TABLE DES MATIERES

A - Généralités: situation du problème 1

1. Généricité et souplesse 3

2. Contraintes d'adéquation du modèle 6

3. Axes de description 7

3.1. Axe componentiel ou analytique. 7

3.2. Axe relationnel ou différentiel 7

B - Architecture globale du lexique : les trois couches 9

1. Unité sémantique - Unité sémantique d'affixe 10

2. Articulation avec les autres couches 11

3. Unité lexicale 14

C - le modèle de la couche sémantique 17

1. Vue synthétique du modèle de la couche sémantique 18

2. Unité sémantique 19

3. Traits sémantiques valués 20

3.1. Traits de type général (propriété) 21

3.2. Traits de type classe sémantique 22

3.3. Traits de type distinctif 24

3.4. Traits de type domaine 25

3.5. Traits de type niveau de langue 26

3.6. Traits de type connotation et évaluation 26

3.7. Traits de type pragmatique 27

3.8. Traits de type divers 27

4. Relations sémantiques entre Unités Sémantiques 29

4.1. Relations de type paradigmatique entre unités substituables 30

4.2. Relations sémantiques de type dérivation 31

4.3. Relations sémantiques de type collocation 33

5. Prédicat 36

5.1. Mise en évidence de la notion de prédicat 36

5.2. Description 37

5.3. Conséquences sur les points du modèle abordés précédemment 38

5.3.1. Au niveau componentiel 38

5.3.2. Au niveau des relations sémantiques 39

5.3.3. Au niveau du codage et de l'établissement de critères de cohérence 45

6. Concept 46

7. Le niveau de description "conceptuel" 48

7.1. Prédicat 48

7.2. Relations au niveau conceptuel 49

7.2.1. Relation entre concepts 49

7.2.2. Relation entre prédicats 50

7.2.3. Relation entre prédicats et concepts 50

7.3. Apport de ce niveau de représentation 51

7.4. ƒlaboration de ce niveau 52

D - Correspondance syntaxe sémantique 53

1. La notion de prédicat : rappel 54

2. Position syntaxique/Argument prédicatif 55

3. Contraintes sur la description de base 57

3. 1 Portée du filtrage 57

3. 2 Filtrage par des traits 59

3. 2. 1 Filtrage par des traits de la syntaxe 59

3. 2. 2 Filtrage par des traits sémantiques 60

4. Réalisation des arguments 62

4. 1. Correspondance argument position 62

4. 2. Argument flottant 63

4.3 Valeurs par défaut 64

5. Gestion de l'optionnalité 69

6. Quelques cas plus complexes 71

6.1. Unité sémantique contenant, pour l'un de ses arguments, un prédicat et un (ou plusieurs) de ses arguments. 71

6.2. Le cas des Unités syntaxiques composées 76

6.2.1. Intérieur/extérieur d'un composé syntaxique 76

6.2.2. Insertion de modifieur 78

7. Mécanismes mis en Ïuvre 83

E - BIBLIOGRAPHIE DE Référence 85

F - MANUEL DE L'UTILISATEUR 87

1. Généralités 88

2. Usem 89

2.1. Généralités 89

2.2. Représentation prédicative (RepresentationPredicative) 91

2.3. Représentation conceptuelle pondérée (RepresentationConceptuelle_Pond) 91

3. Unité sémantique d'affixe (Usem_Aff) 93

4. Prédicat 94

4.1 Généralités 94

4.2. Argument 96

4.3. SelectEtPreciseArg, InformeArg 97

4.4. Prédicat instancié (PredInstancie) 99

4.5. Liste de prédicats (ListePred) 100

4.5.1. Variable 101

4.5.2. SelectPredArg 101

5. Concept 102

6. Trait sémantique valué 104

6.1. Trait sémantique valué pondéré (Trait_Sem_ValPond) 104

6.2. Généralités 104

6.3. Trait sémantique (Trait_Sem) 106

7. Relation valuée pondérée 108

7.1. Généralités 108

7.2. Relation sémantique 109

7.3. Relation sémantique entre Usem (R_Usem) 109

7.4. Relation sémantique entre les autres objets (R_Pred, R_Concept, R_Pred_Concept, R_Concept_Pred) 110

7.5. Correspondance d'arguments (Corresp_Arg_Arg) 110

8. Correspondance syntaxe sémantique (Corresp_Usyn_Usem) 111

8.1. Correspondance 111

8.2. Correspondance Argument-Position simple et flottante (Corresp_Arg_Pos_Simple Corresp_Arg_Pos_Flottant) 112

8.3. ContraintDescription 113

8.3.1. ContraintIntervConst 113

8.3.2. ContraintConstruction 113

8.3.3. ContraintStructInterne 114

8.3.4. Contraint_mdc 114

8.3.5. ContraintPosition 114

8.3.6. ContraintSyntagme 115

G - Schémas entitéS- relation 117

1. Unité sémantique, Représentation prédicative, Représentation conceptuelle 118

2. Usem d'affixe (Usem_Aff) 119

3. Prédicat, Liste prédicats, Argument, Variable, Rôle sémantique 120

4. Concept 121

5. Trait valué pondéré, Trait valué, Valeur_trait, Trait sémantique, 122

6. Relations valuées pondérées(-pred, -concept, -Usem) 123

7. Relations valuées pondérées (-pred-concept, -concept-pred) 124

8. Relations sémantiques (Usem-Usem, pred-pred, concept-concept, pred-concept, concept-pred) 125

9. SelectEtPréciseArg, Informe argument, Prédicat instancié 126

10. Correspondance syntaxe sémantique, 127

11. Contraint description, Contraint IntervConst, Contraint mdc, Contraint construction, Contraint struct interne, Contraint position, Contraint syntagme 128

H- DTD SGML 129

I - Introduction - Traduction du Modèle Conceptuel 130

II - DTD GENELEX commentée 131

1. DTD genelex.dtd 131

2. DTD semant.dtd 134

3. Contraintes semant.ctr 157

4. Entités semant.ent 161

5. Entités custom.ent 163

6. DTD syntaxe.dtd 164

7. Contraintes syntaxe.ctr 183

8. Entités syntaxe.ent 187

9. DTD morpho.dtd 189

10. Contraintes morpho.ctr 195

11. Entités morpho.ent 196

A - Généralités : situation du problème

La sémantique lexicale est encore relativement jeune et peu formalisée, contrairement à la morphologie et même la syntaxe. Celles-ci prêtent relativement moins à discussions et polémiques, et on peut même voir se dessiner des grandes tendances et des notions partagées par différentes approches théoriques, ce qui facilite la définition d'un modèle générique. Il n'existe pas à l'heure actuelle de consensus concernant la sémantique lexicale.

Nous aborderons donc le sujet avec beaucoup de prudence et de modestie et nous limiterons nos ambitions *unificatrices* concernant le modèle sémantique.

Dans l'élaboration du modèle, il nous faut tenir compte des expériences et du savoir-faire existants en matière de sémantique lexicale, tant dans la tradition lexicographique dictionnairique que dans les travaux actuels en linguistique computationnelle.

D'une part, il existe une tradition de lexicographie (autour des dictionnaires papier) dont nous devons savoir tirer parti bien qu'elle ne soit pas soumise aux mêmes impératifs que ceux qui régissent la construction d'un dictionnaire électronique.

En effet, la description des entrées lexicales des dictionnaires papier se fait, pour un dictionnaire donné, en répondant à des normes ou en suivant des recommandations de codage qui doivent permettre d'éviter de trop grands écarts de codage dans un même dictionnaire et des incohérences. Cependant, il est toujours fait appel à l'évaluation du rédacteur, évaluation partiellement subjective, et le lecteur humain s'accommode plutôt bien de ces caractéristiques. Par exemple, la circularité qui est le propre de ces dictionnaires, est sans doute acceptable pour un utilisateur humain, car elle reflète en partie notre perception du sens des mots. Par ailleurs, le lecteur remédie aux imprécisions, aux ambiguïtés, au présupposé et à l'implicite des définitions en faisant largement appel à ses connaissances générales, linguistiques et extra-linguistiques.

Une utilisation par un système informatique des informations issues d'un dictionnaire électronique (au modèle GENELEX ou non) ne peut cependant pas aussi bien s'en accommoder. Une structuration systématique et cohérente des informations ainsi qu'un ensemble d'outils descriptifs complémentaires semblent nécessaires à un modèle de description de la sémantique lexicale en vue de la réalisation de dictionnaires électroniques. Les informations issues d'un tel dictionnaire doivent en effet pouvoir être utilisées par des applications de traitement automatique de la langue, sans préjuger lesquelles : il s'agira donc de répondre à des besoins très différents.

D'autre part, un certain nombre de recherches dans le domaine de la sémantique lexicale ont été menées, mais peu de réalisations génériques en grandeur réelle et à visée industrielle ont pu voir le jour jusqu'à présent, mis à part les projets EDR et CYC. Il faut aussi citer ici les travaux de I. Mel'Cuk et son équipe [Mel'cuk 1984 -1988].

En effet, les travaux en sémantique lexicale les plus soignés portent le plus souvent sur un sujet particulier (mouvement, temporalité, aspect, connotations...) mais leur modélisation demeure généralement dispersée. Le cadre global pouvant accueillir ces modélisations partielles, indispensable pour un modèle générique, reste à définir.

Dans un premier temps, nous préciserons l'approche qui est la nôtre, les objectifs que nous nous sommes fixés pour le modèle de la couche sémantique. Nous présenterons ensuite l'architecture globale du lexique avant d'aborder plus précisément le modèle de la couche sémantique.

1. Généricité et souplesse

Comme pour la morphologie et la syntaxe, l'approche de la sémantique dans le cadre de GENELEX doit garantir la généricité du format, afin d'assurer la réutilisabilité d'un fonds dictionnairique au modèle GENELEX. Par généricité, il faut entendre le fait de permettre à différents systèmes de traitement automatique de la langue, s'insérant dans des cadres théoriques différents, et visant à des applications différentes, d'extraire les informations lexicales pertinentes, et également de permettre à des utilisateurs humains d'exprimer ou de consulter de façon agréable les informations lexicales.

Cela signifie donc que le modèle doit à la fois :

- pouvoir rendre compte, sans trop de distorsions, des différentes théories ou approches (étant donné un certain état de l'art à ce jour), sans être lui-même lié à une théorie particulière.

- pouvoir évoluer aisément, par enrichissement et sans remise en question globale (cela suppose de fixer un cadre souple pouvant supporter les enrichissements ultérieurs).

La couche sémantique du modèle GENELEX sert à représenter les informations sémantiques propres au lexique. Il s'agit donc de sémantique lexicale générale. Le modèle du lexique ne vise pas d'applications particulières et n'oriente pas la description de celui-ci vers un domaine particulier. Il doit pouvoir être utilisé dans le cadre d'applications concrètes et la limite de ce qu'il faut prévoir de décrire se trouve là où commenceront les informations dites liées à une application, informations que l'on pourrait qualifier de pragmatiques du domaine pour une part et encyclopédiques pour une autre part, qu'il s'agisse de connaissances générales ou liées aux besoins de l'application.

Cependant, les limites entre ces catégories d'informations sont relativement floues, et on peut considérer qu'il y a une certaine continuité, et même que la sémantique lexicale gagne à pouvoir inclure quelques informations qui semblent être plus des connaissances générales sur le monde que des informations purement linguistiques. C'est pourquoi chaque instanciation du modèle devra définir les limites de ce que le lexique GENELEX doit contenir.

C'est ainsi que doivent pouvoir être représentés différents niveaux d'informations, qui peuvent sembler incompatibles, et qui seront nécessaires pour des approches et des familles d'applications différentes.

Il s'agit de :

Un niveau de **représentation sémantique linguistique**.

Cette représentation, très proche de la langue, est construite principalement à partir de l'observation du lexique en contexte (dans des énoncés réellement produits en situation et non construits à cet effet) et des relations sémantiques entre éléments du lexique (on trouvera ici les informations sémantiques fines que l'on souhaite trouver dans un dictionnaire papier dérivé d'un dictionnaire GENELEX et qui sont nécessaires à la traduction automatique de qualité ou à la génération, à la compréhension automatique de texte pour génération de résumé...).

Un niveau de **représentation sémantique conceptuelle.**

Cette représentation, issue au moins pour une part des courants de l'intelligence artificielle et de certaines autres réflexions, est d'une plus grande "abstraction". Elle s'appuiera sur des primitives, associées à un formalisme de représentation des connaissances.

Sur ce niveau pourront se brancher des informations extra-linguistiques concernant la représentation du monde.

Les connaissances mises en jeu concerneront donc bien souvent non seulement le contenu intensionnel proprement linguistique mais aussi en partie les propriétés de l'objet dénoté, et elles pourront comporter une description partielle d'un monde donné.

Ce niveau de représentation sera probablement associé en amont (hors modèle GENELEX) à un système permettant de raisonner sur ces connaissances.

Il est souhaitable de se donner les moyens de faire cohabiter ces deux niveaux et de préciser l'**articulation** qui permettra à la fois de rendre compte du contenu sémantique proprement linguistique et de l'interfacer avec le niveau conceptuel. C'est un des choix du modèle sémantique GENELEX que d'offrir la possibilité d'articuler ces deux niveaux.

La très grande diversité des approches et des théories qu'il s'agit de pouvoir accueillir nous oblige par ailleurs à adopter un modèle très riche offrant ainsi la possibilité d'exprimer une théorie donnée dans un sous-ensemble du modèle.

L’approche choisie doit donc être **plus multi-théorique que a-théorique**, tout en restant cohérente. Elle fournira par conséquent un **cadre descriptif très large** et permettant parfois d’exprimer des informations voisines de différentes façons.

Cette grande souplesse et cette ouverture présentent, il est vrai, un certain risque de redondance et même d’incohérence dont le lexicographe devra être conscient. Comme aux niveaux morphologique et syntaxique, il lui faudra faire certains choix globaux et s’y tenir, et le logiciel gestionnaire de la base lexicographique GENELEX devra permettre, en fonction du sous-ensemble du modèle utilisé, de vérifier certaines contraintes d'intégrité.

Dans ce cadre, il est clair que les choix globaux de codage et de structuration de l'information constituent un certain "paramétrage" du modèle, et que ce paramétrage devra être compatible entre deux dictionnaires que l'on souhaite faire communiquer ; la mise en correspondance des instanciations du modèle est donc un prérequis à la fusion.

2. Contraintes d'adéquation du modèle

GENELEX est un projet à visées avant tout industrielles, et le modèle devra supporter un grand nombre et une grande variété de données, de degrés de finesse très variables. Il ne devra rebuter ni les éditeurs de dictionnaires ni les lexicographes "traditionnels" ou issus de la "linguistique computationnelle" et des grands courants de l'intelligence artificielle, ni les utilisateurs.

Un modèle formellement trop complexe laisserait nombre de lexicographes perplexes et risquerait d’être malaisément utilisable.

Il est souhaitable également d'avoir des degrés variables de finesse d’expression ou tout du moins de choisir une représentation du niveau sémantique permettant une extraction aisée de dictionnaires plus ou moins précis au niveau sémantique.

La modélisation de la sémantique s’appuie sur les couches morphologique et syntaxique et doit tenir compte des informations de ces niveaux. L’approche doit donc être cohérente et en particulier décrire les correspondances entre informations des différents niveaux, par exemple préciser la réalisation des arguments sémantiques par rapport aux informations du niveau syntaxique.

Bien qu’il ne soit pas souhaitable de privilégier une théorie, il est sans doute nécessaire de proposer un minimum de méthodologie pour décrire les entrées sémantiques : définir des critères d’éclatement d’entrées et de structuration des entrées "polysémiques" éventuellement.

Il faut noter qu’il est important de limiter la quantité d’informations en factorisant ce qui peut l’être et en donnant les moyens de calculer certaines informations provenant de ces factorisations. Il faudra donc utiliser des mécanismes d’héritage ou de partage entre différents niveaux de représentations et entre unités liées par certaines relations.

3. Axes de description

Une unité sémantique représente l'unité de sens (acception et son explicitation) associée à une ou plusieurs unités syntaxiques issues de la même unité morphologique lorsqu'il s'agit d'Usyn simples, ou bien l’unité de sens associée à une Usyn composée. Les informations qu'elle comporte doivent permettre de conna"tre le comportement de cette unité du point de vue du sens : décrire son apport

sémantique propre dans son interaction avec les éléments avec lesquels elle est combinée conformément à la syntaxe. Les règles générales de composition du sens afin de construire la représentation sémantique des énoncés et même des textes sont bien entendu supposées définies à l'extérieur du dictionnaire GENELEX.

La description d'une unité sémantique peut se faire principalement suivant deux grands axes complémentaires :

3.1. Axe componentiel ou analytique.

Cet axe consiste à décrire "de l'intérieur" l'unité sémantique en la "décomposant" en informations sémantiques élémentaires (par des traits et des références à des classes ou à des prédicats de base, par exemple) ; les informations associées à cet axe sont de l'ordre de la définition par composants de sens.

Elles permettent aussi de rattacher le sens de l'unité à des unités du niveau abstrait conceptuel tout en indiquant le contenu spécifique.

Cet axe se prête bien à la description conceptuelle avec plusieurs niveaux d'abstraction ou de généralisation possibles (éventuellement jusqu'aux primitives utilisées en Intelligence Artificielle).

3.2. Axe relationnel ou différentiel

Celui-ci consiste à situer les unités sémantiques les unes par rapport aux autres en précisant la nature des relations de sens qu'elles entretiennent.

Cet axe rend compte des relations de proximité de sens entre unités et permet aussi de préciser les relations de dérivation sémantique et de collocation ou préférence.

On pourra exprimer des relations entre unités sémantiques (donc se situant au niveau linguistique) mais aussi des relations entre unités du niveau abstrait conceptuel vers lesquelles les unités sémantiques pointent.

Par ailleurs, il n'est pas possible de décrire de façon isolée les sens des entrées lexicales, sans décrire également les interactions de sens particulières au contexte d'occurrence de l'entrée décrite, et il faut donc prendre aussi en considération les répercussions de l'organisation syntagmatique sur le niveau sémantique. Autrement dit, il nous faut décrire aussi comment se fait le passage entre le niveau syntaxique (contexte d'occurrence de l'entrée décrite) et le niveau sémantique et les répercussions que le contexte syntaxique de cette correspondance et son instanciation peuvent avoir sur la représentation sémantique de l'entrée.

Les entrées dites prédicatives devront comporter un certain nombre d'informations particulières concernant les arguments et leurs correspondances, non seulement dans la description du passage syntaxe sémantique, mais aussi à tous les niveaux du modèle sémantique mettant en jeu des relations sémantiques entre entrées prédicatives.

Certaines informations nécessaires à la description de l'unité sémantique dépendent en partie de la catégorie de l'unité morphologique dont elle est issue. Par exemple, il est classique de distinguer les

noms qui correspondent à des entités, des verbes qui décrivent un état, un procès ou une transition, des adjectifs ou adverbes qui modifient le sens des éléments des catégories précédentes en précisant certains de leurs attributs...

Cette distinction n'est que partiellement pertinente. C'est pourquoi le modèle GENELEX ne figera en rien le type d'informations requises ou autorisées en fonction de la catégorie, et il est clair par exemple que les unités lexicales de catégorie verbe ne sont pas les seules à avoir un contenu qui peut être décrit de façon prédicative.

D'autre part, le modèle n'**exclut a priori aucune catégorie ou sous-catégorie grammaticale** de la description sémantique : un déterminant ou une préposition pourront parfaitement être décrits en sémantique.

B - Architecture globale du lexique : les trois couches

1. Unité sémantique - Unité sémantique d’affixe

Une **unité sémantique** concentre l’ensemble des informations sémantiques correspondant à une acception d’une entrée du dictionnaire. Elle porte ces informations elle-même, ou permet d’y accéder via d’autres objets qui lui sont associés. On peut distinguer les unités sémantiques représentant le sens d’unités autonomes, que l’on désigne par **Usem**, des unités sémantiques associées aux affixes que l’on désigne par **Usem_Aff**

Une unité sémantique autonome (**Usem**) décrit le sens d’une unité morphologique dans un contexte syntaxique donné : elle est dans ce cas le correspondant en sémantique d’une Usyn simple. Elle peut aussi être le correspondant en sémantique d’une Usyn composée considérée comme un tout par la sémantique, même si elle fait intervenir des composants (Um ou Usyn) dans la description syntaxique.

Une Usem ne prétend pas donner complètement le sens de la structure syntaxique. La grammaire apportera pour sa part d'autres informations sémantiques et en particulier sur le type de compositionnalité, et ces informations permettront un certain calcul d'une représentation sémantique en combinant les informations sémantiques associées à chacun des éléments de la structure et celles issues de la grammaire.

Une **unité sémantique d’affixe** concentre sur elle-même ou permet d'accéder à l’ensemble des informations sémantiques correspondant à l’apport sémantique particulier d’un affixe dans la formation d’un mot, le mode de formation étant défini dans la couche morphologique. Elle est directement associée à l’Um affixe dont elle décrit minimalement le sens ou l’un des sens.

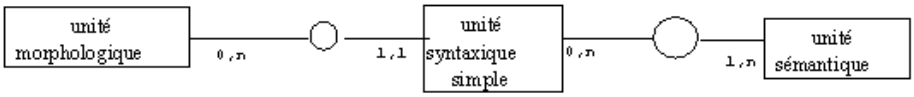
2. Articulation avec les autres couches

On accède à une unité sémantique par une ou plusieurs unités syntaxiques, lesquelles sont associées à une même unité morphologique dont l’unité sémantique représente une acception (cas des Usyn simple). Une unité sémantique est donc associée indirectement mais de façon non ambigu‘ à une et une seule Um dans le cas d’Usyn simple L'unité sémantique décrit le sens associé à *SELF* dans la

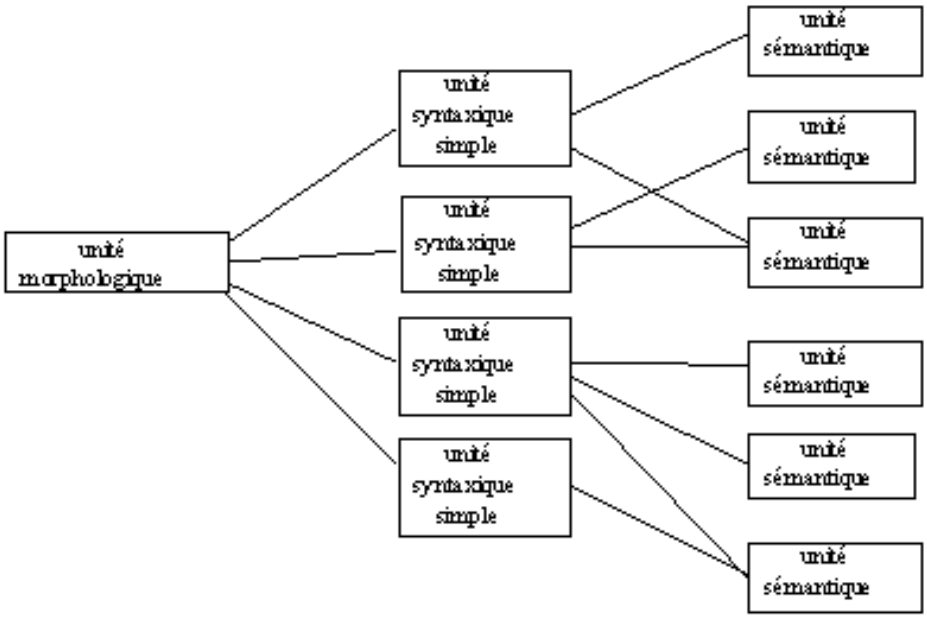
Description de Base (DB) caractéristique de l'unité syntaxique. Lorsque l'unité sémantique est décrite comme prédicative, elle peut apporter également des connaissances sur les arguments du prédicat et sur leur origine de surface.

D'autre part, une unité syntaxique (et implicitement l'unité morphologique dont elle est issue) peut être associée à plusieurs unités sémantiques, c'est-à-dire qu'une même construction syntaxique peut correspondre à un ou plusieurs sens.

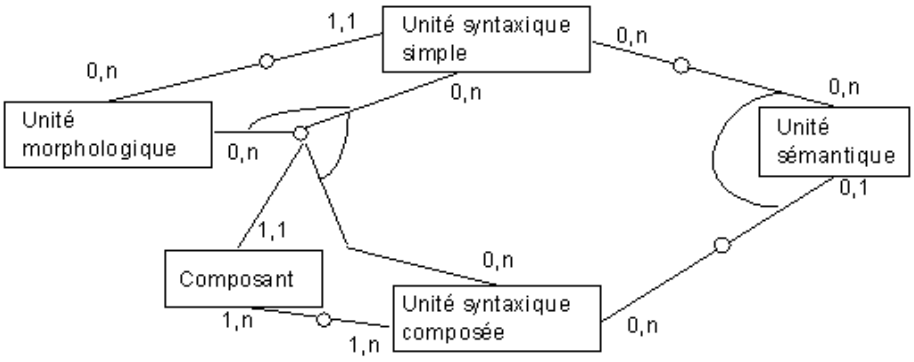
Cela peut s'exprimer par le modèle entité-relation suivant :



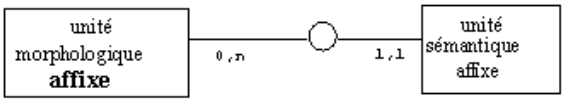
dont une instanciation possible, pour une unité morphologique donnée, serait :



En incluant les Unités syntaxiques composées ,ce modèle devient :



Pour la sémantique des affixes on a :



Il est clair que, tout en restant dans le cadre du modèle GENELEX, le lexicographe effectuera différents choix concernant le codage du niveau syntaxique et que ces choix auront des

répercussions sur la structure générale du dictionnaire.

En effet, de nombreuses factorisations dans la description des occupants de positions, un usage intensif des optionnalités dans les DB et la non utilisation de conditions dénotationnelles mèneront à une instanciation de dictionnaire GENELEX comportant généralement peu d'unités syntaxiques pour une unité morphologique, et beaucoup d'unités sémantiques pour une unité syntaxique, donc la nécessité de préciser beaucoup de filtrages sur la DB au moment de mettre en correspondance une unité syntaxique avec une unité sémantique.

Plus on fait de regroupements en syntaxe, plus on devra faire de filtrage sur la syntaxe en sémantique. Inversement, moins on fait de regroupements en syntaxe, moins on doit préciser de filtrage sur la syntaxe depuis la sémantique. Une utilisation du modèle GENELEX devra donc se situer entre ces deux pôles : un découpage en peu d'unités syntaxiques, très "factorisant" et des informations de filtrage nombreuses, ou un découpage en de plus nombreuses unités syntaxiques, mais ne nécessitant que très peu (voire pas) d'informations de filtrage.

A l'inverse, le fait d'éclater les unités syntaxiques en ne factorisant que peu les informations et en utilisant des contraintes dénotationnelles et des rôles thématiques permettra de distinguer dès le niveau syntaxique des différences de construction qui trouveront bien souvent un parallèle au niveau sémantique dans le fait qu'elles correspondent à des unités sémantiques (et donc à des acceptions) différentes.

Une unité sémantique peut être associée à différents contextes syntaxiques décrits dans plusieurs Usyn.

Dans le cas général, un dictionnaire au format GENELEX comportera un grand nombre d'unités sémantiques, plus grand que le nombre d'unités morphologiques, même si certaines unités sémantiques ont en commun de nombreuses informations (elles peuvent partager des traits, un prédicat, pointer vers le même concept...).

3. Unité lexicale

Une **unité lexicale simple** complètement renseignée est donc représentée **virtuellement** dans le modèle GENELEX, par l'association d'une unité morphologique, une unité syntaxique, et une unité sémantique. Il s'agit donc d'un **cheminement particulier dans les différentes couches**, avec accumulation des informations associées à chacune des couches. Il est clair que ce cheminement peut être incomplet si aucune information sémantique n'a été considérée nécessaire, auquel cas l'unité lexicale sera réduite au couple unité morphologique-unité syntaxique. Certaines unités peuvent même ne contenir que des informations morphologiques, dans ce cas l'unité lexicale est entièrement renseignée par l'unité morphologique.

S'il s'agit d'**une unité lexicale "composée"**, elle peut être décrite à différents niveaux :

- dans la couche morphologique. Dans ce cas, elle est décrite comme Um composée, s'appuyant sur les informations morphologiques de ses composants ; elle est au niveau syntaxique et sémantique décrite de la même façon que les Um simples : il s'agit donc ici aussi d'un cheminement dans les trois couches.

- dans la couche syntaxique. Elle est alors décrite par une Usyn composée, s'appuyant pour sa description sur des Usyn et/ou Um composantes. Les informations

morphologiques la concernant proviennent donc de celles portées par ses unités composantes. Cette Usyn est associée à des Usem suivant le même principe que pour les Usyn simples. Il s'agit ici d'un cheminement particulier de la couche syntaxique (pointant vers la couche morphologique) et sémantique.

- dans la couche sémantique. Cette "composition" sémantique est décrite par le fait que certaines Usem sont en relation de collocation. Les Usem en relation de collocation ont chacune leur propre cheminement dans les trois couches.

Les **Unités morphologiques affixes** sont traitées à part. En effet, on ne leur associe pas de comportement syntaxique en tant que tel, leur comportement combinatoire au niveau du mot étant défini éventuellement dans la couche morphologique. Le modèle leur donne cependant une sémantique en leur associant directement une ou plusieurs **Unités sémantiques d'affixes.** Ici il s'agit d'un cheminement de la couche morphologique à la couche sémantique, sans parcours de la couche syntaxique.

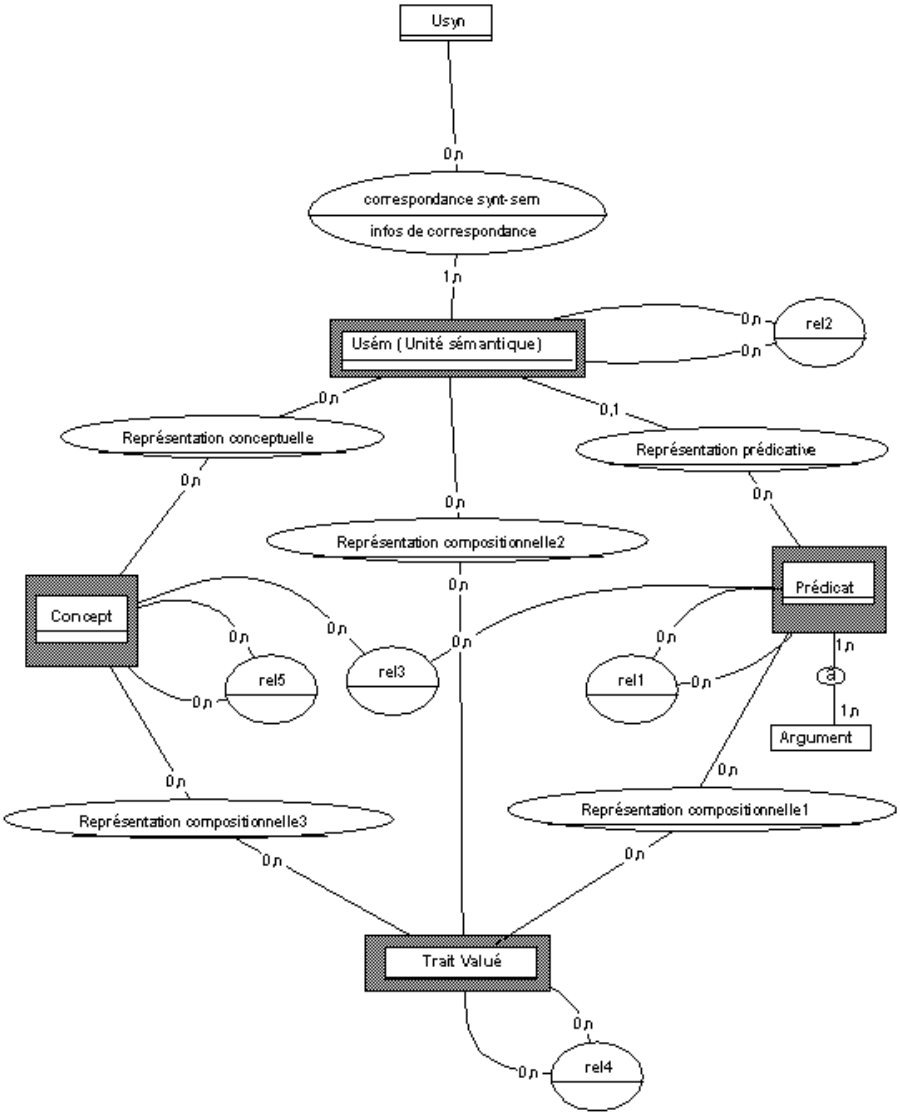
Nous allons maintenant entrer dans le détail du modèle de la couche sémantique. Ce qui concerne la correspondance entre les couches syntaxique et sémantique sera ensuite explicité.

C - le modèle de la couche sémantique

Nous allons dans les points qui suivent aborder les principales notions identifiées dans le modèle. La modélisation de la couche sémantique doit être vue suivant les deux axes complémentaires de l'analyse componentielle et relationnelle. On vise aussi à rendre compte, en les faisant s'articuler, des deux points de vue que sont la sémantique lexicale (linguistique) et la représentation conceptuelle de connaissances (orientées IA). On s'intéressera dans un premier temps aux informations ne concernant pas l'aspect prédicatif des entrées, celles-ci étant abordées par la suite.

La plupart des exemples donnés dans ce rapport concerne la sémantique des classes dites *ouvertes* (qui correspondent en morphologie à des entrées de catégories grammaticales *nom, verbe, adjectif, adverbe* en opposition aux classes dites *fermées* (qui correspondent en morphologie à des entrées de catégories grammaticales *déterminant, pronom, préposition, conjonction)* . Cela ne signifie pas que ces éléments ne peuvent avoir de description sémantique au modèle GENELEX. Il en va tout autrement, et tant les déterminants que les prépositions trouveront tout à fait leur place dans la couche sémantique d'un dictionnaire au modèle GENELEX.

1. Vue synthétique du modèle de la couche sémantique



2. Unité sémantique

L’unité sémantique (Usem) est le point d’entrée de la couche sémantique, comme l’Usyn est le point d’entrée de la couche syntaxique, et l’Um celui de la couche morphologique.

L’Usem décrit un sens, une acception, d’une Usyn, que celle-ci soit simple ou composée, et si elle est simple, qu’elle soit associée à une Um simple ou composée. Dans la mesure où l’Usem est associée à une ou plusieurs Usyn, le sens est décrit en étant associé à un ou des contextes syntaxiques.

L’Usem s’appuie pour sa description sur différents objets : Prédicat et Argument, Concept, et Trait sémantique valué. De plus, elle porte elle-même certains attributs qui complètent sa caractérisation. Ce sont les attributs permettant de lui donner une définition libre, celle du lexicographe ou du dictionnaire papier, et aussi une définition dite "formelle". Ses caractéristiques d’usage sont également précisées par une combinaison de valeurs d’emploi (CombVE) comme en syntaxe et en morphologie.

3. Traits sémantiques valués

Un trait sémantique valué est l'association d'un trait sémantique nommé et d'une de ses valeurs, un trait sémantique valué est donc en quelque sorte un couple attribut-valeur. On verra plus loin que que les traits sémantiques sont de différents types, qu'ils peuvent être binaires (à valeur +/-) ou posséder une liste de valeurs, ouverte ou fermée, et que le modèle offre la possibilité d'exprimer un certain nombre de connaissances sur ces traits sémantiques.

Une partie importante des informations sémantiques peut s'exprimer de façon componentielle par l'intermédiaire de traits sémantiques valués. C'est un moyen de décomposition assez souple et certainement très puissant. L'approche de F.Rastier [Rastier 1987], qui tente d'unifier les représentations des différents niveaux (mot, phrase, texte) nous semble très intéressante comme base de réflexion sur l'organisation du lexique.

Cependant, il ne faut pas oublier que le principal problème auquel on se heurte lorsqu'on entreprend la description du lexique selon cette méthode est :

où s'arrête la décomposition ?

En effet, une description fine du sens des mots associée à l'utilisation d'une telle approche en grandeur réelle sur tout le lexique dit général (et non dans un petit domaine bien cerné) mène à l'identification de traits toujours plus nombreux dont la généralité dans la langue et même la pertinence sont parfois discutables.

En multipliant les traits, on en arrive à se définir une sorte de métalangage de traits, riche et presque inexploitable automatiquement, auxquels on associe éventuellement une paraphrase, qui peut être complexe, indiquant le sens du trait.

Nous ne souhaitons pas préconiser une telle approche pour GENELEX, même si l'utilisation de traits présentant une certaine récurrence dans le lexique général nous semble intéressante et même nécessaire.

La détermination d'un ensemble minimal de traits utilisés pour la description *générique* du lexique complet d'une langue demeure par ailleurs un problème dont la solution ne peut être exempte d'arbitraire.

Les traits sémantiques n'ont pas tous le même statut et ils expriment différentes familles d'informations. Le modèle offre la possibilité de caractériser les traits sémantiques utilisés, et en particulier propose un certain nombre de types de traits, qui vont être décrits dans les points suivants. Chaque type de traits peut, dans une utilisation du modèle, être associé à 0, un ou plusieurs traits sémantiques.

Quels que soient les types de traits utilisés, le modèle n'impose aucun nom de trait, aucune liste de valeurs. Cela signifie qu'une instanciation du modèle se définit (a priori ou de façon incrémentable, lors du codage du lexique) l'ensemble des traits dont elle se dote, et qui constitue en quelque sorte son instanciation du langage de traits du modèle.

3.1. Traits de type général (propriété)

Les **traits sémantiques généraux** (on pourrait aussi dire **propriétés**) sont ceux utilisés

classiquement pour exprimer les restrictions de sélection sémantiques. Celles-ci peuvent d'ailleurs s'exprimer sous la forme de tels traits dès la couche syntaxique pour les utilisateurs qui le désirent. Ils sont alors utilisés dans les contraintes syntaxico-sémantiques sur les occupants de positions.

Ces traits sont généraux en ce sens qu'ils ne sont pas dédiés à une classe sémantique ou un domaine particulier. Ils s'expriment souvent avec des valeurs *plus* ou *moins* .

Les traits `animé, humain, comptable, masse, abstrait, concret, action, processus, état` rentrent habituellement dans cette catégorie de traits généraux.

Il faut bien remarquer qu'on n'indiquera au niveau du lexique que les informations correspondant à un usage *standard ou typique* et sans effet de sens recherché. C'est ainsi que, bien souvent, les figures (métaphores ou métonymies...) seront détectées lors des traitements d'analyse par le fait que certaines restrictions de sélection portant sur des traits sémantiques généraux seront violées.

Ex : *Ma voiture boit de l'essence*

renvoie à un prédicat à deux arguments contraints sémantiquement :

boire `(Arg0[animé : +], Arg1 [liquide : +]`

`[comestible : +])`

dont les contraintes sont ici violées, avec effet recherché.

Le lexique ne rendra bien sûr pas compte de tels tropes, à moins que ceux-ci ne se soient lexicalisés, auquel cas une entrée sémantique particulière pourra être dédiée à la description de cet usage.

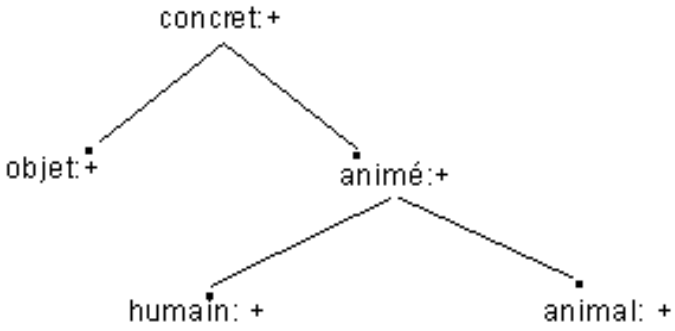
Il faut pouvoir préciser que certains traits sont "obligatoires" (et donc la violation d'une restriction de sélection portant sur l'un d'eux est à prendre en considération par des traitements spécifiques, soit pour identifier l'effet recherché, soit pour rejeter l'interprétation sémantique associée au profit d'une autre compatible avec les informations morphologiques et syntaxiques.

D'autres traits moins forts pourraient être appelés "préférentiels".

Le modèle permet de rendre compte de la "pondération" différente de ces informations par un attribut précisant la pondération associée à cette information de trait valué.

Il est intéressant du point de vue du pouvoir d'expression, d'offrir la possibilité de structurer les connaissances ; c'est pourquoi le modèle permet de faire entrer les traits dans des treillis. La factorisation et la mise en oeuvre de mécanismes d'héritage sont ainsi rendues possibles.

Ex :



La présence de `[animal : +]` permettrait de déduire :

`[humain : -]`

`[animé : +]`

`[concret : +]`

Remarque : La structuration des traits valués se fait de façons différentes suivant le type des valeurs. Les traits "binaires" (à valeur + ou -) seront structurés nécessairement sur la valeur +, les deux valeurs étant exclusives l'une de l'autre. Les traits à liste de valeurs, qu'elle soit connue du modèle (liste fermée) ou non (liste ouverte), auront un système d'héritage, une valeur n'excluant les autres que si le trait est défini comme monovalué.

3.2. Traits de type classe sémantique

Un trait particulier très important est celui qui caractérise l'unité sémantique en terme de **classe sémantique**. La notion de classe sémantique est relativement vague bien qu'intuitive. Une classe regroupe une information sémantique non triviale qu'on souhaite identifier et désigner sans nécessairement la décomposer en un ensemble de traits sémantiques élémentaires ; elle représente une notion conceptuelle générique fonctionnellement pertinente dans le système linguistique de la langue décrite et à laquelle on souhaite rattacher une unité sémantique.

Suivant l'approche théorique ou méthodologique retenue, la description du lexique en classes sémantiques et l'élaboration de la liste des classes utilisées dans la couche sémantique sera une donnée préexistante, sorte de description a priori, ou au contraire émergera du lexique lui-même.

En effet, les classes sémantiques peuvent être mises en évidence de façon paradigmatique par certains prédicats sémantiquement assez contraints (jouant le rôle d'opérateurs) qui peuvent prendre comme argument de rang n un élément de la classe. C'est l'approche de G. Gross [Gross-1994], qui permet de mettre en évidence les classes sémantiques.

Par exemple, on peut considérer qu'il existe une classe sémantique `vêtement` que certains prédicats *spécialisés* mettront en évidence (*porter, essayer, aller à* ...). D'autres classes sémantiques pourraient être par exemple : `profession, véhicule, aliment, moyen de transport`...

Quelle que soit la méthodologie choisie pour identifier les classes, le découpage présentera un caractère partiellement arbitraire : en effet, même si on fait appara"tre les classes sémantiques à partir du rôle possible que peuvent jouer leurs éléments comme actants de certains prédicats

spécialisés, le choix de la liste de prédicats qui devra être associée à une classe sémantique déterminera la classe et réciproquement.

Bien qu'elles soient nommées et aient une "existence" propre à l'intérieur du modèle, il semble intéressant de pouvoir décrire les classes sémantiques par un ensemble de traits sémantiques : la description conserve donc son caractère componentiel à ce niveau également, si on le désire, tout en mettant en évidence la classe partageant un ensemble de traits sémantiques pertinent pour la langue. Les éléments d'une classe sémantique sont donc implicitement porteurs de l'ensemble des traits associés à la classe.

La notion de classe sémantique peut aussi être vue comme taxinomique, et un treillis ou une arborescence de classes pourront donc être décrit. Deux classes soeurs dans cette structure divergent (au moins implicitement, mais explicitement de préférence) par la valeur d'au moins un trait parmi les traits sémantiques valués impliqués par la classe sémantique ; ce trait divergent pourra être un trait général ou un trait distinctif ou spécifique.

Les différents éléments appartenant à une même classe sémantique (c'est-à-dire porteurs du même trait valué de classe sémantique) pourront être par ailleurs porteurs de certains traits divergents, qui joueront le rôle de traits distinctifs à l'intérieur de la classe.

Suivant une certaine approche théorique (à définir pour une instanciation de dictionnaire au modèle GENELEX et qui sera précisée par les propriétés que l'on donnera aux traits de classe utilisés), une unité sémantique peut appartenir à différentes classes sémantiques ; un trait de classe sémantique représente alors en quelque sorte un point de vue sur l'unité sémantique.

Ex : *avion* peut alors appartenir aux classes `véhicule` et `objet volant` dans une telle approche.

Une utilisation plus restrictive des traits de classe sémantique est aussi possible : elle peut alors s'appuyer sur un arbre de traits valués de classe sémantique, dans lequel on situe les entrées décrites par ces traits de classification (monovalué et donc exclusif dans ce cas).

Remarque sur la représentation des classes sémantiques dans le modèle

Le modèle GENELEX permet de représenter les classes sémantiques par des traits sémantiques valués associés à des traits dits "de classe sémantique". Quelle que soit la méthode linguistique utilisée pour identifier les classes sémantiques, la représentation dans le modèle est la même.

Suivant les options choisies, on se donnera la contrainte d'affecter une ou plusieurs classes aux unités, il s'agit là d'une caractéristique du trait de classe (mono/multi-valué). Le modèle ne prévoit pas de garder trace de l'ensemble de prédicats ayant fait appara"tre la classe (si telle a été la méthode employée). Il permet d'associer à un trait valué de classe sémantique des traits sémantiques valués qui sont impliqués par l'appartenance à cette classe (et qui peuvent, si tel a été le choix d'instanciation du modèle, être l'ensemble de traits valués qui ont permis d'identifier les éléments de la classe), et aussi un ensemble de traits valués qui sont incompatibles avec l'appartenance à la classe. Il permet également d'associer à un tel trait valué de classe un ensemble de traits qu'il est pertinent (ou obligatoire) de renseigner.

D'autre part, une structuration taxonomique des classes (par une relation est_un entre traits valués) permet de structurer l'ensemble des traits valués de classe, et donc d'avoir une précision variable.

3.3. Traits de type distinctif

Pour différencier les éléments d'une même classe sémantique, et ce dans une approche componentielle, on utilisera des **traits spécifiques distinctifs**. Ces traits se différencient des traits sémantiques généraux en ce sens qu'ils sont généralement liés à une certaine classe sémantique ou à un domaine, et qu'on ne les utilise pas pour décrire des unités sémantiques dans tout le lexique.

Un exemple de traits spécifiques distinctifs serait, à l'intérieur de la classe sémantique des moyens de transport, les traits : *urbain* et *collectif*.

3.4. Traits de type domaine

Le sens d'une unité lexicale est bien souvent lié à un domaine d'activité particulier. Il est donc nécessaire de pouvoir préciser ce domaine par un trait. L'ensemble des valeurs de ce trait constitue une sorte de description des activités rencontrées dans un monde donné.

Il s'agit d'une information qui permet de relier l'unité sémantique à l'univers dans lequel elle est utilisée, et le lien avec des connaissances du domaine à caractère plus spécialisé (hors du dictionnaire générique pour une bonne part) pourrait se faire, au moins en partie, par l'intermédiaire de ce trait.

Les valeurs du trait *domaine* représentent une hiérarchie. Par exemple, *médecine* serait une valeur possible, mais aussi *chirurgie, psychiatrie...* qui sont hiérarchiquement sous la dépendance de *médecine*. Sans préconiser d'adopter une description trop fine des domaines, il est intéressant de prévoir qu'une hiérarchie à deux ou trois niveaux par exemple soit décrite dans GENELEX. Le trait *domaine* pourra prendre ses valeurs dans l'un quelconque de ces niveaux.

Remarque : Les types de traits sémantiques que nous avons présentés jusqu'ici permettent de décrire le contenu objectif ou inhérent de l'unité sémantique. Il est important de pouvoir exprimer également un ensemble d'informations partiellement subjectives, évaluatives ou expressives, associées au contexte d'emploi en situation des unités décrites. Ces informations périphériques ou non-essentielles sont utiles, tant pour pouvoir interpréter correctement un énoncé ou un texte, que pour la génération. Les traits présentés maintenant contiennent ce type d'information.

Les types de traits présentés plus haut peuvent être utilisés au niveau purement lexical de l'Usem, ou dans la caractérisation d'objets descriptifs plus abstraits (concepts, prédicats lexicaux ou non).

Les traits dont nous allons parler maintenant sont plutôt destinés à un usage de description de niveau lexical fin (mais rien n'empêche de les utiliser dans tous les niveaux de description) .

3.5. Traits de type niveau de langue

Une unité sémantique peut comporter un certain **niveau de langue**. En effet, l'acception peut être

marquée du point de vue de l'emploi sans que l'entrée morphologique (associée à différentes constructions et différents sens) ne le soit, ni la construction décrivant le contexte d'occurrence de l'unité sémantique.

Ex : Une des acceptions de *canard (journal)* est d'un niveau de langue `familier.`

Cette information qui du point de vue de sa sémantique s'apparente à un trait valué, est représentée de façon unifiée dans le modèle GENELEX par l'affectation d'un COMBVE (combinaison de valeurs d'emploi) qui rend compte non seulement du niveau de langue, mais également de la fréquence et de la datation de l'emploi. C'est une information qui, pour la couche sémantique, est nécessairement portée par l'Usem **.**

3.6. Traits de type connotation et évaluation

Les phénomènes que l'on désigne couramment par connotation sont très fortement marqués par un contexte culturel. Ils sont liés à la description d'informations très prototypiques et ils interviennent si l'on veut décrire avec précision la sémantique d'une unité lexicale, ce qui est nécessaire pour certaines applications et d'un point de vue de pure lexicographie. C'est pourquoi le modèle propose d'en rendre compte par des traits de connotation.

Un exemple de tel trait serait celui de *`force`* à valeur *`plus`* pour une acception de *`homme`* et à valeur *`moins`* pour une entrée de *`femme`*.

Certaines unités sémantiques servent en quelque sorte de référence dans la langue, elles présentent un aspect de neutralité ou de généricité d'emploi que n'ont pas d'autres unités proches sémantiquement.

Il en va ainsi de certaines unités associées à des adjectifs : par exemple, dans l'ensemble {*bon, mauvais, excellent, nul* } *bon* semble être l'entrée neutre, les autres étant marquées en quelque sorte par rapport à *bon* . Ainsi, on peut demander : *c'est bon ?* sans nécessairement attendre de réponse positive, mais il est plus difficile d'interpréter de façon neutre la question : *C'est mauvais ?*

Un trait particulier pourra permettre d'identifier ces entrées "*référence"* .

> *Les représentations de connaissances pragmatiques dépassent les ambitions de la couche sémantique de GENELEX. Cependant, les limites sont relativement floues et il faut permettre à des connaissances pragmatiques de se greffer sur la couche sémantique. C'est le but des traits suivants.*

3.7. Traits de type pragmatique

On pourra noter sous ce type de traits des informations concernant l'ancrage référentiel des unités décrites, en fonction des différentes conditions d'énonciation.

On pourra marquer par exemple: `papa : [déictique : +]`

`père : [relationnel: +]`

`homme : [absolu: +]`

On pourra aussi marquer des informations concernant l'usage argumentatif de l'unité décrite.

La référence à un niveau d'informations pragmatique est aussi offerte par le modèle par le biais des listes de Prédicats.

Le modèle GENELEX se veut ouvert, au niveau sémantique plus encore qu'aux niveaux morphologique ou syntaxique. C'est pourquoi il propose un certain nombre de types de traits, mais laisse aussi la possibilité aux utilisateurs d'exprimer et de structurer un peu différemment l'information, en se situant toujours dans le cadre très général offert par le formalisme de traits valués.

3.8. Traits de type divers

Il est laissé à l'utilisateur toute liberté d'utiliser le langage d'expression des traits valués dans un cadre plus large que celui qui est précisé dans ce qui précède.

Divers traits dont le statut n'est pas contraint par le modèle mais laissé entièrement libre aux utilisateurs pourront donc être utilisés, et ils permettront de rendre compte par exemple, d'informations concernant l'aspect, la temporalité, de marquer éventuellement différentes grandes catégories sémantiques associées à une catégorie morpho-syntaxique donnée.

On pourra ici marquer l'appartenance à un champ lexical.

Dans un autre ordre d'idées, on pourra affecter des traits de catégorisation sémantique aux unités, par exemple `relation humaine` pour certaines unités (ami, frère, proche...) ainsi que les catégories sémantiques des verbes.

Des relations de lexicalisation entre Trait sémantique et Usem et entre Trait sémantique valué et Usem permettent de plonger le méta-langage des traits sémantiques dans le langage, c'est-à-dire dans le lexique.

En se basant sur certains traits valués (présents ici à titre d'exemple) on pourra avoir :

l'unité sémantique *vêtement* correspond par son sens à ce que désigne le trait valué `[classe : vêtement].` Elle peut donc être en relation de lexicalisation avec le trait valué en question.

l'unité sémantique *couleur* correspond par son sens à ce que désigne le trait `[couleur : ....].` Elle peut donc être dans une relation de lexicalisation avec lui.

l'unité sémantique *rouge* correspond par son sens à ce que désigne le trait valué `[couleur : rouge].` Elle peut donc être en relation de lexicalisation avec ce trait valué.

4. Relations sémantiques entre Unités Sémantiques

Le point précédent développait l'approche analytique permettant de décrire l'unité sémantique ou les objets descriptifs plus abstraits de la sémantique (prédicat, concept) à l'aide de traits valués, composants sémantiques de l'unité décrite. Cette approche peut être poussée à l'extrême et alors se

suffire à elle-même : les relations entre unités sémantiques se déduisent de la présence de tel ou tel trait avec telle ou telle valeur, le jeu de traits composant un métalangage.

Le modèle de la couche sémantique GENELEX ne souhaite pas imposer cette approche dans sa totalité, et souhaite se donner la possibilité de décrire l'unité sémantique également *de l'extérieur* par les relations qu'elle entretient avec les autres unités sémantiques, ou avec les prédicats ou les concepts que partage un ensemble de celles-ci.

En effet, les relations sémantiques permettent de décrire d'un point de vue global la structure du lexique pour un ensemble de relations d'une part, et de décrire les collocations ou préférences sémantiques locales d'autre part.

Il faut bien remarquer qu'il est intéressant de voir le lexique comme un ensemble d'éléments lexicaux dont la place à l'intérieur de l'ensemble (et donc l'apport sémantique particulier) est définie par le tissu de relations qui le lient à d'autres éléments de cet ensemble.

Nous nous intéresserons ici à la grande famille des relations sémantiques situées au niveau lexical. Ces relations ont été étudiées de façon particulièrement fine par I . Mel'cuk et ses collaborateurs [Mel’cuk 1984 -1988], qui en font un des axes prioritaires de la description d'une unité lexicale. Ce travail dans son ensemble nous a semblé une référence particulièrement riche d'enseignements et d'expérience lexicographique. Sans vouloir reprendre totalement son approche globale, il est certainement intéressant de se référer à ces études et d'être capable de rendre compte de ce qu’il appelle les "fonctions lexicales".

Par ailleurs un certain nombre de relations sont traditionnellement identifiées par la lexicographie. Cela constitue un apport essentiel à notre réflexion et nous ne saurions l'ignorer. Le modèle permettra donc de représenter différents types de relations.

D'autre part, le niveau "*représentation de connaissances conceptuelles"* pourra s'appuyer sur l'identification de *concepts ou de prédicats* et de leurs relations ontologiques avec d'autres *concepts ou prédicats ;* cet aspect sera développé plus loin.

Les relations sémantiques entre Unités sémantiques du modèle sont de différentes natures : paradigmatiques, de dérivation (sens large ou sens strict), de collocation. On verra plus loin qu’il existe également des relations sémantiques entre des objets présentant un certain caractère d’abstraction par rapport aux Usem, c'est-à-dire des relations sémantiques entre prédicats, entre concepts, et entre un prédicat et un concept. Par ailleurs, le lien qui associe une Usem à son prédicat est porteur d’information, ce lien est décrit dans ce qu’on appelle la représentation prédicative.

Nous nous intéressons à présent aux relations sémantiques entre Usem.

4.1. Relations de type paradigmatique entre unités substituables

Il est indispensable de rendre compte des relations sémantiques figurant traditionnellement dans les dictionnaires.

Il s'agit de relations liant deux unités sémantiques associées à des unités morphologiques de même catégorie morpho-syntaxique. Les unités reliées par cet ensemble de relations sont *substituables* dans leurs contextes d'occurrence, moyennant des modifications du sens desdits contextes qui dépendent bien sžr de la relation en question.

Sans vouloir en dresser une liste précise et exhaustive, on y trouvera entre autres les relations de

- synonymie

- antonymie (qui recouvre contraire, réciproque, et complémentaire)

- opposition

- taxinomie

- partie-tout (méronymie)

Ces relations sont très fortement présentes dans tout le lexique, indépendamment des catégories morpho-syntaxiques (*ouvertes*) et des domaines ou classes sémantiques. Ce sont des **relations générales fondamentales** en ce qui concerne la structure du lexique.

Nous ne nous étendrons pas sur ces relations, bien qu'elles soient sans doute les plus indispensables à la description du lexique.

Ces relations sémantiques méritent d'être décrites par leur type, mais aussi par le fait qu'elles ont ou non quelques propriétés : réflexivité, transitivité, symétrie (pour les relations de synonymie par exemple), ou antisymétrie, ou le fait que ce sont ou non des relations d’ordre.

D'autre part, les relations elles-mêmes peuvent être liées entre elles par des relations de généralisation (est-un), d’incompatibilité, ou d’implication. C'est ainsi que l'on pourra décrire différents degrés de synonymie, et qu'on pourra avoir à la fois une vue fine de ces différentes relations, et par la structuration de cette famille de relations, une vue générale ne comportant virtuellement que la relation de synonymie large englobant tout cet ensemble. Cela permet ainsi une vue des informations sémantiques du lexique à précision variable.

Les relations sémantiques sont par ailleurs regroupées en grandes familles ou "types", précisés par un "sous-type", et certaines de ces relations peuvent être choisies comme "pivot", c’est-à-dire être communes aux différentes langues.

Le modèle sémantique offre par ailleurs la possibilité de donner à l’information d’association de deux Usem liées par une relation sémantique une modalité comportant à la fois une pondération et un point de vue sur cette information.

4.2. Relations sémantiques de type dérivation

Un important ensemble de relations dont il convient de rendre compte est celui des dérivations sémantiques. Ces relations sont, comme les précédentes, à la fois générales et fondamentales en ce sens qu'elles structurent l'ensemble du lexique, et qu'elles lient deux unités sémantiques.

Rappelons que la couche syntaxique permet de décrire des transformations syntaxiques entre Usyn, que ces Usyn soient associées aux mêmes UM ou non. Une famille de transformations pourra concerner ce que l’on pourrait appeler la dérivation syntaxique, et pour certains couples d’Usem, la relation de dérivation sémantique qui les lie trouvera un écho dans la couche syntaxique dans une relation de transformation/dérivation syntaxique liant les Usyn associées.

Par ailleurs, ces relations lient des unités lexicales dont les catégories morphologiques de l'UM associée à l'Usem via une ou plusieurs Usyn sont généralement distinctes.

Ces relations dépendent des catégories qu'elles relient. Un grand nombre d'entre elles sont issues d'unités sémantiques prédicatives, nous reviendrons donc plus en détail sur l'aspect prédicatif dans le point suivant. Il convient d'ores et déjà de préciser qu'il s'agit par exemple de :

- Relations pointant vers le ième argument typique du prédicat associé à l'entrée, que ce soit comme unité sémantique associée à un nom (nom de l'actant) ou un adjectif (qualifiant cet actant suivant différentes modalités)

```
Usem Vendre ---> Usem Client(Nom, Arg2)

Usem Lire--> Usem Livre(Nom, Arg1)

Usem Ecrire1--> Usem Livre(Nom, Arg1)

Usem Ecrire1--> Usem Ecrivain(Nom, Arg0)

Usem Croire1 ---> Usem Crédule(Adjectif, Arg0)

Usem Croire1 ---> Usem Incrédule(Adjectif,Arg0)

Usem Croire2 ---> Usem Croyant (Adjectif, Arg0)
```

- Relations pointant vers le circonstant typique d'une action : instrument, lieu, résultat, moyen, but ...

```
Usem Tennis ---> Usem Court(lieu)

Usem Découper --> Usem Ciseaux(moyen)
```

- Relations pointant vers l'adjectif qualifiant le fait de pouvoir/ne pas pouvoir être le ième argument.

```
Usem Manger --> Usem Comestible(Arg1)

Usem Croire1 ---> Usem Crédible(Adjectif, Arg1)

Usem Boire --> Usem Potable(Arg1)
```

- Relations de dérivation sémantique au sens strict . Le noyau de sens des entrées liées par une telle relation est en quelque sorte le même, mais les éléments en relation sont de catégories différentes, et ont donc des contextes d'insertion différents.

```
Usem Prison--> Usem Carcéral

Usem Jeu--> Usem Ludique

Usem Territoire-->Usem Territorial
```

Ces relations sont particulièrement importantes pour tout ce qui concerne les relations paraphrastiques entre énoncés synonymes. Elles permettent d'effectuer un choix lexical en fonction d'un contexte global au moment de la génération, et de choisir, pour un sens donné, l'unité sémantique de la catégorie voulue par le contexte plus global et les contraintes grammaticales et stylistiques.

- Relation pointant vers les activités typiques associées à une unité lexicale nominale (permettant de rendre compte du rôle *Telic* de Pustejovsky)

```
Usem Livre--> Usem Lire(Arg1)

Usem Livre--> Usem Ecrire(Arg1)
```

Remarque 1: les relations liant les Usem sont ici représentées simplement par "-->", mais il s'agit de relations sémantiques explicitées par le modèle.

Remarque 2 : Certaines de ces relations entre Usem peuvent sembler redondantes si l'on considère le pouvoir d'expression offert par le modèle autour du prédicat, et des connaissances qu'il a sur ses arguments (voir plus loin). Cependant, pour bien saisir comment s'articulent les différents moyens de description offerts par le modèle, il faut comprendre:

- que les relations dont il est question ici se placent délibérément au niveau du lexique, et qu'elles peuvent avoir une étendue ou une portée très grande, et un niveau réduit d'abstraction sur le lexique.

- que les informations portées par le Prédicat sont pour la plupart d'un niveau d'abstraction beaucoup plus grand (traits sémantiques valués, concepts, prédicats instanciés,..) ; elles peuvent, il est vrai, s'exprimer par des Usem, donc à un niveau lexical pour des valeurs par défaut ; ce sont des informations "définitoires" du Prédicat, et ce que l'on peut exprimer sur le Prédicat et ses arguments est donc plus limité. Il est vrai que certaines relations entre Usem sont déductibles des informations portées par le Prédicat et par les relations qu'il entretient avec d'autres objets descriptifs. Dans ce cas, la stratégie de codage, pour une utilisation du modèle, doit préciser s'il vaut mieux coder certaines informations à la fois entre Usem et, pour celles qui peuvent l'être, au niveau du Prédicat, ou bien s'il faut ne coder sur les Usem que celles qu'on ne sait pas recalculer à partir de niveaux plus abstraits de description.

- Dans tous les cas, le modèle se doit de permettre le codage au niveau de l'Usem : en effet, le niveau dit "abstrait" n'est pas nécessairement utilisé par toutes les instanciations du modèle.

4.3. Relations sémantiques de type collocation

Les couches morphologique et syntaxique du modèle permettent de représenter le phénomène de ce qui est appelé la "composition", et qui recouvre une grande variété de cas. La composition en morphologie est réservée à la composition "figée" présentant quelques caractéristiques de forme ou a-syntaxiques, la composition en syntaxe permet de décrire du quasi-figé au quasi-libre, et on l'a vu, certains "composés" peuvent être codés en morphologie ou en syntaxe, et certains autres peuvent être identifiés dans la couche syntaxique ou non. On peut considérer que le phénomène de collocation correspond à ce que l'on peut aussi nommer la composition en sémantique. Les unités lexicales liées par des relations de collocation se rencontrent généralement, en surface, dans un même syntagme ; elles se sélectionnent mutuellement et généralement, chacune apporte sa part sémantique, laissant ainsi le jeu de la compositionnalité de sens se faire.

Dans la plupart des cas, il semble que les couples d'Usem mis en jeu par ces relations donnent une importance plus grande à l'une des deux unités, la *base* en quelque sorte de la collocation qui sélectionne l'unité avec laquelle elle est en relation de collocation (son modifieur privilégié, son verbe support, etc) par une *fonction lexicale* donnée.

Ces relations sont particulièrement cruciales pour les applications de traduction automatique et de génération ; bien souvent elles permettent, en analyse, de désambigu•ser les deux entrées ensemble.

On pourra faire entrer dans cette famille les relations suivantes :

- Relations qui lient une unité lexicale de catégorie nom et une autre de catégorie adjectif, modifieur (d'intensité par exemple) associé de façon privilégiée au nom (collocation et préférence)

`Usem Prix--> Usem Exorbitant`

(préféré par collocation à `Usem Exagéré`)

`Usem Peur--> Usem Panique`

`Usem Pluie--> Usem Diluvien`

- Relations qui lient une unité lexicale de catégorie nom et une autre de catégorie verbe, dont le nom est actant privilégié :

`Usem Fièvre--> Usem Monter`

`Usem Prix--> Usem Monter`

- Relations qui lient une unité lexicale nominale et prédicative à l'unité du *verbe support* auquel il s'associe pour une réalisation syntagmatique.

Le verbe support est presque vide, cependant il n'est alors pas totalement dénué d'apport sémantique, car généralement il contient minimalement l'aspect, ce qui peut justifier qu'il soit associé à une unité sémantique.

`Usem Attention--> Usem porter (inchoatif)`

```
Usem Attention--> Usem maintenir (duratif)

Usem Attention--> Usem faire (neutre)
```

-Relations qui lient deux unités lexicales de catégorie nom, l'une jouant un certain rôle par rapport à l'autre, rôle que précise la relation.

```
Usem Loup--> Usem Meute

(groupe de)

Usem Pain--> Usem Miette

(petite partie de)

Usem Lycée--> Usem Proviseur

(responsable de)
```

5. Prédicat

Il a déjà été fait référence tout au long de ce document à la notion de prédicat, notion essentielle dans la modélisation de la sémantique de nombreuses unités lexicales. Bien que la notion soit assez courante, on rencontre parfois le terme *Prédicat* avec une acception différente, et nous allons préciser ce que nous entendons par *Prédicat*, comment cette notion est représentée dans le modèle GENELEX, et comment elle s'articule avec les autres objets du modèle.

5.1. Mise en évidence de la notion de prédicat

Considérons les énoncés :

Le Mexique achète deux usines atomiques à la France pour une somme modique.

L'achat de deux usines atomiques à la France par le Mexique renforce les liens d'amitié entre les deux pays.

Les nouveaux achats du Mexique seront mis en service au début du mois prochain.

L'acheteur des usines a payé comptant.

Intuitivement, on perçoit qu'ils partagent quelque chose de particulier, qu'on souhaite mettre en évidence : une relation associée à une structure actancielle, qui se réalise de différentes façons lexicalement et dans la structure de surface. C'est ce qu'il y a de commun à *acheter*, *achat (action)*, *achat(résultat)*, *acheteur* qui semble être une donnée de la langue, et que l'on souhaite identifier et

décrire dans la notion de prédicat, ici le prédicat *acheter(acheteur, objet, vendeur, prix).*

On entendra donc par **prédicat lexical** une telle relation qui décrit une **situation** et qui est identifiée dans la langue décrite. Elle comprend un certain nombre **d'intervenants** ou **participants** qui y jouent un certain **rôle** ; ce sont les **actants ou arguments sémantiques** associés à la situation décrite. Le prédicat lexical a une ou plusieurs lexicalisations et celles-ci s'insèrent dans des contextes syntaxiques ne faisant pas toujours mention de tous les intervenants de la situation ; certains peuvent rester implicites.

La notion de prédicat est ici mise en évidence par l'observation de ce que nomme la langue. Elle est généralisée dans le modèle, que la langue identifie ou non la relation qui lie les différents arguments sémantiques du prédicat. Le modèle distingue donc les prédicats **lexicaux** (identifiés par la langue) des prédicats **non lexicaux**.

Il faut bien remarquer que la notion de prédicat lexical est une tentative de distinguer et de représenter les prédicats que la langue identifie et que la liste des prédicats lexicaux ainsi identifiés est propre à chaque langue, et non prédéfinie a priori. Les prédicats plus abstraits (non lexicaux) peuvent être définis a priori ou comme généralisations de prédicats lexicaux. On affectera aux prédicats un attribut `type` permettant d'expliciter leur nature lexicale ou non. Formellement, un prédicat lexical ne se distingue d'un prédicat non lexical que par le fait qu'une Usem au moins pointe directement vers lui via une Représentation Prédicative.

5.2. Description

Le modèle sémantique GENELEX identifie donc explicitement des prédicats "extraits" des unités sémantiques dont ils rendent compte d'une bonne partie du sens et qui leur sont rattachés.

Un prédicat est caractérisé par sa structure actancielle qui se définit de la façon suivante :

- On associe au prédicat un certain nombre d'**arguments** (ou actants) ; le nombre d'arguments est le nombre de participants à la situation ou à l'action décrite par le prédicat ; il correspond à une structure saturée, qui ne sera peut-être que partiellement remplie dans les contextes associés aux lexicalisations du prédicat. Le choix de la liste des arguments n'est cependant pas toujours simple, en particulier le choix de considérer comme argument ou non un élément que l'on pourrait qualifier de modifiant circonstanciel ; chaque instanciation de dictionnaire devra se donner ses propres critères.

Chacun des arguments joue un rôle sémantique particulier dans la situation décrite, celui-ci s'exprime par le fait qu'on lui affecte un ou plusieurs **rôles sémantiques.** Ces rôles sont proches des rôles thématiques utilisés en syntaxe. Cependant, on peut souhaiter à ce niveau avoir un ensemble de valeurs plus précises permettant de décrire de façon fine le rôle sémantique joué dans la situation décrite par le prédicat. Les valeurs des rôles devront donc être compatibles avec celles provenant de la syntaxe. Par défaut, on pourra décider d'utiliser la même liste de valeurs que celle affectée aux rôles thématiques, si l'instance du dictionnaire concernée en comporte.

D'autre part, le modèle permet de structurer hiérarchiquement les rôles sémantiques, afin de pointer à un niveau plus ou moins précis et d'associer des rôles plus ou moins précis en fonction des arguments de prédicats.

Dans la structure d'arguments associée au prédicat, chacun des arguments est chargé d'informations sémantiques plus ou moins complexes, et auxquelles sont associés différents statuts, ce qui accro"t le pouvoir de description des arguments.

Les différents **statuts** portés par les informations sémantiques associées aux arguments sont les suivants:

- DEFAUT : il s'agit d'une information par défaut sur l'argument, lorsque le contexte n'instancie pas celui-ci

- VERIF : il s'agit d'une information de vérification ; la représentation sémantique de l'argument instancié (quand il l'est) doit vérifier ces informations

- ENRICHIT : il s'agit d'une information qui enrichit la représentation de l'argument, que celui-ci soit instancié ou non

- DEFAUT_VERIF : il s'agit d'une information de vérification ou enrichissement compatible en cas d'instanciation de l'argument ou de défaut suivant que le contexte représente ou non l'argument.

Ces informations sémantiques sur les arguments sont de différentes natures : des traits sémantiques valués, des concepts, éventuellement un prédicat instancié (en cas d'argument complexe) et une Usem.

Les traits sémantiques utilisés pour ces contraintes seront ceux que nous avons décrits comme correspondant à une information *objective,* ainsi que les traits laissés libres à l'utilisateur (*divers*).

L'Usem est utilisée afin de donner à chacun des arguments une valeur par défaut, la valeur qu'il prend lorsque la structure sémantique issue d'une structure de surface n'affecte pas de valeur à ce prédicat. Cette valeur par défaut est généralement liée au prédicat, et on doit donc la décrire au niveau du prédicat. Cependant, lors de certaines correspondances avec la syntaxe, la valeur par défaut est différente de celle attribuée par le prédicat : la correspondance la précise alors .

Le prédicat est donc défini, au moins partiellement, par cette structure actancielle, qui constitue sa "signature" sémantique.

5.3. Conséquences sur les points du modèle abordés précédemment

Les informations concernant le prédicat seront décrites au niveau de celui-ci et donc héritées par chacune des unités sémantiques partageant le prédicat. Les unités sémantiques prédicatives ont donc une bonne part de leurs informations qui sont portées par le prédicat lexical et qu'elles partagent avec les autres unités sémantiques associées au même prédicat lexical. Cela a des conséquences sur le contenu que l'on associera aux unités sémantiques, qui doit être le complément de ce que l'on trouvera au niveau du prédicat.

5.3.1. Au niveau componentiel

Le prédicat peut lui-même être porteur de certains traits sémantiques, et il conna"t un certain nombre

d’informations sémantiques concernant ses arguments.

Les traits sémantiques portés par le prédicat peuvent être des traits "généraux" ou de classification de prédicats.

Les informations (dont le statut est de vérification) concernant les arguments du prédicat sont des informations permettant un filtrage associé à la correspondance syntaxe-sémantique (en particulier quand il s’agit de traits) ou une détection d'usages non standard des unités lexicales. Il doit dans tous les cas y avoir cohérence et compatibilité entre les informations de correspondance syntaxe-sémantique et les informations sur la structure actancielle du prédicat. Les traits précisés par la correspondance syntaxe-sémantique seront présents s'ils précisent ou complètent ceux portés par la structure argumentale du prédicat.

5.3.2. Au niveau des relations sémantiques

On peut décrire un certain nombre de relations au niveau des prédicats, ce qui permet de factoriser des relations entre unités sémantiques qui ne seraient alors plus qu'implicitement liées sémantiquement, ces relations étant calculables à partir des relations entre prédicats. L'expression plus concise qu'elles permettent justifie l'utilisation de ces relations entre prédicats. Celles-ci n’excluent cependant pas les relations entre Usem, même si on peut y voir un certain risque de redondance : ici encore c’est la stratégie lexicographique d’une instance de dictionnaire qui fixe les limites de ce que l’on code sur les Usem, sur les prédicats, et qui gère les problèmes de redondance.

Cette concision peut s'appuyer sur deux types de liens dans le modèle :

1 - On peut parler du **lien entre une unité sémantique et son prédicat lexical**

(il ne s’agit pas à proprement parler de ce que nous appelons relation sémantique, ce lien est représenté dans le modèle par l’objet Représentation Prédicative)

En effet, la façon dont l’Usem incorpore le prédicat est porteuse de sens. On peut définir, pour chacune des unités sémantiques qui pointent vers le prédicat, la nature du lien qui le lie à celui-ci. Ainsi, le verbe *acheter* aura une unité sémantique qui sera associée au prédicat *acheter* en tant que lexicalisation *principale (vedette)* mettant l'accent sur la situation et ses conditions de réalisation. Le nom *acheteur* sera une lexicalisation nominale incorporant l'argument 0 du prédicat. Une acception de *achat* sera la lexicalisation du prédicat incorporant l'argument 1 (objet), une autre acception concernera le prédicat comme nom de l’action.

La Représentation Prédicative précise si un argument est concerné dans l’association d’un Prédicat à une Usem et lequel, et, si oui, si cet argument est incorporé par l’Usem ou non. D’autre part, le lien est nommé et donc on peut caractériser différemment les liens qui associent une Usem à son Prédicat.

2 - Les **relations sémantiques entre prédicats lexicaux**. Ces relations préciseront la façon dont se correspondent les arguments de chacun des prédicats en relation, et la relation qui les lie permet de préciser également des informations sémantiques sur les arguments, celles-ci se rajoutant aux informations que le prédicat conna"t sur ses arguments.

Ex: *Clouer* est un exemple de prédicat pour lequel on peut douter du nombre d’arguments

nécessaires à sa description. On peut choisir que le moyen de fixation est un argument du prédicat, ou non.

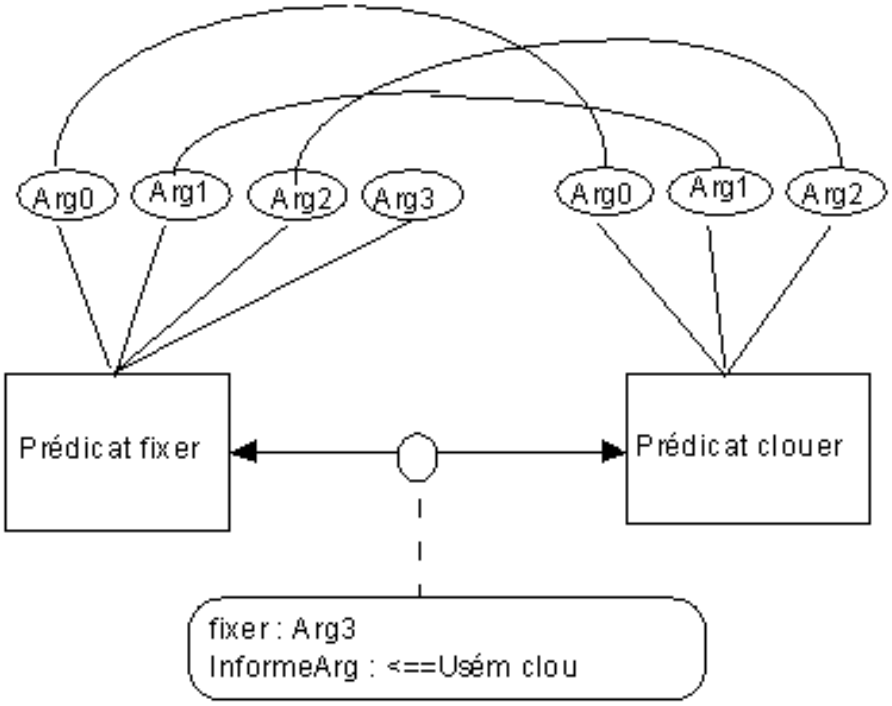
On peut donc considérer que le prédicat *Clouer* a trois arguments, il est alors décrit comme *Clouer (agent, objet, lieu)* . On décrira alors sa relation avec le prédicat *Fixer* comme suit

Clouer(agent, objet, lieu)

<---Spécialisation/Généralisation--->

fixer(agent, objet, lieu, moyen)

La relation sémantique qui lie les deux prédicats précise alors la valeur de l'argument 3 de *Fixer* qui est forcée dans *Clouer* : le moyen de fixation est un Clou



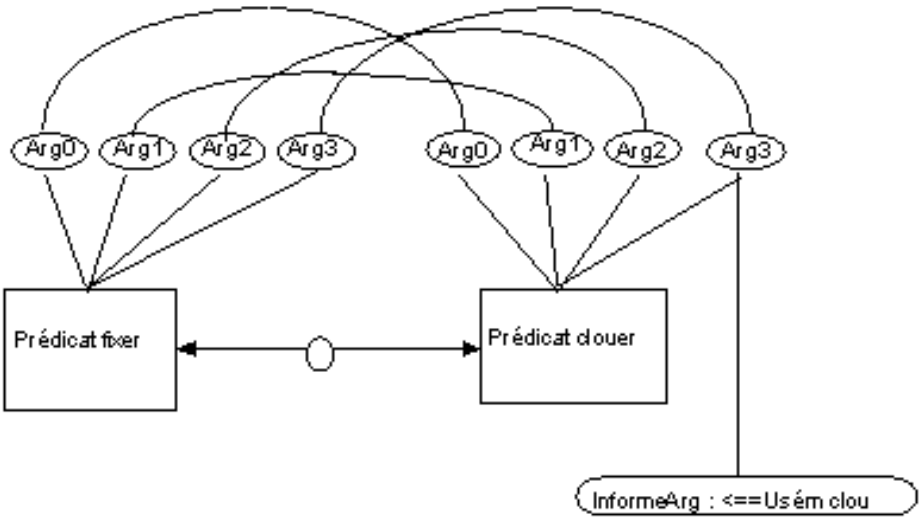
On peut aussi décider que le prédicat *Clouer* a quatre arguments, il est alors décrit comme *Clouer (agent, objet, lieu, moyen).* On décrira alors sa relation avec le prédicat *Fixer* comme suit :

Clouer(agent, objet, lieu, moyen)

<---Spécialisation/Généralisation--->

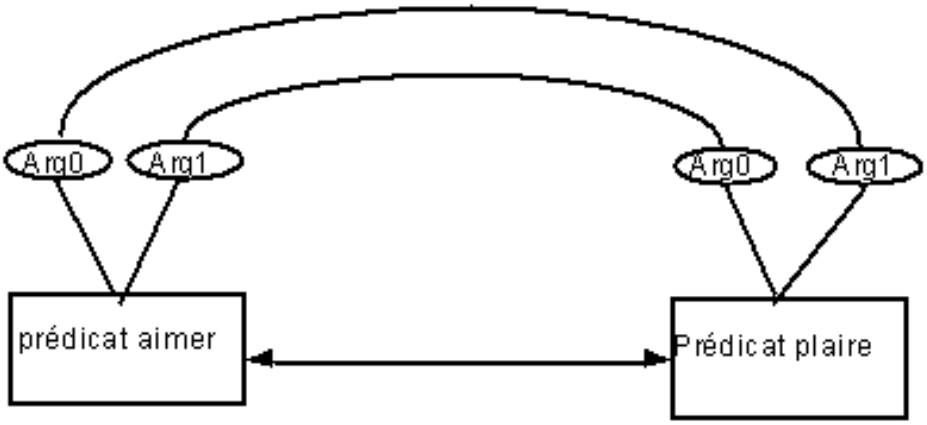
fixer(agent, objet, lieu, moyen)

La description du prédicat force alors la valeur de l'argument 3 de *Clouer* par l'Usem *Clou* (ou le concept `Clou` suivant les choix faits), et dans ce cas les deux Arg3 sont mis en correspondance tout simplement.



Autre exemple de relation entre prédicats

Ex: aimer(affecté, objet) <--- synonyme converse large ---> plaire(agent, affecté)



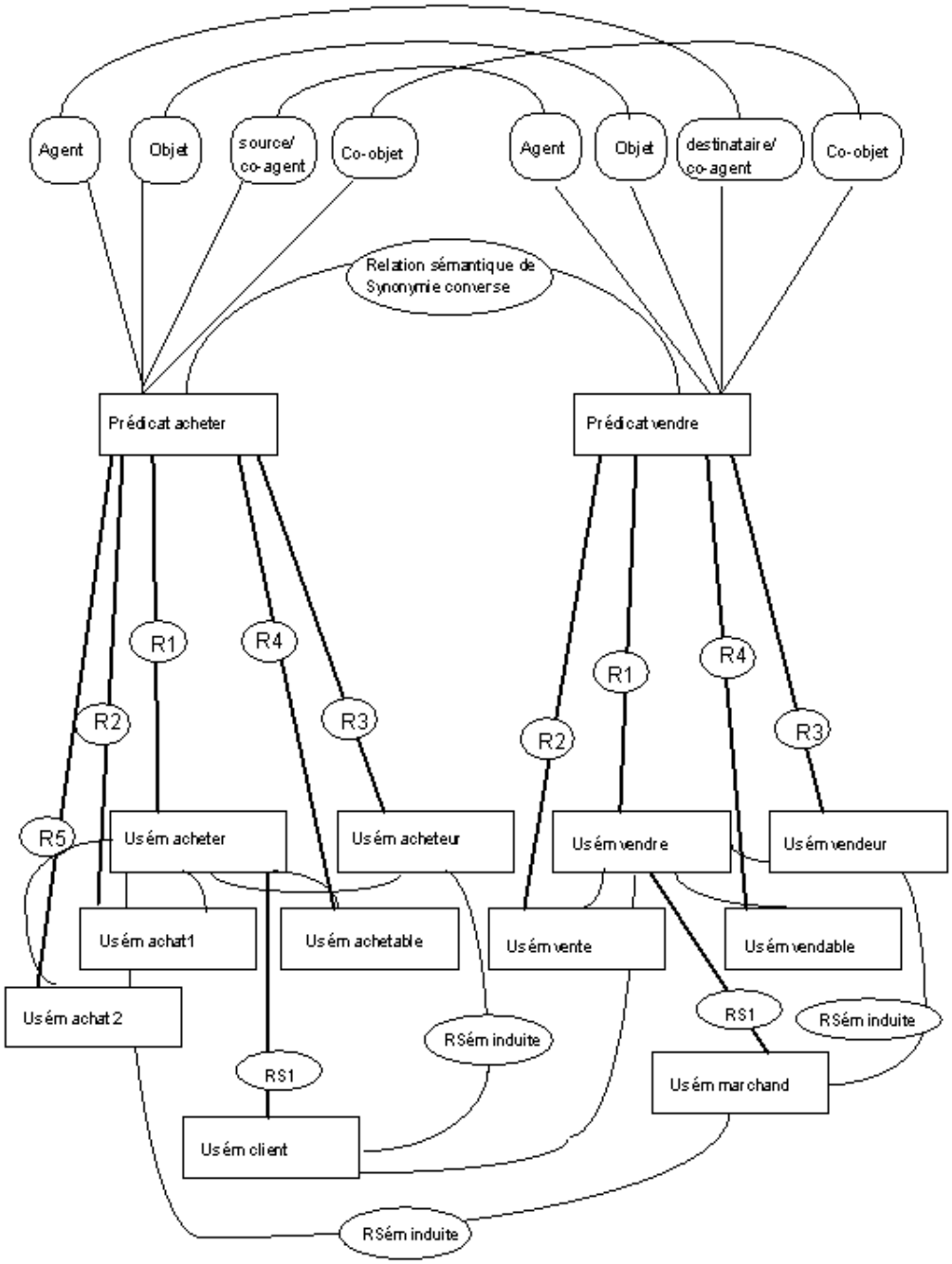
Une partie de la description des relations sémantiques non paradigmatiques décrites plus haut peut être reformulée à partir des relations qui lient le prédicat et les unités sémantiques qui le contiennent. Les relations de dérivation sémantique peuvent par exemple bien souvent s'exprimer comme des relations entre unités partageant un même prédicat et entretenant avec celui-ci des relations différentes. Ces relations entre unités sémantiques sont alors déductibles de celles qui lient les unités sémantiques au prédicat.

Ex : Supposons que l'on ait identifié et décrit les prédicats *vendre (Arg0, Arg1, Arg2, Arg3)* et *acheter(Arg0, Arg1, Arg2, Arg3)* partagés par des unités sémantiques acceptions respectivement de *acheter, acheteur, achat1, achat2, achetable* et *vendre, vendeur, vente, vendable*

Il peut être particulièrement intéressant de mettre en rapport les deux prédicats par une relation qui précise également comment les arguments se correspondent.

Nous illustrerons les possibilités du modèle par le dessin suivant, lequel correspond à certaines

simplifications et à certains choix de codage, mais se veut illustratif du modèle.



R1,R2,R3,R4,R5 sont des associations entre Usem et prédicats qui précisent une certaine modalité d'association

R1 : vedette, pas d'argument concerné

R2 : pas d'argument concerné

R3 : argument 0(agent) concerné et incorporé

R4 : argument 1(objet) concerné et non incorporé, qualité de ce qui peut être cet argument

R5 : argument 1(objet) concerné et incorporé

Ces liens d'associations (Représentations prédicatives) sont explicitement présents.

RS1 est une relation sémantique entre Usem, elle associe une Usem (verbale) et son agent prototypique, elle est explicitement présente.

Les arcs reliant des Unités sémantiques représentent des relations sémantiques (diverses) déductibles des informations explicitement présentes, elles ne sont pas explicitement dans le dictionnaire. Il s'agit des relations de dérivation sémantique d'unités associées au même prédicat, d'une part, et des relations déductibles des relations sémantiques explicitement présentes associées aux informations d'argument.

Il faut bien remarquer qu'il s'agit d'un certain choix de codage et d'organisation des informations de la couche sémantique, qui n'est pas le seul possible. En particulier, on aurait pu préférer coder comme information associée par défaut à l'argument agent le lien qui lie "acheter" à "client" ou bien "vendre" et "marchand" en faisant pointer l'argument vers sa valeur par défaut Usem.

5.3.3. Au niveau du codage et de l'établissement de critères de cohérence

D'un point de vue de méthodologie lexicographique, la structure prédicative peut être suffisamment précise et contraignante pour fournir des indices de proximités sémantiques (par proximités de structure d'arguments) ou même pour définir des classes sémantiques ou, au contraire, pour pouvoir rejeter certaines relations entre unités sémantiques ou entre prédicats. C'est donc une information très intéressante à exploiter au moment du codage et pour la vérification de la cohérence d'une instance de dictionnaire.

En effet, le nombre d'arguments, le rôle sémantique de chacun d'eux, ainsi que les informations sémantiques qui leur sont associées (traits sémantiques en particulier) permettent de déterminer des classes d'entrées prédicatives, et cela, que l'on souhaite coder ces informations au niveau des entrées sémantiques ou au niveau des prédicats. Cette information peut aussi servir à la vérification "intellectuelle" de la cohérence des relations.

Par exemple, une relation de synonymie suppose que les structures prédicatives soient les mêmes ou très peu différentes (la synonymie pure n'existant pas dans la langue, on doit certainement s'accommoder de certaines divergences), et l'écart des structures prédicatives peut donner une "mesure" du degré de synonymie. De même, deux prédicats ayant des structures prédicatives semblables ont très probablement un rapport de sens non nul ; ceci est d'autant plus vrai que le nombre d'arguments est grand.

6. Concept

Une unité sémantique peut être associée à un concept qui est le plus souvent commun à plusieurs unités sémantiques. Il faut alors voir la notion de **concept** comme une généralisation du contenu *objectif* ou *cognitif* d'une unité sémantique décrite d'un point de vue **non-prédicatif** .

Il peut être intéressant de mettre en évidence l**a classe d'équivalence d'unités sémantiques** décrites d'un point de vue non prédicatif liées par une relation de **synonymie**. Cela peut se faire en identifiant la classe par le concept commun auquel seraient rattachées les unités sémantiques qui lui appartiennent.

Un certain nombre de traits sémantiques valués, communs à ces unités sémantiques peuvent donc être affectés au concept. Les unités sémantiques appartenant à cette classe possèdent, par héritage ou explicitement, ces mêmes traits valués et d'autres traits propres (par exemple : niveau de langue et connotation). Ainsi les unités sémantiques de *chien, clebs, clébard, cabot, chien-chien, toutou* peuvent toutes pointer vers un même concept que nous nommerons *chien.* Un tel concept, identifié par le lexicographe à partir d'unités sémantiques est issu du lexique et peut être dit lexical en ce sens qu'il factorise ce que la langue décrite nomme (généralement de plusieurs façons). Une unité sémantique particulière, la moins "marquée" pourra être choisie comme représentante privilégiée du concept auquel elle est associée, elle est alors la "vedette" associée au concept par la Représentation Conceptuelle.

Dans une approche de représentation de connaissances, on pourra également s'appuyer sur des concepts *non lexicaux* (qui n'ont pas d'expression dans la langue autrement que par une périphrase). On peut donc par exemple s'appuyer sur ce niveau conceptuel pour décrire une taxinomie qui comporte des trous lexicaux. Nous reviendrons sur ce sujet lorsque nous parlerons de relations entre concepts.

D'autres concepts qui ne sont pas non plus issus directement du lexique, peuvent également être utilisés pour une représentation de plus en plus abstraite et générale. Ils factorisent ce qu'ont de commun un ensemble de concepts d'un degré d'abstraction moindre.

On obtient ainsi la possibilité d'avoir virtuellement une **représentation sémantique à profondeur et niveau de détail variables**.

Le niveau proprement lexical des unités sémantiques est le plus précis, il permet de rendre compte du sens d'une entrée jusque dans les détails. Ce niveau pointe vers des représentations plus abstraites, et on peut aussi se contenter d'un de ces niveaux abstraits en ne s'intéressant qu'aux informations qui y figurent.

Dans un même dictionnaire GENELEX peuvent ainsi cohabiter (et communiquer par projections d'un niveau vers l'autre) différents niveaux de dictionnaires. Une application donnée pourra donc dériver aisément le dictionnaire d'application contenant l'information du niveau voulu par ses besoins en sémantique.

7. Le niveau de description "conceptuel"

La plupart des informations sémantiques décrites jusqu'à présent se situent principalement dans un contexte de sémantique lexicale où les objets descriptifs émergent de la langue décrite et en restent proches. Ces objets descriptifs ne supposent pas de prise de distance par rapport au lexique par une généralisation et une abstraction très grande s'appuyant sur des primitives. Les outils descriptifs que nous présentons maintenant sont destinés à permettre une telle abstraction.

Il convient de remarquer que le modèle GENELEX n'impose aucunement l'usage de primitives ni le mode de définition de celles-ci : il autorise aussi bien la description de primitives prédéfinies sur lesquelles on projette les unités sémantiques que l'émergence à partir des descriptions lexicales de

primitives plus abstraites. Dans les deux approches, les concepts et les prédicats, ainsi que les informations qui leur sont associées, permettront de rendre compte d'un niveau de description sémantique abstrait, éventuellement orienté *Intelligence Artificielle* .

7.1. Prédicat

A la notion de concept abstrait pour les entités décrites de façon non-prédicative fait pendant la notion de **prédicat généralisé, primitif ou non.**

Ces prédicats sont décrits par le même objet formel que les prédicats lexicaux, leur différence de statut provient d'un attribut `type` valué différemment et du fait qu'ils ne sont pas pointés directement par une Usem.

Un premier niveau de tels prédicats peut être identifié par les classes d'équivalence de prédicats lexicaux en relation de synonymie (plus ou moins stricte suivant le degré de généralité voulu pour le prédicat primitif).

Par exemple, des prédicats lexicaux *acheter* et *acquérir* pourront également pointer vers un même prédicat généralisé (ou même primitif), nommons-le *achat*, et les prédicats lexicaux *vendre, bazarder, liquider, céder* pourront pointer vers une autre généralisation : le prédicat généralisé (ou primitif) *vente*.

Un niveau de généralisation supplémentaire permet de décrire des prédicats en s'appuyant sur d'autres prédicats généralisés ou primitifs d'une plus grande généralité. C'est ainsi que les deux prédicats identifiés ci-dessus pourraient s'exprimer en fonction d'un même prédicat primitif, nommons-le *transaction.*

Les prédicats les plus généraux sont aussi ceux qui comportent le plus d'arguments. En effet, un prédicat plus spécialisé comporte une partie d'information implicite (et peut donc éventuellement absorber ou inclure un ou plusieurs arguments explicites du prédicat général). Cela se doit d'être précisé dans la description d'un prédicat par un autre prédicat plus général. Ainsi, les verbes de mouvement pourront se trouver projetés sur un prédicat primitif de mouvement identifiant différents arguments : provenance, destination, lieu traversé, moyen...

Pour toutes ces raisons, les relations qui associent des prédicats entre eux doivent permettre de préciser les correspondances de structures argumentaires tout comme on le prévoit au niveau des prédicats lexicaux décrits précédemment.

D'autre part, les prédicats primitifs pourront eux aussi être porteurs de traits sémantiques valués.

7.2. Relations au niveau conceptuel

Un certain nombre de relations que nous avons mentionnées comme relations de sémantique lexicale peuvent être utilisées de façon intéressante au niveau des objets conceptuels. Il s'agit en particulier de toutes les relations qui rendent compte de connaissances sur le monde et qui peuvent présenter un certain caractère encyclopédique.

7.2.1. Relation entre concepts

Les relations décrivant des taxinomies et des méronymies (relations partie-tout) entre unités non prédicatives peuvent très bien s'exprimer par des relations entre concepts.

Cela permet en effet de distinguer le niveau des unités sémantiques où ne figurent que les unités issues de la langue, du niveau des concepts-représentation de connaissances où il est possible de trouver des concepts sans lexicalisation qui interviennent dans des hiérarchies. Le problème des trous lexicaux peut donc être géré ainsi sans introduire d'incohérence dans le modèle.

On considérera que les relations entre concepts sont implicitement héritées par les lexicalisations de ceux-ci.

7.2.2. Relation entre prédicats

Nous avons déjà fait mention de relations sémantiques entre prédicats lexicaux. Ces relations peuvent éventuellement s'étendre à tous les prédicats. Cependant, il est préférable de conserver au niveau *"généralisé"* et *non nécessairement lexical* le rôle de représentation de connaissances sur le monde.

On pourra utiliser des relations de généralisation entre prédicats pour définir des prédicats plus généraux à partir de prédicats spécifiques, une ou plusieurs généralisations menant le plus souvent à des prédicats primitifs.

Des relations hiérarchiques ou **taxinomiques** peuvent être définies ainsi entre prédicats.

Des relations de type **partie-tout** peuvent également lier des prédicats (pas nécessairement lexicaux). Il faut préciser ici que le modèle permet d'exprimer d'une façon plus précise et potentiellement plus riche la présupposition, l'implication ou la référence à des scénarios. Cependant, pour une utilisation minimale, les relations partie-tout entre prédicats peuvent être une façon *(sans doute légèrement détournée)* de coder des relations de présupposition ou d'implication et des ensembles de prédicats qui formeraient une sorte de noyau de scénario.

7.2.3. Relation entre prédicats et concepts

Ici encore, on pourra reporter certaines relations du niveau lexical (entre unités sémantiques) au niveau du concept et du prédicat lexical ou primitif.

Des relations plus générales peuvent également relier des prédicats et des concepts.

Un prédicat peut être lié à l'un de ses circonstants typiques (non décrits dans la structure prédicative) par une relation entre prédicat et concept. En effet, on peut préférer faire porter cette connaissance sur le concept qui peut se lexicaliser de différentes façons plutôt que sur une Usem particulière ; le lien vers le concept est alors "porteur" de liens implicites vers les différentes Usem associées au concept. C'est ainsi que les concepts associés par exemple à un instrument typique, un lieu typique, un moyen typique pourront être associés aux prédicats dont ils sont circonstants typiques.

Il faudra bien se doter, au niveau d'une instance de dictionnaire GENELEX, d'une stratégie fine permettant de coder de façon homogène les informations concernant les arguments et/ou les circonstants typiques d'un prédicat. En effet, les prédicats décrivent leurs arguments de façon assez

précise, et, en particulier, peuvent leur associer des concepts. D'autre part, les arguments du prédicat peuvent être plus nombreux que les positions appelées en syntaxe ; leur réalisation en syntaxe est alors décrite depuis la sémantique. Il reste donc à l'équipe lexicographique à faire les choix et à définir sa stratégie de codage afin d'identifier les "circonstants" syntaxiques qui sont des arguments du prédicat (et donc décrits en tant que tels) et à décrire éventuellement, par ailleurs, par des relations sémantiques particulières entre prédicats et concepts ou entre Unités sémantiques, les "vrais" circonstants.

7.3. Apport de ce niveau de représentation

Le niveau de représentation conceptuelle que constituent les concepts et les prédicats non-lexicaux permet d'exprimer des connaissances plus ou moins générales et plus ou moins indépendantes de la langue suivant les choix faits pour un dictionnaire donné et à un moment donné.

Le niveau de représentation proche de la langue (unité sémantique, prédicats lexicaux) est en quelque sorte plongé ou projeté dans le niveau conceptuel. La projection est plus ou moins réductrice suivant le degré d'abstraction choisi pour les *primitives.*

D'autre part, un dictionnaire peut comporter à un certain moment un niveau de description comportant peu de généralisations, et s'enrichir ultérieurement d'un niveau de primitives supplémentaires qui viendra se greffer sur l'existant.

De même, un dictionnaire peut comporter très peu d'informations au niveau des unités sémantiques et des prédicats lexicaux, ceux-ci servant alors presque exclusivement à atteindre un niveau conceptuel abstrait renseigné de façon satisfaisante à un moment donné. Dans une phase ultérieure, le niveau lexical peut être enrichi sans que soit remise en cause la description du niveau conceptuel.

Cette structuration des informations permet donc à la fois :

- d'enrichir et de faire évoluer un dictionnaire GENELEX sans être obligé de remettre en cause l'information existante.

- d'obtenir virtuellement une *profondeur variable* en ne considérant que les informations présentes à un certain niveau ou pouvant être projetées d'un autre niveau sur le niveau lexical.

Ex : On peut désirer ne travailler qu'au niveau lexical (unités sémantiques et prédicats lexicaux). On ne s'intéressera donc qu'aux informations de ce niveau et à celles héritées des niveaux conceptuels dans lesquels elles sont plongées. On aura alors le niveau de description le plus fin du dictionnaire.

On peut aussi décider de ne s'intéresser qu'aux traits sémantiques dits *objectifs* et négliger les traits évaluatifs, de connotation, de niveau de langue. La description sera alors plus pauvre tout en restant au niveau "lexical".

On peut au contraire ne s'intéresser qu'aux informations formulées en terme de primitives et totalement ignorer les informations du niveau lexical. La projection est immédiate.

La structuration de l'information en niveaux de plus en plus abstraits permet donc d'obtenir d'une façon assez simple l'expression d'une profondeur variable et de répondre à des besoins très

différents. Des applications d'indexation ou d'interrogation de bases de données s'appuieront probablement sur un niveau conceptuel, quant aux applications de traduction ou génération, le niveau "lexical" leur sera certainement très utile.

7.4. Elaboration de ce niveau

Nous avons déjà fait mention à plusieurs reprises des différentes approches de méthodologie lexicographique possibles pour la description du lexique. Ces deux approches consistent à :

- Faire émerger les éléments de description de l'observation de la langue, et donc ne s'attacher à remplir le niveau "conceptuel" qu'après avoir décrit finement le lexique, afin d'en distinguer les généralités, puis de les décrire dans les objets formels proposés par le modèle : prédicats, concepts, relations entre ces objets.

- S'appuyer sur des éléments de description prédéfinis.

Le niveau dit *conceptuel* permet lui aussi les deux approches. On peut souhaiter identifier les primitives à partir de l'observation des objets d'un niveau moins abstrait de la langue et de généralisations faites pas à pas. On peut aussi avoir une batterie de primitives (tant concepts que prédicats et relations, et même traits) que l'on représente alors dans le modèle, et vers lesquelles on projette les unités sémantiques.

D - Correspondance syntaxe sémantique

1. La notion de prédicat : rappel

La notion de prédicat est essentielle au modèle sémantique et la correspondance syntaxe sémantique s'appuie sur cet objet. Rappelons ici à quoi il correspond.

Un prédicat exprime, dans une situation donnée, la relation (par exemple relation de *don*) liant ses arguments. Il est partiellement défini par sa structure argumentaire qui est en quelque sorte sa "signature". Chaque argument du prédicat joue un rôle sémantique au sein de cette structure. Un prédicat peut contraindre sémantiquement ses arguments : leur imposer des valeurs de traits, l'appartenance à une classe sémantique...

Au niveau du modèle, les prédicats sont identifiés en tant que tels, et sont des objets à part entière. Un prédicat peut être partagé par plusieurs unités sémantiques.

La mise en correspondance d'une unité sémantique avec une unité syntaxique est donc, en terme de prédicats, la mise en correspondance d'une construction syntaxique avec une structure prédicative, lorque l'unité sémantique est décrite comme prédicative.

2. Position syntaxique/Argument prédicatif

Une unité syntaxique est caractérisée par une Description de Base (DB), laquelle associe un SELF et une Construction. La Construction comporte des positions occupées par des syntagmes auxquels sont attachées différentes informations. Aux DB du niveau syntaxique feront pendant, au niveau sémantique, des informations concernant le prédicat et la structure prédicative associée à l'unité

sémantique quand celle-ci en a effectivement une (généralement cela correspondra aux cas où l'unité décrite a une fonction TETE, par exemple un verbe tête de la structure de complémentation décrite par la Construction).

Ces deux structures doivent être mises en correspondance et le modèle doit permettre minimalement d'exprimer sur les arguments des prédicats des connaissances concernant leur origine de surface au niveau syntaxique.

La correspondance n'est bien évidemment pas systématiquement une bijection qui associerait une position à un argument du prédicat au niveau sémantique, même si elle peut parfois prendre une telle forme (c'est le cas pour de nombreuses constructions verbales par exemple).

La relation de mise en correspondance Usyn-Usem doit donc porter des informations permettant d'identifier au niveau syntaxique, dans la construction de base qui caractérise l'unité syntaxique, les occupants de position qui correspondront aux arguments de l'entrée prédicative.

Elle doit aussi comporter les informations permettant de filtrer dans la famille de réalisations possibles de structures syntaxiques associées à la Description de Base (Construction + SELF portant les particularités de l'entrée pour la construction) celles auxquelles on veut effectivement associer l'Unité sémantique.

Dans la correspondance avec une unité sémantique donnée, l'optionnalité des positions telle qu'elle est décrite en syntaxe sur la Construction a parfois besoin d'être contrainte. Il faut alors pouvoir gérer un écart sur le caractère optionnel de certaines positions de la Construction en spécifiant que telle position optionnelle au niveau de l'Usyn devient obligatoire ou au contraire interdite pour telle acception. Enfin, si une position demeure optionnelle pour une acception donnée, on doit pouvoir décrire les informations sémantiques affectées par défaut à un argument lorsque la position optionnelle reste inoccupée (phénomène de complément *interne*, ou d'emploi absolu de verbes transitifs, où le contexte syntaxique peut forcer la sémantique de l'argument non exprimé en surface)

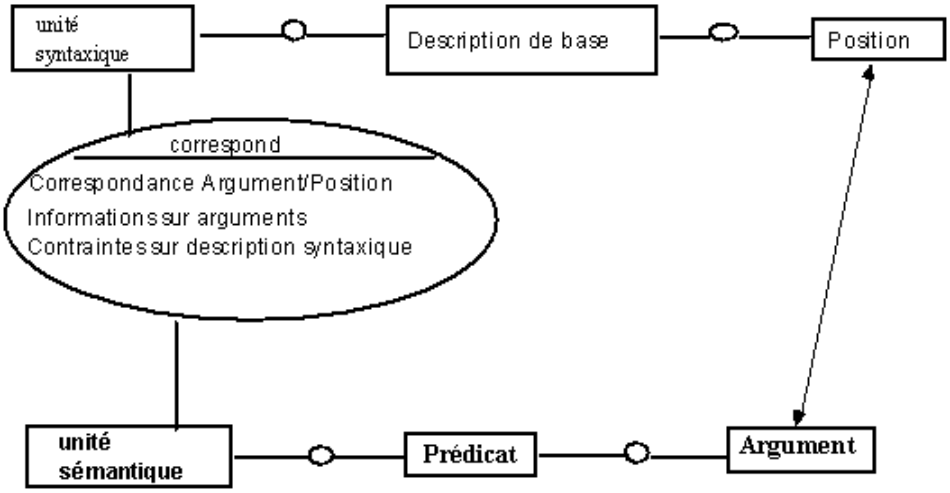
Ex : Pierre nage (la brasse)

nager (arg0, arg1 [defaut= nage])

Pierre mange (une pomme)

manger(arg0, arg1[[defaut : `comestible : +]` *])*

La correspondance Usyn-Usem peut être représentée de façon simplifiée comme suit :



3. Contraintes sur la description de base

Les DB sont associées à une construction ayant plusieurs réalisations de surface qui, gr‰ce au jeu des optionnalités et des alternatives concernant les occupants de position, ne diffèrent pas seulement par leur lexicalisation. On doit pouvoir associer à une unité syntaxique (caractérisée entre autres par sa DB) une unité sémantique moyennant des filtrages sur l'ensemble des réalisations possibles de la construction que représente la DB, et donc sur les instanciations de positions de la Construction et sur les réalisations de SELF.

3. 1 Portée du filtrage

Ces restrictions par filtrage peuvent concerner :

-- Un élément de la distribution d'une position d'étiquette syntagmatique donnée ;

par exemple : on peut vouloir sélectionner l'occupant SN lors de l'association à une unité sémantique d'une unité syntaxique caractérisée par une DB comportant une position pouvant être occupée par un `SN` ou un `P[mode : inf]` ou un `P[sscat : complétive][Conj : que]`. Ce filtrage s'exprime en inhibant les différents éléments de la distribution qui ne sont pas autorisés par la correspondance, c'est-à-dire incompatibles avec le sens attribué à l'Usem pointée.

-- Certains traits que l'on souhaite rajouter à des éléments de la distribution qui ne sont pas inhibés mais simplement plus contraints qu'ils ne le sont dans la description.

par exemple : on peut vouloir restreindre au mode subjonctif un occupant de position `P[Conj : que][sscat : complétive]` dont le mode n'est pas précisé par la DB : un trait supplémentaire est alors "rajouté" par la correspondance syntaxe-sémantique.

-- L'optionnalité d'une position :

Par exemple, on peut vouloir rendre **obligatoire** une position optionnelle de la DB pour pouvoir associer une unité sémantique à l'unité syntaxique décrite par la DB.

Ex :

manger

```
CatGram : V

db : Self cb

cb : P0 IntervConst (P1)

P0 : SN

Self : IntervConst : V

P1 : SN
```

Supposons que *manger* reçoive cette description unique, sans contraintes dénotationnelles, qui recouvre le sens courant et le sens métaphorique: "*manger les virgules ou manger les mots* ".

On veut imposer dans la correspondance vers le sens abstrait la réalisation en surface de la position P1, c'est-à-dire que l'on rend obligatoire pour un sens donné la position qui est décrite comme optionnelle dans la description.

-- L'interprétation sémantique de la réalisation d'une position :

Ex :

croire

```
CatGram : V

db : Self cb

cb : P0 IntervConst P1

P0 : SN

Self : IntervConst : V

P1 : SN

P[Conj: que]

[sscat : complétive]

P[mode : infinitif]
```

peut être associé à une Unité sémantique pour le sens :

croire quelqu'un, prédicat à deux arguments

avec le filtrage sur le syntagme (restriction sur la distribution de la position) et sur trait sémantique

comme suit :

```
P1 : SN[humain : +]
```

et à une autre pour le sens :

considérer comme vrai, prédicat à deux arguments

avec le filtrage : (à la fois sur la distribution et sur l'interprétation sémantique associée à la position)

```
P1 : SN[humain : -] [abstrait : +]

                    P[Conj : que][sscat : complétive]
```

et probablement encore d'autres sens filtrant d'autres instanciations de position.

Remarque 1 :

Comme nous le faisions remarquer plus haut, le choix de codage en syntaxe aura des répercussions sur la correspondance syntaxe-sémantique.

Une grande factorisation au niveau des unités syntaxiques aura pour conséquence le fait que l'on aura besoin de beaucoup d'informations de filtrage au moment du passage à la sémantique ; inversement, on peut envisager d'avoir besoin de peu d'informations de filtrage dans le cas d'éclatements nombreux au niveau syntaxique, en particulier lorsque des informations fines concernant les rôles thématiques et les contraintes dénotationnelles seront utilisées.

Il ne faut surtout pas oublier que la correspondance syntaxe sémantique se fait en s'appuyant sur la description de base de l'Usyn concernée par la correspondance. En effet, les choix faits en syntaxe ont été de décrire le comportement syntaxique d'une Usyn principalement par le contenu descriptif de la description dite "de base" (DB), mais aussi par les descriptions qui sont associées à l'Usyn en tant que transformées de la DB ou de l'une des descriptions transformées. On considère en quelque sorte qu'il s'agit de différentes expressions syntaxiques d'un même comportement syntaxique "profond", et que les modifications de sens associées aux différents contextes syntaxiques que représentent les différentes Description sont minimes quant au sens de l'unité décrite et concernent plutôt, par exemple, une thématisation différente. Les transformations syntaxiques font donc partie exclusivement des informations de la couche syntaxique, et la correspondance syntaxe-sémantique ne les "voit" pas. Cela signifie en particulier que l'on ne peut pas filtrer lors de la correspondance sur une description transformée, que l'on ne peut pas non plus, de la sémantique, bloquer certaines transformations. La conséquence de ces choix est que cela fournit en quelque sorte un critère d'éclatement en différentes Usyn : si l'on a besoin, pour certains sens associés, d'intervenir sur des transformations (les restreindre ou les bloquer), c'est sans doute qu'il y

a deux comportements syntaxiques sous-jacents, et donc qu'il est préférable d'identifier deux Usyn.

Pour mémoire, rappelons que deux Usyn peuvent également être en relation de transformation syntaxique. Elles peuvent, si elles proviennent d'une même UM, être associées à une même Usem.

3. 2 Filtrage par des traits

3. 2. 1 Filtrage par des traits de la syntaxe

On s'appuie, pour exprimer le filtrage des constructions, sur les mêmes objets descriptifs et le **même langage** que celui employé pour décrire **les occupants de positions**, c'est-à-dire des syntagmes et un ensemble de traits restrictifs comme au niveau syntaxique (cf. Rapport sur la couche syntaxique). La correspondance s'appuie sur les objets de la description de base et rajoute des informations supplémentaires qui jouent le rôle de filtre : inhibition ou présence obligatoire d'une position optionnelle dans la construction, inhibition d'un syntagme de la distribution d'une position ou ajout de traits supplémentaires, traits de la syntaxe ou traits sémantiques se projetant sur l'interprétation sémantique du syntagme.

On se donne pour filtrer les occupants de positions un certain langage qui se base sur des éléments descriptifs du niveau syntaxique, en particulier les traits. Le filtrage peut également se faire sur des traits sémantiques, dont la présence au niveau syntaxique (par le biais de traits libres associés à des traits de la sémantique) n'est en aucun cas obligatoire, et suivant qu'ils sont présents dès la syntaxe ou non, le filtrage aura un sens un peu différent.

Les traits syntaxiques utilisés dans le filtrage sont à comprendre comme l'ajout de contraintes sur la description de base, et donc la définition d'un sous-ensemble de réalisations par rapport aux réalisations que regroupe la description de base.

3. 2. 2 Filtrage par des traits sémantiques

On peut aussi filtrer par des traits sémantiques valués. Il faut ici préciser le rôle que l'on veut faire jouer à ces traits sémantiques valués car plusieurs mécanismes sont possibles dans un cadre syntaxique instancié et elles offrent chacune des possibilités d'expression intéressantes.

3. 2. 2. 1 Filtrage par vérification-attestation de la présence de l'information

Une utilisation des traits sémantiques lors de la correspondance consiste à exiger la **présence explicite du trait valué** dans la représentation sémantique associée au syntagme décrit ; dans ce cas, le filtrage par les traits valués signifie une vérification à la fois de la présence du trait et de la valeur attendue.

L'absence du trait attendu, tout comme la présence du trait (monovalué) valué différemment (pour ce qui est des traits de la sémantique) et par une valeur non compatible (la hiérarchie des traits permet de structurer certaines valeurs et de calculer des compatibilités), signifie que l'unité sémantique pointée par la correspondance n'est pas l'un des sens devant être associé à la structure syntaxique dont les positions sont instanciées d'une façon non satisfaisante. Cette acception est donc incompatible avec l'interprétation sémantique de certaines instanciations du contexte syntaxique

décrit par la DB. Le trait sémantique valué apporte une information utile, en analyse, à la désambigu•sation.

La contrainte de filtrage, dans cette interprétation, n'enrichit en aucun cas la représentation sémantique de l'argument, elle joue exclusivement un rôle de vérification. Si on utilise déjà les traits sémantiques au niveau de la syntaxe, le filtrage devra être compatible avec ce qu'autorise l'unité syntaxique. Dans le cas contraire, le filtrage portera a posteriori sur la représentation sémantique de l'occupant de la position filtrée. On peut identifier ces traits comme des "traits contraints".

Ces traits s'exprimeront dans le modèle par des AjouteTraitSem au statut : FILTRE (voir plus loin le Manuel utilisateur).

3. 2. 2. 2 Filtrage par vérification-enrichissement de l'information

Une autre utilisation possible est celle qui consiste à vérifier la **compatibilité** de l'information de filtrage avec la représentation sémantique associée au syntagme décrit, ce qui signifie qu'un trait présent et valué différemment devrait permettre d'exclure le sens associé à l'Usem pointée, mais que l'absence d'un trait autorise la correspondance vers cette Usem, et que l'argument associé à la position contrainte par le trait a dans ce cas sa représentation sémantique enrichie de ce trait.

La contrainte exprimée joue alors un rôle à la fois de **vérification de compatibilité** et **d'enrichissement** de la représentation sémantique. La correspondance Usyn-Usem force "si nécessaire" en quelque sorte la représentation sémantique des arguments. On pourra parler dans ce cas de "traits projetés".

Ces traits s'exprimeront dans le modèle par un AjouteTraitSem au statut : FILTRE_AJOUTE (voir plus loin le Manuel utilisateur).

3. 2. 2. 3 Filtrage par enrichissement obligatoire de l'information

Une troisième possibilité est celle qui consiste à "forcer" le trait sémantique quelle que soit sa compatibilité avec la représentation sémantique du syntagme. On ne peut pas dans ce cas parler de filtrage mais plutôt d'enrichissement forcé. Un exemple d'utilisation de cette possibilité pourrait être, sur le verbe *manger* (sens courant) de projeter sur la position `sujet` un trait sémantique [animé = +] qui forcerait l'interprétation sémantique de la position à porter ce trait, ce qui correspondrait par exemple à l'interprétation de la phrase : *Ma voiture boit de l'essence.*

Ces traits de filtrage sémantique sur la syntaxe s'exprimeront dans le modèle par un AjouteTraitSem au statut : FORCE (voir plus loin le Manuel utilisateur).

Remarque : Il est intéressant de noter que ces traits sémantiques utilisés lors de la correspondance syntaxe-sémantique permettent de filtrer sur certaines interprétations sémantiques du contexte syntaxique et d'enrichir une représentation sémantique du contexte, indépendamment du fait que l'unité en correspondance soit décrite comme prédicative ou non, et donc indépendamment du fait que les positions correspondent ou non à des arguments de la sémantique. Cela met en évidence l'intérêt de la description fine du contexte syntaxique et d'un filtrage fin lors de la correspondance indépendamment des représentations prédicatives que l'on peut associer aux Usem.

4. Réalisation des arguments

4. 1. Correspondance argument position

Dans un grand nombre de cas, la correspondance Position syntaxique-Argument sémantique se fait d'une façon simple : le SELF de la DB caractéristique de l'Usyn correspond à l'élément décrit (nom, verbe, adjectif...) qui occupe la fonction TETE, et ses compléments essentiels se trouvent décrits dans la Construction associée. Préciser la réalisation des arguments consiste alors le plus souvent à associer à chacun d'eux une position. L'Unité sémantique, qui représente en quelque sorte le sens associé à SELF pointe vers le prédicat suivant une certaine modalité qui contraint éventuellement l'instanciation de la structure d'arguments associée.

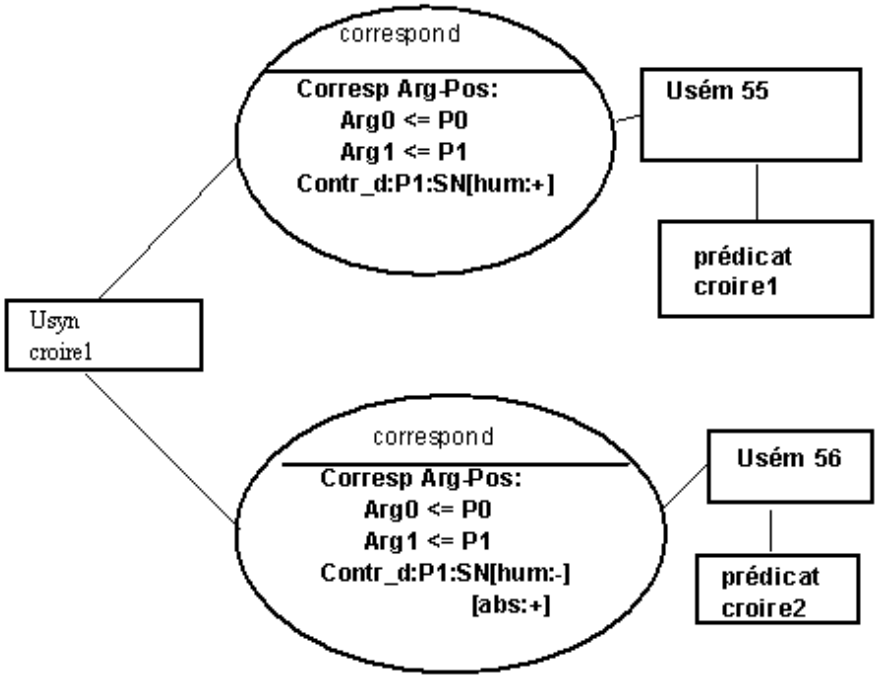
Il faut cependant remarquer que la correspondance argument-position ne concerne que les unités décrites comme prédicatives, et qu'aucune contrainte n'impose que tous les arguments sémantiques soient décrits au niveau syntaxique (Ex : les arguments flottants) ni que toutes les positions de la syntaxe pointent vers des arguments. La grammaire générale se chargera de construire l'interprétation globale en fonction de ses choix, et en particulier de préciser le rôle joué par l'interprétation d'une position qui n'est pas associée à un argument.

Ex :

croire

```
CatGram : V

db : Self cb

cb : P0 IntervConst P1

P0 : SN

Self : IntervConst : V

P1 : SN

PRO

P[Conj : que]

[sscat : complétive]

P[mode : infinitif]
```

sera associé à plusieurs Usem prédicatives, moyennant différents filtrages, mais des correspondances argument- position éventuellement semblables :



Remarque : Contr_d représente ici l'ensemble des informations qui contraignent la Description syntaxique (ContraintDescription) dans le modèle.

4. 2. Argument flottant

Une unité sémantique prédicative sera associée à un prédicat lexical qui comportera un ensemble d'arguments. Ces arguments džment caractérisés en termes de traits, rôles sémantiques (et peut-être parfois valeurs par défaut) est une des caractéristiques du prédicat lexical, sa "signature". Il correspond à la **structure maximale saturée** du prédicat.

Certains arguments du prédicat sémantique ne font pas partie en surface de la structure syntaxique de référence associée car il s'agit de compléments "adjoints" ou "circonstants" que l'on ne souhaite pas nécessairement décrire au niveau de l'Usyn comme compléments essentiels. On peut cependant décrire leur réalisation syntaxique. Ils sont décrits comme étant des **arguments flottants**, présents en sémantique et dont la réalisation en syntaxe sera précisée depuis la sémantique.

Un argument flottant pointe donc vers une position de la syntaxe entièrement décrite (fonction, rôles thématiques, distribution...), ainsi qu'un CheminSyntagme qui précise le niveau où s'insère cette position "flottante". L'absence de Chemin_syntagme signifie simplement que la position flottante s'insère au plus haut niveau (de la CB, SB, ou de l'Usyn appelante suivant la valeur prise par l'attribut "**portée**"). Dans tous les cas, on ne précise pas où s'insère la position "flottante" dans la liste de positions ; en effet, une position associée à un argument flottant est plutôt de l'ordre des adjoints ou circonstants, et on ne souhaite pas préciser un ordre canonique sur ces éléments. La syntaxe générale (hors dictionnaire) saura décrire les différents points d'insertion pour de telles "positions flottantes".

Plusieurs unités sémantiques peuvent partager un même prédicat lexical. Ces unités sémantiques proviennent d'unités syntaxiques caractérisées par différentes DB qui ne comportent pas nécessairement le même nombre de positions et les mêmes optionnalités acceptées par la

correspondance Usyn-Usem.

4.3 Valeurs par défaut

Il peut donc arriver qu'une construction envisagée ne représente pas en surface (pas même comme compléments adjoints) ce qui correspond à un ou plusieurs des arguments d'un prédicat que cependant on veut lui associer au niveau sémantique. Cette situation peut provenir également du cas où une position est optionnelle (les contraintes de filtrage n'exigent pas la présence de cette position pour la correspondance mise en jeu), il s'agit alors de préciser ce qu'il advient d'un argument qui lui correspondrait.

Dans un pareil cas, un argument "absent" en surface peut avoir une **valeur ou des traits valués par défaut ;** ces informations par défaut ou implicites ne dépendent pas dans la majorité des cas de la construction caractéristique de l'unité syntaxique mais du prédicat, les valeurs par défaut portées par le prédicat sont donc généralement suffisantes.

Remarque :

Il arrive qu'elles dépendent du couple en correspondance et qu'elles soient plus précises que les valeurs par défaut liées au prédicat. On fait alors porter l'information par défaut sur la correspondance Usyn-Usem.

Ex :

Max fume

Max fume la pipe

Max fume le jambon

Max fume le champ de son voisin

fumer VB a une Usyn caractérisée par

```
db : P0 IntervConst (P1)

P0 : SN

IntervConst : V

P1 : SN
```

(on a choisi pour ce codage de ne pas utiliser de contraintes dénotationnelles et cette DB rend donc compte des quatre contextes précédents)

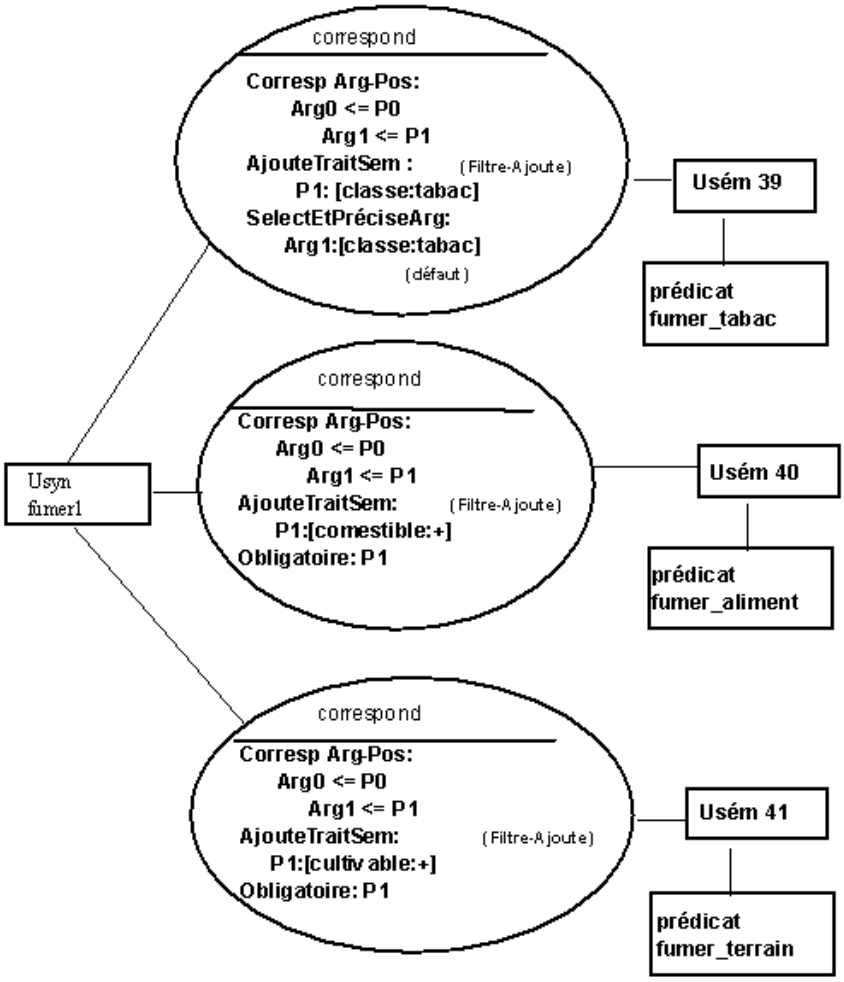
On veut associer à cette Usyn plusieurs Usem liées chacune à un prédicat :

fumer_tabac(arg0, arg1)

fumer_aliment(arg0, arg1),

fumer_terrain(arg0, arg1)

Seul un des sens supporte l'optionnalité de P1. La valeur par défaut n'est alors pas différente de celle issue de la "signature" du prédicat, et ne figure donc pas dans la correspondance.



Remarque : On fait ici référence au trait de classe sémantique, tout comme il y est fait référence dans le rapport syntaxe. La liste des valeurs possibles pour ce trait n'est pas imposée par le modèle.

A noter que les informations de traits sémantiques valués peuvent être portées par des AjouteTraitSem ou par des SelectEtPréciseArg. Elles n'ont pas tout à fait le même statut : on ne peut préciser FILTRE, FILTRE-AJOUTE ou FORCE que sur les AjouteTraitSem qui sont plus orientés "filtrage-enrichissement" sémantique du contexte syntaxique associé à l'Usyn. On ne précisera les valeurs par défaut que par des SelectEtPréciseArg lesquels peuvent non seulement porter une information de trait sémantique valué, mais aussi de concept, de prédicat ou même d'Usém

Ex :

Le chat boit du lait

Max boit (ambigu)

Max boit tout son salaire

boire (Verbe) a une Usyn caractérisée par

```
cb : P0 IntervConst (P1)

P0 : SN

IntervConst : V

P1 : SN
```

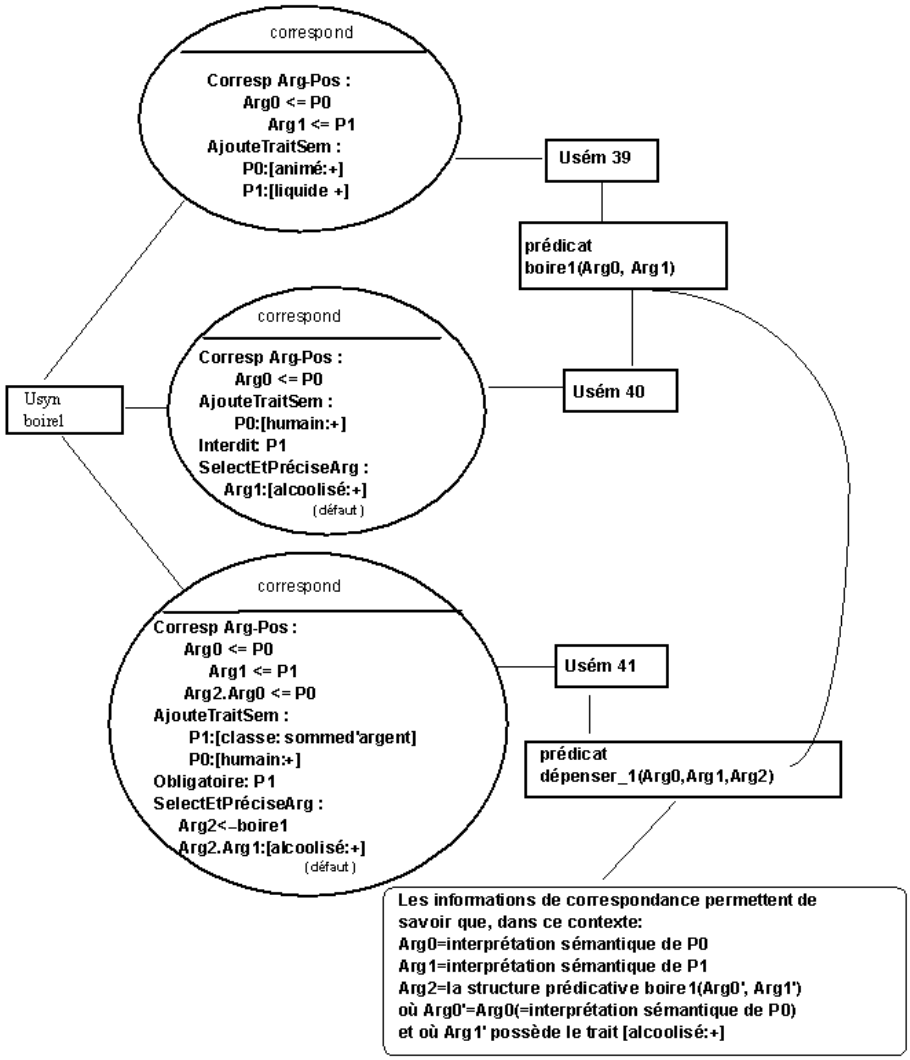
(on a choisi pour ce codage de ne pas utiliser de contraintes dénotationnelles et cette DB rend compte des trois contextes d'occurrence)

> On veut associer à cette Usyn trois unités sémantiques dont deux partagent le même prédicat :

boire1(Arg0 : [animé : +], Arg1 : [liquide +]).

La correspondance précise des restrictions plus grandes sur les arguments pour l'une des correspondances, avec une valeur par défaut sur la position optionnelle P1.

Le sens *dépenser (son argent) en boisson* introduit une correspondance argument-position non triviale et nous amène à élargir le langage du calcul d'arguments : lorsque l'un des arguments au niveau le plus haut introduit un prédicat (ici dépenser_1(Arg0 : animé : +, Arg1, Arg2)), on peut préciser par un SelectEtPreciseArg le prédicat "contenu" par la correspondance, et qui force un argument du prédicat du plus haut niveau, par exemple donner pour Arg2 du prédicat associé un prédicat dont les arguments ont leur réalisation précisée également par une correspondance argument-position. (Voir ci-dessous le point où les cas complexes sont abordés plus précisément.)



Remarque : Arg2.Arg0<=P0 signifie que l'argument 2 de dépenser_1 (prédicat associé à l'unité sémantique au premier niveau) a pour premier argument ce qui est associé à P0 en syntaxe. Le deuxième argument n'a pas de correspondant en surface. Par défaut, il hérite cependant du trait [alcoolisé : +], par le chemin : Arg2.Arg1.

5. Gestion de l’optionnalité

Une Usyn décrit une famille de constructions syntaxiques, et comporte éventuellement des Positions dont la réalisation est **optionnelle**. Au moment de la correspondance avec le niveau sémantique, il faut pouvoir préciser que telle acception rend **interdite** ou **obligatoire** telle ou telle Position optionnelle dans la Construction.

Le cas échéant, on précisera dans la correspondance les **valeurs par défaut** de positions dont l'optionnalité est maintenue au niveau du passage : valeur de l'argument issu d'une position optionnelle lorsque celle-ci est non réalisée, si la valeur par défaut liée à cette construction est différente de la valeur par défaut propre au prédicat, ou si celui-ci n'en a pas. Ces valeurs par défaut seront exprimées par un SelectEtPreciseArg, elles peuvent consister en traits sémantiques valués, concept, prédicat, ou Usem.

Ex capable AJ

Cet individu est capable de mentir pour arriver à ses fins

de mensonge pour que tu viennes

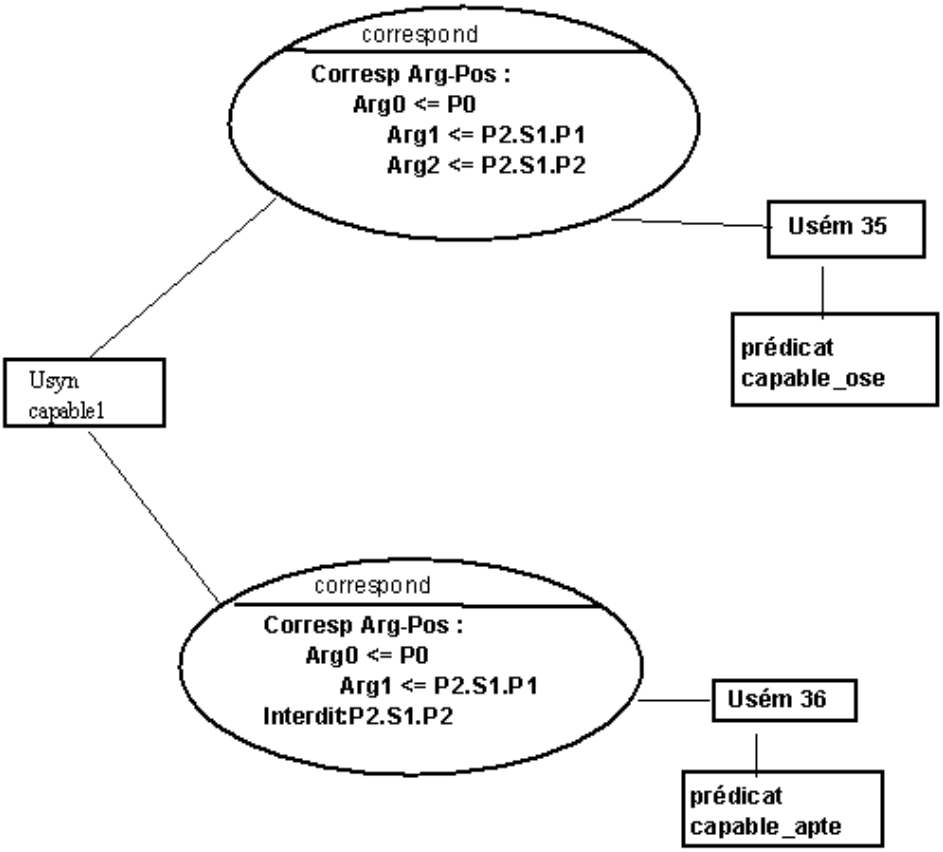
pour une récompense

Cet individu est capable de voler

Cet enfant est capable de comprendre

```
db : P0 P1 P2

P0 : SN[animé +]

P1 : V[Lex : être]

P2 : SADJ : P0 IntervConst P1 (P2)

P0 : SADV

IntervConst : ADJ[Lex : SELF]

P1 : SP[Prep : de]

P[mode : infinitif]

[Prep : de]

P2 : P[mode : infinitif]

[Prep : pour]

SP[Prep : pour]

P[mode : subjonctif]

[Conj : pour que]
```

En supposant que l'on veuille lui associer deux unités sémantiques correspondant aux prédicats : *capable_ose (Arg0, Arg1, Arg2)* et *capable_apte(Arg0, Arg1)* prédicats respectivement à trois et deux arguments, on aurait alors :



Remarque :

La description de positions par une DB (arborescence) suppose que l'on puisse décrire des chemins à travers ces arborescences afin d'associer une position d'un niveau quelconque à un argument. C'est le sens qu'il faut donner à P1.S1.P2 : position P2 de la liste de positions associée au premier syntagme de la position P1. Il s'agit ici d'une représentation conventionnelle.

Le modèle formel s'appuiera sur les éléments descriptifs de la syntaxe, et en particulier sur les CheminPosition et CheminSyntagme de cette couche.

6. Quelques cas plus complexes

Il est des cas où l'on souhaite exprimer des correspondances encore plus complexes en termes de structure d'arguments.

6.1. Unité sémantique contenant, pour l'un de ses arguments, un prédicat et un (ou plusieurs) de ses arguments.

Considérons l'adjectif *meilleur* (comparatif).

Ni le modèle morphologique ni le modèle syntaxique ne le mettent en relation avec *bon* . Il est donc sans doute souhaitable de pouvoir établir qu'au niveau sémantique, l'unité sémantique *meilleur* contient à la fois le sens *plus* et le sens *bon* .

On aimerait éventuellement pouvoir faire référence au prédicat de supériorité

plus(Arg0, Arg1)

(où les deux arguments peuvent bien sžr être eux-mêmes structurés sous forme de prédicat-arguments)

afin d'obtenir des représentations homogènes des structures comparatives, qu'elles soient régulières ou non et il semble intéressant de pouvoir aussi dire que les deux arguments ont pour prédicat *bon* (en fait il doit y avoir autant d'Usem associées à "*meilleur"* comparatif que de "*bon"* supportant le comparatif).

On doit pouvoir exprimer cela dans la correspondance Usyn-Usem, si l'on prend le parti de décrire *meilleur,* au niveau prédicatif, comme *plus(bon(Arg0),bon(Arg1))* . On veut donc "donner" à l'unité sémantique les informations nécessaires aux calculs d'argument sur *plus* et, en tant qu'argument de *plus,* sur *bon* également. L'Usem *meilleur* est donc rattachée directement au prédicat *plus* et indirectement au prédicat *bon.* Cette correspondance est faite au niveau sémantique et non au niveau morphologique ou syntaxique.

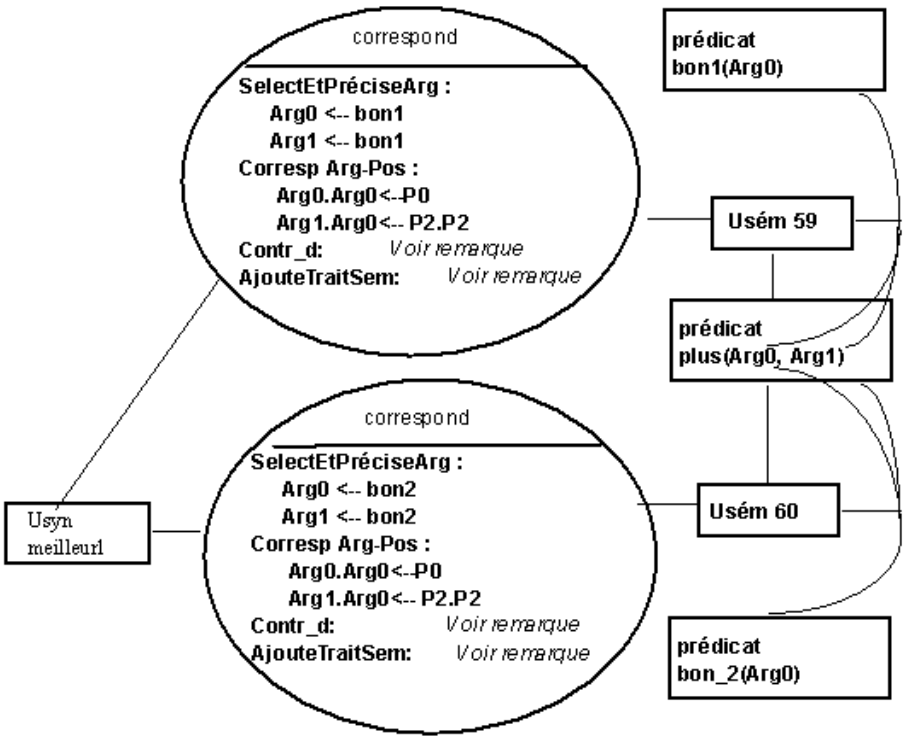
Dans cette perspective, l'unité sémantique devra pouvoir pointer sur le prédicat *plus* en forçant les arguments de *plus* à être eux-mêmes une structure prédicative (un prédicat précisé par la correspondance et éventuellement ses arguments).

Ex : *meilleur* (Um de catégorie adjectif) a une Usyn (correspondant à son comportement comparatif de *bon* à un argument) caractérisée par

```
cb : P0 P1 P2

P0 : SN

P1 V[souscat : copule]

P2 SADJ : (P0) IntervConst P1 P2

P0 : SADV

IntervConst :Adj

[sscat : comparatif]
```

```
P1 : Conj[Lex : que]

P2 : SN
```

En supposant que l'on ait deux sens de *bon (bon_1 et bon_2)* liés à deux prédicats unaires, on aurait pour cette Usyn la représentation suivante :



Le calcul précise les arguments du prédicat associé à l'unité sémantique (au premier niveau), ici le prédicat binaire *plus* . Ce prédicat prend pour arguments les deux structures prédicatives dont les arguments sont eux-mêmes précisés par le calcul, ainsi Arg0<=*bon_1*(P0) signifie que l'argument Arg0 de *bon_1* correspond à P0. Le calcul de cet argument est donc implicitement décrit également.

Remarque :

Les informations de filtrage par traits de P0 et P2.P2 dépendent des prédicats lexicaux *bon_1* et *bon_2* et du codage des correspondances permettant d'atteindre ces prédicats. Ils doivent donc être codés de façon homogène et nous ne le préciserons pas ici.

Nous ne décrirons pas non plus ici le comportement syntaxique correspondant à *meilleur* superlatif qui correspondrait bien entendu à une ou plusieurs autres Usyn, associées elles-mêmes à des Usem.

On rendra compte suivant le même principe (autres couples Usyn-Usem) des structures associées à :

Luc est meilleur en maths que Max en français

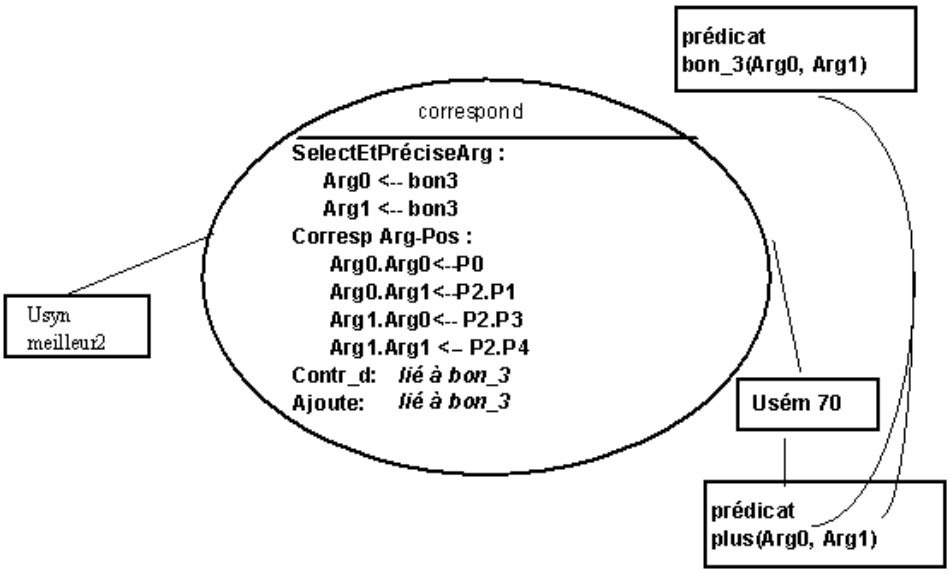
Luc est meilleur en maths qu'en français

Il est meilleur en maths que Max

qui sont à associer à un prédicat *bon_3* à deux arguments. Avec le jeu des co-références associés à des occupants de position d'étiquette `e` (non exprimé en surface), on pourra décrire ces trois contextes sur une même Usyn.

Ex : *meilleur* AJ

```
db : P0 P1 P2

P0 : SN [coref : i]

P1 V[souscat : copule]

P2 SADJ : P0 IntervConst P1 P2 P3 P4

P0 : SADV[souscat : degré]

IntervConst Adj [souscat : comparatif]

P1 SP [prep : en]

[coref : j]

P2 : Conj[Lex : que]

P3 : SN

e [coref : i]

P4 SP [prep : en]

e [coref : j]

Si P3 = e[coref : i],

alors ! P4 = e[coref : j]
```



6.2. Le cas des Unités syntaxiques composées

Dans le principe, l'association Usyn composée Usem ne pose pas de problèmes particuliers. En effet, la correspondance s'effectue comme pour les Usyn simples en précisant le filtrage, l'origine des arguments, les valeurs par défaut... Comme pour les Usyn simples, l'Usem doit en général décrire le sens du *self* de la DB. Cependant, une différence importante et qui n'est pas sans répercussion sur la correspondance syntaxe-sémantique est le fait que ces composés syntaxiques représentent non pas une forme, mais un ensemble de réalisations possibles, et que dans cet ensemble, certaines variations ne sont que des variations de formes sans répercussions sur le sens, et que d'autres sont des variations figées ou semi-figées de la forme ayant une répercussion sur le sens. C'est ainsi que l'ensemble des formes associées à un composé syntaxique Fil-de-fer-barbelé {fil de fer barbelé, fil barbelé, barbelé} correspond à un même sens, alors que les formes {en connaissance de cause, en toute connaissance de cause, en parfaite connaissance de cause} associées à une autre Usyn composée correspondent à un même noyau de sens, lequel est modifié par "toute" ou "parfaite" ; la position "modifieur", bien qu'étant partiellement figée au niveau lexical par une liste finie de lexicalisations, joue alors un rôle de modification "libre" d'intensité du point de vue sémantique. D'autres problèmes de cet ordre peuvent se poser, voyons plus précisément lesquels sur certains cas particuliers.

6.2.1. Intérieur/extérieur d'un composé syntaxique

L'association d'une Usyn composée à une Usem prédicative peut servir de guide dans le découpage *intérieur/extérieur* de la façon suivante : le prédicat correspond à l'intérieur du composé qui ne présente pas de compositionnalité sémantique, et les arguments sont à associer à des positions de la DB externe. Ainsi, pour le composé *X met Y en Ïuvre pour Z, mettre* et *en Ïuvre* formeront l'intérieur du composé, et seront en fait associés au prédicat ternaire *mettre_en_Ïuvre(Arg0, Arg1, Arg2).* Ainsi, lorsqu'il n'y a pas compositionnalité de sens, ce critère est simple.

Mais bien souvent, on ne peut pas dire qu'il n'y ait pas du tout compositionnalité de sens, et c'est alors au lexicographe de choisir s'il veut ou non mettre l'accent sur la compositionnalité en associant au composé une Usem prédicative dont le ou les arguments proviennent de ce qui a été identifié comme l'intérieur du composé.

Remarque :

La compositionnalité de sens est un phénomène analogue au figement des expressions complexes : on peut l'envisager en terme de continuum : exclusivement compositionnel, partiellement compositionnel, non compositionnel. Associer une seule Usem à toute une structure interne bien que celle-ci présente une certaine compositionnalité n'est cependant pas réducteur, au niveau sémantique, dans la mesure où les relations sémantiques entre les unités sémantiques permettront de rendre compte de proximités de sens dues à la compositionnalité de la structure interne.

Ex : on peut traiter comme un seul composé syntaxique ou comme 4 composés syntaxiques différents les entrées suivantes :

filtre à air

filtre à eau

filtre à huile

filtre à café

suivant que l'on veut leur associer une même Usem prédicative *filtre(Arg0)* dont l'argument sera précisé lors de la correspondance comme provenant respectivement de *air, eau, huile, café,* ou quatre différentes Usem *filtre_café, filtre_eau, filtre_huile, filtre_café,* dont la relation avec *filtre* ne sera mise en évidence qu' au niveau sémantique par le biais d'une relation taxinomique. Suivant le choix lexicographique effectué, la correspondance syntaxe/sémantique aura ou non besoin d'accéder à des positions de la structure interne du composé.

On peut aussi coder l'ensemble des *filtre à X* sur un seul composé en syntaxe, et leur associer par filtrage les n unités sémantiques différentes (en fonction du X).

Remarque :

On pourrait également coder ces composés en morphologie. Le problème de la mise en évidence de la compositionnalité ne se poserait alors plus : l'Um composée serait associée à une Usyn simple elle-même associée à une Usem. Les relations sémantiques sont dans ce cas la seule façon de rendre compte de proximités de sens.

On peut aussi décider de décrire ces structures comme comportement syntaxique de *filtre* et lui associer l'Usem prédicative *filtre(Arg0)* . Ce codage peut d'ailleurs sembler le plus économique, et donc le meilleur, si l'on ne souhaite pas mettre en évidence au niveau sémantique en tant qu’unités autonomes *filtre à air, filtre à eau* . On rendra alors compte de la "composition en sémantique" par des relations de collocations entre Usem *filtre* et Usem *eau*, Usem *air*...

Dans les cas où le découpage intérieur-extérieur du composé syntaxique n'est pas évident et présente un aspect arbitraire, et dans la mesure où le codage de la couche syntaxique doit pouvoir être fait sans connaître les choix qui seront pris au niveau sémantique, on est amené à faire porter l'ensemble des informations de correspondance non seulement sur la construction décrivant la structure syntaxique externe, mais aussi éventuellement sur celle qui décrit la structure interne du composé. Cependant, il ne faut pas oublier que l'approche du modèle GENELEX est tout de même, quand c'est possible, de faire apparaître dans la construction externe les positions auxquelles on associe des arguments au niveau sémantique.

D’autre part, le codage des composés syntaxiques se fait aussi par un "calcul" sur la "Composition" qui décrit le composé en se basant sur les unités composantes et les différentes interactions entre unités composantes. La correspondance syntaxe-sémantique s’appuiera donc également sur la description des composés par "Composition".

Le langage d'expression de la correspondance doit donc préciser explicitement, lorsqu'il se réfère à une position, s'il s'agit d'une position de la Construction externe, de la structure interne, et en cas de description par Composition, de quel composant il s’agit.

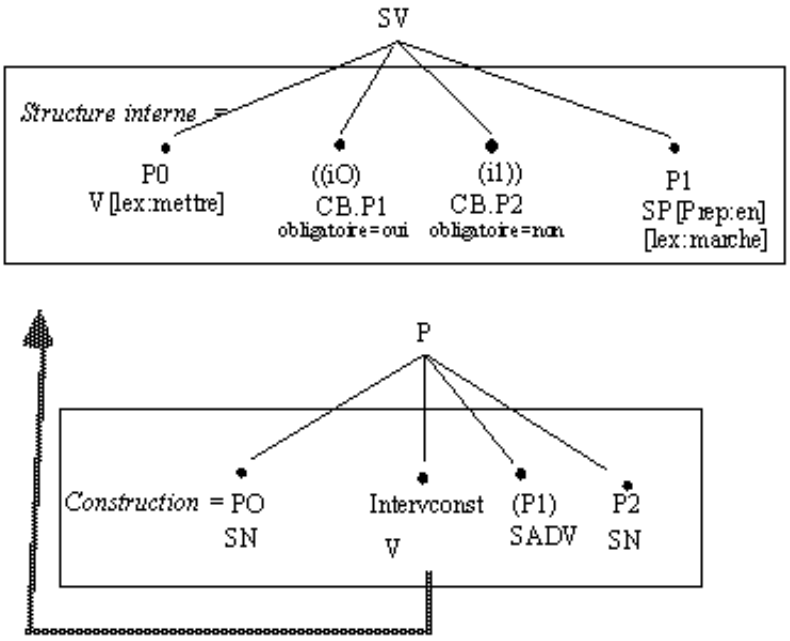
6.2.2. Insertion de modifieur

Une autre particularité liée aux composés en syntaxe est ce qui concerne l'insertion de modifieurs, qu’ils soient totalement, partiellement ou pas du tout lexicalisés. En effet, lorqu’ils sont lexicalisés, ils figurent généralement dans la description "interne" du composé, que ce soit la structure interne ou la Composition. Le cas le plus fréquent est sans doute celui où le modifieur porte (selon la syntaxe générale) sur le verbe TæTE de la structure interne ; il s'agit d'un cas simple où la compositionnalité génùrale fera porter la modification sur la représentation sémantique associée à l'Usyn composée

Ex : *Il met **fréquemment** en marche le chauffage*

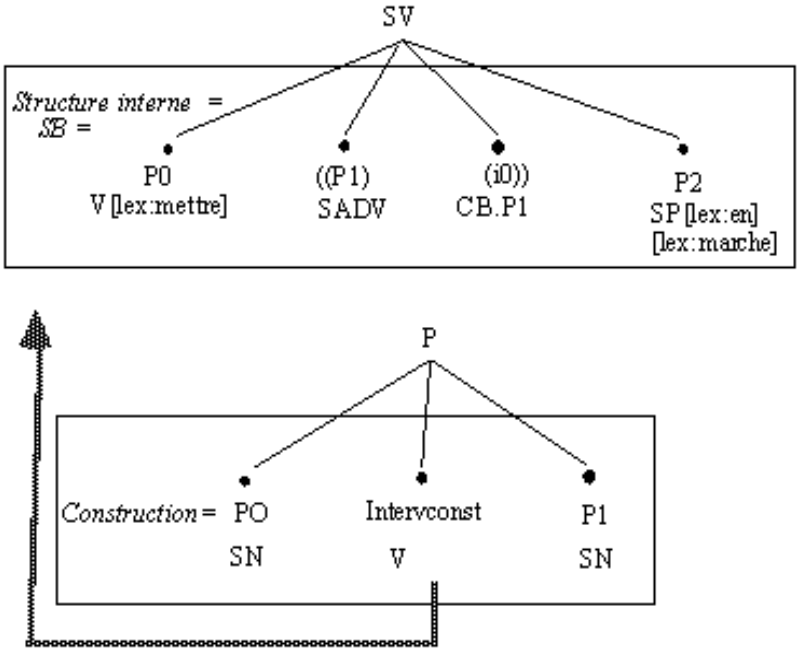
*il met **fréquemment** le chauffage en marche*

pourra être représenté de la façon suivante :



La correspondance au niveau sémantique n'a alors rien à préciser en ce qui concerne la portée du modifieur : celle-ci est conforme aux règles générales .

On peut également faire figurer l'insertion du modifieur directement dans la SB comme suit :



Il conviendrait alors d'adopter la règle implicite (au niveau de la grammaire) qui fixe la portée du modifieur de la TETE de la Construction comme modifieur de l'entrée composée décrite (cela signifie donc que le modifieur de *mettre* serait dans ce contexte interprété comme modifieur de *mettre en marche).*

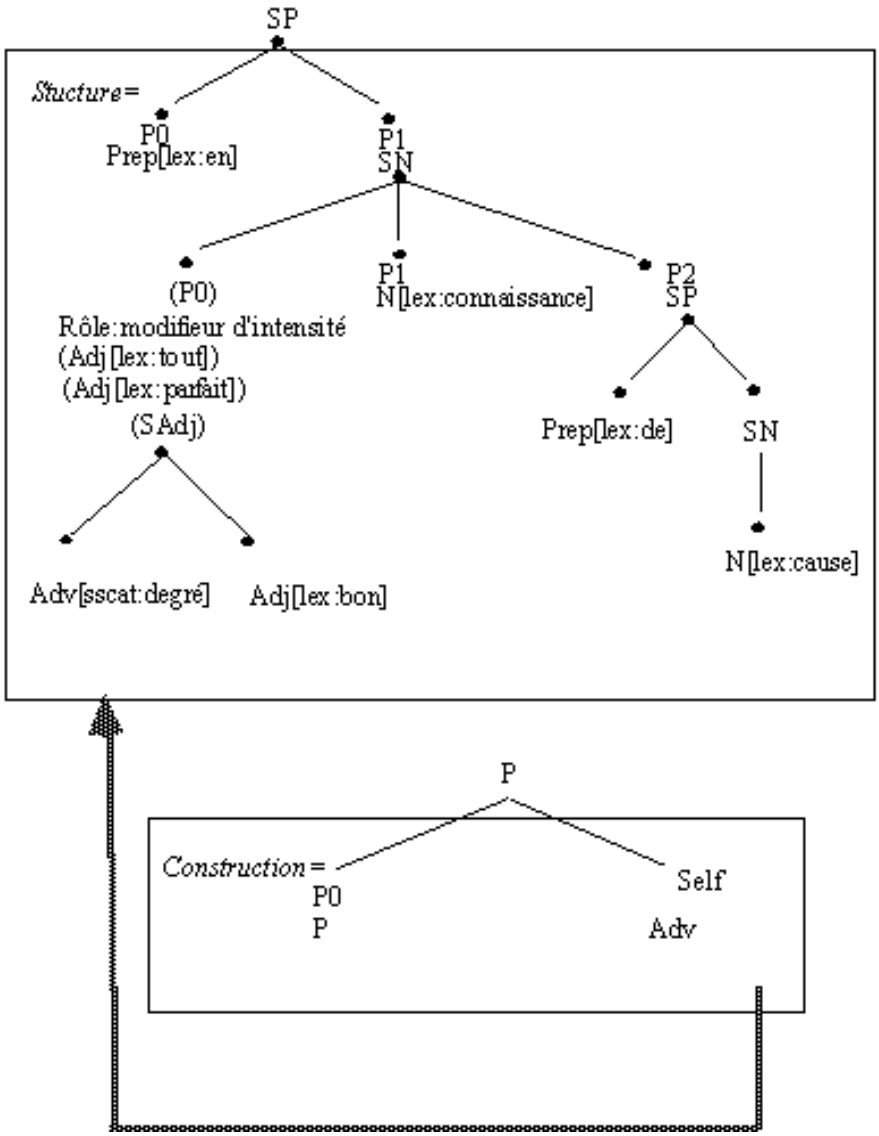
Cela ne va cependant pas sans poser de problèmes car, à l'intérieur du composé, certains composants

optionnels sont à considérer comme modifieurs, et d'autres jouent seulement un rôle au niveau de la surface sans modifier le sens du composé. La solution qui consiste à insérer les modifieurs au niveau de la DB et à préciser dans la SB leur point d'insertion semble la meilleure chaque fois que c'est possible et que ces modifieurs ont au niveau sémantique un apport tout à fait analogue à celui qu'ils ont pour des unités simples.

Certains cas sont plus complexes, et posent un problème qu'on ne peut résoudre aussi directement. En particulier, l'insertion facultative d'adjectif modifiant un nom de l'intérieur du composé, adjectif qui doit être interprété comme modifieur du composé.

Il s'agit la plupart du temps de modifieurs d'intensité du composé. On peut, au niveau de la représentation syntaxique, proposer de leur donner, dans la SB, un rôle thématique permettant d'indiquer leur interprétation sémantique.

*Ex : En t**oute/parfaite/très bonne** connaissance de cause*



Cependant, dans la mesure où l'utilisation ou non de rôles thématiques dépend de l'approche choisie pour la description du lexique, on ne peut l'imposer pour résoudre ce problème.

La fonction **modifieur** attribuée à ces positions peut aussi être interprétée de la façon souhaitée par

l'interprétation sémantique générale gérée par la *grammaire*. C'est donc une telle convention que le modèle Genelex peut recommander, convention à adopter pour une instanciation du modèle dans un dictionnaire GENELEX donné.

Remarque :

Les structures de base comportant des positions optionnelles ne comportent pas nécessairement pour autant des modifieurs.

Citons pour exemple le composé

(*fil (de fer)) barbelé* .

Il est donc indispensable de pouvoir préciser les insertions correspondant à des modifieurs.

7. Mécanismes mis en Ïuvre

Compte-tenu des besoins exprimés plus haut et afin de rendre compte des diverses informations nécessaires à une mise en correspondance fine des deux couches, la correspondance est porteuse des informations suivantes : **Correspondance Argument-position, SelectEtPréciseArg**, **Contraint_description.**

Les **Correspondance Argument-position** permettent de décrire quel est le correspondant syntaxique d'un argument. Dans le cas où l'argument est "représenté" dans la couche syntaxique, cela consiste à associer un argument et une position en pointant sur l'un et l'autre. Le prédicat peut aussi comporter des arguments profonds absents en surface. Dans ce cas, la correspondance précise les valeurs de ceux-ci lorsqu'elles sont liées au couple Usyn-Usem et non au prédicat lui-même (voir le point suivant).

On se donne la possibilité que la couche sémantique identifie un certain nombre d'arguments dont l'expression en surface ne correspond pas à une position ; ces arguments sont associés à des "positions flottantes" appelées depuis la sémantique. Ces positions (que l'on peut voir comme des compléments circonstanciels ou adjoints en syntaxe, donc absentes de la description de base) sont décrites comme les positions de la syntaxe (un même objet permet d'en rendre compte : on peut associer à ces "positions flottantes" une fonction, un rôle thématique, et une distribution) ; leur niveau d'insertion (mais non leur point d'insertion dans la liste de Position) dans la Construction de la DB est précisé.

SelectEtPréciseArg donne des informations concernant les arguments du prédicat associé à l'Usem : en particulier, il précise les valeurs par défaut d'arguments issus de positions optionnelles ou les informations dépendant du contexte syntaxique particulier associé à l'Usyn concernée par la correspondance syntaxe-sémantique. C'est également cet objet qui permet de préciser qu'un argument du prédicat au plus haut niveau (c'est-à-dire pointé directement par l'Usem entrant en jeu dans la correspondance) est lui-même un prédicat que précise la correspondance.

Contraint_description rajoute des contraintes sur la description de base de l'Usyn. Ces contraintes s'expriment par l'ajout de traits (de la syntaxe) sur des éléments de la distribution d'une position, le

blocage de certains éléments, ou l'interdiction ou le forçage de la réalisation d'une position considérée optionnelle dans la couche syntaxique.

Dans l'exemple précédent, l'Usem36 interdit que la position P2.S1.P2 (optionnelle dans la DB) soit réalisée. Elle est donc marquée comme telle.

De la même façon on préciserait les positions optionnelles obligatoirement réalisées .

E - Bibliographie de référence

On trouvera ici quelques ouvrages de référence. Ces ouvrages contiennent des bibliographies plus complètes qu'il sera intéressant d'aller consulter si l'on veut étudier l'un ou l'autre des aspects du modèle de façon approfondie.

Cruse, D.A.1986. Lexical Semantics. Cambridge : Cambridge University Press

Evens, M W, editor, 1988. Relational models of the lexicon. Cambridge University Press

Gross, G. in Revue Langages , "SELECTION ET SEMANTIQUE, classes d'objets, compléments appropriés, compléments analysables". Septembre 1994. Larousse

Kerbrat Orecchioni, C. 1979. De la sémantique lexicale à la sémantique de l'énonciation. Tome 1. Service de Reproduction des thèses. Université de Lille III

Lecomte, A., Baschung K. & Bès G. Une modélisation des entrées lexicales. 1991. (Rapport Projet Eureka GENELEX)

Mel'cuk I., &al.(1984 & 1988) Dictionnaire Explicatif et Combinatoire du français contemporain. Recherches Lexico-sémantiques I & II. Montréal,Presses de l'Université de Montréal

Pustejovsky, J. 1991. The generative Lexicon. Computational Linguistics. vol 17

Pustejovsky, J. 1989. Current issues in computational lexical semantics, Proceedings of the 4th European ACL, Manchester, pp xvii-xxv

Rastier, F. 1987. Sémantique interprétative. Paris : Presses Universitaires de France.

Rastier, F., 1987. Représentation du contenu lexical et formalismes de l'intelligence artificielle. Langages 87, 77-102

et aussi les actes suivants qui permettent une vue assez large sur les différents problèmes de la sémantique lexicale :

Actes du premier Séminaire de Sémantique Lexicale. du GDR-GRECO Communication Homme-Machine IRIT, Université Paul Sabatier, Toulouse Janvier 1991.

Proceedings of the 2nd Seminar on Computational Lexical Semantics, IRIT, Toulouse Janvier 1992

F - MANUEL DE L'UTILISATEUR

1. Généralités

Le modèle propose un ensemble d'éléments descriptifs permettant une description de **sémantique lexicale** très **fine** basée sur une analyse componentielle ou sur un ensemble très riche de relations sémantiques lexicales, ou plus probablement sur une combinaison des deux axes descriptifs.

Le modèle propose également un ensemble d'éléments de description abstraits, généralisations ou **abstractions à partir du lexique**, certains éléments n'étant plus directement liés à celui-ci, mais plutôt éléments de représentation de connaissances.

Comme dans les couches morphologique et syntaxique, le modèle sémantique évite au maximum l'usage d'éléments descriptifs non explicités ; cela explique le grand nombre d'objets intermédiaires (certains ayant statut de "méta-connaissance") intervenant dans la couche sémantique.

Ces deux modes de description cohabitent sans problème dans un même dictionnaire, et le niveau "méta-descriptif" émerge à partir du niveau lexical, **les deux "vues" possibles sur le lexique étant connectées**.

La plupart des informations descriptives sont associées aux objets de la couche sémantique moyennant la **modalité** des connaissances qu'elles expriment. Une pondération symbolique est affectée, et un ensemble de valeurs de pondération est proposé par le modèle, lequel peut être augmenté suivant les besoins de l'équipe lexicographique construisant une instance de dictionnaire GENELEX.

Les valeurs de pondération symbolique proposées sont les suivantes :

- DEFINITOIRE
- PROTOTYPIQUE
- ACCESSOIRE
- EXCEPTIONNEL
- CONNOTATIF

Outre la pondération symbolique, il est possible de pondérer numériquement ces connaissances. D'autre part, la modalité d'une connaissance comporte aussi un **point de vue** qui donne toute latitude à faire cohabiter différents points de vue sur un même objet et de le décrire suivant ce point

de vue. Ce champ est entièrement libre**.**

2. Usem

2.1. Généralités

L'Unité Sémantique (**Usem**) est le point d'entrée dans la couche sémantique du modèle.

L'Usem est une unité fortement liée à la langue ; elle permet une description de sémantique lexicale fine. Sa description s'appuie sur des objets qui sont diverses abstractions sur les composantes de sens que l'on veut pouvoir affecter aux Usem, certaines ayant un statut plutôt décompositionnel ou analytique, d'autres étant plutôt des généralisations ou abstractions à partir des Usem.

Les Usem peuvent être pointées par plusieurs **Correspondances Syntaxe Sémantique** (Corresp_Usyn_Usem) à condition que ces Correspondances Syntaxe Sémantique appartiennent à des Usyn simples associées à une même Um.

Dans les cas les plus courants, une Usem décrit un sens (ou une acception) d'une unité morphologique (Um simple ou composée) dans les contextes syntaxiques décrits par une ou plusieurs de ses Usyn simples éventuellement filtrées.

Une Usem peut également représenter un des sens d'un composé décrit en syntaxe, auquel cas on ne peut pas parler de l'Um associée, mais d'une Usyn qui regroupe une ou plusieurs Usyn ou Um dans une composition fonctionnant comme un tout du point de vue linguistique.

Les Usem peuvent toutes porter une **CombVE,** quadruplet de valeurs d'emploi :

Niveau de langue (attribut *niveaulgue)* à valeurs : FAMILIER, VULGAIRE, ARGOTIQUE, POPULAIRE, LITTERAIRE, SAVANT, STANDARD). Si le pouvoir expressif proposé par cet attribut est insuffisant au besoin d'une certaine approche de la sémantique, on pourra également compléter l'information par un ou des traits valués pondérés exprimant une information plus fine concernant le niveau de langue.

Fréquence (attribut *fréquence*) à valeurs RARE, COURANT

Variation géographique (attribut *vargeog*) à valeurs libres

Datation (attribut datation) à valeurs ARCHAIQUE, VIEILLI, MODERNE

Le **Prédicat** et le **Concept** sont les deux éléments de description représentant une certaine abstraction (plus ou moins grande), par rapport au lexique; l'Usem s'appuie sur ces objets et passe par ceux-ci pour accéder aux niveaux de description de plus en plus abstraits (et de plus en plus indépendants des particularités lexicales).

Les Usem ont 0 ou une **Représentation Prédicative (**RepresentationPredicative**),** et de 0 à N **Représentations Conceptuelles** associées (RepresentationConceptuelle_Pond). Une Usem qui se décrit par une structure prédicative peut enrichir les représentations sémantiques des arguments du

prédicat, cet enrichissement sémantique est précisé par les différents SelectEtPreciseArg que comporte éventuellement la Représentation Prédicative.

On associe aux Usem 0 à n **Traits Sémantiques Valués Pondérés (Trait_Sem_ValPond)**, caractéristiques décompositionnelles (ou componentielles) du sens. Si l'on choisit de décrire l'Usem en s'appuyant sur des objets plus abstraits généralisés, on s'appliquera à ne noter ici que les informations lexicales fines non partagées par les différentes Usem s'appuyant sur les mêmes abstractions, les autres informations étant implicitement présentes par héritage des objets descriptifs abstraits.

Une Usem est insérée dans un réseau de relations sémantiques (lexicales) entre Usem, ce qui constitue l'un des axes de description de l'Usem. Elle entretient donc un certain nombre de relations sémantiques avec d'autres Usem, ce qui s'exprime par le fait qu'elle est **source** ou **cible** de **Relations Valuées Pondérées** liant des Usem (**R_ValPond_Usem**). Par convention de codage, l'Usem est **source** des R_ValPond_Usem ench‰ssées, les R_ValPond_Usem précisant leur cible ainsi que la relation sémantique liant les deux Usem et la modalité de la mise en relation de ces deux unités.

Une Usem comporte une **définition libre** : il s'agit d'un champ entièrement libre destiné à un lecteur humain ; elle contient par exemple la définition du dictionnaire papier et elle est utile au lexicographe. Aucun contrôle n'est effectué sur ce champ, et le modèle ne dicte pas de règle de structuration de définition, ni de syntaxe de la définition.

Un champ **définition formelle** est également attribué à l'Usem. Il peut être utile par exemple pour associer aux déterminants une formule logique, il peut contenir l'expression de lambda-calcul associée à l'entrée pour ceux qui le souhaitent. Comme le champ précédent, aucun contrôle d'aucune sorte n'est effectué sur le contenu de ce champ.

2.2. Représentation prédicative (RepresentationPredicative)

Une Usem prédicative décrit sa composante prédicative par une et une seule **Représentation Prédicative** pour laquelle l'Usem peut-être **vedette** ou non. L'Usem unique vedette dans son association avec un prédicat est la lexicalisation privilégiée de ce prédicat, la plus neutre, celle qui se décrit par ce prédicat en ajoutant le moins d'information : dans beaucoup de cas où un prédicat est partagé par un ensemble d'Usem formant un réseau sémantique autour d'un verbe noyau, le verbe noyau est l'expression la plus pure du prédicat, son Usem en est la vedette (ex : l'Usem du verbe "acheter" vedette du prédicat *acheter* partagé par les Usem "*acheteur*", "*achat1*", "*achat2*", "*achetable*" ...). La **Représentation Prédicative** est associée à une Usem de façon essentielle (on peut aussi dire définitoire), elle explicite particulièrement l'Usem. C'est la raison pour laquelle il n'est pas nécessaire (ni d'ailleurs possible dans le modèle) de donner une modalité à cette association, c'est aussi la raison pour laquelle on ne peut associer à une Usem qu'une (ou aucune) Représentation Prédicative.

La relation qui lie l'Usem au prédicat est nommée librement par l'attribut **type de lien** qui caractérise la "fonction" de passage de l'Usem au Prédicat. On peut y préciser la sémantique du lien de façon fine (exemple : "aptitude à être argument1" entre l'Usem *lavable* et le prédicat *laver)*.

On précise par **chemin_arg_concerne** le rang de l'argument concerné par la relation (ex : *acheteur* : prédicat *acheter*, argument0) **arg_inclus** indiquera si l'Usem a incorporé l'argument (ex : *acheteur*)

ou non (ex : *achetable* "caractéristique de ce qui peut être argument," chemin_arg_concerne : 1 arg_inclus : NON_I).

2.3. Représentation conceptuelle pondérée (RepresentationConceptuelle_Pond)

Une Usem peut être (indépendamment de sa représentation par Représentation Prédicative) associée à 0, un ou plusieurs concepts en précisant les modalités de cette association (pondération symbolique, pondération numérique et point de vue). La modalité de la Représentation Conceptuelle permet d'associer différents concepts (avec différentes modalités) à une même Usem sans qu'il y ait d'incohérence. De même, une unité décrite comme prédicative en sémantique lexicale fine, pourra cependant être associée à un concept, suivant un certain point de vue qui sera alors précisé.

Les Usem associées à un prédicat en ayant incorporé un argument pourront avoir une représentation prédicative et également une représentation conceptuelle qui rendra compte de la nature de l'argument incorporé, en particulier en s'insérant dans un réseau taxinomique.

Ex : Une Usem "*filtre"* pourra être associée au prédicat "*filtrer"* si on a considéré dans la description de celui-ci que l'instrument faisait partie des arguments du prédicat. On pourra lui associer également un concept "*filtre*" qui s'insèrera dans le réseau des "outils".

L'Usem peut être vedette ou non dans sa relation à un concept, ce qui signifie qu'elle est ou non la représentante lexicale privilegiée du concept, et ce qui signifie aussi qu'il s'agit d'un concept nommé ou lexicalisé (voir Concept) .

La relation qui lie l'Usem au Concept est nommée librement par l'attribut **type de lien** qui caractérise la "fonction" de passage de l'Usem au Concept. On peut y préciser la sémantique du lien de façon fine.

3. Unité sémantique d'affixe (Usem_Aff)

L'unité sémantique d'affixe(Usem_Aff) provient directement d'une Um affixe (Um_Aff) dont elle décrit minimalement le noyau de sens, et qu'elle caractérise du point de vue de la sémantique. Cette information peut permettre un minimum de prédiction de sens pour les dérivations régulières et productives de formes non recensées dans le dictionnaire.

Une Usem_Aff peut être décrite à l'aide de **Traits valués pondérés** (les traits seront de préférence d'un type particulier que précisera l'attribut *concerne* du Trait_semantique par la valeur AFFIXE), de **Concept pondéré** (RepresentationConceptuelle_Pond_Aff) ou de **Prédicat.** Ces éléments peuvent se combiner comme suit : un Prédicat et une liste vide ou non de Traits valués pondérés, un Concept pondéré et une liste vide ou non de Traits valués pondérés, ou une liste non vide de Traits valués pondérés. En effet, un Usem_Aff qui ne comporterait aucune information sémantique n'aurait pas de raison d'être.

Comme pour les Usem, les champs definition_libre et definition_formelle permettent d'associer une définition textuelle ou plus formelle. Des combinaisons de valeur d'emploi sont également précisées (combve)

Ex : un affixe **mis-** (mis-anthrope, miso-gyne)

pourrait avoir une Usem_Aff pointant vers le **prédicat** : "détester"

(à noter qu'on ne précise rien sur le calcul des arguments du prédicat)

Ex : un affixe p**isc-** (pisci-culture)

pourrait avoir une Usem_Aff pointant vers le **concept** : "poisson"

Ex : un affixe -**tion-**- (fabric-ation)

pourrait avoir une Usem_Aff pointant vers le Trait_Sem Value "action = +".

(relié au Trait_Sem *action* avec *concerne = AFFIXE)*

4. Prédicat

4.1 Généralités

Prédicat est l'un des éléments centraux du modèle sémantique : une Usem décrite comme Prédicative s'appuiera sur un Prédicat de type LEXICAL pour décrire son noyau Prédicatif.

Prédicat peut être de différents **types**, proposés par le modèle :

-**LEXICAL** : directement issu d'une Usem, que celle-ci en soit la vedette ou non.

-**GƒNƒRALISƒ** : non directement issu d'une Usem, introduit dans une relation de généralisation d'un autre Prédicat (lui-même éventuellement lexicalisé), et en même temps généralisé par un autre Prédicat .

-**PRIMITIF** : non directement issu d'une Usem, n'admet aucune relation de généralisation. Un Prédicat PRIMITIF peut-être donné a priori comme élément de description préexistant ou prédéfini, indépendant de la langue éventuellement (l'attribut pivot le précise). Si l'ensemble des Prédicat PRIMITIFS est une donnée a priori, les Usem et objets descriptifs intermédiaires se projetteront sur eux. L'ensemble des Prédicat PRIMITIFS peut aussi émerger des descriptions du lexique, leur liste sera dans ce cas connue a posteriori.

-**LEXICAL_PRIMITIF** : à la fois issu directement du lexique et sans généralisation, et éventuellement pivot.

-**TROU_LEXICAL** : pris entre deux Prédicat lexicaux, généralisation non lexicalisée de l'un, generalisé par un autre Prédicat LEXICAL.

Prédicat peut avoir un statut de **pivot**, c'est-à-dire avoir été choisi comme élément de description pivot, indépendant des langues et élément descriptif partagé par plusieurs langues.

Prédicat peut avoir une **définition libre** : il s'agit d'un champ entièrement libre destiné à un lecteur humain. C'est là que sera consignée la définition textuelle ou toute expression utile au lexicographe. Aucun contrôle n'est effectué sur ce champ, et le modèle ne dicte pas de règle de structuration de définition, ni de syntaxe de la définition.

Prédicat peut avoir une **définition formelle** : il s'agit d'un autre champ entièrement libre qui pourra contenir une définition formelle (formule de lambda-calcul par exemple). Comme sur le champ précédent, aucun contrôle d'aucune sorte n'est effectué.

Prédicat peut être **source ou cible de relations** valuées pondérées entre Prédicat (R_ValPond_Pred) ; il peut aussi être source ou cible de relations valuées pondérées liant Prédicat et Concept (R_ValPond_Pred_Concept et R_ValPond_Concept_Pred). Parmi les relations entre Prédicat, une relation de type généralisation joue un rôle particulier, puisqu'elle structure les Prédicat entre eux. Prédicat porte les relations valuées pondérées entre Prédicat : R_ValPond_Pred et entre Prédicat et Concept : R_ValPond_Pred_Concept.

Prédicat conna"t la liste de ses arguments, celle-ci est une caractéristique essentielle du Prédicat, indissociable de celui-ci. (les Argument sont aux Prédicat à peu près ce que les Position sont aux Construction.)

La **liste des Arguments** référés a un **ordre pertinent** pour le modèle car, pour un Prédicat donné, le rang d’un Argument dans la liste sera utilisé à l'extérieur du Prédicat pour référer à son nième argument ; par convention, la numérotation des Argument commence à 0. Les Argument ne portent pas explicitement leur rang comme attribut, car le rang est plutôt une nécessité pratique de description (pour accéder de l'extérieur à un Argument) qu'une information pertinente du point de vue linguistique. Le rang n'étant pas une caractéristique de l'Argument, les Prédicat peuvent partager des Argument indépendamment du rang qui leur est affecté par ces Prédicat.

Un Prédicat peut pointer vers des **traits valués pondérés**, caractéristiques sémantiques liées à l' axe décompositionnel de la couche sémantique.

4.2. Argument

Argument est fortement associé à la définition de la notion de prédicat, dont il est un élément totalement indispensable : en effet un prédicat n'est prédicat que s'il a au moins un Argument.

Prédicat conna"t un certain nombre d'informations sur chacun de ses arguments. Les Argument peuvent être, en tant qu'éléments de description, **partagés par plusieurs Prédicat**, et ils interviennent à différents rangs dans la liste d'arguments de ces différents Prédicat (un peu de la même facon que les Position de la syntaxe peuvent être partagées par plusieurs Construction, Prédicat étant alors à mettre en parallèle avec Construction).

Argument est caractérisé par **au moins UN** et peut-être plusieurs **Rôles sémantiques** (attribut role_sém_l qui pointe vers des Role_Sem) et des informations sémantiques exprimées par la liste d’InformeArg informe_arg_decrit_l ; les InformeArg référés directement par l'Argument ne portent pas l'information de rang, car elle est totalement inopérante dans ce cas (cette information est utile lorsqu'on accède de l'extérieur du Prédicat à un argument en tant qu'argument d'un Prédicat) .

Les rôles sémantiques (**Role_Sem**) sont décrits hiérarchiquement par une relation est_un figée dans le modèle. Cela se traduit par l'attribut est_un_l qui fait pointer un Role_Sem vers son ou ses "pères" dans la hiérarchie.

La connexion avec les rôles thématiques utilisés en syntaxe se fait en associant à certains Role_Sem de la couche sémantique un rôle thématique de la couche syntaxique. On peut donc avoir une liste de rôles différente de la liste des rôles proposée en syntaxe ; la seule contrainte pour assurer une cohérence entre les deux couches est que les rôles de la syntaxe doivent être "plongés" dans une hiérarchie de Role_Sem, les rôles de la sémantique pouvant être plus précis. La cohérence voudra aussi que le rôle thématique (quand il est utilisé en syntaxe) soit compatible avec le rôle sémantique correspondant à l'argument issu de la position portant le rôle thématique en ce sens que le rôle sémantique peut être le correspondant du rôle thématique ou un descendant de celui-ci dans la hiérarchie des rôles.

4.3. SelectEtPreciseArg, InformeArg

SelectEtPreciseArg pointe vers un Argument de Prédicat à un niveau quelconque de profondeur ; quand les arguments de prédicats sont eux-mêmes des prédicats, on accède aux arguments par la suite des rangs d'arguments de chaque niveau. L'attribut *chemin_arg* comporte donc une suite de rangs d'Argument, et permet dans tous les cas d'atteindre l'Argument concerné. Cet Argument est enrichi d'informations sémantiques de différents statuts par les **InformeArg** pointés.

InformeArg intervient (directement ou via **SelectEtPreciseArg**) à différents endroits de descriptions pour ajouter des informations qui s'accumulent sur un argument d'un prédicat. Prédicat en lui-même conna"t ses Argument et ces connaissances sur ses Argument s'expriment par des InformeArg portés par les Argument eux-mêmes. Une Usem se décrit par un Prédicat en rajoutant éventuellement quelques informations sur les Argument. La correspondance Usyn Usem peut rajouter également des connaissances sur les Argument du Prédicat auquel est associée l'Usem. Toutes ces connaissances sur les Argument s'accumulent, chaque niveau apportant ce qui lui est propre, aucun apport n'étant par ailleurs obligatoirement présent.

Les correspondances d'Argument (qui interviennent dans les relations valuées pondérées entre Usem prédicatives ou entre prédicats) apportent éventuellement des informations sur les arguments (de la source ou de la cible) sous la forme de **SelectEtPreciseArg**.

Les connaissances sur un Argument ont un statut qui permet de leur donner différentes modalités, essentiellement liées à la gestion des défauts, vérifications ou construction de la représentation sémantique de l'Argument.

Un InformeArg porte un ensemble d'informations sur un argument, ces informations ayant toutes le meme **statut** (DEFAUT, VERIF, ENRICHIT, ou DEFAUT_VERIF).

> DEFAUT : information sur l'argument, en l'absence de représentation sémantique de celui-ci : absent en surface, ou occupé par un élément "vide" sémantiquement. La valeur concernée peut même être une Usem.
>
> VERIF : quand l'argument présent a une représentation sémantique, celle-ci doit

vérifier les contraintes affectées de ce statut: par exemple porter les trait_sem_valpond_l pointés ici, ou avoir pour généralisation les Concept pointés par concept_l.

ENRICHIT ajoute de l'information sémantique à l'information connue sur l'argument, que celle-ci existe ou non.

DEFAUT_VERIF combine l'apport par défaut, en cas d'absence d'informations, à la vérification en cas d'argument "plein" sémantiquement.

pred_instancie pointe vers un prédicat instancié PredInstancie, qui sera associé plutôt à un statut de DEFAUT ou VERIF (surtout si le prédicat n'est pas lexical) ou DEFAUT_VERIF ; ENRICHIT sera plutôt réservé au cas où l'information est rajoutée par un Prédicat instancié sur l'un de ses Argument.

concept_l pointe vers une liste de Concept, le plus souvent 1 ou 0; le cas >1 sert a insérer dans une polyhiérarchie de Concept dont on voudrait que l'argument hérite. Les différents statuts sont compatibles avec cette information.

usem pointe vers une Usem, le statut associé est le plus souvent DEFAUT, du fait de la forte contrainte de lexicalisation que représente l'information portée par une Usem, ou ENRICHIT, en particulier quand l'information est rajoutée par un Prédicat instancié PredInstancie .

trait_sem_valpond_l pointe vers une liste de traits sémantiques valués pondérés, les différents statuts sont également compatibles avec cette information.

Plusieurs InformeArg ou SelectEtPreciseArg peuvent porter sur un même Argument, l'information, avec différents statuts, s'accumulant.

4.4. Prédicat instancié (PredInstancie)

PredInstancie est l'association d'un Prédicat et d'informations plus ou moins précises et complètes sur la représentation sémantique de ses Argument. Les Argument peuvent être entièrement décrits, "instanciés" par des Usem ou seulement partiellement renseignés, par des Concepts, ou des Trait_Sem_ValPond. La liste informe_arg_l peut même être totalement vide, auquel cas cet élément se contentera de pointer vers un prédicat (et implicitement sur les SelectEtPreciseArg de la description de celui-ci). Formellement, les informations accumulées sur les arguments à différents moments de la description s'expriment dans le même langage d'InformeArg (via SelectEtPreciseArg ou non). Un prédicat instancié (PredInstancie) est donc tout simplement un Prédicat qui sait sur ses arguments un peu ou beaucoup plus que les connaissances propres à sa définition de Prédicat .

4.5. Liste de prédicats (ListePred)

Une liste **de prédicats instanciés** permet de regrouper ces prédicats instanciés, de donner un statut à leur ensemble et de le faire intervenir comme élément autonome de description. On peut, pour une liste de prédicats, associer à certains ensembles d'arguments des **Variables**, pour préciser le partage

d'arguments entre différents Prédicat. (un peu à la manière des variables de Prolog).

Une liste de prédicats instanciés donne le moyen de définir un prédicat en l'insérant ou non dans cette liste, d'expliciter les présuppositions liées à un prédicat, les implications, et même de décrire un scénario auquel prédicat et concept pourront faire référence.

Une liste de prédicats a un **statut** qui précise le statut du regroupement des prédicats enrichis : un certain nombre de valeurs sont proposées par le modèle :

- SCƒNARIO

- DƒFINITION

et l'utilisateur pourra au besoin rajouter ses propres valeurs.

Une liste de prédicats a également un **type** précisant le sens à donner à l'ordre des prédicats enrichis dans la liste. Un certain nombre de valeurs sont proposées par le modèle :

- ORDRE_TEMP

- ORDRE_SPATIAL

- SIMULTANEITE

l'utilisateur pourra au besoin rajouter ses propres valeurs.

L'**ordre** de cette liste est **pertinent**, le sens qu'il faut lui donner est précisé par le type, et le rang dans la liste est utilisé également pour décrire les variables en accédant à des arguments de prédicats de cette liste.

Une liste de prédicats est associée à un prédicat ou un concept par une **Assoc_Liste_Préd** qui précise la modalité de cette association.

Le **rôle** de la liste de prédicats par rapport au prédicat ou au concept qui le fait intervenir est précisé. Un certain nombre de valeurs sont proposées par le modèle :

- A_PRESUPPOSITION

- A_IMPLICATION

- A_DƒFINITION

- A_EXPLICITATION

- PARTICIPE_A

et l'utilisateur pourra au besoin rajouter ses propres valeurs.

Un champ *intervient* précise, pour un Prédicat, s'il est lui-même intervenant direct dans la ListePred,

c'est-à-dire s'il intervient dans l'un des prédicats instanciés de cette liste (il y figurera alors expressément, la liste pouvant être référée par d'autres objets de la sémantique).

4.5.1. Variable

Une variable de liste de prédicats associe des arguments de plusieurs Predicat_instancie, lesquels "s'unifient" pour la Liste de prédicats qui les fait intervenir.

Cette association se fait en pointant sur une liste de SelectPredArg, chemins vers des arguments de Prédicat devant s'unifier.

4.5.2. SelectPredArg

Un chemin vers un argument (SelectPredArg) permet d'atteindre un argument d'un prédicat qui intervient dans une liste de prédicats.

Son rang dans la liste (nieme_pred) désigne un prédicat instancié de la liste.

L'argument concerné de ce prédicat est atteint gr‰ce à l'attribut chemin_arg. Dans la plupart des cas, cette liste de rang contiendra un seul élément ; dans les cas rares où ce prédicat a un argument complexe, c'est-à-dire est lui-même prédicatif, on continuera éventuellement le chemin vers un argument en descendant dans la structure prédicative intervenant comme réalisation de l'argument complexe et en allant pointer récursivement vers son argument de rang suivant dans la liste. Ce cas complexe explique l'existence ici d'une liste de NUMBERS.

Pour illustrer les notions de liste de prédicats instanciés, de variables de telles listes, et de chemin vers un argument, voici un exemple :

Le prédicat `acheter (X : acheteur, Y : objet, Z : vendeur, W : argent)` pourrait être associé à une liste de prédicats enrichis (à statut définitoire, ordre temporel)

- `posséder(Z, Y)` prédicat instancie 1

- `désirer(X, Y)` prédicat instancie 2

- `posséder(X, W)` prédicat instancie 3

- `donner(X, W, Z)` prédicat instancie 4

- `donner(Z, Y, X)` prédicat instancie 5

- `posséder (X, Y)` prédicat instancie 6

- ...

(à noter que le même Prédicat (`posséder)` intervient trois fois dans la liste, dans deux prédicats enrichis différents, et le Prédicat donner, deux fois.

La variable représentée ici par X serait associée dans le modèle à 5 chemins vers des arguments :

- prédicat 2, argument 1

- prédicat 3, argument 1

- prédicat 4, argument 1

- prédicat 5, argument 3

- prédicat 6, argument 1

de même, la variable Y serait associée à :

- prédicat 1, argument 2

- prédicat 2, argument 2

- prédicat 5, argument 2

- prédicat 6, argument 2

un même mécanisme s'applique bien sžr à W et Z.

5. Concept

Concept est l'un des éléments centraux du modèle sémantique : il s'agit d'une abstraction "cognitive" sur des Usem ou d'une généralisation de telles abstractions.

Un Concept peut être de différents types, proposés par le modèle :

-**LEXICAL** : directement issu d'une Usem, que celle-ci soit vedette ou non.

-**GƒNƒRALISƒ** : non directement issu d'une Usem, introduit dans une relation de généralisation d'un autre Concept éventuellement lexicalisé, et en même temps généralisé par un autre Concept (et donc non PRIMITIF).

-**PRIMITIF** : non directement issu d'une Usem, n'admet aucune relation de généralisation. Un concept PRIMITIF peut être donné a priori comme élément de description préexistant ou prédéfini, indépendant de la langue éventuellement (l'attribut pivot le précise). Si l'ensemble des concepts PRIMITIFS est une donnée a priori, les Usem et objets descriptifs intermédiaires se projetteront sur eux. L'ensemble des concepts PRIMITIFS peut aussi émerger des descriptions du lexique, leur liste sera dans ce cas connue a posteriori.

-**LEXICAL_PRIMITIF** : à la fois issu directement du lexique et sans généralisation, et éventuellement pivot.

-**TROU_LEXICAL** : pris entre deux Concepts lexicaux, généralisation non lexicalisée de l'un, généralisé par l'autre (les concepts de ce type seront utiles en particulier pour décrire proprement un certain nombre de taxinomies qui comportent des "trous" au niveau lexical).

Un Concept peut avoir un statut de **pivot**, c'est-à-dire avoir été choisi comme élément de description pivot, indépendant des langues et élément descriptif partagé par plusieurs langues.

Un Concept peut avoir une **définition libre** : il s'agit d'un champ entièrement libre destiné à un lecteur humain. C'est là que sera consignée la définition du lexicographe. Aucun contrôle n'est effectué sur ce champ, et le modèle ne dicte pas de règle de structuration de définition, ni de syntaxe de la définition.

Un Concept peut avoir une **définition formelle** : il s'agit d'un autre champ entièrement libre qui pourra contenir une définition formelle (formule de lambda-calcul ...). Comme sur le champ précédent, aucun contrôle d'aucune sorte n'est effectué.

Un Concept peut être **source ou cible de relations** valuées pondérées entre Concept (R_ValPond_Concept) ; il peut aussi être source ou cible de relations valuées pondérées liant prédicats et concepts (R_ValPond_Concept_Pred, R_ValPond_Pred_Concept). Parmi les relations entre Concept, une relation de type généralisation joue un rôle particulier, puisqu'elle structure les Concept entre eux. Par convention, Concept portera les R_ValPond_Concept et les R_ValPond_Concept_Pred dont il est la source.

Un Concept peut avoir un certain nombre de **traits sémantiques** qui lui sont liés tout particulièrement en ce sens qu'ils sont particulièrement **pertinents**, et donc qu'il est intéressant (**trait_sem_pertinent_l**) ou **obligatoire** (**trait_sem_obligatoire_l**) de renseigner pour toute Usem ou concept moins général qui lui seraient associés. Cela peut permettre par exemple d’associer à un Concept des traits particularisants que les Usem ou Concept associés (directement ou par généralisations successives) à celui-ci devront ou pourront renseigner.

On peut attribuer à un concept des **traits valués pondérés**, caractéristiques sémantiques liées à l' axe décompositionnel de la couche sémantique. On utilisera les mécanismes d'héritage pour éviter de préciser tous les traits à tous les niveaux. (Une relation de type PARTICULARISATION pourra supposer héritage des informations portées par le "père", il s'agit d'un choix d'instanciation du modèle.)

Concept peut être associé à une liste de prédicats (de type SCENARIO le plus souvent...) gr‰ce aux Assoc_Liste_Pred qui précisent la modalité de cette association.

6. Trait sémantique valué

La notion de trait valué est une notion très courante en représentation des connaissances et en linguistique computationnelle (souvent plutôt désignée comme "couple attribut-valeur"). Elle permet d'affecter des connaissances "élémentaires" à des objets, et l'utilisation informatique de ces connaissances est assez commode. Une modélisation objet, d'autre part se prête bien à l'utilisation de tels traits valués.

D'autre part, la sémantique componentielle s'exprime au travers de traits sémantiques.

Le modèle de la couche sémantique propose donc un langage de traits sémantiques valués que peuvent porter les principaux objets descriptifs du modèle (Usem, Prédicat, Concept).

Les traits sémantiques valués du modèle GENELEX sont structurés, et entretiennent un certain

nombre de relations entre eux. D'autre part, quand c'est possible, on s'est efforcé de rompre l'hermétisme d'un méta-langage de traits en connectant celui-ci au langage (qu'il sert par ailleurs à décrire).

6.1. Trait sémantique valué pondéré (Trait_Sem_ValPond)

Un trait sémantique valué pondéré est l'association d'un trait sémantique valué et d'une pondération symbolique à liste de valeurs proposée et adaptable par l'utilisateur à ses besoins. On peut également indiquer une pondération numérique et la marque d'un point de vue.

6.2. Généralités

Un trait sémantique valué est l'association d'un trait sémantique et d'une valeur de ce trait.

Il est possible de préciser si ce trait a un statut de **pivot**, c'est-à-dire s'il est indépendant de la langue, et s'il peut donc intervenir avec la même définition dans des descriptions d'objets linguistiques de différentes langues.

Un trait valué réfère au **Trait sémantique** auquel on associe la valeur, ce trait sémantique ayant lui-même des propriétés et étant décrit dans un objet à part entière.

La **valeur** associée au trait est précisée par le trait valué (*valeurtrait ou valeurbin).*

La valeur d'un trait valué peut être nommée dans la langue décrite, le champ **usem_lexicalise_val** réfère donc éventuellement à l'Usem qui nomme le trait valué ; c'est ainsi que les traits valués, bien que constituant un méta-langage, se trouvent ici connectés à la langue ; tous les traits valués ne sont bien sžr pas associés à une lexicalisation.

Les traits valués sont hiérarchisés par une relation est_un. Un trait valué s'insère dans une hiérarchie ou un treillis ; les traits valués réfèrent à une liste de traits valués "pères"en les faisant intervenir dans la relation est_un. Cela signifie que l'on peut hiérarchiser ces traits afin de faire jouer l'héritage de valeurs.

La relation **est_un (***est_un_l***)** permet de hiérarchiser entre eux les traits valués d'un même trait, c'est-à-dire les valeurs de ce trait (ou lorsqu'il s'agit de traits à valeur binaire, de hiérarchiser entre eux les traits valués portant la valeur +). On pourra ainsi exprimer que [humain = +] est dans une relation de hiérarchie avec [animé = +] et que l'affectation de [humain = +] à une unité permet de déduire que [animé = +] est également vrai.

Il faut bien remarquer que les traits valués à valeur binaire, bien que formellement décrits dans la même entité, ont une sémantique un peu différente des traits ayant potentiellement un grand nombre de valeurs comme un trait de classe sémantique. En effet, [humain = +] aura la sémantique : a la caractéristique "humain", [humain = -] signifiant : n'a pas la caractéristique "humain", et l'héritage hiérarchique permet de déduire un ensemble de propriétés "positives" sur une branche, et le rejet des propriétés des autres branches. [humain = +] implique que [humain = -] est impossible.

Une relation **d'incompatibilité** relie certains traits valués, cela s'exprime par le champ *incompatible_l* qui réfère à un ensemble de traits valués qui sont incompatibles avec le Trait valué y référant .

Une relation **d'implication** (*implique_l*) lie certains traits valués. Suivant le type du Trait_sem, le sens de cette relation n'est pas tout à fait le même, voyons pourquoi.

Un trait valué est associé à un Trait_Semantique qui porte un champ "*concerne*" indiquant si ce trait intervient dans la description d'unité sémantique d'affixe (caractérisation sémantique des affixes) ou dans les différents éléments intervenant dans la description sémantique des Usem.

Si le trait portant l'*implique_l* est un Trait_sémantique concernant un affixe, le sens des traits valués portés par *implique_l* est le suivant *: l*es Usem comportant cet affixe dans le mode de dérivation de leur Um et l'affixe ayant le sens correspondant à l'Usem_Aff dans l'Usem, les Traits valués impliqués par le Trait valué D_AFFIXE sont projetés sur le sens de l'élément (Usem si elle existe) faisant intervenir l'affixe. Cette information ne présente pas beaucoup d'intérêt pour les dérivés entièrement décrits dans les trois couches. Mais elle permet aussi (et surtout) de faire une prédiction minimale sur la représentation sémantique de derivés non décrits dans le dictionnaire, mais de formation regulière. L'implication de Trait valués se fait ici entre deux objets différents. (Trait valué porté par une Usem_Aff ==> Trait valué porté par une Usem si elle est présente ou par une Usem "fictive" si elle n'est pas explicitement présente ; ces Usem comportent au niveau morphologique une Um affixe associée à l'Usem_Aff.)

Lorsque la relation d'implication est portée par un Trait_sémantique concernant "AUTRE", il signifie qu'un objet descriptif portant ce Trait_valué n'aura pas besoin de porter explicitement les Traits valués impliqués qu'il portera cependant implicitement. L'implication ici se fait sur le même objet qui porte explicitement un trait valué, et implicitement les traits impliqués par le précédent.

La **correspondance** entre traits à valeur sémantique de la syntaxe et les traits valués de la sémantique se fait par une relation de correspondance. Le champ *trait_synt_corresp* réfère au trait "sémantique" utilisé en syntaxe auquel il correspond, et permet d'en donner la définition complète au niveau sémantique. Ainsi sont mis explicitement en correspondance les traits sémantiques de restriction de la syntaxe avec les traits utilisés pour la représentation sémantique.

Un trait valué peut être associé à une liste de Traits sémantiques qu'il est **pertinent** (trait_sem_pertinent_l) ou **obligatoire** (trait_sem_obligatoire_l) de renseigner. Cela peut permettre par exemple d'associer des traits différenciateurs à renseigner à un trait valué de classe sémantique. C'est un excellent guide pour le lexicographe, et cela clarifie l'usage que l'on attend de certains traits.

Un exemple (classique) illustrera ces traits pertinents :

En supposant que nous ayons un trait valué :

```
classe sémantique = siège
```

on pourra lui associer comme traits sémantiques pertinents :

```
a_un_dossier

a_des_bras

pour_une_personne

confortable
```

. . .

qui permettront de caractériser les Usem de chaise, fauteuil, banc, canapé, tabouret...

6.3. Trait sémantique (Trait_Sem)

Un Trait sémantique s'associe à une valeur pour produire un Trait valué (lequel a un certain nombre de propriétés décrites plus haut).

Un Trait sémantique peut être identifié dans la langue, il réfère alors à son Usem lexicalisante par l'attribut **usem_lexicalise**.

Il peut être lié à la description d'une langue ou pivot, c'est-à-dire choisi comme étant partagé par plusieurs langues, élément de description partagé, c'est le sens de l'attribut **pivot**.

Un Trait_Semantique porte un champ "*concerne*" indiquant si ce trait intervient dans la description d'unité sémantique d'affixe AFFIXE (caractérisation sémantique des affixes) ou dans les différents éléments intervenant dans la description sémantique des Usem AUTRE.

Un trait sémantique a un certain **type.** Les valeurs proposées (l'utilisateur peut étendre cette liste) sont les suivantes :

- PROPRIETE (trait sémantique général)

- CLASSE (trait de classe sémantique)

- DOMAINE (domaine d'activité ou de spécialité)

- DISTINCTIF (en général lié à une classe ou un domaine)

- PRAGMATIQUE

- DIVERS

- CONNOTATION

Trait_sémantique a un **statut** qui peut être **MONOVALUE**, (ce qui signifie qu'un élément ne peut porter qu'un Trait_valué associé à ce Trait_sémantique, car les valeurs sont exclusives les unes des autres de fait) ou **MULTIVALUE** (ce qui signifie que plusieurs Traits valués associés au même Trait sémantique peuvent être portés par un même objet descriptif de la couche sémantique).

Un trait sémantique admet un certain type de valeurs, précisé par l'attribut **type_liste_valeurs**. Les différents types de valeurs sont les suivants :

BINAIRE (+/-),

LISTE_FERMEE (liste de valeurs connue en extension et précisée),

LISTE_OUVERTE.

La liste des valeurs (quand elle est de type LISTE_FERMEE) doit être précisée par **valeurtrait_l** qui référe aux entités Valeur_trait.

L'ensemble des traits valués associés à un même trait sémantique peut être structuré. La structuration des différents Traits valués associés à un Trait_sémantique à valeurs non binaires est indiquée par l'attribut **structure_liste_valeurs**. Cette structuration s'exprime localement par la relation **est-un** entre Traits valués.

(les traits binaires ont un statut à part ; ils seront structurés entre eux également, avec héritage de certains traits valués et exclusion d'autres traits valués, une valeur + permettant d'exclure la valeur - sur le même trait)

Trait_sémantique est en partie caractérisé par la structure associée :

- TREILLIS_TOTAL : l'ensemble des Trait_valués associés forme un treillis

-TREILLIS_PARTIEL : les Traits valués entrent localement dans des structures de treillis, plusieurs treillis étant autorisés (et donc certains Traits valués peuvent être isolés)

-HIƒRARCHIE_TOTALE : l'ensemble des Traits valués associés s’inscrit dans un arbre qui structure la totalité des valeurs.

-HIƒRARCHIE_PARTIELLE : l'ensemble des Traits valués associés s’inscrit dans un ou plusieurs arbres qui structurent localement les valeurs. Tous les Traits valués ne sont pas nécessairement dans une relation structurante.

Un trait sémantique peut être associé à une liste de Traits sémantiques qu’il est **pertinent** (trait_sem_pertinent_l) ou **obligatoire** (trait_sem_obligatoire_l) de renseigner.

7. Relation valuée pondérée

7.1. Généralités

Le modèle GENELEX offre la possibilité de relier entre eux différents objets par des relations sémantiques. Le choix de modélisation a été de pondérer systématiquement les mises en relations entre objets et donc d'introduire la notion de relation valuée pondérée.

Une relation valuée pondérée est donc un objet de description porté par son objet source, comportant un objet cible, une relation sémantique qui relie la source et la cible, et la modalité de cette mise en relation.

Les relations valuées pondérées sont de différents types suivant qu'elles concernent :

- une cible et une source Usem

- une cible et une source Prédicat

- une cible et une source Concept

- une cible Prédicat et une source Concept

- une cible Concept et une source Prédicat

On l'a vu plus haut, le passage entre le niveau Usem et les niveaux plus abstraits se fait par la structure prédicative qui met en relation Usem et prédicat, d'une part, et par les représentations conceptuelles qui mettent en relation Usem et Concept d'autre part. Mis à part ces objets, les relations sémantiques du modèle ne mélangent pas les niveaux de représentation, les relations sémantiques sont donc au niveau purement lexical entre Usem ou entre objets d'une nature plus abstraite.

Suivant la nature des objets source et des objets cible, les relations sémantiques seront distinctes, et elles auront des propriétés parfois différentes également.

Nous ne passerons pas ici en revue les différents types de relations valuées pondérées ni les différents types de relations sémantiques, il suffit de savoir que ces types sont caractérisés par la nature de leur source et de leur cible (les relations sémantiques suivent le même typage). On pourra se référer à la DTD SGML pour voir de plus près les détails du modèle. Le modèle entité relation est également suffisamment parlant.

7.2. Relation sémantique

Les relations sémantiques sont de différentes natures en fonction des objets qu'elles relient, mais aussi du point de vue de leur contenu.

7.3. Relation sémantique entre Usem (R_Usem)

On s'intéressera ici un peu plus en détail aux relations sémantiques qui lient des Usem (par le biais de Relations valuées pondérées entre Usem)

Ces relations peuvent préciser :

* la **catégorie** morpho-syntaxique de l'Usem **source**

* la **catégorie** morpho-syntaxique de l'Usem **cible**

* le **type** de la relation

L’ensemble de types suivant est proposé par le modèle GENELEX (cet ensemble peut être étendu par l'utilisateur) :

- PARADIGMATIQUE

- DERIVATION

- COLLOCATION

* le **sous-type** de la relation, pour caractériser plus finement la relation sémantique.

Les valeurs suivantes sont proposées (l'utilisateur peut étendre cette liste) :

-SYNONYMIE

- CONTRAIRE

- OPPOSITION

- CONVERSE

- TAXINOMIE

- PARTIE_TOUT

(sous-types associés au type PARADIGMATIQUE)

- STRICTE

- NON_STRICTE

(sous-types associés au type DERIVATION)

Une relation sémantique peut avoir un certain nombre de propriétés qui sont celles des relations binaires: **reflexivité**, **transitivité**, **symétrie, antisymétrie, relation d'ordre**

Une relation sémantique entre Usem peut conna"tre la relation sémantique inverse (source => cible, cible =>source)

Certaines relations sémantiques sont incompatibles, une relation sémantique peut conna"tre les relations avec lesquelles elle est incompatible.

Les relations sémantiques peuvent être organisées dans une structure hiérarchique (hiérarchie partielle) qui s'exprime par la relation est_un entre celles-ci. L'attribut **est_un_l** précise la structuration hiérarchique entre relations, en référant les R_usem "mères" de la relation décrite .

7.4. Relation sémantique entre les autres objets (R_Pred, R_Concept, R_Pred_Concept, R_Concept_Pred)

Les autres relations sémantiques ont un certain nombre de propriétés en commun avec les relations entre Usem : relation inverse, relations incompatibles, structuration hiérarchique par relation est_un, équivalence.

Elles peuvent être indépendantes de la langue ou non, c'est le sens de l'attribut **pivot.**

Elles ont également un type dont les valeurs sont proposées (la liste peut être étendue par l'utilisateur) :

- GƒNƒRALISATION

- PARTICULARISATION

(pour celles dont la source et la cible sont de même nature)

- ESSENTIEL

et un sous-type dont les valeurs sont totalement libres.

Les relations de généralisation/particularisation sont importantes car elles structurent en partie les objets du plus lexical vers le plus abstrait, l'héritage de traits valués pondérés devant pouvoir jouer (il s'agit d'un choix de stratégie lexicographique).

7.5. Correspondance d'arguments (Corresp_Arg_Arg)

Lors des mises en relation sémantique de Prédicat ou d'Usem prédicatives, sont en relation en fait deux structures prédicatives et il est important de pouvoir mettre en correspondance les arguments de prédicat en jeu en tant que source ou cible d'une relation valuée pondérée. C'est le rôle des correspondances d'arguments, qui associe deux arguments de la façon suivante :

les chemins vers les Argument source et cible sont indiqués : chemin_arg_source et chemin_arg_cible

Certaines informations supplémentaires peuvent être ajoutées dans la correspondance : elles peuvent concerner l'argument source (informe_arg_precise_source_l) ou l'argument cible (informe_arg_precise_cible_l) sous la forme d'InformeArg. C'est ainsi que l'on peut contraindre les interprétations sémantiques acceptables de certains arguments dans la correspondance, et aussi que l'on peut préciser les valeurs d'arguments absents dans la mise en correspondance de prédicats (lorsque les prédicats n'ont pas le même nombre d'arguments).

8. Correspondance syntaxe sémantique (Corresp_Usyn_Usem)

La correspondance entre syntaxe et sémantique se fait par des éléments de correspondances (**Corresp_Usyn_Usem**) portées par les Usyn. Pour une Usyn, il y a autant de Corresp_Usyn_Usem que d'Usem pointées par une Usyn. Une Usem participe à autant de Corresp_Usyn_Usem qu'elle a de contextes syntaxiques associés (chacun étant décrit par l'Usyn comportant la Corresp_Usyn_Usem).

Une Usyn correspond à une ou plusieurs Usem et une Usem peut également correspondre à une ou plusieurs Usyn ; lorsqu'on met en correspondance une Usyn et une Usem, on peut avoir besoin d'informations complémentaires sur les modalités de cette correspondance, c'est ce que permet d'expliciter Correspondance.

8.1. Correspondance

Correspondance donne accès à ces informations complémentaires, à savoir :

- informations permettant de préciser l'expression en surface d'un argument de prédicat au niveau sémantique (en simplifiant, il s'agit de faire se correspondre des arguments de prédicat et des positions de construction). Cette information est portée par **des Correspondances Argument-Position** qui sont simples (**Corresp_Arg_Pos_Simple**) quand les positions sont présentes dans la description de la couche syntaxique, ou associées à des "positions flottantes", (**Corresp_Arg_Pos_Flottant**). Il y a autant de Corresp_Arg_Pos_(Simple ou Flottant) que d'arguments à préciser.

- des informations de filtrage sur la description syntaxique, permettant de restreindre, du point de vue de la syntaxe ou de la sémantique, le sous-ensemble des réalisations de surface autorisées par l'Usem : c'est le sens de l'objet **ContraintDescription** qui décrit les restrictions portant sur la description de base de l'Usyn.

- des informations concernant l'apport sémantique "implicite" propre à la construction ; cet apport concerne les arguments du prédicat associé à l'Usem prédicative. Ces informations s'expriment par des **InformeArg.**

8.2. Correspondance Argument-Position simple et flottante (Corresp_Arg_Pos_Simple Corresp_Arg_Pos_Flottant)

Corresp_Arg_Pos_Simple permet de conna"tre pour un argument du prédicat ce à quoi il correspond en surface, en fonction de la description syntaxique de l'Usyn qui comporte la réalisation de l'argument.

Corresp_Arg_Pos_Simple associe à un argument de la couche sémantique une position de la syntaxe, en donnant accès à chacun de ces deux éléments.

Corresp_Arg_Pos_Simple permet de pointer sur une Position de la syntaxe, qu'elle soit Position de la construction externe, de la structure interne, ou d'une composition. L'attribut *portee* permet de préciser cette information.

Le CheminPosition est parcouru à partir de la construction externe, de la structure interne, ou de la construction d'une composante. En effet, en cas de composition, on précise la composition concernée (*nieme_composition*) et, dans cette composition, la composante concernée (*nieme_composante*), le CheminPosition devant être parcouru à partir de la construction externe de la composante. Les attributs nieme_composition et nieme_composante n'interviennent donc qu'en cas de composition.

L'attribut chemin_arg permet d'accéder à l'Argument mis en correspondance avec une Position. Il est identifié par une suite de rangs (le plus souvent un seul) qui à chaque pas permet d'accéder à un

ième argument, qui peut être simple, auquel cas le chemin s'arrête nécessairement, ou être lui-même une structure prédicative, auquel cas on accède au ième argument de cette structure par le rang suivant dans la liste.

Corresp_Arg_Pos_Flottant permet d'associer à un Argument essentiel du prédicat sa famille de réalisations syntaxiques lorsque celle-ci correspond à un élément non décrit au niveau de la syntaxe dans la description de base car non essentiel en tant qu'élément de la syntaxe. La Position absente de la Construction de base (donc de la description faite en syntaxe) doit donc être décrite depuis la sémantique, et insérée dans la description syntaxique.

Certains Arguments de prédicats associés à des "compléments adjoints": circonstants, modifieurs... non precisés pour l'Usyn, mais que l'on considère en sémantique comme devant être décrits, le seront gr‰ce à des Corresp_Arg_Pos_Flottant. Cela permet une plus grande indépendance des codages des couches syntaxique et sémantique.

L'attribut *position*: pointe vers une Position qui donne la description de la famille de réalisations en syntaxe (et précisera donc éventuellement fonction, rôles thématique et toujours une distribution).

CheminSyntagme permet de préciser en cas de réécriture le niveau de l'insertion virtuelle de la Position associée à l'Arg_flottant en pointant sur le syntagme réécrit.

L'insertion de la position flottante se fait alors dans la liste de réécriture de ce syntagme, la fonction et les rôles thématiques sont précisés par rapport à la tête de ce syntagme. Le rang d'insertion dans cette liste ne sera pas précisé, il s'agira d'une position flottante.

L'absence de CheminSyntagme implique que l'insertion se fait au niveau le plus haut, c'est-à-dire suivant le cas (précisé par l'attribut *portee*) dans la liste de Positions de la Construction externe (cas le plus fréquent), de la structure interne (en cas de composition decrite par un syntagme de structure) ou d'un des éléments de la composition (precisé par *nieme_composition* et *nieme_composante* qui interviennent dans le cas d'une composition décrite par un calcul de composition, auquel cas il ne peut y avoir de CheminSyntagme : on pointera au bon niveau en choisissant la composante voulue.

8.3. ContraintDescription

Un **ContraintDescription** permet de restreindre l'ensemble des réalisations syntaxiques parmi celles que décrit virtuellement la description de base de l'Usyn et de ne conserver que celles qui sont compatibles avec l'acception identifiée par l'unité sémantique.

Ce filtrage se fait en contraignant les réalisations de Self (0 ou un ContraintIntervConst), en contraignant la construction externe (0 ou un ContraintConstruction), ou la structure interne (0 ou un ContraintStructInterne et 0 ou un Contraint_mdc) suivant les choix d'instanciation du modèle qui auront été faits. Plusieurs de ces éléments peuvent être présents en même temps, cependant on ne peut avoir simultanément ContraintStructInterne et Contraint_mdc que si les contraintes exprimées sont compatibles et cohérentes.

Par ailleurs, au moins un de ces éléments doit être présent, un ContraintDescription ne peut être complètement vide d'information (en effet, il s'agit d'un élément optionnel de la Correspondance).

8.3.1. ContraintIntervConst

Par **ContraintIntervConst,** on peut choisir de restreindre les réalisations possibles de Self à 1 ou n syntagmes (parmi ceux que Self comprend comme intervenant dans la construction).

ContraintIntervConst pointe sur autant de ContraintSyntagme qu'il y a de syntagmes à contraindre en les inhibant ou en rajoutant des traits restrictifs (ceux du modèle de la couche syntaxique) ou des caractéristiques sémantiques.

Lorsque **ContraintIntervConst** n'est pas présent, c'est que toutes les réalisations de Self comme intervenant dans construction sont autorisées.

8.3.2. ContraintConstruction

ContraintConstruction contraint la construction externe en contraignant une ou plusieurs de ses positions, en la rendant obligatoire ou interdite (si la position est optionnelle dans la Construction), ou en contraignant sa distribution. Il y a autant de ContraintPosition que de positions contraintes dans la Construction, qu'elles soient inhibées ou rendues obligatoires, ou que leur distribution soit contrainte, et qu'elles se trouvent à un niveau de profondeur quelconque ; le CheminPosition des ContraintPosition est à parcourir à partir de la Construction de base.

8.3.3. ContraintStructInterne

ContraintStructInterne joue un rôle tout à fait analogue à celui de ContraintConstruction ; la seule différence est qu'il s'agit de positions de la structure interne. Il y a autant de ContraintPosition que de positions contraintes dans la structure interne ; le CheminPosition des ContraintPosition est à parcourir à partir de la Structure Interne.

8.3.4. Contraint_mdc

Contraint_mdc sert à contraindre la mise en correspondance avec la sémantique des composés décrits par Mode de Composition. **Contraint_Position_mdc** précise, pour chaque position contrainte, de quelle composante de quelle composition elle est position. La position une fois identifiée est contrainte par un ContraintPosition tout comme les positions de Construction ou de Structure interne. Il y a autant de ContraintPosition_mdc que de positions contraintes dans le mode de composition.

8.3.5. ContraintPosition

ContraintPosition permet d'accéder à une position (de la construction externe, ou de la structure interne ou de la composition suivant l'élément qui fait intervenir le ContraintPosition) par un CheminPosition (de la syntaxe) qui est parcouru à partir de la construction de base, de la structure interne, ou d'une composante (en cas de description d'Usyn par Composition).

modif_optionnalité permet de préciser qu'une position optionnelle dans la description devient obligatoire ou au contraire interdite pour l'acception (usem) pointée.

Une liste de syntagmes contraints (ContraintSyntagme) permet de préciser dans la distribution de la position contrainte les syntagmes et les contraintes supplémentaires qui leur sont associées. Il y aura

autant de ContraintSyntagme que de syntagmes sur lesquels on veut apporter des restrictions. La distribution n'est jamais modifiée en rel‰chant des contraintes ou en rajoutant de nouveaux syntagmes, toujours en contraignant davantage.

D'autre part, un syntagme peut être filtré ou enrichi d'un point de vue sémantique par des traits de la sémantique.

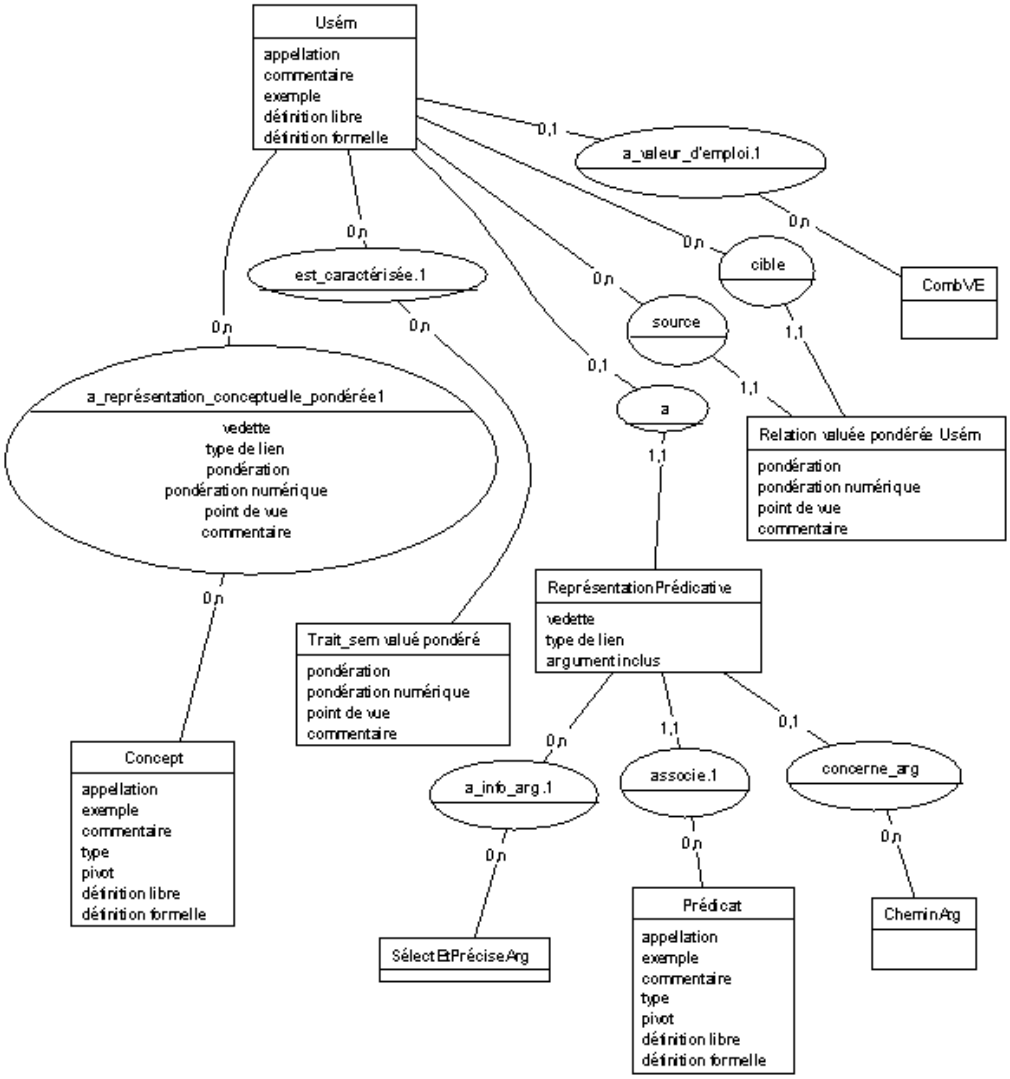
8.3.6. ContraintSyntagme

ContraintSyntagme permet de sélectionner un syntagme (d'une distribution) et de préciser s'il est inhibé et devient donc absent de la distribution associée à l'acception. S'il n'est pas inhibé, il peut être restreint par l'ajout de traits restrictifs de la syntaxe. Il peut être contraint sur le critère de l'information sémantique associée à son interprétation sémantique également (par AjouteTrait_Sem).

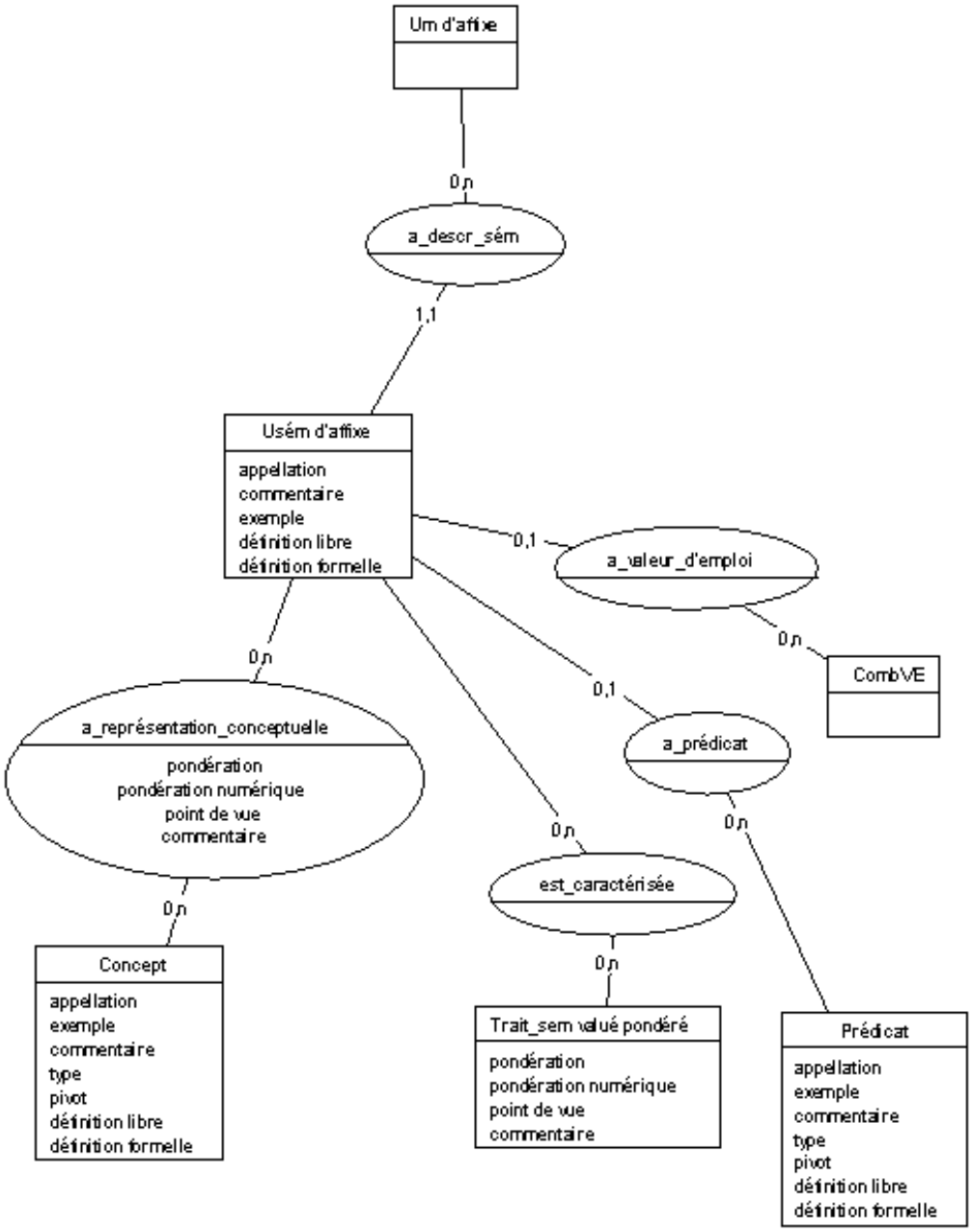
AjouteTrait_Sem permet de filtrer sur l'interprétation sémantique associée au syntagme en vérifiant la présence d'un trait qui doit nécessairement être présent pour autoriser la correspondance (statut FILTRE) ou qui peut être présent ou ajouté à condition d'être compatible avec la représentation sémantique (statut FILTRE_AJOUTE). **AjouteTrait_Sem** peut aussi enrichir systématiquement (sans vérification) la représentation sémantique associée (statut FORCE).

G - Schémas entitéS- relations

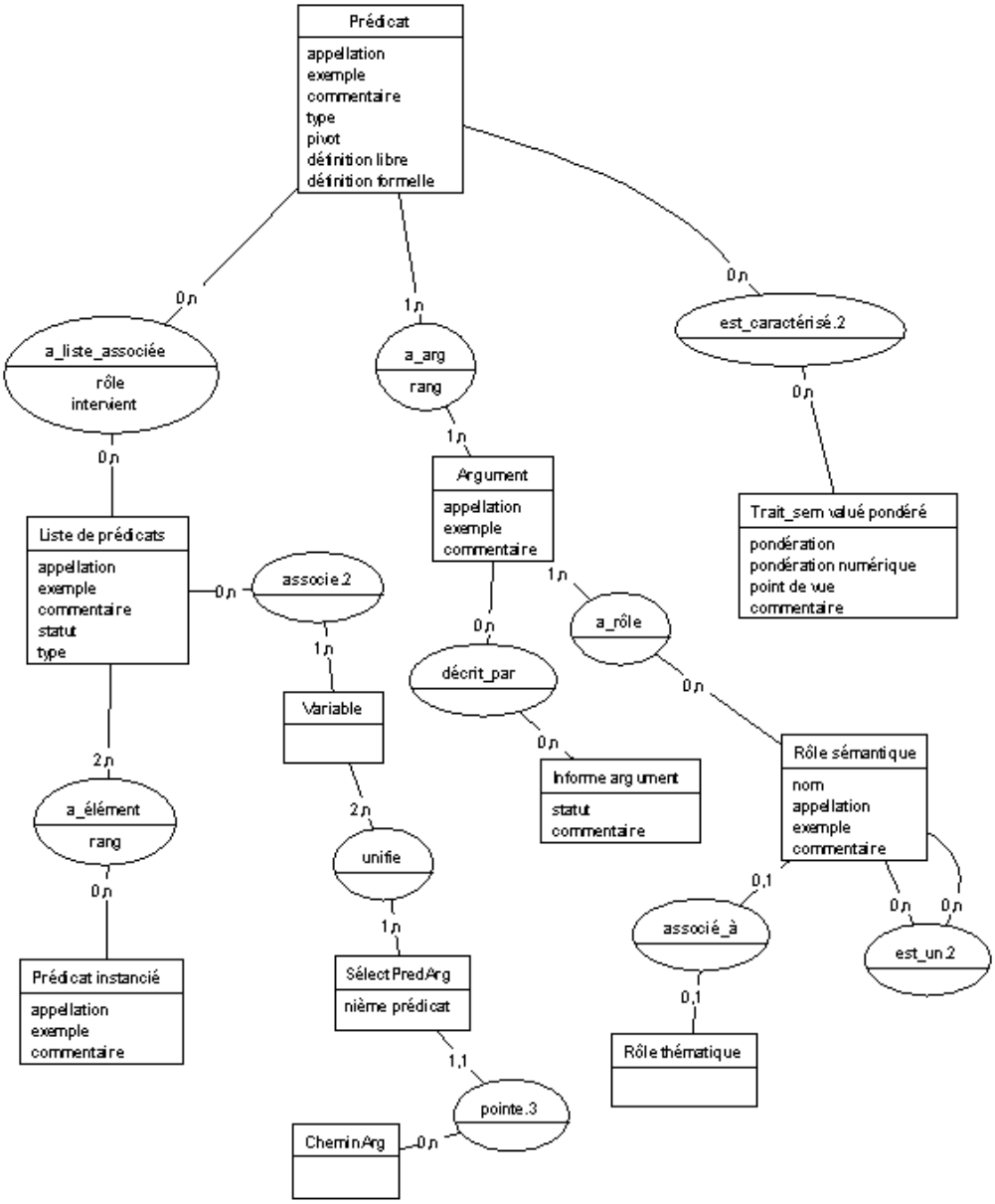
1. Unité sémantique, Représentation prédicative, Représentation conceptuelle



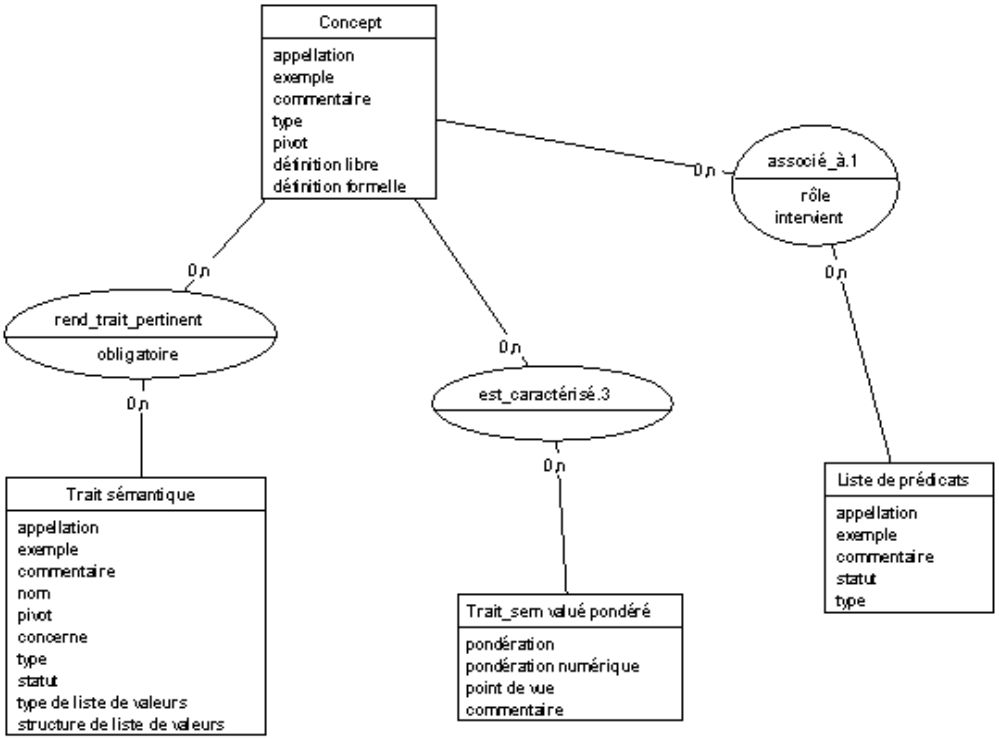
2. Usem d'affixe (Usem_Aff)



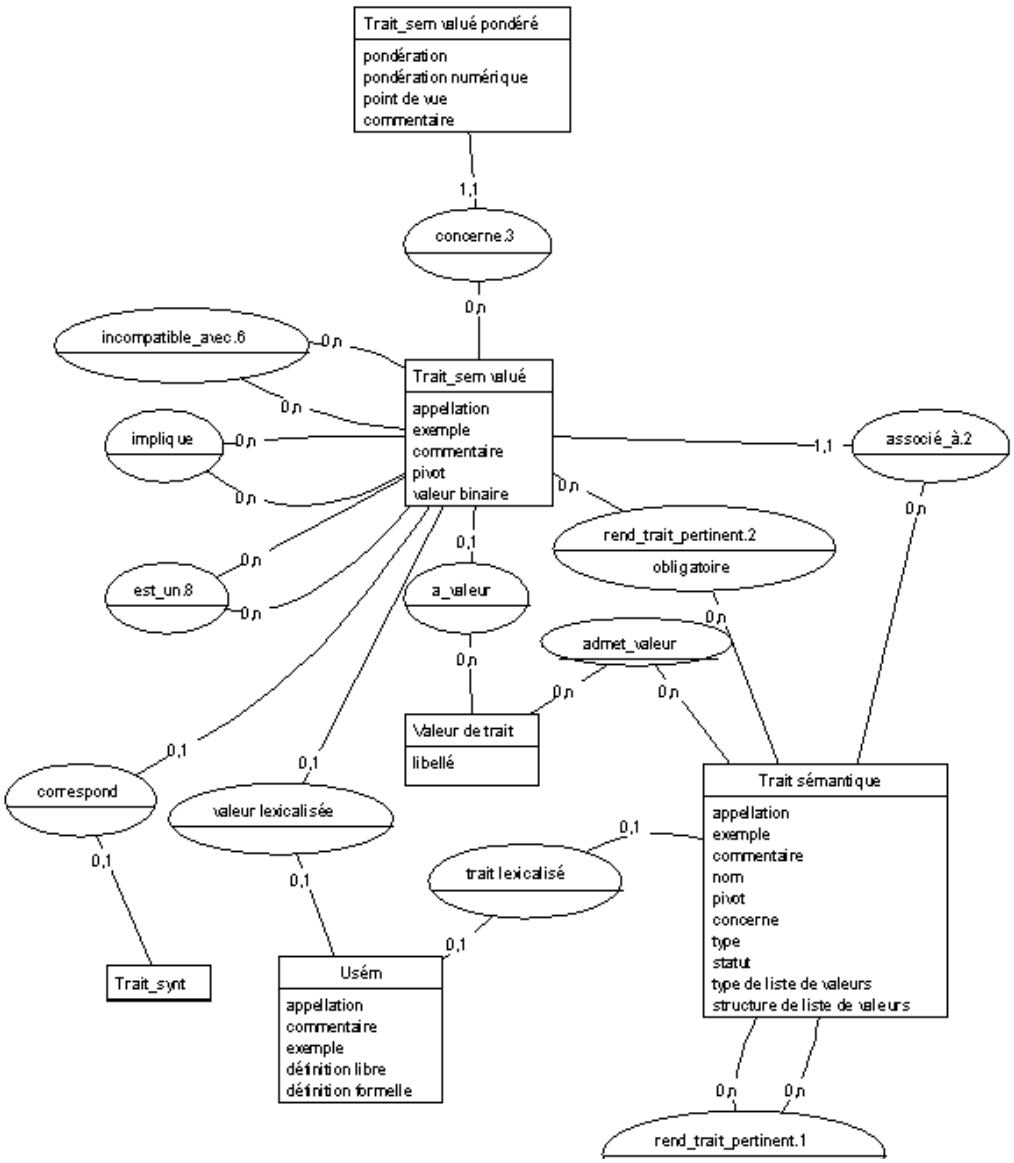
3. Prédicat, Liste prédicats, Argument, Variable, Rôle sémantique

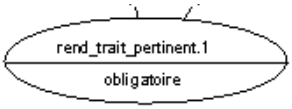

Prédicat
appellation
exemple
commentaire
type
pivot
définition libre
définition formelle
0,n
1,n
0,n
a_liste_associée
rôle
intervient
a_arg
rang
est_caractérisé.2
0,n
1,n
0,n
Argument
appellation
exemple
commentaire
Trait_sem valué pondéré
pondération
pondération numérique
point de vue
commentaire
Liste de prédicats
appellation
exemple
commentaire
statut
type
0,n
associe.2
1,n
1,n
a_rôle
0,n
décrit_par
Variable
0,n
Informe argument
statut
commentaire
2,n
a_élément
rang
0,n
Rôle sémantique
nom
appellation
exemple
commentaire
2,n
unifie
0,1
0,n
0,n
associé_à
est_un.2
Prédicat instancié
appellation
exemple
commentaire
1,n
SélectPredArg
nième prédicat
0,1
Rôle thématique
1,1
pointe.3
CheminArg
0,n

4. Concept

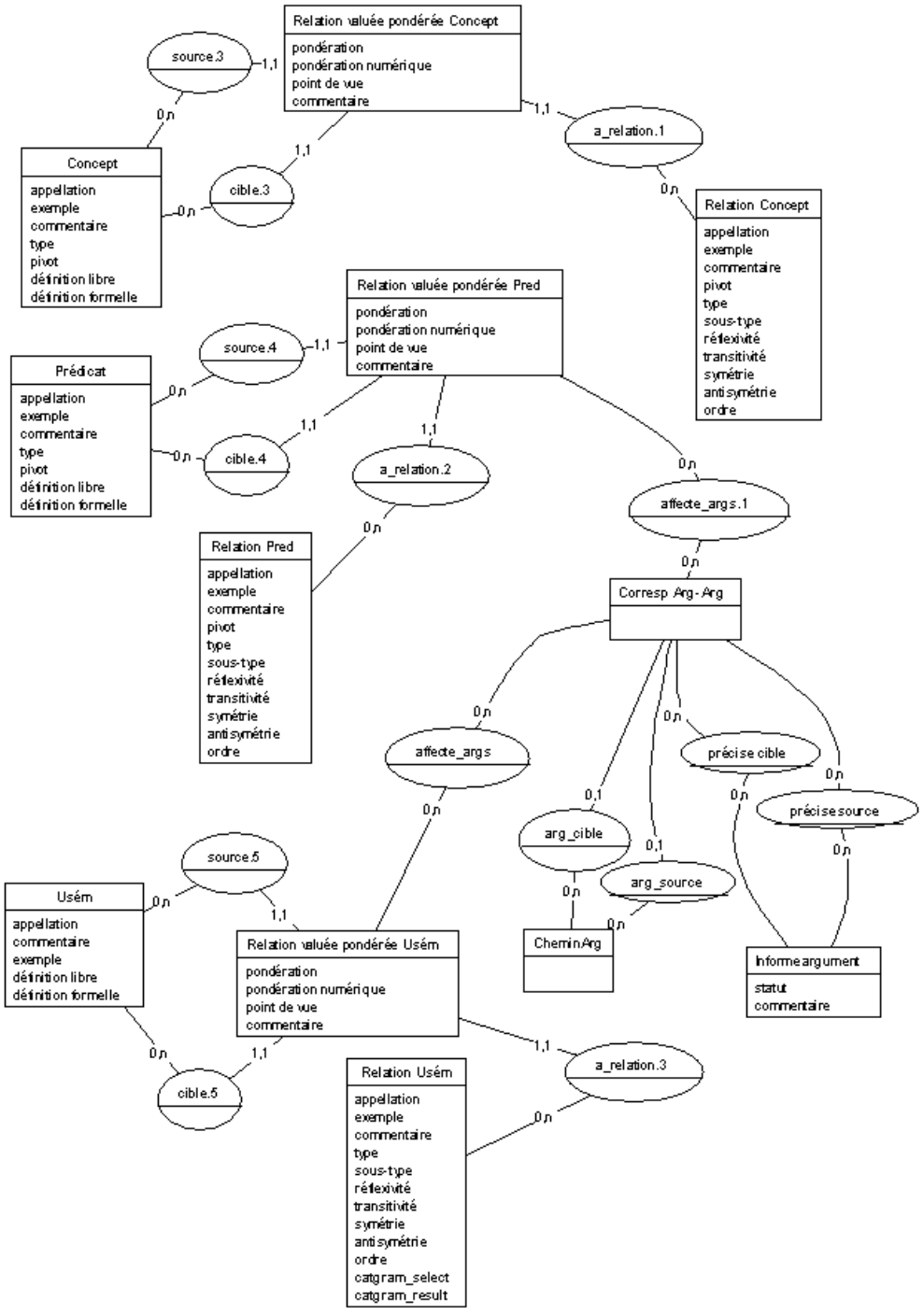


5. Trait valué pondéré, Trait valué, Valeur_trait, Trait sémantique,

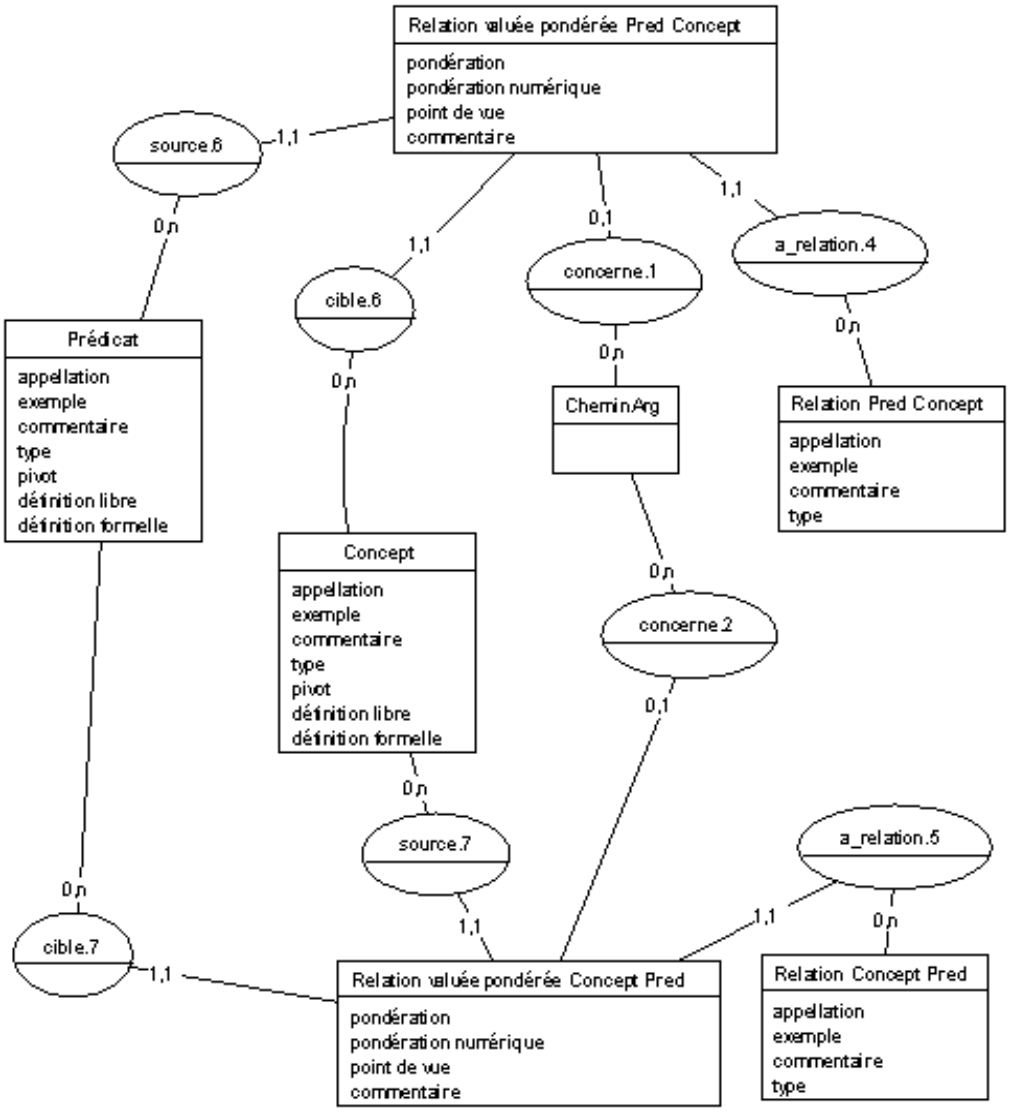




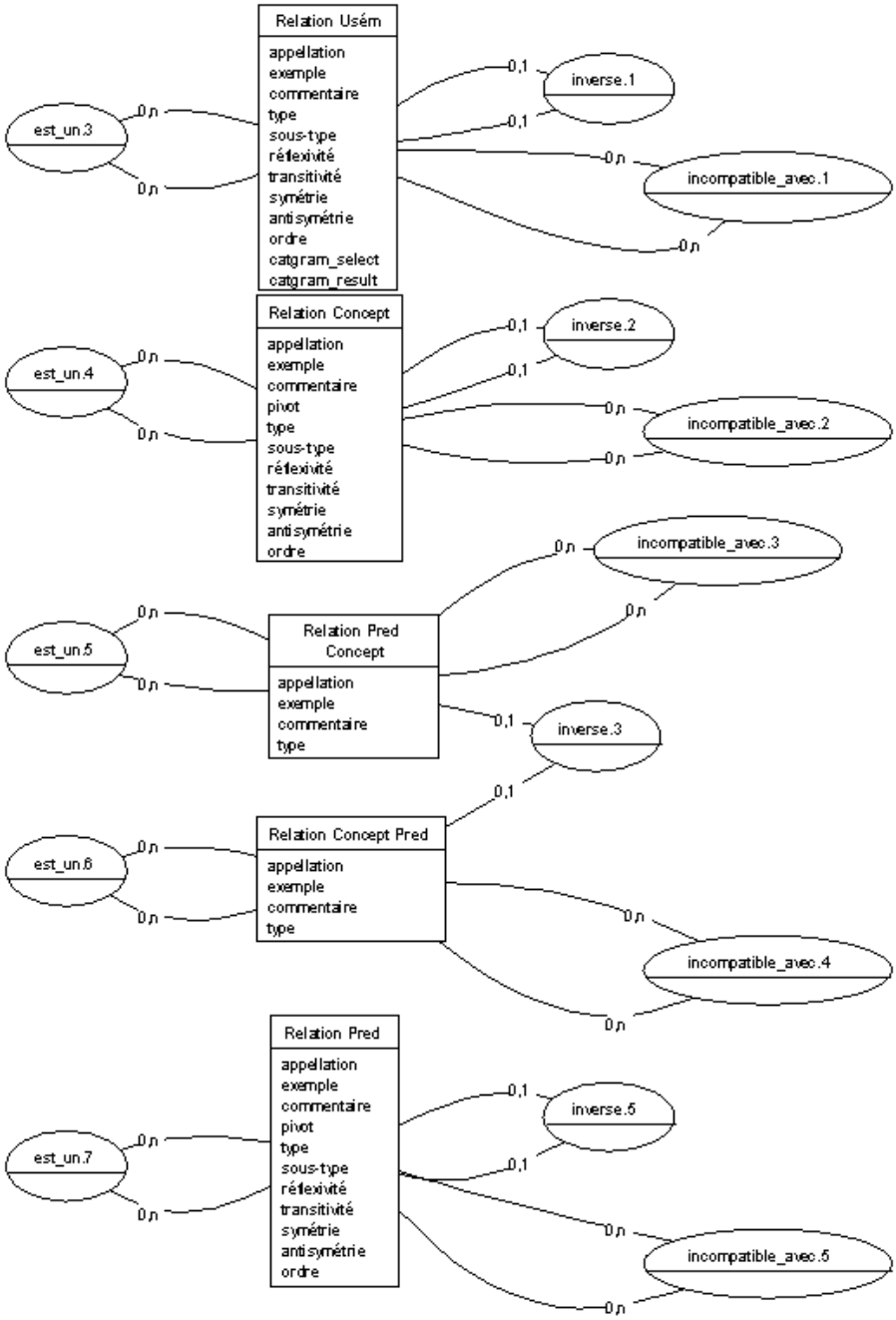
6. Relations valuées pondérées(-pred, -concept, -Usem)



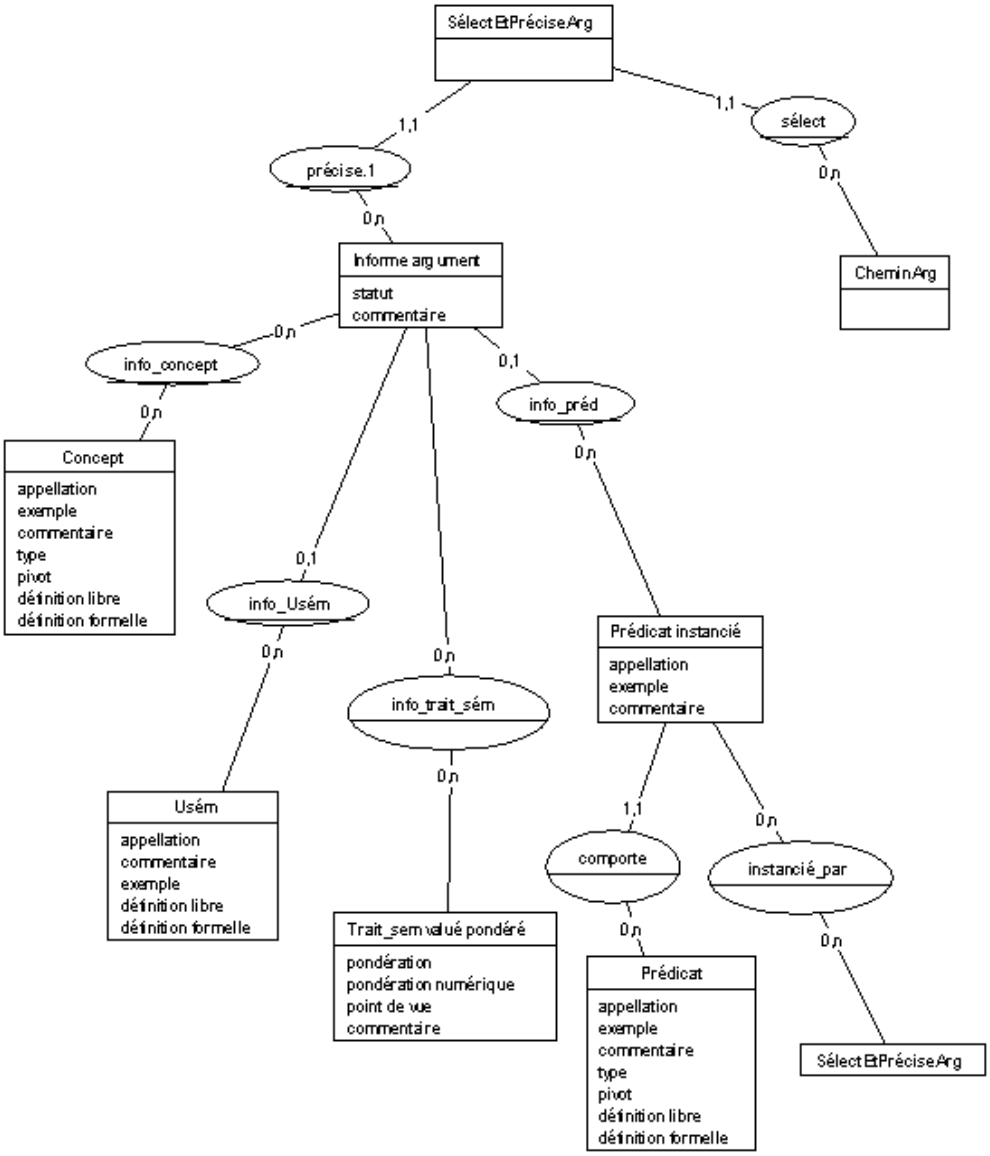
7. Relations valuées pondérées (-pred-concept, -concept-pred)



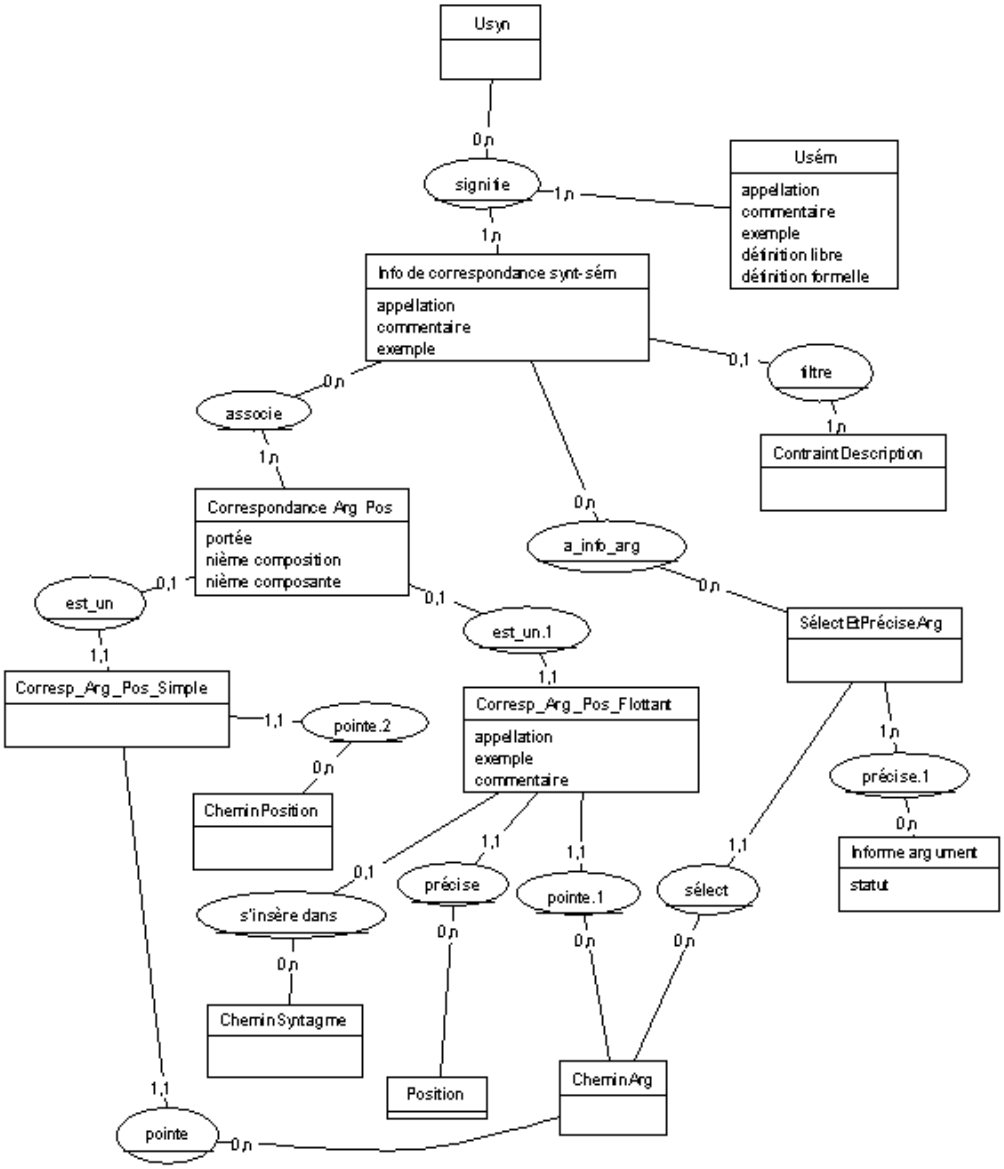
8. Relations sémantiques (Usem-Usem, pred-pred, concept-concept, pred-concept, concept-pred)



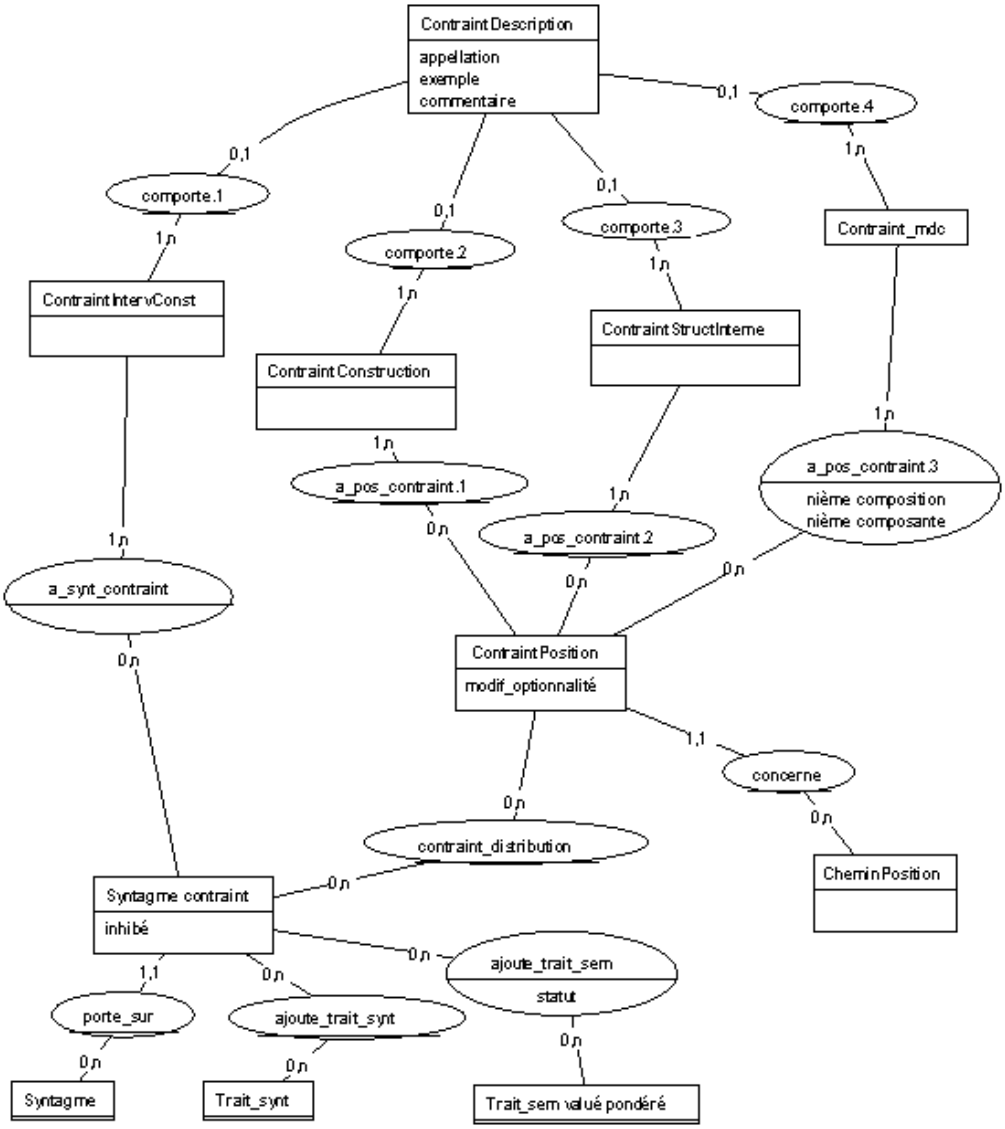
9. SelectEtPréciseArg, Informe argument, Prédicat instancié



10. Correspondance syntaxe sémantique,



11. Contraint description, Contraint IntervConst, Contraint mdc, Contraint construction, Contraint struct interne, Contraint position, Contraint syntagme



H- DTD SGML

I - Introduction - Traduction du Modèle Conceptuel

Le modèle conceptuel GENELEX a été largement exprimé au travers de modèles Entité-Attribut-Relation (Merise).

Beaucoup de contraintes d'intégrité sont exprimées dans ce formalisme : typage des objets, typage des relations, cardinalité des relations, etc. Cependant, ce modèle n'étant pas fait pour exprimer des règles - à moins de complications extrêmes-, certaines contraintes ont dž être exprimées dans le document d'accompagnement (restriction sur les combinatoires de valeurs). Il s'ensuit que le modèle Conceptuel de GENELEX combine l'utilisation du formalisme Entité-Attribut-Relation (EAR) et de commentaires en langage naturel.

Une DTD (Définition de Type de Document) SGML est un modèle physique du type grammaire qui décrit le marquage des données.

Lors du passage du Modèle Conceptuel à la DTD GENELEX, nous nous sommes efforcés de traduire de manière systématique les modèles EAR et avons tenté d'exprimer formellement la plupart des contraintes d'intégrité décrites en langue naturelle.

Certaines règles de traduction du formalisme EAR vers SGML ont été mises en Ïuvre :

(i) Les Entités EAR deviennent des Eléments SGML.

(ii) Les Attributs d'Entités EAR deviennent des Attributs d'Eléments.

Lorsque les valeurs d'un Attribut sont exclusives les unes des autres et qu'elles constituent un vocabulaire fermé, ces valeurs sont représentées sous forme d'Attributs SGML listés.

(iii) Les Relations non attribuées pointant sur une Entité EAR non partagée sont traduites par des liens de hiérarchie entre les Eléments de la DTD. Leur cardinalité est traduite par les indicateurs d'occurrences SGML : **? + ***

(iv) Les Relations non Attribuées pointant sur une Entité EAR partagée sont exprimées par des liens de référence entre les Eléments.

(v) Les Relations Attribuées sont traduites par des Eléments SGML attribués mis en relation - hiérarchie ou référence - avec les Eléments traduisant les Entités EAR.

Un fichier de contraintes ("sémant.ctr") a été créé pour faciliter la lecture des références croisées. Ces contraintes apparaissent comme des commentaires et seront donc ignorées par un parser SGML ; elles expriment dans une syntaxe intuitive le typage des références.

II - DTD GENELEX commentée

1. DTD *genelex.dtd*

```
<!--Consortium GENELEX @(#) genelex.dtd 3.1@(#) 94/02/25 18:26:42 -->

<!-- **************A L'ADRESSE DES UTILISATEURS ******************

Vos remarques concernant la DTD seront etudiees par le consortium

GENELEX. Celui-ci assurera la diffusion de la nouvelle version qui

pourrait en decouler.

********************************************************** -->

<!DOCTYPE Genelex [

<!ENTITY % ISOlat1 PUBLIC "ISO 8879-1986//ENTITIES Added Latin 1//EN">
```

```
%ISOlat1

<!ENTITY % CustEnt PUBLIC "-//GLX-TEAM//ENTITIES Custom Entity Set//FR">

<!ENTITY % MorpEnt PUBLIC "-//GLX-TEAM//ENTITIES Morphologie Entity Set//FR">

<!ENTITY % SyntEnt PUBLIC "-//GLX-TEAM//ENTITIES Syntaxe Entity Set//FR">

<!ENTITY % SemEnt PUBLIC "-//GLX-TEAM//ENTITIES Semantique Entity Set//FR">

%CustEnt

%MorpEnt

%SyntEnt

%SemEnt

<!--

Un document Genelex est compose de plusieurs parties:

- la description morphologique

- la description syntaxique

- ...

Pour selectionner la partie souhaitee, il suffit de specifier le

mot cle approprie (INCLUDE or IGNORE) dans les declarations

d'entite suivantes :

-->

<!ENTITY % isMor "INCLUDE" >

<!ENTITY % isSyn "INCLUDE" >

<!ENTITY % isSem "INCLUDE" >


<!ELEMENT Genelex - O (GenelexMorpho? & GenelexSyntaxe?

& GenelexSemant? & CombVE*)>

<!ATTLIST Genelex

nom CDATA #REQUIRED
```

```
langue CDATA #REQUIRED
version CDATA #IMPLIED
date_creation1 CDATA #IMPLIED
date_creationglx CDATA #IMPLIED
date_modif CDATA #IMPLIED
propriete CDATA #IMPLIED
copyright CDATA #IMPLIED
integrite (SANS_B|%pBooleen) SANS_B
certification CDATA #IMPLIED>
<!-- ********************************************************** -->
<!ENTITY % pGlose
"appellation CDATA #IMPLIED
exemple CDATA #IMPLIED
commentaire CDATA #IMPLIED">
<!-- D'une maniere generale dans tout le fichier :
- appellation : permet de nommer de facon comprehensible
et si possible univoque l'objet
- exemple : permet d'illustrer l'emploi (citation d'auteur,
exemple de dictionnaire ou de linguiste)
- commentaire : champ libre pour l'utilisateur
-->
<!-- ********************************************************** -->
<!ELEMENT CombVE - O EMPTY>
<!ATTLIST CombVE
id ID #REQUIRED
```

```
datation (SANS_D|%pDatation) SANS_D

niveaulgue (SANS_NL|%pNiveauLgue) SANS_NL

frequence (SANS_F|%pFrequence) SANS_F

vargeog CDATA #IMPLIED>


<![ %isMor [

<!ENTITY % GLXmor PUBLIC

"-//GLX-TEAM//DTD Description Morphologie//FR">

<!ENTITY % MorpCtr PUBLIC

"-//GLX-TEAM//DTD Contraintes Morphologie//FR">

%GLXmor

%MorpCtr

]]>

<![ %isSyn [

<!ENTITY % GLXsyn PUBLIC

"-//GLX-TEAM//DTD Description Syntaxe//FR">

<!ENTITY % SyntCtr PUBLIC

"-//GLX-TEAM//DTD Contraintes Syntaxe//FR">

%GLXsyn

%SyntCtr

]]>

<![ %isSem [

<!ENTITY % GLXsem PUBLIC

"-//GLX-TEAM//DTD Description Semantique//FR">

<!ENTITY % SemaCtr PUBLIC

"-//GLX-TEAM//DTD Contraintes Semantique//FR">
```

```
%GLXsem

%SemaCtr

]]>

]>
```

2. DTD *semant.dtd*

```
<!--Consortium GENELEX @(#) semant.dtd 2.1@(#) 94/09/21 17:30:20 -->

<!-- **************A L'ADRESSE DES UTILISATEURS ******************

Vos remarques concernant la DTD seront etudiees par le consortium

GENELEX. Celui-ci assurera la diffusion de la nouvelle version qui

pourrait en decouler.

********************************************************** -->

<!ELEMENT GenelexSemant - O (

Usem+

& Usem_Aff*

& Predicat*

& Argument*

& InformeArg*

& PredInstancie*

& Role_Sem*

& Concept*

& Trait_Sem_ValPond*

& Trait_Sem_Value*
```

```
& ValeurTrait*
& Trait_Sem*
& R_Usem*
& R_Pred*
& R_Concept*
& R_Pred_Concept*
& R_Concept_Pred*
& ListePred*
& Correspondance*
& ContraintDescription*
& ContraintIntervConst*
& ContraintSyntagme*
& ContraintConstruction*
& ContraintStructInterne*
& ContraintPosition*
& Corresp_Arg_Pos_Simple*
& Corresp_Arg_Pos_Flottant* )>
<!ENTITY % pModalite
"ponderation (SANS_POND
|%pPonderation
|%pPonderation_cust) SANS_POND
ponderation_num NUMBER #IMPLIED
point_de_vue CDATA #IMPLIED">
<!-- Modalite permet de marquer la modalite de l'element de
description semantique qui la porte: ce sera principalement
```

```
un Trait_Sem_ValPond ou une R_ValPond_Sem_...
Cette modalite s'exprime par differents attributs :
ponderation symbolique dont certaines valeurs sont proposees
par le modele et dont la liste des valeurs peut etre personnalisee,
ponderation numerique,
point de vue qui permet de marquer un point de vue particulier
sur le dictionnaire: que ce soit le point de vue general ou d'un
domaine particulier.
-->
<!-- ********************************************************** -->
<!-- ***** UNITE SEMANTIQUE ***** -->
<!-- ********************************************************** -->
<!ENTITY % pUsemAtt
"%pGlose
definition_libre CDATA #IMPLIED
definition_formelle CDATA #IMPLIED
combve IDREF #IMPLIED
trait_sem_valpond_l IDREFS #IMPLIED">
<!-- Regroupe differents attributs communs aux Usem et aux Usem_Aff :
definition_libre est un champ entierement libre destine a un
lecteur humain : il contient la definition du dictionnaire papier,
utile au lexicographe. Aucun controle n'est effectue sur ce champ, et
le modele ne dicte pas de regle de structuration de definition, ni de
syntaxe de la definition.
definition_formelle est un autre champ entierement libre qui
pourra contenir une definition formelle. Elle est utile pour associer
```

aux determinants une formule logique, elle peut contenir l'expression
de lambda-calcul associee a l'entree pour ceux qui le souhaitent.
Comme pour le champ precedent, aucun controle d'aucune sorte n'est
effectue sur son contenu.
combve pointe vers une CombVE (combinaison de Valeurs
d'Emploi). Cet element est egalement present dans les couches
morphologique et syntaxique, il regroupe : datation, niveau de langue,
frequence et variante geographique.
trait_sem_valpond_l se refere aux Trait_Sem_ValPond associes.
Il s'agit ici des caracteristiques semantiques, axe decompositionnel
de la couche semantique; on notera ici de preference les informations
lexicales non partagees par les differentes Usem independamment
d'objets descriptifs abstraits partages.
-->
<!ELEMENT Usem - O (RepresentationPredicative?,
RepresentationConceptuelle_Pond*,
R_ValPond_Usem*) >
<!ATTLIST Usem
id ID #REQUIRED
%pUsemAtt>
<!-- Dans les cas les plus courants, l'Usem (Unite semantique) decrit
un sens d'une Um (Unite morphologique), simple ou composee, dans les
contextes syntaxiques decrits par une ou plusieurs de ses Usyn
eventuellement filtrees.
L'Usem peut egalement decrire un des sens d'un compose decrit en

syntaxe, auquel cas on ne peut pas parler de l'Um dont elle provient, mais d'une ou plusieurs familles d'Usyn ou d'Um (compositions) dont elle ne decrit pas le sens isolement, mais en tant que fonctionnant comme un tout du point de vue linguistique.

Une Usem (tout comme une Um ou une Usyn d'ailleurs) est liee a une langue donnee, et permet une description de semantique lexicale fine. Elle est le point d'entree de la couche semantique. Sa description s'appuie sur des objets qui sont diverses abstractions sur les composantes de sens que l'on veut pouvoir lui affecter, certaines ayant un statut plutot decompositionnel ou analytique, d'autres etant plutot des generalisations ou abstractions a partir des Usem.

Le Predicat et le Concept sont les deux elements de description sur lesquels l'Usem s'appuie et par lesquels elle passe pour acceder au niveau de description abstrait (partiellement independant des particularites lexicales).

La connexion avec la sous-couche "abstraite" se fait par le biais des entites enchassees

RepresentationConceptuelle_Pond et

RepresentationPredicative,

qui associent a l'Usem un predicat (moyennant certaines modalites) et/ou un ou plusieurs concepts (moyennant certaines modalites et une ponderation).

L'Usem predicative porte sur elle-meme les Trait_Sem_ValPond qui lui sont propres. Le predicat qui lui est associe peut etre partage, il est lui-meme decrit et porte egalement des Trait_Sem_ValPond. Tout ce qui peut etre factorise doit l'etre, et les informations partagees

```
par differentes Usem partageant le meme predicat seront portees de
preference par le predicat.
De la meme facon, plusieurs Usem pointant vers le meme Concept avec
une meme ponderation (DEFINITOIRE par exemple) ne portent
de preference que les informations qui leur sont propres, factorisant
ce qu'elles ont en commun sur la description unique du Concept.
L'Usem peut etre vedette dans son association a un predicat ou un
concept.
A noter que les informations comportant une ponderation DEFINITOIRE ou
PROTOTYPIQUE (l'utilisation de ces ponderations n'est cependant pas
obligatoire...) permettent de construire une "definition" de l'Usem.
-->
<!ENTITY % pRepresentation
"vedette (SANS_B|%pBooleen) SANS_B
type_de_lien CDATA #IMPLIED">
<!-- Cette entite regroupe deux attributs :
type_de_lien precise l'association entre Usem et Concept ou
Predicat. Dans le cas du Predicat, on peut y preciser la semantique
du lien de facon fine (exemple: aptitude a etre Arg0)
vedette permet d'identifier l'Usem qui est la lexicalisation
privilegiee d'un concept ou d'un predicat.
-->
<!ELEMENT RepresentationPredicative - O (SelectEtPreciseArg*)>
<!ATTLIST RepresentationPredicative
%pRepresentation
```

```
chemin_arg_concerne NUMBERS #IMPLIED
arg_inclus (SANS_I|%pArgInclus) SANS_I
predicat IDREF #REQUIRED>
<!-- Permet d'associer une Usem a un Predicat en precisant les
modalites de cette association. L'Usem peut etre vedette, ce qui
signifie qu'elle est le representant lexical privilegie du Predicat,
et ce qui signifie aussi qu'il s'agit d'un Predicat nomme ou
lexicalise (voir l'entite Predicat).
chemin_arg_concerne precise si un argument est
particulierement concerne par l'Usem pointant vers le predicat et
donne le chemin d'acces a cet argument : il s'agit d'un chemin, car
on peut avoir en argument d'un predicat une structure predicative.
Le chemin_arg_concerne est une suite ordonnee de NUMBERS qui doit etre
interpretee comme suit: le premier nombre i pointe vers l'argument de
rang i du predicat pointe, le suivant, s'il existe, pointe alors sur
l'argument du predicat intervenant comme ieme argument, et ainsi de
suite.
argument_inclus precise si l'argument concerne est inclus
(ex: acheteur : predicat acheter, argument 0 inclus)
SelectEtPrecise_Arg permettent de rajouter de l'information
sur les arguments du predicat associe.
-->
<!ELEMENT RepresentationConceptuelle_Pond - O EMPTY>
<!ATTLIST RepresentationConceptuelle_Pond
%pRepresentation
%pModalite
```

```
commentaire CDATA #IMPLIED
concept IDREF #REQUIRED>
<!-- Permet d'associer une Usem a un concept en precisant les
modalites de cette association. L'Usem peut etre vedette, ce qui
signifie qu'elle est le representant lexical privilegie du concept,
et ce qui signifie aussi qu'il s'agit d'un concept nomme ou
lexicalise (voir l'entite Concept).
Cette association est assortie de ponderations (symbolique
et/ou numerique) ainsi que d'un point de vue.
-->
<!-- ******************************************************** -->
<!-- ***** UNITE SEMANTIQUE D AFFIXE ***** -->
<!-- ******************************************************** -->
<!ELEMENT Usem_Aff - O (RepresentationConceptuelle_Pond_Aff*)>
<!ATTLIST Usem_Aff
id ID #REQUIRED
%pUsemAtt
predicat IDREF #IMPLIED>
<!-- Usem_Aff (ou Unite semantique d'affixe) provient directement
d'une Um affixe dont elle decrit minimalement le noyau de sens ou
qu'elle caracterise du point de vue de la semantique.
Cette information peut permettre un minimum de prediction de sens
pour les derivations regulieres et productives de formes non
recensees dans le dictionnaire.
Une Usem_Aff peut etre decrite a l'aide de Trait_Sem_ValPond,
```

```
de Concepts et/ou d'un Predicat, ces elements pouvant se combiner
comme suit :
- un Predicat avec ou sans liste de Trait_Sem_ValPond,
et/ou - des Concepts avec ou sans liste de Trait_Sem_ValPond,
ou - une liste de Trait_Sem_ValPond
On peut egalement lui attribuer une definition (libre et/ou formelle)
et une CombVE.
-->
<!ELEMENT RepresentationConceptuelle_Pond_Aff - O EMPTY>
<!ATTLIST RepresentationConceptuelle_Pond_Aff
%pModalite
commentaire CDATA #IMPLIED
concept IDREF #REQUIRED>
<!-- Association, pour decrire une Usem_Aff, d'un Concept et d'un
ensemble d'informations decrivant sa modalite : ponderation
symbolique et numerique, point de vue -->
<!-- ********************************************************* -->
<!-- ***** PREDICAT ET CONCEPT ***** -->
<!-- ********************************************************* -->
<!ENTITY % pPredOuConcept
"%pGlose
type (SANS_TPC
|%pTypePredOuConcept) SANS_TPC
pivot (%pBooleen) NON
definition_libre CDATA #IMPLIED
definition_formelle CDATA #IMPLIED
```

trait_sem_valpond_l IDREFS #IMPLIED">

<!-- Entite regroupant des attributs communs aux predicats et concepts

type peut avoir differentes valeurs proposees par le modele:

- LEXICAL : directement issu d'une Usem, que celle-ci soit

vedette ou non

- GENERALISE : non directement issu d'une Usem, introduit

dans une relation de generalisation d'un autre Concept ou Predicat

eventuellement lexicalise, et en meme temps generalise par un autre

Concept ou Predicat (et donc non PRIMITIF)

- PRIMITIF : non directement issu d'une Usem, n'admet aucune

relation de generalisation; peut-etre donne comme element de

description preexistant ou predefini, independant de la langue

eventuellement (l'attribut pivot le precise)

- LEXICAL_PRIMITIF : a la fois issu directement du lexique et

sans generalisation (et eventuellement pivot)

- TROU_LEXICAL : pris entre deux Concepts ou Predicats

lexicaux, generalisation non lexicalisee de l'un, generalise par un

Concept ou un Predicat LEXICAL (les concepts de ce type seront utiles

en particulier pour decrire proprement un certain nombre de taxinomies

qui comportent des "trous" au niveau lexical).

pivot precise si le Concept ou le Predicat a ete choisi comme

element de description pivot, c'est-a-dire comme un element descriptif

partage independamment des langues.

definition_libre est un champ entierement libre destine a un

lecteur humain : il contient une definition textuelle ou toute

```
expression utile au lexicographe. Aucun controle n'est effectue sur
ce champ, et le modele ne dicte pas de regle de structuration de
definition, ni de syntaxe de la definition.
definition_formelle est un autre champ entierement libre qui
pourra contenir une definition formelle. Comme sur le champ precedent,
aucun controle d'aucune sorte n'est effectue sur le contenu de ce
champ.
trait_sem_valpond_l se refere aux Trait_Sem_ValPond associes
au Predicat ou au Concept; il s'agit de caracteristiques semantiques
liees a l'axe decompositionnel de la couche semantique.
On utilisera les mecanismes d'heritage pour eviter de preciser tous
les traits a tous les niveaux (une relation de type PARTICULARISATION
supposera heritage des informations portees par le "pere").
-->
<!ELEMENT Predicat - O (R_ValPond_Pred* & R_ValPond_Pred_Concept*
& Assoc_ListePred*)>
<!ATTLIST Predicat
id ID #REQUIRED
%pPredOuConcept
argument_l IDREFS #REQUIRED>
<!-- Predicat est l'un des elements centraux du modele semantique:
une Usem decrite comme predicative s'appuiera sur un Predicat de type
LEXICAL pour son noyau predicatif.
argument_l se refere aux Argument du Predicat, elements
descriptifs indispensables de celui-ci (les Argument sont aux
Predicat a peu pres ce que les Position sont aux Construction...).
```

```
La liste des Argument (argument_l) est une liste dont l'ordre est
pertinent, elle associe pour un Predicat donne, un argument et son
rang dans la liste (numerotation commencant a la valeur 0).
De l'exterieur, pour un certain predicat, on pointe vers un de ses
arguments par son rang.
Les arguments ne portent pas explicitement leur rang comme attribut,
car le rang est plutot une necessite de codage (pour acceder de
l'exterieur a un argument) qu'une information pertinente du point de
vue linguistique.
Dans la liste, il n'est pas exclu que deux elements soient identiques,
c'est a dire pointent vers le meme Argument: il s'agit bien d'une
liste et non d'un ensemble.
R_ValPond_Pred et R_ValPond_Pred_Concept portent les relations
valuees ponderees dont le predicat est source; la cible est
respectivement un Predicat ou un Concept.
Assoc_Liste_Pred pointe vers les listes de predicats
instancies eventuellement associees au predicat.
-->
<!ELEMENT Argument - O EMPTY>
<!ATTLIST Argument
id ID #REQUIRED
%pGlose
role_sem_l IDREFS #REQUIRED
informe_arg_decrit_l IDREFS #IMPLIED>
<!-- Argument est fortement associe a la description de Predicat,
```

```
dont il est un element totalement indispensable: en effet un Predicat
n'existe que s'il a au moins un argument.
Un Predicat connait un certain nombre d'informations sur chacun de ses
arguments. Les Argument peuvent etre, en tant qu'elements de
description, partages par plusieurs Predicat, et intervenant a
differents rangs dans la liste d'Argument de differents Predicat
(on peut faire l'analogie avec les Position de la syntaxe qui peuvent
etre partagees par plusieurs Construction, Predicat etant alors a
mettre en parallele avec Construction).
role_sem_l pointe vers le ou les Role_Sem (au moins un)
informe_arg_decrit_l pointe vers des InformeArg qui decrivent
ou contraignent semantiquement l'argument.
-->
<!ELEMENT InformeArg - O EMPTY>
<!ATTLIST InformeArg
id ID #REQUIRED
commentaire CDATA #IMPLIED
statut (SANS_S
|%pStatutInfoArg
|%pStatutInfoArg_cust) SANS_S
usem IDREF #IMPLIED
pred_instancie IDREF #IMPLIED
concept_l IDREFS #IMPLIED
trait_sem_valpond_l IDREFS #IMPLIED>
<!-- Cet element intervient pour ajouter des informations qui
s'accumulent sur un argument d'un predicat. Le predicat en lui-meme a
```

des connaissances sur ses arguments et ces connaissances s'expriment par des InformeArg portes par les arguments eux-memes.
Une Usem se decrit par un predicat en rajoutant eventuellement quelques informations sur les arguments (via SelectEtPreciseArg).
La correspondance Usyn Usem peut rajouter elle aussi des connaissances sur les arguments du predicat auquel est associee l'Usem (toujours via SelectEtPreciseArg), lorsque le contexte syntaxique "force" semantiquement un ou des arguments. Toutes ces connaissances sur les arguments s'accumulent, chaque niveau apportant ce qui lui est propre, aucun apport n'etant par ailleurs obligatoirement present.
Les connaissances sur un argument ont un statut qui permet de leur donner differentes modalites, essentiellement liees a la gestion des defauts, verification ou construction de la representation de l'argument.
statut : un InformeArg porte un ensemble d'informations sur un argument, ces informations ayant toutes le meme "statut" selon la valeur de cet attribut :
- DEFAUT : information sur l'argument, en l'absence de representation semantique de celui-ci : absent en surface, ou occupe par un element vide semantiquement.
- VERIF : quand l'argument present a une representation semantique, celle-ci doit verifier les contraintes affectees de ce statut : par exemple porter les trait_sem_valpond_l pointes ici, ou avoir pour generalisation le concept pointe.
- ENRICHIT ajoute de l'information semantique a l'information

connue sur l'argument, que celle-ci existe ou non.

- DEFAUT_VERIF combine l'apport par defaut, en cas d'absence

d'informations, a la verification en cas d'argument "plein"

semantiquement.

usem pointe vers une Usem.

pred_instancie pointe vers un predicat instancie

PredInstancie.

concept_l pointe vers une liste de Concept, le plus souvent un

Concept unique; le cas >1 sert a inserer dans une polyhierarchie de

concept dont on voudrait que l'argument herite.

trait_sem_valpond_l pointe vers une liste de traits

semantiques values ponderes.

Plusieurs InformeArg peuvent porter sur un meme argument.

Parmi les differentes informations optionnelles, au moins une (autre

que le commentaire) devra etre renseignee : un InformeArg apporte necessairement une information sur l’argument.

-->

<!ELEMENT SelectEtPreciseArg - O EMPTY>

<!ATTLIST SelectEtPreciseArg

chemin_arg NUMBERS #REQUIRED

informe_arg_precise IDREF #REQUIRED>

<!-- Permet de rajouter de l'exterieur du predicat et en passant par

lui, de l'information, sur ses arguments.

chemin_arg est le chemin vers l'argument : une suite de rangs,

un seul le plus souvent, mais parfois plus d'un si l'argument pointe

au premier niveau est lui-meme un predicat comportant un argument

```
vers lequel on veut pointer.

informe_arg_precise pointe vers l'InformeArg qui comprend

l'information semantique qui precise l'argument en se rajoutant a

celle qui est donnee par l'argument lui-meme.

-->

<!ELEMENT PredInstancie - O (SelectEtPreciseArg*)>

<!ATTLIST PredInstancie

id ID #REQUIRED

%pGlose

predicat IDREF #REQUIRED>

<!-- Association d'un predicat et d'informations plus ou moins

precises et completes sur la representation semantique de ses

arguments. Les arguments peuvent etre entierement decrits,

"instancies" ou seulement partiellement renseignes. La liste de

SelectEtPreciseArg peut meme etre totalement vide, auquel cas cet

element se contentera de pointer vers un predicat.

-->

<!ELEMENT Role_Sem - O EMPTY>

<!ATTLIST Role_Sem

id ID #REQUIRED

%pGlose

nom CDATA #REQUIRED

roleth_assoc IDREF #IMPLIED

est_un_l IDREFS #IMPLIED>

<!-- Les Role_Sem (roles semantiques) sont decrits hierarchiquement

par la relation est_un.
```

```
roleth_assoc permet d'associer (au besoin) un role thematique
de la couche syntaxique avec son equivalent Role_Sem de la couche
semantique.
est_un_l fait pointer un Role_Sem vers son ou ses "peres"
dans la hierarchie.
-->
<!ELEMENT Concept - O (R_ValPond_Concept* & R_ValPond_Concept_Pred*
& Assoc_ListePred*)>
<!ATTLIST Concept
id ID #REQUIRED
%pPredOuConcept
trait_sem_pertinent_l IDREFS #IMPLIED
trait_sem_obligatoire_l IDREFS #IMPLIED>
<!-- Concept est l'un des elements centraux du modele semantique:
il s'agit d'une abstraction "cognitive" sur des Usem ou d'une
generalisation de telles abstractions.
trait_sem_pertinent_l associe a un Concept une liste de
Trait_Sem qu'il est pertinent de renseigner. Cela peut permettre par
exemple d'associer des traits particularisants a renseigner sur des
Usem ou des Concept qui sont une particularisation du Concept portant
cette information.
Il s'agit d'une information qui permet de maintenir une certaine
coherence et completude de l'information codee sur les entrees
associees au Concept (directement ou par generalisation).
trait_sem_obligatoire_l joue un role analogue au precedent,
```

```
mais a une semantique plus contraignante : les traits pointes doivent
obligatoirement etre renseignes.
Un Concept est source de relations valuees ponderees vers des
concepts ou des predicats, c'est le sens des elements enchasses :
R_ValPond_Concept et R_ValPond_Concept_Pred.
Un Concept peut aussi etre associe a une ou plusieurs ListePred
(exemple classique en I.A : le concept de restaurant sera associe a
un certain scenario), c'est la raison de la presence de l'element
enchasse Assoc_ListePred.
-->
<!-- ********************************************************** -->
<!-- ***** TRAITS SEMANTIQUES ***** -->
<!-- ********************************************************** -->
<!ELEMENT Trait_Sem_ValPond - O EMPTY>
<!ATTLIST Trait_Sem_ValPond
id ID #REQUIRED
%pModalite
commentaire CDATA #IMPLIED
trait_sem_value IDREF #REQUIRED>
<!-- Trait_Sem_ValPond est l'association :
d'un trait value,
d'une ponderation symbolique a liste de valeurs proposee
(qui peut etre personnalisee),
d'une ponderation numerique,
et de la marque d'un point de vue.
-->
```

```
<!ELEMENT Trait_Sem_Value - O EMPTY>
<!ATTLIST Trait_Sem_Value
id ID #REQUIRED
%pGlose
pivot (%pBooleen) NON
valeurtrait IDREF #IMPLIED
valeurbin (SANS_B|%pBin) SANS_B
trait_sem IDREF #REQUIRED
usem_lexicalise_val IDREF #IMPLIED
trait_synt_corresp IDREF #IMPLIED
est_un_l IDREFS #IMPLIED
incompatible_l IDREFS #IMPLIED
implique_l IDREFS #IMPLIED
trait_sem_pertinent_l IDREFS #IMPLIED
trait_sem_obligatoire_l IDREFS #IMPLIED>
<!-- Trait_Sem_Value est l'association d'un trait semantique et d'une
valeur de ce trait.
pivot permet de preciser si ce trait a un statut de pivot,
c'est a dire independant de la langue, et donc pouvant intervenir
avec la meme definition dans des descriptions d'Usem de differentes
langues.
valeurtrait, valeurbin : soit la valeur du trait est donnee
par un element valeurtrait, soit elle est binaire et donnee par
valeurbin.
trait_sem se refere au Trait_Sem auquel on associe la valeur,
```

ce trait ayant lui-meme des proprietes et sa propre description.
usem_lexicalise_val se refere eventuellement a l'Usem qui
lexicalise le trait value. Les trait values constituent un
meta-langage qui peut se trouver connecte a la langue, car tous les
traits values ne sont pas associes a leur lexicalisation.
trait_synt_corresp pointe vers le trait "semantique" utilise
en syntaxe auquel il correspond, et permet d'en donner la definition
complete au niveau semantique.
est_un_l permet d'inserer le Trait_Sem_Value dans une
hierarchie et meme un treillis, les Trait_Sem_Value referes par cette
liste sont les noeuds "peres" du trait value portant cet attribut.
incompatible_l se refere a un ensemble de Trait_Sem_Value dont
la semantique est incompatible avec celle du Trait_Sem_Value portant
cet attribut.
implique_l se refere a un ensemble de Trait_Sem_Value qui sont
impliques par celui-ci.
Suivant le type du Trait_Sem (attribut "concerne" de ce dernier), le
sens de cette relation n'est pas tout a fait le meme :
- S'il s'agit d'un Trait_Sem d'AFFIXE, cela signifie que les
Usem comportant cet affixe dans le mode de derivation de leur Um
porteront les Trait_Sem_Value impliques par le Trait_Sem_Value
d'AFFIXE si la semantique de l'affixe entrant dans leur composition
correspond a celle qui est decrite. Cette information permet surtout
de faire une prediction minimale sur la representation semantique de
derives non decrits dans le dictionnaire, mais de formation
reguliere. L'implication de Trait_Sem_Value se fait ici entre deux

```
objets conceptuellement differents.
- S'il s'agit d'un Trait_Sem AUTRE (= autre qu'Affixe), cela
signifie qu'un objet descriptif portant ce Trait_Sem_Value n'aura pas
besoin de preciser les Trait_Sem_Value impliques qu'il portera
implicitement. L'implication ici se fait sur des objets de meme
nature portant le trait "impliquant" et les traits impliques.
trait_sem_pertinent_l associe a un Trait_Sem_Value une liste
de Trait_Sem qu'il est pertinent de renseigner. Cela peut permettre
par exemple d'associer des traits differenciateurs a renseigner a un
trait value de classe semantique.
trait_sem_obligatoire_l a une semantique analogue au
precedent, et une contrainte supplementaire sur le codage des
elements portant ce trait : ils devront porter obligatoirement des
traits values ponderes associes a ces traits obligatoires.
-->
<!ELEMENT ValeurTrait - O EMPTY>
<!ATTLIST ValeurTrait
id ID #REQUIRED
libelle CDATA #REQUIRED>
<!-- ValeurTrait represente la valeur des traits semantiques values
Trait_Sem_Value non binaires. Ces valeurs peuvent etre referencees
par plusieurs traits semantiques (Trait_Sem) differents.
-->
<!ELEMENT Trait_Sem - O EMPTY>
<!ATTLIST Trait_Sem
```

```
id ID #REQUIRED
%pGlose
nom CDATA #REQUIRED
pivot (%pBooleen) NON
concerne (SANS_C|%pConcerne) SANS_C
type (SANS_TYPE
|%pTypeTrait_Sem
|%pTypeTrait_Sem_cust) SANS_TYPE
statut (%pValuation) MONOVALUE
type_liste_valeurs (%pTypeListe) BINAIRE
structure_liste_valeurs (SANS_STRUCT
|%pStructureListe) SANS_STRUCT
usem_lexicalise IDREF #IMPLIED
valeurtrait_l IDREFS #IMPLIED
trait_sem_pertinent_l IDREFS #IMPLIED
trait_sem_obligatoire_l IDREFS #IMPLIED>
<!-- Trait_Sem s'associe a une valeur pour produire un Trait_Sem_Value
(lequel a un certain nombre de proprietes, voir plus haut).
pivot : le Trait_Sem peut etre lie a la description d'une
langue ou pivot, c'est a dire choisi comme etant partage par
plusieurs langues, element de description partage.
concerne permet de preciser si le trait concerne
exclusivement les affixes ou les autres elements de la couche
semantique.
type precise le type ou la famille du Trait_Sem (une liste de
valeurs est proposee, l'utilisateur peut l'etendre).
```

statut permet de specifier si le Trait_Sem est :

- MONOVALUE : ce qui signifie qu'un element ne peut porter

qu'un Trait_Sem_Value associe a ce Trait_Sem, car les valeurs sont

exclusives les unes des autres.

- MULTIVALUE : ce qui signifie que plusieurs Trait_Sem_Value

associes au meme Trait_Sem peuvent etre portes par un meme objet

descriptif de la couche semantique.

type_liste_valeurs precise la nature des valeurs associees au

Trait_Sem :

- BINAIRE : + ou -

- LISTE_FERMEE : liste de valeurs connue en extension.

- LISTE_OUVERTE : liste non limitative de valeurs.

structure_liste_valeurs precise la structuration des

differents Trait_Sem_Value associes au Trait_Sem; cette structuration

s'exprime localement par la relation est_un entre Trait_Sem_Value.

Trait_Sem est en partie caracterise par la structure associee:

- TREILLIS_TOTAL : l'ensemble des Trait_Sem_Value associes

forme un treillis.

- TREILLIS_PARTIEL : les Trait_Sem_Value entrent localement

dans des structures de treillis, plusieurs treillis etant autorises

(et donc certains Trait_Sem_Value peuvent etre isoles).

- HIERARCHIE_TOTALE : l'ensemble des Trait_Sem_Value associes

est structuree dans un arbre.

- HIERARCHIE_PARTIELLE : l'ensemble des Trait_Sem_Value

associes est structure dans un ou plusieurs arbres. Tous les

```
Trait_Sem_Value ne sont pas necessairement dans une relation
structurante.
usem_lexicalise permet d'identifier le Trait_Sem dans la
langue, en se referant a une Usem lexicalisante.
valeurtrait_l permet de preciser en extension la liste des
valeurs (quand son type est LISTE_FERMEE) en se referant aux elements
ValeurTrait.
trait_sem_pertinent_l associe a un Trait_Sem une liste de
Trait_Sem qu'il est pertinent de renseigner pour les elements portant
un Trait_Sem_ValPond associe a ce Trait_Sem, independamment de leur
valeur.
trait_sem_obligatoire_l a une semantique analogue au
precedent, et une contrainte supplementaire sur le codage des
elements portant ce trait : ils devront porter des Trait_Sem_ValPond
associes a ces traits obligatoires.
-->
<!-- ******************************************************** -->
<!-- ***** RELATIONS SEMANTIQUES PONDEREES ***** -->
<!-- ******************************************************** -->
<!ENTITY % pR_ValPond_Sem
"%pModalite
commentaire CDATA #IMPLIED">
<!ELEMENT (R_ValPond_Usem|R_ValPond_Pred) - O (Corresp_Arg_Arg*)>
<!ELEMENT R_ValPond_Concept - O EMPTY>
<!ATTLIST (R_ValPond_Usem|R_ValPond_Pred|R_ValPond_Concept)
%pR_ValPond_Sem
```

```
cible IDREF #REQUIRED
r_sem IDREF #REQUIRED>
<!-- Ces elements sont enchasses dans la description de la source :
relation valuee ponderee liant deux Usem (R_ValPond_Usem), deux
predicats (R_ValPond_Pred), deux concepts (R_ValPond_Concept),
suivant une certaine modalite :
- une ponderation symbolique a liste de valeurs proposee qui
peut etre personnalisee,
- une ponderation numerique,
- et la marque d'un point de vue.
Corresp_Arg_Arg precise comment les arguments se
correspondent lorsque les Usem sont predicatives, ou lorsque la
relation lie deux Predicat.
cible pointe vers la cible de la relation valuee ponderee :
Usem, Predicat ou Concept suivant la nature de l'element.
r_sem pointe sur la R_Usem, R_Pred ou R_Concept liant les
deux objets.
Une R_ValPond_Usem/Pred/Concept est donc a voir comme l'entite qui
decrit le fait que la relation semantique R_Usem/Pred/Concept lie
l'Usem/Pred/Concept source et l'Usem/Pred/Concept cible suivant une
certaine modalite.
-->
<!ELEMENT (R_ValPond_Pred_Concept|R_ValPond_Concept_Pred) - O EMPTY>
<!ATTLIST (R_ValPond_Pred_Concept|R_ValPond_Concept_Pred)
%pR_ValPond_Sem
```

```
cible IDREF #REQUIRED
r_sem IDREF #REQUIRED
chemin_arg_concerne NUMBERS #IMPLIED>
<!-- Ces element sont enchasses dans la description de la source :
relation valuee ponderee liant un Predicat/source et un Concept/cible
(R_ValPondPred_Concept) ou un Concept/source et un Predicat/cible
(R_ValPond_Concept_Pred) suivant une certaine modalite :
- une ponderation symbolique a liste de valeurs proposee qui
peut etre personnalisee,
- une ponderation numerique,
- et la marque d'un point de vue.
cible pointe sur le Predicat ou le Concept cible de la
relation.
r_sem se refere a la R_Pred_Concept/R_Concept_Pred liant le
Predicat et le Concept.
chemin_arg_concerne donne, si necessaire, le chemin d'acces,
a partir du predicat, vers l'argument particulierement concerne par
la relation; une meme relation semantique peut donc intervenir
independamment de l'argument concerne par celle-ci.
Une R_ValPond_Pred_Concept/Concept_Pred est donc a voir comme
l'entite qui precise que la relation semantique R_Pred_Concept
ou R_Concept_Pred lie le Predicat/Concept source et le
Concept/Predicat cible suivant une certaine modalite.
-->
<!ELEMENT Corresp_Arg_Arg - O EMPTY>
<!ATTLIST Corresp_Arg_Arg
```

```
chemin_arg_source NUMBERS #IMPLIED
chemin_arg_cible NUMBERS #IMPLIED
informe_arg_precise_source_l IDREFS #IMPLIED
informe_arg_precise_cible_l IDREFS #IMPLIED>
<!-- Corresp_Arg_Arg met en correspondance les Argument de Predicat
en jeu en tant que source ou cible d'une relation valuee pondere. Les
chemins vers les arguments source et cible sont indiques :
chemin_arg_source (le plus souvent un seul rang donnant acces
au premier niveau).
chemin_arg_cible (le plus souvent un seul rang donnant acces
au premier niveau).
Certaines informations semantiques supplementaires sur les arguments
peuvent etre ajoutees dans la correspondance :
informe_arg_precise_source_l : informations sur l'argument
source (InformeArg).
informe_arg_precise_cible_l : informations sur l'argument
cible (InformeArg).
C'est ainsi que l'on peut contraindre les interpretations semantiques
acceptables de certains arguments dans la correspondance, et aussi
que l'on peut preciser les valeurs "par defaut" liees a la
correspondance d'arguments dans la mise en correspondance de
predicats a nombre d'arguments differents.
(Ex: fixer(X,Y,clou,Z) <=> clouer(X,Y,Z))
-->
<!-- ********************************************************** -->
```

```
<!-- ***** RELATIONS SEMANTIQUES ***** -->

<!-- ********************************************************** -->

<!ENTITY % pR_SemPropriete

"reflexivite (REFLEXIF

|NON_REFLEXIF) NON_REFLEXIF

transitivite (TRANSITIF

|NON_TRANSITIF) NON_TRANSITIF

symetrie (SYMETRIQUE

|NON_SYMETRIQUE) NON_SYMETRIQUE

antisymetrie (ANTISYMETRIQUE

|NON_ANTISYMETRIQUE) NON_ANTISYMETRIQUE

ordre (PARTIEL|TOTAL

|NON_PERTINENT) NON_PERTINENT">

<!-- Proprietes des relations semantiques. -->

<!ELEMENT R_Usem - O (CatGram_Select? & CatGram_Result?)>

<!ATTLIST R_Usem

id ID #REQUIRED

%pGlose

r_sem_inverse IDREF #IMPLIED

incompatible_l IDREFS #IMPLIED

est_un_l IDREFS #IMPLIED

type (SANS_T

|%pTypeR_Usem

|%pTypeR_Usem_cust) SANS_T

sstype (SANS_ST

|%pSsTypeR_Usem
```

|%pSsTypeR_Usem_cust) SANS_ST

%pR_SemPropriete>

<!-- R_Usem est l'entite formelle qui decrit une relation semantique

liant deux Usem.

CatGram_Select : filtre eventuellement la categorie

grammaticale de la source.

CatGram_Result : filtre eventuellement la categorie

grammaticale de la cible.

r_sem_inverse se refere a la relation R_Usem inverse

(source=>cible, cible=>source).

incompatible_l se refere aux relations semantiques R_Usem qui

sont incompatibles avec la relation decrite, et qui ne peuvent donc

entrer dans une R_ValPond liant des elements deja lies par celle-ci.

est_un_l precise la structuration hierarchique entre

relations, en se referant aux R_Usem "meres" de la relation decrite.

type precise le type de la relation : un ensemble de types

est propose par le modele GENELEX et cet ensemble peut etre etendu par

l'utilisateur.

sstype permet de caracteriser plus finement la relation

semantique. Des valeurs sont proposees, et l'utilisateur peut etendre

cette liste.

Une relation semantique peut enfin avoir un certain nombre de

proprietes qui sont celles des relations binaires : reflexivite,

transitivite, symetrie, antisymetrie et le fait d'etre une relation

d'ordre.

```
-->
<!ELEMENT (R_Pred|R_Concept) - O EMPTY>
<!ATTLIST (R_Pred|R_Concept)
id ID #REQUIRED
%pGlose
r_sem_inverse IDREF #IMPLIED
incompatible_l IDREFS #IMPLIED
est_un_l IDREFS #IMPLIED
pivot (%pBooleen) NON
type (SANS_T
|%pTypeR_PredOuConcept
|%pTypeR_PredOuConcept_cust) SANS_T
sstype CDATA #IMPLIED
%pR_SemPropriete>
<!-- R_Pred (R_Concept) est l'entite formelle qui decrit une relation
semantique liant deux Predicat (Concept).
r_sem_inverse se refere a la relation semantique inverse
R_Pred (R_Concept) (source=>cible, cible=>source).
incompatible_l se refere aux relations R_Pred (R_Concept) qui
sont incompatibles avec la relation decrite, et qui ne peuvent donc
entrer dans une R_ValPond liant des elements deja lies par celle-ci.
est_un_l precise la structuration hierarchique entre
relations, en se referant aux R_Pred (R_Concept) "meres" de la
relation decrite.
pivot permet de marquer les relations qui sont "primitives"
et sur lesquelles on choisit de s'appuyer independamment des langues :
```

on sait alors transposer des reseaux de relations d'une langue a une autre a partir de couples (bilingues) de predicats ou concepts.
type precise le type de la relation : un ensemble de types est propose par le modele GENELEX et cet ensemble peut etre etendu par l'utilisateur.
sstype permet de caracteriser plus finement la relation.
Une relation semantique peut enfin avoir un certain nombre de proprietes qui sont celles des relation binaires : reflexivite, transitivite, symetrie, antisymetrie et le fait d'etre une relation d'ordre.
-->

```
<!ELEMENT (R_Pred_Concept|R_Concept_Pred) - O EMPTY>
<!ATTLIST (R_Pred_Concept|R_Concept_Pred)
id ID #REQUIRED
%pGlose
r_sem_inverse IDREF #IMPLIED
incompatible_l IDREFS #IMPLIED
est_un_l IDREFS #IMPLIED
type CDATA #IMPLIED>
```

<!-- R_Concept_Pred (R_Pred_Concept) est l'entite formelle qui decrit une relation semantique liant un Concept (resp. un Predicat) a un Predicat (resp. un Concept).
r_sem_inverse se refere a la relation semantique inverse (source=>cible, cible=>source) : l'inverse d'une R_Concept_Pred est une R_Pred_Concept et reciproquement.

```
incompatible_l se refere aux R_Concept_Pred (R_Pred_Concept)
qui sont incompatibles avec la relation decrite, et qui ne peuvent
donc entrer dans une R_ValPond liant des elements deja lies par
celle-ci.
est_un_l precise la structuration hierarchique entre
relations, en se referant aux R_Concept_Pred (R_Pred_Concept) "meres"
de la relation decrite.
type precise le type de la relation : ce champ est
entierement libre.
-->
<!ELEMENT Assoc_ListePred - O EMPTY>
<!ATTLIST Assoc_ListePred
role (ASSOC|%pRoleListePred
|%pRoleListePred_cust)
ASSOC
listepred IDREF #REQUIRED
intervient (%pBooleen) NON>
<!-- Assoc_ListePred permet d'associer a un Predicat ou un Concept
une ListePred en precisant la modalite de cette association :
role precise le role de la ListePred dans la description du
Predicat ou du Concept y faisant appel. Un certain nombre de valeurs
sont proposees par le modele : A_PRESUPPOSITION, A_IMPLICATION,
A_DEFINITION, A_EXPLICITATION, et l'utilisateur pourra au besoin
rajouter ses propres valeurs.
listepred pointe vers la ListePred concernee.
intervient precise, pour un Predicat, s'il est lui-meme
```

```
intervenant direct dans la ListePred , c'est a dire s'il est un
element de cette liste.
-->
<!ELEMENT ListePred - O (Variable*)>
<!ATTLIST ListePred
id ID #REQUIRED
%pGlose
statut (%pStatutListePred
|%pStatutListePred_cust) DEFINITION
type (SANS_TL|%pTypeListePred
|%pTypeListePred_cust) SANS_TL
pred_instancie_l IDREFS #REQUIRED>
<!-- ListePred decrit une liste ordonnee de PredInstancie aux
arguments desquels on pourra associer des Variables, pour preciser le
partage d'arguments entre differents Predicat.
statut permet de preciser le statut du regroupement de
PredInstancie : une liste de valeurs est proposee par le modele
(SCENARIO, DEFINITION) et l'utilisateur pourra au besoin rajouter ses
propres valeurs.
type precise le sens a donner a l'ordre des PredInstancie
dans la liste : un certain nombre de valeurs sont proposees par le
modele (ORDRE_TEMP, ORDRE_SPATIAL, SIMULTANEITE) ordre temporel,
spatial,... et l'utilisateur pourra au besoin rajouter ses propres
valeurs.
pred_instancie_l se refere aux PredInstancie qui forment une
```

```
liste dont l'ordre est pertinent.
-->
<!ELEMENT Variable - O (SelectPredArg+) >
<!-- L'entite enchassee Variable n'a de sens que pour la ListePrede
qui la contient; elle permet d'associer sur une meme variable des
arguments de plusieurs PredInstancie, lesquels "s'unifient" pour la
ListePred qui les fait intervenir. Cette association se fait en
pointant sur une liste de SelectPredArg, chemins vers les arguments
de Predicat "s'unifiant".
Par exemple, dans un scenario de restaurant, le serveur porte des
aliments, qu'il sert au client, lequel les mange : trois SelectPredArg
permettant d'atteindre les arguments de trois predicats instancies
dans la liste de predicats du scenario "restaurant", ces trois
arguments s'unifiant dans une meme Variable.
-->
<!ELEMENT SelectPredArg - O EMPTY>
<!ATTLIST SelectPredArg
nieme_pred NUMBER #REQUIRED
chemin_arg NUMBERS #REQUIRED>
<!-- SelectPredArg determine un chemin qui permet d'atteindre un
argument d'un Predicat de ListePred.
nieme_pred designe le PredInstancie nieme de la liste
ListePred.
chemin_arg designe la suite de rangs d'arguments du Predicat
(un seul le plus souvent) ainsi atteint; dans les cas rares ou ce
Predicat a un argument complexe, c'est a dire lui-meme predicatif, on
```

```
continuera eventuellement le chemin_arg en descendant dans la
structure predicative intervenant comme realisation de l'argument
nieme et en allant pointer recursivement vers son nieme argument :
c'est la raison de l'existence ici d'une liste de NUMBERS
-->
<!-- ******************************************************** -->
<!-- ***** CORRESPONDANCE SYNTAXE SEMANTIQUE ***** -->
<!-- ******************************************************** -->
<!ELEMENT Correspondance - O (SelectEtPreciseArg*)>
<!ATTLIST Correspondance
id ID #REQUIRED
%pGlose
contraint_description IDREF #IMPLIED
corresp_arg_pos_l IDREFS #IMPLIED>
<!-- Correspondance precise pour une Corresp_Usyn_Usem, ses
differents aspects :
contraint_description pointe sur un objet
ContraintDescription permettant de filtrer un sous-ensemble des
realisations syntaxiques associees a l'Usyn, en rendant la
description syntaxique plus contrainte.
corresp_arg_pos_l : sert a associer a chaque argument du
predicat sur lequel s'appuie l'Usem (on pourrait aussi dire : que
l'Usem lexicalise avec ses propres particularites) sa realisation en
syntaxe. Il y a, en general, autant de corresp_arg_pos que d'arguments
presents en surface, mais eventuellement plusieurs corresp_arg_pos
```

en alternative pour un meme argument dans le cas d'un compose
syntaxique avec variantes de lexicalisation dans des MDC differents.
SelectEtPreciseArg permet de preciser si necessaire sur des
arguments l'information semantique (qui s'ajoute aux informations
qu'a le predicat sur ses arguments dans sa definition et dans la
modalite de lexicalisation du predicat par l'Usem) propre au contexte
syntaxique particulier de l'Usyn, donc propre au couple Usyn-Usem.
Les informations semantiques propres au predicat ou a la facon dont
l'Usem lui est associee seront exprimees ailleurs.
Aucun de ces attributs n'est obligatoirement renseigne, et on pourra
donc avoir un element vide ne contenant que la Glose dans le cas
d'association Usyn Usem n'ayant rien a preciser.
-->
<!ELEMENT ContraintDescription - O (Contraint_mdc?)>
<!ATTLIST ContraintDescription
id ID #REQUIRED
%pGlose
contraint_intervconst IDREF #IMPLIED
contraint_struct_interne IDREF #IMPLIED
contraint_construction IDREF #IMPLIED>
<!-- Cet element permet de restreindre l'ensemble des realisations
syntaxiques parmi celles associees a la description de base de l'Usyn
Contraint_mdc contraint le mode de composition d'un compose
contraint_intervconst contraint les realisations de Self
contraint_struct_interne contraint les realisations des
positions de la structure interne d'un compose

```
contraint_construction contraint les realisations des
positions associees a la construction de base.
Plusieurs de ces elements peuvent etre presents en meme temps,
cependant si on a simultanement contraint_struct_interne et
contraint_mdc, qui sont associes a deux modes de description
concurrents de la composition, on veillera a la coherence des
informations.
-->
<!ELEMENT ContraintIntervConst - O EMPTY>
<!ATTLIST ContraintIntervConst
id ID #REQUIRED
contraint_syntagme_l IDREFS #REQUIRED>
<!-- Permet de contraindre un ou plusieurs syntagmes de
l'intervconst en leur rajoutant des traits de la syntaxe ou en
les inhibant.
-->
<!ELEMENT ContraintSyntagme - O (AjouteTrait_Sem*)>
<!ATTLIST ContraintSyntagme
id ID #REQUIRED
syntagme IDREF #REQUIRED
inhibe (%pBooleen) NON
ajoute_trait_synt_l IDREFS #IMPLIED>
<!-- Cet element permet de selectionner et contraindre un syntagme.
syntagme permet de selectionner le syntagme concerne (en tant
qu'appartenant a la distribution d'une position ou a l'intervconst).
```

```
inhibe precise s'il est inhibe et devient donc absent de la
distribution pour l'acception donnee.
ajoute_trait_synt_l restreint un syntagme non inhibe par
l'ajout de traits restrictifs de la syntaxe.
AjouteTrait_Sem permet de le contraindre sur le critere de
l'information semantique associee a son interpretation semantique.
-->
<!ELEMENT AjouteTrait_Sem - O EMPTY>
<!ATTLIST AjouteTrait_Sem
statut (%pStatutAjoute) FILTRE_AJOUTE
trait_sem_valpond IDREF #REQUIRED>
<!-- Permet de filtrer sur l'interpretation semantique associee au
syntagme en forcant/verifiant la presence d'un trait.
statut precise :
- qu'il doit necessairement etre present pour autoriser la
correspondance (valeur FILTRE),
- qu'il doit etre present ou ajoute a condition d'etre
compatible avec la representation semantique (valeur FILTRE_AJOUTE),
- ou qu'il doit enrichir de toute facon la representation
semantique associee (valeur FORCE).
trait_sem_valpond pointe vers le trait value pondere en
question.
-->
<!ELEMENT ContraintConstruction - O EMPTY>
<!ATTLIST ContraintConstruction
id ID #REQUIRED
```

```
contraint_position_l IDREFS #REQUIRED>
<!-- Contraint la construction externe d'une Usyn en contraignant une
ou plusieurs de ses positions :
- en la rendant obligatoire ou interdite, s'il s'agit d'une
position optionnelle,
- ou en contraignant sa distribution.
Il y a autant de ContraintPosition pointes par contraint_position_l
que de positions contraintes, qu'elles soient inhibees ou rendues
obligatoires, ou que leur distribution soit contrainte.
-->
<!ELEMENT ContraintStructInterne - O EMPTY>
<!ATTLIST ContraintStructInterne
id ID #REQUIRED
contraint_position_l IDREFS #REQUIRED>
<!-- Contraint la structure interne d'une Usyn composee en
contraignant une ou plusieurs de ses positions :
- en la rendant obligatoire ou interdite, s'il s'agit d'une
position optionnelle,
- ou en contraignant sa distribution.
Il y a autant de ContraintPosition pointes par contraint_position_l
que de positions contraintes, qu'elles soient inhibees ou rendues
obligatoires, ou que leur distribution soit contrainte.
-->
<!ELEMENT Contraint_mdc - O (ContraintPosition_mdc+)>
<!-- Contraint le mode de composition d'une Usyn composee en
```

```
contraignant une ou plusieurs positions d'une ou plusieurs Usyn
composantes. Il y a autant de ContraintPosition_mdc que de
constructions externes de composantes a contraindre.
-->
<!ELEMENT ContraintPosition_mdc - O EMPTY>
<!ATTLIST ContraintPosition_mdc
nieme_composition NUMBER #REQUIRED
nieme_composante NUMBER #REQUIRED
contraint_position_l IDREFS #REQUIRED>
<!-- Pour une composition (nieme_composition) (en cas d'alternative de
compositions) contraint une de ses composantes (nieme_composante) en
contraignant une ou plusieurs de ses positions :
- en la rendant obligatoire ou interdite, s'il s'agit d'une
position optionnelle,
- ou en contraignant sa distribution.
Il y a autant de ContraintPosition pointes par contraint_position_l
que de positions contraintes pour un meme couple
nieme_composition/nieme_composante, qu'elles soient inhibees ou
rendues obligatoires, ou que leur distribution soit contrainte.
Le ContraintPosition_mdc doit preciser nieme_composition et
nieme_composante de sorte que la position a contraindre soit position
de la construction externe de l'Usyn ainsi pointee. Une Um ne peut
etre contrainte que comme appartenant a la distribution d'une
position de l'Usyn l'appelant.
-->
<!ELEMENT ContraintPosition - O (CheminPosition)>
```

```
<!ATTLIST ContraintPosition
id ID #REQUIRED
modif_optionnalite (SANS_MODIF|OBLIGATOIRE
|INTERDITE) SANS_MODIF
contraint_syntagme_l IDREFS #IMPLIED>
<!-- ContraintPosition permet d'acceder a une position (de la
construction externe, ou structure interne ou composition suivant
l'element qui fait intervenir le ContraintPosition) par un :
CheminPosition qui est parcouru a partir de la construction
de base, de la structure interne, ou de la construction externe d'un
composant en cas de description d'Usyn par Composition.
modif_optionnalite permet de preciser qu'une position
optionnelle dans la description devient obligatoire ou au contraire
interdite pour l'acception (Usem) pointee.
contraint_syntagme_l permet de preciser dans la distribution
de la position contrainte les syntagmes et les contraintes
supplementaires qui leur sont associees. Il y aura autant de
ContraintSyntagme que de syntagmes sur lesquels on veut apporter des
restrictions. D'autre part, un syntagme peut etre filtre ou enrichi
d'un point de vue semantique par des traits de la semantique.
-->
<!-- ******************************************************** -->
<!-- ***** CORRESPONDANCE ARGUMENT POSITION ***** -->
<!-- ******************************************************** -->
<!ENTITY % pCorresp_Arg_Pos
```

```
"chemin_arg NUMBERS #REQUIRED
portee (%pPorteeCorresp) EXTERNE
nieme_composition NUMBER #IMPLIED
nieme_composante NUMBER #IMPLIED">
<!-- Ensemble d'attributs communs aux differents cas de
correspondance argument-position, permettant de selectionner un
argument et de preciser son expression dans la structure syntaxique.
chemin_arg permet de selectionner l'argument concerne par une
suite de rangs (le plus souvent un, sauf cas de predicat comme
argument d'un predicat).
portee precise si la realisation syntaxique se fait
- sur une position de la construction externe (cas le plus
frequent, portee = EXTERNE),
- sur une position de la structure interne (en cas de
composition decrite par un syntagme de structure, portee =
STRUCT_INTERNE),
- ou sur un des elements de la composition (portee =
COMP_INTERNE).
Dans ce dernier cas :
nieme_composition precise la composition concernee,
nieme_composante la composante dans la composition precisee.
-->
<!ELEMENT Corresp_Arg_Pos_Simple - O (CheminPosition)>
<!ATTLIST Corresp_Arg_Pos_Simple
id ID #REQUIRED
%pCorresp_Arg_Pos>
```

```
<!-- Associe a un argument de la couche semantique une position de
la syntaxe.
portee, nieme_composition, nieme_composante permettent de
savoir sur quoi s'applique CheminPosition,
chemin_arg pointe vers l'argument concerne; le parcours des
rangs se fait a partir du predicat directement pointe par l'Usem
concernee par la correspondance syntaxe-semantique qui fait appel au
Corresp_Arg_Pos_Simple.
-->
<!ELEMENT Corresp_Arg_Pos_Flottant - O (CheminSyntagme?)>
<!ATTLIST Corresp_Arg_Pos_Flottant
id ID #REQUIRED
%pGlose
%pCorresp_Arg_Pos
position IDREF #REQUIRED>
<!-- Permet d'associer a un argument du predicat essentiel sa famille
de realisations syntaxiques, lorsque celle-ci correspond a un element
non decrit dans la description de base car non essentiel en tant
qu'element de la syntaxe ("complements adjoints" : circonstants,
modifieurs... non precises pour l'Usyn).
position pointe vers une Position qui donne la description de
la famille de realisations en syntaxe (et precisera donc une
distribution et eventuellement fonction et roles thematiques).
CheminSyntagme permet de preciser en cas de reecriture le
niveau de l'insertion virtuelle de la position associee a
```

l'Arg_flottant en pointant sur le syntagme reecrit : l'insertion se fait alors dans la liste de reecriture de ce syntagme, la fonction et les roles thematiques sont precises par rapport a la tete de ce syntagme.
L'absence de CheminSyntagme implique que l'insertion se fait au niveau le plus haut, c'est-a-dire suivant le cas (precise par l'attribut portee) :
- dans la liste de positions de la construction externe (cas le plus frequent),
- dans la liste de positions de la structure interne (en cas de composition decrite par un syntagme de structure),
- ou au niveau d'un des elements de la composition (precise par nieme_composition et nieme_composante qui interviennent dans le cas d'une composition decrite par un calcul de composition, auquel cas il ne peut y avoir de CheminSyntagme : on pointera au bon niveau en choisissant le composant voulu).
-->

3. Contraintes *semant.ctr*

<!--Consortium GENELEX @(#) semant.ctr 2.1 -->

<!--CONTRAINTE Usem

```
combve TYPE CombVE

trait_sem_valpond_l TYPE Trait_Sem_ValPond -->

<!--CONTRAINTE RepresentationPredicative

predicat TYPE Predicat -->


<!--CONTRAINTE RepresentationConceptuelle_Pond

concept TYPE Concept -->

<!--CONTRAINTE Usem_Aff

predicat TYPE Predicat

combve TYPE CombVE

trait_sem_valpond_l TYPE Trait_Sem_ValPond -->


<!--CONTRAINTE RepresentationConceptuelle_Pond_Aff

concept TYPE Concept -->

<!--CONTRAINTE Predicat

trait_sem_valpond_l TYPE Trait_Sem_ValPond argument_l TYPE Argument -->

<!--CONTRAINTE Argument

role_sem_l TYPE Role_Sem

informe_arg_decrit_l TYPE InformeArg -->

<!--CONTRAINTE InformeArg

usem TYPE Usem

pred_instancie TYPE PredInstancie

concept_l TYPE Concept

trait_sem_valpond_l TYPE Trait_Sem_ValPond -->

<!--CONTRAINTE SelectEtPreciseArg

informe_arg_precise TYPE InformeArg -->
```

```
<!--CONTRAINTE PredInstancie
predicat TYPE Predicat -->
<!--CONTRAINTE Role_Sem
roleth_assoc TYPE RoleTh
est_un_l TYPE Role_Sem -->
<!--CONTRAINTE Concept
trait_sem_valpond_l TYPE Trait_Sem_ValPond
trait_sem_obligatoire_l TYPE Trait_Sem
trait_sem_pertinent_l TYPE Trait_Sem -->
<!--CONTRAINTE Trait_Sem_ValPond
trait_sem_value TYPE Trait_Sem_Value -->
<!--CONTRAINTE Trait_Sem_Value
valeurtrait TYPE ValeurTrait
trait_sem TYPE Trait_Sem
usem_lexicalise_val TYPE Usem
trait_synt_corresp TYPE Trait_Aspect|Trait_Bin
|Trait_Libre
est_un_l TYPE Trait_Sem_Value
incompatible_l TYPE Trait_Sem_Value
implique_l TYPE Trait_Sem_Value
trait_sem_pertinent_l TYPE Trait_Sem
trait_sem_obligatoire_l TYPE Trait_Sem -->
<!--CONTRAINTE Trait_Sem
usem_lexicalise TYPE Usem
valeurtrait_l TYPE ValeurTrait
```

```
trait_sem_pertinent_l TYPE Trait_Sem

trait_sem_obligatoire_l TYPE Trait_Sem -->

<!--CONTRAINTE R_ValPond_Usem

cible TYPE Usem

r_sem TYPE R_Usem -->

<!--CONTRAINTE R_ValPond_Pred

cible TYPE Predicat

r_sem TYPE R_Pred -->

<!--CONTRAINTE R_ValPond_Concept

cible TYPE Concept

r_sem TYPE R_Concept -->

<!--CONTRAINTE R_ValPond_Pred_Concept

cible TYPE Concept

r_sem TYPE R_Pred_Concept -->

<!--CONTRAINTE R_ValPond_Concept_Pred

cible TYPE Predicat

r_sem TYPE R_Concept_Pred -->

<!--CONTRAINTE Corresp_Arg_Arg

informe_arg_precise_source_l TYPE InformeArg

informe_arg_precise_cible_l TYPE InformeArg -->

<!--CONTRAINTE R_Usem

r_sem_inverse TYPE R_Usem

incompatible_l TYPE R_Usem

est_un_l TYPE R_Usem -->

<!--CONTRAINTE R_Pred

r_sem_inverse TYPE R_Pred
```

```
incompatible_l TYPE R_Pred

est_un_l TYPE R_Pred -->

<!--CONTRAINTE R_Concept

r_sem_inverse TYPE R_Concept

incompatible_l TYPE R_Concept

est_un_l TYPE R_Concept -->

<!--CONTRAINTE R_Pred_Concept

r_sem_inverse TYPE R_Concept_Pred

incompatible_l TYPE R_Pred_Concept

est_un_l TYPE R_Pred_Concept -->

<!--CONTRAINTE R_Concept_Pred

r_sem_inverse TYPE R_Pred_Concept

incompatible_l TYPE R_Concept_Pred

est_un_l TYPE R_Concept_Pred -->

<!--CONTRAINTE Assoc_ListePred

listepred TYPE ListePred -->

<!--CONTRAINTE ListePred

pred_instancie_l TYPE PredInstancie -->

<!--CONTRAINTE Corresp_Usyn_Usem

usem_cible TYPE Usem

correspondance TYPE Correspondance -->

<!--CONTRAINTE Correspondance -

contraint_description TYPE ContraintDescription

corresp_arg_pos_l TYPE Corresp_Arg_Pos_Simple

|Corresp_Arg_Pos_Flottant -->
```

```
<!--CONTRAINTE ContraintDescription

contraint_intervconst TYPE ContraintIntervConst

contraint_struct_interne TYPE ContraintStructInterne

contraint_construction TYPE ContraintConstruction -->

<!--CONTRAINTE ContraintIntervConst

contraint_syntagme_l TYPE ContraintSyntagme -->

<!--CONTRAINTE ContraintSyntagme

syntagme TYPE Syntagme

ajoute_trait_synt_l TYPE (Trait_Lex|Trait_Introd

|Trait_Prep|Trait_Conj

|Trait_ProRel

|Trait_ProIntrog

|Trait_Mode|Trait_Temps

|Trait_Personne

|Trait_Genre

|Trait_Nombre

|Trait_NombrePosseur

|Trait_SsCatMorph

|Trait_SsCatSynt

|Trait_Aux

|Trait_Pronominal

|Trait_Neg|Trait_Accord

|Trait_Passif

|Trait_Tournure

|Trait_Coref

|Trait_Aspect
```

```
|Trait_Bin|Trait_Libre) -->
<!--CONTRAINTE AjouteTrait_Sem
trait_sem_valpond TYPE Trait_Sem_ValPond -->
<!--CONTRAINTE ContraintConstruction
contraint_position_l TYPE ContraintPosition -->
<!--CONTRAINTE ContraintStructInterne
contraint_position_l TYPE ContraintPosition -->
<!--CONTRAINTE ContraintPosition_mdc
contraint_position_l TYPE ContraintPosition -->
<!--CONTRAINTE ContraintPosition
contraint_syntagme_l TYPE ContraintSyntagme -->
<!--CONTRAINTE Corresp_Arg_Pos_Flottant
position TYPE Position -->
```

4. Entités *semant.ent*

```
<!--Consortium GENELEX @(#) semant.ent 2.1@(#) 94/09/07 14:35:28 -->
<!ENTITY % pPonderation
"PROTOTYPIQUE|DEFINITOIRE|ACCESSOIRE|EXCEPTIONNEL
|CONNOTATIF">
<!ENTITY % pArgInclus
"INCLUS|PAS_INCLUS">
<!ENTITY % pPorteeCorresp
"EXTERNE|STRUCT_INTERNE|COMP_INTERNE">
<!ENTITY % pStatutInfoArg
"DEFAUT|VERIF|ENRICHIT|DEFAUT_VERIF">
```

```
<!ENTITY % pTypeR_Usem
"PARADIGMATIQUE|DERIVATION|COLLOCATION">
<!ENTITY % pSsTypeR_Usem
"SYNONYMIE|CONTRAIRE|OPPOSITION|CONVERSE
|TAXINOMIE|PARTIE_TOUT|STRICTE|NON_STRICTE">
<!ENTITY % pTypeR_PredOuConcept
"GENERALISATION|PARTICULARISATION|ESSENTIEL">
<!ENTITY % pRoleListePred
"A_PRESUPPOSITION|A_IMPLICATION|A_DEFINITION
|A_EXPLICITATION|PARTICIPE_A">
<!ENTITY % pStatutListePred
"SCENARIO|DEFINITION">
<!ENTITY % pTypeListePred
"ORDRE_TEMP|ORDRE_SPATIAL|SIMULTANEITE">
<!ENTITY % pConcerne
"AFFIXE|AUTRE">
<!ENTITY % pTypeTrait_Sem
"PROPRIETE|CLASSE|DOMAINE|DISTINCTIF|PRAGMATIQUE
|DIVERS|CONNOTATION">
<!ENTITY % pValuation
"MONOVALUE|MULTIVALUE">
<!ENTITY % pTypeListe
"BINAIRE|LISTE_FERMEE|LISTE_OUVERTE">
<!ENTITY % pStructureListe
"TREILLIS_TOTAL|TREILLIS_PARTIEL|HIERARCHIE_TOTALE
```

```
|HIERARCHIE_PARTIELLE">
<!ENTITY % pTypePredOuConcept
"LEXICAL|GENERALISE|PRIMITIF|LEXICAL_PRIMITIF
|TROU_LEXICAL">
<!ENTITY % pStatutAjoute
"FILTRE_AJOUTE|FILTRE|FORCE">
```

5. Entités *custom.ent*

```
<!--Consortium GENELEX @(#) custom.ent 3.3@(#) 94/09/08 13 : 26 : 36 -->
<!-- Ce fichier est accessible a l'utilisateur qui peut y definir
des valeurs supplementaires d'attribut en remplacant le texte
"_VALEURS_A_DEFINIR_XXX" de l'entite par la liste des valeurs dont il
veut disposer : "VALEUR1|VALEUR2|...|VALEURn"
Pour un certain nombre des elements utilisant ces entites, une liste
minimale de valeurs est definie par ailleurs dans les fichiers
"syntaxe.ent" et "semant.ent".
On pourra envisager d'etendre ce fonctionnement a d'autres attributs.
-->
<!ENTITY % pEtiquetteSynt_cust "_VALEURS_A_DEFINIR_ES" >
<!ENTITY % pSsCatSynt_cust "_VALEURS_A_DEFINIR_SCS" >
<!ENTITY % pFonction_cust
"TETE|SUJET|OBJET_DIRECT|OBJET_INDIRECT|ATTRIBUT_SUJET
|ATTRIBUT_OBJET|EPITHETE_GAUCHE|EPITHETE_DROIT
```

```
|SPECIFIEUR|MODIFIEUR|GENITIF" >

<!ENTITY % pRoleTh_cust

"AGENT|PATIENT|DESTINATAIRE|SOURCE|BUT|CAUSE|MANIERE

|LOCATIF|TEMPS|INSTRUMENT|THEME" >

<!ENTITY % pPonderation_cust "_VALEURS_A_DEFINIR_PO">

<!ENTITY % pStatutInfoArg_cust "_VALEURS_A_DEFINIR_SIA">

<!ENTITY % pTypeTrait_Sem_cust "_VALEURS_A_DEFINIR_TTS">

<!ENTITY % pTypeR_Usem_cust "_VALEURS_A_DEFINIR_TRU" >

<!ENTITY % pSsTypeR_Usem_cust "_VALEURS_A_DEFINIR_SRU" >

<!ENTITY % pTypeR_PredOuConcept_cust "_VALEURS_A_DEFINIR_TR" >

<!ENTITY % pRoleListePred_cust "_VALEURS_A_DEFINIR_RLP">

<!ENTITY % pStatutListePred_cust "_VALEURS_A_DEFINIR_SLP" >

<!ENTITY % pTypeListePred_cust "_VALEURS_A_DEFINIR_TLP" >
```

6. DTD *syntaxe.dtd*

```
<!--Consortium GENELEX @(#) syntaxe.dtd 4.2@(#) 94/06/23 14:14:36 -->


<!-- **************A L'ADRESSE DES UTILISATEURS ******************

Vos remarques concernant la DTD seront etudiees par le consortium

GENELEX. Celui-ci assurera la diffusion de la nouvelle version qui

pourrait en decouler.

********************************************************************* -->

<!ELEMENT GenelexSyntaxe - O (

Usyn+ &

Description+ &

Self+ &
```

IntervConst* &

ComportAppele* &

Optionnalite* &

Construction* &

Position_C* &

Position_S* &

Insertion* &

Syntagme_T* &

Syntagme_NT_C* &

Syntagme_NT_S* &

MdC* &

TransfDescription* &

ModifConstruction* &

ModifPosition* &

TransfSyntagme* &

ModifSyntagme_T* &

ModifSyntagme_NT* &

ModifIntervConst* &

Trait_Lex* &

Trait_Introd* &

Trait_Prep* &

Trait_Conj* &

Trait_ProRel* &

Trait_ProIntrog* &

Trait_RefLex* &

Trait_RefIntrod* &

```
Trait_RefPrep* &

Trait_RefConj* &

Trait_RefProRel* &

Trait_RefProIntrog* &

Trait_Mode* &

Trait_Temps* &

Trait_Personne* &

Trait_Genre* &

Trait_Nombre* &

Trait_NombrePosseur* &

Trait_SsCatMorph* &

Trait_SsCatSynt* &

Trait_Aux* &

Trait_Pronominal* &

Trait_Neg* &

Trait_Accord* &

Trait_Passif* &

Trait_Tournure* &

Trait_Coref* &

Trait_Aspect* &

Trait_Bin* &

Trait_Libre* &

RoleTh* &

Fonction*)>

<!-- ********************************************************** -->
```

```
<!-- ***** UNITE SYNTAXIQUE ET DESCRIPTION ***** -->
<!-- ********************************************************** -->
<!ELEMENT Usyn - O ((Composition* & TransfUsyn*), Corresp_Usyn_Usem*)>
<!ATTLIST Usyn
id ID #REQUIRED
%pGlose
attestation CDATA #IMPLIED
combve IDREF #IMPLIED
description IDREF #REQUIRED
description_l IDREFS #IMPLIED
transfdescription_l IDREFS #IMPLIED>
<!-- Le champ attestation permet de preciser la source de l'emploi
releve (nom ou titre du dictionnaire, du texte d'auteur, ou de
l'article de linguistique).
L'attribut description note la description de base,
la liste description_l enregistre les descriptions transformees,
la liste transfdescription_l note les transformations entre les
descriptions associees a l'Usyn ; ces transformations peuvent operer
entre la description de base et les transformees mais aussi sur les
transformees entre elles. -->
<!ELEMENT Corresp_Usyn_Usem - O EMPTY>
<!ATTLIST Corresp_Usyn_Usem
usem_cible IDREF #REQUIRED
correspondance IDREF #IMPLIED>
<!-- Ces elements sont enchasses dans les Usyn, ils servent a la
connexion entre les couches syntaxique et semantique. Pour une Usyn,
```

```
il y a autant de Corresp_Usyn_Usem que d'Usem pointees par cette
Usyn. Une Usem peut etre pointee par plusieurs Usyn, avec a chaque
fois les informations de correspondance dans les Corresp_Usyn_Usem.
usem_cible designe l'Usem pointee.
correspondance precise toutes les informations completant
cette correspondance (fitrage, correspondance argument-position,
enrichissement semantique du au contexte) et peut etre vide si rien
n'est a preciser.
Une Usem est pointee via l'attribut usem_cible par autant de
Corresp_Usyn_Usem qu'elle a de contextes syntaxiques possibles
(decrits par les descriptions associees aux Usyn).
-->


<!ELEMENT Description - O (Condition*) >
<!ATTLIST Description
id ID #REQUIRED
%pGlose
um_representante CDATA #IMPLIED
self IDREF #REQUIRED
construction IDREF #IMPLIED>
<!-- ********************************************************** -->
<!-- ***** SELF ***** -->
<!-- ********************************************************** -->
<!ELEMENT Self - O EMPTY>
<!ATTLIST Self
```

```
id ID #REQUIRED
syntagme_nt_s IDREF #IMPLIED
syntagme_nt_s_l IDREFS #IMPLIED
transfsyntagme_l IDREFS #IMPLIED
IntervConst IDREF #IMPLIED
comportappele_l IDREFS #IMPLIED>
<!-- Le champ syntagme_nt_s n'est instancie que pour les Usyn
composees, il en exprime alors la structure interne, eventuellement
reduite a l'etiquette syntagmatique, par un Syntagme_NT_S avec ou
sans reecriture
Ex : mettre en marche SV
Les champs syntagme_nt_s_l et transfsyntagme_l ne concernent de
meme que les Usyn composees et servent a noter d'eventuelles
transformations sur la structure interne.
Le champ IntervConst donne les realisations de Self intervenant
dans la construction externe :
- en tant qu'occupant de la construction si celle-ci decrit
un contexte d'apparition dans lequel s'inscrit le Self,
- en tant que predicat associe a une construction decrivant un
schema de complementation.
La liste comportappele_l permet d'indiquer les alternatives
de comportement de Self en tant qu'appele par un element
non decrit dans sa construction -->
<!ELEMENT IntervConst - O EMPTY>
<!ATTLIST IntervConst
id ID #REQUIRED
```

```
fonction IDREF #IMPLIED
roleth_l IDREFS #IMPLIED
syntagme_t_l IDREFS #REQUIRED>
<!-- L'element IntervConst comporte des syntagmes terminaux.
On peut y exprimer des variations de realisation de Self :
Ex : N + [Nombre : PLURIEL]
Ex : V
V + [Pronominal : SE]
V + [Pronominal : SE_EN]
Pour les mots simples, on autorise un ecart entre la categorie
fonctionnelle et la categorie grammaticale de l'unite morphologique.
Ex : description du comportement adjectival de
l'Unite Morphologique du NOM abricot
Pour les composes syntaxiques, on rend compte de la categorie
fonctionnelle (externe) du compose qui peut differer de son
etiquette de syntagme structurel interne
Ex : mettre en Ïuvre (VERBE / SV)
On peut de plus indiquer sur cet element une fonction et des roles
thematiques : ce sont les valeurs portees par le Self lorsqu'il
s'insere dans la construction. -->
<!ELEMENT ComportAppele - O EMPTY>
<!ATTLIST ComportAppele
id ID #REQUIRED
fonction IDREF #IMPLIED
roleth_l IDREFS #IMPLIED
```

```
syntagme_t IDREF #REQUIRED>
<!-- ComportAppele note un comportement de Self en tant qu'appele par
un element non decrit dans sa construction.
Un comportement regroupe :
- une etiquette syntagmatique terminale et une liste de traits,
exprimant la categorie fonctionnelle d'appele et les traits
restrictifs associes :
cette association est equivalente a un syntagme terminal
Ex : PREP + [SsCatSynt : LIEU]
(si le Self porte un IntervConst, l'etiquette syntagmatique
du syntagme du ComportAppele doit etre la meme que l'etiquette
de l'un des syntagmes dudit IntervConst)
- une fonction en tant qu'appele
- une liste de roles thematiques -->
<!-- ********************************************************** -->
<!-- ***** CONSTRUCTION ***** -->
<!-- ********************************************************** -->
<!ENTITY % pConstSyntNT
"%pGlose
etiquettesynt (SANS_E
|%pEtiquetteSynt_NT
|%pEtiquetteSynt_cust) SANS_E
solidarite CDATA #IMPLIED
optionnalite IDREF #IMPLIED">
<!-- L'attribut solidarite indique par des tirets les paires
de positions solidaires ;
```

```
Ex : P0 SELF-P1-P2 -->
<!ELEMENT Optionnalite - O (ConditionOpt*)>
<!ATTLIST Optionnalite
id ID #REQUIRED
%pGlose
libelle CDATA #REQUIRED>
<!ELEMENT ConditionOpt - O (SiOpt+, AlorsOpt+)>
<!ELEMENT (SiOpt|AlorsOpt) - O EMPTY>
<!ATTLIST (SiOpt|AlorsOpt)
negation (%pBooleen) NON
nieme_position NUMBER #REQUIRED>
<!-- L'optionnalite est exprimee :
- d'une part par un libelle indiquant pour chaque position par
des parentheses si son effacement est possible dans une
realisation de la construction ; toutes les positions de la
construction ou du syntagme sont rappelees dans ce champ,
le point d'insertion du Self n'y apparait pas
Ex : P0 (P1) P2 (P3)
- d'autre part par des conditions exprimant d'eventuelles
interdependances entre les positions -->
<!ELEMENT Construction - O EMPTY>
<!ATTLIST Construction
id ID #REQUIRED
%pConstSyntNT
squelettique (%pBooleen) NON
```

```
listepositions (%pTypeListPos) FERMEE

insereself NUMBER #IMPLIED

trait_l IDREFS #IMPLIED

position_c_l IDREFS #REQUIRED>

<!-- L'attribut squelettique indique si l'element est un squelette

de construction - donc a mettre en regard avec un

ModifConstruction pour obtenir une Construction a part entiere.

L'attribut listepositions note si la liste position_c_l est

a comprendre comme une liste FERMEE - toutes les positions

sont donnees - ou OUVERTE.

L'attribut insereself indique lorsqu'il est renseigne le point

d'insertion de Self dans la liste de positions : avant la

position de rang correspondant a sa valeur.

La liste position_c_l renvoie a des Position_C.

Il s'agit d'une liste ordonnee, l'ordre des elements dans

la liste correspond a l'ordre canonique (valeur initiale = 0),

et les elements pourront par la suite etre references par

leur rang dans cette liste -->

<!-- ********************************************************** -->

<!-- ***** POSITION ET INSERTION ***** -->

<!-- ********************************************************** -->

<!ENTITY % pPosition

"%pGlose

repetable (SANS_B|%pBooleen) SANS_B

fonction IDREF #IMPLIED

roleth_l IDREFS #IMPLIED">
```

```
<!-- L'attribut repetable indique si une position peut etre
realisable plusieurs fois -->
<!ELEMENT (Position_C|Position_S) - O EMPTY>
<!ATTLIST Position_C
id ID #REQUIRED
%pPosition
syntagme_c_l IDREFS #IMPLIED
transfsyntagme_l IDREFS #IMPLIED>
<!ATTLIST Position_S
id ID #REQUIRED
%pPosition
syntagme_s_l IDREFS #REQUIRED
transfsyntagme_l IDREFS #IMPLIED>
<!-- L'attribut syntagme_c_l reference les occupants possibles
de Position_C :
- syntagme terminal (Syntagme_T)
- syntagme non terminal dont la reecriture est ou non
decrite (Syntagme_NT_C)
L'attribut syntagme_s_l reference les occupants possibles
de Position_S :
- syntagme terminal (Syntagme_T)
- syntagme non terminal dont la reecriture est ou non
decrite (Syntagme_NT_S) -->
<!ELEMENT Insertion - O (CheminPosition)>
<!ATTLIST Insertion
```

```
id ID #REQUIRED
obligatoire (SANS_B|%pBooleen) SANS_B
retire_syntagme_c_l IDREFS #IMPLIED
retire_transfsyntagme_l IDREFS #IMPLIED>
<!-- L'insertion dans un syntagme structurel est utilisee uniquement
pour representer le cas d'une insertion renvoyant a une position
d une position decrite dans la construction syntaxique externe :
l element CheminPosition donne acces a cette position.
Les attributs de l'Insertion indiquent les eventuelles modifications
a appliquer a la position referencee : retrait de syntagmes et
transformations entre syntagmes. La repetabilite reste celle de
la position externe referencee.
Ex : le compose "mettre en Ïuvre" a dans sa construction
externe une position objet direct contenant par exemple
un syntagme nominal et un pronom personnel :
mettre en Ïuvre un processus, le mettre en Ïuvre
l'insertion n est ici possible que pour le SN :
mettre un processus en Ïuvre
L'attribut obligatoire indique si, lors d'une realisation de la
position externe referencee, le phenomene d'insertion est
obligatoire ou facultatif -->
<!-- ******************************************************* -->
<!-- ***** CONDITION ***** -->
<!-- ******************************************************* -->
<!ELEMENT Condition - O (Si+, Alors+)>
<!ATTLIST Condition
```

```
appellation CDATA #IMPLIED>
<!-- Les Conditions permettent d'exprimer des contraintes mutuelles
entre realisations de positions.
Ex : si P0 == P[SsCatSynt : COMPLETIVE]
alors P1 != P[SsCatSynt : COMPLETIVE]
Les listes de Predicats (Si et Alors) permettent d'exprimer
des conjonctions sur ces predicats.
Les disjonctions sont exprimees au moyen de la liste de Conditions
portee par la Description (liste "et"). -->
<!ELEMENT (Si|Alors) - O (CheminPosition|CheminSyntagme
|SelectIntervConst)>
<!ATTLIST (Si|Alors)
portee (%pPortee) EXTERNE
negation (%pBooleen) NON>
<!-- Un predicat selectionne :
- un syntagme ou une position de la construction externe
(champ portee EXTERNE),
- un syntagme ou une position du syntagme structurel du Self
(champ portee INTERNE),
- une realisation du Self en tant qu intervenant de
construction (champ portee INTERVENANT).
La selection se fait
- pour un syntagme grace au CheminSyntagme,
- pour une position grace au CheminPosition,
- pour une realisation en tant qu intervenant grace
```

```
au SelectIntervConst.
Un predicat porte eventuellement une negation qui exprime
l inhibition du syntagme ou de la position pointee. -->
<!ELEMENT CheminSyntagme - O (CheminSyntagme?)>
<!ATTLIST CheminSyntagme
nieme_position NUMBER 0
syntagme IDREF #REQUIRED>
<!-- Cet element permet de selectionner un syntagme particulier.
La recursivite est utilisee pour descendre dans une eventuelle
reecriture. L'element aboutit toujours a un syntagme.
Les positions sont referencees par nieme_position qui indique
leur rang dans la liste ou elles apparaissent ;
la valeur 0 reference le premier element de la liste. -->
<!ELEMENT CheminPosition - O (CheminSyntagme?,PositionBut)>
<!-- Pour une construction ou un syntagme donne, cet element permet
de selectionner une de ses positions - eventuellement apparaissant
dans une reecriture de syntagme -.
L'element PositionBut indique la position selectionnee ;
si celle-ci apparait dans une reecriture de syntagme,
on utilise l'element CheminSyntagme pour atteindre
ledit syntagme -->
<!ELEMENT PositionBut - O EMPTY>
<!ATTLIST PositionBut
nieme_position NUMBER 0>
<!ELEMENT SelectIntervConst - O EMPTY>
<!ATTLIST SelectIntervConst
```

```
syntagme_t IDREF #REQUIRED>

<!-- ********************************************************** -->

<!-- ***** SYNTAGMES TERMINAUX ET NON TERMINAUX ****** -->

<!-- ********************************************************** -->

<!ELEMENT Syntagme_T - O EMPTY>

<!ATTLIST Syntagme_T

id ID #REQUIRED

%pGlose

etiquettesynt (SANS_E

|%pEtiquetteSynt_T

|%pEtiquetteSynt_cust) SANS_E

trait_l IDREFS #IMPLIED>

<!ELEMENT Syntagme_NT_C - O EMPTY>

<!ATTLIST Syntagme_NT_C

id ID #REQUIRED

%pConstSyntNT

listepositions (%pTypeListPos) FERMEE

insereself NUMBER #IMPLIED

trait_l IDREFS #IMPLIED

position_c_l IDREFS #IMPLIED>

<!-- La liste position_c_l renvoie a des Position_C.

Il s'agit d'une liste ordonnee, l'ordre des elements dans

la liste correspond a l'ordre canonique et les elements pourront

par la suite etre references par leur rang dans cette liste. -->

<!ELEMENT Syntagme_NT_S - O EMPTY>
```

```
<!ATTLIST Syntagme_NT_S
id ID #REQUIRED
%pConstSyntNT
listepositions (%pTypeListPos) FERMEE
insereinsertion_l NUMBERS #IMPLIED
trait_l IDREFS #IMPLIED
position_s_l IDREFS #IMPLIED
insertion_l IDREFS #IMPLIED>
<!-- La liste position_s_l renvoie a des Position_S ;
la liste insertion_l renvoie a des Insertions.
Il s'agit de listes ordonnees, leurs elements pourront par la suite
etre references par leur rang d'apparition dans leur liste.
La liste d'entiers insereinsertion_l indique les points ou
s'inserent parmi les Positions les Insertions presentes dans la
liste insertion_l -->
<!-- Le Syntagme_T est un occupant de position terminal (etiquette
reduite a la Categorie grammaticale ou a "e" pour la trace).
Le Syntagme_NT_C/S est un occupant de position non terminal,
on utilisera sa liste de positions (#IMPLIED) si l'on veut preciser
sa reecriture.
Le Syntagme_NT_S sert de plus pour decrire la structure interne
d'une unite composee (champ syntagme_nt_s du Self).
La liste trait_l renvoie a des traits restrictifs et permet ainsi
de specifier sur un syntagme un ensemble de contraintes :
contraintes lexicales, morphologiques, morpho-syntaxiques,
syntaxico-semantiques, voire semantiques.
```

```
Le champ appellation permettra de noter le nom usuel du syntagme.
Ex : Syntagme_NT_C
etiquettesynt = "P"
trait_l -> [Mode : INFINITIF]
appellation = "phrase infinitive"
La categorie "e" permet de noter les traces pour les tenants
de la grammaire generative et de les considerer comme des syntagmes
fantomes auxquels on peut associer les restrictions necessaires.
Ex : Syntagme_T
etiquettesynt = "e"
trait_l -> [Personne : 3][Nombre : SINGULIER]
appellation = "elt vide" -->
<!-- ******************************************************** -->
<!-- ***** COMPOSITION ***** -->
<!-- ******************************************************** -->
<!ELEMENT Composition - O (R_ComposeUm|R_ComposeUsyn)+>
<!-- Les elements Composition portes par l unite syntaxique
enregistrent les listes alternatives de lexicalisations :
Ex : (avoir admiration pour)
(eprouver admiration pour)
(eprouver admiration envers)
(porter admiration a) -->
<!-- La liste contenue de R_ComposeUm et R_ComposeUsyn donne la
liste des composants pour une alternative de composition donnee. -->
<!-- On referera dans la suite les composants par des traits RefLex
```

```
portant deux valeurs d indices :

- l indice de la compositon dans la liste de compositions

- l indice du composant dans la liste des composants

Ex. : [RefLex : 1,2] :

Ce mecanisme sera utilise :

- soit dans le syntagme structurel interne de Self,

- soit dans le mode de calcul MdC. -->

<!ENTITY % pCompose

"type (%pTypeComposant) APPELE">

<!ELEMENT R_ComposeUm - O (RestrictUm*)>

<!ATTLIST R_ComposeUm

%pCompose

um IDREF #REQUIRED>

<!ELEMENT R_ComposeUsyn - O EMPTY>

<!ATTLIST R_ComposeUsyn

%pCompose

usyn IDREF #REQUIRED

mdc IDREF #IMPLIED>

<!-- Dans le cas ou l'on veut noter la formation du compose selon

l'Usyn composante :

- lexicalisation de ses positions par d'autres composants

(seulement dans le cas d'un comportement d'appelant)

et/ou

- heritage de ses positions avec eventuellement ajout de

restrictions,

on associe un MdC decrivant l'ensemble de ces phenomenes.
```

NB : La lexicalisation d'une position d une Usyn appelante par un
autre composant est faite par l'introduction d'un Syntagme
portant un trait [RefLex : nieme_cposition,nieme_cposant] dans
le MdC de l'Usyn. Ce Syntagme peut de plus porter d autres
traits restrictifs ; on devra dans ce cas s'assurer que ces
traits sont compatibles avec le MdC et le Self de l Usyn
fonctionnant en tant que composant appele.
Ex : Le MdC de l appelante indique :
SN[RefLex : 3,1][SsCatSynt : DET_VIDE]
L'appelee (3,1) ne devra pas dans son MdC imposer
une lexicalisation de sa position Determinant. -->
<!ELEMENT MdC - O (HeritePosition* & FiltreSelf?)>
<!ATTLIST MdC
id ID #REQUIRED
%pGlose>
<!-- Le MdC decrit le mode de composition d une Usyn Composee sur une
de ses Usyn composantes. Il permet de specifier :
- les contraintes emises par l Usyn composee sur les Usyn
composantes,
- l'organisation mutuelle des Usyn composantes, en indiquant
quel composant occupe quelle position d'un autre composant.
Le MdC s applique a l usyn composante et permet de
- heriter des positions de sa construction
- filtrer son Self
Une position non referencee par le MdC est inhibee. Lorsqu aucun

```
filtre n est precise sur le Self, celui-ci est par defaut herite -->
<!ELEMENT HeritePosition - O (CheminPosition)>
<!ATTLIST HeritePosition
destination (%pDestination) EXTERIEUR
optionnel (HERITAGE
|%pBooleen) HERITAGE
modifposition IDREF #IMPLIED>
<!-- L'attribut destination indique si la position heritee se retrouve
dans l'exterieur ou l'interieur du compose.
L'attribut optionnel permet de noter d'eventuelles modifications
de l'optionnalite de la position heritee. -->
<!ELEMENT FiltreSelf - O EMPTY>
<!ATTLIST FiltreSelf
modifIntervConst IDREF #IMPLIED
modifsyntagme_nt IDREF #IMPLIED>
<!-- Le FiltreSelf doit au moins realiser une des deux operations :
- Modification de l'IntervConst
- Modification du syntagme structurel interne. -->
<!-- ********************************************************** -->
<!-- ***** TRANSFORMATIONS ***** -->
<!-- ********************************************************** -->
<!-- Il existe trois types de transformations :
- TransfUsyn : transformations operant entre deux Unites Syntaxiques
- issues de la meme Um
Ex : neutralite
- issues d'Um differentes
```

```
Ex : derivation syntaxique
- TransfDescription : transformations operant entre deux
Descriptions (c'est-a-dire deux couples Self/Construction).
Ex : passivation
- TransfSyntagme : transformations operant entre des occupants
de Position, c'est-a-dire des Syntagmes, terminaux ou non.
Ex : pronominalisation -->
<!ELEMENT TransfUsyn - O (ModifDescription?)>
<!ATTLIST TransfUsyn
%pGlose
usyn_resultat IDREF #REQUIRED>
<!-- Les TransfUsyn sont pointees par les Usyn origine et indiquent
les Usyn resultat.
Elles s'appliquent entre la Description de base de l'Usyn origine
et la Description de base de l'Usyn resultat -->
<!ELEMENT TransfDescription - O (ModifDescription?)>
<!ATTLIST TransfDescription
id ID #REQUIRED
%pGlose
description_origine IDREF #REQUIRED
description_resultat IDREF #REQUIRED>
<!-- Les TransfDescriptions sont portees par l'Usyn et operent entre
les Descriptions de cette Usyn.
L'origine d une TransfDescription peut etre la Description de base
mais aussi une Description transformee de l'Usyn. -->
```

```
<!ELEMENT ModifDescription - O EMPTY>
<!ATTLIST ModifDescription
%pGlose
modifconstruction IDREF #IMPLIED
modifIntervConst IDREF #IMPLIED
modifsyntagme_nt IDREF #IMPLIED>
<!-- Les ModifDescriptions peuvent rendre compte de trois phenomenes :
- transformation sur la construction
- transformation sur les realisations de Self associees
a la construction externe en tant qu'occupant ou predicat
(champ IntervConst de Self)
Ex : V -> V[Passif : PLUS]
- transformation sur le syntagme structurel interne decrivant
un compose
Ex : pour les beaux yeux de SN
-> pour ses beaux yeux
Dans le premier cas, on peut toujours choisir entre un mode
calculatoire ou descriptif :
- en mode calculatoire, la Construction pointee par la
Description resultat est declaree squelettique : la
construction resultat elle meme est a construire a partir
de ce squelette en lui appliquant les modifications
indiquees sur ModifConstruction
- en mode descriptif, la Construction pointee par la
Description resultat est totalement decrite : le
ModifConstruction indique alors des correspondances,
```

```
a des degres de finesse variable, entre Construction
origine et Construction resultat.
Dans les deux autres cas, les resultats - IntervConst et
Syntagme_NT_S - sont totalement decrits : on est en mode
descriptif -->
<!ELEMENT ModifConstruction - O (TransfPosition*)>
<!ATTLIST ModifConstruction
id ID #REQUIRED
%pGlose
etiquettesynt (HERITAGE
|%pEtiquetteSynt_NT
|%pEtiquetteSynt_cust) HERITAGE
insereself NUMBER #IMPLIED
solidarite CDATA #IMPLIED
optionnalite IDREF #IMPLIED
retire_trait_l IDREFS #IMPLIED
ajoute_trait_l IDREFS #IMPLIED>
<!-- En mode descriptif, cet element permet de noter - eventuellement
de facon partielle - des informations sur le passage de la
construction origine a la construction resultat ; il peut
minimalement s'agir de mises en correspondance de positions
par le contenu TransfPosition*.
En mode calculatoire, cet element permet de construire la
construction resultat a partir de la construction origine
et du squelette : le squelette est habille avec des elements
```

```
ou attributs provenant de la construction origine ou precises sur
le ModifConstruction.
Si les attributs etiquettesynt, insereself, solidarite,
optionnalite et ceux groupes par pGlose
- sont renseignes : ils remplissent (voire ecrasent)
les attributs correspondants du squelette
- ne sont pas renseignes : les attributs correspondants
et non renseignes sur la construction squelette sont herites
de la construction origine
Les traits de la Construction origine sont herites moyennant
des retraits et ajouts notes ici.
La liste contenue de TransfPositions exprime la formation
des positions de la construction resultat a partir des positions
du squelette et des positions de la construction origine. -->
<!ELEMENT TransfPosition - O (CheminPosition, CheminPosition?)>
<!ATTLIST TransfPosition
%pGlose
modifposition IDREF #IMPLIED>
<!-- Selection de la Position d'origine
Le premier element contenu CheminPosition pointe vers une position
de la construction a l'origine de la transformation.
Selection de la Position resultat
La position resultat est eventuellement selectionnee par le
second element contenu CheminPosition
L'attribut ModifPosition indique les modifications a operer pour
passer de la position origine a la position resultat -->
```

```
<!ELEMENT ModifPosition - O EMPTY>
<!ATTLIST ModifPosition
id ID #REQUIRED
%pGlose
repetable (HERITAGE
|%pBooleen) HERITAGE
fonction IDREF #IMPLIED
roleth_l IDREFS #IMPLIED
retire_syntagme_l IDREFS #IMPLIED
retire_transfsyntagme_l IDREFS #IMPLIED
ajoute_syntagme_l IDREFS #IMPLIED
ajoute_transfsyntagme_l IDREFS #IMPLIED
transfsyntagme_l IDREFS #IMPLIED>
<!-- Le fonctionnement de ModifPosition est similaire a celui du
ModifConstruction ; on aura en particulier la meme alternative
entre un fonctionnement calculatoire - la position resultat est
a construire a partir de la position pointee sur le squelette
et les informations notees ici - et un fonctionnement descriptif
- la position resultat est deja totalement renseignee. -->
<!ELEMENT TransfSyntagme - O EMPTY>
<!ATTLIST TransfSyntagme
id ID #REQUIRED
%pGlose
syntagme_origine IDREF #REQUIRED
syntagme_resultat IDREF #IMPLIED
```

```
modifsyntagme IDREF #IMPLIED>
<!-- Les TransfSyntagme notent les relations de transformation
existant entre :
- des syntagmes occupant une meme position - Syntagmes
terminaux ou non -, les syntagmes a l'origine et au
resultat de la transformation sont alors tous deux
necessairement renseignes,
- des syntagmes mis en correspondance lors d'une
transformation de plus haut niveau ; dans ce cas,
le syntagme resultat ne sera pas necessairement
renseigne si on est a l'interieur d'une transformation
de Construction fonctionnant en mode calculatoire. -->
<!ENTITY % pModifSyntagme
"%pGlose
retire_trait_l IDREFS #IMPLIED
ajoute_trait_l IDREFS #IMPLIED">
<!ELEMENT ModifSyntagme_T - O EMPTY>
<!ATTLIST ModifSyntagme_T
id ID #REQUIRED
%pModifSyntagme>
<!ELEMENT ModifSyntagme_NT - O ((TransfPosition|TransfInsertion)*)>
<!ATTLIST ModifSyntagme_NT
id ID #REQUIRED
%pModifSyntagme
etiquettesynt (HERITAGE
|%pEtiquetteSynt_NT
```

```
|%pEtiquetteSynt_cust) HERITAGE
insereself NUMBER #IMPLIED
insereinsertion_l NUMBERS #IMPLIED
retire_position_l NUMBERS #IMPLIED
retire_insertion_l NUMBERS #IMPLIED
solidarite CDATA #IMPLIED
optionnalite IDREF #IMPLIED>
<!-- Modifier un Syntagme consiste a :
- retirer ou ajouter des traits,
- et dans le cas d'un syntagme non terminal :
- changer l'etiquette,
- changer l'optionnalite, la solidarite, et
les eventuels points d'insertion notes.
- modifier les positions et insertions de
l'eventuelle reecriture -->
<!ELEMENT TransfInsertion - O (CheminInsertion, CheminInsertion)>
<!-- L'element TransfInsertion met en rapport une insertion du
syntagme de structure origine et l'insertion correspondante
du syntagme de structure resultat. -->
<!ELEMENT CheminInsertion - O (CheminSyntagme?,InsertionBut)>
<!ELEMENT InsertionBut - O EMPTY>
<!ATTLIST InsertionBut
nieme_insertion NUMBER 0>
<!-- L attribut nieme_insertion selectionne une insertion par son rang
dans la liste insertion_l ou elle apparait -->
```

```
<!ELEMENT ModifIntervConst - O EMPTY>

<!ATTLIST ModifIntervConst

id ID #REQUIRED

fonction IDREF #IMPLIED

roleth_l IDREFS #IMPLIED

retire_syntagme_t_l IDREFS #IMPLIED

ajoute_syntagme_t_l IDREFS #IMPLIED

transfsyntagme_l IDREFS #IMPLIED>

<!-- L'element ModifIntervConst permet de noter les modifications

subies, en transformation ou en composition, par le contenu

IntervConst du Self. -->

<!-- ********************************************************** -->

<!-- ***** TRAITS RESTRICTIFS ***** -->

<!-- ********************************************************** -->

<!-- Les traits restrictifs permettent de specifier les syntagmes

auxquels ils sont combines

Ex : P[Conj : que][Mode : SUBJONCTIF] =>completive

P[SsCatSynt : RELATIVE] =>relative

P[Mode : INFINITIF] =>infinitive

P[SsCatSynt : COORDONNE] =>phrase coordonnee

P[SsCatSynt : SUBORDONNEE] =>subordonnee

SN[Nombre : PLURIEL]

SN[Coref : I] -->

<!--******************** Traits LEXICAUX ***********************-->

<!--************************************************************-->

<!ENTITY % pTrait_Lexical
```

```
"id ID #REQUIRED
valeur CDATA #IMPLIED
um IDREF #IMPLIED">
<!-- Ces traits permettent d'exprimer une restriction d'ordre lexical
sur un Syntagme,
- soit en entrant dans le champ "valeur" une chaine de caracteres
representant la graphie - ce moyen ne desambiguise pas
les homographes -,
- soit en referant une unite morphologique (par son identifiant).
On autorise un ecart (fonde sur l'ecart entre categorie
morpho-syntaxique et categorie fonctionnelle) entre la categorie
du syntagme qui supporte le trait lexical et la categorie de l'Um
referee par ce trait lexical
Ex : NOM[Lex : courageux[um : UM04]]
avec en Morphologie :
Um[id : UM04 ; catgram : ADJECTIF[umg : courageux]] -->
<!ELEMENT Trait_Lex - O EMPTY>
<!ATTLIST Trait_Lex
%pTrait_Lexical
saturesynt (%pBooleen) OUI>
<!ELEMENT Trait_Introd - O EMPTY>
<!ATTLIST Trait_Introd
%pTrait_Lexical>
<!ELEMENT Trait_Prep - O EMPTY>
<!ATTLIST Trait_Prep
```

```
%pTrait_Lexical>

<!ELEMENT Trait_Conj - O EMPTY>

<!ATTLIST Trait_Conj

%pTrait_Lexical>

<!ELEMENT Trait_ProRel - O EMPTY>

<!ATTLIST Trait_ProRel

%pTrait_Lexical>

<!ELEMENT Trait_ProIntrog - O EMPTY>

<!ATTLIST Trait_ProIntrog

%pTrait_Lexical>

<!ENTITY % pTrait_RefLexical

"id ID #REQUIRED

nieme_cposition NUMBER 0

nieme_cposant NUMBER 1">

<!-- Ces traits referent par leurs coordonnees (nieme_cposition

indique le rang dans la liste de Composition portee par l Usyn

et nieme_cpsosant le rang dans la liste mixte de R_ComposeUm/Usyn

portee par la Composition) l'Um ou Usyn entrant dans la

composition de l'Unite. La valeur 0 sur nieme_cposition refere

tous les composants de rang nieme_cpsant independamment des

compositions -->

<!ELEMENT Trait_RefLex - O EMPTY>

<!ATTLIST Trait_RefLex

%pTrait_RefLexical

saturesynt (%pBooleen) OUI>

<!ELEMENT Trait_RefIntrod - O EMPTY>
```

```
<!ATTLIST Trait_RefIntrod

%pTrait_RefLexical>

<!ELEMENT Trait_RefPrep - O EMPTY>

<!ATTLIST Trait_RefPrep

%pTrait_RefLexical>

<!ELEMENT Trait_RefConj - O EMPTY>

<!ATTLIST Trait_RefConj

%pTrait_RefLexical>

<!ELEMENT Trait_RefProRel - O EMPTY>

<!ATTLIST Trait_RefProRel

%pTrait_RefLexical>

<!ELEMENT Trait_RefProIntrog - O EMPTY>

<!ATTLIST Trait_RefProIntrog

%pTrait_RefLexical>

<!--****************** Traits MORPHOLOGIQUES******************-->

<!--************************************************************-->

<!ELEMENT Trait_Mode - O EMPTY>

<!ATTLIST Trait_Mode

id ID #REQUIRED

valeur (%pMode) INDICATIF>

<!ELEMENT Trait_Temps - O EMPTY>

<!ATTLIST Trait_Temps

id ID #REQUIRED

valeur (%pTemps|COMPOSE) PRESENT>

<!-- Le trait de temps permet d'exprimer les restrictions de temps
```

```
liees a certaines tournures.

Ex : etre arrive socialement

V[Lex : arriver]

[Temps : COMPOSE]

[Aux : ETRE] -->

<!ELEMENT Trait_Personne - O EMPTY>

<!ATTLIST Trait_Personne

id ID #REQUIRED

valeur (%pPersonne) 3>

<!ELEMENT Trait_Genre - O EMPTY>

<!ATTLIST Trait_Genre

id ID #REQUIRED

valeur (%pGenre) MASCULIN>

<!ELEMENT Trait_Nombre - O EMPTY>

<!ATTLIST Trait_Nombre

id ID #REQUIRED

valeur (%pNombre) SINGULIER>

<!ELEMENT Trait_NombrePosseur - O EMPTY>

<!ATTLIST Trait_NombrePosseur

id ID #REQUIRED

valeur (%pNombrePosseur) SINGULIER_POSSEUR>

<!--****************** Traits MORPHO-SYNTAXIQUES ***************-->

<!--*************************************************************-->

<!ELEMENT Trait_SsCatMorph - O EMPTY>

<!ATTLIST Trait_SsCatMorph

id ID #REQUIRED
```

```
valeur (%pSsCatGram) COMMUN>
<!-- Les valeurs des traits de sous-categorie morphologique
sont predefinies -->
<!ELEMENT Trait_SsCatSynt - O EMPTY>
<!ATTLIST Trait_SsCatSynt
id ID #REQUIRED
valeur (%pSsCatSynt
|%pSsCatSynt_cust) COORDONNE>
<!-- Les traits de sous-categorie syntaxique peuvent porter sur des
categories terminales ou non terminales et sont definis a facon,
en plus de certaines valeurs deja proposees dans le modele GENELEX.
Ex : un KILO de pommes
N[SsCatSynt : DETERMINATIF]
SN[SsCatSynt : DET_VIDE] -->
<!ELEMENT Trait_Aux - O EMPTY>
<!ATTLIST Trait_Aux
id ID #REQUIRED
valeur (%pAux) AVOIR
temps (SANS_T|%pTemps
|COMPOSE) SANS_T
mode (SANS_M|%pMode) SANS_M>
<!-- Ce trait permet d'associer a un verbe donne (l'entree decrite ou
bien un verbe dans le contexte de l'entree) l'auxiliaire qui
correspond a un emploi et qui doit lui etre associe.
Ex : se lever (etre leve) // lever (avoir leve)
```

V[Lex : lever] V[Lex : lever]

[Aux : ETRE] [Aux : AVOIR]

[Pronominal : SE]

L'apparition du Trait_Aux en conjonction avec un Trait_Temps portant

la valeur COMPOSE indique que l'auxiliaire est necessairement

present dans l'emploi que l'on decrit.

Ex : etre arrive

V[Lex : arriver]

[Aux : ETRE]

[Temps : COMPOSE]

Les attributs temps et mode precisent lorsque c'est necessaire

le temps et le mode de l'auxiliaire lui-meme.

Ex : etant donne

V[Lex : donner]

[Temps : COMPOSE]

[Aux : ETRE[Mode : PARTICIPE][Temps : PRESENT]] -->

<!ELEMENT Trait_Pronominal - O EMPTY>

<!ATTLIST Trait_Pronominal

id ID #REQUIRED

valeur (%pPronominal) SE>

<!-- Ce trait permet d'associer a un verbe donne (l'entree decrite

ou bien un verbe dans le contexte de l'entree) la particule

preverbale NON REFERENTIELLE qui correspond a un emploi et qui doit

lui etre associee (Cf. "vrais" pronominaux).

Ex : s'en aller

V[Lex : aller]

```
[Pronominal : SE_EN] -->
<!ELEMENT Trait_Neg - O EMPTY>
<!ATTLIST Trait_Neg
id ID #REQUIRED
valeur (%pNeg) LIBRE>
<!-- La presence d'un Trait_Neg indique que l'emploi decrit est a la
forme negative ; on peut de plus preciser dans le champ valeur une
restriction sur la lexicalisation de la negation -->
<!ELEMENT Trait_Accord - O EMPTY>
<!ATTLIST Trait_Accord
id ID #REQUIRED
valeur (%pIJKL) I>
<!ELEMENT Trait_Passif - O EMPTY>
<!ATTLIST Trait_Passif
id ID #REQUIRED
valeur (%pBin) PLUS>
<!ELEMENT Trait_Tournure - O EMPTY>
<!ATTLIST Trait_Tournure
id ID #REQUIRED
valeur (%pTournure) INTERROGATIVE>
<!--****************** Traits SYNTAXICO-SEMANTIQUES ***********-->
<!--*********************************************************-->
<!ELEMENT Trait_Coref - O EMPTY>
<!ATTLIST Trait_Coref
id ID #REQUIRED
```

```
valeur (%pIJKL) I>

<!-- La coreference doit toujours etre resoluble : si un trait de

valeur I est present, il existe au moins un autre trait de valeur I

ou NON_I lui repondant.

Les traits Coref ne forcent pas la co-realisation des syntagmes les

portant ; si on souhaite imposer cette co-realisation, on le fera

comme d'habitude par l'usage de Conditions -->

<!--***************** Traits SEMANTIQUES ***********************-->

<!--************************************************************-->

<!ELEMENT Trait_Aspect - O EMPTY>

<!ATTLIST Trait_Aspect

id ID #REQUIRED

valeur (%pAspect) PROCESSIF>

<!ELEMENT Trait_Bin - O EMPTY>

<!ATTLIST Trait_Bin

id ID #REQUIRED

nom CDATA #REQUIRED

valeur (%pBin) PLUS>

<!-- Ce type de trait permet par exemple d'exprimer les

"conditions denotationnelles",

Ex : anime, humain -->

<!ELEMENT Trait_Libre - O EMPTY>

<!ATTLIST Trait_Libre

id ID #REQUIRED

nom CDATA #REQUIRED

valeur CDATA #REQUIRED>
```

```
<!-- Ce type de trait pourra etre exploite pour specifier des classes
ou des familles semantiques
Ex : nom : classe
valeur : vetement -->
<!-- Les Traits Bin et Libre pourront de plus etre utilises pour
d'eventuels autres traits non predefinis dans cette DTD -->
<!--******************** ROLE THEMATIQUE ********************-->
<!--*************************************************************-->
<!ELEMENT RoleTh - O EMPTY>
<!ATTLIST RoleTh
id ID #REQUIRED
valeur (%pRoleTh_cust) AGENT>
<!--************************ FONCTION ***************************-->
<!--*************************************************************-->
<!ELEMENT Fonction - O EMPTY>
<!ATTLIST Fonction
id ID #REQUIRED
valeur (%pFonction_cust) TETE>
<!-- Les entites suffixees par _cust sont definies dans un fichier
"custom.ent" propre a l'utilisateur ; celui-ci peut ainsi ajouter
les valeurs d'attributs dont il souhaite disposer -->
```

7. Contraintes *syntaxe.ctr*

```
<!--Consortium GENELEX @(#) syntaxe.ctr 4.2@(#) 94/06/23 14:22:02 -->
<!--CONTRAINTE Usyn
```

```
combve TYPE CombVE

(description

|description_l) TYPE Description

transfdescription_l TYPE TransfDescription -->

<!--CONTRAINTE Description

self TYPE Self

construction TYPE Construction -->

<!--CONTRAINTE Self

(syntagme_nt_s

|syntagme_nt_s_l) TYPE Syntagme_NT_S

transfsyntagme_l TYPE TransfSyntagme

intervconst TYPE IntervConst

comportappele_l TYPE CompportAppele -->

<!--CONTRAINTE IntervConst

fonction TYPE Fonction

roleth_l TYPE RoleTh

syntagme_t_l TYPE Syntagme_T -->

<!--CONTRAINTE CompportAppele

fonction TYPE Fonction

roleth_l TYPE RoleTh

syntagme_t TYPE Syntagme_T -->

<!--CONTRAINTE Construction

optionnalite TYPE Optionnalite

trait_l TYPE (Trait_Lex|Trait_Introd

|Trait_Prep|Trait_Conj
```

```
|Trait_ProRel

|Trait_ProIntrog

|Trait_Mode|Trait_Temps

|Trait_Personne|Trait_Genre

|Trait_Nombre

|Trait_NombrePosseur

|Trait_SsCatMorph

|Trait_SsCatSynt

|Trait_Aux|Trait_Pronominal

|Trait_Neg|Trait_Accord

|Trait_Passif|Trait_Tournure

|Trait_Coref|Trait_Aspect

|Trait_Bin|Trait_Libre)

position_c_l TYPE Position_C -->

<!--CONTRAINTE Position_C

fonction TYPE Fonction

roleth_l TYPE RoleTh

syntagme_c_l TYPE (Syntagme_T|Syntagme_NT_C)

transfsyntagme_l TYPE TransfSyntagme -->

<!--CONTRAINTE Position_S

fonction TYPE Fonction

roleth_l TYPE RoleTh

syntagme_s_l TYPE (Syntagme_T|Syntagme_NT_S)

transfsyntagme_l TYPE TransfSyntagme -->

<!--CONTRAINTE Insertion

retire_syntagme_c_l TYPE (Syntagme_T|Syntagme_NT_C)
```

```
retire_transfsyntagme_l TYPE TransfSyntagme -->
<!--CONTRAINTE CheminSyntagme
syntagme TYPE (Syntagme_T|Syntagme_NT_C
|Syntagme_NT_S) -->
<!--CONTRAINTE SelectIntervConst
syntagme_t TYPE Syntagme_T -->
<!--CONTRAINTE Syntagme_T
trait_l TYPE (Trait_Lex|Trait_RefLex
|Trait_Mode|Trait_Temps
|Trait_Personne|Trait_Genre
|Trait_Nombre
|Trait_NombrePosseur
|Trait_SsCatMorph
|Trait_SsCatSynt
|Trait_Aux|Trait_Pronominal
|Trait_Neg|Trait_Accord
|Trait_Passif|Trait_Tournure
|Trait_Coref|Trait_Aspect
|Trait_Bin|Trait_Libre) -->
<!--CONTRAINTE Syntagme_NT_C
optionnalite TYPE Optionnalite
trait_l TYPE (Trait_Lex|Trait_Introd
|Trait_Prep|Trait_Conj
|Trait_ProRel
|Trait_ProIntrog
```

```
|Trait_Mode|Trait_Temps

|Trait_Personne|Trait_Genre

|Trait_Nombre

|Trait_NombrePosseur

|Trait_SsCatMorph

|Trait_SsCatSynt

|Trait_Aux|Trait_Pronominal

|Trait_Neg|Trait_Accord

|Trait_Passif|Trait_Tournure

|Trait_Coref|Trait_Aspect

|Trait_Bin|Trait_Libre)

position_c_l TYPE Position_C -->

<!--CONTRAINTE Syntagme_NT_S

optionnalite TYPE Optionnalite

trait_l TYPE (Trait_Lex|Trait_Introd

|Trait_Prep|Trait_Conj

|Trait_ProRel

|Trait_ProIntrog

|Trait_RefLex

|Trait_RefIntrod

|Trait_RefPrep|Trait_RefConj

|Trait_RefProRel

|Trait_RefProIntrog

|Trait_Mode|Trait_Temps

|Trait_Personne|Trait_Genre

|Trait_Nombre
```

```
|Trait_NombrePosseur
|Trait_SsCatMorph
|Trait_SsCatSynt
|Trait_Aux|Trait_Pronominal
|Trait_Neg|Trait_Accord
|Trait_Passif|Trait_Tournure
|Trait_Coref|Trait_Aspect
|Trait_Bin|Trait_Libre)
position_s_l TYPE Position_S
insertion_l TYPE Insertion -->
<!--CONTRAINTE R_ComposeUm
um TYPE (Um_S|Um_Agg|Um_C) -->
<!--CONTRAINTE R_ComposeUsyn
usyn TYPE Usyn
mdc TYPE MdC -->
<!--CONTRAINTE HeritePosition
modifposition TYPE ModifPosition -->
<!--CONTRAINTE FiltreSelf
modifintervconst TYPE ModifIntervConst
modifsyntagme_nt TYPE ModifSyntagme_NT -->
<!--CONTRAINTE TransfUsyn
usyn_resultat TYPE Usyn -->
<!--CONTRAINTE TransfDescription
(description_origine
|description_resultat) TYPE Description -->
```

```
<!--CONTRAINTE ModifDescription
modifconstruction TYPE ModifConstruction
modifintervconst TYPE ModifIntervConst
modifsyntagme_nt TYPE ModifSyntagme_NT -->
<!--CONTRAINTE ModifConstruction
optionnalite TYPE Optionnalite
(retire_trait_l
|ajoute_trait_l) TYPE (Trait_Lex|Trait_Introd
|Trait_Prep|Trait_Conj
|Trait_ProRel
|Trait_ProIntrog
|Trait_Mode|Trait_Temps
|Trait_Personne|Trait_Genre
|Trait_Nombre
|Trait_NombrePosseur
|Trait_SsCatMorph
|Trait_SsCatSynt
|Trait_Aux|Trait_Pronominal
|Trait_Neg|Trait_Accord
|Trait_Passif|Trait_Tournure
|Trait_Coref|Trait_Aspect
|Trait_Bin|Trait_Libre) -->
<!--CONTRAINTE TransfPosition
modifposition TYPE ModifPosition -->
<!--CONTRAINTE ModifPosition
fonction TYPE Fonction
```

```
roleth_l TYPE RoleTh
(retire_syntagme_l
|ajoute_syntagme_l) TYPE (Syntagme_T|Syntagme_NT_C
|Syntagme_NT_S)
(retire_transfsyntagme_l
|ajoute_transfsyntagme_l
|transfsyntagme_l) TYPE TransfSyntagme -->
<!--CONTRAINTE TransfSyntagme
(syntagme_origine
|syntagme_resultat) TYPE (Syntagme_T|Syntagme_NT_C
|Syntagme_NT_S)
modifsyntagme TYPE (ModifSyntagme_T
|ModifSyntagme_NT) -->
<!--CONTRAINTE ModifSyntagme_T
(retire_trait_l
|ajoute_trait_l TYPE (Trait_Lex|Trait_RefLex
|Trait_Mode|Trait_Temps
|Trait_Personne|Trait_Genre
|Trait_Nombre
|Trait_NombrePosseur
|Trait_SsCatMorph
|Trait_SsCatSynt
|Trait_Aux|Trait_Pronominal
|Trait_Neg|Trait_Accord
|Trait_Passif|Trait_Tournure
```

```
|Trait_Coref|Trait_Aspect

|Trait_Bin|Trait_Libre) -->

<!--CONTRAINTE ModifSyntagme_NT

(retire_trait_l

|ajoute_trait_l) TYPE (Trait_Lex|Trait_Introd

|Trait_Prep|Trait_Conj

|Trait_ProRel

|Trait_ProIntrog

|Trait_RefLex

|Trait_RefIntrod

|Trait_RefPrep|Trait_RefConj

|Trait_RefProRel

|Trait_RefProIntrog

|Trait_Mode|Trait_Temps

|Trait_Personne|Trait_Genre

|Trait_Nombre

|Trait_NombrePosseur

|Trait_SsCatMorph

|Trait_SsCatSynt

|Trait_Aux|Trait_Pronominal

|Trait_Neg|Trait_Accord

|Trait_Passif|Trait_Tournure

|Trait_Coref|Trait_Aspect

|Trait_Bin|Trait_Libre)

optionnalite TYPE Optionnalite -->

<!--CONTRAINTE ModifIntervConst
```

```
fonction TYPE Fonction

roleth_l TYPE RoleTh

(retire_syntagme_t_l

|ajoute_syntagme_t_l) TYPE Syntagme_T

transfsyntagme_l TYPE TransfSyntagme -->

<!--CONTRAINTE (Trait_Lex|Trait_Introd|Trait_Prep|Trait_Conj

|Trait_ProRel|Trait_ProIntrog)

um TYPE (Um_S|Um_Agg|Um_C) -->
```

8. Entités *syntaxe.ent*

```
<!--Consortium GENELEX @(#) syntaxe.ent 4.1@(#) 94/06/23 14:19:20 -->

<!-- **************A L'ADRESSE DES UTILISATEURS ******************

Vos remarques concernant la DTD seront etudiees par le consortium

GENELEX. Celui-ci assurera la diffusion de la nouvelle version qui

pourrait en decouler.

********************************************************************* -->

<!ENTITY % pEtiquetteSynt_T

"%pCatGram|e">

<!ENTITY % pEtiquetteSynt_NT

"P|Nbarre|SN|SV|SADJ|SADV|SP">

<!ENTITY % pSsCatSynt "RELATIVE|COMPLETIVE

|COORDONNE|SUBORDONNEE

|INTERROGATIVE_DRI|INTERROGATIVE_DRD

|COMPARATIF|SUPERLATIF

|TEMPS|LIEU|MANIERE|DEGRE|QUANTITE
```

```
|COPULE|DET_VIDE|DETERMINATIF">

<!ENTITY % pIJKL "I|J|K|L|NON_I|NON_J|NON_K|NON_L">

<!ENTITY % pAux "ETRE|AVOIR">

<!ENTITY % pPronominal "SE|LE|LA|LES|Y|EN

|SE_LE|SE_LA|SE_LES|SE_Y|SE_EN">

<!ENTITY % pNeg "LIBRE|NE_PAS|NE_PLUS|NE_JAMAIS|NE

|NE_GUERE|NE_POINT|NE_MAIS

|NE_QUE|NE_PAS_QUE|NE_PLUS_QUE

|NE_JAMAIS_QUE|NE_GUERE_QUE|NE_RIEN_QUE">

<!ENTITY % pAspect "PROCESSIF|RESULTATIF|STATIF">

<!ENTITY % pBin "PLUS|MOINS">

<!ENTITY % pTournure "INTERROGATIVE|EXCLAMATIVE">

<!ENTITY % pPortee "EXTERNE|INTERNE|INTERVENANT">

<!ENTITY % pTypeComposant

"APPELANT|APPELANT_APPELE|APPELE">

<!ENTITY % pDestination "EXTERIEUR|INTERIEUR">

<!ENTITY % pTypeListPos "OUVERTE|FERMEE">
```

9. DTD *morpho.dtd*

```
<!--Consortium GENELEX @(#) morpho.dtd 3.2@(#) 94/09/07 14:35:28 -->

<!-- **************A L'ADRESSE DES UTILISATEURS ******************

Vos remarques concernant la DTD seront etudiees par le consortium

GENELEX. Celui-ci assurera la diffusion de la nouvelle version qui
```

```
pourrait en decouler.
********************************************************** -->
<!ELEMENT GenelexMorpho - O (
(Um_S|Um_C|Um_Agg|Um_Aff)* &
Etymon* &
Mfg* &
Mfp* &
CombTM* &
Mfc* &
Comb_Comb*)>
<!-- ******************************************************* -->
<!-- ******* DEFINITION DES UNITES MORPHOLOGIQUES ************ -->
<!-- ******************************************************* -->
<!ENTITY % pUmAtt
"id ID #REQUIRED
appellation CDATA #IMPLIED
attestation CDATA #IMPLIED
combve IDREF #IMPLIED
etymon_l IDREFS #IMPLIED">
<!ELEMENT Um_S - O ((Umg|Ump)+ & Derivation* & FormeBreve*)>
<!ATTLIST Um_S
%pUmAtt
catgram (SANS_C|%pCatGram) SANS_C
sscatgram (SANS_SC|%pSsCatGram) SANS_SC
autonomie (SANS_B|%pBooleen) SANS_B
```

```
usyn_l IDREFS #IMPLIED>
<!-- Le contenu (Umg|Ump)+ oblige une Unite Morphologique Simple a
avoir au moins soit une Unite Graphique soit une Unite Phonique.
Le contenu Derivation permet d'indiquer les eventuelles derivations
aboutissant a l'Um_S.
Le contenu FormeBreve permet de mettre l Um_S consideree en relation
avec d`autres Unites qui en sont une forme breve.
usyn_l pointe vers les differentes Usyn qui decrivent le(s)
comportement(s) syntaxique(s) de l’Um -->
<!ELEMENT Um_C - O (R_Compose+ & FormeBreve*)>
<!ATTLIST Um_C
%pUmAtt
catgram (SANS_C|%pCatGram) SANS_C
sscatgram (SANS_SC|%pSsCatGram) SANS_SC
usyn_l IDREFS #IMPLIED>
<!-- Une Unite Morphologique Composee n'a pas d'Umg ni d'Ump : ses
formes graphiques et phonemiques sont deduites de celles des unites
qui la composent. Chaque Composant est indique par une relation
R_Compose.
usyn_l pointe vers les differentes Usyn qui decrivent le(s)
comportement(s) syntaxique(s) de l’Um -->
<!ELEMENT Um_Agg - O ((Umg|Ump)+ & R_Compose+)>
<!ATTLIST Um_Agg
%pUmAtt
obligatoire (SANS_B|%pBooleen) SANS_B>
<!-- Une Unite Morphologique Agglutinee utilise la relation R_Compose
```

```
pour indiquer ses elements incorpores - "agglutinants" -.
L'attribut obligatoire indique le caractere obligatoire ou
facultatif de l'emploi de l'agglutine vs. l'emploi de la forme
developpee correspondante. -->
<!ELEMENT Um_Aff - O ((Umg|Ump)+ & CatGram_Select* & CatGram_Result*
& Genre_Result*)>
<!ATTLIST Um_Aff
%pUmAtt
typaff (SANS_T|%pTypaff) SANS_T
usem_aff_l IDREFS #IMPLIED>
<!-- L'Attribut typaff permet de noter le type d'une Unite
Morphologique Affixe ; dans le cas d'un affixe ne prenant son type
qu'en contexte de derivation, cet attribut aura la valeur SANS_T.
Les Contenus CatGram_Select/Result et Genre_Result permettent
d'indiquer pour une Unite Morphologique Affixe d'eventuelles
restrictions sur la categorie
grammaticale et le genre des Unites derivees resultantes.
usem_aff_l pointe vers les differentes Usem_Aff qui decrivent le(s)
sens de l'Um -->
<!ELEMENT CatGram_Result - O EMPTY>
<!ATTLIST CatGram_Result
catgram (SANS_C|%pCatGram) SANS_C>
<!ELEMENT CatGram_Select - O EMPTY>
<!ATTLIST CatGram_Select
catgram (SANS_C|%pCatGram) SANS_C>
```

```
<!ELEMENT Genre_Result - O EMPTY>

<!ATTLIST Genre_Result

genre (SANS_G|%pGenre) SANS_G>

<!-- ******************************************************** -->

<!-- ******* FORME GRAPHIQUE / FORME PHONIQUE ********** -->

<!-- ******************************************************** -->

<!ENTITY % pUmgpAtt

"nieme NUMBER #IMPLIED

vedette (SANS_B|%pBooleen) SANS_B

appellation CDATA #IMPLIED

attestation CDATA #IMPLIED

combve IDREF #IMPLIED

mf IDREF #IMPLIED

corresp_l NUMBERS #IMPLIED">

<!ELEMENT Umg - O (Lib & Radg*)>

<!ELEMENT Ump - O (Lib & Radp*)>

<!ATTLIST (Umg|Ump)

%pUmgpAtt>

<!-- Dans le cas d'une Unite possedant des variantes Graphiques et/ou

Phoniques, soit plusieurs Umg et/ou Ump, ces Umg et Ump porteront un

nieme identifiant leur numero de variante. La correspondance entre

Umg et Ump est alors faite a l'aide de la liste d'entiers corresp_l.

Si on souhaite de plus identifier une vedette parmi ces variantes,

on la notera par l'attribut vedette.

Le champ mf note le mode flexionnel : ne pas renseigner le champ

signifie qu'on ne connait pas le mode flexionnel de l'Umg/p
```

```
consideree. Dans le cas d'Unites ne se flechissant pas
(prepositions), on affectera un mf vide. -->
<!ENTITY % pRadgpAtt
"nieme NUMBER #IMPLIED
contexte_var CDATA #IMPLIED">
<!ELEMENT (Radg|Radp) - O (Lib)>
<!ATTLIST (Radg|Radp)
%pRadgpAtt>
<!-- le radical a deux usages :
- il est utilise dans le calcul des formes flechies par le
Mfg/p. Le radical egal au libelle de l umg/p n est pas
enregistre comme element radical mais seulement comme
libelle ; on pourra cependant y faire reference en tant
que radical nieme 0
- il est utilise dans la derivation -->
<!-- ********************************************************* -->
<!-- ******* ETYMOLOGIE ********** -->
<!-- ********************************************************* -->
<!ELEMENT Etymon - O (Lib?)>
<!ATTLIST Etymon
id ID #REQUIRED
langue CDATA #IMPLIED
sens CDATA #IMPLIED
date CDATA #IMPLIED
appellation CDATA #IMPLIED>
```

```
<!-- ********************************************************** -->

<!-- ******* MODE DE FLEXION GRAPHIQUE ET PHONIQUE ********** -->

<!-- ********************************************************** -->

<!ENTITY % pMfAtt

"id ID #REQUIRED

%pGlose">

<!ELEMENT (Mfg|Mfp) - O (CombTM_Cff+)>

<!ATTLIST (Mfg|Mfp)

%pMfAtt>

<!ELEMENT CombTM_Cff - O (Cff+)>

<!ATTLIST CombTM_Cff

combtm IDREF #REQUIRED>

<!ELEMENT Cff - O (Retrait,Ajout)>

<!ATTLIST Cff

nieme NUMBER #IMPLIED

nieme_radgp NUMBER 0

contexte_var CDATA #IMPLIED

corresp_l NUMBERS #IMPLIED>

<!-- Les attributs nieme et corresp_l permettent de mettre en rapport

d'eventuelles variantes de calcul de formes flechies.

Ex : calcul de je peux/je puis

L'attribut nieme_radgp indique le Nieme radical a selectionner pour

former la forme flechie. La valeur 0 signifie qu on fait reference

au libelle Lib -->

<!ELEMENT (Lib|Ajout|Retrait) O O (#PCDATA)>

<!-- combinaison de traits morphologiques -->
```

```
<!ELEMENT CombTM - O EMPTY>
<!ATTLIST CombTM
id ID #REQUIRED
mode (SANS_M|%pMode) SANS_M
temps (SANS_T|%pTemps) SANS_T
personne (SANS_P|%pPersonne) SANS_P
genre (SANS_G|%pGenre) SANS_G
nombre (SANS_N|%pNombre) SANS_N
nombreposseur (SANS_NP|
%pNombrePosseur) SANS_NP>
<!-- ********************************************************* -->
<!-- ******* DERIVATION MORPHOLOGIQUE ********** -->
<!-- ********************************************************* -->
<!ELEMENT Derivation - O (RestrictUm* & R_Derive+)>
<!ATTLIST Derivation
appellation CDATA #IMPLIED
commentaire CDATA #IMPLIED>
<!-- La liste contenue de R_Derive permet d'indiquer les differents
composants d'une derivation.
L'enregistrement de derivations concurrentes se fait en notant
plusieurs elements Derivation sur l'unite derivee.
RestrictUm s'applique ici au derive. -->
<!ELEMENT R_Derive - O (RestrictUm*)>
<!ATTLIST R_Derive
ordre_lineaire NUMBER #IMPLIED
```

```
statut (SANS_S|%pStatut) SANS_S

retraitg CDATA #IMPLIED

retraitp CDATA #IMPLIED

um IDREF #REQUIRED>

<!-- Le champ um designe le composant de derivation.

RestrictUm s'applique ici au composant de derivation. -->

<!ELEMENT RestrictUm - O EMPTY>

<!ATTLIST RestrictUm

nieme_umg NUMBER #IMPLIED

nieme_radg NUMBER #IMPLIED

nieme_ump NUMBER #IMPLIED

nieme_radp NUMBER #IMPLIED>

<!-- Dans le contexte d'une Unite Morphologique, cet element exprime

une restriction sur cette unite en permettant d'en selectionner une

variante (ou un radical) graphique et/ou phonemique -->

<!-- ******************************************************* -->

<!-- ******* FORME BREVE ********** -->

<!-- ******************************************************* -->

<!ELEMENT FormeBreve - O EMPTY>

<!ATTLIST FormeBreve

typebref (SANS_T|%pTypeBref) SANS_T

um IDREF #REQUIRED>

<!-- L'attribut um designe l'unite que l'on souhaite noter comme forme

breve de l'unite portant la relation Forme_Breve. -->

<!-- ******************************************************* -->

<!-- ******* COMPOSITION MORPHOLOGIQUE ********** -->
```

```
<!-- ********************************************************** -->
<!ELEMENT R_Compose - O (RestrictUm*)>
<!ATTLIST R_Compose
ordre_lineaire NUMBER #IMPLIED
separg (ATTAQUE_G|%pSeparg) ATTAQUE_G
separp (ATTAQUE_P|%pSeparp) ATTAQUE_P
um IDREF #REQUIRED
mfc IDREF #IMPLIED>
<!-- L'attribut um designe l'Um_S/Aff/Agg/C composante.
L'attribut ordre_lineraire indique la place du composant dans le
compose. Les attributs separg/p donnent la liste des separateurs
possibles devant le composant. -->
<!-- modes de flexion des composes -->
<!ELEMENT Mfc - O EMPTY>
<!ATTLIST Mfc
id ID #REQUIRED
%pGlose
comb_comb_l IDREFS #REQUIRED>
<!ELEMENT Comb_Comb - O EMPTY>
<!ATTLIST Comb_Comb
id ID #REQUIRED
contexte_var CDATA #IMPLIED
combcpose IDREF #REQUIRED
combcposant_l IDREFS #REQUIRED>
<!-- L'element Comb_Comb met en rapport une combinaison de traits
```

```
flexionnels du Compose avec une - eventuellement plusieurs en cas de
compose admettant des variantes de flexion - combinaison de traits
flexionnels du Composant. L'attribut contexte_var etiquette les
variantes de flexion des composes.
Ex : des pare-soleil(s)
Le pluriel du compose est forme a partir du singulier
(ancienne orthographe) ou du pluriel du composant soleil
(nouvelle orthographe).
Ces informations "ancienne orthographe" et "nouvelle
orthographe" seront notees dans l'attribut contexte_var.
On prevoira donc un separateur entre les zones de ce CDATA,
l'ordre des zones devant correspondre a l'ordre des IDREFS
dans combcposant_l
-> "ancienne orthographe | nouvelle orthographe" -->
<!-- ******************************************************* -->
<!-- ******* MECANISMES SIMPLIFICATEURS ********** -->
<!-- ******************************************************* -->
<!-- Les SHORTREF suivants permettent l'omission des balises ouvrantes
des elements ajout et retrait ; ces elements peuvent alors
apparaitre dans le fichier balise sous la forme de deux chaines de
caracteres separees par une virgule -->
<!ENTITY e-s-ajout "<ajout>" >
<!SHORTREF s-ajout
"," e-s-ajout >
<!USEMAP s-ajout retrait >
<!-- Dans le cas de figure : <cff>,s</> ou l'element retrait ne
```

```
contient aucun PCDATA, le USEMAP introduira <retrait><ajout> -->

<!ENTITY e-s-retrait "<retrait><ajout>" >

<!SHORTREF s-retrait

"," e-s-retrait >

<!USEMAP s-retrait Cff >
```

10. Contraintes *morpho.ctr*

```
<!--Consortium GENELEX @(#) morpho.ctr 3.2@(#) 94/06/23 14:12:39 -->

<!--CONTRAINTE Um_S

combve TYPE CombVE

etymon_l TYPE Etymon

usyn_l TYPE Usyn -->

<!--CONTRAINTE Um_C

combve TYPE CombVE

etymon_l TYPE Etymon

usyn_l TYPE Usyn -->

<!--CONTRAINTE Um_Agg

combve TYPE CombVE

etymon_l TYPE Etymon -->

<!--CONTRAINTE Um_Aff
```

```
combve TYPE CombVE

etymon_l TYPE Etymon

usem_aff_l TYPE Usem_Aff -->

<!--CONTRAINTE Umg

combve TYPE CombVE

mf TYPE Mfg -->

<!--CONTRAINTE Ump

combve TYPE CombVE

mf TYPE Mfp -->

<!--CONTRAINTE CombTM_Cff

combtm TYPE CombTM -->

<!--CONTRAINTE R_Derive

um TYPE (Um_S|Um_Agg

|Um_Aff) -->

<!--CONTRAINTE FormeBreve

um TYPE (Um_S|Um_C

|Um_Agg|Um_Aff) -->

<!--CONTRAINTE R_Compose

um TYPE (Um_S|Um_C

|Um_Agg|Um_Aff)

mfc TYPE Mfc -->

<!--CONTRAINTE Mfc

comb_comb_l TYPE Comb_Comb -->

<!--CONTRAINTE Comb_Comb

combcpose TYPE CombTM
```

```
combcposant_l TYPE CombTM -->
```

11. Entités *morpho.ent*

```
<!--Consortium GENELEX @(#) morpho.ent 3.1@(#) 94/06/23 14:23:22 -->

<!-- **************A L'ADRESSE DES UTILISATEURS ******************

Vos remarques concernant la DTD seront etudiees par le consortium

GENELEX. Celui-ci assurera la diffusion de la nouvelle version qui

pourrait en decouler.

********************************************************** -->

<!ENTITY % pBooleen "OUI|NON" >

<!ENTITY % pDatation "ARCHAIQUE|VIEILLI|MODERNE" >

<!ENTITY % pNiveauLgue "FAMILIER|VULGAIRE|ARGOTIQUE|POPULAIRE

|LITTERAIRE|SAVANT|STANDARD" >

<!ENTITY % pFrequence "RARE|COURANT" >

<!ENTITY % pCatGram "NOM|ADJECTIF|ADVERBE|VERBE|PREPOSITION

|CONJONCTION|INTERJECTION|DETERMINANT|PRONOM

|PARTICULE" >

<!ENTITY % pSsCatGram "PROPRE|COMMUN|POSSESSIF|DEMONSTRATIF

|PARTITIF|DEFINI|INDEFINI|CARDINAL|ORDINAL

|EXCLAMATIF|QUALIFICATIF|INTERROGATIF

|RELATIF|COMPLETIF|COORDINATION|SUBORDINATION

|PERSONNEL_FORT|PERSONNEL_FAIBLE|IMPERSONNEL

|COMPARATIF_EGALITE|COMPARATIF_SUPERIORITE

|COMPARATIF_INFERIORITE

|SUPERLATIF_SUPERIORITE|SUPERLATIF_INFERIORITE
```

```
|SUPERLATIF_ABSOLU">

<!ENTITY % pMode "INDICATIF|SUBJONCTIF|CONDITIONNEL|IMPERATIF

|INFINITIF|PARTICIPE" >

<!ENTITY % pTemps "PRESENT|IMPARFAIT|PASSE_SIMPLE|FUTUR|PASSE" >

<!ENTITY % pPersonne "1|2|3" >

<!ENTITY % pGenre "MASCULIN|FEMININ|NEUTRE" >

<!ENTITY % pNombre "SINGULIER|PLURIEL" >

<!ENTITY % pNombrePosseur

"SINGULIER_POSSEUR|PLURIEL_POSSEUR" >

<!ENTITY % pTypaff "PREFIXE|SUFFIXE|INFIXE" >

<!ENTITY % pStatut "%pTypaff|BASE" >

<!ENTITY % pTypeBref "ABREVIATION|SIGLE|ACRONYME" >

<!ENTITY % pSeparg "TIRET|APOSTROPHE|ESPACE|JOINTURE

|TIRET_ESPACE|TIRET_JOINTURE

|TIRET_ESPACE_JOINTURE

|APOSTROPHE_JOINTURE" >

<!ENTITY % pSeparp "LIAISON_t|LIAISON_z|LIAISON_k

|LIAISON_n|LIAISON_r|FRONTIERE_MOT" >
```