UNE MODÉLISATION PAR OBJETS DE PRINCIPES LINGUISTIQUES

Cathy Berthouzoz[1]
LATL/Université de Genève
Rue de Candolle 2/ 1203 Genève/ CH
berthouzoz@latl.unige.ch

ABSTRACT

In order to explain language acquisition despite of poverty of stimulus, Chomsky has postulated Universal Grammar (UG). The Government and Binding Theory (GB) describes UG as a system of universal principles and parameters. The grammar of any language may be derived from UG by assigning the proper value to these parameters. UG is thus interesting from a theoretical point of view, but is is even more appealing from a practical point of view, especially in the context of machine translation systems. Such systems require one grammar per processed language. By use of UG the development of the system can be much simplified: each grammar is an instance of UG with the proper value set to the parameters, the set of specific rules being constrained. We present in this paper a modeling of UG according to the linguistic objects to which the principles of the grammar apply. These objects include grammatical categories and syntactic chains. The key notion of inheritance constitutes a natural way of modeling UG in view of its linguistic processing. We will show that this model reconciliate linguistic modularity – principles are organised in specialized subsystems – with computational modularity – specialization, information hiding and minimal interface.

KEY WORDS government and binding theory, universal grammar, principles and parameters theory, object paradigm, grammar modeling.

1. INTRODUCTION

Pour expliquer l'acquisition du langage malgré la pauvreté du stimulus, Chomsky a postulé l'existence de la grammaire universelle, GU dans la suite du texte (Chomsky, 1981), et surtout son caractère inné. La GU, décrite dans la théorie du Gouvernement et Liage (*Government and Binding Theory*, voir Chomsky, 1981, 1986a, 1986b; Rizzi, 1986, 1990; Haegeman, 1994 entre autres), est un système de principes universels et de paramètres dont la valeur permet de dériver la grammaire de chaque langue naturelle. La GU se veut être un modèle de la compétence linguistique de l'être humain.

La GU peut être utilisée en Traitement Automatique du Langage Naturel (TALN) comme alternative aux grammaires indépendantes du contexte (Door, 1990; Fong, 1991; Crocker, 1996; Merlo, 1996; Wehrli, 1997 entre autres). L'utilisation de la GU peut aussi simplifier le développement d'un système de traduction multilingue. En effet, un tel système utilise a priori une grammaire par langue traitée. Avec la GU comme base grammaticale, le système de règles spécifiques demeure relativement contraint, chaque grammaire étant une instance de la GU pour laquelle la valeur des paramètres est fixée.

A la suite de Fodor (1983) qui a postulé la modularité des fonctions cognitives, les linguistes se sont posé la question de la modularité des grammaires de type Principes et Paramètres (Berwick et al., 1991; Walther, 1994; Merlo, 1996 entre autres). La modularité du point de vue linguistique signifie que les principes sont organisés en sous-systèmes spécialisés, tels que par exemple le module thématique, qui règle l'assignation des rôles thématiques aux syntagmes nominaux et phrastiques, et le module du cas, qui gère l'assignation des cas aux syntagmes nominaux. Mais elle ne rejoint pas complètement la modularité du point de vue informatique, qui ajoute à la spécialisation des modules, dissimulation de l'information et interface minimale entre les modules. Or, les modules de la GU interagissent fortement entre eux, et n'ont pas de structures de données propres. Un moyen de faire concorder ces deux notions de modularité est de définir la modularité non pas selon les principes eux-mêmes, mais selon les objets sur lesquels s'appliquent les principes.

Nous présentons dans ce papier une modélisation de la GU selon les objets spécifiques sur lesquels s'appliquent les principes, à savoir les catégories grammaticales et les chaînes syntaxiques. L'utilisation du concept central d'héritage , qui est à l'origine de tout un courant de l'informatique – le paradigme des objets – constitue un moyen naturel de modéliser la GU en vue d'un traitement informatique. Nous montrerons que cette modélisation réconcilie la modularité linguistique avec la modularité informatique. Nous rejoignons avec cette approche la conclusion à laquelle est arrivée Merlo (1995), à savoir : « La modularité, si elle existe, doit être cherchée dans le contenu linguistique, et non pas dans l'organisation de la théorie. »

[1] Je remercie chaleureusement Christopher Laenzlinger et Thierry Etchegoyhen pour leur relecture attentive, leurs commentaires instructifs et leurs discussions productives.

La suite de l'article est structurée de la façon suivante : la section 2 décrit le paradigme des objets, la section 3 la GU dans le paradigme des objets et la section 4 les objets principaux de la GU.

2. LE PARADIGME DES OBJETS

Le paradigme des objets a émergé en réponse à la crise du logiciel du début des années 80, caractérisée par les problèmes de réalisation, de maintenance, voire même de conception des larges systèmes informatiques. Il se caractérise par la modularité, l'extensibilité et la réutilisabilité des composants informatiques.

Dans ce paradigme, l'état du système est distribué entre les objets et chaque objet gère son propre état (ses attributs), qu'il dissimule aux autres. Ceux-ci n'ont qu'une vue abstraite de l'objet. Ils ne peuvent accéder aux données que par les opérations mises à disposition par l'objet dans son interface. On parle aussi d'encapsulation des données. Chaque objet est une entité indépendante, dont la représentation interne (l'état) peut être modifiée sans faire référence aux autres objets, ce qui en fait une brique de base réutilisable selon les besoins. Des classes (types d'objets) regroupent les objets qui possèdent les mêmes caractéristiques. Une classe peut être étendue par de nouveaux attributs et de nouvelles opérations, les attributs et opérations de la classe de base, appelée super-classe, étant hérités par la classe dérivée, ou sous-classe. Ce mécanisme d'héritage est la propriété fondamentale de l'orientation-objets. Les objets communiquent entre eux à l'aide de messages (opérations, responsabilités) plutôt qu'à l'aide de variables partagées, ce qui garantit une plus grande sécurité du système. Un objet répond à un message à l'aide de la méthode – spécification de l'opération – appropriée, qu'il propose dans son interface. Le polymorphisme – plusieurs objets ont des méthodes qui protent le même nom, mais n'ont pas forcement la même fonctionnalité – permet de faire réagir différemment des objets en réponse au même message.

Le modèle par objets est abstrait du monde réel, formant ainsi une base plus solide pour la suite du développement que celui de l'approche traditionnelle, où il est en fait un point de vue du concepteur du système. Plusieurs méthodes de conception par objets ont été développées, entre autres la méthode de Booch (Booch, 1991), OMT (Rumbaugh et al., 1991), OOSE (Jacobson et al., 1992), et UML[2] (UML, 1997), qui est un condensé unifié des méthodes précédentes.

Bien que le concept de modularité soit à la base des objets – terme englobant encapsulation des données et des opérations, interactions minimales, spécialisation, réutilisabilité – les propriétés spécifiques aux objets – classes, héritage, messages, polymorphisme – leur donnent une plus grande puissance que les modules.

Le paradigme des objets s'impose dans tous les domaines de l'informatique, y compris en linguistique informatique. De nombreux chercheurs ont évoqué les avantages que pourraient présenter des approches par objets dans le développement d'applications de traitement du langage naturel : voir par exemple Zampolli (1991) pour la réutilisabilité des bases de connaissances linguistiques, Zajac (1992) pour l'application des techniques du génie logiciel au traitement du langage naturel, en particulier le développement incrémental des grammaires, et Hahn et al. (1997) pour un exemple de système de TALN orienté-objets. La notion d'héritage est aussi largement répandue en TAL (par exemple Daelemans & Gazdar,1990; Reinhard & Gibbon (1991) et Evans *et al.* (1995)).

3. LA GU DANS LE PARADIGME DES OBJETS

La GU est définie comme un système de principes universels, qui ne varient pas d'une langue à l'autre, et de paramètres qui tiennent compte des propriétés spécifiques de chaque langue. Les valeurs des paramètres sont proposées par la GU parmi un choix restreint. Par exemple, les composants syntaxiques des phrases, *sujet*, *verbe*, *objet*,...sont communs à toutes les langues, par contre l'ordre de base de ces composants dépend de la langue. La grammaire d'un langage donné sera donc une instantiation de la GU avec une certaine valeur des paramètres.

Les principes de la GU s'organisent en sous-systèmes spécialisés que l'on nomme modules. Chaque sous-système s'occupe d'un processus ou d'un groupe de phénomènes particuliers, et comprennent la théorie Xbarre, qui définit la structure hiérarchique en constituants de la phrase, la théorie du Gouvernement, qui règle les relations structurales entre les constituants, la théorie Thêta, qui règle l'assignation des rôles thématiques aux arguments, la théorie du Cas, qui règle la distribution des syntagmes nominaux dans la phrase et la théorie du Liage, leur interprétation (co-)référentielle. L'application de certains principes nécessite que des éléments ne se trouvent pas dans leur position d'origine. La théorie des Chaînes gère la constitution des chaînes entre les éléments déplacés et leurs traces laissées dans leur position canonique. La GU est encore contrainte par un principe général, le Principe de Projection Etendu, qui fait le lien entre le lexique et le composant syntaxique, et postule l'existence d'un sujet dans toute phrase bien formée. Cela implique l'existence de la catégorie vide PRO, comme sujet des infinitives notamment, sur laquelle s'applique la théorie du Contrôle.

Chaque sous-théorie, même si elle regroupe des fonctionnalités spécifiques et reliées entre elles, n'a pas de structures de données propres, ni une interface minimale avec les autres sous-théories. Sur le plan informatique, cela pose un problème pour une approche modulaire dans laquelle chaque sous-système correspond à un module, car cela suppose le partage de nombreuses variables complexes entre les modules. Par contre, la modularisation de la GU selon les objets sur lesquels s'appliquent les principes est possible grâce à l'orientation-objets: le

[2] © 1997 by Rational Software Corporation.

concept d'héritage permet de définir les objets par subsomption, du plus général à l'intérieur, vers le plus particulier à l'extérieur, le polymorphisme permet à un même message d'être traité différemment selon l'objet qui le reçoit. Un parallèle peut être tiré entre les concepts du paradigme des objets et ceux de la GU, comme illustré ci-dessous.

Classes - catégories grammaticales et syntaxiques D'après la définition, une classe regroupe un ensemble d'objets qui possèdent les mêmes caractéristiques. On a donc une correspondance directe entre les classes et les catégories grammaticales, noms, verbes, adjectifs, etc., et syntaxiques, syntagmes nominaux, adjectivaux, verbaux, etc... De même que chaque mot d'une phrase fait partie d'une catégorie grammaticale, il générera un objet de la classe correspondante – plusieurs dans les cas d'ambiguité – avec les attributs et opérations caractéristiques de sa classe.

Héritage - universalité La notion d'héritage permet de dériver, ou étendre, des classes à partir de classes existantes, les classes dérivées héritant des attributs et méthodes de la classe de base. L'héritage est utilisé sur deux niveaux. Tout d'abord à l'intérieur même de la GU, pour définir une hiérarchie de classes compatibles entre elles. Les classes de base définissent les opérations applicables à toutes les classes de la hiérarchie, les classes dérivées implémentant les opérations spécifiques de la sous-classe. Par exemple, le schéma Xbarre s'applique à toutes les catégories syntaxiques, il définit donc une opération de la classe de base. Par contre l'assignation de rôle thématique concerne quelques catégories seulement, en l'occurrence les syntagmes nominaux et phrastiques; elle définira donc une opération des classes concernées.

L'héritage est aussi utilisé au niveau inter-linguistique : il permet de définir des classes de base abstraites correspondant aux objets de la GU, et de les étendre dans des classes plus spécifiques correspondant aux grammaires particulières, qui fixent la valeur des paramètres et implémentent les règles spécifiques de chaque langue. L'héritage garantissant la compatibilité entre les objets de la classe de base et ceux de ses sous-classes, les grammaires particulières sont compatibles avec la GU.

Messages - principes Les messages sont le seul moyen de communication entre les objets. Le message envoyé dit ce qu'il faut faire, l'objet détermine *comment* il faut le faire, c'est-à-dire quelle méthode invoquer, d'après son type dynamique. On peut voir une correspondance quasi directe entre les messages et les principes de la GU. En effet, les principes sont des *contraintes* de bonne formation sur la structure de la phrase, ils disent quelles conditions les 'mots' doivent remplir pour former une phrase correcte, selon leur catégorie grammaticale ou syntaxique et selon la langue pour les principes sur lesquels s'appliquent les paramètres. Par exemple, la théorie Xbarre définit la projection maximale XP composée de X^0, de ses spécificateurs et de ses compléments. L'ordre d'attachement ainsi que le nombre et la catégorie des spécificateurs et des compléments potentiels sont des paramètres dépendant de la langue et/ou de la catégorie de X^0. Le même message est envoyé à tous les objets, qui l'interpréteront différemment selon leur catégorie et leur langue. Des messages peuvent être envoyés à tous les objets ou seulement à certains d'entre eux, comme par exemple les messages de la théorie du Cas qui ne s'appliquent qu'aux syntagmes nominaux.

Polymorphisme - régularité Le polymorphisme est en relation directe avec la notion de message : des objets différents peuvent répondre aux mêmes messages parce qu'ils ont les mêmes méthodes dans leur interface, leur implémentation pouvant être différente. On retrouve cette notion dans la GU à deux niveaux. Tout d'abord au niveau de la structure de la phrase, par le principe de projection qui dit que l'information lexicale est représentée syntaxiquement, quelle que soit sa catégorie grammaticale, celle-ci déterminant le genre d'information disponible, et par la théorie Xbarre, qui construit la projection maximale de chaque mot selon le même schéma. Pour illustrer ce point, la classe de base *ItemLexical* (*cf.* 4.1) contient une opération *Projeter* qui projette l'information lexicale dans une structure syntaxique, selon la catégorie de l'objet, même si le même message est envoyé à tous les objets de la classe. Enfin, la définition même de la GU comme grammaire universelle, dont les principes s'appliquent à toutes les langues, ceux-ci étant interprétés différemment selon la valeur de certains paramètres propres à chaque langue.

4. LES OBJETS DE LA GU

Les principes de la GU s'appliquent sur deux sortes d'objets linguistiques, les catégories grammaticales et les chaînes syntaxiques. Par catégories grammaticales, nous entendons les catégories lexicales – les mots du langage avec leurs traits caractéristiques, les catégories minimales – les projections au niveau syntaxique des items lexicaux et les têtes fonctionnelles, ainsi que les catégories maximales, c'est-à-dire les constituants formés des catégories minimales, de leurs spécificateurs et de leurs compléments, que l'on appelle projections maximales. Les chaînes syntaxiques sont formées des catégories ci-dessus dans leur position de surface, auxquelles sont associées des catégories vides, ou traces, en position canonique. Dans cette section, nous allons d'abord donner un aperçu général de tous les objets, puis décrire dans le détail le schéma Xbarre (4.2), la théorie du Gouvernement (4.3), la théorie des Chaînes (4.4), la théorie Thématique (4.5) et la théorie du Cas (4.6).

4.1. APERÇU GÉNÉRAL

La première partie du Principe de Projection Étendu dit que l'information lexicale est projetée au niveau syntaxique. Un objet de la classe *ItemLexical*[3] correspond à l'entrée lexicale d'un mot. Ses attributs comprennent la forme graphique du mot, *graph*, sa forme phonologique, *phono*, ses traits syntaxiques, *syn*, et ses traits sémantiques, *sem*. Les caractéristiques morphologiques dépendant de la catégorie lexicale du mot, cette classe est étendue dans les sous-classes *MNom, MVerbe, MAdjectif, MDéterminant, MAdverbe, MPréposition, MConjonction*, qui ont comme attribut propre *morpho* encodant les traits morphologiques du mot. Cet attribut comprend les caractéristiques morphologiques associées à la catégorie du mot, par exemple *genre* et *nombre* pour *MNom*, *temps*, *mode*, *personne* pour *MVerbe*, etc... Selon le schéma Xbarre, l'information lexicale est projetée dans une structure syntaxique de même catégorie. La classe *ItemLexical* met donc à disposition l'opération *Projeter* qui génère au niveau minimal un objet de la classe *Tête*, et au niveau maximal un objet de la classe *Syntagme*. Le détail de l'opération est défini dans les sous-classes spécifiques, puisque chaque catégorie lexicale projette un objet de même catégorie au niveau syntaxique.

Les objets de la classe *Tête* comprennent les têtes lexicales, qui sont fixes, *Nom*, *Verbe*, *Adjectif*, *Préposition*, *Déterminant*, *Conjonction* et *Adverbe*, projetées à partir des éléments lexicaux, et les têtes fonctionnelles, qui n'ont en général pas de contenu lexical, mais qui remplissent une fonction particulière dans la phrase, comme *Inflection*, *Complémenteur*, qui permet la construction de phrases subordonnées et/ou interrogatives, et *Prédicat*, qui permet la construction de structures prédicatives autres que verbales[4].

Les différents syntagmes de la théorie Xbarre sont des sous-classes de la classe de base *Syntagme*: les projections lexicales *NP, VP, AdjP, DP, PrepP, AdvP, ConjP* sont projetées à partir de l'information lexicale, grâce à l'opération *Projeter* de la classe *ItemLexical*. Les projections fonctionnelles *IP, CP, PredP* sont projetées à partir de certaines projections maximales grâce à l'opération syntaxique *Projeter* de la classe *Syntagme*, dont le détail est défini dans les sous-classes. Par exemple, *IP* est projeté à partir de *VP*, *CP* est projeté à partir de *IP*, *PredP* est projeté à partir de *AdjP* ou *AdvP*. Certaines projections lexicales peuvent aussi être projetées à partir de projections maximales, comme *DP* qui peut être projeté de *NP* si le nom n'a pas de déterminant. Inversément, certaines projections fonctionnelles sont projetées à partir d'éléments lexicaux, telless les conjonctions de type complémenteur, qui projettent des *CP*.

Les relations structurales établies par la théorie du Gouvernement s'appliquent sur tous les nœuds de la structure syntaxique, catégories vides y compris. C'est pourquoi les briques de base sont constituées de trois classes d'objets assurant la compatibilité entre les différentes catégories. Au sommet de la hiérarchie se trouve la classe *Nœud*, qui assure la compatibilité entre les objets des catégories maximales et minimales. La classe *CatégorieMaximale* regroupe les sous-classes *Syntagme, PRO*, une catégorie vide apparaissant comme sujet des infinitives, *pro*, une catégorie vide apparaissant comme sujet de phrase tensée dans les langues dites à sujet nul, *TraceMaximale*, une catégorie vide associée à un syntagme, et *OP*, l'opérateur nul nécessaire dans certaines constructions. Un attribut de la classe *CatégorieMaximale* est *position*, qui prend ses valeurs dans *(arg, nonArg)* pour différentier les positions argumentales de celles non-argumentales. La classe *CatégorieMinimale* regroupe les sous-classes *Tête* et *TraceMinimale*, une catégorie vide associée à une tête. Les objets de la classe *Nœud* contiennent un attribut *indice* qui identifie la chaîne à laquelle appartient le nœud. Les objets des classes *TraceMaximale* et *TraceMinimale* ont un attribut *catégorie* identifiant la catégorie grammaticale de la tête ou du syntagme associé.

Les chaînes syntaxiques forment la deuxième sorte d'objets linguistiques. Les attributs de la classe *Chaîne* sont *séquence*, une suite d'objets de type *Nœud*, un attribut partagé par tous les éléments de la chaîne, *indice*, qui identifie la chaîne. Les chaînes n'ayant pas les mêmes propriétés selon qu'elles sont constituées d'objets de classe *CatégorieMaximale* ou *CatégorieMinimale*, la classe *Chaîne* est dérivée dans les sous-classes *ChaîneMinimale*, chaîne formée de catégories minimales, et *ChaîneMaximale*, chaîne formée de catégories maximales, elle-même étendue dans les sous-classes *ChaîneA*, formée de catégories maximales dont la tête est en position argumentale, et *ChaîneAbarre*, formée de catégories maximales dont la tête est en position non argumentale.

4.2. LE SCHÉMA XBARRE

Le schéma Xbarre définit une structure de données arborescente qui encode la structure hiérarchique de la phrase. La régularité du schéma Xbarre permettant d'éliminer le niveau intermédiaire défini par la théorie (Laenzlinger & Wehrli, 1991), une projection maximale XP comprend la tête de la projection X, une liste de spécificateurs, spec, et une liste de compléments, compl. Ces listes peuvent comprendre des ajouts et des arguments, la différence n'étant pas établie structurellement, mais fonctionnellement. La règle générale est

---

[3] La notation utilisée est la suivante : les objets, classes, attributs et opérations seront écrits en italique, les classes et les opérations commençant par une majuscule. Les attributs et opérations d'un objet sont préfixés par l'objet ou la classe en question.

[4] Cette liste de têtes fonctionnelles n'est pas exhaustive.

donnée en (1a), la structure complète d'une phrase est donnée en (1b-c) et la structure d'un syntagme nominal, selon l'hypothèse-DP d'Abney (1987) adoptée ici, est donnée en (1d).

(1) Schéma Xbarre
   a) XP → {spec X compl}
   b) CP → {spec Complémenteur IP}
   c) IP → {spec Inflection VP}
   d) DP → {spec Déterminant NP}

L'ordre de base des constituants n'est pas spécifié par la GU, mais il est déterminé par deux paramètres dont la valeur dépend de chaque langue. Le type du complément est déterminé par les traits de sous-catégorisation de l'élément lexical. Par contre, la position spécificateur est une position plus ou moins ouverte, c'est-à-dire qu'a priori n'importe quel élément peut y être inséré dans la mesure où sa présence ne viole aucun principe de la grammaire.

Un objet de la classe *Syntagme* possède les attributs correspondants *tête*, un objet de la classe *CatégorieMinimale*, *spec* et *compl*, deux listes d'objets de la classe *CatégorieMaximale*. L'ordre entre les différents éléments est donné par les deux attributs *xAvantCompl* et *specAvantX*, dont les valeurs sont prises dans l'ensemble *(vrai, faux, indéterminé)*. La richesse des relations grammaticales est spécifiée dans cette structure assez plate par l'attribut *dépendance*, qui prend ses valeurs dans l'ensemble *(argument, ajout, projection)*.

Pour le français et l'anglais, les deux paramètres d'ordre ont la valeur *vrai*, quelque soit la classe de *tête*. Dans certaines langues, les paramètres peuvent avoir une valeur différente selon la catégorie de la tête : en allemand, *XAvantCompl* a la valeur *faux* pour les catégories *Inflection*, *Verbe* et *Adjectif*, vrai pour les autres. Certaines langues dites à ordre libre des mots, tel le Warlpiri, n'ont pas de valeur fixe pour ces paramètres, qui prennent la valeur *indéterminé.*

4.3. LA THÉORIE DU GOUVERNEMENT

Cette théorie gère les relations structurales dans la phrase. La notion de gouvernement (11) est basée sur les relations de dominance (2), de commande (5), de barrière (10) et de minimalité relativisée (12), elle-même basée sur la notion de gouverneur potentiel typique (13). L'application de ces définitions dans notre modèle définit des opérations sur la classe de base *Nœud*. Nous utiliserons la notation *x.Op(y)* pour signifier que l'opération *Op* de l'objet *x* s'applique sur l'objet *y*. Ces opérations sont implémentées dans les sous-classes de *Nœud* lorsque *x* est spécifié appartenir à une certaine classe.

(2) **Dominance** (Chomsky, 1986b)
   Un noeud A est dominé par un noeud B si et seulement si le noeud A est dominé par chaque segment de B.

La relation de dominance définit les opérations *DominerImmédiatement* (3) et *Dominer* (4).

(3) x.DominerImmédiatement(y) [5] est vrai ssi
   $x \in$ Syntagme, et
   a) $y \in$ Syntagme, $y \in$ {x.spec, x.compl} et y.dépendance = argument, ou
   b) $y \in$ Tête et y = x.tête.

(4) x.Dominer(y) est vrai ssi
   $x \in$ Syntagme, et
   a) x.DominerImmédiatement(y), ou
   b) $\exists z \in$ Syntagme tq
      i) x.DominerImmédiatement(z) est vrai, et
      ii) z.Dominer(y) est vrai.

(5) **commande** (Chomsky, 1986b)
   A commande B si et seulement si a) A ne domine pas B, b) tous les X qui dominent A dominent aussi B[6].

Les relations de commande définissent les opérations *M-commander* (6) et *C-commander* (7).

[5] Nous utiliserons la notation x.Op(y) pour signifier que l'opération Op de l'objet x s'applique sur l'objet y.
[6] Si X est le premier noeud branchant, on a affaire à la c-commande stricte, si X est la première projection maximale, on a affaire à la m-commande.

(6) x.M-commander(y) est vrai ssi
  a) x.Dominer(y) est faux, y.Dominer(x) est faux, et
  b) $\forall z \in$ Syntagme, z.Dominer(x) est vrai et z.Dominer(y) est vrai.

(7) x.C-commander(y) est vrai ssi
  a) x.Dominer(y) est faux, y.Dominer(x) est faux, et
  b) $\forall z \in$ Syntagme, $x \neq$ z.tête et $y \notin$ {z.spec}, z.Dominer(x) est vrai et z.Dominer(y) est vrai.

(8) **Barrière** (Chomsky, 1986b)
A est une barrière pour B si A est une projection maximale et A domine immédiatement C, C est une catégorie bloquante[7] pour B (notion d'héritage) ou si A est une catégorie bloquante pour B, A n'est pas IP.

La notion de barrière définit les opérations *Bloquer* (9) et *EstBarrière* (10).

(9) x.Bloquer(y) est vrai ssi
  $x \in$ Syntagme, et
  a) x.Dominer(y) est vrai, et
  b) $\sim\exists z \in$ Tête tq z.AssignerRôleThématique(x) est vrai et z.Tête-gouverner(x) est vrai.

(10) x.EstBarrière(y) est vrai ssi
  $x \in$ Syntagme, et
  a) $\exists z \in$ Syntagme tq x.DominerImmédiatement(z) est vrai et z.Bloquer(y) est vrai, ou
  b) $x \notin$ IP, x.Bloquer(y) est vrai.

(11) **Gouvernement** (Rizzi, 1990)
Le gouvernement peut se faire par tête ou par antécédent. X α-gouverne[8] Y ssi
  a)
    i) X est une tête, X m-commande Y, X est soit N, V, Adj, P et I, ou
    ii) X est dans une position-W, (W = A, Abarre, X^0), X c-commande Y, X et Y sont coindicés,
  b) Il n'y a aucune barrière entre X et Y, et
  c) la Minimalité Relativisée est respectée.

(12) **Minimalité Relativisée** (Rizzi 1990)
X α-gouverne Y ssi il n'y a pas de Z tel que
  a) Z est un α-gouverneur potentiel typique pour Y, et
  b) Z c-commande Y et ne c-commande pas X.

(13) **α-gouverneur potentiel typique** (Rizzi 1990)
  a) Z est un tête-gouverneur potentiel typique pour Y ssi Z est une tête m-commandant Y.
  b) Z est un antécédent-gouverneur potentiel typique pour Y ssi
    i) Y est dans une chaîne-A, Z est un spécificateur-A c-commandant Y, ou
    ii) Y est dans une chaîne-Abarre, Z est un spécificateur-Abarre c-commandant Y, ou
    iii) Y est dans une chaîne-X^0, Z est une tête c-commandant Y.

Les relations de gouvernement définissent les opérations *Tête-Gouverner* (14), *Tête-gouverneur-potentiel* (15), *Antécédent-gouverner* (16) et *Antécédent-gouverneur-potentiel* (17).

(14) x.Tête-Gouverner(y) est vrai ssi
  $x \in$ Tête, et
  a) $x \in$ { Nom, Verbe, Adjectif, Préposition, Inflection}, x.M-commander(y) est vrai, et
  b) $\exists z \in$ Syntagme tq z.Dominer(x) est vrai, z.Dominer(y) est vrai, $\forall u \in$ Syntagme, z.Dominer(u) est vrai[9] et u.EstBarrière(y) est faux, et
  c) $\sim\exists$ v tq v.Tête-gouverneur-potentiel(y) est vrai, v.C-commander(y) est vrai et v.C-commander(x) est faux.

[7] C est une catégorie bloquante pour B ssi C n'est pas L-marqué et C domine B. A L-marque B ssi A est une catégorie lexicale qui théta-gouverne (gouverne et assigne un rôle thématique) B.
[8] α est une variable qui peut prendre comme valeur tête ou antécédent.
[9] Ces trois conditions permettent de sélectionner u dans le même domaine que x et y.

(15) x.Tête-gouverneur-potentiel(y) est vrai ssi
x ∈ Tête, et x.M-commander(y) est vrai.

(16) x.Antécédent-Gouverner(y) est vrai ssi
a) x ∈ CatégorieMaximale, x.position = arg, et
i) y ∈ TraceMaximale, x.indice = y.indice, x.C-commander(y) est vrai, et
ii) ∃ z ∈ Syntagme tq z.Dominer(x) est vrai, z.Dominer(y) est vrai, ∀u ∈ Syntagme, z.Dominer(u) est vrai et u.EstBarrière(y) est faux, et
iii) ~∃v ∈ CatégorieMaximale, tq v ≠ x, v.position = arg, v.Antécédent-gouverneur-potentiel(y) est vrai, v.C-commander(y) est vrai, v.C-commander(x) est faux
b) x ∈ CatégorieMaximale, x.position = nonArg, et
i) y ∈ TraceMaximale, x.indice = y.indice, x.C-commander(y) est vrai, et
ii) ∃ z ∈ Syntagme tq z.Dominer(x) est vrai, z.Dominer(y) est vrai, ∀u ∈ Syntagme, z.Dominer(u) est vrai et u.EstBarrière(y) est faux, et
iii) ~∃v ∈ CatégorieMaximale tq v ≠ x, v.position = nonArg, v.Antécédent-gouverneur-potentiel(y) est vrai, v.C-commander(y) est vrai, v.C-commander(x) est faux
c) x ∈ CatégorieMinimale, et
i) y ∈ TraceMinimale, x.C-commander(y) est vrai, x.indice = y.indice, et
ii) ∃ z ∈ Syntagme tq z.Dominer(x) est vrai, z.Dominer(y) est vrai, ∀u ∈ Syntagme, z.Dominer(u) est vrai et u.EstBarrière(y) est faux, et
iii) ~∃v ∈ CatégorieMinimale tq v ≠ x, v.Antécédent-gouverneur-potentiel(y) est vrai, v.C-commander(y) est vrai, v.C-commander(x) est faux.

(17) x.Antécédent-gouverneur-potentiel(y) est vrai ssi
a) x ∈ CatégorieMaximale, y ∈ TraceMaximale, ∃ c ∈ ChaîneA tq y ∈ c, et
∃ z ∈ Syntagme, tq x ∈ z.spec, x.position = arg, x.C-commander(y) est vrai, ou
b) x ∈ CatégorieMaximale, y ∈ TraceMaximale, ∃ c ∈ ChaîneAbarre tq y ∈ c, et
∃ z ∈ Syntagme, tq x = z.spec, x.position = nonArg, x.C-commander(y) est vrai, ou
c) x ∈ CatégorieMinimale, y ∈ TraceMinimale, ∃ c ∈ ChaîneMinimale tq y ∈ c, et
d) ∃ z ∈ Syntagme, tq x = z.tête, x.C-commander(y) est vrai.

4.4. LA THÉORIE DES CHAÎNES

Certains principes de la grammaire exigent que des éléments, projections maximales ou têtes, ne restent pas dans leur position canonique, mais se déplacent dans d'autres positions. Le Principe de Projection et le Principe de Préservation des Structures exigent que la position de base continue d'exister, remplie par une trace de l'élément déplacé. Les mouvements sont codés dans les chaînes syntaxiques, qui comprennent les constituants déplacés et les traces qu'ils ont laissées dans la position de base. Le premier élément de la chaîne est appelé tête de la chaîne, c'est une projection maximale ou une tête, le dernier est appelé pied de la chaîne, c'est une trace qui se trouve dans la position de base, les autres étant des traces intermédiaires. La tête de la chaîne détermine son type : une projection maximale en position argumentale (18a) détermine une chaîne-A, une projection maximale en position non argumentale (18b) détermine une chaîne-Abarre, et une tête (18c) détermine une chaîne-X^0.

(18)
a) *$[_{IP}$ $[_{DP}$ Cette histoire$]_i$ est $[_{VP}$ racontée t_i $[_{PP}$ par tous les habitants$]]]$.*
b) *$[_{CP}$ $[_{DP}$ Que$]_i$ veux- $[_{IP}$ tu manger t_i ce soir?$]]$*
c) *$[_{CP}$ $[_{C}$ Mangeras$]_i$- $[_{IP}$ $[_{DP}$ tu$]$ t'_i $[_{VP}$ t_i une glace?$]]]$*

Les chaînes correspondant aux exemples ci-dessus sont respectivement ($[_{DP}$ Cette histoire$]_i$, t_i), ($[_{DP}$ Que$]_i$, t_i) et ($[_{V}$ mangeras$]_i$, t'_i , t_i). Les traces sont soumises au Principe des Catégories Vides (19).

(19) **Principe des catégories vides**
Les traces doivent être proprement gouvernées. A gouverne proprement B ssi A théta-gouverne B et/ou A antécédent-gouverne B.

Les traces sont des objets des classes *TraceMaximale* et *TraceMinimale*, qui doivent être antécédent-gouvernées, ce qui détermine une condition de bonne formation générale, applicable aux chaînes maximales comme minimales (20).

(20) Soit c ∈ Chaîne, c.séquence = (x_1, x_i, x_{i+1}, …, x_n), avec n ≥ 1. Alors, c est bien formée si :

$\forall x_1$, avec $1 \leq i \leq n$, x_i.Antécédent-Gouverner(x_{i+1}) est vrai.

D'autres conditions de bonne formation s'appliquent sur les chaînes maximales. Elles découlent de certains principes de la grammaire, qui ont été reformulés en termes de chaînes : le Critère Thématique présenté à la section 4.5 et le Filtre du Cas présenté à la section 4.6.

4.5. LA THÉORIE THÉMATIQUE

La théorie thématique règle l'assignation des rôles thématiques aux arguments. Le prédicat verbal donne le corps à la phrase : le verbe et ses arguments déterminent les constituants indispensables de la phrase. Les relations sémantiques entre le prédicat et les arguments sont spécifiées dans la grille thématique du prédicat, qui est une liste non ordonnée de rôles thématiques, dont les principaux sont l'agent, le thème, le bénéficiaire (21), et qui forment une hiérarchie (22). Larson émet un principe de projection thématique très général qui tient compte de cette hiérarchie (23).

(21) *Jean*$_{+\text{AGENT}}$ *donne un livre*$_{+\text{THEME}}$ *à Marie*$_{+\text{BÉNÉFICIAIRE}}$.

(22) **Hiérarchie thématique** (Grimshaw, 1990)
Agent > Expérient > Thème > Bénéficiaire > ...

(23) **Principe de Projection Thématique** (Larson, 1988)
Si un prédicat a détermine des rôles thématiques t1,...tn, alors le rôle le plus bas dans la hiérarchie thématique est assigné à l'argument le plus bas dans la structure, etc... jusqu'au rôle le plus haut qui est assigné à l'argument le plus haut.

Dans le cas de deux arguments, la position complément est remplie par le rôle le plus bas dans la hiérarchie et la position spécificateur par le rôle le plus haut. Quand il y a trois arguments, il y a construction d'un VP-shell (Larson, 1988), et le rôle le plus haut est projeté dans la position spécificateur du VP englobant. Quand il n'y a qu'un seul argument, le principe est court-circuité par la hiérarchie thématique : si l'unique rôle est un agent ou un expérient, il est projeté en position spécificateur, si c'est un thème, il est projeté dans la position complément.

Les objets des classes *CatégorieMaximale* contiennent un attribut spécifique, *rôle*, qui prend ses valeurs dans l'ensemble *(agent, expérient, thème, bénéficiaire, but, instrument,…)*. D'être assigneur de rôle thématique est une propriété des objets de la classe *ItemLexical*, typiquement *MPréposition*, *MVerbe* et *MAdjectif*, propriété qui est percolée à la *Tête* correspondante. L'assignation de rôle thématique définit l'opération *AssignerRôleThématique* (24) des objets de la classe *Tête*. Soient les objets x et y, alors

(24) x. AssignerRôleThématique (y,r) est vrai ssi
$x \in$ Tête, et
a) x.M-commander(y)
b) $\exists c \in$ ChaîneMaximale tq c.séquence = (…,y), y.position = arg, et
c) $\exists r \in$ x.grille-thématique.
Si x. AssignerRôleThématique (y,r) est vrai, alors y.rôle $\leftarrow$ r.

Le Critère Thématique (25) énonce une condition de bonne formation des chaînes argumentales, énoncée formellement en (26). Une chaîne argumentale est une chaîne dont le pied se trouve en position argumentale. Les objets des classes *CatégorieMaximale* contiennent aussi l'attribut spécifique, *rôle*, qui reçoit un rôle thématique.

(25) **Critère Thématique** (Rizzi 1986)
Les chaînes argumentales doivent porter un et un seul rôle thématique, et chaque rôle thématique doit être assigné à une et une seule chaîne argumentale.

(26) Soit $c \in$ ChaîneMaximale, c.séquence = (x_1, x_i, x_{i+1}, …, x_n), avec $n \geq 1$. Alors, c est bien formée si
a) $\exists z \in$ Syntagme et $\exists v \in$ Tête tq v = z.tête, v.Tête-Gouverner(x_n) et v.AssignerRôleThématique(x_n,r) est vrai, et
b) $\sim\exists ch \in$ ChaîneMaximale, tq ch.indice $\neq$ c.indice, ch.séquence = (x_1, x_j, x_{j+1}, …, x_m), avec $m \geq 1$, et v.AssignerRôleThématique(x_m,r) est vrai, alors
Si les conditions ci-dessus sont remplies, alors c.rôle $\leftarrow$ r.

4.6. LA THÉORIE DU CAS

La théorie du Cas gère l'assignation du Cas abstrait aux syntagmes nominaux. Elle fait la distinction entre cas structural et cas inhérent. Les cas structuraux sont assignés sous gouvernement de tête et comprennent le nominatif, assigné par l'inflection à son spécificateur, et l'accusatif, assigné par le verbe à son complément. Le cas inhérent étant une propriété lexicale, c'est un paramètre de la langue. Les assigneurs de cas inhérent sont typiquement la préposition et le verbe en français, et la préposition, le verbe et le nom en anglais. Les cas inhérents comprennent entre autres l'accusatif, le datif et le génitif. L'assignation de cas inhérent se fait sous théta-gouvernement (Condition de Cas Inhérent, Chomsky, 1986a).

Les objets de la classe *ChaîneMaximale* et *DP* comprennent un attribut spécifique, *cas*, qui prend ses valeurs dans l'ensemble *(nominatif, accusatif, datif, génitif, etc…).* L'assignation de cas définit l'opération *AssignerCas* (27) des objets de la classe *Tête*.

(27) x.AssignerCas(y,cs) est vrai ssi
 $x \in$ Tête, et
 a) x.Tête-Gouverner(y) est vrai, et
  i) x = Inflection, $y \in$ x.spec, y.dépendance = argument, ou
  ii) x = Verbe, $y \in$ x.compl, y.dépendance = argument.
 b) x.Tête-Gouverner(y) est vrai, x.AssignerRôleThématique(y,r) est vrai et
 … les conditions spécifiques dépendent des langues.
 Si les conditions ce-dessus sont remplies, alors y.cas = cs.

Pour satisfaire le Filtre du Cas (28), les syntagmes nominaux qui ne se trouvent pas dans une position où un cas peut être assigné peuvent se déplacer dans une position libre. C'est typiquement le cas du sujet, qui se déplace de la position spécificateur de VP, qui n'est pas une position de cas structurel, en position spécificateur de IP, où il peut recevoir le cas nominatif (29a), ou encore le complément d'objet direct qui se déplace aussi en position spécificateur de IP (29b) dans les constructions passives et à montée, car le verbe ne peut plus assigner le cas structurel à son complément (Généralisation de Burzio, 1986).

(28) **Filtre du Cas**
 Chaque chaîne dont la tête est un syntagme nominal doit porter un et un seul cas abstrait.

(29)
 a) *$[_{IP}$ Jean$_{i+NOM}$ a $[_{VP}$ t_i mangé une pomme$_{+ACC}$]].*
 b) *$[_{IP}$ Jean$_{i+NOM}$ a $[_{VP}$ été arrêté t_i]].*
 c) $[_{CP}$ *$[_{DP}$ Qui$]_i$ as- tu$_{+NOM}$ $[_{VP}$ appelé t_{i+ACC}]]?*

Les chaînes A reçoivent le cas par le DP qui s'est déplacé (29a-b), tandis que les chaînes Abarre le reçoivent par l'intermédiaire de leur trace (29c). Le filtre du Cas détermine une condition de bonne formation des chaînes maximales (30).

(30) Soit $c \in$ ChaîneMaximale, c.séquence = $(x_1, x_i, x_{i+1}, \ldots, x_n)$, avec $n \geq 1$. Alors, c est bien formée si :
 a) $x_1 \in$ DP, et
  i) x_n.cas # vide, et $\forall\ x_j$ avec $1 \leq j < n$, x_j.cas = vide
 b) $y \in$ DP, $\exists\ c \in$ ChaîneA, c.séquence = (y,…), c.cas = vide, ou
 c) $y \in$ TraceMaximale, $\exists\ c \in$ ChaîneAbarre, c.séquence = (…, y), c.cas = vide,
 Si les conditions sont satisfaites, alors c.cas = y.cas.

5. CONCLUSION

Nous avons proposé dans ce papier une modélisation de la grammaire universelle selon les objets sur lesquels s'appliquent les principes, en partant du parallèle que nous avons pu tirer entre les concepts de la GU et ceux du paradigme des objets. Nous avons défini deux grandes classes d'objets linguistiques, les catégories grammaticales et les chaînes syntaxiques. Les principes de la GU définissent des opérations sur ces objets, au niveau le plus général, sur les classes de base, ou au niveau plus spécifique des sous-classes. Même si la modélisation n'est pas complète, nous espérons avoir ainsi montré que la GU peut être modularisée, non pas dans le sens d'un module par sous-théorie ou sous-ensemble de principes, mais d'une classe par type d'objet.

Puisque la GU est par définition universelle, les grammaires spécifiques à chaque langue seront des instances de ces classes, avec une valeur donnée aux paramètres orientant l'application des principes. Les constructions spécifiques à une langue donnée donneront lieu à la création de sous-classes spécialisées, sans pour autant modifier la GU elle-même.

Nous pensons que la GU telle que modélisée ici peut être utilisée comme base grammaticale en analyse et en génération, et nos efforts se tourneront dorénavant vers une implémentation de ce système.

BIBLIOGRAPHIE

Abney S., 1987. "The English noun phrase in its sentential form". Thèse de doctorat non publiée. MIT.

Berwick R., Abney S., Tenny C. éditeurs, 1991. *Principle-Based Parsing: Computation and Psycholinguistics*. Kluwer Academic Press.

Booch G., 1991. *Object-Oriented Design with Applications*. Benjamin/Cummings.

Burzio L., 1986. *Italian Syntax. A government and Binding Approach*. Reidel, Dordrecht.

Chomsky N., 1981. *Lectures on Government and Binding*. Foris Publications.

Chomsky N., 1986a. *Knowledge of Language : Its nature, Origin and Use*. Praeger.

Chomsky N., 1986b. *Barriers*. Linguistics Inquiry Monograph. MIT.

Crocker M., 1996. Computational Psycholinguistics: An Interdisciplinary Approach to the Study of Language, Kluwer Academic Publishers, The Netherlands.

Daelemans W., Gazdar G. (eds), 1990. Inheritance in Natural Language Processing. Université de Tilburg.

Door B., 1990. *Lexical Conceptual Structure and Machine Translation*. Thèse de doctorat. MIT.

Evans R., Gazdar G., Weir D., 1995. *Encoding Lexicalized Tree Adjoining Grammars with a non monotonic Inheritance Hierarchy*. Actes des ACL, pp 77-84.

Fodor J. A., 1983. *The modularity of mind : An essay on faculty psychology*. MIT Press.

Fong S., 1991. *Computational Properties of Principal-Based Grammatical Theories*. Thèse de doctorat. MIT.

Haegeman L., 1994. *Introduction to Government and Binding Theory*. Blackwell.

Hahn U., Neuhaus P., Bröker N., 1997. Message-Passing Protocols for Real-World Parsing - An Object-Oriented Model and its Preliminary Evaluation. *Proceedings of the International Workshop on Parsing Technologies (IWPT-97)*, Boston/MA: MIT, pp. 101-112

Jacobson, I., Christerson, M., Jonsson, P., Övergaard, G. *Object-Oriented Software Engineering*, Addison-Wesley, 1992.

Grimshaw J., 1990. *Argument Structure*. Linguistic Inquiry Monograph. MIT Press.

Laenzlinger C., Wehrli E., 1991. FIPS : Un analyseur interactif pour le français. *T.A. Informations* **2**, 35-49.

Larson, R. K, 1988. On the double object construction. *Linguistic Inquiry* **19**, 335-391.

Merlo P., 1995. Modularity and Information Content Classes in Principle-Bases Parsers. *Computational Linguistics* vol. **21**, num. **4**, pp 515-541.

Merlo P., 1996. *Parsing with Principles and Classes of Iinformation*. Kluwer Academic Publisher.

Reinhard S., Gibbon D., 1991. *Prosodic inheritance and morphological generalisation*. Actes de l'EACL, pp 131-136.

Rizzi L., 1986. Conditions de bonne formation sur les chaînes. *Modèles Linguistiques* **7**, 119-157.

Rizzi L., 1990. *Relativized Minimality*. Linguistic Inquiry Monograph. MIT Press.

Rumbaugh J., Blaha M., Premerlani W., Eddy F., Lorensen W., 1991. *Object-Oriented Modeling and Design*. Prentice-Hall.

UML, 1997. Unified Modeling Language v1.1. Rational Software Corporation.

Walther C., 1994. *Modularity and Parsing*. Thèse de doctorat. Université de Genève.

Wehrli E., 1997. *L'analyse syntaxique des langues naturelles*. Masson.

Zajac R., 1992. *Toward computer-aided linguistic engineering*. Actes de Coling, pp 828-834.

Zampolli A., 1991. *Towards Reusable Linguistic Resources*. Actes de EACL.