# Grammatical development with XMG

anonymous

anonymous
anonymous@anonymous.org

**Abstract.** This paper is dedicated to the compact representation of Tree Adjoining Grammars, we provide a methodology for grammatical development with eXtensible Metagrammar Generator (XMG). The provided methodology has been set up together with the development of large French TAG. Further the grammatical representation language and the assorted development methodology presented can be reused for grammatical development with other strongly lexicalised syntactic formalisms.

XMG is an interpreter for a language of grammatical representation designed in the spirit of PATR II. It has been specifically designed to ease the practical development of strongly lexicalised grammars. A first presentation of the language is given by [1] and a description of the assorted interpreter is given by [2].

Here, we are concerned with providing a grammatical development methodology for Tree Adjoining Grammars (TAG [3]) using XMG. This methodology has been applied to the actual development of an open source French TAG. We begin by motivating the language used: section 1 recalls the motivation of lexicalised syntactic formalisms and section 2 indicates the kind of generalisations we want to capture in grammatical descriptions. Provided that, section 3 introduces the actual language used for grammatical description. Section 4 provides a methodology for large grammatical development with that language.

## 1 Motivation : generalisations in the lexicon

For the representation of natural languages, TAG is used in its lexicalised version (LTAG). In a lexicalised TAG, each grammatical unit is an elementary tree anchored by at least one word, the anchor. An anchor is a leaf node of an elementary tree.

The motivations for lexicalising syntactic formalisms are various. We provide some of them according to two different point of views. On linguistic grounds, it has been already identified by [4] that transformations of generative grammar are subject to lexical exceptions. Further strongly lexicalised formalisms as TAG allow a convenient representation of multi-word expressions [5]. Multi-words expressions cannot be easily handled in phrase structure grammar systems such as HPSG [6]. On practical grounds, strong lexicalism allows to improve parsing efficiency. Indeed it allows the parser to select the only subset of the grammar needed to parse a given sentence.

Strictly speaking, in a lexicalised context, a lexicon for a tree adjoining grammar is an enumeration of independently described trees. In Figure 1 we illustrate this by providing some sample entries for the transitive French verb *manger* (*to eat*). Each sample tree is glossed with a sample sentence illustrating the context represented by that tree[1]. These trees represent some sample alternative con-

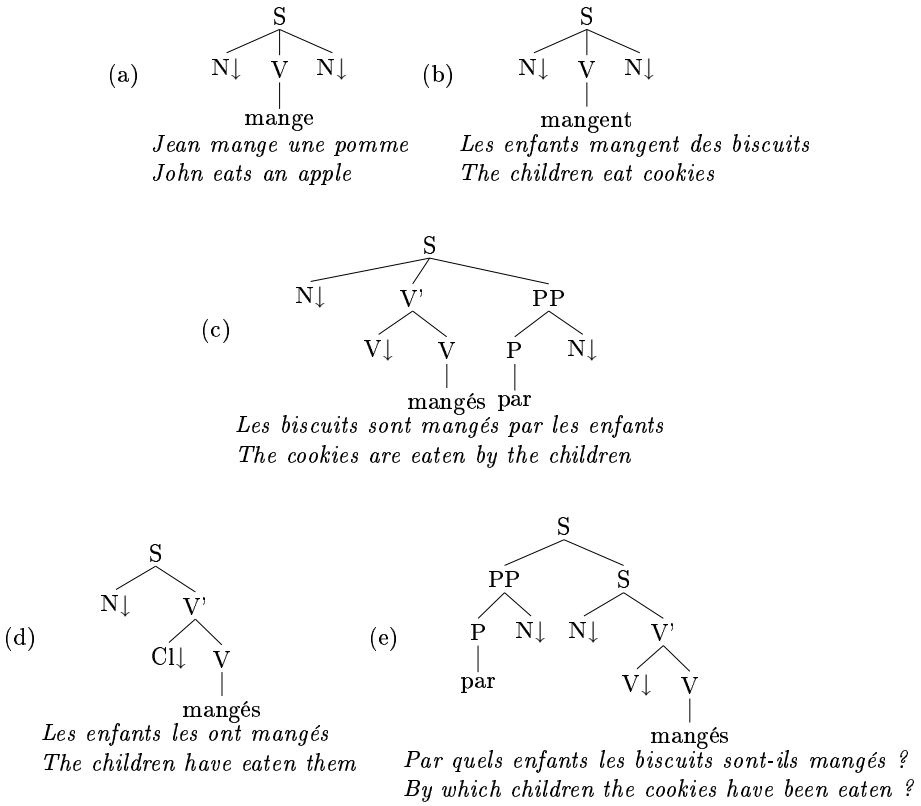
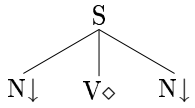



**Fig. 1.** Some alternative entries for transitive *manger* (*to eat*)

texts in which *manger* can occur in French. Trees (a, b) illustrate the need of two different elementary trees for handling morphological alternatives, (c) illustrates a diathesis alternate realisation (passive) of the predicate and finally (d,e) illustrate alternative realisations of the predicate's arguments : cliticisation of the object (d) and a questioned realisation of the By Object of the passive variant (e).

[1] In the grammar we have implemented there are actually 153 trees related to the transitive verb *manger*.

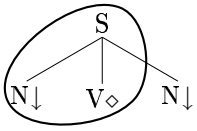
In the remaining of the paper we will consider only tree schemata as the units of interest, as it usually the case in TAG implementations (see e.g. [7]). A tree schema is an elementary tree where the lexical item (the anchor) is left underspecified (we use the $\diamond$ notation to indicate this). Actual elementary trees are generated on the fly by the parser. Such tree schemata are grammatical units among which we can identify generalisations. Using these generalisations, we shall show how to take advantage of them to ease the development of a computational grammar.

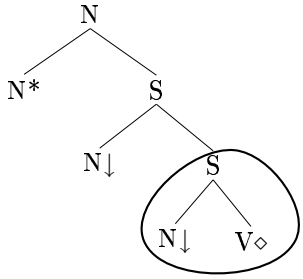



## 2 Two kinds of grammatical generalisations

Methodologically we want to capture two kinds of generalisations : structure sharing on the one side and alternatives on the second side.

*Structure sharing* Structure sharing is usually well understood. These generalisations aim at factoring out the structure considered. The two following trees illustrate this:



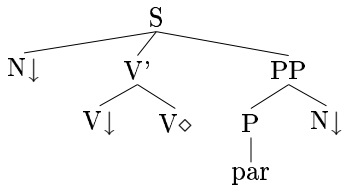




*Jean mange une pomme*
*John eats an apple*

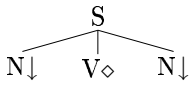
*La pomme que Jean mange*
*The apple John eats*

Where the encircled subtrees represent the realisation of a subject, which is a common substructure shared by both units.

*Alternatives* Alternatives on the other side is a specificity of the lexicon which has been sometimes ignored or left unclear in the recent TAG literature [8, 9]. The active/passive is an instance of such alternative generalisations. Historically, the expression of such alternatives in the lexicon is initially due to [10] who argues in favor of the expression of a counterpart to some transformations of the Generative Grammar in the lexicon. In a language of grammatical representation we have to capture the fact that the two following trees are active/passive alternatives of a same predicate argument structure without focusing whether these trees actually share some common substructure.




*by passive tree*




*active tree*

The two methodological requirements we put forward are reflected in lexical rule based systems. [11] has in some respects ported the proposal of [12] to TAG. He uses an inheritance hierarchy to express structure sharing and lexical rules (called metarules) to express alternatives.

It worth be noticed that alternatives have a special status in that these generalisations contribute to describe *sets* of related grammatical units (such as an active-passive alternative). For instance an ACTIVETOPASSIVE lexical rule outputs a new passive tree provided a base active tree, which makes a set of two trees. Alternatives are an important point in grammatical representation: indeed a tree family in a TAG is a set of trees that represent alternative realisations of a given predicate argument structure.

## 3 The language and its intuition

The language of grammatical description presented in this section aims at taking advantage of the two kinds of generalisations we have identified to ease grammatical development[2].

Its original motivation is to provide an alternative language of grammatical representation to a lexical rule based one. This comes from the fact that we do not want to use lexical rules in grammatical development for TAG in order to avoid rule sequencing problems. The sequencing problems are increased in TAG since the number of lexical rules required is more important than in a phrase structure grammar lexical description. Indeed since TAG has no independent context-free rules, their counterparts are expressed in the lexicon.

The language crucially relies on the notion of *class* (or template). A class is a *description* of partial grammatical structures. With TAG such a description is a tree description possibly augmented with feature structures[3]. Using classes allows to give a name to such a description that can be reused to stand for this description in any grammatical description. These classes allow us to capture grammatical generalisations.

The descriptions that can be expressed in a class follow the abstract syntax given here:

$$Class \quad ::= \quad Name \quad \rightarrow \quad Goal$$

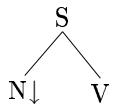
$$Goal \quad ::= \quad \phi \quad | \quad Name \quad | \quad Goal \vee Goal \quad | \quad Goal \wedge Goal$$

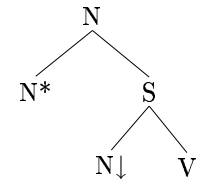
[2] A formal presentation of the language is detailed in [1].

[3] We do not detail the use of feature structures in this paper.

Essentially, it says that a class associates a name to a description, called the Goal. The Goal is either:

- An actual description of a grammatical structure ($\phi$). Using TAG, we associate a name to a tree description. The following classes exemplify this:

(1) a. CanonicalSubject → 


b. RelativisedSubject → 


c. VerbForm → 


The tree descriptions are expressed in a common tree description language detailed by [1]. We use common graphical notations to represent the formulae of the tree description language[4]. A tree description is interpreted as a finite first order tree.

- A name of a class otherwise defined. Thus in the following description, the class ActiveVerbForm reuses the description associated to the class VerbForm:
  (2) ActiveVerbForm → VerbForm
- A disjunction or choice of descriptions. The following class associates the name Subject to an alternative: a subject is either (in this example) a canonical or a relativised subject
  (3) Subject → CanonicalSubject ∨ RelativisedSubject
- A conjunction of descriptions. Finally the language allows to express conjunctions of descriptions. A conjunction of description is understood as the conjunction of two tree descriptions where node names of each conjunct are renamed.
  (4) IntransitiveVerb → Subject ∧ ActiveVerbForm

*Interpretation of the language* This language of grammatical description is interpreted by [1] as a logic program of the DCG paradigm [13] where the terminals of the language are tree descriptions and where conjunction of descriptions is a counterpart for concatenation. The language of the DCG is a tree language, that of the grammatical units of the grammar. We further require the number

[4] Straight lines indicate immediate dominance. Linear precedence is indicated by the symbol $\prec^+$ and adjacency or immediate precedence is not indicated by any special symbol. Trees are decorated with node labels. We let implicit node names who play no significative role in our discussion.

of grammatical units to be a finite set. Therefore the DCG is restricted to be non recursive (directly or indirectly). Given an axiom, an interpreter for this language generates the whole language of the grammar[5].

In the context of our current example, given the axiom IntransitiveVerb, an interpreter outputs the results as depicted in Figure 2. This picture shows on the left the conjunction of descriptions used to output the models depicted on the right of the arrow. In other words the interpreter builds a *set* of trees, each of them is an alternative realisation of an Intransitive context.

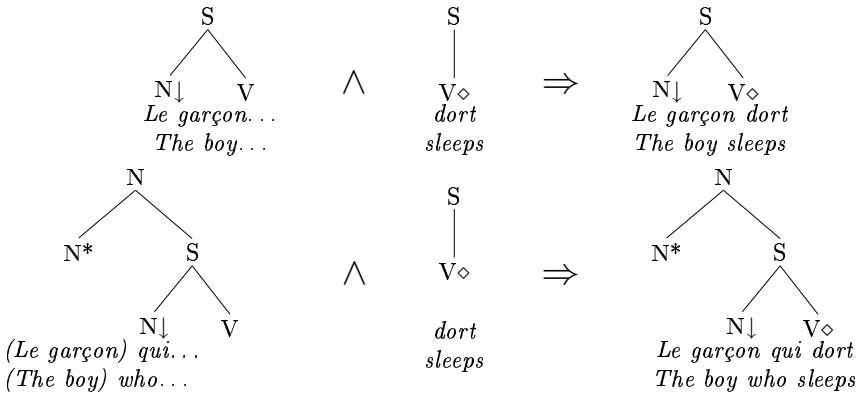


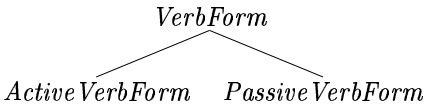

**Fig. 2.** Tree Generation for IntransitiveVerb

As illustrated in figure 2, conjunction is regarded as a conjunction of tree descriptions yielding a new description whose class of admissible models is a class of finite first order trees, the minimal models [1].

*Inheritance hierarchies* After [12], works on grammatical organisation often rely on inheritance hierarchies. Though this notion is not central to our language, we can easily interpret classes as being organized in inheritance hierarchies. We illustrate this with the following example

(5) a. ActiveVerbForm $\rightarrow$ VerbForm $\wedge \phi$
b. PassiveVerbForm $\rightarrow$ VerbForm $\wedge \psi$

where a class $C_1$ whose description uses a class named $C_2$ is interpreted as inheriting information from $C_2$[6]. Thus both ActiveVerbForm and PassiveVerbForm are interpreted as inheriting information from VerbForm. That may be graphically represented




[5] [2] describe an actual implementation of such an interpreter. The interpretation of the grammatical description language as a logic program enables them to implement it following the model of a PROLOG interpreter, a Warren Abstract Machine [14].

[6] $\phi$ and $\psi$ stand for the additional description specified in the inheriting class which worth not be detailed here.

as depicted on the right. It worth be noticed that this inheritance notation should not be confused with HPSG type-inheritance relations. The latter are part of the HPSG theory where ours are just a graphical notation allowing to illustrate the organisation of classes playing much the same role as macros.

*Name spaces* Finally, we have also found convenient to allow an explicit management of name spaces. Each class defines its own name space: each identifier (e.g. a node identifier in a tree description) is local to the class where it is declared. Nonetheless, in the context of inheritance, we have found convenient to allow a given class to explicitly import the name space explicitly exported by its superclass.

## 4 Overall methodology

In this section we provide the methodology we used for the development of the verbal part of a French TAG grammar. This methodology takes most of its inspiration from the three dimensional methodology given by [8, 9]. The description works by describing fragments of trees. A fragment represents either a realisation of an argument of the predicate either the realisation of the predicate itself. Intuitively the trees are described by combining one or more fragments representing arguments with a fragment representing the predicate form. This generalises the preliminary example given in section 3 where the classes CanonicalSubject and RelativisedSubject describe realisation of an argument and where the class VerbForm describes the realisation of the predicate itself.

We work with four levels of abstractions: a first one aims at defining and organising classes describing tree fragments (Section 4.1). A second one aims at grouping descriptions into syntactic functions (Section 4.2). A third one aims at describing verbal diathesis alternatives (Section 4.3) while the fourth one aims at capturing the notion of tree family (Section 4.4).

In the remaining of this section, we illustrate this methodology by providing some sample classes that will allow us to generate some representative alternatives of a ditransitive family. Finally we end up this section by reporting the results of scaling up the given methodology in the context of development of a large French grammar.

### 4.1 Tree fragments

Methodologically, this level of abstraction is only concerned with capturing structure sharing. The building blocks of the grammatical description are tree fragments. We mainly associate names to tree descriptions and organize them within an inheritance hierarchy. Figure 3 provide sample classes describing fragments. These fragments represent different possible constructions of French verbal dependants in a TAG.

To further factorise information we organise the fragments in an inheritance hierarchy. Figure 4 provides a graphical representation of this hierarchy following

the conventions introduced in section 3. This hierarchy illustrates that verbal arguments described in Figure 3 break in four categories. First, the canonical complements are those arguments realised after the verb. The canonical object is a noun and the prepositional complements are introduced by specific prepositions *à* for the canonical indirect object and *par* for the canonical by object. Second the canonical subject is a noun realised in front of the verb. Third, Wh arguments (or questioned arguments) are realised in front of a sentence headed by a verb and may possibly be realised at an unbounded distance of the predicate. Wh object is an extracted noun and questioned prepositional objects are extracted prepositional phrases that are introduced by a specific preposition. Fourth the relativised subject represents a relative pronoun realised in front of the sentence. Extracted subjects in French cannot be realised at an unbounded distance of the predicate.

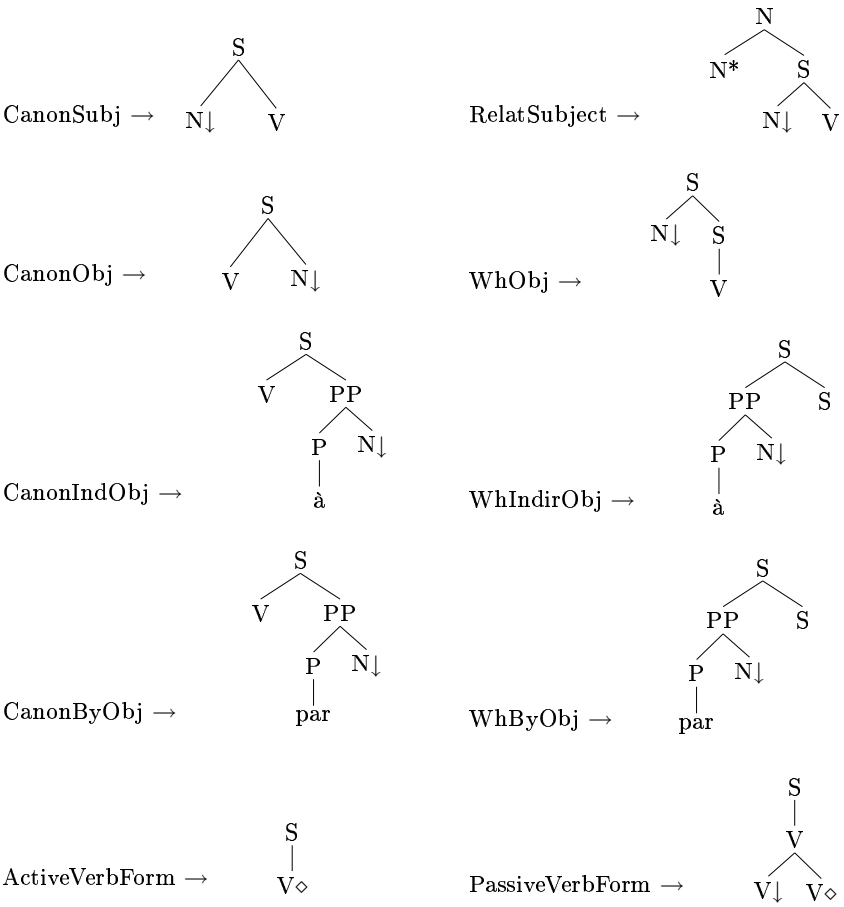



**Fig. 3.** Elementary tree fragments used as building blocks of the grammar

Finally, classes in the hierarchy are specified in a way that each class has access to the identifiers declared by its immediate or non immediate superclasses.

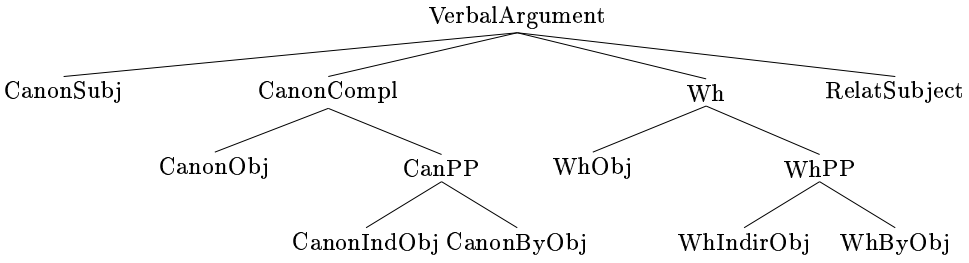



**Fig. 4.** Organisation of elementary fragments in an inheritance hierarchy

### 4.2 Syntactic functions

This second level of abstraction is mainly concerned with capturing alternatives. We want here to capture generalisations that are syntactic functions. To do this, we take advantage of the tree fragments described in the previous section. We can reuse them by manipulating the name of the classes where they are defined.

Syntactic functions are understood here as notions that allow to caracterise verbal dependants by abstracting over the problem of word ordering. In this line of idea, each syntactic function is a name associated to a set of alternative realisations in syntax as it is illustrated by the classes defined below:

(6) a. Subject → CanonicalSubject ∨ RelatSubject
b. Object → CanonicalObject ∨ WhObject
c. ByObject → CanonicalByObject ∨ WhByObject
d. IndirectObject → CanonicalIndirObject ∨ WhIndirObject

We define here a subject as an abstraction for talking about a dependant that can be either realised in front of the verb in a canonical position (represented here by the class CanonicalSubject) either in front of the verb as a relative pronoun that cannot be realised at an unbounded distance of the predicate (class RelatSubject). Thus, the subject class we have provided here allows to caracterise contexts such as these:

(7) a. **Jean** mange (canonical subject)
**John** eats
b. Le garçon **qui** mange (relativised subject)
The boy **who** eats

As a matter of illustration, the indirect object (class IndirectObject) is a function that abstracts over the realisation of an argument introduced by the preposition *à* at the right of the verb (CanonicalIndirObject) or realised in extracted position (possibly realised at an unbounded distance from the predicate) as illustrated by the following examples:

(8) a. Jean parle **à Marie** (canonical indirect object)
John talks **to Mary**
b. **A qui** Jean parle-t-il ? (wh indirect object)
**To whom** is John talking ?
c. **A qui** Pierre croit-il que Jean parle ? (wh indirect object)
**To whom** Peter thinks that John talks ?

To sum up, the overall methodology we used for describing syntactic functions is compatible with the informal classification of French syntactic functions made by [15]. Each syntactic function is associated to a set of possible syntactic constructions. The set or system of syntactic functions for French is then defined in a way that there are no two syntactic functions associated to the same set of constructions.

### 4.3 Diathesis alternations

In this third level, we take advantage of the abstractions captured so far to represent diathesis alternations. Again we are interested here in describing alternatives. Diathesis alternations are those alternations of mapping between arguments and syntactic functions, as for instance the active/passive alternation. In a diathesis alternation the actual form of the verb constrains the way arguments of the predicate are realised in syntax. Thus, in the following example,

(9) a. Jean **envoie** une lettre
John sends a letter
b. Une lettre **est envoyée** par Jean
A letter is sent by John

It is considered that both (9a) and (9b) are alternative realisations of a predicate argument structure such as *send(John, a letter)*. Diathesis alternations capture the fact that if the verb is at the active form, then the first predicative argument, *John*, behaves as a subject and the second predicative argument,*a letter*, behaves as an object. If the verb is in the passive form then the first predicative argument, *John*, behaves as a by Object (or agentive complement) and the second predicative argument, *a letter*, behaves as the subject. We can represent these facts in our language by defining the following class:

(10) TransitiveAlternation $\rightarrow$
Subject $\wedge$ ActiveVerbForm $\wedge$ Object)
$\vee$(Subject $\wedge$ PassiveVerbForm $\wedge$ ByObject)

Finally let us note that a traditional case of "erasing"[7], such as the agentless passive (or passive without agent) can be expressed in our language by adding

[7] It is often argued that a language of grammatical representation must be equipped with an "erasing device" like lexical rules because of phenomena such as the passive without agent. In this framework it turns out that this kind of device is not needed since we do not grant any special status to base trees. There is no notion of ordering the derivation of trees by successively applying lexical rules. Instead the whole set of trees is described as a whole.

an additional alternative where the By Objet or agentive complement is not expressed. Thus (11) is an augmentation of (10) where we have added the agentless passive alternative (indicated in bold face).

(11) TransitiveAlternation →
(Subject ∧ ActiveVerbForm ∧ Object)
∨(Subject ∧ PassiveVerbForm ∧ ByObject)
∨**(Subject ∧ PassiveVerbForm)**

This methodology can be further augmented to implement an actual linking in the line of [16]. For the so-called erasing cases, one can map the "erased" predicative argument to an empty realisation in syntax. We refer the reader to [17] for further details.

### 4.4 Tree families

Finally, we can capture tree families. In the TAG literature, a tree family is a set of trees that represent alternative realisations of a predicate argument structure. In the line of the example we have developped so far, we can easily describe a family of trees representing a ditransitive context. More precisely we can define a family where trees are variant realisations of a predicate argument structure with a nominal subject, a nominal object and an indirect nominal object as follows:

(12) DitransitiveFamily → TransitiveDiathesis ∧ Indirectobject

The trees generated for such a family will, among others, handle contexts such as these:

(13) a. Jean offre des fleurs à Marie
John offers flowers to Mary
b. A quelle fille Jean offre-t-il des fleurs ?
To which girl does John offer flowers ?
c. Le garçon qui offre des fleurs à Marie
The boy who offers flowers to Mary
d. Quelles fleurs le garçon offre-t-il à Marie ?
Which flowers does the boy offer to Mary ?
e. Les fleurs sont offertes par Jean à Marie
The flowers are offered by John to Mary
f. Par quel garçon les fleurs sont-elles offertes à Marie ?
By which boy the flowers are offered to Mary

All these sentences are regarded as alternatives of the prototypical sentence (13a). (13b) illustrates an alternative (questioned) realisation of the Indirect Object, (13c) and (13d) respectively illustrate the realisation of a relativised subject and of a questioned object.(13e) illustrates that the family handles passive alternatives thanks to the use of the TransitiveDiathesis class, and finally

we show with (13f) that the agentive complement of the passive may be realised as well in a questioned position.

It is straightforward to extend the grammar by introducing new families. For instance, one can introduce a transitive family, that is a set of trees that are alternatives of predicate with a nominal subject and of a nominal object with the class definition (14b) or an intransitive family (alternatives of a predicate with a nominal subject) with the definition (14a).

(14) a. IntransitiveFamily $\rightarrow$ Subject $\wedge$ ActiveVerbForm
b. TransitiveFamily $\rightarrow$ TransitiveDiathesis

*Grammatical generation* The last step of our presentation concerns the generation of an actual grammar with a given grammatical description. Recall that generating the grammar is understood as the generation of the set of words of the language of a grammar of the DCG paradigm. Each of theses words being a conjunction of tree descriptions licensed by the grammatical description.

To do this, we have to provide an axiom (or a goal) to the DCG. In practice we have found convenient to allow the user to explicitly define the axioms he want to be generated by the interpreter. Following the example developped so far, the user can ask the interpreter to generate the three families described before.

(15) a. value IntransitiveFamily
b. value TransitiveFamily
c. value DitransitiveFamily

The interest of this explicit valuation relies on the possibility to generate either the total grammar being described by valuating every family or subgrammars by valuating only some of the families. These subgrammars can be used either for testing or checking the correctness of the resulting trees on smaller sets, either to generate grammars for applications where a large grammar is not needed.

*Constraining the generated models* The units manipulated in grammatical description are tree descriptions. Tree descriptions are expressed in a language whose formulae are interpreted as finite trees, the minimal models (See [1] for further details). During grammatical development however we have found convenient to let the user further constraining the class of admissible models. This is achieved by using both properties and principles [1].

User definable properties may be attached to nodes in the tree descriptions. These properties may then be used as parameters of predefined principles constraining the generated models. We illustrate the use of properties and principles in the context of our running example. Till now according to our methodology, the families defined in (14b) and (12) allow the generation of trees with multiple extractions. Although some rare cases of multiple extractions are known to happen in French[18]. It is generally preferable to rule them out of grammatical implementations.

To do this, we introduce a property *extracted* used as a parameter of a principle of unicity. The principle of unicity requires that in a valid model there may be at most one node associated to the property specified as parameter. To rule out double extraction we have associated that property to a node in fragments representing extraction and intanciated the principle of unicity with the parameter *extracted*.

Figure 5 illustrates this. The ditransitive family allows to generate the conjunction of fragments illustrated. From left to right the fragments represent a canonical subject, a questioned object, a questioned indirect object and an active verb form. We have indicated the extracted property with the subscript $E$. No model can be generated given these descriptions since two different nodes cannot be associated to the property.

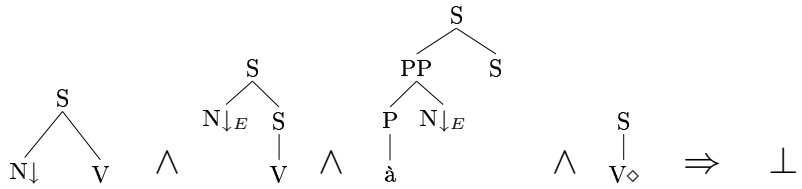



**Fig. 5.** Unicity of extraction

Beyond our working example, in the development of of a larger French grammar we also have used unicity for constraining French clitic unicity, and for constraining functional unicity. Besides unicity, we have used a coloration principle described in [1], a principle of ordering between sibling nodes in order to achieve French clitic ordering. And finally we have introduced an island principle that accounts for designing elementary trees that are compatible with the expression of islands constraints in TAG[8].

*Towards a realistic French Grammar* It should be clear that the example we have given till here remains quite simplified. The language and the methodology provided here has been successfully reused to extend the grammatical description for handling a significative amount of phenomena related to the syntax of French verbs and valency controlled verbal dependants. In short the actual grammar we have implemented uses Feature Based Tag and is close to the one described by [18, 8]. The grammatical description puts the focus on the syntax of French Verb and French verbal dependants. We have also added descriptions that handle the syntax of French predicative adjectives. The possible constructions described for

[8] In TAG, following [18, 7] setting an S node to substitution blocks extraction out of the constituent dominated by that S node, setting an S node to foot allows the extraction out of the dominated constituent. The island principle we use allows to account for wh-islands: if an elementary tree contains both a node marked as extracted and a leaf node $n$ being sentential, than the node $n$ is marked as substitution otherwise it is marked as foot.

verbal dependants are the canonical, clitic, cleft, questioned and relativised arguments. Subject inversion and some cases of ellipsis (such as NP-equi deletion) have been also described. Regarding syntactic functions we have described the nominal subject, nominal object, nominal indirect object, nominal genitive, nominal obliques and nominal locatives. Sentential arguments have been described as well : sentential subjects and sentential objects both finite and non finite are described with different possible complementizers. Sentential questions have also been described. The following diathesis alternations have been taken into account : active, passive, impersonal passive, agentless passive, impersonal and middle. There is at the time of this writing 44 families described. All these families implements subject verb agreement, past participle agreement both with the extracted object and with the subject, and subject control agreement. Islands constraints have been dealt with following TAG traditional usage. Additional descriptions have been added to handle non verbal specific constructions. Thus the grammar contains some basic support for the syntax of noun phrases, constituent coordination, various kinds of adverbial modifiers and so forth. The actual grammar and some documentation for interfacing it with a lexicon is available in open source by anonymous CVS via the web site `http://sourcesup.cru.fr/XMG`.

*Evaluation* A preliminary test of the grammar has been realised by some of our colleagues with their participation in the EASY-EVALDA shallow parsing evaluation campaign. This campaign has been led in France in 2004 and involved parsing of unrestricted corpora. The results are unknown at the time we are writing. However we do not expect much results from this since the grammar we have developped is a *competence* grammar designed to recognize only grammatical sentences. It is not appropriate as it is for parsing unrestricted text (including sentences from oral dialogue and mail corpora). A more serious evaluation has still to be led against a test suite dedicated to parsing. Nevertheless, concerning evaluation, we face two severe problems that are general to the French community. First, we miss an available test suite for French. TSNLP is not anymore publicly available. Second, we miss a publicly available lexical database describing the subcategorisation of French verbs. Current efforts are now to be driven in this way.

## 5 Conclusion and Perspectives

This paper provides a practical argument showing that a simple language made of abstractions representing descriptions and the ability to reuse them using conjunction and disjunction is enough for describing a full sized computational grammar. We can handle alternatives with disjunction instead of using lexical rules.

Regarding the TAG community, we do not need to use complicated devices for grammatical description such as does [8] or [9]. Since its beginnings [19], the metagrammar was considered as a highly complicated as well as non standard grammatical development framework. In this paper we have shown that

the metagrammar [8, 9, 20] may be reduced to a simple grammatical description language by (1) explicitly intoducing the possibility to express alternatives and (2) by providing a practical argument: that of developping a full scaled grammar with this language. The use of a different language has nevertheless allowed us to take advantage of the grammatical development methodology introduced by [8]. Further, we have found that the language used here for grammatical development is strikingly similar to that used for grammatical development by the LFG community [21] apart that the structures used in the grammar are different : LFG grammars use feature structures where TAG grammars use trees. This can ease comparisons between development methodologies in the two formalisms.

Moreover, the language and the development methodology has contributed to generate a large French grammar released in open source. To our knowledge, it is the first time such a grammar is made publicly available for French.

Finally, it can be shown [22] that the language and the grammatical development methodology is compatible with the effective expression of a syntax-semantics interface for TAGS in the line of [23]. This can be done by adding to the language classes with parameters. The effective implementation of a parser supporting semantics has been achieved by [22]. Yet the actual realisation of a user friendly grammatical development platform remains to be done.

On formal grounds, further investigations are expected to extend the framework and the methodology to a larger family of syntactic formalisms. Preliminary investigations have been successfully led by G. Perrier who has reused the language and its assorted methodology in the development of a French Interaction Grammar[24] where the grammatical structures are D-Trees instead of trees. We are also interested in investigating how to reuse the language and the methodology to describe the lexicon of an eXtensible Dependency Grammar [25] where the elementary grammatical structures used are directed graphs.

On linguistic grounds, further investigations concern the development of alternative grammatical development strategies. Indeed the methodology presented here reflects a traditional — generative and syntactocentric — approach to syntax. An alternative method for implementing the syntax-semantics interface would be to consider driving grammatical development by using lexical classes in the line of [26]. These are thought to carry a more semantically fine-grained approach to the syntax-semantics interface problem.

## References


1. Crabbé, B., Duchier, D.: Metagrammar redux. In: Constraint Solving and Language Processing, Copenhagen (2004)
2. Duchier, D., Leroux, J., Parmentier, Y.: The metagrammar compiler: An nlp application with a multi-paradigm architecture. In: Mozart 2004, Charleroi (2004)
3. Joshi, A.K., Schabès, Y.: Tree adjoining grammars. In Rozenberg, G., Salomaa, A., eds.: Handbook of Formal Languages. Springer Verlag, Berlin (1997)
4. Gross, M.: On the failure of generative grammar. Language **55** (1979) 859–885
5. Abeillé, A.: Lexical and syntactic rules in a tree adjoining grammar. In: Proceedings of the 28th Meeting of the Association for Computational Linguistics. (1990)

6. Sag, I., Baldwin, T., Bond, F., Copestake, A., Flickinger, D.: Multiword expressions: A pain in the neck for nlp. In: Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLING 2002), Mexico City (2002) 1–15
7. XTAG Research Group: A lexicalized tree adjoining grammar for english. Technical Report IRCS-01-03, IRCS, University of Pennsylvania (2001)
8. Candito, M.H.: Organisation Modulaire et Paramétrable de Grammaires Electroniques Lexicalisées. PhD thesis, Université de Paris 7 (1999)
9. Xia, F.: Automatic Grammar Generation from two Different Perspectives. PhD thesis, University of Pennsylvania (2001)
10. Bresnan, J., Kaplan, R.M.: The Mental Representation of Grammatical Relations. The MIT Press, Cambridge MA (1982)
11. Becker, T.: HyTAG: A new Type of Tree Adjoining Grammars for Hybrid Syntactic Representation of Free Word Order Language. PhD thesis, Universität des Saarlandes (1993)
12. Flickinger, D.: Lexical Rules in the Hierarchical Lexicon. PhD thesis, Stanford University (1987)
13. Pereira, F., Warren, D.: Definite clause grammars for language analysis —a survey of the formalism and a comparison to augmented transition networks. Artificial Intelligence **13** (1980) 231–278
14. Aït Kaci, H.: Warren's abstract machine, a tutorial reconstruction. MIT Press (1991)
15. Iordanskaja, L., Mel'Čuk, I.: Towards establishing an inventory of surface-syntactic relations : Valency-controled surface-syntactic dependents of the french verb. ((to appear))
16. Bresnan, J., Zaenen, A.: Deep unaccusitivity in LFG. In Dziwirek, K., Farell, P., Mejias-Bikandi, E., eds.: Grammatical Relations : A Cross-Theoretical Perspective. CSLI, Stanford (1990) 45–57
17. Crabbé, B.: Lexical classes for structuring the lexicon of a TAG. In: Prospects and advances in the Syntax/Semantics interface, Nancy (2003)
18. Abeillé, A.: Une grammaire d'arbres adjoints pour le français. Editions du CNRS, Paris (2002)
19. Candito, M.H.: A principle based hierarchical representation of LTAGs. In: COLING 96, Copenhagen (1996)
20. Gaiffe, B., Crabbé, B., Roussanaly, A.: A new metagrammar compiler. In: Proc. TAG+6, Venise (2002)
21. Dalrymple, M., Kaplan, R., King, T.H.: Lexical structures as generalizations over descriptions. In: LFG 04, Christchurch (2004)
22. Gardent, C., Parmentier, Y.: Large scale semantic construction for tree adjoining grammar. In: Proceedings of LACL 2005, Bordeaux (submitted)
23. Gardent, C., Kallmeyer, L.: Semantic construction in feature-based tree adjoining grammar. In: 10th conference of the European Chapter of the Association for Computational Linguistics. (2003)
24. Perrier, G.: Les grammaires d'interaction. Habilitation à diriger les recherches en informatique (2003)
25. Debusmann, R., Duchier, D., Koller, A., Kuhlmann, M., Smolka, G., Thater, S.: A relational syntax-semantics interface based on dependency grammar. In: Proceedings of the COLING 2004 Conference, Geneva/SUI (2004)
26. Levin, B.: English Verb Classes and Alternations. The University of Chicago Press (1993)