

Gestion des connaissances

Introduction (très rapide) à
l'Intelligence Artificielle

E. Tranvouez (erwan.tranvouez@univ-amu.fr)



**Génie Industriel &
Informatique**

Qu'est ce que l'Intelligence Artificielle

2

48

- ◆ A l'origine (60's), cherche à produire des machines capable de reproduire la capacité humaine de résoudre des problèmes
 - => cherche à imiter/émuler le raisonnement humain (=> test de Turing)
- ◆ 2 approches :
 - **Reproduction interne des raisonnements** => le système d'IA (un programme, une machine) reproduit le mécanisme de raisonnement (en manipulant des symboles par exemple et pas seulement des opérations) avec 2 sous-approches qu'on peut grossièrement par la dichotomie micro/macro :
 - ◆ **Macro / cognitive** : on reproduit le raisonnement en terme de mécanisme d'inférence/déduction => influence sciences Psy*
 - ◆ **Micro / connexionnisme** : les comportements intelligents peuvent s'expliquer par des interactions/instructions simples => influence neurosciences
 - **Reproduction externe des raisonnements** => telle une boîte noire seuls importent le résultat, l'approche est pragmatique est cherche à reproduire le résultat => ex. Siri/Ok Google, voiture autonome ?

Retour sur test de Turing: *Imitation Game*

3

48

◆ Hypothèse (simplifiée¹/reformulée²) :

- Une personne interagissant avec une machine (via texte) peut elle la confondre avec un être humain ?
- Si oui, alors cette machine serait *intelligente*.

◆ La machine nécessiterait alors :

- Traitement de la langue naturelle
- Représentation des connaissances
- Raisonnement automatique
- Capacité d'apprentissage

1. À l'origine, 3 joueurs A (♂), B (♂|♀) et C(♀). A est séparé des 2 autres joueurs. A doit deviner qui est ♂ et qui est ♀ en les interrogeant textuellement. => que se passe t'il si A est remplacé par une machine ?
2. Par A.C. Clarke ? => une personne interrogeant (textuellement) une machine saura t'il décider si elle est humaine ou non ?

Domaines d'application de l'IA

- ◆ Diagnostic
- ◆ Planification
- ◆ Contrôle automatique (conduite)
- ◆ Robotique
- ◆ Jeux (échecs => IBM Deep Blue)
- ◆ Reconnaissance de forme (=> sémantique)
- ◆ Classification automatique
- ◆ Modèles émotionnels !

Techniques dites IA

- ◆ Systèmes d'Inférence, Raisonnement logique
- ◆ Recherche heuristiques dans des graphes
- ◆ Réseau de neurones (*=> deep learning*)
- ◆ Traitement du signal ?
- ◆ Méthodes statistiques (*machine learning, Big Data*)
- ◆ Etc.

Is SIRI Real ?

<https://www.youtube.com/watch?v=7-SVvtxHJGU>

Représentation des connaissances : retour sur les ontologies

6

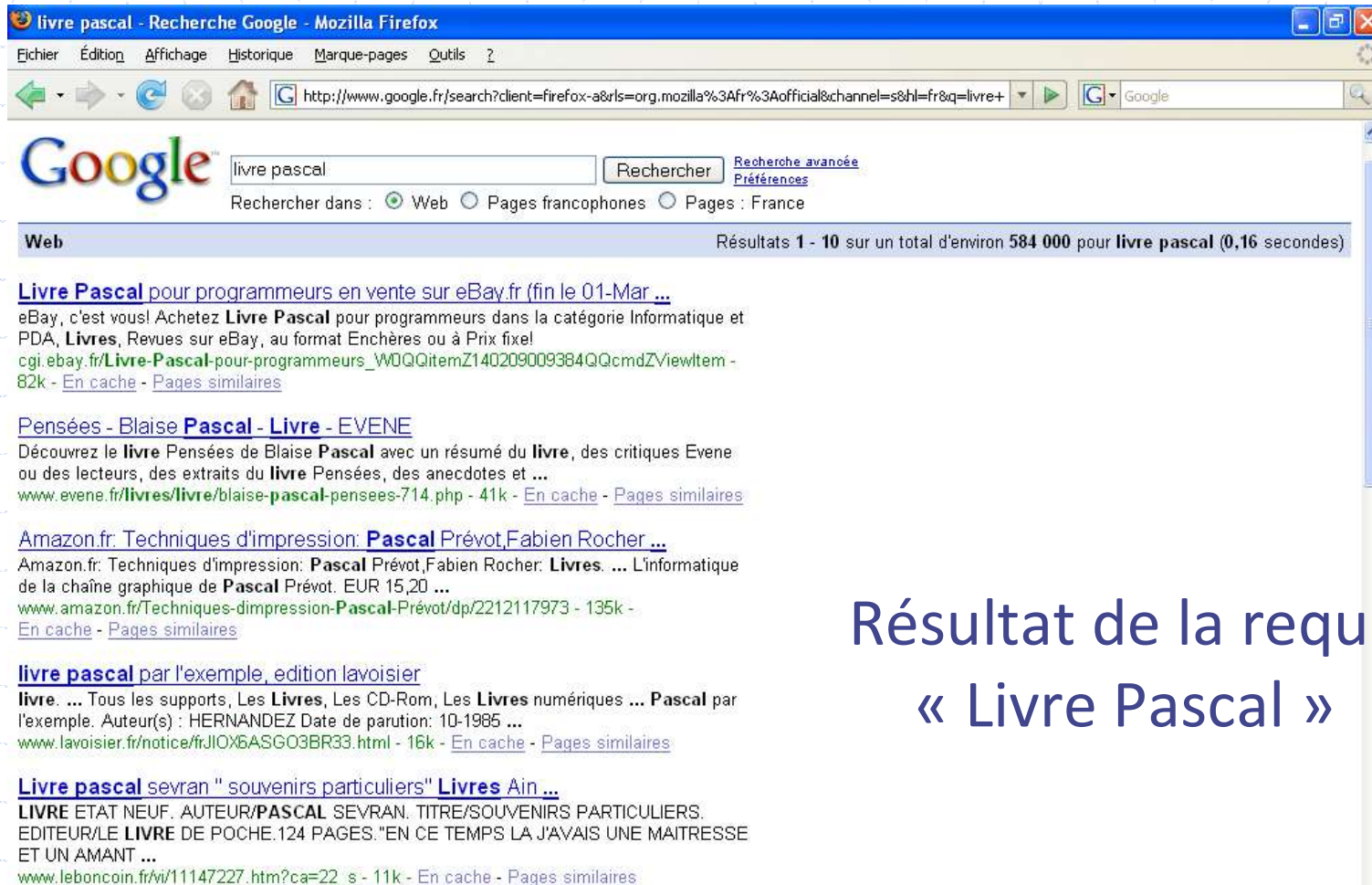
48

- ◆ En relation avec le (futur ?) web sémantique, le W3C a proposé plusieurs langages de description dont :
 - RDF : Ressource Description Framework
 - OWL : Web Ontology Language
- ◆ Ces langages s'appuient sur XML (et XML Schéma pour valider les fichiers)

Zoom sur le web sémantique

- ◆ Étendre le web actuel et notamment comment les informations sont stockées (*HTML essentiellement*) et accédées (*moteur de recherche à base de mots clés + autres heuristiques*).
- ◆ Passage de l'information à la connaissance (*donc contextualisation & conceptualisation de l'information*)
- ◆ XML apporte une façon de représenter les informations qui ouvre la voie à des requêtes plus évoluées : RDF (cf. tr. suivants)
- ◆ Contre exemple (*caricatural*) :
 - ◆ Recherche de « Livre pascal » (cf. résultat ci après)
 - ◆ La représentation des informations via XML permettrait des recherches plus précises en reformulant :
 - Livre dont l'auteur est pascal
 - Avec l'ajout d'ontologie ie un dictionnaire de synonyme au niveau des concepts permettrait d'étendre la requête : Livre, Bouquin, Ouvrage etc...
 - L'idée sous jacente : revenir au saint Graal de l'informatique : le traitement du langage naturel...

Applications dans le web...



The screenshot shows a Mozilla Firefox browser window with the title "livre pascal - Recherche Google - Mozilla Firefox". The address bar contains the URL "http://www.google.fr/search?client=firefox-a&rls=org.mozilla%3Afr%3Aofficial&channel=s&hl=fr&q=livre+&". The search bar contains the text "livre pascal" and the search button is labeled "Rechercher". Below the search bar, there are radio buttons for "Web", "Pages francophones", and "Pages : France". The search results are displayed under the heading "Web" and show "Résultats 1 - 10 sur un total d'environ 584 000 pour livre pascal (0,16 secondes)".

livre pascal pour programmeurs en vente sur eBay.fr (fin le 01-Mar ...
eBay, c'est vous! Achetez **Livre Pascal** pour programmeurs dans la catégorie Informatique et PDA, **Livres**, Revues sur eBay, au format Enchères ou à Prix fixe!
cgi.ebay.fr/Livre-Pascal-pour-programmeurs_W0QQitemZ140209009384QQcmdZViewItem-82k - [En cache](#) - [Pages similaires](#)

Pensées - Blaise **Pascal** - **Livre** - EVENE
Découvrez le **livre** Pensées de Blaise **Pascal** avec un résumé du **livre**, des critiques Evene ou des lecteurs, des extraits du **livre** Pensées, des anecdotes et ...
www.evene.fr/livres/livre/blaise-pascal-pensees-714.php - 41k - [En cache](#) - [Pages similaires](#)

Amazon.fr: Techniques d'impression: **Pascal** Prévot,Fabien Rocher ...
Amazon.fr: Techniques d'impression: **Pascal** Prévot,Fabien Rocher: **Livres**. ... L'informatique de la chaîne graphique de **Pascal** Prévot. EUR 15,20 ...
www.amazon.fr/Techniques-dimpression-Pascal-Prévot/dp/2212117973 - 135k - [En cache](#) - [Pages similaires](#)

livre pascal par l'exemple, edition lavoisier
livre. ... Tous les supports, Les **Livres**, Les CD-Rom, Les **Livres** numériques ... **Pascal** par l'exemple. Auteur(s) : HERNANDEZ Date de parution: 10-1985 ...
www.lavoisier.fr/notice/fr/JIOX6ASGO3BR33.html - 16k - [En cache](#) - [Pages similaires](#)

Livre pascal sevrans "souvenirs particuliers" **Livres** Ain ...
LIVRE ETAT NEUF. AUTEUR/**PASCAL** SEVRAN. TITRE/SOUVENIRS PARTICULIERS. EDITEUR/LE **LIVRE** DE POCHE.124 PAGES."EN CE TEMPS LA J'AVAIS UNE MAITRESSE ET UN AMANT ...
www.leboncoin.fr/vi/11147227.htm?ca=22_s - 11k - [En cache](#) - [Pages similaires](#)

Résultat de la requête
« Livre Pascal »

Zoom sur le web sémantique

◆ Devant la masse d'information à traiter nécessite des processus automatiser d'indexation, recherche d'information (voire connaissance).

◆ Ex. d'applications :

- *Annotation de document*: ie enrichir l'indexation et donc augmenter les documents trouvés les plus pertinents lors d'une recherche
- *Définition d'un référentiel (ontologie)*: nécessaire pour définir les mots possibles pour annoter et rechercher les documents
- ...

RDF : un support pour le contenu

10

48



RDF pour Ressource Description Framework <http://www.w3.org/RDF>

- ❑ Base du Web Sémantique
- ❑ Conçu pour décrire un contenu (=> métadonnées = sens de certaines balises)
- ❑ Exprimé en XML, afin de permettre un traitement automatique (=> moteur recherche)



Prédicats : les expressions RDF servent à caractériser des ressources (par ex. page web) :

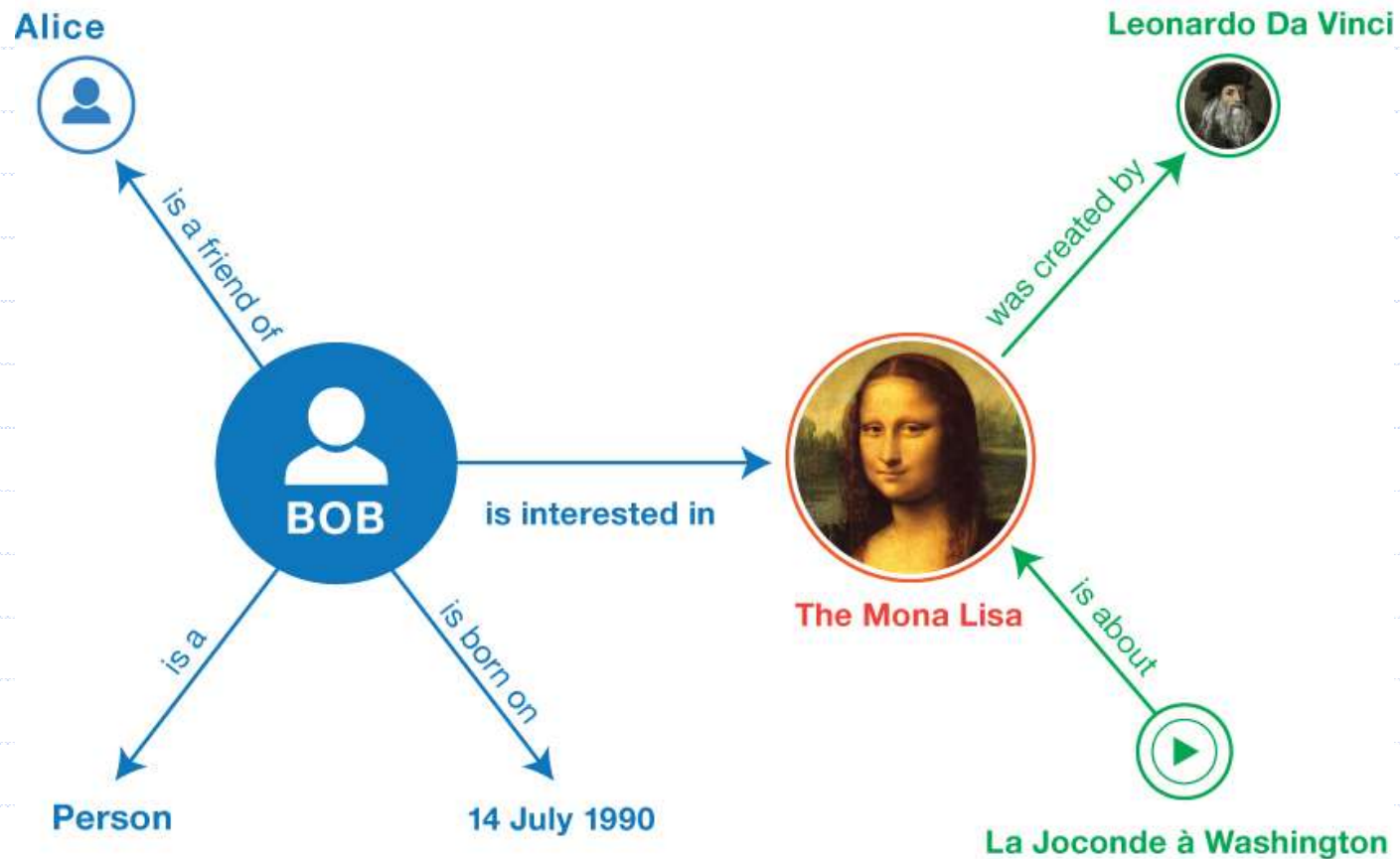
- ❑ Au travers d'affirmations (*statements*) exprimées par la formule générale :
- ❑ => Sujet Prédicat > Objet
- ❑ Sujet & Object devront être identifiés => URI (Unified Ressource Identifier)
- ❑ On peut faire un // avec modélisation OO avec Objet/Attributs/Valeurs
- ❑ Ex : Eugénio (sujet)
estAuteurDe (prédicat)
<http://xxx> (objet)

```
<rdf:Description about="http://www.grapa.net/peugeot404/">  
  <s:Creator>Eugenio</s:Creator>  
</rdf:Description> </rdf:RDF>
```



=> Ensemble des affirmations représentable par des graphes avec des arcs orientés

Exemple de graphes représentant un schéma RDF



Exemple de graphes représentant un schéma RDF

12

48

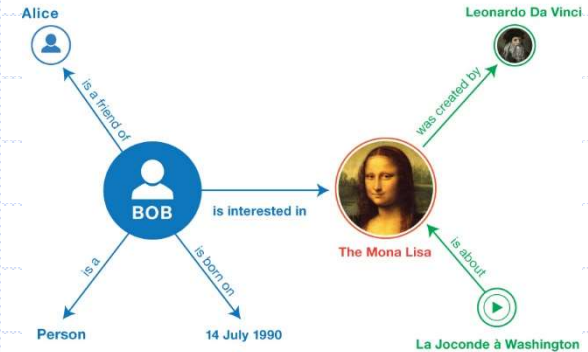
◆ On peut alors effectuer des requêtes sur ce graphe via des langages dédiés.

◆ Ex : SPARQL

```
<http://example.org/alice#me> a foaf:Person .  
<http://example.org/alice#me> foaf:name "Alice"  
<http://example.org/alice#me> foaf:mbox <mailto:alice@example.org>  
<http://example.org/alice#me> foaf:knows <http://example.org/bob#me> .  
<http://example.org/bob#me> foaf:knows <http://example.org/alice#me>
```

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>  
SELECT ?name (COUNT(?friend) AS ?count)  
WHERE {  
  ?person foaf:name ?name .  
  ?person foaf:knows ?friend .  
} GROUP BY ?person ?name
```

Requête



« Données »

?name	?count
"Alice"	3
"Bob"	1
"Charlie"	1

Résultat

Source : W3C <https://www.w3.org/TR/sparql11-overview/>

RDFS : concepts proposés par RDF Schéma

13

48

◆ Rappel XML : espace de nommage (*Namespace*)

- `<espace_nommage:element attribut=valeur />`
- Utile lorsqu'un même élément peut avoir différentes définitions selon le contexte
- Ex : `projet:nom` `personne:nom`
- Sinon on ne saurait pas ce que décrit `<nom>Alberta</nom>`

◆ Classe : `rdfs:Class`

◆ Lien sous-classe (ie héritage) : `rdfs:subClassOf`

◆ Propriétés (`rdf:property`) :

- librement nommées sont associées à un « type » `rdfs:domain` ou une collection de type `rdfs:range` ou une instance d'une classe `rdf:type`.

RDFS : concepts proposés par RDF Schéma

14

48

Table 1: RDF Schema Constructs

Construct	Syntactic form	Description
<u>Class</u> (a class)	C <code>rdf:type</code> <code>rdfs:Class</code>	C (a resource) is an RDF class
<u>Property</u> (a class)	P <code>rdf:type</code> <code>rdf:Property</code>	P (a resource) is an RDF property
<u>type</u> (a property)	I <code>rdf:type</code> C	I (a resource) is an instance of C (a class)
<u>subClassOf</u> (a property)	C1 <code>rdfs:subClassOf</code> C2	C1 (a class) is a subclass of C2 (a class)
<u>subPropertyOf</u> (a property)	P1 <code>rdfs:subPropertyOf</code> P2	P1 (a property) is a sub-property of P2 (a property)
<u>domain</u> (a property)	P <code>rdfs:domain</code> C	domain of P (a property) is C (a class)
<u>range</u> (a property)	P <code>rdfs:range</code> C	range of P (a property) is C (a class)

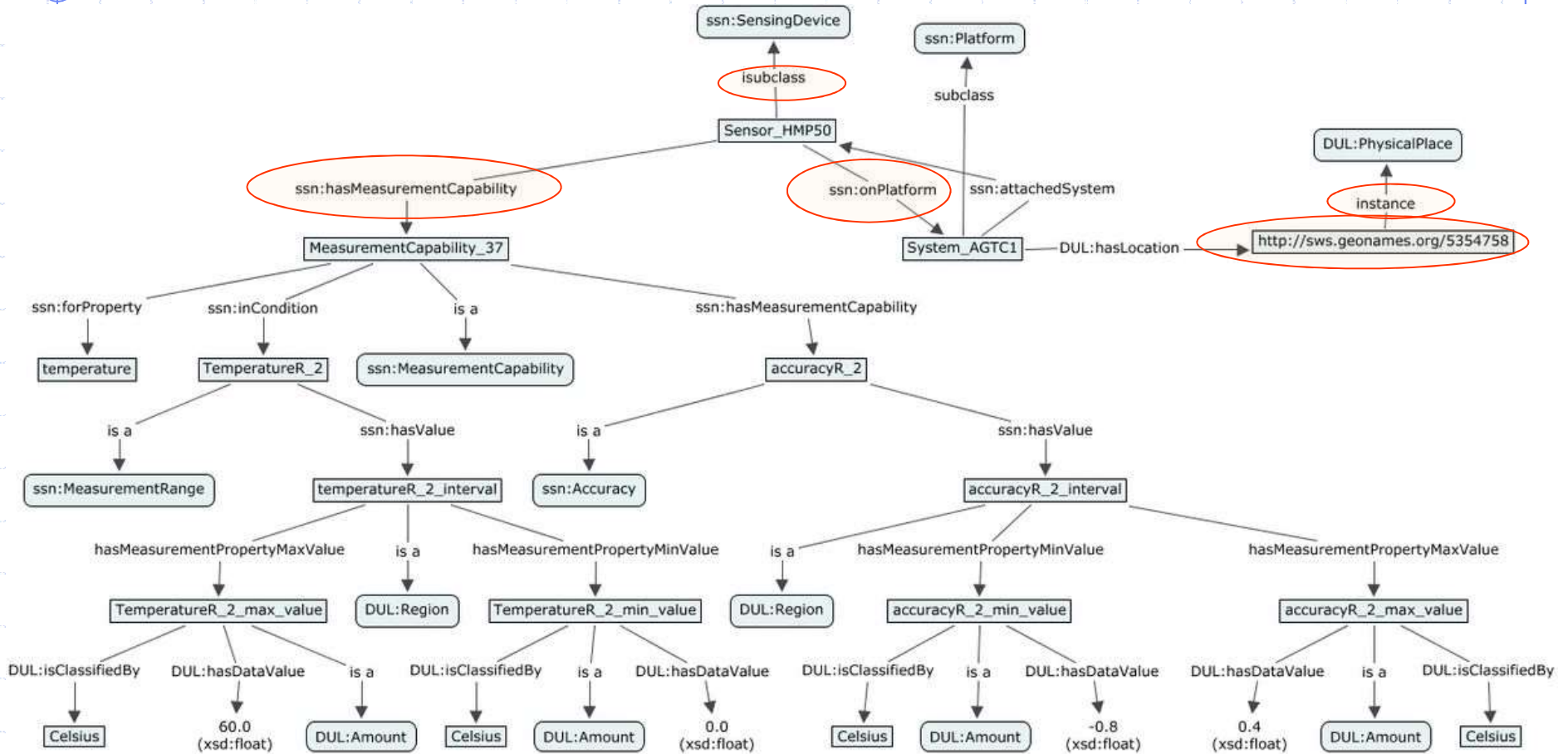
OWL : Ontology Web Language

- ◆ Basé sur RDF en ajoutant aux liens SubClassOf et InstanceOf
 - Classe équivalente (synonyme)
 - Égalité, différence, contraire, cardinalité
 - Transitivité des propriétés...
- ◆ Composé de 3 niveaux de langages
 - **OWL Lite** : permet classification hiérarchique => métamodèle
 - **OWL DL** (Description Logics) : intègre en partie la logique des prédicats (logique du premier ordre), raisonnement décidable => on peut avoir une réponse à une question sur l'ontologie.
 - **OWL Full** : plus de liberté dans l'expression des relations entre concepts mais on perd la décidabilité (la question peut ne pas avoir de réponse) => réseau sémantique
- ◆ Peut être sauvegardé (sérialisé) dans un fichier RDF !

Exemple d'Ontologie OWL : station météo

16

48



Protege 2000 : edition d'ontologies

◆ Démo : Wine ontologie

◆ Site officiel : <http://protege.stanford.edu/>

The screenshot displays the Protege 3.0 beta interface. The title bar reads "newspaper Protégé 3.0 beta (http://protege.stanford.edu/applet_demo/Newspaper/newspaper.pprj, Standard Files)". The menu bar includes "File", "Edit", "Project", "Window", and "Help". The toolbar contains various icons for file operations and editing. The main interface is divided into several panes:

- CLASS BROWSER:** Shows a tree view of classes for the project "newspaper". The selected class is "News_Service", which is a subclass of "Author". Other classes visible include "Columnist", "Editor", "Reporter", "Content", "Advertisement", "Personals_Ad", "Standard_Ad", "Article", "Layout_info", and "Library".
- CLASS EDITOR:** Shows the configuration for the selected class "News_Service". It includes fields for "NAME" (News_Service), "DOCUMENTATION" (This is a use of classes as abstract placeholders. We need some way to indicate a fixed (yet changeable) list of values. In a way, this is just a list of strings.), and "CONSTRAINTS". The "ROLE" is set to "Concrete".
- TEMPLATE SLOTS:** A table showing the slots for the class "News_Service".

Name	Cardinality	Type	Other Facets
contact_person	single	Instance of Person	
name	single	String	

At the bottom of the window, it says "Java Applet Window".

Symboles Protégé 2000

◆ Hiérarchie :

- *Asserted Hierarchy* : hiérarchie définie manuellement par l'utilisateur (lorsqu'il crée une sous-classe d'une classe). Toute classe n'héritant pas d'une classe sera « rattachée » à la racine Thing. (~ Object en java)
- *Inferred Hierarchy* : reconstruit la hiérarchie en appliquant un raisonnement sur les liens entre les classes => nécessite de démarrer un « *reasoner* ».

◆ Classes / Concept

- primitive : classe « basique » ie avec attributs (*slots*), lien d'héritage
- Définie (*defined*) : à laquelle on a ajouté des informations type logique de description (relation d'équivalence par exemple)

◆ Attributs/champs/propriétés (*facets*) (cf. apres)

◆ Individu (instance d'une classe)

Zoom sur propriétés

◆ Peut être :

- Un type de donnée
- Caractérise une relation avec un autre individu : exemple Stage proposePar Entreprise

◆ Caractérisation des propriétés

- Fonctionnelle : ne peut avoir qu'une valeur (fonction)
- Transitive : $a \text{ est_dans } b, b \text{ est_dans } c \Rightarrow a \text{ est_dans } c$
- Symétrique : $a \text{ ami_de } b \Rightarrow b \text{ ami_de } a$
- Anti-symétrique : $a \text{ fils_de } b \Rightarrow b \text{ fils_de } a$ n'est pas possible

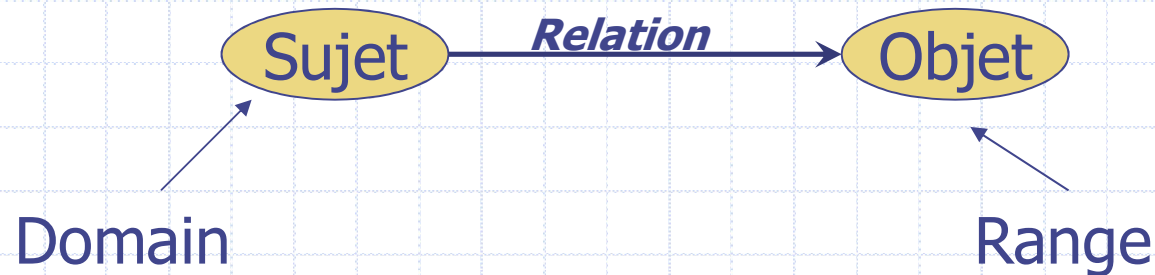
- ◆ La caractérisation permet de restreindre les chemins que l'on peut suivre dans un graphe \Rightarrow raisonnement !

Zoom sur typage propriétés : domaine & portée

20

48

- ◆ Si l'on reprends le concept de triplet RDF



- ◆ Un domaine (*domain*) caractérise/type un sujet
- ◆ Un Range caractérise/type un objet

Systemes à Base de Connaissances (Systemes Experts)

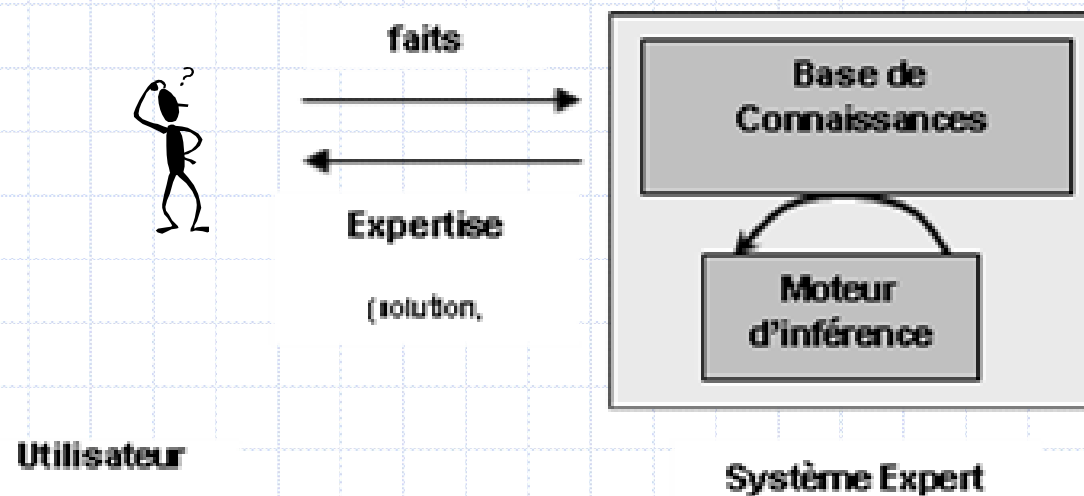
211
42
48

◆ Définition (Edward Feigenbaum)

- *SE : « an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require human expertise for their solutions. » 1982*
- *Systeme informatique qui émule la capacité humaine de prise de décision. Le terme émule souligne que le système doit plus que simuler (même entrées même sorties) et doit reproduire le comportement d'un expert humain.*
- *=> Systemes ayant la capacité de gérer la connaissance*

Principe de fonctionnement

- ◆ Le système va enchaîner des raisonnements (Règles) pour produire une conclusion...



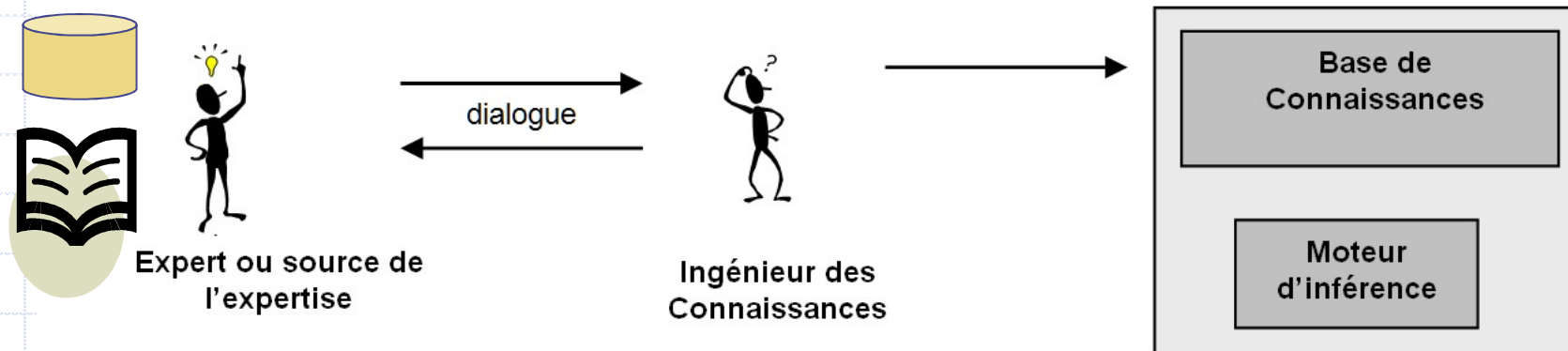
- ◆ ... peut justifier la décision (raisonnement suivi = argument)

Représentation de l'Expertise

- Connaissances formalisées sous la forme de règles SI...ALORS.

1.1 SI Le feu est rouge ALORS S'arreter	1.2 SI Le feu est orange Ma vitesse est excessive ALORS Ne pas s'arrêter.	1.3 SI Le feu est orange Ma vitesse est raisonnable ALORS S'arrêter
--	--	--

- Démarche d'ingénierie des connaissances



Principe du moteur d'inférence

- ❑ On sépare le raisonnement des faits :
 - ❑ **Faits** : informations connues
 - ❑ **Règles** : que faire lorsque des faits sont connus
 - => Modifier/Supprimer le/s ancien/s fait/s
 - Ajouter de nouveaux faits
 - ❑ **Moteur d'inférence** : c'est ce qui « déroule » le raisonnement : doit mettre en relation les faits et les règles :
 - ❑ S'appuie sur mécanisme de « base » du raisonnement :
 - ❑ Modus Ponens : Si je sais A, et que $A \Rightarrow B$, alors je sais B
 - ❑ 2 approches de « déroulement »
 - ❑ Chaînage avant / Chaînage arrière

Chaînage arrière :

- ❑ **Principe** : on émet des hypothèses (faits/buts) et les conditions permettant de prouver ces hypothèses
 - ❑ $A \leftarrow B C$: A se reformule en :
A est vrai **SI** B est vrai **ET** C est vrai
 - ❑ Mais on peut avoir aussi $A \leftarrow B D$, $A \leftarrow D E$
- ❑ On parle parfois de raisonnement tiré par les Buts (on a une hypothèse, on veut la vérifier). Ex. Processus de Diagnostique
- ❑ On interroge alors le système pour savoir si B est vrai : **B ?**
 - ❑ => le système va chercher à prouver B en fonction des règles qui existent ... Une hypothèse B peut être prouvée si :
 - ❑ 10 : B est vrai (présent dans la base fait)
 - ❑ 20 : Une règle existe permettant de prouver B. Cette règle requiert la preuve d'un ou plusieurs faits
 - ❑ 30 : vérifier si ces faits sont présents dans la base de fait, SINON GOTO 10 pour chacun des faits.

Chaînage arrière :

A partir de la base de faits : K, E, H

R1 : A ← B, C, D

R2 : A ← G, I

R3 : B ← K

R4 : C ← E, G

R5 : D ← G

R6 : D ← B, H

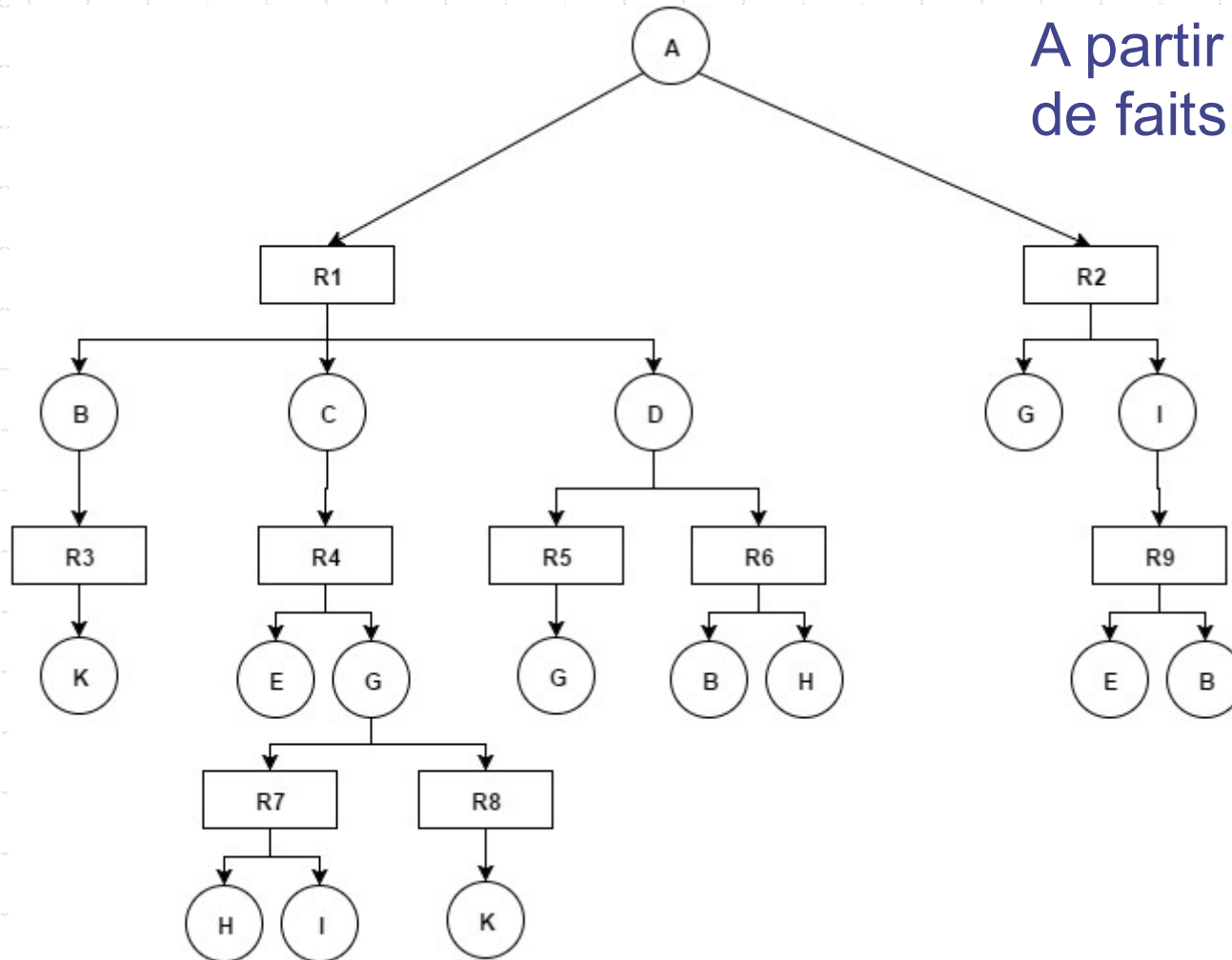
R7 : G ← H, I

R8 : G ← K

R9 : I ← E, B

⇒ Prouver A (A?)

Chânage arrière :



A partir de la base
de faits : K, E, H

A?

Chaînage avant:

□ **Principe** : on émet des prémisses qui si elles sont vérifiées permettent de déduire des conclusions

□ $B \ C \Rightarrow A$: se reformule en :

SI B est vrai ET C est vrai ALORS A est vrai

□ L'idée est

1. De définir les faits initiaux

2. Lancer le raisonnement (*modus ponens*)

1. Lister les règles dont la partie **Prémisse** est vérifiées (règles candidates)

2. Classer les règles selon critère retenu (largeur, profondeur, ...)

3. Sélectionner une règle

3. « Exécuter une règle », c'est à dire rendre effectif la partie Conclusion de la règle (maj de la base de faits).

4. Goto 2 tant qu'il reste une règle candidate.

□ Que faire quand 2 règles sont candidates ?

Chaînage avant : exemple

□ A partir de la base de faits : K, E, H

R1 : B, C, D \Rightarrow A

R2 : G, I \Rightarrow A

R3 : K \Rightarrow B

R4 : E, G \Rightarrow C

R5 : G \Rightarrow D

R6 : B, H \Rightarrow D

R7 : H, I \Rightarrow G

R8 : K \Rightarrow G

R9 : E, B \Rightarrow I

□ Quel est l'état de la base de fait après traitement des règles ?

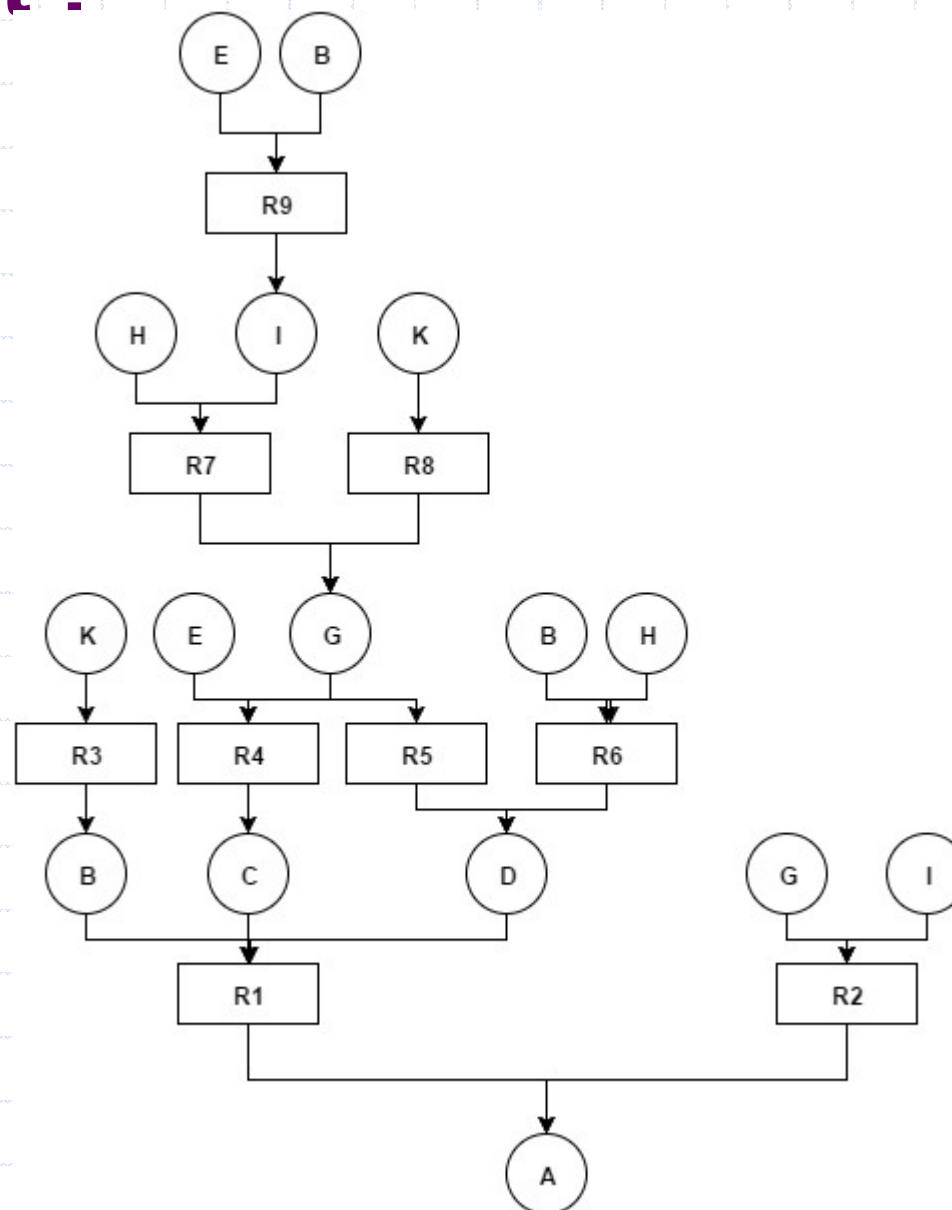
□ En cas de conflit (nb de règles candidates > 2): règle ayant les conditions les plus récentes (ordre d'insertion des faits), si conflit persiste, ordre de création des règles.

Chaînage avant :

vision « arbre » brute

30

48



Chaînage avant :

A partir de la base de faits : K, E, H

En cas de conflit : règle ayant les conditions les plus récentes (ordre d'insertion des faits), si conflit persiste, ordre de création des règles.

Base de faits initiale	Règles Applicables	Règle Appliquée	Base de faits résultante
K, E, H	R3, R8	R3	K, E, H, B
K, E, H, B	R8, R6, R9	R6	K, E, H, B, D
K, E, H, B, D	R8, R9	R9	K, E, H, B, D, I
K, E, H, B, D, I	R8, R7	R7	K, E, H, B, D, I, G
K, E, H, B, D, I, G	R8, R4, R5, R2	R2	K, E, H, B, D, I, G, A
K, E, H, B, D, I, G, A	R8, R4, R5	R4	K, E, H, B, D, I, G, A, C
K, E, H, B, D, I, G, A, C	R8, R4, R1	R1	K, E, H, B, D, I, G, A, C
K, E, H, B, D, I, G, A, C	R8, R4	R4	K, E, H, B, D, I, G, A, C
K, E, H, B, D, I, G, A, C	R8	R8	K, E, H, B, D, I, G, A, C

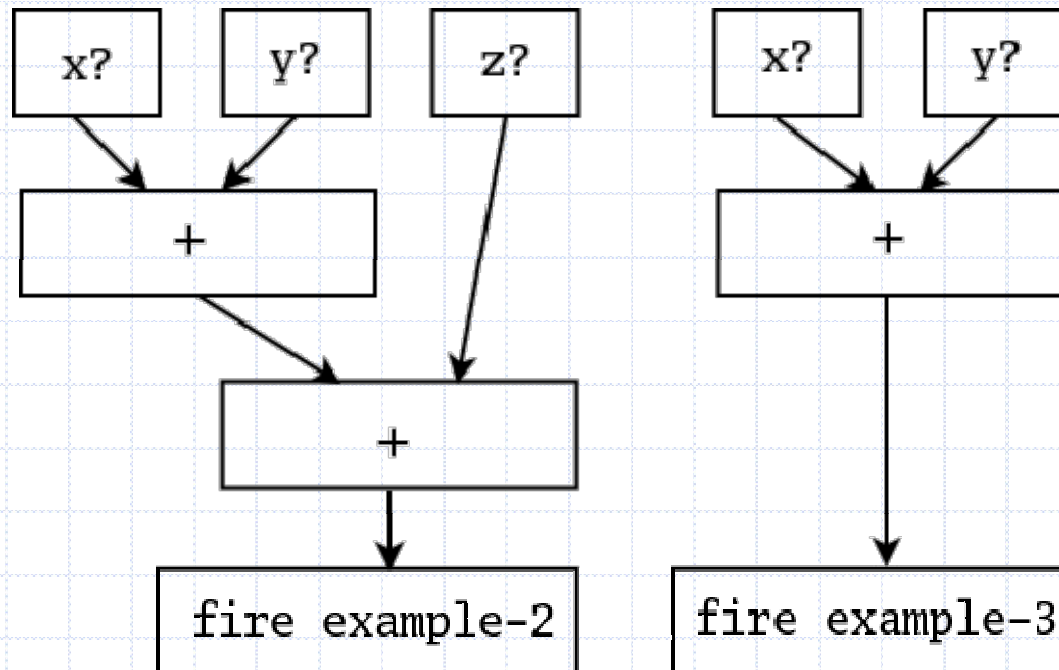
Algorithme Rete / CLIPS

- ❑ CLIPS¹ moteur d'inférence (chaînage avant) produit par la NASA.
- ❑ Principe de l'algorithme (Forgy 1982) :
 - ❑ Cout important de la vérification de l'activation des règles à chaque cycle...
 - ❑ ... alors qu'en fait le taux de modification des faits (nombre faits modifiés ./ Nb de faits) est relativement faible
 - ❑ => organiser les règles en nœuds, chaque nœud décrivant une condition à vérifier, et « partager » les nœuds avec les règles concernées.
 - ❑ => on trouve plus rapidement les règles à « activer »
 - ❑ **Remarque** : on se base uniquement sur la partie Prémisse de la règle.

¹. <http://www.siliconvalleyone.com/founder/clips/index.htm> - <http://clipsrules.sourceforge.net/>

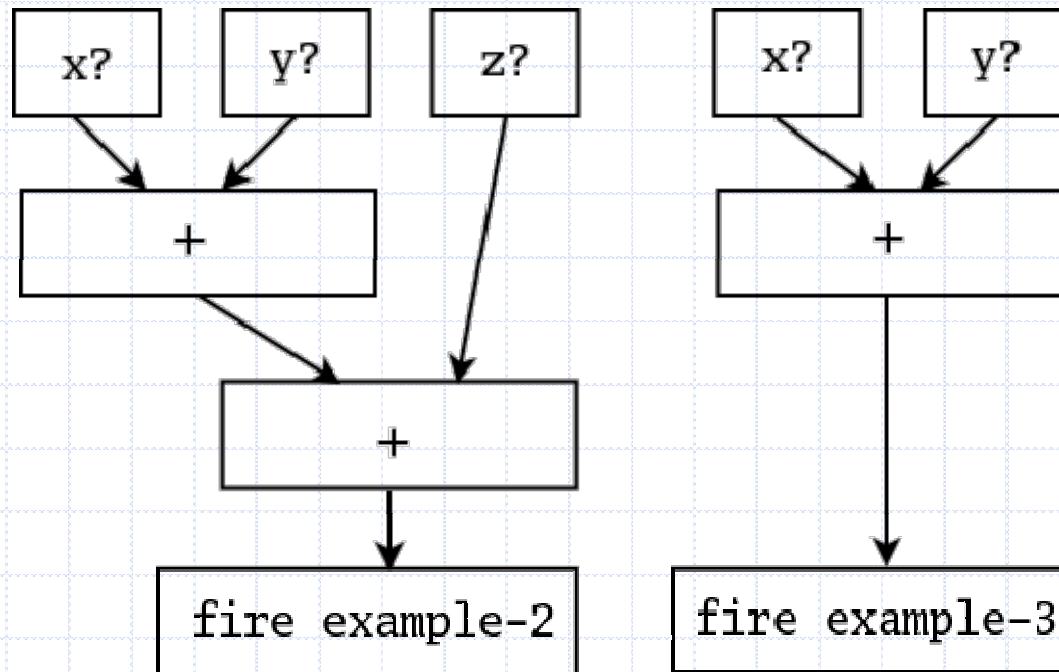
Algorithme Rete exemple

(defrule example-2 (x) (y) (z) =>) (defrule example-3 (x) (y)) =>)

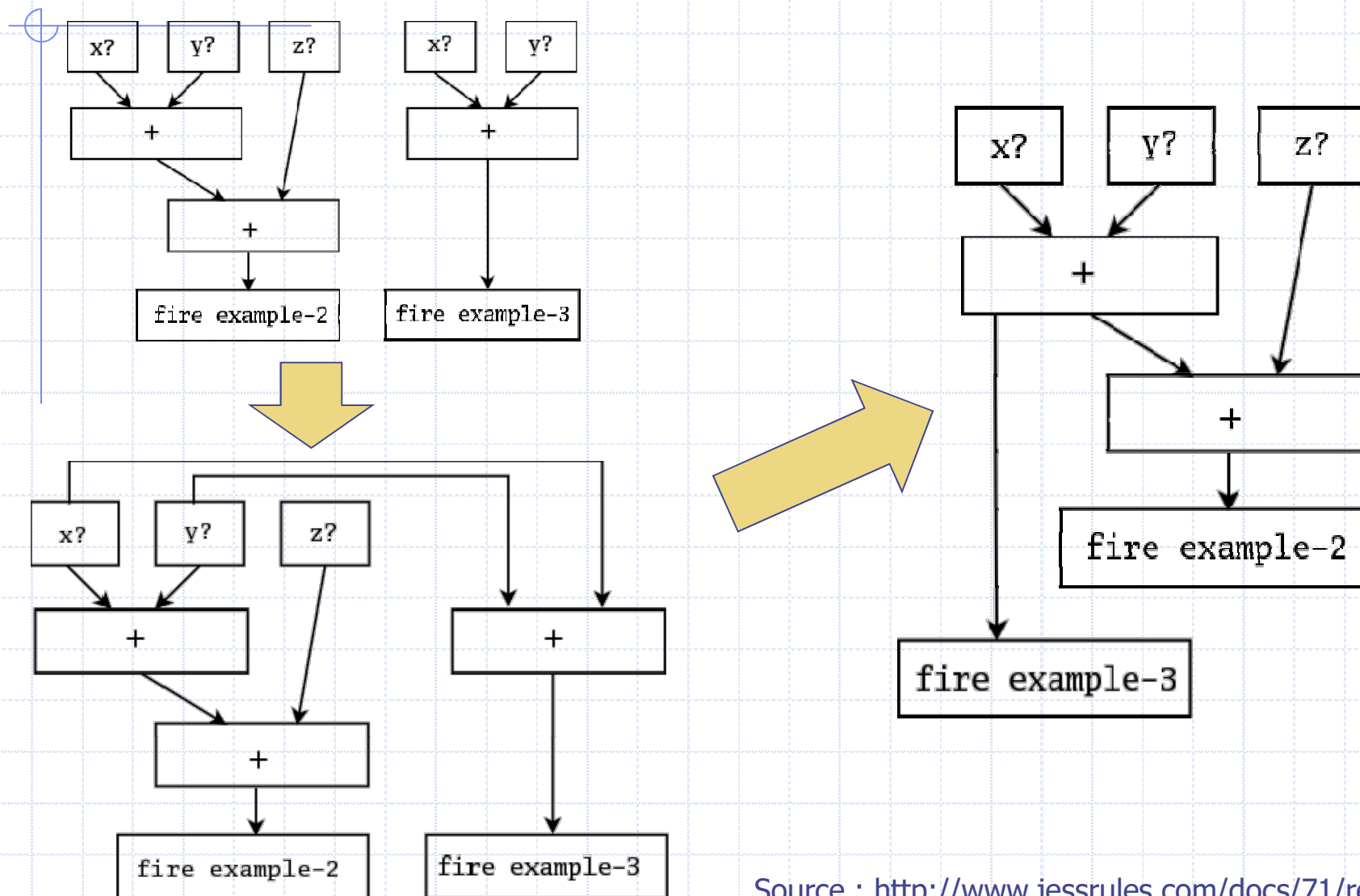


Algorithme Rete exemple

(defrule example-2 (x) (y) (z) =>) (defrule example-3 (x) (y)) =>)



Algorithme Rete exemple



Modélisation des connaissances

Illustration sur marché de l'emploi : *Model this !*

- **Entreprise :**

ADECCO BAGNOLS/CEZE recherche pour l'un de ses clients, société spécialisée en recherches nucléaires, un Ingénieur informatique pour un contrat de travail temporaire d'une durée de 18 mois.

- **Poste et missions :**

Rattaché au Chef de Service, vous aurez pour missions :

- participer au choix des logiciels et des matériels,
- identifier et résoudre les problèmes des utilisateurs,

Vous serez également responsable de la sécurité sur le réseau et vous assurerez la confidentialité des données qui y transitent.

Vous devrez également :

- identifier et anticiper les besoins en informatique de l'entreprise.

- **Profil :**

De formation Bac + 5 en Informatique, vous avez acquis une première expérience réussie dans la fonction, idéalement en industrie nucléaire.

Vous devez également être titulaire du PR1 CC en cours de validité.

Vos connaissances en "Infostructure" et/ou en "Système réseaux" sont des atouts pour collaborer efficacement avec les différents acteurs.

Vous êtes reconnu pour votre dynamisme et votre rigueur. Votre savoir faire et votre adaptabilité feront la différence.

Offre d'emploi INGENIEUR INFOI

Référence apec: 161209458W-20018

Date de publication: 07/12/2015

Société: ADECCO TRAVAIL
TEMPORAIRE

[Voir toutes les](#)

Nombre de postes: 1 en Interim de 18 m

Statut: Cadre du secteur pi

Lieu: BAGNOLS SUR CEZ

Salaire: 31 K€ brut/an

Expérience: Tous niveaux d'expérience

Dossier suivi par: FLORIANE MERLE
CHARGÉE D'AFFAIRES

[En savoir plus](#)

[Postuler à cette offre](#)

[Sauvegarder cette offre](#)

Programmation avec CLIPS (C Language Production System)

37

48

- ❑ Logiciel gratuit produit par la NASA
 - ❑ <http://clipsrules.sourceforge.net>
- ❑ Qualifié de générateur de Systèmes Experts (ie Moteur + Langage)
- ❑ Syntaxe « à la LISP », notation pré fixée $2+3 \Rightarrow + 2 3$
- ❑ Porté en Java \Rightarrow Jess



Concept CLIPS : Types de données

38

48

Type de donnée	Définition	Exemple
Integer	[+ -] nombre	1 +3 -1 65
Float	Integer [.nombre] [e [+ -] nombre]	1.5 1.0 9e-1 3.5e1
Symbol	<lettre chiffre ! # ^ *><caractères> Attention : CLIPS distingue minuscules et majuscules	Bonjour, bonjour, hello-world, 345B, 127-0-0-1
String	<"><caractère>*<">	"hello world" ""\ "hello\ " "10 francs "
External address	Adresse mémoire d'une structure de donnée externe i.e. retournée par une fonction utilisateur ¹ <Pointer-XXXXXX> où XXXXXX est l'adresse mémoire externe	<Pointer-00CF61AB>
Instance name	<[> Symbol <]> : nom d'une instance de classe dans COOL	[pump-1]
Instance address	Adresse mémoire d'un objet COOL. <Instance-XXX> où XXX est le nom de l'instance.	[Instance-pump-1]

Concept CLIPS : Quelques commandes

39

48

- ❑ Afficher un texte sur la console (t)
 - ❑ (printout t " texte ")
- ❑ Lancer le moteur d'inférence :
 - ❑ (run)
 - ❑ (run n) : lance le moteur et s'arrête après n déclenchement de règle (exécution)
- ❑ Ajouter un fait
 - ❑ (assert <Fait>)
- ❑ Supprimer un fait
 - ❑ (retract <idFait>)
- ❑ Modifier un fait :
 - ❑ (modify <idFait> (<champ_a_modifier> <nouvelle_valeur>)

Concept CLIPS : *Faits*

- ❑ Faits dits non structurés ne nécessitent pas de déclaration préalable

- ❑ (nom-fait [valeurs]*)
- ❑ => (personne albert)
- ❑ => (mère albert sophie)

- ❑ Faits structurés (ordonnés)

- ❑ Comparable à Struct en C
- ❑ (etudiant (prenom albert) (age 22) (cours (maths info)))
- ❑ Requier une déclaration avec deftemplate

```
(deftemplate nom-du-fait-structuré
  (slot champ1)
  (multislot champ2)
  (slot champ3)
)
```


Concept CLIPS : Règles

□ Syntaxe générale

```
(defrule nom-de-la-regle ;  
  partie condition  
  ; faits, Patterns cad patron  
  d'appariement  
  =>  
  ; partie conclusion  
  ; insertion de faits, calculs, ...  
)
```

□ Ex :

```
(defrule fermeture-de-la-porte-1  
  ?f <- (porte ouverte)  
  =>  
  (printout t "Fermer la porte SVP" crlf) ;  
  (retract ?f)  
  (assert (porte fermee) )  
)
```

Retour sur concours ... modélisation du processus d'orientation

42

48

◆ Orienter, terme valise ...

- Quelles études faire ?
- Si études ingénieur, quelle formation suivre après le Bac ?
- Si déjà étudiant, quelles voies possibles ?
- ... problème classique en informatique ... se lancer dans une spec sans avoir cerné le périmètre !
- Voire pire, considérer une interprétation sans considérer d'autres ... because it's obvious ...

Modélisation du processus d'orientation : hypothèse 3 => quelle voie ?

43

48

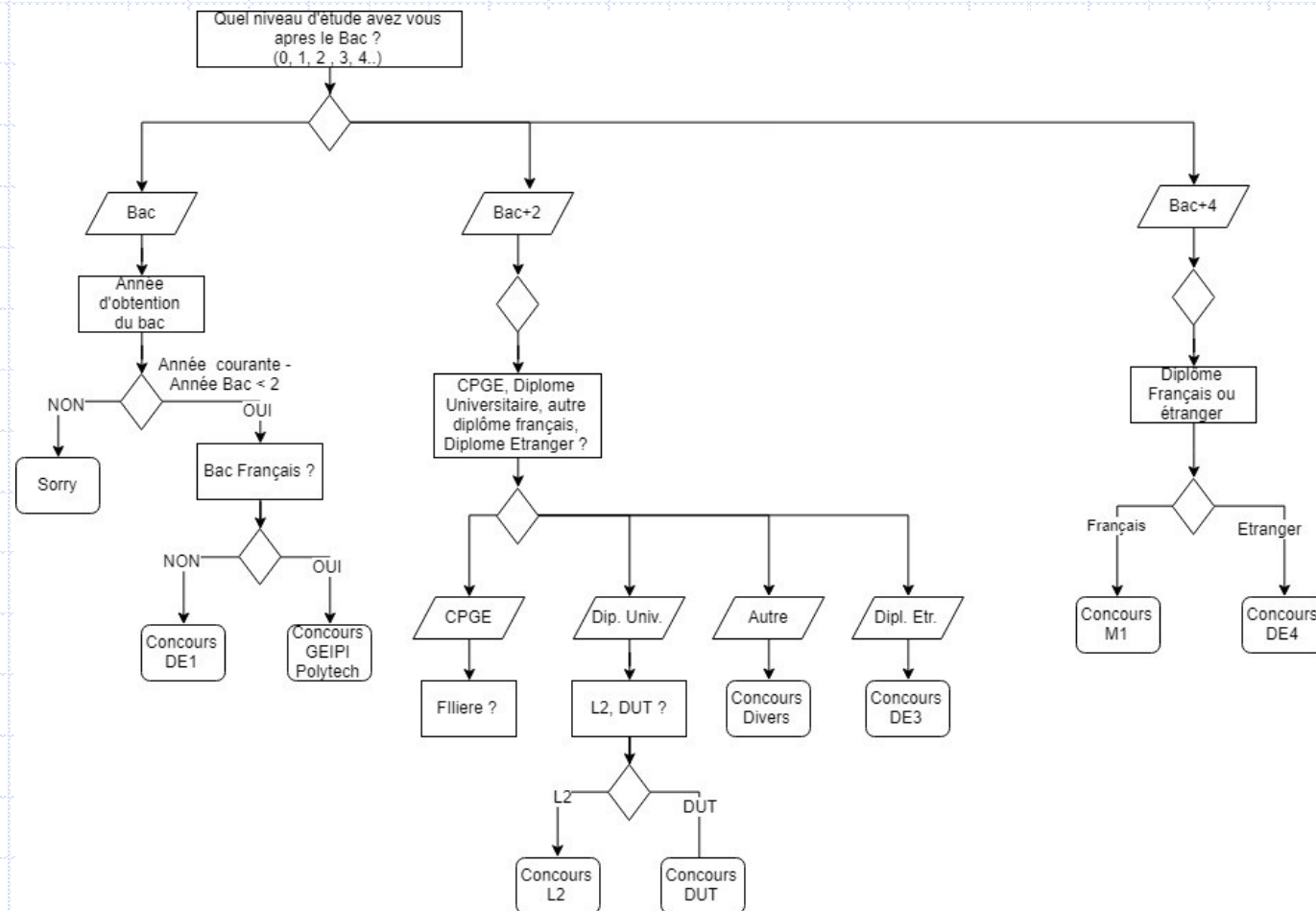
- ◆ De manière manuelle on demandera que fait l'étudiant actuellement, et a force de question, on cerne le profil et identifie la voie concernée ...
- ◆ => série de questions réponses ...
- ◆ => arbre de décision !
- ◆ Simple ? *Let's see ...*

Modélisation du processus d'orientation : hypothèse 3 => Arbre de décision ...

44

48

◆ Simplifié 😊



Modélisation du processus d'orientation : hypothèse 3 => Decision tree to implementation ?

45

48

◆ D'un côté la formalisation de l'arbre en micro règle Si
Alors se traduirait relativement facilement :

◆ *Poser question niveau d'étude après le Bac*
Si nb Années = 0 => Question sur année obtent° du bac
Si nb Années =2 ou 3 => Préciser type étude
Si nb Années > 3 => Demander si diplôme français ou non

◆ *Implémentation dans un langage :*

- *Définir format des informations / connaissances => CLIPS = Faits*
- *Traduire règle utilisant ces faits, en exprimant les contraintes des valeurs ... en tenant compte des contraintes spécifiques au langage*

Modélisation du processus d'orientation : hypothèse 3 => Exemple implémentation en CLIPS

46

48

```
(defrule annee_etudes
(initial-fact)
=>
(printout t "Quel niveau d'étude avez vous ? " crlf)
(printout t "valeurs attendus = nb d'années apres le bac ou son équivalent
(0 si actuellement en lycée) " crlf)
(bind ?nb_annees (read) )
(assert (niveau_etude ?nb_annees) )
)

(defrule question_niveau_bac
(niveau_etude ?nb_annees)
(test (< ?nb_annees 2) )
=>
(printout t "Vous êtes donc soit élève bachelier soit déjà titulaire du
Bac." crlf)
(printout t "En quelle année (20xx) avez vous obtenu votre bac ou quand
l'obtiendrez vous ?" crlf)
(bind ?annee_bac (read) )
(assert (annee_bac ?annee_bac) )
)
```

Modélisation du processus d'orientation : hypothèse 3 => Exemple implémentation en CLIPS

47

48

```
(defrule reponse_bac_trop_vieux
(current_year ?annee_courante)
(annee_bac ?annee_bac)
(test (> (- ?annee_courante ?annee_bac ) 1 ) )
=>
(printout t "Désolé..." crlf)
)
```

```
(defrule reponse_bac_OK
(current_year ?annee_courante)
(annee_bac ?annee_bac)
(test (< (- ?annee_courante ?annee_bac ) 2 ) )
=>
(printout t "S'agit-il d'un bac Français ? Oui / Non " crlf)
(bind ?bac_francais_ON (upcase (read) ))
(assert (bac_francais ?bac_francais_ON) ) ) )
```

Modélisation du processus d'orientation : hypothèse 3 => Exemple implémentation en CLIPS

48

48

```
(defrule reponse_bac_francais_OUI
(bac_francais OUI)
=>
(printout t "Vous relevez du concours GEIPI POlytech" crlf)
(assert (concours GEIPI_Polytech) )
)

(defrule reponse_bac_francais_NON
(bac_francais NON)
=>
(printout t "Vous relevez du concours DE1" crlf)
(assert (concours DE1) )
)
```