Produire des ressources électroniques à partir de descriptions formelles : application aux langues peu dotées

Denys Duchier[1] Yannick Parmentier[1] Simon Petitjean[2] Emmanuel Schang[3]
(1) LIFO, Université d'Orléans, 45067 Orléans, France
(2) CRC991, Heinrich Heine Universität, D-40225 Düsseldorf, Allemagne
(3) LLL, Université d'Orléans, 45067 Orléans, France
denys.duchier@univ-orleans.fr, yannick.parmentier@univ-orleans.fr,
simon.petitjean@phil.uni-duesseldorf.de, emmanuel.schang@univ-orleans.fr

RÉSUMÉ
Dans cet article, nous montrons comment les langages de description, et le système XMG2 en particulier, peuvent être utilisés pour produire à moindre coût (en termes d'hommes.années) une ressource électronique linguistiquement précise et relativement couvrante, ouvrant ainsi la voie à la création de ressources pour les langues peu dotées. Nous insistons également sur le fait que l'utilisation d'un langage de description permet (sans aucun sur-coût) de documenter une langue et d'évaluer la capacité d'une théorie linguistique à décrire un certain jeu de données.

ABSTRACT
Creating e-resources from formal descriptions : application to under-resourced languages

In this paper, we show how description languages, and the XMG2 framework in particular, can be used to design at a lesser cost (in terms of man-year) a linguistically precise and relatively large electronic resource. This paves the way to the creation of resources for under-resourced languages. We furthermore emphasize the fact that the use of description languages makes it possible (at no additional cost) to document a natural language and evaluate the capacity of a linguistic theory to describe a given data set.

MOTS-CLÉS : langage formel, langage de description, ressources électroniques, métagrammaire.

KEYWORDS: formal language, description language, electronic resources, metagrammar.

1 Introduction

Le fait de pouvoir disposer de ressources linguistiques (telles que des lexiques, des grammaires ou encore des corpus annotés) dans un format électronique *ouvert* est non seulement important pour permettre une sauvegarde de la connaissance que l'on a d'une langue, mais aussi pour permettre d'adopter une vue introspective sur celle-ci (en appliquant des traitements sur ces ressources pour en vérifier la cohérence ou encore en extraire des informations statistiques plus ou moins fiables selon leur taille et leur représentativité), ou encore bien sûr pour permettre le développement d'applications de traitement automatique des langues nécessitant une modélisation relativement précise de la langue (cas de la traduction automatique ou du dialogue homme-machine par exemple).

Malheureusement, le coût de développement de telles ressources est habituellement très élevé (de l'ordre de plusieurs homme.années pour des grammaires électroniques noyaux comme par exemple la

grammaire d'arbres adjoints du français FTAG (Abeillé *et al.*, 1999)).

Alors que les premières ressources linguistiques électroniques étaient produites à la main, il est devenu courant de recourir à des méthodes (totalement ou en partie) automatiques pour les extraire à partir de données de base. Ainsi, il est devenu possible d'extraire un lexique à partir d'un corpus brut (Sagot *et al.*, 2006), ou encore de produire une grammaire noyau par transformation de règles canoniques (Prolo, 2002). [1]

Parmi les approches semi-automatiques de création de ressources linguistiques, on peut distinguer les approches basées sur des *langages de description* des autres approches, par le fait qu'elles ne nécessitent aucune ressource externe préalable, et sont de ce fait adaptées à la création rapide de prototypes de ressources linguistiques *from scratch* [2]. Les langages de description reposent sur des mécanismes d'abstraction permettant de définir de manière déclarative les régularités (voire dans certains cas les irrégularités) d'une langue. [3] Cette description déclarative est ensuite traitée automatiquement (*compilée*) pour produire les unités décrites (par exemple les entrées d'un lexique ou d'une grammaire électronique). Comme nous le verrons, ces descriptions offrent divers avantages dont principalement (i) le fait de constituer en elle-même, de par leur structure modulaire et hiérarchique, une documentation des unités de la langue (mots ou règles syntaxiques dans notre cas), et (ii) le fait de permettre de vérifier la portée d'une théorie linguistique (en observant l'adéquation entre structures décrites et structures observées chez les locuteurs de la langue considérée).

Dans ce qui suit, nous utiliserons le système libre et ouvert XMG2 (Petitjean *et al.*, 2016) pour illustrer l'utilisation pratique des langages de description pour produire des ressources linguistiques. XMG2 est un système permettant notamment de définir des langages de description de manière modulaire, et de générer à la volée un compilateur pour chacun de ces langages. [4]

L'article est structuré comme suit. En Section 2, nous présentons le concept de description formelle de ressource linguistique (métagrammaire) et l'état de l'art dans ce domaine. En Section 3, nous présentons brièvement le système XMG2. En Section 4, nous montrons deux cas d'utilisation des langages de description (et du système XMG2) pour décrire des langues peu dotées. Concrètement, nous montrons comment un langage de description permet de décrire un lexique nominal en définissant des combinaisons d'affixes en fonction d'une racine nominale donnée pour produire un lexique électronique d'une langue bantoue (sous-section 4.1), ou encore de décrire de manière concise un ensemble de règles grammaticales arborescentes en capturant des généralisations entre celles-ci pour produire une grammaire du são-tomense (sous-section 4.2). Enfin (Section 5) nous concluons et présentons des pistes de recherche dans ce domaine.

2 Décrire des ressources linguistiques : les métagrammaires

Les langages de description sont un outil utilisé depuis un certain temps déjà en informatique, comme en témoignent les exemples du langage HTML pour les interfaces graphiques (ou pages web) sur

1. À noter que l'annotation automatique de corpus repose souvent sur la disponibilité préalable de ressources externes, par exemple des corpus d'entraînement pour la création de corpus arborés par analyse syntaxique automatique.

2. « À partir de rien ».

3. Cette description déclarative est souvent appelée *métagrammaire*, car son utilisation originelle résidait dans la description de grammaires.

4. XMG2 peut être qualifié de *méta-compilateur*, puisqu'il permet de compiler le compilateur d'un langage de description en fonction d'une définition de ce langage.

internet ou encore du langage LaTeX pour la description de documents. Leur utilisation en traitement automatique des langues est elle aussi relativement ancienne puisqu'elle remonte aux années 80 avec notamment les langages PATRII (Shieber, 1984) et DATR (Evans & Gazdar, 1996) pour la représentation de ressources lexicales. Dans ces contextes, un langage de description n'est autre qu'un langage formel dont la syntaxe est définie au moyen d'une grammaire hors-contexte dans la hiérarchie de Chomsky (1957), et dont la sémantique est définie en termes d'interprétation des expressions de ce langage, pour produire par exemple un affichage (cas du langage HTML), un document PDF (cas du langage LaTeX) ou encore une ressource linguistique particulière (cas des langages PATRII/DATR). Il convient de noter que les langages PATRII/DATR étaient assez limités en termes d'expressivité, n'offrant que peu de moyens de définir des abstractions (seules des macros ou des règles de transformation étaient possibles).

Le concept de métagrammaire est apparu quant à lui à la fin des années 90, dans les travaux de Candito (1999). La métagrammaire correspondait alors à la description d'une grammaire d'arbres adjoints (Joshi *et al.*, 1975) des verbes en français et italien. Cette description reposait sur une analyse tridimensionnelle de la langue. Dans une première dimension était décrite une hiérarchie de cadres de sous-catégorisation pour les prédicats verbaux (représentation de la valence, l'ordre et la catégorie des arguments du verbe), dans une deuxième dimension des règles de transformation (redistributions des fonctions grammaticales comme par exemple lors du passage de l'actif au passif) et enfin, en troisième dimension, une hiérarchie de réalisations syntaxiques pour les différentes fonctions grammaticales présentes dans chacun des cadres de sous-catégorisation considérés. La motivation à ces travaux était de contrôler la richesse syntaxique des formalismes grammaticaux fortement lexicalisés (comme le sont les grammaires d'arbres adjoints) au moyen d'un outil descriptif à la fois formel et linguistiquement motivé (ici par la structure tridimensionnelle).

Ces travaux ont marqué le début d'une lignée de recherches sur les langages de description pour les ressources linguistiques, notamment les grammaires d'arbres. Les limitations des travaux de Candito résidaient dans (i) la structure tridimensionnelle rigide de la métagrammaire, et (ii) l'utilisation de formules logiques de description d'arbres dont les variables avaient une portée globale à la métagrammaire. [5] Les principales alternatives à l'approche de Candito correspondent aux systèmes LexOrg (Xia, 2001) (qui propose d'utiliser des variables locales dans les formules logiques de description d'arbres), et FRMG (Villemonte De La Clergerie, 2010) et XMG (Crabbé *et al.*, 2013) (permettant tous deux de définir finement la portée des variables utilisées dans les formules de description). C'est ce dernier système XMG qui a servi d'inspiration au développement du système XMG2, que nous allons présenter dans la section suivante. À la différence du système XMG, qui *ne* permet *que* de compiler des grammaires décrites au moyen du langage de description XMG, le système XMG2 permet (1) de définir modulairement un langage de description, et (2) de compiler un compilateur pour ce langage (qui sera ensuite utilisé par le linguiste pour décrire et compiler sa ressource linguistique).

3 Le système XMG2

Définition modulaire d'un langage de description. Comme nous l'avons mentionné précédemment, le système XMG2 [6] permet de compiler un compilateur pour divers langages de description. Ces

5. Un identifiant de variable dénotant une information linguistique donnée ne pouvait plus être réutilisé ailleurs dans la description pour référer à une autre information, ce qui revient à gérer un ensemble d'identifiants de variables très grand.

6. `http://dokufarm.phil.hhu.de/xmg/?animal=xmg`

langages doivent au préalable avoir été définis formellement. Pour cela, XMG2 offre un ensemble de langages élémentaires (appelés *briques de langage* dans la terminologie XMG2) qui peuvent être assemblés déclarativement (Petitjean *et al.*, 2016). Les briques de langages actuellement disponibles nativement dans XMG2 incluent notamment :

– B1 un langage de description de structures de traits,
– B2 un langage de description d'arbres à base de dominance/précédence entre nœuds (Rogers & Vijay-Shanker, 1994),
– B3 un langage de description de formules sémantiques « plates » (Bos, 1995),
– B4 un langage de description de *frames* sémantiques (Lichte & Petitjean, 2015).

Un assemblage de ces briques prend la forme d'un fichier texte au format YAML (liste de clés-valeurs) comme décrit dans (Petitjean, 2014; Petitjean *et al.*, 2016). Il est ainsi possible par exemple d'assembler les briques de langage B1 et B2 ci-dessus, et à partir de cet assemblage, de générer un compilateur pour un langage de description permettant de décrire des arbres syntaxiques dont les nœuds utiliseraient des structures de traits comme étiquettes (cf exemple de Petitjean *et al.* (2016)).

Utilisation d'un langage de description. À partir de la définition formelle d'un langage de description L par assemblage de briques de langage, le système XMG2 génère (méta-compile) un compilateur pour ce langage. Ce dernier peut alors être utilisé par un expert linguiste pour sa tâche de description des unités d'une langue donnée. En d'autres termes, cet expert va écrire une métagrammaire au moyen du langage L, et cette métagrammaire sera ensuite compilée par le compilateur précédemment généré pour produire les unités de la langue (c'est-à-dire la ressource linguistique), comme illustré sur la Figure 1.

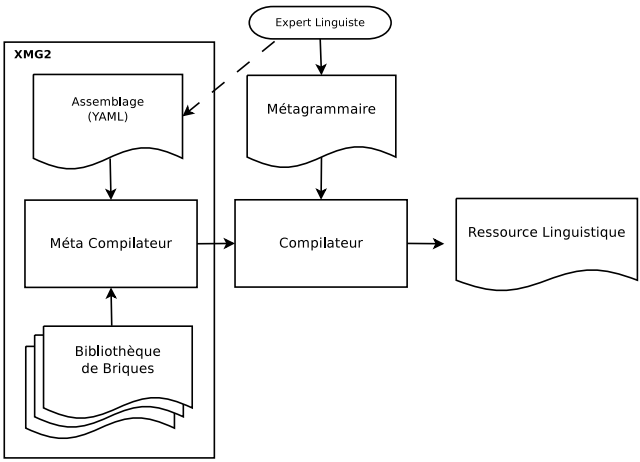


FIGURE 1 – Utilisation du système XMG2

Dans l'exemple de l'assemblage B1-B2 ci-dessus, une fois le compilateur pour le langage L_{B1-B2} généré, il peut être utilisé par un expert linguiste pour compiler une métagrammaire écrite dans le langage L_{B1-B2} et ainsi produire une ressource électronique (au format XML). Les éléments de cette ressource sont des arbres dont les nœuds sont équipés de structures de traits (règles d'une grammaire lexicale fonctionnelle ou d'une grammaire d'arbres adjoints par exemple).

À propos du logiciel XMG2. Il convient de noter que le logiciel XMG2 est distribué sous licence GPL, ce qui signifie entre autres que (1) son code source est accessible librement, et (2) qu'il est possible de le télécharger et de l'utiliser gratuitement. [7] En outre, les formats manipulés par le système XMG2 (format utilisé pour la définition d'un assemblage de briques, format utilisé pour la métagrammaire, format XML utilisé pour représenter la ressource linguistique électronique produite) sont tous ouverts (leur spécification est disponible librement), ce qui facilite la réutilisation de ces ressources. Ces formats pourraient constituer une norme de répresentation de ressources linguistiques *de facto*, si par exemple une communauté d'experts se mettait à les utiliser pour représenter leurs propres ressources (que celles-ci aient été produites par le système XMG2 ou un autre).

4 Application aux langues peu dotées

Nous allons ici illustrer l'utilisation de langages de description (et du système XMG2) pour produire deux types de ressources électroniques pour des langues peu dotées. Dans un premier temps, nous allons montrer comment décrire un lexique morphologique de l'ikota (langue bantoue), et ensuite comment décrire une grammaire d'arbres du são-tomense.

4.1 Décrire un lexique nominal de l'ikota

L'ikota est une langue bantoue parlée principalement au Gabon (où le nombre de locuteurs est estimé à 25000) et en République Démocratique du Congo. C'est une langue tonale à deux tons qui compte dix classes nominales et possède un accord généralisé dans le syntagme nominal. Elle compte en outre 3 classes verbales, et un infinitif où l'élément verbal est préfixé par une classe nominale.

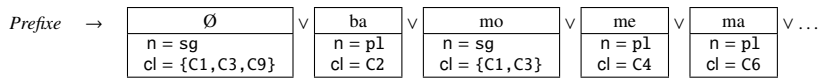
Notre modélisation linguistique se base sur les travaux de Duchier *et al.* (2012), qui utilisaient déjà le concept de métagrammaire (notamment le langage XMG), pour décrire la morphologie verbale de l'ikota. Ici, nous décrivons les formes nominales en ikota comme composées d'une racine nominale (RN) précédée d'un préfixe défini en fonction de la classe nominale de la racine considérée, comme indiqué ci-dessous :

classe nominale	préfixe
CL 1	mò-, Ø-
CL 2	bà-
CL 3	mò-, Ø-
CL 4	mè-
CL 5	ì-, ʤ-
CL 6	mà-
CL 7	è-
CL 8	bè-
CL 9	Ø-
CL 14	ò-, bò-

Nous allons utiliser cette modélisation pour créer un lexique noyaux de formes nominales de l'ikota (lexique qui permettra entre autre de confronter cette modélisation linguistique aux données de terrain).

7. Plusieurs compilateurs pour langages de description (dont L_{B1-B2}) ont été pré-méta-compilés et sont disponibles en ligne à l'adresse `http://xmg.phil.hhu.de/index.php/upload/compile_grammar`.

Pour ce faire, nous allons définir des blocs élémentaires (appelées *classes* dans la terminologie XMG2), et contenant des contributions pour les préfixes définis ci-dessus et pour les racines nominales. Les valeurs possibles pour les préfixes (et les traits associés) sont définis dans une classe *Prefixe* :



Notons que cette description des préfixes peut s'écrire directement au moyen de la brique de langage `tf_morph` disponible dans le logiciel XMG2, comme suit (le symbole `|` représente la disjonction) :

```
class Prefixe
{
        <morph>{
        {
        { n=sg; cl=@{C1,C3,C9}; prefix <- ""}
        |
        { n=pl; cl=C2; prefix <- "ba"}
        |
        { n=sg; cl=@{C1,C3}; prefix <- "mo"}
        |
        { n=pl; cl=C4; prefix <- "me"}
        |
        { n=pl; cl=C6; prefix <- "ma"}
        |
        { n=sg; cl=C5; prefix <- "dz"}
        |
        { n=sg; cl=C7; prefix <- "e"}
        |
        { n=pl; cl=C8; prefix <- "be"}
        |
        { n=sg; cl=C14; prefix <- "bo"}
        }
    }
}
```

De manière similaire, nous pouvons définir des classes pour décrire un nom comme appartenant à une certaine classe nominale au singulier *ou* à une autre au pluriel.

Nous pouvons ensuite définir les racines nominales (RN) considérées en indiquant à chaque fois leur classe nominale, comme par exemple « mbòka » (village, classe 14) :

$$RN \quad \rightarrow \quad (mbòka \wedge \mathsf{cl{=}C14}) \vee \ldots$$

Enfin, un nom sera décrit comme correspondant à la conjonction d'un préfixe et d'un racine nominale (avec la contrainte additionnelle que les traits associés à chacune de ces contributions doivent pouvoir s'unifier) :

$$Nom \quad \rightarrow \quad Prefixe \wedge RN$$

Toutes les combinaisons préfixe-RN sont calculées par le compilateur et seules celles étant valides (traits unifiables) sont conservées. Le lexique ainsi produit est en adéquation avec les observations de terrain (Magnana Ekoukou, 2015).

On remarque que la structure de la métagrammaire est très proche de la modélisation linguistique. Cela est rendu possible par l'utilisation d'un langage de description permettant de définir des alternatives de valeurs pour des champs ordonnés linéairement. En d'autres termes, l'utilisation d'un langage de description configurable (ici, par assemblage modulaire) permet d'avoir une métagrammaire reflétant la modélisation linguistique et constituant de ce fait en elle-même une documentation de la ressource (dans notre exemple, elle contient une motivation à la structure interne des noms).

Le lexique décrit ici permet de générer quelques formes fléchies par racine nominale. La description peut paraître verbeuse, mais l'ajout de nouvelles racines étant limité à la classe *RN*, l'extension de la ressource est grandement facilitée. On peut ainsi passer rapidement à une échelle supérieure et vérifier empiriquement la capacité du modèle linguistique à prédire la structure des noms en ikota. De plus, le lexique ainsi généré, dont les formes fléchies sont étiquettées avec des traits morpho-syntaxiques, pourrait ensuite servir au développement d'outils comme par exemple un analyseur morphologique.

4.2 Décrire une grammaire du são-tomense

Le são-tomense (appelé aussi forro) est un créole du portugais parlé sur l'île de São-Tomé. Comme de nombreux créoles, le são-tomense contient des marqueurs pré-verbaux exprimant le temps, l'aspect et le mode (marqueurs TMA). Ces marqueurs ne sont présents que dans certains ordres. Notre modélisation linguistique correspond aux travaux de Schang *et al.* (2012), et traite ces marqueurs comme des projections verbales.

Nous allons ainsi décrire des règles grammaticales pour les verbes en são-tomense prenant la forme d'arbres élémentaires, et contenant des nœuds pour les différentes combinaisons de marqueurs autorisées. Un exemple de combinaison autorisée est donné en Figure 2.

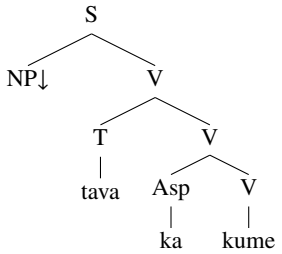


FIGURE 2 – Arbre élémentaire pour le verbe *manger* au passé progressif

Pour représenter les différents marqueurs, nous définissons des classes métagrammaticales (descriptions de fragments d'arbres) pour chacun d'eux (cf fragments (b) et (c) en Figure 3). Afin de contrôler les combinaisons entre ces fragments, nous étiquettons les nœuds V d'un trait proj(ection) restreignant les valeurs autorisées. On ajoute de plus un fragment pour le sujet canonique (cf fragment (a)) et pour la racine verbale (cf fragment (d)). Le compilateur va ensuite chercher les combinaisons autorisées de ces fragments (c'est-à-dire chercher tous les modèles d'arbres contenant les fragments en question) et ne conserver que l'arbre souhaité.

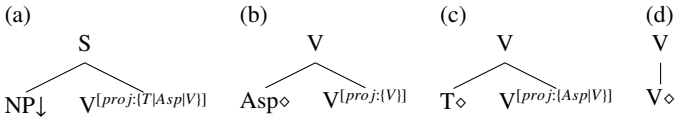


FIGURE 3 – Classes métagrammaticales pour les marqueurs TMA en são-tomense.

Pour vérifier cette modélisation, et produire une grammaire exemple du são-tomense, nous pouvons utiliser le langage de description `synsem` inclus dans XMG2 (assemblage des briques B1 et B2 vues précédemment). Ce langage permet de décrire des arbres syntaxiques dont les nœuds contiennent des structures de traits. Par exemple, le fragment d'arbre (a) s'écrit comme suit dans le langage `synsem` :

```
class SujetCan
export ?X ?Y ?Z
declare ?X ?Y ?Z
{ <syn>{
        node ?X [cat = s, proj = v]{
                node ?Y (mark=subst)[cat = np]     node ?Z [cat = v, proj = @{t,asp,v}]
                }
        }
}
```

L'expression `node ?X { node ?Y }` représente une formule de logique de description d'arbre contenant deux variables de nœuds identifiées par `?X` et `?Y`, et telles que le nœud dénoté par la variable `?X` domine directement celui dénoté par la variable `?Y`. L'expression `node ?X node ?Y` représente deux noeuds frères ordonnés (précédence). Les nœuds ainsi dénotés contiennent en outre une catégorie syntaxique et un trait proj dont la valeur peut être une disjonction (symbole `@`) de valeurs constantes. Les autres fragments sont définis de manière similaire. Une classe `Intransitif` est finalement définie comme la conjonction des classes `SujetCan`, `Aspect`, `Temps` et `RV`. On indique alors au compilateur qu'on souhaite évaluer la classe `Intransitif` et ce dernier produira le (ou les) arbre(s) correspondant(s) au format XML.

Cette illustration de l'utilisation d'un langage de description pour décrire une grammaire d'arbres est limitée. On peut là aussi se demander si écrire 4 classes pour produire un arbre est un gain. L'idée est que la description peut être facilement étendue et que les fragments peuvent être réutilisés dans divers contextes. On a ainsi un gain qui augmente au cours du développement de la ressource (et la maintenance est facilitée puisque l'information y est factorisée). Dans le cas du français, la métagrammaire de Crabbé *et al.* (2013) décrit 6000 schémas d'arbre non lexicalisés à partir de 293 classes. Comme l'ont de plus montré Crabbé *et al.* (2013), le développement d'une grammaire couvrante s'accompagne d'une méthodologie de conception. En suivant cette méthodologie, la métagrammaire est structurée hiérarchiquement, et cette hiérarchie (qui capture des généralisations linguistiques) peut être vue comme une documentation de la syntaxe de la langue décrite.

5 Conclusion et travaux futurs

Dans cet article, nous avons présenté comment les langages de description peuvent être utilisés pour développer des ressources linguistiques à faible coût, s'avérant ainsi particulièrement utiles pour tester des modélisation linguistiques et construire des ressources noyaux pour des langues peu dotées. Nous avons également vu comment le système XMG2 permet une définition modulaire de langages par assemblage de briques élémentaires. Les compilateurs pour ces langages sont alors générés automatiquement et utilisables directement pour des tâches de descriptions linguistiques. Enfin, nous avons illustré concrètement l'utilisation de langages de description pour produire deux ressources exemples pour deux langues sous-dotées, l'ikota et le são-tomense.

Les travaux autour des métagrammaires ne sont pas nouveaux (des métagrammaires pour grammaires d'arbres du français, de l'anglais et de l'allemand existent depuis près d'une dizaine d'années), cependant les avancées récentes autour de la définition modulaire de langages de description ouvre la voie à des utilisations personnalisables en fonction de la langue et de la modélisation linguistique choisies. Des travaux sont en cours pour décrire d'autres langues (telles que l'arabe ou le guadeloupéen), et d'autres niveaux linguistiques (sémantique notamment). Cela passe par la définition de nouvelles briques, et aussi de procédés de résolution des descriptions pour produire les unités voulues.

Références

Abeillé A., Candito M. & Kinyon A. (1999). FTAG : current status and parsing scheme. In *Proceedings of Vextal '99*, p. 283–292, Venice, Italy.

Bos J. (1995). Predicate Logic Unplugged. In *Proceedings of the tenth Amsterdam Colloquium, Amsterdam*.

Candito M. (1999). *Organisation Modulaire et Paramétrable de Grammaires Electroniques Lexicalisées*. PhD thesis, Université Paris 7.

Chomsky N. (1957). *Syntactic Structures*. The Hague : Mouton.

Crabbé B., Duchier D., Gardent C., Le Roux J. & Parmentier Y. (2013). XMG : eXtensible MetaGrammar. *Computational Linguistics*, **39**(3), 591–629.

Duchier D., Magnana Ekoukou B., Parmentier Y., Petitjean S. & Schang E. (2012). Décrire la morphologie des verbes en ikota au moyen d'une métagrammaire. In *19e conférence sur le Traitement Automatique des Langues Naturelles (TALN 2012) - Atelier sur le traitement automatique des langues africaines (TALAf 2012)*, p. 97–106, Grenoble, France.

Evans R. & Gazdar G. (1996). DATR : a language for lexical knowledge representation. *Computational Linguistics*, **22**(2), 167–216.

Joshi A. K., Levy L. S. & Takahashi M. (1975). Tree Adjunct Grammars. *Journal of the Computer and System Sciences*, **10**, 136–163.

Lichte T. & Petitjean S. (2015). Implementing semantic frames as typed feature structures with XMG. *Journal of Language Modelling*, **3**(1), 185–228.

Magnana Ekoukou B. (2015). *Description de l'Ikota (B25), langue bantu du Gabon. Implémentation de la morphosyntaxe et de la syntaxe*. PhD thesis, Université d'Orléans, France.

Petitjean S. (2014). *Génération Modulaire de Grammaires Formelles*. PhD thesis, Université d'Orléans, France.

Petitjean S., Duchier D. & Parmentier Y. (2016). XMG2 : Describing Description Languages. In *Logical Aspects of Computational Linguistics (LACL 2016)*, volume 10054 of *Lecture Notes in Computer Science*, p. 255–272, Nancy, France : Springer-Verlag.

Prolo C. A. (2002). Generating the XTAG English Grammar Using Metarules. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING'2002)*, p. 814–820, Taipei, Taiwan.

Rogers J. & Vijay-Shanker K. (1994). Obtaining trees from their descriptions : An application to tree-adjoining grammars. *Computational Intelligence, 10 :401–421*.

Sagot B., Clément L., De La Clergerie É. & Boullier P. (2006). The Lefff 2 syntactic lexicon for French : architecture, acquisition, use. In *LREC 06*, p. 1–4, Gênes, Italy.

Schang E., Duchier D., Magnana Ekoukou B., Parmentier Y. & Petitjean S. (2012). Describing São Tomense Using a Tree-Adjoining Meta-Grammar. In *11th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+11)*, p. 82–89, Paris, France.

Shieber S. M. (1984). The design of a computer language for linguistic information. *COLING-84*, p. 362–366.

Villemonte De La Clergerie É. (2010). Building factorized TAGs with meta-grammars. In *TAG+10*, p. 111–118, New Haven, CO, United States.

Xia F. (2001). *Automatic Grammar Generation from two Different Perspectives*. PhD thesis, University of Pennsylvania.