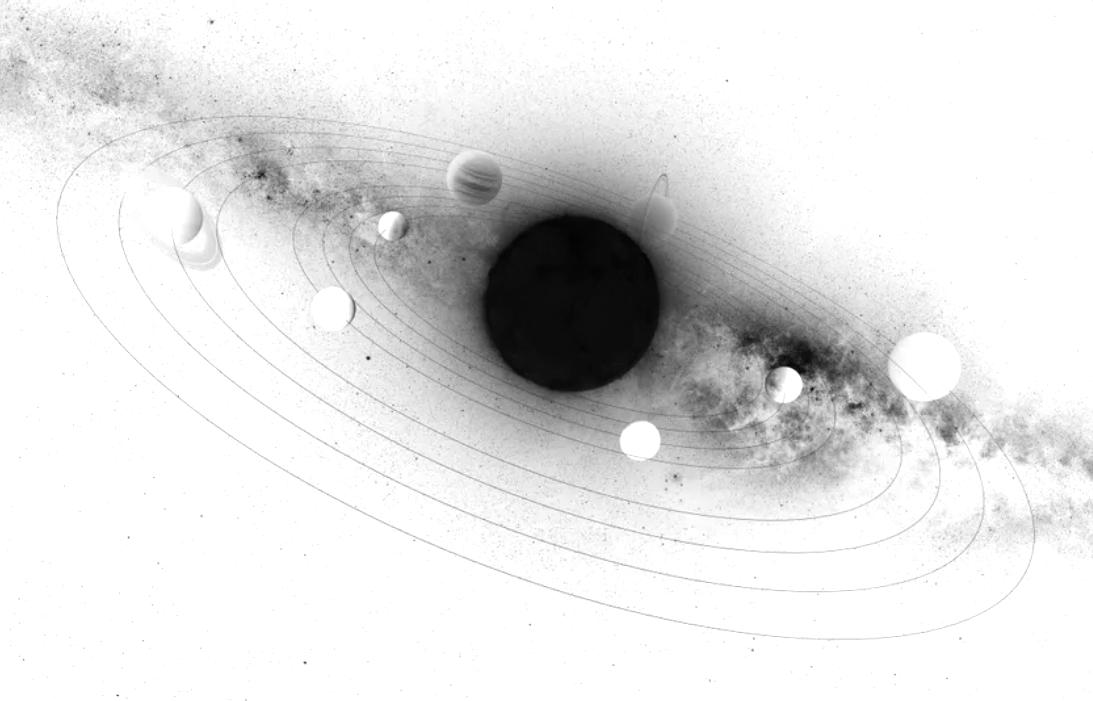
Le système solaire

Ontologie Protégé - IA301 - MS IA



**Sami Allani**

**Marie Campo**

**Samuel Flandrin**

**Abdelkarim Moussaid**

**Léa Papillon**

**Louis-Gabriel Pouillot**

# Sommaire

[**Sommaire**](#_fjpkdld6ohhc) **2**

[**Introduction**](#_r0rnfz2gdnts) **3**

[**I- Le modèle de données**](#_6ch4jormrtmd) **4**

[Les classes](#_lni52kci78qn) 4

[Les propriétés des objets](#_1asztzrc1g04) 4

[Les propriétés des données](#_wowyxm4zfhpi) 4

[Les individus](#_xbshred2lnro) 5

[Les relations](#_xn2zp9q357mp) 5

[**II- Les inférences**](#_st27qjhjek7j) **6**

[Le moteur d’inférence](#_c78t6jrjaid5) 6

[Exemples d’inférences](#_ksaw2dvli9e9) 6

[Inférences pour Jupiter](#_9izihi8q78sk) 6

[Vue Explanation d’une inférence](#_9iamcyc7a11g) 7

[**III- Requêtes**](#_k9iq13rpcwax) **9**

[Retrouver la Terre et Venus](#_50urjqsbex4j) 9

[Utiliser des valeurs numériques](#_z0c7exnglti) 9

[Une requête plus complexe](#_7e1ncyy3t8wi) 10

[**IV- Visualisations**](#_lp7br4jga29o) **12**

[Visualisation des classes (vue d’ensemble)](#_8rm87gw5blsy) 12

[Visualisation des classes (avec quelques object properties)](#_3acijbg7dzca) 12

[Visualisation des classes et object properties (zoom)](#_cptljiincjza) 13

[Visualisation d’une sélection de arc types et node types](#_9eym6kk0z8wk) 13

[**Conclusion**](#_dny3tec0jyz8) **15**

[**Livrables**](#_c2a0o6nv9719) **15**

[**Versions de logiciels et bibliothèques**](#_85e9xznm432z) **15**

# 

# Introduction

L’ontologie autour du Système Solaire présentée dans ce document a été créée dans le cadre de Travaux Pratiques du module IA301 “Logique et IA Symbolique”, du MS IA.

An ontology is “a formal specification of a shared conceptualization (Gruber 1993), a formalism to define concepts, individuals, relationships and constraints (functions, attributes) within a domain.”

Une ontologie est une représentation formelle d’un champ de connaissances donné. Elle est structurée en “concepts” hiérarchisés, éventuellement liés par des relations, et dont il est possible d’instancier des “individus”.

* Concept : définition formelle d’une agrégation de choses
* Individus : instance d’un concept
* Relations : lien entre concepts / individus

La représentation d’un champ de connaissance sous forme d’une ontologie présente plusieurs intérêts.

D’abord, elle permet aux experts du domaine de s’accorder sur des définitions communes (caractéristiques intrinsèques des concepts) et sur un vocabulaire uniforme (standardisé), qui favorisent les collaborations. Une fois définie, cette ontologie constitue une représentation pérenne du domaine, qui demeure lorsque l’expert part.

Ces représentations formelles peuvent ensuite servir de support à des raisonnements logiques (déduction, abduction…).

Dans le cadre de ce projet, nous avons choisi de représenter formellement les entités du Système Solaire (planètes, étoiles, satellites, astéroïdes, transneptuniens, etc.). Nous présentons ici le modèle que nous avons créé sur Protégé (concepts, relations, instances), et les fonctionnalités d’inférence qui nous ont permis de compléter automatiquement l’ontologie. Nous verrons également quelques exemples de requêtes qui nous ont permis d’exploiter le modèle.

# I- Le modèle de données

## Les classes

* owl:Thing
  + Astres
  + Astéroïdes
  + Ceinture\_de\_Kuiper
    - Objet-Transneptuniens
  + Etoile
  + Nuage\_de\_Oort
    - Comètes
  + Planètes
  + Satellites
  + ObjetsArtificiels
  + Sonde

## Les propriétés des objets

* owl:topObjectProperty
  + aCommeSatellite
  + aEtéVisitéPar
  + aMoinsDeSatellitesQue
  + aPlusDeSatellitesQue
  + aVisité
  + estDeLaMêmeCouleurQue
  + estEnRotationSynchroneAvec
  + estEnRésonanceOrbitaleAvec
  + tourneAutourDe

## Les propriétés des données

* owl:topDataProperty
  + aUneActivitéInterne
  + couleur
  + densité
  + distance
  + développeUneQueue
  + engendreDesRéactionsNucléaires
  + forme
    - irrégulière
    - sphérique
  + masse
  + nature
    - gazeuse
    - tellurique
  + possèdeDesAnneaux
  + possèdeUneAtmosphère
  + périodeDeRotation
  + périodeDeRévolution
  + rayon

## Les individus

| Albion | Giotto | Mercure | Soleil |
| --- | --- | --- | --- |
| Callisto | Hale-Bopp | Mimas | Stardust |
| Cassini\_Huygens | Hayabusa | Miranda | Tchoury |
| Change'e | Haykutake | Near\_Schoemaker | Tempel1 |
| Charon | Hygie | Neowise | Terre |
| ComèteDeHalley | Hyperion | Neptune | Titan |
| Cérès | Io | New\_Horizon | Titania |
| Dawn | Itokawa | Obéron | Triton |
| DeepImpact | Juno | Pallas | Uranus |
| Deimos | Jupiter | Phobos | Vega |
| Encelade | Luna | Pioneer | Venera |
| Epiméthée | Lune | Pluton | Venus |
| Eris | Magellan | Protée | Vesta |
| Eros | Makémaké | Rhéa | Viking |
| Europe | Mariner | Rosetta | Voyager1 |
| Galileo | Mars | Saturne | Voyager2 |
| Ganymède | Maven | Sedna | Wild |

## Les relations

Les propriétés listées précédemment ont les caractéristiques suivantes :

**aVisité** : est la propriété inverse de aEtéVisitéPar

**aPourSatellite** : inverse de tourne AutourDe

**tourneAutourDe** : pas transitif

**estEnRésonanceOrbitaleAvec** : symétrique mais pas transitif ni euclidien - les orbites sont parcourues dans des temps qui sont en rapport entre elles ; par exemple Io en résonance avec Ganymède et Europe, mais Europe et Ganymède ne sont pas en résonance ensemble.

**estEnRotationSynchroneAvec** (période de révolution autour de la planète = période de rotation du satellite sur lui-même) : pas symétrique, le satellite présente la même face à la planète mais la planète ne présente pas la même face au satellite (exemple Terre/lune)

**estDeLaMêmeCouleurQue** : symétrique, transitif et réflexif

**aPlusDeSatellitesQue** : inverse de aMoinsDeSatellitesQue, transitif mais pas symétrique.

# II- Les inférences

## Le moteur d’inférence

Un moteur d'inférence permet de conduire des raisonnements logiques et de dériver des conclusions à partir d’une base de connaissances ou de faits.

Protégé embarque plusieurs moteurs d’inférence par défaut. Nous avons utilisé le moteur Hermit 1.4.3.456 dans le cadre de ce TP.

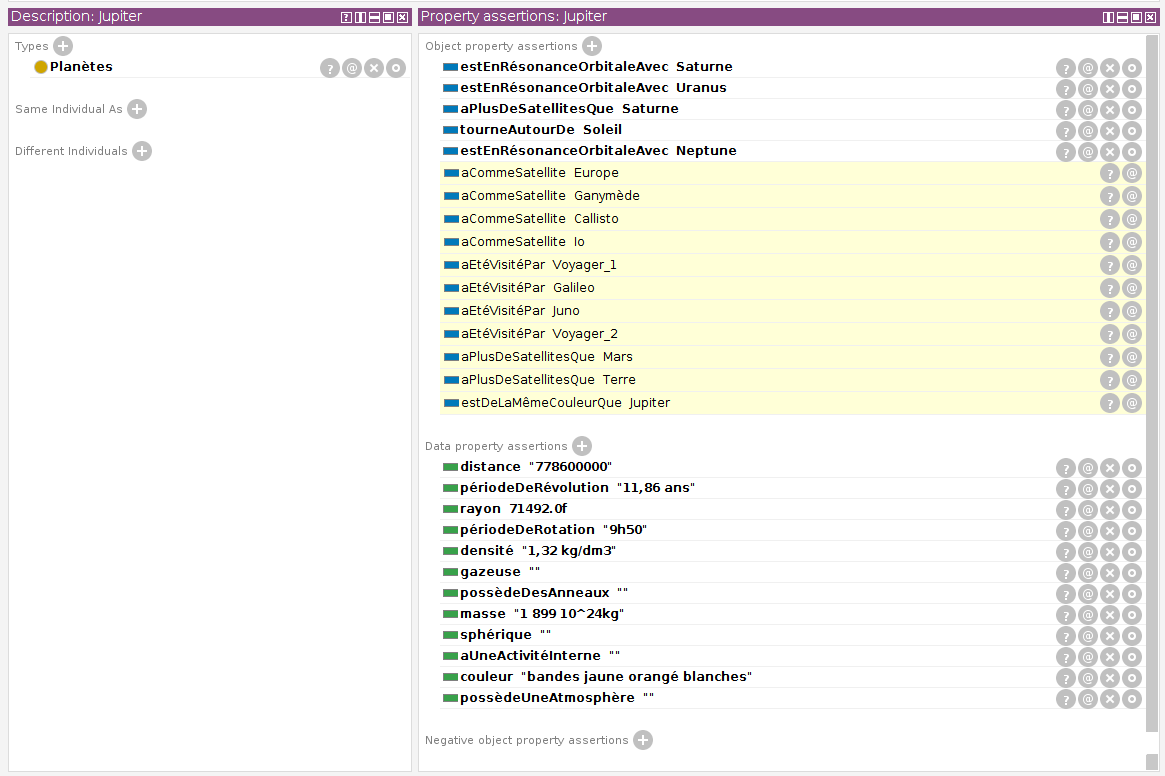
En s’appuyant sur les connaissances ajoutées manuellement et sur les propriétés des relations présentées précédemment, l’algorithme de simulation des raisonnements déductifs a pu réaliser plusieurs inférences qui sont présentées ci-dessous :

## Exemples d’inférences

#### Inférences pour Jupiter

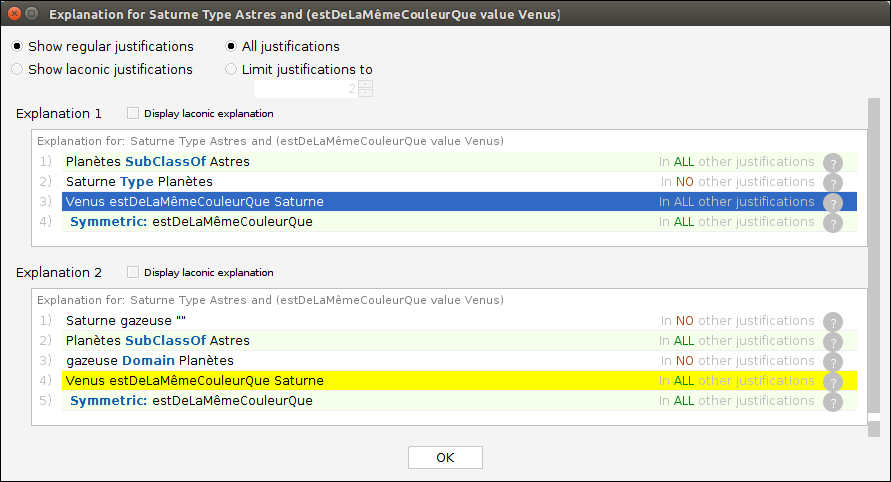
Nous illustrons les inférences réalisées pour Jupiter dans la vue ci-dessous.

Nous voyons que Jupiter a plus de satellites que Mars car Jupiter en a plus que Saturne, et Saturne en a plus que Mars. Les satellites de Jupiter sont également déduits, ainsi que les sondes qui ont survolé cette géante gazeuse.

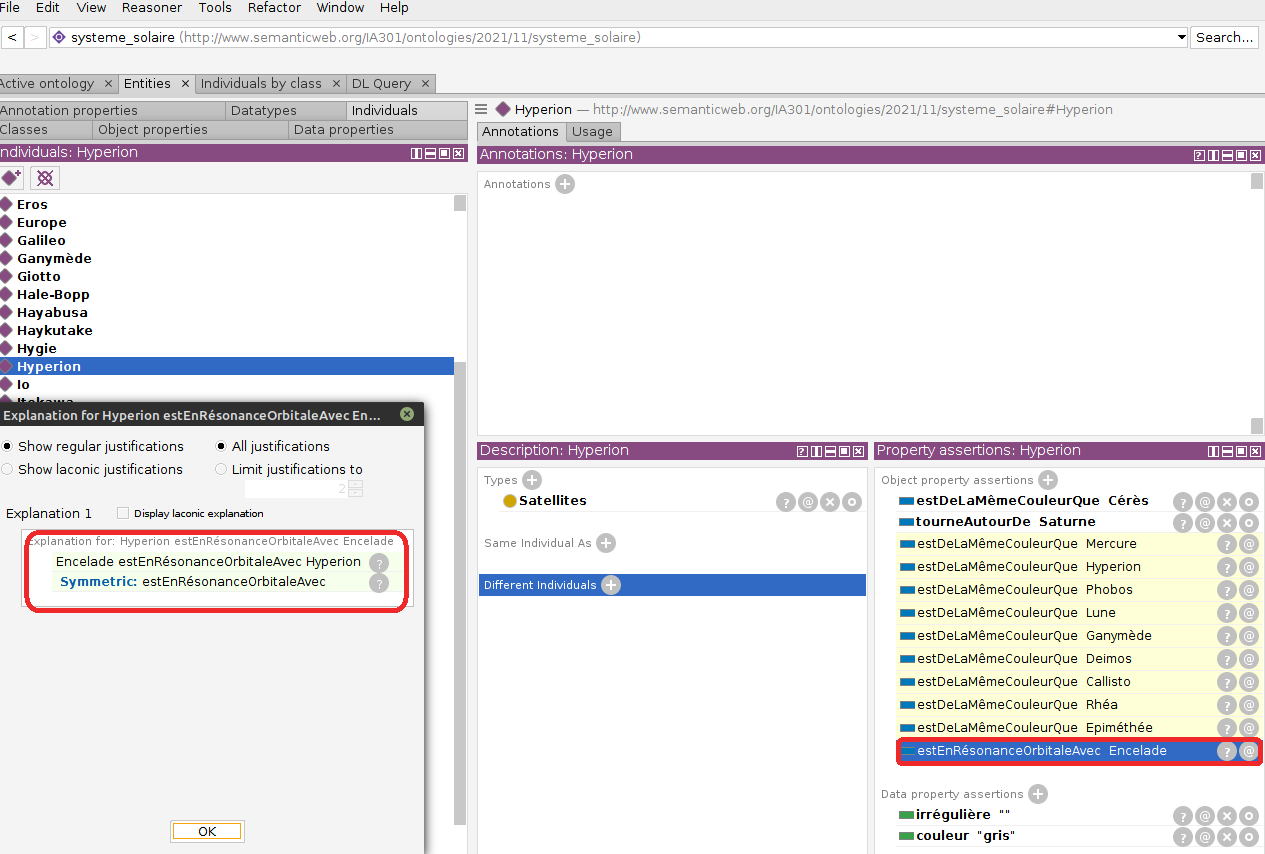


#### Vue Explanation d’une inférence

Nous mettons en évidence les raisons du résultat de ces inférences sur la fenêtre *Explanation* pour une requête Descriptive Language (DL Query) telle que *“Astres and estDeLaMêmeCouleurQue value Venus*”. Saturne est de la même couleur que Vénus car on a dit que Titan était de la même couleur que Vénus, et que Saturne.



Par symétrie également on obtient que Hypérion est en résonance orbitale avec Encelade puisqu’on a indiqué qu’Enselade était en résonance avec Hypérion. Ce qu’on peut visualiser ci-dessous.



# III- Requêtes

Avec le moteur d’inférence nous avons utilisé l'outil de requêtes DL Query pour réaliser plusieurs requêtes présentées dans cette partie.

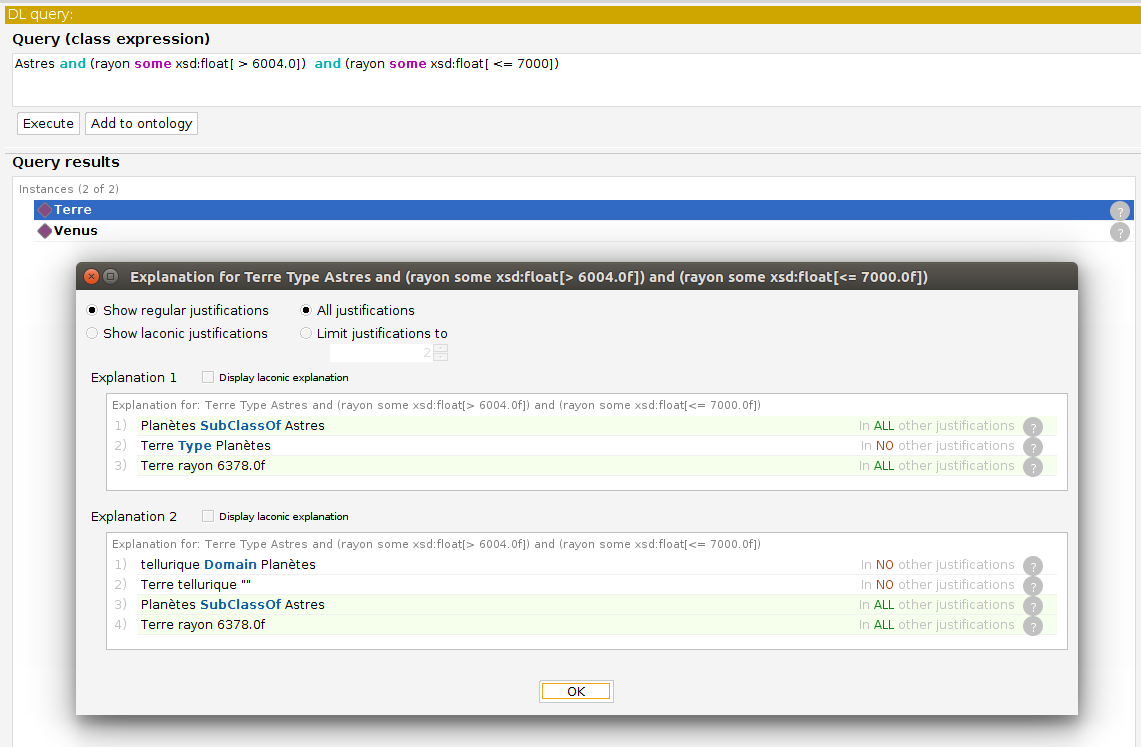
## Retrouver la Terre et Venus

Requête : Astres and sphérique value "" and tourneAutourDe value Terre

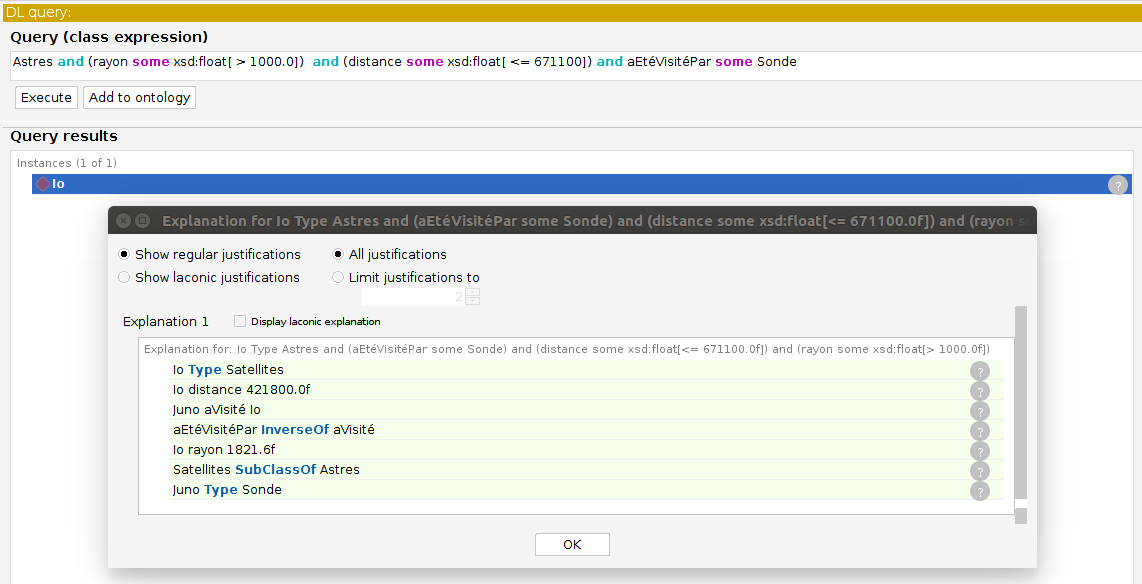
#### 

## Utiliser des valeurs numériques

Requête avec des valeurs numériques sur le rayon : Astres and (rayon some xsd:float[ > 6004.0]) and (rayon some xsd:float[ <= 7000])



## Une requête plus complexe

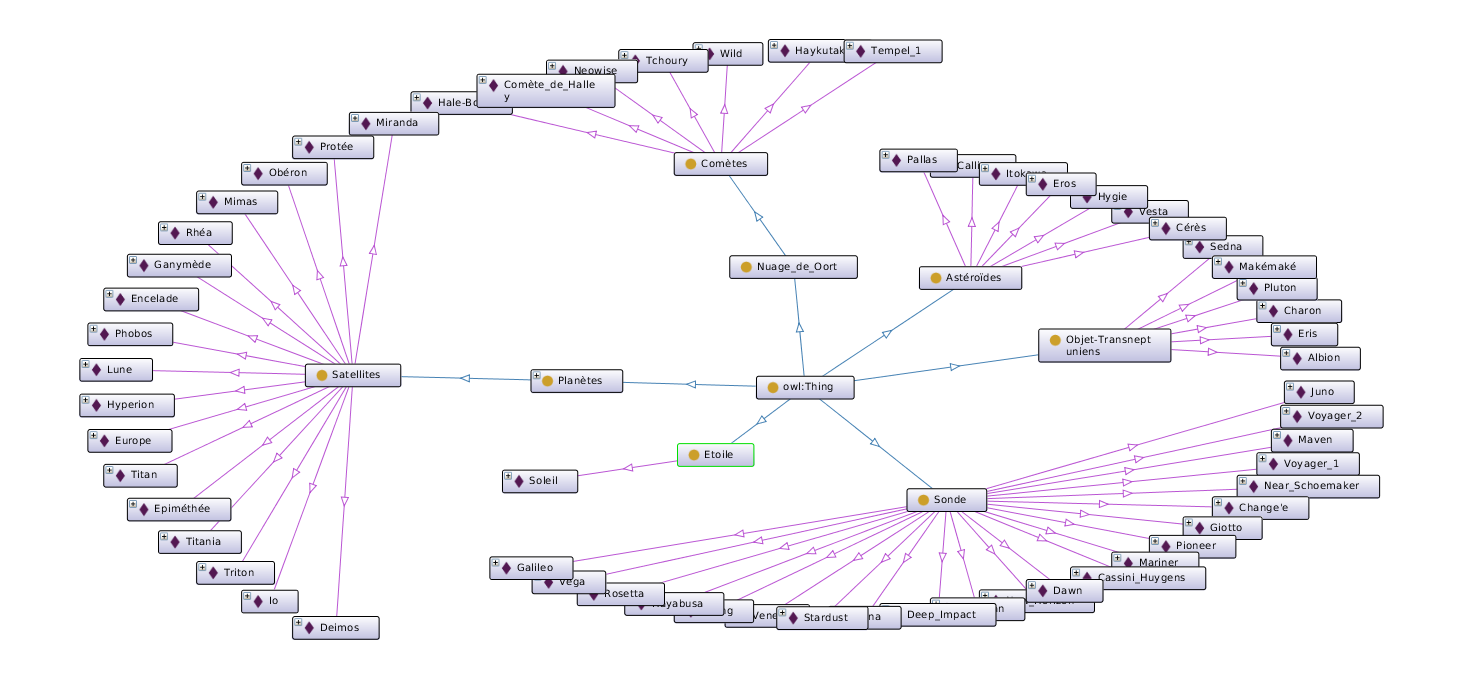
Requête : Astres and (rayon some xsd:float[ > 1000.0]) and (distance some xsd:float[ <= 671100]) and aEtéVisitéPar some Sonde  
  
Nous avons retrouvé Io par son rayon, sa distance à Jupiter autour de laquelle elle gravite et du fait qu’elle a été visitée par une sonde, Juno.

# 

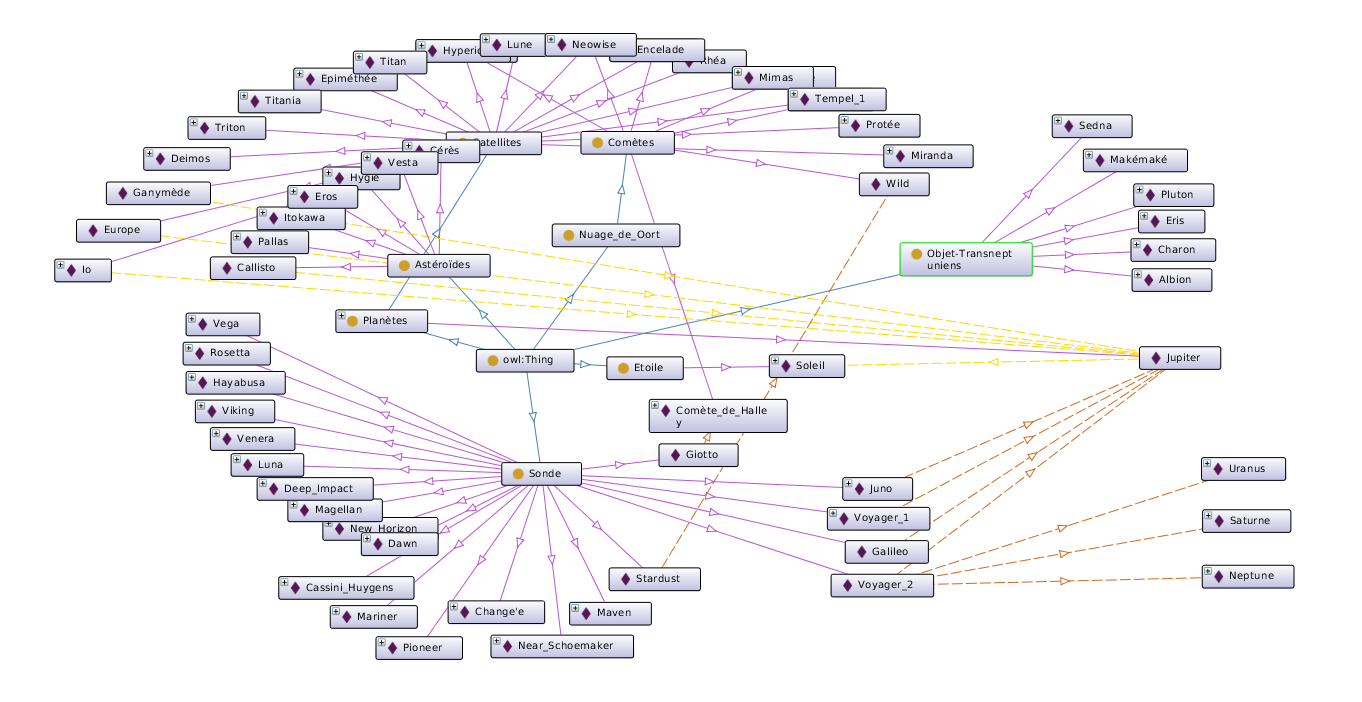
# IV- Visualisations

Pour réaliser les visualisations de cette partie nous avons utilisé Ontograf qui est intégré à Protégé.

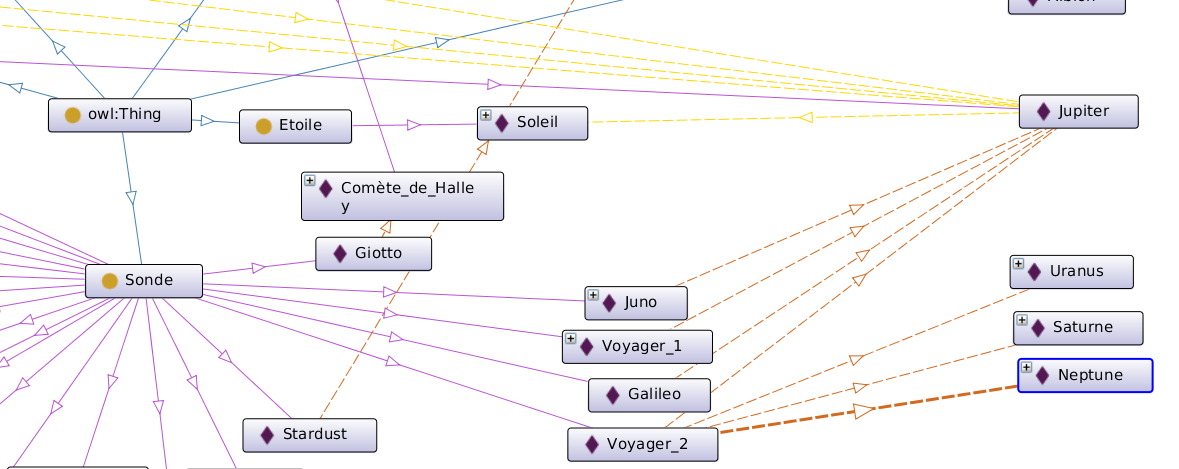
## Visualisation des classes (vue d’ensemble)



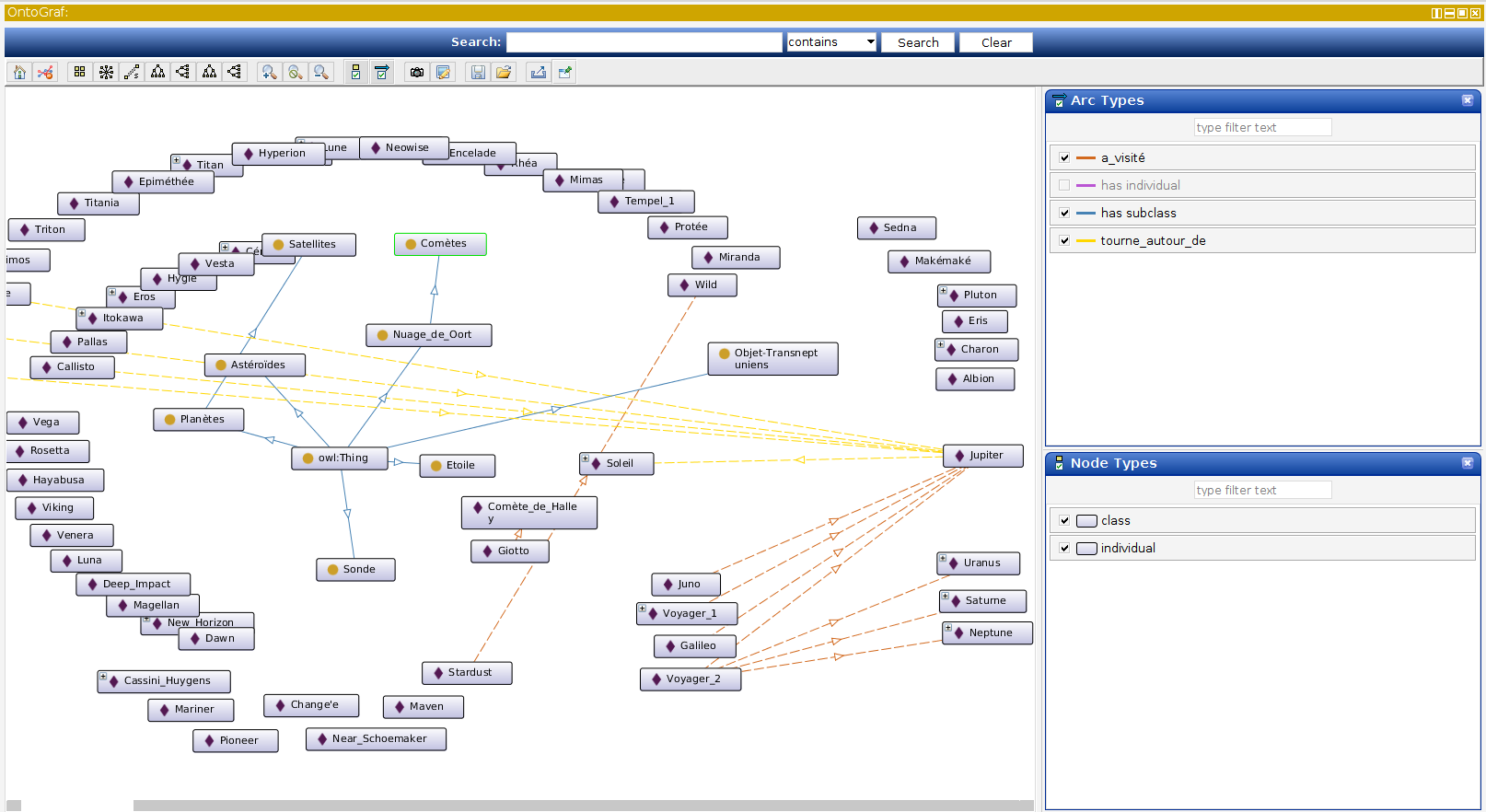
## Visualisation des classes (avec quelques object properties)



## Visualisation des classes et object properties (zoom)



## Visualisation d’une sélection de arc types et node types



# 

# Conclusion

En conclusion, ce projet nous a permis de nous initier au logiciel de construction d’ontologie “Protégé”. Nous avons trouvé le logiciel assez intuitif, complet et puissant (même si nous avons remarqué quelques bugs lors de l’utilisation). Pour la visualisation nous avons expérimenté OntoGraf.

Le sujet nous a confronté aux difficultés de la construction d’une ontologie (identification et définition des concepts et de leurs relations, insertion fastidieuse des données…) et nous a permis de voir l’intérêt des fonctionnalités fournies par Protégé : inférence de connaissances exploitation des données par requêtage, etc. Nous avons également pu éditer le fichier .owl à la main pour accélérer les mises à jours des données.

Ce travail a également été l’occasion pour nous d’en apprendre plus sur le système solaire.

# 

# Livrables

* le présent rapport
* l’ontologie owl *systeme\_solaire.owl*
* le jupyter notebook *DecisionTree.ipynb* (1ère partie du TP)

# 

# Versions de logiciels et bibliothèques

* <https://protege.stanford.edu/> v. 5.5.0
* Hermit Reasoner v. 1.4.3
* [https://www.graphviz.org/](https://www.graphviz.org/download/) v. 2.49.3