Département de formation doctorale en informatique **École doctorale IAEM Lorraine**
UFR STMIA

Construction d'une ontologie de descripteurs en Astronomie à partir de tables de données

MÉMOIRE

soutenu le 27 juin 2005

pour l'obtention du

DEA Informatique de l'université Henri Poincaré – Nancy 1

(Ecole Doctorale IAEM Lorraine / DEA Informatique de Lorraine)

par

Alexandre Richard

Composition du jury

	Noëlle Carbonell	Professeur UHP-Nancy1
	Didier Galmiche	Professeur UHP-Nancy1
	Dominique Méry	Professeur UHP-Nancy1 et ESIAL
Encadrant :	Amedeo Napoli	Directeur de recherche CNRS

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thloria.

Remerciements

Je tiens à adresser mes plus vifs remerciements à Amedeo Napoli, chef du projet ORPAILLEUR, pour avoir encadré ce travail et pour m'avoir fait profiter de ses conseils et de son expérience.

Je remercie également l'Observatoire Astronomique de Strasbourg et en particulier Sébastien Derriere et Andrea Preite Martinez pour leur contribution à ce travail.

Je remercie Emmanuel Nauer pour ses conseils et sa contribution à ce travail.

Je remercie Rim Alhulou, Yannick Toussaint et Mathieu d'Aquin pour leur aide précieuse.

Je remercie l'ensemble de l'équipe ORPAILLEUR.

Enfin, je tiens à remercier Noëlle Carbonell, Didier Galmiche et Dominique Méry, professeurs à l'Université Henri Poincaré - Nancy1 pour m'avoir fait l'honneur de participer à mon jury, ainsi que tous ceux qui se sont intéressés à ce travail.

Résumé

Les descripteurs UCD (United Content Descriptors) sont un moyen standardisé de description des corps célestes en astronomie, permettant une compatibilité entre des sources de données astronomiques d'origines très différentes. Cependant, une des limites actuelles des UCD est l'absence de structure permettant de raisonner sur ces descripteurs. Notre travail concerne la mise en place d'une ontologie des UCD, structure qui associe une sémantique à des termes et qui permet de raisonner par rapport à cette sémantique. Comme exemple d'application possible de cette ontologie, nous avons procédé à la mise en correspondance d'UCD avec des descriptions issues de catalogues d'astronomie à l'aide d'une classification partielle s'appuyant sur des méta-données.

Mots-clés: ontologies, UCD, classification d'instances, méta-données

Abstract

The Unified Content Descriptors (UCD) are standardized descriptors used to describe celestial bodies in astronomy, making astronomical data from various sources compatible. Still, one the current limitations of UCD is the lack of structure for reasoning on these descriptors. Our work consists in setting up a UCD ontology in order to associate a semantic with the terms, thus enabling reasoning with this semantic. As an example of the possible uses of this ontology, we have worked on associating UCD with existing descriptions in astronomical catalogues using a metadata-based partial classification.

Keywords: ontologies, UCD, instance classification, metadata

Table des matières

Table des figures

Introduction

Nos travaux s'inscrivent dans le cadre du projet de recherche *"Masse de données en Astronomie"* dont les acteurs sont le CDS - Centre de Données en astronomie de Strasbourg[1] - et l'équipe ORPAILLEUR du LORIA[2] - Laboratoire Lorrain de Recherche en Informatique et ses Applications.

Ce projet inclut des travaux sur les descripteurs UCD (United Content Descriptors), qui sont un moyen standardisé et universel de description des corps célestes en astronomie. Ceci autorise une compatibilité entre des sources de données astronomiques d'origines très différentes. Cependant, il n'existe pas à l'heure actuelle de structure permettant de raisonner sur ces descripteurs. L'enjeu de notre travail est donc la mise en place d'une telle structure, sous la forme d'une ontologie.

Les possibilités de raisonnement offertes par l'existence d'une ontologie incluent entre autres la vérification de cohérence et la désambiguïsation des UCD, ou encore leur classification les uns par rapport aux autres. Comme exemple d'application de notre ontologie, nous nous sommes intéressés à une des préoccupations actuelles concernant les UCD : la mise en correspondance d'UCD avec des descriptions issues de catalogues d'astronomie à l'aide d'une classification partielle utilisant des méta-données.

Le mémoire est organisé de la façon suivante : le premier chapitre de ce mémoire présente le cadre applicatif de nos travaux : les UCD et les catalogues d'astronomie. Le deuxième est une introduction aux ontologies ; nous y présentons également le travail de construction de l'ontologie des UCD qui est au cœur de notre approche. Enfin, le troisième chapitre couvre un exemple d'exploitation de cette ontologie pour la mise en correspondance d'UCD avec des descriptions issues de catalogues d'astronomie.

[1] http ://cdsweb.u-strasbg.fr/
[2] http ://www.loria.fr/

Chapitre 1

Cadre des travaux

1.1 Problématique générale

Les UCD sont appelés à devenir le vocabulaire universel de description des corps célestes en astronomie. Ils ont été conçus pour être exploitables aussi bien par des êtres humains que des machines. Cependant, il n'existe pas à l'heure actuelle de structure permettant de raisonner sur ces descripteurs. L'enjeu de notre travail est donc la mise en place d'une telle structure, sous la forme d'une ontologie.

Les intérêts de construire une ontologie sont multiples et incluent entre autres la vérification de cohérence et la désambiguïsation des UCD et leur classification les uns par rapport aux autres [Uschold and Gruninger, 1996] [Staab and Studer, 2004]. Comme exemple d'application des possibilités offertes par notre ontologie, nous nous sommes intéressés au problème suivant : la mise en correspondance d'UCD avec des descriptions existantes. En effet, s'il est possible d'utiliser directement les UCD comme descripteurs dans des documents nouvellement créés, un des problèmes qui se pose est d'associer des UCD à des descriptions existantes. En particulier, les UCD devraient à terme être utilisés dans des catalogues d'astronomie. Ainsi, l'exemple ci-dessous d'une description issue dans un catalogue d'astronomie :

```
Bytes   Format  Units   Label  Explanations
66- 72  F7.1    mas/yr  pmRA   Proper motion mu_alpha.cos(delta), ICRS(T12)
```

qui correspond au mouvement propre d'un corps céleste exprimé via la mesure de l'ascension droite[3] de ce corps doit être associée à l'UCD correspondant à cette description, soit : `pos.pm.ra`.

Dans les sections suivantes, nous présentons le cadre applicatif de ce problème : les catalogues d'astronomie et les UCD.

1.2 Les catalogues d'astronomie

Les catalogues d'astronomie auxquels nous faisons référence dans ce mémoire sont des ensembles de données astronomiques que l'on peut trouver ou bien seuls ou bien au sein d'articles d'astronomie où ils centralisent alors les données auxquelles fait référence l'article. Depuis 1993,

[3] l'ascension droite est une des composantes des coordonnées équatoriales en astronomie

l'accent à été mis sur les versions électroniques de ces catalogues, qui constituent un vecteur de publication ne cessant de prendre de l'importance.

Dans ce mémoire, nous ferons toujours référence aux versions électroniques de ces catalogues.

1.2.1 Composition des catalogues

Les propositions de standards [Ochsenbein, 2000] pour la description des catalogues sont à présent largement suivies. Selon ces standards, un catalogue est un arbre de fichiers ASCII organisés à la façon d'une arborescence Unix. Parmi les fichiers de cette arborescence, on trouve en particulier :

- des fichiers de données contenant les données astronomiques, le plus souvent sous forme de tables ;
- un fichier nommé `ReadMe` qui est le fichier de description du catalogue.

1.2.2 Le fichier `ReadMe`

Chaque catalogue possède un fichier `ReadMe` qui fournit toutes les méta-données relatives au catalogue et permet son interprétation par des procédures automatisées. Pour mieux comprendre le rôle de ce fichier `ReadMe`, considérons l'exemple de table de données de la Figure 1.1. Il s'agit d'une table issue du catalogue I/221. Son `ReadMe` est donné en Figure 1.2 page 5.

MACS	RAJ2000 "h:m:s"	DEJ2000 "d:m:s"	Npos	Mag mag	PosFlag	MagFlag	BochumFlag
2314-769#001	23 14 43.779	-76 57 01.19	1	16.60	0	0	0
2315-768#001	23 15 16.057	-76 52 36.74	1	17.65	0	0	0
2315-769#001	23 15 31.900	-76 58 28.66	1	16.56	0	0	0
2315-767#001	23 15 37.294	-76 43 36.98	1	17.49	0	0	0
2315-768#002	23 15 42.722	-76 52 49.34	1	17.80	0	0	0
2315-767#002	23 15 44.318	-76 44 10.77	1	17.85	0	0	0
2315-768#003	23 15 45.469	-76 53 25.46	1	18.48	0	0	0
2315-768#004	23 15 47.222	-76 52 59.27	1	18.53	0	0	0
2315-766#001	23 15 52.018	-76 39 01.74	1	15.96	0	0	0
2316-767#001	23 16 00.353	-76 42 51.85	1	17.44	0	0	0
2316-766#001	23 16 00.892	-76 39 06.30	1	18.18	0	0	0
2316-766#002	23 16 16.574	-76 40 38.09	1	17.28	0	0	0
2316-765#001	23 16 22.416	-76 34 58.08	1	17.98	0	0	0
2316-767#002	23 16 27.057	-76 46 25.45	1	18.38	0	0	0
2316-766#003	23 16 28.762	-76 41 45.75	1	17.40	0	0	0
2316-765#002	23 16 30.370	-76 35 11.23	1	17.71	0	0	0
2316-768#001	23 16 33.549	-76 50 05.22	1	18.21	0	0	0
2316-769#001	23 16 34.501	-76 55 09.21	1	17.24	0	0	0

FIG. 1.1 – Extrait de table de données du catalogue I/221.

```
I/221               The Magellanic Catalogue of Stars - MACS (Tucholke+ 1996)
================================================================================
The Magellanic Catalogue of Stars - MACS
     Tucholke H.-J., de Boer K.S., Seitter W.C.
    <Astron. Astrophys. Suppl. Ser., 119, 91-98 (1996)>
    <The Messenger 81, 20 (1995)>
    =1996A&AS..119...91T
    =1995Msngr..81...20D
================================================================================
ADC_Keywords: Magellanic Clouds ; Positional data

Description:
    The Magellanic Catalogue of Stars (MACS) is based on scans of ESO
    Schmidt plates and contains about 244,000 stars covering large areas
    around the LMC and the SMC. The limiting magnitude is B<16.5m and the
    positional accuracy is better than 0.5" for 99% of the stars. The
    stars of this catalogue were screened interactively to ascertain that
    they are undisturbed by close neighbours.


File Summary:
--------------------------------------------------------------------------------
 FileName    Lrecl    Records    Explanations
--------------------------------------------------------------------------------
ReadMe          80          .    This file
lmc.dat         52     175779    The Large Magellanic Cloud
smc.dat         52      67782    The Small Magellanic Cloud
--------------------------------------------------------------------------------

Byte-by-byte Description of file: lmc.dat smc.dat
--------------------------------------------------------------------------------
   Bytes Format  Units   Label    Explanations
--------------------------------------------------------------------------------
   1- 12  A12    ---     MACS     Designation
  14- 15  I2     h       RAh      Right Ascension J2000 , Epoch 1989.0 (hours)
  17- 18  I2     min     RAm      Right Ascension J2000 (minutes)
  20- 25  F6.3   s       RAs      Right Ascension J2000 (seconds)
      27  A1     ---     DE-      Declination J2000 (sign)
  28- 29  I2     deg     DEd      Declination J2000 , Epoch 1989.0 (degrees)
  31- 32  I2     arcmin  DEm      Declination J2000 (minutes)
  34- 38  F5.2   arcsec  DEs      Declination J2000 (seconds)
      40  I1     ---     Npos     Number of positions used
  42- 46  F5.2   mag     Mag      ?=99.00 Instrumental Magnitude
                                        (to be used only in a relative sense)
      48  I1     ---     PosFlag  [0,1] Position Flag   (0: ok,
                                        1: internal error larger than 0.5")
      50  I1     ---     MagFlag  [0,1] Megnitude Flag  (0: ok,
                                        1: bad photometry or possible variable)
      52  I1     ---  BochumFlag *[0] Bochum Flag
--------------------------------------------------------------------------------
Note on BochumFlag: 1 if in Bochum catalog of astrophysical information
    on bright LMC stars) (yet empty)
--------------------------------------------------------------------------------

Author's address:
    Hans-Joachim Tucholke    <tucholke@astro.uni-bonn.de>

================================================================================
(End)            Hans-Joachim Tucholke [Univ. Bonn]                  20-Nov-1995
```


FIG. 1.2 – ReadMe du catalogue I/221.

Chaque colonne de la table de données correspond à un type de mesure ou d'observation qui est décrite par un en-tête ou `label` composé d'un seul mot (ex : RAJ2000). Cet en-tête est explicité dans le fichier `ReadMe` par une ligne d'un tableau nommé `Byte-by-byte description of file`. Dans notre travail, nous nous concentrerons sur l'exploitation de ce type de tableau que nous décrivons maintenant ; une description exhaustive des fichiers `ReadMe` est cependant disponible dans [Ochsenbein, 2000].

Les deux premières colonnes du tableau ne décrivant que le format informatique des données, les colonnes qui nous intéressent sont les 3 suivantes :

- `Units` qui donne l'unité de la mesure (dans le cas où la colonne décrite contient des mesures)
- `Label` qui donne l'en-tête de colonne à laquelle correspond la ligne du `ReadMe`
- `Explanations` qui donne une courte explication textuelle de la colonne

Dans le problème d'association d'UCD à une description que nous souhaitons résoudre, la description que nous considérerons en entrée du système sera le contenu de ces 3 colonnes.

1.3 Les UCD

1.3.1 Présentation des UCD

Les UCD (Unified Content Descriptor) sont un moyen de description formel de contenus astronomiques [Derriere et al., 2004] contrôlé par l'Alliance de l'Observatoire Virtuel International (IVOA[4]). Le vocabulaire servant à construire les UCD est contrôlé pour éviter au maximum les ambiguïtés ; il est restreint pour éviter la prolifération de termes et de synonymes. Les UCD servent à décrire des mesures ou des informations sur ces mesures, notamment le type des mesures. Par exemple, `phys.temperature` est un UCD qui renvoie à une température.

Les UCD ont déjà connu une révision avec le passage de la version UCD1 à la version UCD1+ [Derriere et al., 2004] [Derriere and Preite Martinez, 2004]. Cette révision s'est en particulier traduite par l'adoption d'une nouvelle syntaxe. Nos travaux concernent les UCD1+ et toutes les références aux UCD dans ce mémoire renvoient aux UCD1+.

Un objectif d'interopérabilité

Les UCD ont tout d'abord pour vocation de permettre l'interopérabilité entre des sources de données hétérogènes. En effet, les astronomes exploitent des sources de données variées dont les contenus présentent des hétérogénéités telles que :

- des descriptions dans des langues différentes
  exemple : `Motion in Right Ascension (h:m:s/yr)`
  `Mouvement (mesuré en ascension droite - h:m:s/a)`
  représentent tous les deux le concept du mouvement d'un corps céleste décrit par l'évolution de son ascension droite[5]*, celle-ci étant mesurée en heures : minutes : secondes par an.*
- des raccourcis d'écriture différents pour une même description
  exemple : `posang` / `pa` / `apa`
  sont trois raccourcis différents de l'expression "position angle" qui exprime une orientation décrite par un angle.

[4] http ://www.ivoa.net/

[5] l'ascension droite est une des composantes des coordonnées équatoriales en astronomie

L'utilisation d'un langage au vocabulaire contrôlé doit permettre des descriptions homogènes et non-ambiguës des concepts manipulés par les membres de l'Observatoire Virtuel International (IVO).

1.3.2 Syntaxe des UCD1+

L'ensemble des règles assurant la validité d'un UCD est détaillé dans [Derriere et al., 2004] et [Derriere and Preite Martinez, 2004]. Ce qui suit en reprend les points essentiels.

Un UCD1+ est une chaîne de caractères constituée de mots, eux-mêmes constitués d'atomes. Les mots sont séparés par des points-virgules et les atomes par des points.

Exemples :

- `pos.eq.ra;meta.main`
  est un UCD
- `pos.eq.ra`, `meta.main`
  sont des *mots*
- `pos`, `eq`, `ra`, `meta`, `main`
  sont des *atomes*

Un UCD1+ est composé au minimum de un mot, et chaque mot est composé d'au moins un atome. Le contrôle des UCD1+ vient du fait que les mots sont validés. En effet, les briques de base des UCD1+ sont les mots autorisés et non les atomes.

Règles lexicales

Les UCD sont insensibles à la casse et leur lexique obéit à la grammaire suivante (présentée sous sa forme Backus-Naur) :

```
<alpha>           ::=  a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z
                       A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z
<digit>           ::=  0|1|2|3|4|5|6|7|8|9
<char>            ::=  <alpha>|<digit>|-|_
<semicolon>       ::=  ;
<period>          ::=  .
<colon>           ::=  :
<word-component>  ::=  <alpha>|<digit>|<word-component><char>
<namespace-ref>   ::=  <word-component>
<word>            ::=  <word-component>|<word><period><word-component>
<nword>           ::=  <namespace-ref><colon><word>|<word>
<\textsc{ucd}>    ::=  <nword>|<\textsc{ucd}><semicolon><nword>
```

Remarques :

- Les caractères d'espacement ou de tabulation sont interdits dans les UCD.
- Les caractères autorisés pour les UCD1+ sont des lettres *minuscules*, *majuscules*, des *chiffres*, le *tiret* et le *souligné*.
- Le *point*, le *point-virgule* et le *deux-points* sont des caractères spéciaux utilisés uniquement pour la composition d'atomes, de mots et d'espaces de noms[6].

[6] Il est possible d'utiliser des espaces de noms *(namespaces)*, marqués par la présence d'un deux-points. Toutefois, l'usage d'espaces de noms n'est pas recommandé sauf à titre temporaire pour des mots n'ayant pas encore été validés par l'IVOA.

Exemples d'UCD correctement écrits :

- `pos` : représente le concept de position
- `stat.error;pos.eq.ra` : représente une erreur sur la mesure d'une ascension droite
- `stat.error;pos.eq.ra;stat.max` et `STat.error;POS.eq.ra;ivoa:stat.max` représentent une erreur statistique sur le maximum d'une mesure d'ascension droite et sont identiques (les UCD sont insensibles à la casse et les espaces de noms sont optionnels)

Règles syntaxiques

La grammaire que nous venons de voir autorise la bonne écriture des UCD1+ mais n'en garantit pas la validité. Pour être valide, un UCD1+ doit également vérifier les règles suivantes : n'être composé que de mots valides et respecter les règles de composition ci-après s'il est composé de plus d'un mot.

Les règles de composition se présentent sous la forme suivante : dans la liste des mots valides, chaque mot est associé à un code qui définit de quelle façon il peut intervenir dans un UCD composé. Les codes des UCD1+ pour la révision la plus récente sont :

- P : mot nécessairement en première place dans un UCD
- S : mot ne pouvant être en première place dans un UCD
- Q : mot pouvant être à une place quelconque dans un UCD
- E : mot nécessairement suivi de 1 mot de la catégorie *em*, pouvant être à une place quelconque dans un UCD[7]
- C : mot nécessairement suivi de 2 mots de la catégorie *em*, pouvant être à une place quelconque dans un UCD[8]

Ainsi, si `phot.color` possède le code `C`, alors :

- `phot.color;em.opt.B` est non valide (un seul mot de type `em` après `phot.color`)
- `phot.color;em.opt.B;em.opt.U` est valide (deux mots de type `em` après `phot.color`)

Nous avons fait la synthèse de toutes ces restrictions sous la forme de la grammaire suivante :

[7] Les mots de la catégorie *em* sont les mots (valides) commençant par l'atome *em* (exemple : `em`, `em.wl`, `em.wl.central`), ils représentent les concepts liés aux rayonnements électromagnétiques. Ici le mot de la catégorie `em` sert à préciser quelle partie du spectre électromagnétique a été mesuré.

[8] Les mots ayant le code `C` représentent des concepts liés aux couleurs et les 2 mots de de la catégorie `em` servent à exprimer quelles plages d'ondes ont été utilisés pour la computation de la couleur.

```
<wordP>      ::=  [P words]
<wordQ>      ::=  [Q words]
<wordE>      ::=  [E words]
<wordC>      ::=  [C words]
<wordS>      ::=  [S words]
<word-em>    ::=  [em category words]
<namespace>  ::=  [namespaces]
<semicolon>  ::=  ;
<colon>      ::=  :
<nwordP>     ::=  <namespace><colon><wordP>|<wordP>
<nwordS>     ::=  <namespace><colon><wordS>|<wordS>
<nwordQ>     ::=  <namespace><colon><wordQ>|<wordQ>
<nwordE>     ::=  <namespace><colon><wordE><semicolon><word-em>
<nwordC>     ::=  <namespace><colon><wordC><semicolon><word-em><semicolon>
                  <word-em>|<wordC><semicolon><word-em><semicolon><word-em>
<nwordQEC>   ::=  <nwordQ>|<nwordE>|<nwordC>
<nwordPQEC>  ::=  <nwordP>|<nwordPQEC><semicolon><nwordQEC>
<ucd>        ::=  <nwordP>|<nwordPQ>|<nwordPQEC><semicolon><nwordS>
```

Avec :
- `[P words]` représente l'ensemble des mots ayant un code `P`
- `[Q words]` représente l'ensemble des mots ayant un code `Q`
- `[E words]` représente l'ensemble des mots ayant un code `E`
- `[C words]` représente l'ensemble des mots ayant un code `C`
- `[S words]` représente l'ensemble des mots ayant un code `S`
- `[em category words]` représente l'ensemble des mots de la catégorie `em`
- `[namespaces]` représente l'ensemble des domaines de noms

L'intérêt direct de cette grammaire est qu'elle permet de garantir la correction d'un UCD en termes de règles d'association entre mots valides.

Recommandation sur l'ordre des mots

Le dernier point concernant la construction d'UCD composés de plusieurs mots est la recommandation suivante [Derriere et al., 2004] : le premier mot est celui qui a le plus de poids quant au concept que décrit l'UCD, qui bien qu'il ne s'agisse que d'une recommandation, est unanimement appliquée pour la construction des UCD.

Exemples de construction d'UCD tenant compte de cette recommandation :
- *la température maximale d'un instrument.*
  Le concept décrit ci-dessus renvoie avant tout à une *température*, le premier mot sera donc `phys.temperature`. Cette température est celle d'un *instrument*, d'où l'on obtient `phys.temperature;instr`. Enfin, il reste à spécifier que l'on fait référence à une *valeur maximale*, d'où l'UCD complet : `phys.temperature;instr;stat.max`
- *l'erreur sur une magnitude mesurée sur la bande V.*
  La description renvoie à une *incertitude*, le premier mot sera donc `stat.error`. Cette incertitude est relative à une *magnitude*, ce qui va se traduire par `stat.error;phot.mag`. Enfin, on spécifie la bande du spectre où a été faite la mesure pour obtenir l'UCD

complet : `stat.error;phot.mag;em.opt.V`

Cette recommandation repose sur des critères sémantiques et ne peut être complètement traduite en termes syntaxiques. Cependant, elle est cohérente avec les codes spécifiant les règles d'association des mots.

Ce chapitre nous a permis de situer le cadre de nos travaux. Dans le chapitre suivant, nous présentons les ontologies et en particulier celle que nous avons construite.

Chapitre 2

Une ontologie des mots pos des UCD1+

2.1 Introduction aux ontologies

Beaucoup de propositions ont été faites pour définir ce qu'est une ontologie. La plus connue en informatique est probablement celle de Gruber [Gruber, 1993], à savoir qu'*une ontologie est la spécification explicite d'une conceptualisation.* Dans cette définition, le terme *conceptualisation* renvoie à un modèle abstrait prenant la forme d'un ensemble de définitions de concepts et de propriétés de concepts. Par ailleurs, la notion de *spécification explicite* renvoie au fait que le modèle doit être représenté dans un langage de représentation de connaissances (muni d'une syntaxe et d'une sémantique associée) pour que le modèle soit utilisable aussi bien par des machines que des êtres humains [Napoli, 1997][Fensel et al., 2003]. De plus, une ontologie est censée fournir une plate-forme favorisant l'interopérabilité d'un ensemble de modules logiciels.[Noy and McGuiness, 2000].

2.1.1 Description des ontologies

Formalisme et composants des ontologies

L'ontologie que nous avons construite est codée dans le formalisme de représentation des logiques de descriptions [Napoli, 1997] [Alhulou, 2003] [Baader, 2003]. Celui-ci est fondé sur trois types d'entités :

- les *concepts* (ou *classes*) qui représentent des classes d'individus ayant des propriétés communes. Ces individus sont l'*extension* du concept
  exemple : le concept `Personne` *représente un ensemble d'êtres humains, cet ensemble d'êtres humains est l'extension du concept* `Personne`.
- des *individus* (ou *instances* des concepts).
  exemple : les individus `Jean` *et* `Marie` *sont des instances du concept* `Personne`.
- les *rôles* (ou *propriétés*) qui représentent des relations binaires entre les concepts.
  exemple : le rôle `estEnfantDe` *entre deux concepts* `Personne` *désigne la relation de filiation entre des instances de ces concepts.*

Chaque rôle possède un *domaine* et un *co-domaine*[9] :

- le domaine est le concept où est définit le rôle (concept de départ)
- le co-domaine est le concept avec lequel le rôle établit une relation (concept d'arrivée).
  exemple : la relation `pratiqueSport` *qui indique qu'une personne (instance du concept*

[9]Le co-domaine est également appelé *range*

`Personne`) pratique un sport (instance du concept `Sport`) a pour domaine le concept `Personne` et pour co-domaine le concept `Sport`.

Les concepts (et les rôles le cas échéant) sont organisés en une hiérarchie par la relation de *subsomption*, notée $\sqsubseteq$, où $D \sqsubseteq C$ se lit "C subsume D" ou "D est subsumé par C". Un concept C subsume un concept D si et seulement si C est plus général que D. De plus, chaque concept partage les propriétés de ses subsumants dans la hiérarchie de concepts.
Remarque : On nomme `Top` ou `Thing`, noté généralement $\top$, le concept le plus général. Ce concept est à la racine de la hiérarchie des concepts et subsume tous les autres concepts. Lorsque les rôles sont organisés en hiérarchie, il existe une racine nommée $\mathrm{d}\top_R$.

Un concept définit des conditions d'appartenance d'un individu à l'extension de ce concept. Suivant ces conditions, le concept est *primitif* ou *défini* :

- s'il ne s'agit que de conditions nécessaires, alors le concept est *primitif*.
  exemple : le concept `Personne` introduit par `Personne` $\sqsubseteq \top$ est primitif
- s'il s'agit d'un ensemble de conditions nécessaires et suffisantes (une instance X fait fait partie de l'extension d'un concept C si et seulement si X a les mêmes propriétés que C), alors le concept est *défini* et l'ensemble de conditions est la *définition* de ce concept. Les concepts définis sont introduits par une équivalence de concepts notée $\equiv$, où $C \equiv D$ signifie $C \sqsubseteq D$ et $D \sqsubseteq C$.
  exemple : le concept introduit par `Sportif` $\equiv$ `Personne` $\sqcap$ (`pratiqueSport` ≥ 1)est défini (Si X est une personne et si X pratique au moins un sport, alors X est un sportif et un sportif est une personne qui pratique au moins un sport).

Les définitions de concepts font intervenir les constructeurs suivants :

- la *conjonction de concepts*, notée $\sqcap$
- la *cardinalité* qui fixe le nombre minimal et maximal de valeurs élémentaires que peut prendre un rôle. Elle est notée $\leq$, $\geq$ ou $=$ suivant qu'il s'agit d'une cardinalité maximale, minimale ou exacte.

Un exemple

Voici une ontologie reprenant les divers éléments définis précédemment :
Concepts primitifs :

`Personne` $\sqsubseteq \top$
`Sport` $\sqsubseteq \top$
`Loisir` $\sqsubseteq \top$
`Alpinisme` $\sqsubseteq$ `Sport`
`Danse` $\sqsubseteq$ `Loisir` $\sqcap$ `Sport`

Concepts définis :

`Sportif` $\equiv$ `Personne` $\sqcap$ (`pratiqueSport` ≥ 1)
`Alpiniste` $\equiv$ `Sportif` $\sqcap$ (`pratiqueAlpinisme` ≥ 1)
`Danseur` $\equiv$ `Personne` $\sqcap$ (`pratiqueDanse` ≥ 1)
`Dilettante` $\equiv$ `Personne` $\sqcap$ (`aLoisir` ≥ 5) $\sqcap$ (`pratiqueSport` ≥ 3)

Rôles :

`pratiqueSport` $\sqsubseteq \top_R$
`aLoisir` $\sqsubseteq \top_R$
`pratiqueAlpinisme` $\sqsubseteq$ `pratiqueSport`
`pratiqueDanse` $\sqsubseteq$ `pratiqueSport` $\sqcap$ `aLoisir`

Un exemple de représentation graphique des hiérarchies de concepts et de rôles est donné en

Figure 2.1 et Figure 2.2

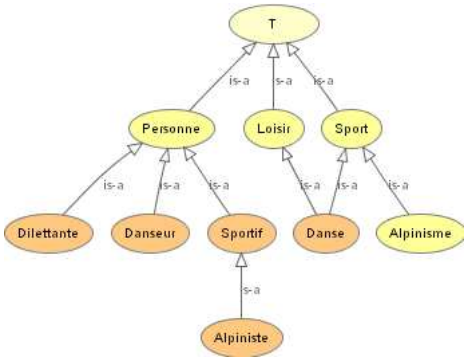

FIG. 2.1 – Hiérarchie des concepts de l'exemple.

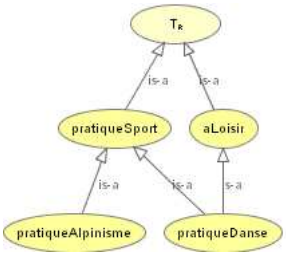

FIG. 2.2 – Hiérarchie des rôles de l'exemple.

2.1.2 Intérêt des ontologies

Les avantages de la mise en place d'une ontologie viennent principalement de la sémantique qu'elle met sur les éléments représentés. Ces avantages peuvent être classés en deux grandes catégories [Staab and Studer, 2004] [Uschold and Gruninger, 1996] :

Communication et interopérabilité
La désambiguïsation et la vérification de cohérence apportée par une ontologie facilitent l'échange d'information ou de connaissances venant de sources hétérogènes entre des êtres humains et/ou des machines. Dans le cadre de recherche d'information, avoir une définition des concepts par leurs propriétés et non par des mots-clefs permet d'effectuer des recherches fondées sur la sémantique des concepts. C'est un intérêt majeur dans l'exemple d'exploitation de l'ontologie que nous présentons dans le chapitre 3.

Spécification, intégration et ré-utilisation
La représentation d'un domaine que propose une ontologie peut aider à la spécification de problèmes liés à ce domaine. De plus, la non-ambiguïté et la cohérence garanties par l'ontologie peuvent faciliter l'intégration d'une application à une plate-forme exploitant l'ontologie. Enfin, même si la construction de l'ontologie est influencée par les applications, tant que la structure de l'ontologie n'est pas remise en cause, l'ontologie et les applications peuvent évoluer séparément.

2.2 Une ontologie des mots `pos`

2.2.1 Introduction au processus de construction

Il n'existe pas de méthode unifiée de création d'une ontologie, cependant il s'agit d'un processus itératif du type de celui présenté sur la Figure 2.3, et décrit dans [Staab and Studer, 2004] [Uschold and King, 1995] :

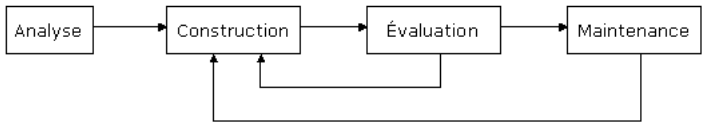

FIG. 2.3 – Construction d'une ontologie.

1. Analyse :
   - que conceptualise l'ontologie ?
   - quelles sont les utilisations de l'ontologie ?
2. Construction :
   - construction des hiérarchies de concepts et de rôles.
   - construction des définitions de concepts.
3. Évaluation :
   - tests de cohérence et test d'utilisation.
   - corrections ou ajustements si nécessaire (boucle sur la phase de construction).
4. Maintenance :
   - utilisation réelle
   - évolutions si nécessaire (boucle sur la phase de construction)

Lors de la création d'une ontologie, un soin particulier doit être apporté au respect des critères suivants [Gruber, 1993] :

Adéquation Une ontologie doit être adaptée à l'usage qui en est fait. Dans une ontologie trop générale ou trop spécifique, les définitions seront trop vagues ou trop complexes (car mettant en jeu des paramètres n'ayant aucun rapport avec le cadre d'exploitation de l'ontologie).

Exemple : dans un cadre où apparaissent seulement des données relatives aux concerts d'artistes de la chanson, le concept suivant a une définition qui n'est pas adaptée, car faisant intervenir les roles `nombreFreresEtSoeurs` *et* `nombreDeVoitures` *qui n'ont rien à voir avec les données relatives aux concerts* :

$$\texttt{StarDeLaChanson} \equiv (\texttt{nombreFreresEtSoeurs} \geq 3) \sqcap (\texttt{publicParConcert} \geq 10000) \sqcap (\texttt{concertsParAn} \geq 50) \sqcap (\texttt{nombreDeVoitures} \geq 5).$$

Clarté Les concepts définis sont toujours préférables aux primitifs.

Cohérence Une ontologie doit être cohérente :
- on ne doit pas pouvoir inférer $\bot$,
- les concepts ne doivent pas être incohérents.

Exemple : on ne doit pas avoir les concepts A et $\neg A$ dans la même ontologie

Extensibilité Une ontologie doit pouvoir être étendue sans remettre en cause sa cohérence. Ceci implique en particulier d'avoir des règles de construction précises.

La construction de l'ontologie nécessite une connaissance suffisante du domaine et donc l'intervention d'experts du domaine tant pour la construction elle-même que pour la vérification de l'ontologie (seule la cohérence peut être vérifiée par un non-expert).

2.2.2 Construction de l'ontologie des mots pos

La catégorie pos

Afin de restreindre le problème complexe de la construction d'une ontologie des UCD, nous avons restreint notre domaine d'étude aux UCD représentant des positions. Les UCD décrivant des positions font principalement intervenir des mots de la catégorie `pos`. La catégorie `pos` est une des 12 catégories des mots valides des UCD1+. Elle est constitué des mots commençant par l'atome `pos`. Ces mots seront représentés par l'expression "mot `pos`" dans ce mémoire. Parmi les mots `pos`, on trouve par exemple :

- `pos` : qui représente le concept de position
- `pos.eq` : qui représente des coordonnées équatoriales
- `pos.eq.ra` : qui représente une ascension droite[10]

Une liste complète des mots `pos` est disponible en Annexe A.

Une ontologie des mots

En réponse aux deux grandes questions de la phase d'analyse, il nous faut :

- construire une ontologie des UCD1+ incluant au moins un mot `pos`
- exploiter cette ontologie pour mettre en correspondance une description issue d'un catalogue d'astronomie avec un UCD.

Cependant, la construction d'une ontologie des UCD incluant au moins un mot `pos` se heurte aux problèmes suivants :

- Il n'existe pas de liste exhaustive des UCD1+ incluant des mots `pos`
- Le nombre d'UCD1+ incluant des mots `pos`, composés de mots valides et respectant les règles de composition vues en 1.3.2 dépasse 10^6

En l'absence d'un moyen de construction automatisé, la construction d'une ontologie d'une telle envergure est inenvisageable. De plus, au delà des problèmes de faisabilité, la taille rendrait l'ontologie difficilement lisible et utilisable. Mais s'il n'est pas possible de construire une ontologie de tous les UCD on peut remarquer que :

- il existe une liste exhaustive des mots `pos`
- il y a actuellement 58 mots `pos` dans cette liste (Ce nombre est sujet à variation avec les révisions des UCD1+ mais l'ordre de grandeur devrait rester inchangé)

Le nombre réduit de mots rend possible la création manuelle d'une ontologie relative aux mots `pos`. Nous avons donc décidé de construire une ontologie de ces mots et d'adapter la phase d'exploitation de l'ontologie pour pouvoir prendre en compte les UCD composés de plus d'un mot.

Choix d'implantation

Le langage retenu pour l'implantation de l'ontologie est le langage OWL[11] proposé par le World Wide Web Consortium (W3C) dans sa version OWL-DL. L'ontologie implantée se présente

[10] l'ascension droite est une des composantes des coordonnées équatoriales

[11] http ://www.w3.org/TR/owl-guide/

sous la forme d'un fichier OWL. Elle a été éditée via le plugin OWL de l'éditeur PROTÉGÉ[12] [Horridge et al., 2004]. Enfin, le système de logique de descriptions RACER[13] a été utilisé pour vérifier de façon automatisée la cohérence de l'ontologie.

Conventions d'écriture

- Par cohérence avec les UCD, la langue de description est l'anglais.
- Les mots :
  - *concept* et *classe*,
  - *rôle* et *propriété*,
  - *individu* et *instance*,

  sont utilisables de manière indifférente (les premiers font partie du vocabulaire des logiques de description, les seconds font partie du vocabulaire de l'implantation en langage OWL).
- Les caractères autorisés pour l'écriture des noms de concepts et de rôles sont les *lettres minuscules* et *majuscules*, ainsi que le *point*, et le *point-virgule*.
- Les noms de concepts commencent par une majuscule, sauf s'il s'agit d'un concept représentant un *mot* auquel cas le nom est identique au mot.
- Les noms de rôles commencent par une minuscule.
- Les noms issus de la contraction de plusieurs mots sont écrits en mettant une majuscule au début de chaque mot.
  exemple : `MotionTypeNutation`, `hasAngularSpeedUnit`.
- L'usage de contractions dans les noms de concepts et de propriétés est à éviter, à l'exception des contractions utilisées dans les UCD.
  exemple : dans un nom, contracter `declination` *en* `dec` *est acceptable car c'est un usage dans les* UCD*, mais contracter ce même mot en* `decl` *est à éviter.*
  Exception : Le concept `CoordinateComponent` a été abbrévié en `Cc`.

Règle de construction

Pour l'exploitation de l'ontologie, nous avons souhaité pouvoir identifier un rôle de manière ambiguë à partir de son co-domaine. Pour permettre cela, nous avons imposé la règle suivante lors de la construction de l'ontologie : **un co-domaine unique et différent pour chaque rôle**.

Processus de construction

Le processus de création de notre ontologie est présenté ci-dessous.

1. **Identification des concepts**
   Règle : un concept représentant un *mot* est nécessairement défini et ne peut représenter qu'un seul mot.
   Exemple : pour nous qui cherchons à représenter les mots `pos`*, des concepts représentant des mots sont par exemple :* `pos.eq`, `pos.eq.ra`, `pos.pm.ra`
2. **Écriture des définitions de concepts et identification des rôles**
   Les définitions se ramenant à des ensembles de rôles, établir les définitions des concepts passe par l'identification des rôles que possède ce concept.

[12] http ://protege.stanford.edu
[13] http ://www.sts.tu-harburg.de/ r.f.moeller/racer/

Exemple : pour établir la définition de ***pos.eq.ra****, nous devons savoir ce qui définit une ascension droite dans notre contexte. Nous obtenons :*
- ***hasOneValue = 1*** *(une ascension droite est une mesure, elle a donc une valeur),*
- ***hasAngleUnit = 1*** *(cette valeur est exprimée avec une unité d'angle),*
- ***hasFrameTypeEq = 1*** *(cette mesure est faite dans un repère équatorial),*
- ***hasCcOriginEqRa = 1*** *(l'ascension droite est mesurée à partir de l'origine des ascensions droites du repère équatorial,).*

Écrire ces définitions suppose que nous avons dans l'ontologie les rôles apparaissant dans ces définitions. Nous construisons donc à ce stade les rôles qui n'existent pas encore dans l'ontologie. De même, construire les rôles implique d'avoir les concepts co-domaines de ces rôles, que nous construisons à ce stade s'ils ne sont pas déjà présents dans l'ontologie, en respectant la règle prévoyant que chaque rôle doit avoir comme co-domaine un concept unique et différent.

3. **Hiérarchisation des concepts**
   La hiérarchie des concepts est organisée par la relation de subsomption.
   - Pour les concepts définis, la hiérarchie découle des définitions : si la définition d'un concept `C1` est plus précise que la définition d'un concept `C2`, alors `C1` $\sqsubseteq$ `C2`.
   - Pour les concepts primitifs, la hiérarchisation est donnée par l'expert qui aide à construire l'ontologie du domaine.

   La hiérarchie complète des concepts est disponible en Figures 2.4 et 2.5 pages 18 et 19.
   Exemple :
   - ***pos*** $\equiv$ *(**hasFrameType** = 1)*
   - ***pos.barycenter*** $\equiv$ *(**hasFrameType** = 1)* $\sqcap$ *(**haBarycentricCoordinates** $\geq$ 2)*

   d'où ***pos.barycenter*** $\sqsubseteq$ ***pos***
   Par contre, ***AngleUnit*** $\sqsubseteq$ ***Unit****, où* ***Unit*** *et* ***AngleUnit*** *sont des concepts primitifs est une relation de subsomption donnée par l'expert*
4. **Hiérarchisation des rôles**
   La hiérarchie des rôles est également organisée par la relation de subsomption, selon la règle suivante : r1 $\sqsubseteq$ r2 *si et seulement si* domaine(r1) $\sqsubseteq$ domaine(r2) et codomaine(r1) $\sqsubseteq$ codomaine(r2). La hiérarchie complète des rôles est disponible en Figure 2.6 page 20.
   Exemple :
   - ***hasUnit*** *a pour domaine* ***Measure*** *et pour co-domaine* ***Unit***
   - ***hasAngleUnit*** *a pour domaine* ***pos.angDistance*** *et pour co-domaine* ***AngleUnit***
   - ***pos.angDistance*** $\sqsubseteq$ ***Measure*** *et* ***AngleUnit*** $\sqsubseteq$ ***Unit***

   d'où ***hasAngleUnit*** $\sqsubseteq$ ***hasUnit****.*
5. **Amélioration éventuelle de la lisibilité**.
   Enfin, on peut souhaiter améliorer la lisibilité de l'ontologie en créant des concepts supplémentaires, subsumants d'un ensemble de concepts plus spécifiques. Toujours dans but de lisibilité, nous avons souhaité que ces concepts facultatifs respectent la condition suivante : l'expert doit juger que le concept ajouté représente quelque chose qui a un sens par rapport à l'ontologie.
   exemple : ***Measure*** $\equiv$ ***(hasValue = 1)*** $\sqcap$ ***(hasUnit = 1)*** *subsume tous les concepts renvoyant à des mesures (les concepts des mesures spécifiques ont tous une valeur et une unité) et représente la mesure en général (ce qui a un sens en astronomie), on peut donc l'intégrer à l'ontologie si on le souhaite.*

Dans ce chapitre, nous avons décrit notre approche pour la mise en place de l'ontologie des mots `pos` des UCD1+. Dans le suivant nous montrons un exemple de son exploitation dans le

cadre du problème d'association d'un UCD à une description textuelle.

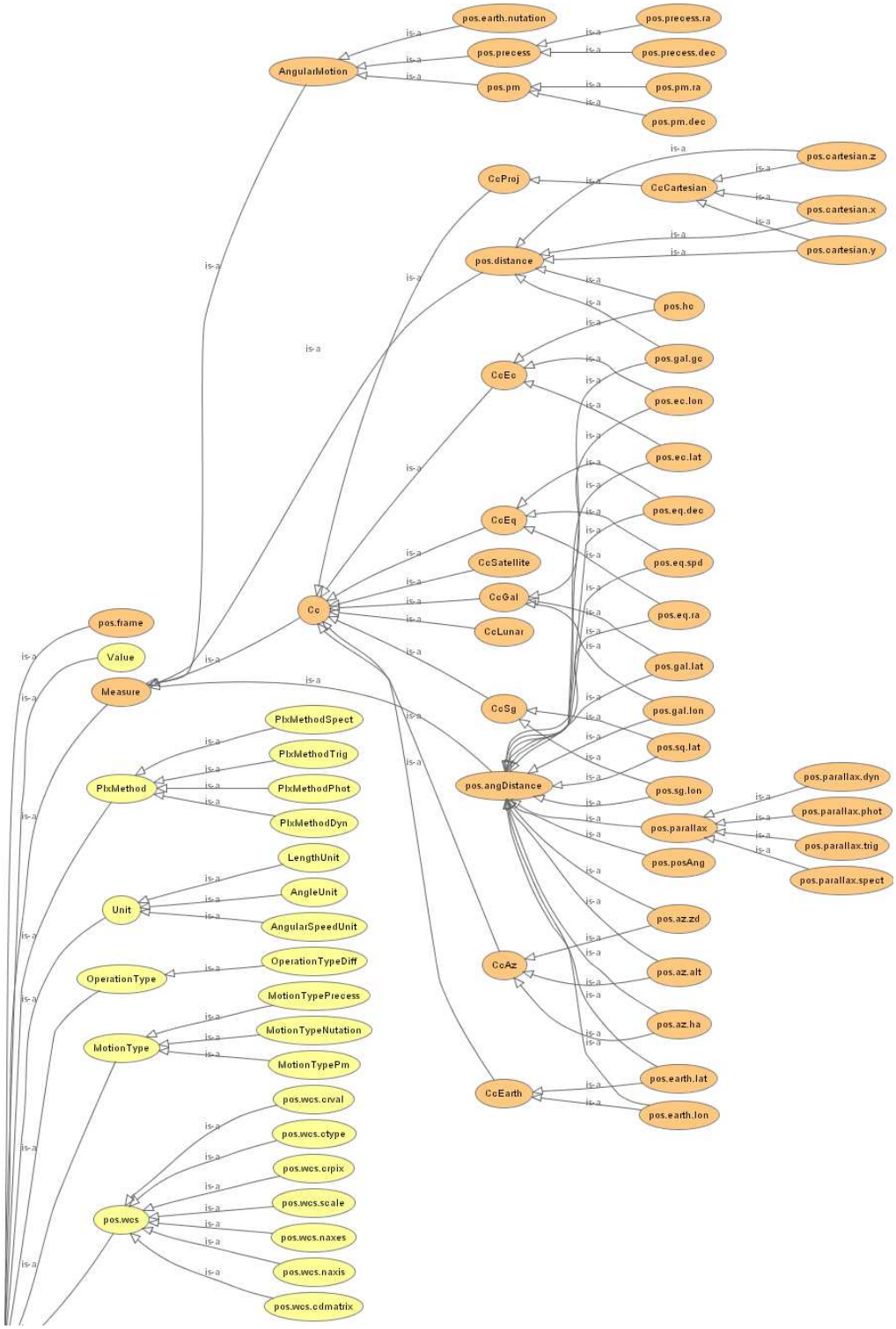

FIG. 2.4 – Hiérarchie des concepts de l'ontologie des mots `pos`.

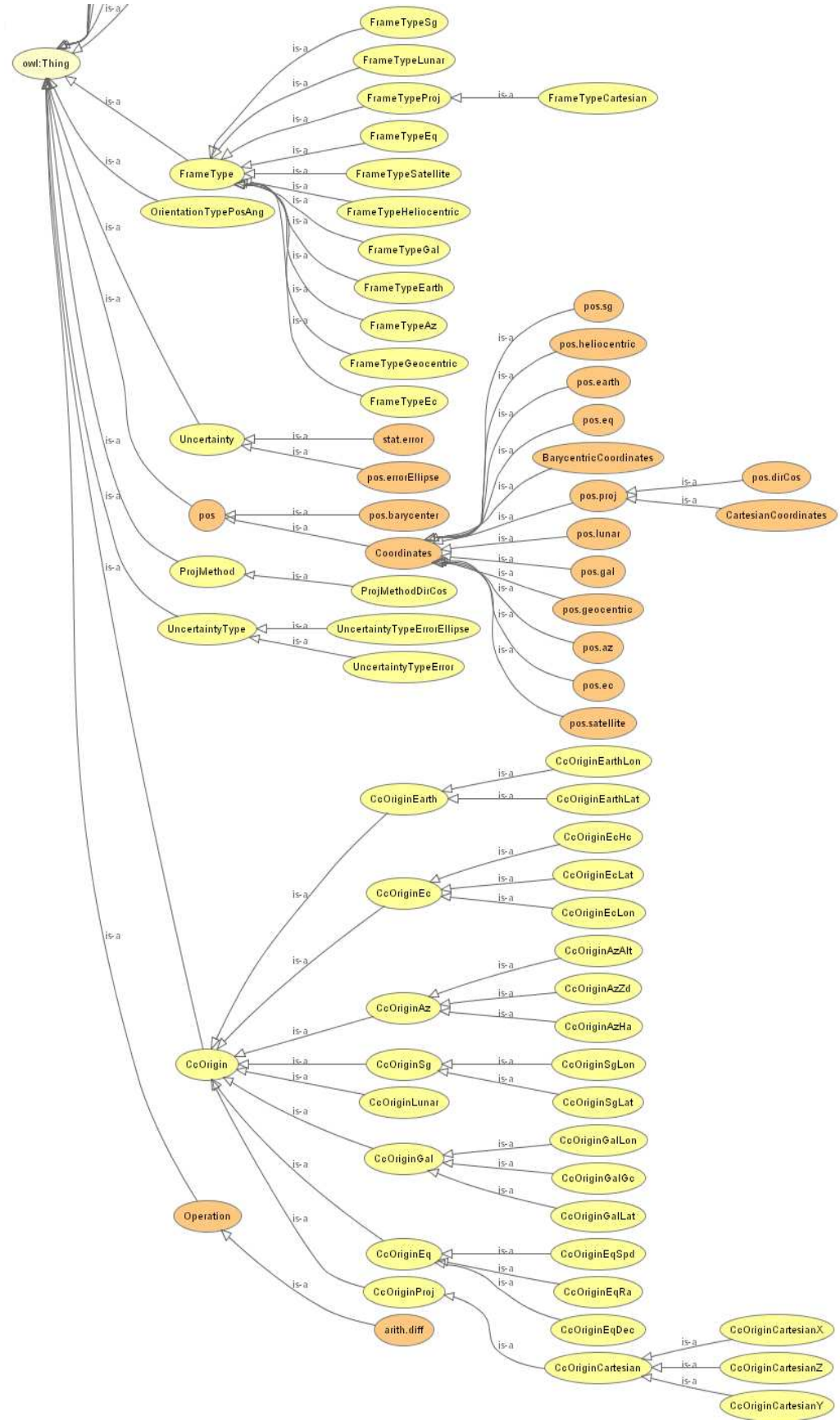

FIG. 2.5 – Hiérarchie des concepts de l'ontologie des mots pos.

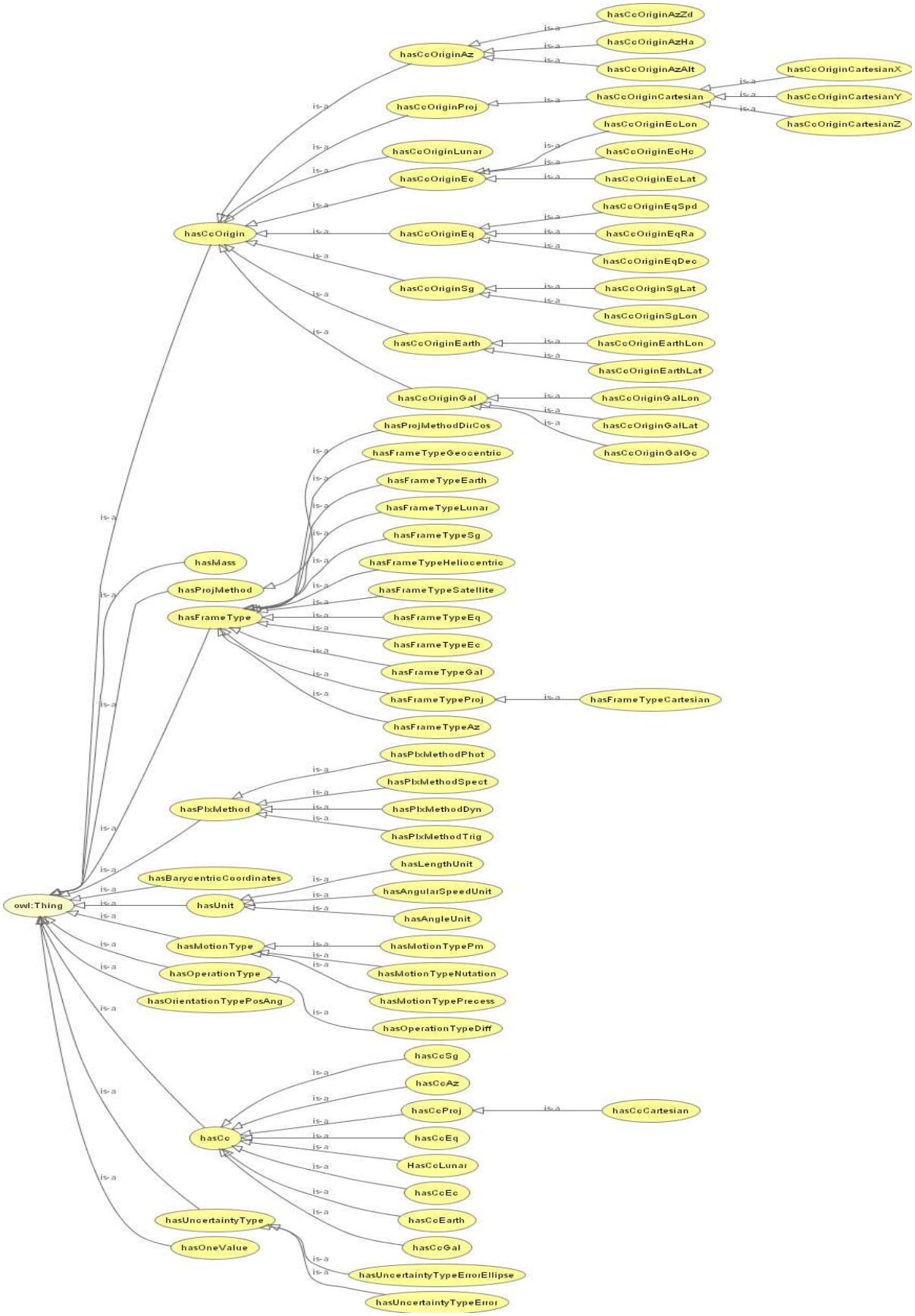

FIG. 2.6 – Hiérarchie des rôles de l'ontologie des mots `pos`.

Chapitre 3

Attribution d'UCD à partir de descriptions textuelles

3.1 Cadre de l'application

Lors de la présentation du cadre de nos travaux dans le chapitre 1, nous avons présenté les catalogues d'astronomie et en particulier le fichier `ReadMe` inclus dans chacun d'entre eux. Nous avons constaté que la partie du `ReadMe` qui nous intéressait était le tableau nommé `Byte-by-byte description of file`. Nous redonnons l'exemple de celui du catalogue I/221 en Figure 3.1.

```
Byte-by-byte Description of file: lmc.dat smc.dat
--------------------------------------------------------------------------------
   Bytes Format  Units   Label    Explanations
--------------------------------------------------------------------------------
    1- 12  A12    ---     MACS     Designation
   14- 15  I2     h       RAh      Right Ascension J2000 , Epoch 1989.0 (hours)
   17- 18  I2     min     RAm      Right Ascension J2000 (minutes)
   20- 25  F6.3   s       RAs      Right Ascension J2000 (seconds)
       27  A1     ---     DE-      Declination J2000 (sign)
   28- 29  I2     deg     DEd      Declination J2000 , Epoch 1989.0 (degrees)
   31- 32  I2     arcmin  DEm      Declination J2000 (minutes)
   34- 38  F5.2   arcsec  DEs      Declination J2000 (seconds)
       40  I1     ---     Npos     Number of positions used
   42- 46  F5.2   mag     Mag      ?=99.00 Instrumental Magnitude
                                          (to be used only in a relative sense)
       48  I1     ---     PosFlag  [0,1] Position Flag   (0: ok,
                                          1: internal error larger than 0.5")
       50  I1     ---     MagFlag  [0,1] Megnitude Flag  (0: ok,
                                          1: bad photometry or possible variable)
       52  I1     ---  BochumFlag *[0] Bochum Flag
--------------------------------------------------------------------------------
```

FIG. 3.1 – Tableau Byte-by-byte description of file du ReadMe du catalogue I/221.

L'étude de ce tableau effectuée dans la section 1.2.2 nous a permis de constater que seules les colonnes ayant les étiquettes `Units`, `Label` et `Explanations` seraient à prendre en compte en entrée du système associant un UCD à une description. Notre but est donc d'associer un UCD à une ligne de ce tableau en ne considérant que le contenu de ces trois colonnes. A l'heure actuelle, le système que nous avons développé n'associe que des UCD composés d'un seul *mot*.

Nous traitons donc ici le cas de l'association d'un UCD composé d'un seul mot `pos` à une ligne d'un `ReadMe`.

3.2 Description du fonctionnement

Le fonctionnement global du système est présenté en figure 3.2. Nous partons d'une ligne de `ReadMe` pour arriver à un tableau trié de concepts de l'ontologie. Dans ce tableau, les concepts les mieux classés sont ceux qui représentent les mots `pos` correspondant le mieux à la ligne de `ReadMe`, autrement dit les UCD correspondant le mieux à la description puisque nous ne considérons que des UCD composés d'un mot `pos`.

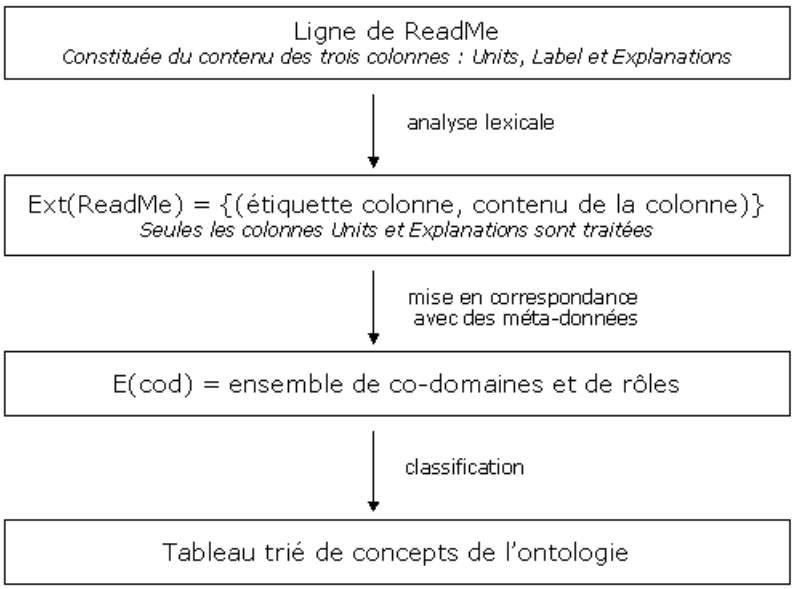

FIG. 3.2 – Schéma global de fonctionnement du système d'attribution d'UCD.

Nous décrivons à présent en détail ce fonctionnement à travers l'exemple de la ligne de ReadMe suivante, tirée du tableau présenté en Figure 3.1 :

```
Units  Label  Explanations
s      RAs    Right Ascension J2000 (seconds)
```

Étape 1 : Analyse lexicale

La première étape est l'utilisation d'un analyseur lexical sur la ligne du `ReadMe` pour en extraire le contenu des colonnes `Units` et `Explanations`. En retour, le système renvoie l'ensemble `{Ext(ReadMe) = {(étiquette de la colonne, contenu de la colonne)}`. En pratique, nous donnons en entrée au système la ligne de `ReadMe` :

```
s RAs Right Ascension J2000 (seconds)
```

et le système rend en sortie :

```
Ext(ReadMe) = {(Units, s), (Explanations, right|ascension|J2000|seconds)}
```

où la *barre verticale* | est un séparateur.

Étape 2 : Utilisation de méta-données

Le but de cette seconde étape est d'obtenir un ensemble E(cod) de co-domaines de rôles à partir de l'ensemble Ext(ReadMe). Pour ce faire, nous exploitons :

Des hypothèses :

- Hypothèse 1 : Un co-domaine unique et différent pour chaque rôle
- Hypothèse 2 : Le co-domaine du rôle identifie le rôle de manière non-ambiguë (cette hypothèse est liée à l'hypothèse 1)

Ces hypothèses peuvent être faites puisque l'ontologie a été construite en respectant une règle prévoyant que chaque rôle devait avoir un co-domaine différent et unique.

Des fichiers intermédiaires :

- Nous appelons ces fichiers "*fichiers de méta-données*",
- Ils sont de la forme : `Méta = {(contenu de colonne, co-domaines de rôle)}`,
- Ils ont été construits à partir de fichiers `ReadMe` de la manière suivante :
  - l'expert fait l'association : contenu de colonne - rôles
  - l'informaticien fabrique le fichier :
    `Méta(étiquette de colonne) = {(contenu de colonne, co-domaines de rôle)}`
    à partir du travail de l'expert et des hypothèses 1 et 2.

En pratique, nous fournissons en entrée l'ensemble Ext(ReadMe) obtenu à l'étape 1 :

`Ext(ReadMe) = {(Units, s), (Explanations, right|ascension|J2000|seconds)}`

Les fichiers de méta-données permettent d'associer des co-domaines aux contenus de colonnes :

`Méta(Units) = {(s, AngleUnit|Value), (mas/yr, AngularSpeedUnit|Value), ... }`
`Méta(Explanations) = {(right, CcOriginEqRa|Value), (ascension, Value), ... }`

Le système donne en sortie l'ensemble `E(cod)` de ces co-domaines (on ignore les doublons) :

`E(cod) = {AngleUnit|CcOriginEqRa|Value}`

Remarque : par les hypothèses 1 et 2, avoir `E(cod)`, c'est aussi avoir l'ensemble des rôles dont les co-domaines sont listés dans `E(cod)`.

Étape 3 : Classification

Dans cette dernière étape, on recherche les concepts de l'ontologie possédant les rôles de `E(cod)` (puisque par hypothèse, avoir les co-domaines c'est avoir les rôles) via un parcours de l'ontologie, puis on classe ces concepts en fonction du nombre de rôles qu'ils partagent avec `E(cod)`. Dans notre exemple, l'ensemble des rôles de `E(cod)` est :
`{hasAngleUnit, hasCcOriginEqRa, hasOneValue}`
Le parcours de l'ontologie donne les concepts de l'ontologie qui partagent des rôles avec `E(cod)` : (Les rôles partagés ont été écrits en italiques)

CONCEPT	ROLES
pos.angDistance	*hasAngleUnit*, *hasOneValue*
pos.az.alt	*hasAngleUnit*, *hasOneValue*, hasCcOriginAzAlt, hasFrameTypeAz
pos.eq.dec	*hasAngleUnit*, *hasOneValue*, hasCcOriginEqDec, hasFrameTypeEq
pos.eq.ra	*hasAngleUnit*, *hasOneValue*, *hasCcOriginEqRa*, hasFrameTypeEq
pos.eq.spd	*hasAngleUnit*, *hasOneValue*, hasCcOriginEqSpd, hasFrameTypeEq
pos.parallax	*hasAngleUnit*, *hasOneValue*, hasPlxMethod
...	...

Ces concepts sont ensuite classés selon le nombre de rôles de E(cod) partagés par les différents concepts. C'est ce tableau trié qui est renvoyé par le système :

CONCEPT	NOMBRE DE ROLES PARTAGES
pos.eq.ra	3
pos.angDistance	2
pos.eq.dec	2
pos.eq.spd	2
pos.az.alt	2
pos.parallax	2
...	...

Comme il existe un unique meilleur classé, c'est lui que le système désigne comme étant le *mot* `pos` correspondant à la ligne de `ReadMe`. Comme nous considérons des UCD composés d'un seul mot `pos`, ce mot `pos` est également l'UCD que nous cherchions à associer à la ligne de `ReadMe`. Dans notre exemple, le système associe à la ligne de `ReadMe` le mot `pos` : `pos.eq.ra`, autrement dit l'UCD `pos.eq.ra`.

S'il y a égalité au classement, on reprend le même processus, mais en exploitant cette fois le contenu de la colonne `Label`. Conservons l'exemple précédent pour expliciter cette deuxième passe :
Etape 1 :

- le système prend en entrée la ligne : `s RAs Right Ascension J2000 (seconds)`,
- le système rend en sortie : `Ext'(ReadMe) = {(Label, RAs)}`.

Etape 2 :

- le système prend en entrée : `Ext'(ReadMe) = {(Label, RAs)}`,
- le fichier de méta-données pour le `Label` est :
  ```
  Méta(Label) = {(RAs, AngleUnit|Value|CcOriginEqRa|FrameTypeEq),
                 (DEs, AngleUnit|Value|CcOriginEqRa|FrameTypeEq),
                 ... },
  ```
- le système rend en sortie :
  `E'(cod) = AngleUnit, Value, CcOriginEqRa, FrameTypeEq}`.

Etape 3 :

- le système prend en entrée l'ensemble des rôles de `E'(cod)` :
  `{hasAngleUnit, hasOneValue, hasCcOriginEqRa, hasFrameTypeEq}`,
- le système parcourt l'ontologie à la recherche de concepts possédant ces rôles :
  (Les rôles partagés ont été écrits en italiques)

CONCEPT	ROLES
pos.angDistance	*hasAngleUnit*, *hasOneValue*
pos.az.alt	*hasAngleUnit*, *hasOneValue*, hasCcOriginAzAlt, hasFrameTypeAz
pos.eq.dec	*hasAngleUnit*, *hasOneValue*, hasCcOriginEqDec, hasFrameTypeEq
pos.eq.ra	*hasAngleUnit*, *hasOneValue*, *hasCcOriginEqRa*, *hasFrameTypeEq*
pos.eq.spd	*hasAngleUnit*, *hasOneValue*, hasCcOriginEqSpd, hasFrameTypeEq
pos.parallax	*hasAngleUnit*, *hasOneValue*, hasPlxMethod
...	...

- le système renvoie en sortie le tableau de ces concepts, trié par nombre de rôles partagés :

CONCEPT	NOMBRE DE ROLES PARTAGES
pos.eq.ra	4
pos.eq.dec	3
pos.eq.spd	3
pos.angDistance	2
pos.az.alt	2
pos.parallax	2
...	...

Remarque : Nous avons souhaité séparer le travail sur la colonne `Label`.En effet, les labels sont porteurs de beaucoup d'information mais sont souvent ambigus.
Exemple : dans deux ***`ReadMe`*** *différents, le même label* ***`alpha`*** *désigne respectivement :*

- *Un angle de phase d'une orbite d'un corps céleste.*
- *Une ascension droite d'un corps céleste.*

Il est donc difficile pour l'expert d'associer des rôles à un label de manière sûre lors de la construction du fichier de méta-données `Méta(Label)`. De fait, une recherche de concept effectuée avec la colonne `Label` est moins sûre qu'une mesure faite qu'avec les colonnes `Units` et `Explanations`. Pour cette raison, nous avons préféré que le traitement de la colonne `Label` ne soit fait qu'en cas de nécessité, c'est à dire si la première passe débouche sur une égalité au premier rang.

Enfin, s'il y a toujours égalité au premier rang après la seconde passe, c'est à un expert de choisir quel est l'UCD correspondant à une ligne de `ReadMe` parmi les concepts à égalité au meilleur rang.

3.3 Bilan de la méthode

3.3.1 Évaluation de la méthode

Pour nos tests, nous avons utilisé des `ReadMe` des catalogues d'astronomie les plus employés et nous avons confronté nos résultats à des attributions d'UCD faites par un expert humain. Pour 75 lignes de `ReadMe` traitées, le bilan est le suivant :

- 4 lignes ont eu un UCD attribué dès la première passe
- 42 lignes supplémentaires ont eu un UCD attribué après la deuxième passe
- 26 lignes relèvent du choix d'un expert (égalité au premier rang du tableau) après la deuxième passe
- 3 lignes ont conduit à un échec (l'UCD attendu n'apparaît pas au premier rang du tableau de concepts renvoyé en fin de traitement)

3.3.2 Comparaison avec les outils existant

A l'heure actuelle, la seule approche exploitée au Centre de Données Astronomiques de Strasbourg (CDS) pour associer un UCD à une description est lexico-syntaxique. Cette approche s'est traduite par la création d'une application qui prend une description textuelle en entrée et utilise des règles lexicales et syntaxiques pour renvoyer en sortie une liste triée de propositions d'UCD correspondant à la description fournie en entrée. Cet outil donne de bons résultats, aussi bien sur des UCD composés d'un seul mot que de plusieurs. Par contre, une limitation majeure d'un

tel système est la dépendance de ses règles lexicales et syntaxiques aux évolutions des UCD et aux langues naturelles utilisées dans les descriptions textuelles : les règles doivent être réécrites pour tenir compte des changements.

Nous proposons une autre approche, pleinement compatible avec les outils existant, et qui rend le processus d'identification indépendant de la langue utilisée dans les `ReadMe` et des évolutions des UCD. En effet, dans le premier cas, il suffit de rajouter des entrées lexicales dans les fichiers de méta-données et dans le deuxième cas de s'assurer que l'ontologie est toujours adaptée et de la faire évoluer le cas échéant, sans avoir à changer la procédure d'association d'un concept à une description textuelle. Il reste néanmoins à prendre en compte les UCD composés de plusieurs mots avant de pouvoir faire une comparaison complète entre les performances de notre outil et celles du moteur d'assignation lexico-syntaxique utilisé au CDS.

Perspectives

Notre stratégie de traitement du cas des UCD composés d'un seul *mot* donnant des résultats encourageants, nos travaux futurs auront pour but la prise en compte des UCD composés de plusieurs mots.

Pour réaliser cela, nous comptons réutiliser le processus d'attribution de *mots* à une description et ajouter une étape supplémentaire de composition des mots obtenus. Par exemple, la ligne de `ReadMe` suivante correspond à l'UCD composé `stat.error;pos.eq.ra` :

```
Units  Label    Explanations
mas    e_RAdeg  ? Standard error in RA (H14)
```

Si on applique à cette ligne de `ReadMe` la procédure d'attribution des mots détaillée dans le chapitre 3, le système retourne :

CONCEPT	NOMBRE DE ROLES PARTAGES
pos.eq.ra	4
stat.error	3
pos.eq.dec	3
pos.angDistance	2
stat.error.sys	2
pos.parallax	2
...	...

On constate que les mots composant l'UCD composé sont parmi les meilleurs classés. L'idée est donc d'utiliser la procédure d'attribution des mots pour obtenir une liste des mots composant l'UCD composé. Ayant ces mots, il faut les composer pour obtenir l'UCD composé. Pour ce faire, nous procédons de la manière suivante :

- nous retrouvons les codes de compositions des mots les mieux classés du tableau dans la liste des mots valides
- utilisons la grammaire de composition des mots décrite en 1.3.2 pour obtenir des composés valide

Dans notre exemple, on obtient les codes suivants :

MOT	CODE DE COMPOSITION
pos.eq.ra	Q
stat.error	P
pos.eq.dec	Q
pos.angDistance	Q
stat.error.sys	P
pos.parallax	Q
...	...

D'où on compose ces mots à l'aide de la grammaire de composition des mots et l'on obtient

des UCD composés valides. Ces UCD sont renvoyés dans un tableau trié selon la somme des scores "rôles partagés" des mots composants ces UCD.
Exemple : `stat.error;pos.eq.ra` est composé de :

- *stat.error : rôles partagés = 3*
- *pos.eq.ra : rôles partagés = 4*

  d'où son score est 7

Le tableau trié renvoyé par le système est le suivant :

UCD	SOMME DES ROLES PARTAGES
stat.error ;pos.eq.ra	7
pos.eq.ra ;pos.eq.dec	7
pos.eq.dec ;pos.eq.ra	7
stat.error ;pos.eq.dec	6
stat.error ;pos.angDistance	5
stat.error.sys ;pos.eq.ra	5
...	...

On retrouve bien parmi les UCD de meilleur rang l'UCD composé `stat.error;pos.eq.ra` que nous espérions obtenir (puisque que c'est l'UCD attribué manuellement par un expert à la ligne de `ReadMe` que nous avons traité).

La stratégie que nous venons de présenter permet d'obtenir en sortie du système un tableau trié d'UCD composés valides correspondant à la ligne de `ReadMe` donnée en entrée. Il nous reste cependant un point à résoudre pour compléter cette stratégie : définir des conditions suffisantes à la recherche d'UCD composés de plusieurs mots pour pouvoir appliquer la bonne stratégie (UCD composé d'un mot ou de plusieurs mots) pour une ligne de `ReadMe` donnée. C'est ce point qui sera décisif dans nos futurs travaux.

Annexe A

Liste des mots pos

code	*mot*	*explication*
Q	pos	Position and coordinates
Q	pos.angDistance	Angular distance
Q	pos.az	Position in alt-azimuthal frame
Q	pos.az.alt	Alt-azimuthal altitude
Q	pos.az.ha	Alt-azimuthal hour-angle
Q	pos.az.zd	Alt-azimuthal zenith distance
S	pos.barycenter	Barycenter
S	pos.cartesian	Cartesian (rectangular) coordinates
Q	pos.cartesian.x	Cartesian coordinate along the x-axis
Q	pos.cartesian.y	Cartesian coordinate along the y-axis
Q	pos.cartesian.z	Cartesian coordinate along the z-axis
Q	pos.dirCos	Direction cosine
V	pos.distance	Linear distance
S	pos.earth	Coordinates related to Earth
Q	pos.earth.lat	Latitude on Earth
Q	pos.earth.lon	Longitude on Earth
Q	pos.earth.nutation	Earth nutation
S	pos.ecliptic	Ecliptic coordinates
Q	pos.ecliptic.lat	Ecliptic latitude
Q	pos.ecliptic.lon	Ecliptic longitude
S	pos.errorEllipse	Positional error ellipse
Q	pos.ephem	Ephemeris
S	pos.eq	Equatorial coordinates
Q	pos.eq.dec	Declination in equatorial coordinates
Q	pos.eq.ra	Right ascension in equatorial coordinates
Q	pos.eq.spd	South polar distance in equatorial coordinates
Q	pos.frame	Reference frame used for positions

code	*mot*	*explication*
S	pos.galactic	Galactic coordinates
Q	pos.galactic.lat	Latitude in galactic coordinates
Q	pos.galactic.lon	Longitude in galactic coordinates
Q	pos.frame	Reference frame used for positions
S	pos.galactic	Galactic coordinates
Q	pos.galactic.lat	Latitude in galactic coordinates
Q	pos.galactic.lon	Longitude in galactic coordinates
S	pos.galactocentric	Galactocentric coordinate system
S	pos.geocentric	Geocentric coordinate system
Q	pos.healpix	Hierarchical Equal Area IsoLatitude Pixelization
S	pos.heliocentric	Heliocentric position coordinate (solar system bodies)
Q	pos.htm	Hierarchical Triangular Mesh
S	pos.lambert	Lambert projection
Q	pos.lunar	Lunar coordinates
Q	pos.lunar.occult	Occultation by lunar limb
Q	pos.parallax	Parallax
Q	pos.parallax.dyn	Dynamical parallax
Q	pos.parallax.phot	Photometric parallaxes
Q	pos.parallax.spect	Spectroscopic parallax
Q	pos.parallax.trig	Trigonometric parallax
V	pos.pm	Proper motion
Q	pos.posAng	Position angle of a given vector
V	pos.precess	Precession
Q	pos.satellite	Position/coordinates of satellite or planet
S	pos.supergalactic	Supergalactic coordinates
Q	pos.supergalactic.lat	Latitude in supergalactic coordinates
Q	pos.supergalactic.lon	Longitude in supergalactic coordinates
P	pos.wcs	WCS keywords
P	pos.wcs.cdmatrix	WCS CDMATRIX
P	pos.wcs.crpix	WCS CRPIX
P	pos.wcs.crval	WCS CRVAL
P	pos.wcs.ctype	WCS CTYPE
P	pos.wcs.naxes	WCS NAXES
P	pos.wcs.naxis	WCS NAXIS
P	pos.wcs.scale	WCS scale or scale of an image

Bibliographie

[Alhulou, 2003] R. Alhulou *Les logiques de descriptions pour le traitement intelligent de données textuelles dans le projet Escrire.* Thèse INRIA, 2003

[Baader, 2003] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, W. Nutt, and P.F. Patel-Schneider *The Description Logic Handbook : Theory, Implementation, and Applications.* Cambridge University Press, 2003.

[Derriere et al., 2004] S. Derriere, N. Gray, R. Mann, A. Preite Martinez, J. McDowell, T. McGlynn, F. Oschenbein, P. Osuna, G. Rixon and R. Williams UCD *(United Content Descriptor) - moving to* UCD1+ *Version 1.06 IVOA Proposed Recommendation 2004-10-26.* `http://www.ivoa.net/documents/latest/ucd.html/`.

[Derriere and Preite Martinez, 2004] S. Derriere, A. Preite Martinez *The* UCD1+ *controlled vocabulary Version 1.06 IVOA Working Draft 2004-08-23.* `http://www.ivoa.net/documents/latest/ucdlist.html/`.

[Fensel et al., 2003] D. Fensel, J. Hendler, H. Lieberman, W. Wahlster *Spinning the semantic web.* The MIT Press, Massachussets Institute of Technology, 2003.

[Gruber, 1993] T. R. Gruber *Toward Principles for the Design of Ontologies Used for Knowledge Sharing.* Formal Ontology in Conceptual Analysis and Knowledge Representation, Kluwer Academic Publishers, Deventer, The Netherlands, 1993

[Horridge et al., 2004] M. Horridge1, H. Knublauch, A. Rector, R. Stevens, C. Wroe *A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools Edition 1.0.* University Of Manchester, 2004

[Napoli, 1997] A. Napoli *Une introduction aux logiques de descriptions.* Rapport de recherche RR 3314, INRIA, 1997.

[Napoli, 2004] A. Napoli *Elements on knowledge representation, description logics, ontologies and knowledge discovery for the semantic web.* In : Summer School on Semantic Web and Ontologies, Aussois, June 23, 2004.

[Noy and McGuiness, 2000] N.F. Noy and D.L. McGuiness *Ontology Development 101 : A Guide to Creating Your First Ontology.* `http://protege.stanford.edu/publications/ontology_development/ontology101.html/`, also available as SMI Technical Report SMI-2001-0880 and KSL Technical Report KSL-01-05.

[Ochsenbein, 2000] F.Ochsenbein *Astronomical Catalogues and Tables Adopted Standards version 2.0.* `http://vizier.u-strasbg.fr/doc/catstd.htx/`.

[Staab and Studer, 2004] S. Staab and R. Studer *Handbook on Ontologies.* Springer, Berlin, 2004.

[Uschold and King, 1995] M. Uschold and M. King *Towards a Methodology for Building Ontologies.* Uschold M. Towards a Methodology for Building Ontologies Workshop on Basic Ontological Issues in Knowledge Sharing, held in conduction with IJCAI-95, 1995.

[Uschold and Gruninger, 1996] M. Uschold and M. Gruninger *Ontologies : Principles, Methods and Applications*. Knowledge Engineering Review, volume 11, number 2, pages 93-155, 1996.