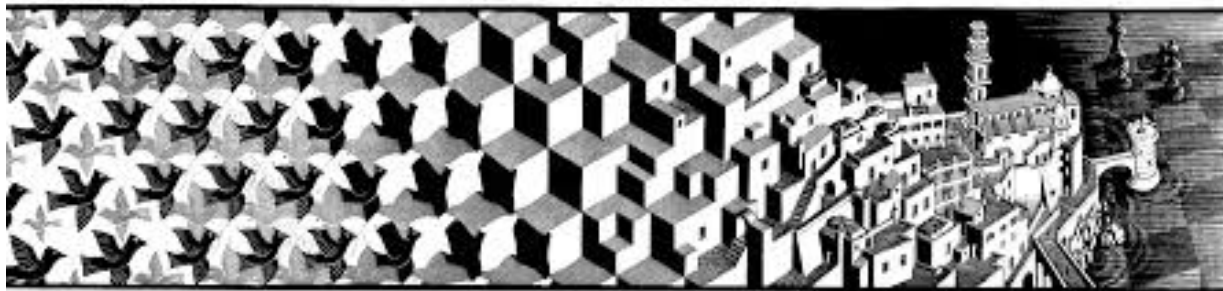


# Conclusions

Ce cours est parti de l'application suivante :  
Comment développer un site de comparaison d'offres ?

On a vite compris qu'il nous faut un langage formel :

- La **logique propositionnelle** exprime bien des faits qui sont **vrai ou faux**  
Mais ce n'est pas assez pour catégoriser.
- La **logique du premier ordre** se fonde sur **objets et relations**  
Elle nous permet de parler des ensembles et donc des catégories...
- Cela nous permet de définir des **ontologies**, des modèles du monde écrits en langage formel pour **raisonner avec les données**.







- *Bases de données* : l'ingénierie des connaissances est fortement liée aux bases des données. En fait, une base de données est un modèle du premier ordre (pensez au schéma entités-associations), et les requêtes SQL peuvent être écrites comme énoncés du premier ordre.
- *Architecture et réseaux* : la logique propositionnelle est utilisée pour spécifier des circuits ou portes. Assigner le symbole de vérité 1 à une variable  $P$  correspond au passage de courant par une certaine porte  $P$ .
- *Conception des sites WEB (semestre II)* : vous allez construire la page WEB d'un magasin en ligne avec base de données associé, un site qui pourrait être consulté par des agent-logiciel en recherche d'offres...



Ce cours est fondé sur les chapitres 7, 8 et 12 de :

- *Stuart Russell et Peter Norvig, Intelligence Artificielle, Pearson, 3eme édition, 2010.*

Regardez aussi le TED talk sur le « linked data » par [Tim Berners Lee](#).

La logique est un sujet à cheval de l'informatique, des mathématiques et de la philosophie. Un des livres de vulgarisation plus fascinants écrits à ce sujet est :

- *Douglas Hofstadter. Gödel, Escher, Bach : Les Brins d'une Guirlande Éternelle. 1979.*

Dans un domaine spécifique on suivra donc le processus suivant :

- 1) *Identifier la tâche* : délimiter la **base des questions** auxquelles on veut répondre, et le type de faits ou observations qui seront disponibles.
- 2) *Collecter les connaissances pertinentes* : si l'on n'est pas experts du domaine, travailler avec des experts pour capturer leur savoir (pas encore dans un langage formel).
- 3) *Décider du vocabulaire des prédicats et des constants* : traduire les concepts importants du domaine dans un langage logique.
- 4) *Encoder les connaissances du domaine* : rédiger des axiomes, des énoncés qui sont vrai dans tous modèles, en traduisant en langage des ensembles les connaissances recueillies au pas 2). Souvent cette tâche suggère des modifications au vocabulaire. Le résultat est une **ontologie**.

..continue :

5)Encoder **la description d'un exemple** du problème : écrire les énoncés atomiques correspondants à un modèle, et contrôler que l'ontologie est complète et bien faite.

6)**Soumettre des requêtes** (en langage formel) et contrôler les réponses. Une base de connaissances n'est qu'une représentation de la réalité, une fois complète on la rendra vivante à l'aide des procédures d'**inférence**. Nous ne verront pas ces procédure dans ce cours, mais pensez au cours de *Base de données* → *Requetes SQL*

7)**Déboguer** la base de connaissances : quand on obtient des réponses incorrectes aux requêtes de test (très souvent!) on corrigera les erreurs dans notre base de connaissance.