Building an RRG computational grammar[1]

Francisco Cortés Rodríguez
Universidad de La Laguna
España

Ricardo Mairal-Usón
UNED
España

Francisco Cortés Rodríguez: Instituto de Lingüística Andrés Bello, Universidad de La Laguna, España. | Correo electrónico: fcortes@ull.es
Ricardo Mairal-Usón: Departamento de Filologías Extranjeras y sus Lingüísticas, Facultad de Filología, Universidad Nacional de Educación a Distancia (UNED), España. | Correo electrónico: rmairal@flog.uned.es

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

87

Abstract

Several grammatical models have shown a growing interest for the development of the conditions necessary to satisfy the so-called criterion of computational adequacy. Within Role and Reference Grammar (RRG [Van Valin & LaPolla, 1997; Van Valin, 2005; Pavey, 2010]), there have been some works seeking to implement the model in different computational environments (Diedrichsen, 2011, 2013; Guest, 2009; Nolan & Periñán-Pascual, 2014; Salem et al., 2008). In this scenario, the works of Van Valin & Mairal (2014), Periñán-Pascual (2013) and Periñán-Pascual & Arcas (2014) have set up the guidelines to devise a parsing system called ARTEMIS (Automatically Representing TExt Meaning via an Interlingua-Based System) for the computational treatment of the syntax and semantics of sentences.

The goal of this paper is to contribute to the development of ARTEMIS focusing specifically on the design of the rules necessary for the effective computational parsing of unmarked simple clauses following the format of the Layered Structure of the Clause as described in RRG. Such rules should yield, as a result for every sentence, a parsed tree following the format of grammatical analyses in RRG.

Keywords: Role and Reference Grammar; FunGramKB; computational grammar; constructions; lexical rules; syntactic rules; constructional rules; attribute-value matrix; conceptual modeling; the layered structure of the clause.

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

88

1. Preliminaries

Human-machine interaction is one of the most outstanding challenges in the research agenda of different disciplines. A significant number of studies devoted to such an interaction precisely targets at natural language processing (NLP) and in particular at understanding the meaning of a given text by a machine. This endeavor has become a major concern provided that today we are living in the era of information where access to massive amount of data has become a daily routine task. Therefore it comes as no surprise that the development of resources and tools for more efficient information retrieval systems has become one top priority within the field of natural language understanding (NLU).

In this context, linguistics, and linguistic models in particular, cannot be silent to this challenge since what we need is solid explanatory frameworks with labels, tags and analytical tools to understand meaning construction. There are quite a few morphological and syntactic parsers that can provide taggings of a given text. However, to the best of our knowledge, semantic taggers are scarce in number and those available merely provide tags in terms of semantic roles and labels of the type 'agent', 'beneficiary', 'recipient', etc[2]. Therefore, the relevant question is whether it is possible to provide a semantic annotation so that a computer can understand the meaning of a natural language text. We maintain that such an endeavor, though complex in many respects, is possible if performed from the point of view of a functional theory like Role and Reference Grammar (RRG henceforth; Van Valin & LaPolla, 1997; Van Valin, 2005; Pavey, 2010). Periñán-Pascual & Arcas (2014: 167-168) highlight three

features of this grammar which make it suitable for its application in NLP: (a) the semantic and communicative grounding of the grammatical objects (rules and structures) in the model; (b) the fact that it is a monostratal theory, in which semantic and syntactic structures are closely interconnected through a bidirectional linking algorithm; and (c) its typological orientation, which comes as an additional value when dealing with multilingual environments.

Because of the amenability of the model to computational testing, several researchers have recently devoted their work to applying RRG in different computational models. Among them are the following works: Diedrichsen, 2013; Guest, 2009; Nolan & Periñán-Pascual, 2014[3]; Nolan & Salem, 2011; Salem et al., 2008; Van Valin & Mairal, 2014. Within this scenario, one of the most outstanding contributions in the computational modelling of RRG has been the creation and development of FunGramKB (<u>Fun</u>ctional <u>Gram</u>mar <u>K</u>nowledge <u>B</u>ase; Periñán-Pascual, 2013; Periñán-Pascual & Arcas, 2007, 2010; Periñán-Pascual & Mairal, 2009, 2012), a multipurpose lexico-conceptual knowledge base for NLP systems, and more particularly for NLU. FunGramKB includes the following components (see appendix 4):

(a) The lexical component, which is language-specific and consists of two submodules: the lexicon (which includes in the format of entries all the linguistic information related to the lexical units) and the morphicon (which deals with all inflectional processes of a language).

(b) The grammatical level, also language dependent where constructional schemata of a given language are stored[4].

---

2   As a case in point, FrameNet (Boas, 2005; Fillmore et al., 2003a, 2003b; Ruppenhofer et al., 2010), which is a lexical database based on frame semantics, has been used in a number of NLP systems (Shen & Lapata, 2007; Ovchinnikova et al., 2010), although some important shortcomings have been found for its direct application in NLP tasks.

3   This volume includes a good number of interesting proposals involving the application of RRG in computational environments.

4   FunGramKB also incorporates the contributions from constructional grammars, especially from the Lexical Constructional Model (LCM; Mairal & Ruiz de Mendoza, 2008, 2009; Ruiz de Mendoza, 2013; Ruiz de Mendoza & Galera, 2014; Ruiz de Mendoza & Mairal, 2007).

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

89

(c) The conceptual component, which is language independent and stores all deep semantic units and structures into different submodules: the ontology (a hierarchical storehouse for concepts in a human mind), the cognicon (or repository of procedural conceptual schemas or scripts to encode stereotypical actions) and the onomasticon (for real world entities and events).

The fact that FunGramKB is ontologically-based has brought about the enrichment of the system of semantic representations from RRG. The Logical Structures from RRG are replaced by Conceptual Logical Structures (CLSs henceforth; Mairal et al., 2012; Van Valin & Mairal, 2013). CLSs keep as a pillar for semantic representations the Aktionsart characterization of lexical units as encoded in the original Logical Structures, but the primitives are now conceptual units that come from the ontology (they are marked with angle brackets ‹C›; see examples below). Therefore, the CLS involves the interaction of both the ontology and the lexicon[5].

The following examples are helpful to illustrate the shift from the Logical Structures to the CLSs (Periñán-Pascual, 2013: 218):

(1) Peter broke the glass.
Logical structure (RRG):
$\langle_{IF}{}^{DEC} \langle_{TNS}{}^{PAST} \langle_{ASP}{}^{PERF} \langle$[**do'** (Peter, Ø)] CAUSE [BECOME **broken'** (glass)]$\rangle\rangle\rangle\rangle$

CLS (FunGramKB):
$\langle_{IF}{}^{DEC} \langle_{TNS}{}^{PAST} \langle_{ASP}{}^{PERF} \langle_{CONSTR-L1}{}^{KER2} \langle[_{AKT}{}^{ACC}$ [+BREAK_00 (%PETER_00- Theme, \$GLASS_00- Referent)]]$\rangle\rangle\rangle\rangle$

As can be seen, ontological concepts like \$GLASS_00 or +BREAK_00 are now used instead of predicates like *glass* or primitives like **broken'**.

There are other noticeable changes, as is the introduction of constructional operators (CONSTR-L1) marking every argumental construction and Aktionsart operators (AKT) as well (cf. Periñán-Pascual, 2013, and Periñán-Pascual & Arcas, 2014, for a detailed description of these features).

Although the CLS brings a heavier 'conceptual' load into semantic representations, it still needs some refining from a computational perspective. In fact, if we want the NLP system to reach a deeper level of comprehension it is necessary to model CLS representations into COREL (COnceptual REpresentation Language) structures. Thus the CLS in (1) is modeled into a COREL scheme of the following type:

(2) +(e1: +DAMAGE_00 (x1: %PETER_00)Theme (x2: \$GLASS_00)Referent (f1: (e2:+SPLIT_00 (x1) Theme (x2)Referent))Result)
'Peter damaged the glass into pieces'

CLSs and COREL Schemes are ambitious resources which aim at providing the semantic representation of a given text but, prior to this, it is necessary to spell out the syntactic structure of the input text. In other words, we need a resource that can automatically map the syntactic representation to the semantic representation of a given piece of language; that is, the resource must be the computational replica of the syntax-to-semantics linking interface in RRG, with the contributions from the LCM and also taking into account the new deep-conceptualist turn in semantic structures provided by FunGramKB. Such a resource is ARTEMIS (Automatically Representing TExt Meaning via an Interlingua-Based System), a NLP prototype primarily designed for natural language understanding. Periñán-Pascual (2013) and Periñán-Pascual & Arcas (2014) offer the first proof-of-concept prototype of ARTEMIS and provide its basic architecture (cf. section 2).

---

5   The FungramKB NLP Lab, a virtual lab for natural language processing within a functional perspective, provides free access to Navigator, which offers the conceptual and linguistic properties of lexical entries in English: www.fungramkb.com.

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

90

However, apart from some preliminary papers dealing with specific aspects of ARTEMIS[6], a full text providing full coverage of the intricacies of the RRG computational grammar was needed.

Within this framework, the primary aim of this paper is to discuss the format of the computational grammar that forms part of this resource. This grammar could be understood as a computational implementation of the layered structure of the clause (LSC) in RRG. Hence, this paper is organized as follows. Section 2 provides a brief overview of the architecture of ARTEMIS and the way the RRG analysis of simple clauses is treated computationally within this prototype. Section 3 deals with the format of the grammar development environment which means dealing with the format of syntactic and lexical rules. Because of space restrictions, the description will concentrate on the rules necessary for the parsing of the elements belonging to the NUCLEUS layer in the clause structure. Section 4 will illustrate how different syntactic structures within the nucleus are dealt with the new set of rules. In the account of such structures we will also offer the rules necessary for the analysis of the innermost layer in the clause structure, namely the PREDICATE. Finally, some concluding remarks and future lines of research are part of section 5.

2. ARTEMIS
2.1. Overall architecture of ARTEMIS

In its current state, ARTEMIS is a proof-of-concept NLP system designed to transduce a natural language fragment (a sentence) to its morphosyntactic form and, subsequently, to its underlying semantic structure. It is linguistically grounded in RRG and the LCM and deploys FunGramKB to obtain the relevant conceptual units for semantic representations. To a great extent, ARTEMIS can

be considered the computational counterpart of the syntax-to-semantics linking algorithm in RRG; the following is a simplified view of the processing phases of a text within this prototype:

Input text › CLS representation › COREL Scheme → Reasoner › Output text

In order to deal with all the tasks in this process, ARTEMIS comprises the following modules: The Grammar Development Environment (GDE), the CLS Constructor and the COREL-Scheme Builder. Whereas the last two modules are in charge of deriving the semantic representations of sentences, the GDE includes the grammatical rules necessary for the morphosyntactic parsing of natural language expressions; such rules should yield as a result a parsed tree for every sentence, following the principles of grammatical analyses in RRG. The following UML[7] diagram (figure 1) models the behavior of the GDE and the CLS Constructor (Periñán-Pascual & Arcas, 2014: 178).
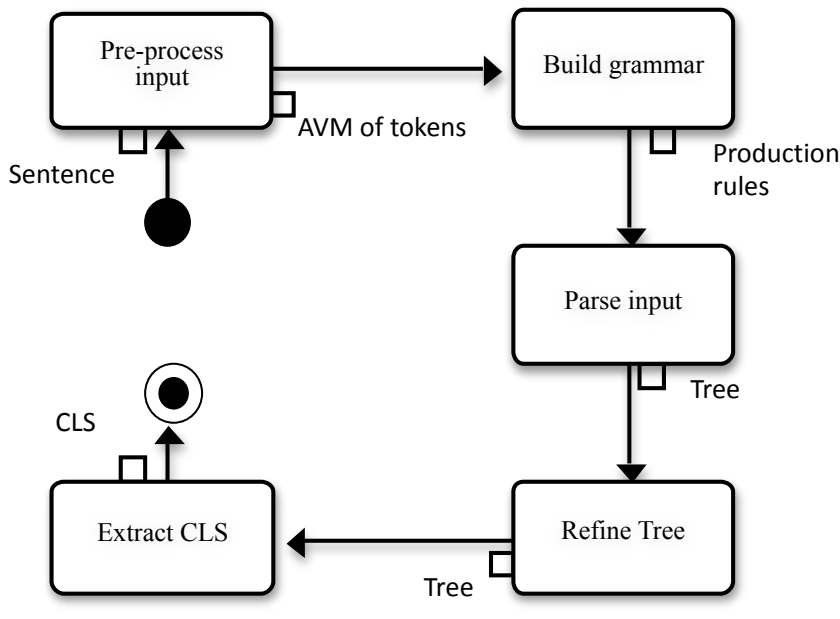
In this routine, there is an initial phase of tokenization intended to split a text into sentences and subsequently into word tokens, which are encoded in Attribute-Value Matrixes (AVMs) (see below section 2.3 for a detailed description). The output of this first phase feeds the Build grammar module, where syntactic, constructional and lexical rules will parse the text and generate a morphosyntactic tree; syntactic rules will provide such a tree in accordance with the RRG layered model for the structure of clauses; constructional and lexical rules will in turn refine such a tree by endowing it with the specific properties of lexical and constructional units. Unlike syntactic rules, which must be pre-defined in the GDE, constructional and lexical rules are constructed automatically in accordance with the tokens from the input stream. Next, there is

---

6   Mairal & Periñán-Pascual (2014), Cortés-Rodríguez (2016) and Díaz-Galán & Fumero (2016) address some more specific issues, i.e. the lexical-grammatical interface and the computational treatment of referential phrases and the auxiliary *do*.
7   UML (Unified Modeling Language; Rumbaugh et al., 1999; Debrauwer & Van der Heyde, 2010) is a general purpose object modeling language to visualize software systems and processes.

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

91

FIGURE 1
The ARTEMIS process (abridged version)

a subsequent tree refinement process in order to relocate, if necessary, some tree nodes and to filter out some node attributes. The last step in the diagram makes reference to the extraction of semantic units for the construction of the CLS.

So far, the GDE consists of two basic types of theoretical constructs, a set of rules that account for syntactic structures and a library of Attribute-Value Matrixes (AVMs) for grammatical units. At the present stage, however, both the library of AVMs and the production rules in the Build grammar stage are still underdeveloped, since in the seminal works where ARTEMIS is described (Periñán-Pascual, 2013, and Periñán-Pascual & Arcas, 2014) the rules proposed are not fully consistent with the functional approach that supports RRG's grammatical analyses for clauses.

Therefore, the goal of our research is to design the syntactic rules necessary for parsing simple clauses, together with the set of lexical rules that will help to obtain the grammatical (syntactic and semantic) information from those lexical tokens that are not stored in the FunGramKB lexicon; i.e. the tokens associated to functional lexical units. This will also lead us to develop the AVMS necessary to encode the grammatical features associated to both those units and the more abstract constituents where they belong in the syntactic structures. Since parsing in the GDE is based on the analysis proposed for clauses within RRG, it is necessary to briefly describe the basic features of syntactic description in this model. Section 2.2 offers such a description.

2.2. The syntax of simple clauses in RRG: The layered structure of the clause[8]

The RRG notion of syntactic clause structure is the so-called Layered Structure of the Clause (LSC) and it is based on two fundamental contrasts:

---

8   This section is an abridged description of the format in which simple clauses are syntactically analyzed in Van Valin & LaPolla (1997: 17-52) and Van Valin (2005: 3-20).

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

92

(1) Between Predicating and Non-predicating elements, in the first place, and

(2) Within non-predicating elements, between XPs which are arguments of the predicate and those which are not.

In this view, the primary constituents of the clause is the NUCLEUS, which houses the PREDICATE (usually a verb, but need not be); the CORE, which contains the Nucleus and the Arguments of the predicate, and the PERIPHERY, which subsumes non-arguments. This is represented in figure 2.

There are additional elements which may occur in a simple sentence, i.e. a single-clause sentence. The first is the **PRECORE SLOT [PrCS]**, the position in which question words appear in languages in which they do not occur *in situ*, e.g.

English, Italian, Zapotec; it is also the location in which the fronted element in a sentence like *Bean soup I can't stand* appears. This position is clause-internal but core-external. In addition to a clause, a **simple sentence** may also include a phrase in a detached position, most commonly in the **LEFT-DETACHED POSITION [LDP]**. This is the location of sentence-initial elements, most commonly adverbials, which are set off from the clause by a pause, e.g. *Yesterday, I bought myself a new car or As for John, I haven't seen him in a couple of weeks*. The left-detached position is never obligatory. There is also a **RIGHT-DETACHED POSITION [RDP]**, as in sentences like *I know them, those boys*.

Thus, a sentence like *What did Robin show to Pat in the library yesterday?* is analysed as in figure 4 (Van Valin, 2005: 7).
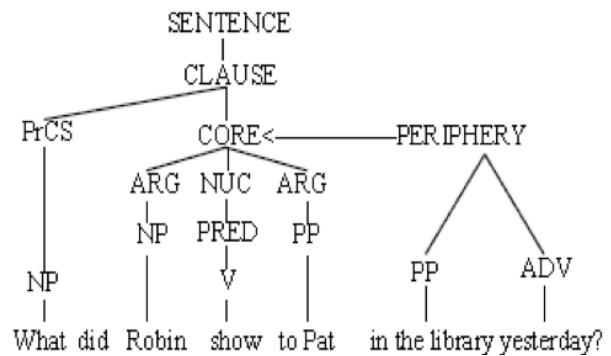
FIGURE 2

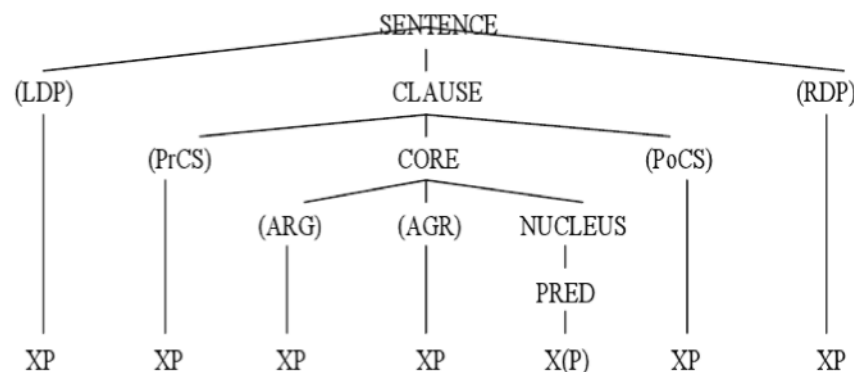Formal representation of the LSC
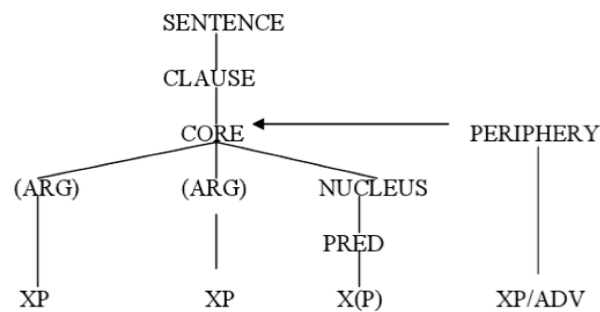
FIGURE 4

The LSC of the clause in English

FIGURE 3

Abstract LSC with extra-core and detached positions (from Van Valin & LaPolla, 1997: 38)

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
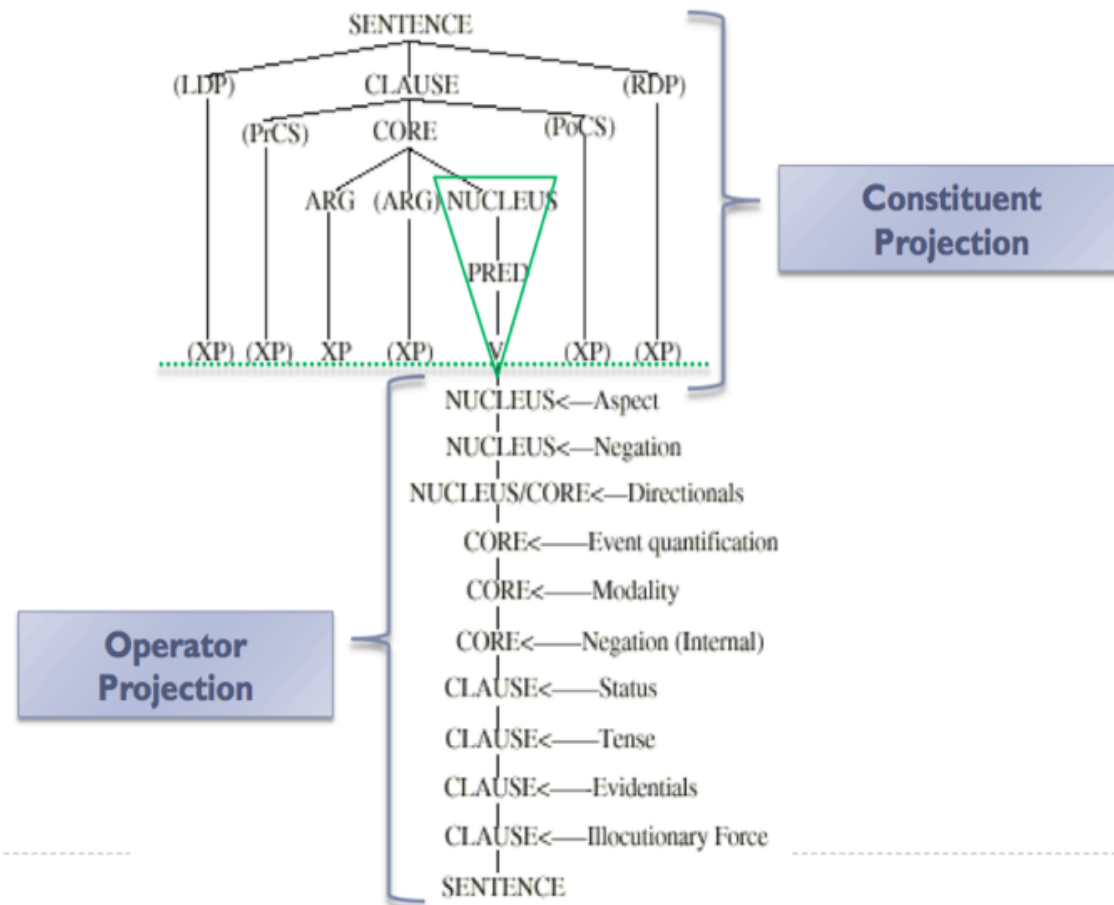Building an RRG computational grammar
93

Note that in this representation the auxiliary verb *did* is not attached to anything, and this is because it is not part of the nucleus, core or periphery. It is, rather, the morphological realization of a tense **OPERATOR** which modifies the clause. Grammatical categories like aspect, tense, and modality are treated as operators modifying different layers of the clause. Each of the clause levels may be modified by one or more operators. Since operators are qualitatively different from predicates and their arguments, they are represented in a distinct projection of the clause. The element common to both projections is the nucleus. The general schema of a projection grammar representation of the layered structure of the clause is given in figure 5.

The top part is called the "constituent projection", the bottom the "operator projection". The two projections are joined through the nucleus, which is the central element in the clause both in terms of defining the range of possible arguments, on the one hand, and being the primary entity to which the grammatical categories encoded as operators are oriented, on the other. In the operator projection, the scope of the operator is indicated by the unit which is the target of the arrow. Each operator at a given level is so represented, and if there is more than one, e.g. both tense and illocutionary force, then the relative scopes are explicitly indicated. Figure 6 shows an analysis of the sentence *Bruno might have been running around the park* giving both the constituent and the operator projections.
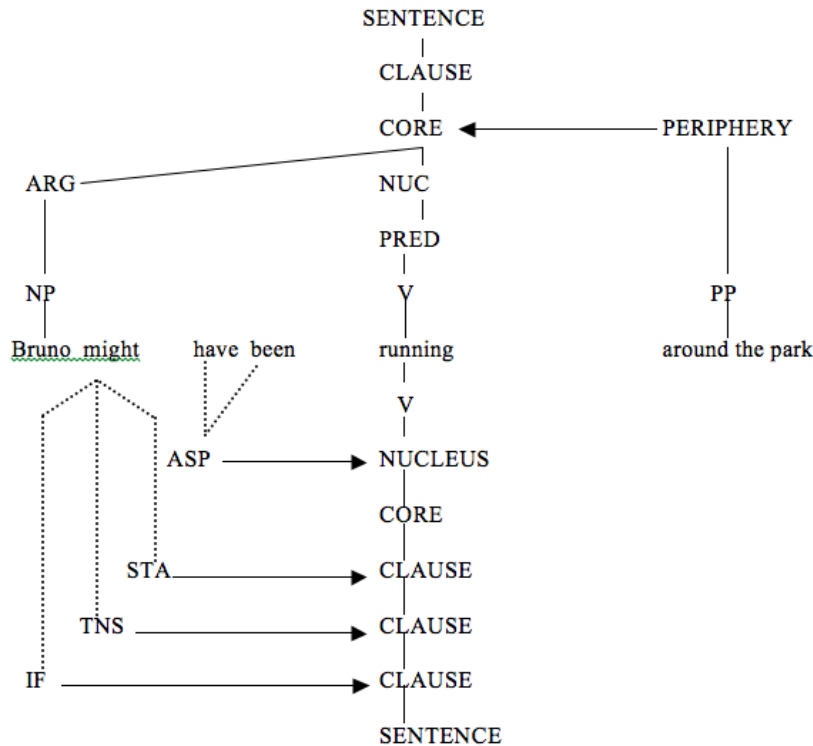
FIGURE 5
The LSC with constituent and operator projections (adapted from Van Valin, 2005: 12)

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

94

FIGURE 6
Constituent and operator projections of an English sentence

2.3. The layered structure of the clause in a computational framework: some necessary adaptations

The implementation of ARTEMIS involves some changes in the RRG descriptive apparatus. Two are especially relevant for the parsing process of simple clauses:

- The integration of a constructional node, L1-CONSTR, in the layered structure of the clause.
- The substitution of the operator projection by feature-bearing matrixes and unification mechanisms.

The first modification is already proposed in Periñán-Pascual & Arcas (2014) and it is a direct consequence of the influence of the LCM in the design of both FunGramKB and ARTEMIS. The LC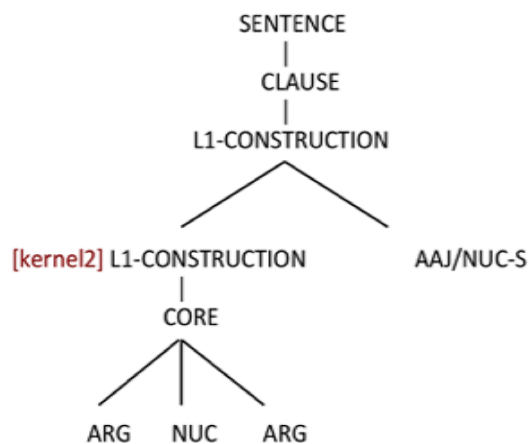M is a model of meaning construction which envisages the semantic structure of clauses as the joint result of lexical and constructional structures. This model distinguishes the following types of constructions: (a) Level 1 constructions, often called argument-structure constructions, like the ones postulated by Goldberg (1995, 2006); (b) Level 2, or implicational constructions (such as *What's X doing Y?*), which describe low-level situational cognitive models (or specific scenarios), giving rise to meaning interpretations which carry a heavily conventionalized implication; (c) Level 3 deals with illocutionary constructions (e.g. *Can you (please) X?*), which are means of encoding high-level situational models (or generic scenarios); and (d) Level 4, or discourse constructions, based on high-level non-situational cognitive models (such as reason-result or condition-consequence), with particular emphasis on cohesion and coherence phenomena.

FunGramKB incorporates these distinctions in one of its lexico-grammatical modules, the so-called Grammaticon, which is a storehouse of cons-

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

95

tructions arranged in different Levels, in accordance with the tenets for constructional organization of the LCM. FunGramKB has, therefore, two sources to motivate both semantically and structurally the basic semantic structure underlying a sentence: the inventory of Level 1 constructions (or argument structure constructions[9]) stored in the Grammaticon, and the information about the basic subcategorization frames of predicates, as encoded in their corresponding lexical entries in the Lexica. Periñán-Pascual (2013: 214) classifies the predicate frames as Kernel-1, Kernel-2 and Kernel-3 Constructions (corresponding to intransitive, monotransitive and ditransitive structures, respectively). The introduction of a distinction between Kernel and Non-kernel (or L1 Constructions) is of tantamount importance for the design of the parsing rules in the GDE, as it involves the introduction of the CONSTR-L1 node which occupies an intermediate layer between the CORE and the CLAUSE nodes in the LSC (cf. Periñán-Pascual & Arcas, 2014: 171-175, for a detailed description). Hence, the format of the enhanced LSC would be as in figure 7.

Note that, in general, Kernel Constructions would account for the configuration of the CORE in unmarked cases, as it houses by default the arguments and the primary predicate of every clause. L1-Constructions, on the other hand, may introduce further constituents into the clause, which can be of two different types: quite often some argument-constructions introduce a secondary predicate (NUC-S in the tree) as for instance in resultatives like *This weather has dried my daisies dead / to death*; other constructions, on the other hand, involve the addition of an Argument-Adjunct (AAJ)[10] as is the case of Beneficiary constituents like *for Bruno* in *Marita bought a toy bone for Bruno*; there are even cases in which both types are simultaneously added as in *Bruno barks doors* (AAJ) *open* (NUC-S) (cf. figure 8).

FIGURE 7
The enhanced format of the LSC

FIGURE 8
The enhanced LSC of an English sentence

---
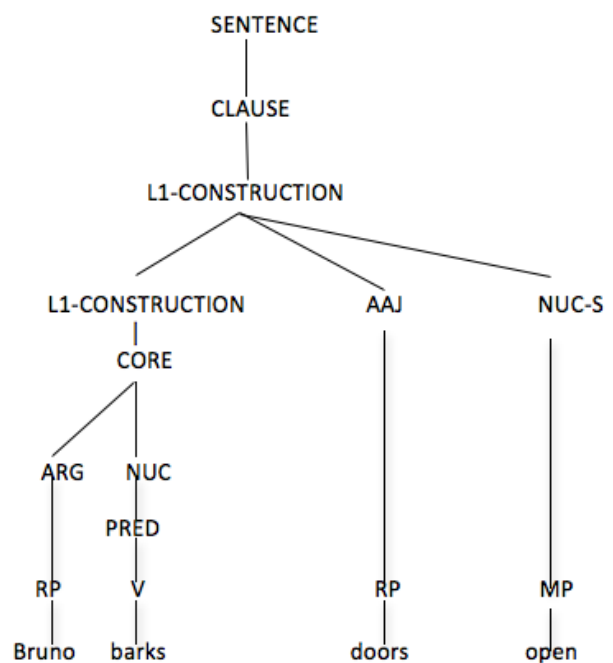
9   Level 2, 3 and 4 Constructions are pragmatic and discursive in nature. Therefore, they lie beyond the scope of the GDE in ARTEMIS.

10   Apart from Arguments and Adjuncts, RRG distinguishes a third type of clausal constituent, namely Argument-Adjuncts: they share with arguments their non-optional status but contrasts with them because they are predicative; i.e. they contribute meaning to the overall structure of the clause, as done also by adjuncts. Furthermore, since Argument-Adjuncts —quite contrarily from Arguments— are not predictable from the logical structure of the lexical predicate, it is but logical to assume that they are introduced by L1 constructions.

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

96

In Cortés-Rodríguez (2016) it is argued that, once the new L1-CONSTR layer is accepted, it seems more sensible to redefine the original Pre-Core slot position as PreC-L1 positions; the PreCore Slot is described in RRG as the place typically occupied by question words in languages in which they do not appear in situ (*Where did you find that bone?*) and also by fronted constituents as in *Excuses like this I cannot accept*. However, in sentences like:

(3) *For whom* did you wrap the gift? (Beneficiary L1-Construction)
(4) *What* did you open the safe with? (Instrumental Construction)
(5) *Into which window* did you kick the ball? (Caused-Motion L1-Construction)

the clause initial phrases are not Kernel (i.e. Core) arguments, but L1-Constructions constituents, introduced by constructional rules (i.e. once the node CONSTR-L1 is inserted in the structure), and it seems logical to consider that they occupy a PreC-L1 positions. The same would hold for any fronted or interrogative Kernel-constituent.

The second adjustment applied to the original LSC has to do with the fact that ARTEMIS follows the object-oriented paradigm and represents feature-oriented structures —as are grammatical units and nodes in the LSC— as AVMs. These are computationally implemented in the form of user-defined objects in the programming language C# (Periñán-Pascual & Arcas, 2014: 178). In doing so, ARTEMIS also follows unification approaches to grammar (Boas & Sag, 2012; Sag et al., 2003) and morphosyntactic parsing is carried out jointly by a set of production rules and a number of feature unification operations intended to satisfy the structural and semantic constraints encoded in the AVMs. The inclusion of this type of linguistic objects has a crucial impact on the so-called operator projection in the LSC. Abstract grammatical categories, such as tense, modality, or illocutionary force (i.e. operators in RRG), which modify the different nodes in the LSC

are dispensed with in the GDE in ARTEMIS since both such grammatical categories and the word tokens (function words) which encode them are endowed with AVMs lodging the corresponding values for each of the relevant categories.

Thus, the general schema represented in figure 5 above is substantially modified since the operator projection is substituted by feature-bearing nodes in the constituent projection, as partially reflected in figure 9.
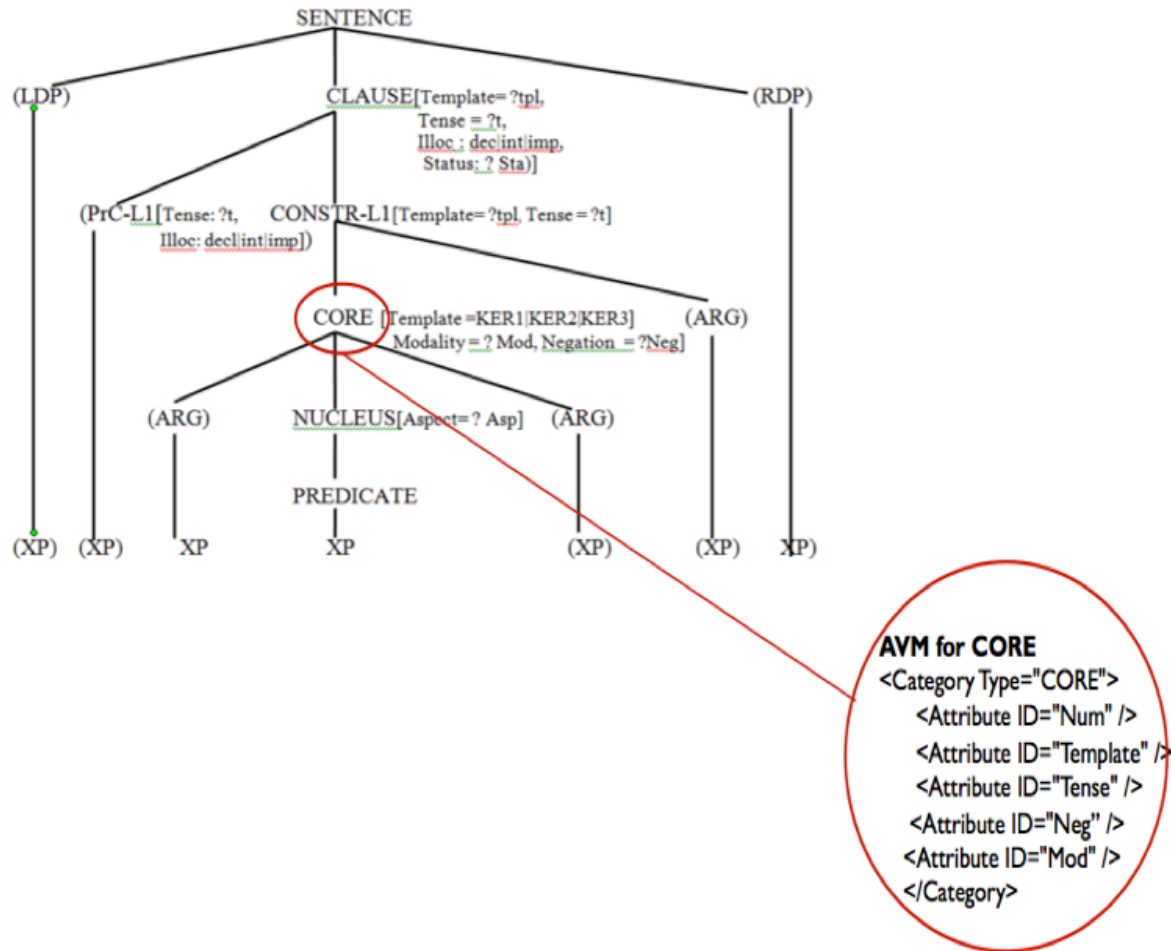
As highlighted in figure 9, grammatical units such as the node CORE are not atomic, but are interpreted as feature complexes housing different types of morphosyntactic information, which are described as Attributes; in this specific example, the AVM for the CORE includes Attributes for Number ("Num"), Template (i.e. type of argument structure), "Tense", (internal) "Negation" and "Modality". The approach to the syntactic analysis of clauses in ARTEMIS is de facto an enhanced theory of the LSC at least in the following aspects:

(i) Syntactic and semantic structures underlying the clause involve the collaboration of both lexical and constructional units; the immediate consequence of this collaboration has been the introduction of an intermediate layer, the CONSTR-L1 level, which houses constituents introduced by argument-constructions (AAJs and NUC-Ss); this new layer has in turn led to reinterpret the extra-core positions as pre- or post- ConstrL1 positions.
(ii) The different types of grammatical information modifying the layers in the LSC are not analyzed separately in an operator projection, but constitute a set of features belonging to the AVMs for grammatical objects (see appendix 3 for the full list of AVMs).

Even though the rules for the analysis of constituents at clause level in ARTEMIS is already partially based on the layered structure of the clause, as evidenced by the existence of nodes

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

97

FIGURE 9
AVMs in the LSC (a partial representation)

such as CORE, PER or NUC in its present state of development, the GDE needs a fully-fledged description of many specific syntactic rules and AVMs to comply with the RRG approach to the grammatical analysis of clauses. Sections 3 and 4 will deal with the rules and AVMs required for a detailed parsing of the innermost layers in the LSC; i.e. the NUC and the PRED.

3. The Grammar Development Environment

In accordance with the description offered in section 2, the GDE in ARTEMIS must, therefore, con-

sist of two types of theoretical constructs: on the one hand, syntactic rules, which are necessary for the construction of the syntactic trees corresponding to the LSC; on the other hand, AVMs for all units participating in the LSC which are not derivable from the knowledge base information (i.e. the lexicon, the grammaticon or the ontology) must be devised as well. Hence, the nodes corresponding to all the layers in our enhanced LSC and the word tokens for function words encoding grammatical information require such type of feature-bearing structures. Figure 10 illustrates how each of these elements will be located in different components and the distribution of tasks among such components in the parsing process.

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar
98

FIGURE 10

Task distribution among ARTEMIS and FunGramKB components in parsing
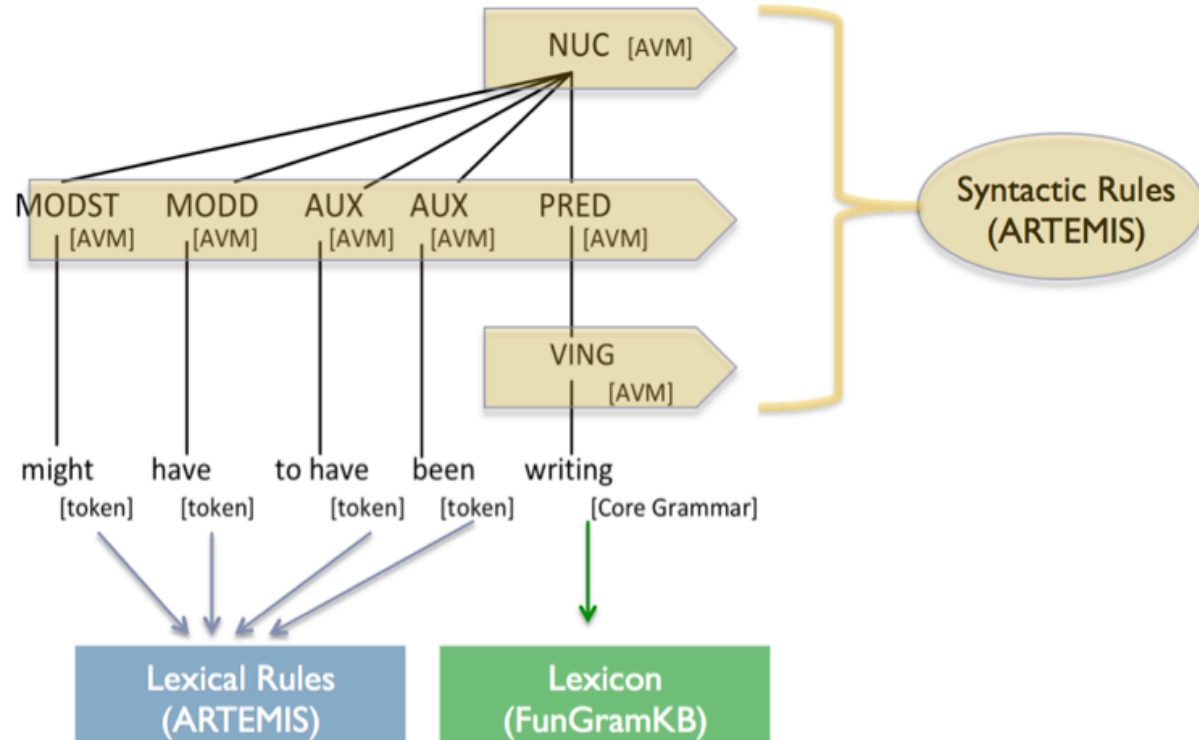
Syntactic rules are in charge of building the enhanced framework of the LSC, by spelling out the internal constituency of each of its nodes, as for instance the NUC and the PRED nodes in the figure above; in other words, the goal of syntactic rules is to convert into a computational format what we have in the enhanced LSC.

Lexical rules will provide the word tokens for function words with morphosyntactic information and content words will be assigned the grammatical and semantic information as encoded in their corresponding entries in the Lexicon, which in turn are connected with the net of conceptual structures in the Ontology in FunGramKB.

Syntactic and lexical rules are therefore the focus of this paper. The following section provides a stepwise methodology for rule-designing within the GDE.

3.1. The format of a syntactic rule

The process for the design of rules within AR-TEMIS involves making a series of decisions, as shown graphically in figure 11.

In Phase (a) it is necessary to make a serialization of the structures under study; that means to express in the format of context-free grammar rules the syntactic phenomenon that is being considered, usually described in the constituent projection in RRG. Such a rule will be incorporated in the Syntactic rules' repository within the GDE. For instance, the following partial (simplified) rule

(6) NUC → MODD PRED

would account for the combination of a verbal predicate and a deontic modal within a Nucleus,

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

99

FIGURE 11

Rule-designing phases in ARTEMIS
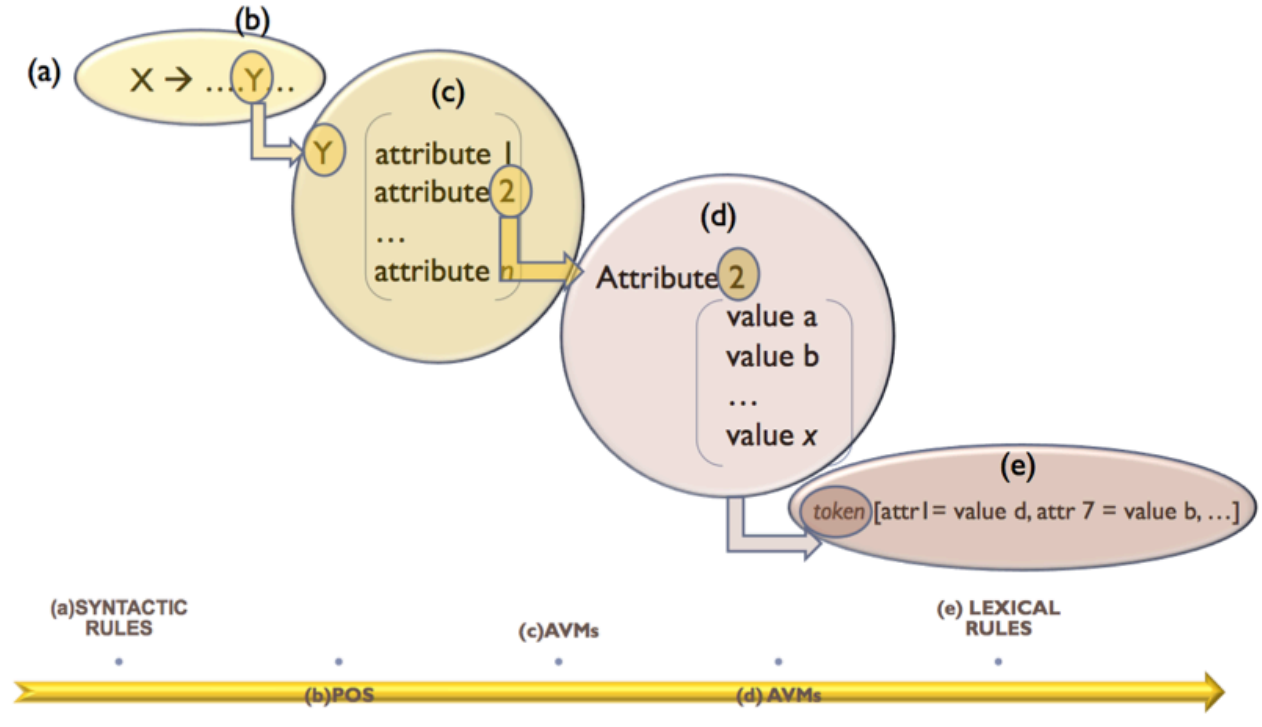
as in the sentence *He* [[*must*]MODD[*leave*]PRED]NUC *early*.

Serialization means to assign a POS label to all the elements in the sequence, something which is not necessarily the case in RRG: it was already mentioned that function words are not attached to any node in the Constituent Projection in the original LSC (as was the case of the auxiliary *did* in the sentence *What did Robin show to Pat in the library yesterday?*, analysed in figure 4 above). Once the representation of operators in a separate projection is abandoned in the revised proposal of the LSC for the GDE, they must be integrated in the rules. Consequently, in (6) the new POS label MODD was created to designate all tokens of deontic modal auxiliary verbs (*must, may, can't, has to*, etc.).

Once a new POS label is created, it must be registered in the POS repository in ARTEMIS

(phase b), and immediately afterwards its corresponding AVM must be designed and encoded in the catalogue for AVMs also in the GDE; turning again to example (6), it was necessary to create the AVM for the MODD category[11]:

(7)
    ‹**Category Type**="**MODD**"›
      ‹Attribute ID="Illoc" /›
      ‹Attribute ID="Mod" /›
      ‹Attribute ID="Num" /›
      ‹Attribute ID="Per" /›
      ‹Attribute ID="Pol" /›
      ‹Attribute ID="Syn"/›
      ‹Attribute ID="Tense" /›
    ‹/Category›

in this process all possible morphosyntactic phenomena that such a POS may express must be

---

11 AVMs are encoded in XML format, similar to that of other platforms for the analysis of human language data, as is NLTK (Natural Language Toolkit; Bird et al., 2009).

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

100

considered, as it is necessary to integrate them in the format of Attributes within the AVM (phase c); therefore, in the case of the AVM for the MODD category, the following Attributes had to be included: Illoc(utionary force), Mod(ality), Num(ber), Per(son), Pol(arity), Syn(tactic coocurrence) and Tense. Deontic modals in English show grammatical marking (i.e. have Values) for some of these Attributes.

Phase (d) involves also the design of AVMs for each of the attributes created in phase (c); these AVMs will include the Values available for each of those grammatical Attributes; thus, the AVM for the Illoc(utionary Force) Attribute will have the following format, including three values, namely 'declarative', 'interrogative' and 'imperative'[12]:

(8)
  ‹**Attribute ID**="**Illoc** " obl="*" num="1"›
    ‹Value›?illoc‹/Value›
    ‹Value Tag="declarative" ›dec‹ /Value›
    ‹Value Tag="interrogative" ›int‹ /Value›
    ‹Value Tag="imperative" ›imp‹ /Value›
  ‹/Attribute›

Finally, if the new POS refers to a type of function word, all tokens of such a category must be integrated in the GDE through the lexical rules, which include all the values that every token expresses, be they from the attributes we have designed (or from other attributes already included in the repository of AVMs), as shown in the following examples:

 (9) *couldn't*: [mod:abl, pol:neg, tns:past]
(10) *couldn't*: [mod:psbl, pol:neg, tns:past]

These lexical rules describe the grammatical information in the two word tokens *couldn't* which

display three values for the categories of Modality ('ability' or 'possibility'), Polarity (both with a 'negative' value) and Tense ('past' on both occasions).

Once the AVMs for the syntactic units, POS nodes and Attributes have been implemented, they can be integrated in the relevant positions in the syntactic rule; the result of this process for the rule in (6) would be:

(11)
    NUC [asp=?, concept=?, illoc=?, num=?, per=?, syn= ?, tpl=?, t= ?] → MODD[illoc= dec, mod= abl | obl | perm | psbl | vol, num= pl | sg | null, per= 1 | 2 | 3, null, syn= toverb | null, = past | pres | null] PRED[concept= ?, tpl=?]

This rule, however, is incomplete as it only accounts for one of the realizational possibilities of the NUC layer in English. When all possible variants of the constituent structure of NUCs are considered, it will be possible to design a completely detailed rule. This rule and the ones for the PRED layer are the topic of the next section. It will also include an explanation of the format of the rules and AVMs presented in appendixes 2 and 3.

3.2. A computational look at the LSC: The NUC and PRED layers in the GDE

This section includes the set of rules designed to adequately predict the internal configuration of the NUCLEUS layer in simple clauses[13]:

(12)
    NUC [asp=?, concept=?, illoc=?, num=?, per=?, tpl=?, t= ?] → PRED[concept= ?, illoc=?, num=?, per=?, tpl=?, t=?] || AUX[asp= pf | pr, illoc= dec |

---

12 Both in phases (c) and (d) it is advisable to take into account if such a feature occurs in other languages and, if this is the case, to consider those functional distinctions which are relevant; for instance, 'perfective' as an aspectual distinction is not present in English but it is encoded as a value in the AVM for aspect, since it exists in Spanish, as shown in the contrast between the two past forms of verbs; e.g. imperfective *comía* ('he was eating') vs. perfective *comió* ('he ate').

13 These rules only represent part of the grammar of simple active declarative clauses, which means that subordinate clauses are beyond the scope of this paper.

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

101

imp , num= pl | sg, per= 1 | 2 | 3, syn= ving | vpar, t= pas | pres] PRED[concept= ?, syn= ving | vpar, tpl=?] || MODD[illoc= dec, mod= abl | obl | perm | psbl | vol, num= p | sg | null, per= 1 | 2 | 3, null, syn= toverb | null, = past | pres | null] PRED[concept= ?, tpl=?] || MODST[illoc= dec, num= pl | sg, | null per= 1 | 2 | 3, | null, sta= inf | nec | poss | subj, syn= toverb | null, t= past | pres] PRED[concept= ?, tpl= ?] || AUX[asp= pf , illoc= dec | imp, num= pl | sg, per= 1 | 2 | 3, syn= apar, t= past | pres] APAR [asp= pr, syn= apar + ving] PRED[concept= ?, syn= ving, tpl=?] || MODD[illoc= dec, mod= abl | obl | perm | psbl | vol, num= pl | sg | null, per= 1 | 2 | 3, null, syn= toverb | null, t= past | pres | null] AUX [asp= pf | pr, syn= ving | vpar] PRED[concept= ?, syn= ving | vpar, tpl=?] || MODD[illoc=dec, mod= abl | obl | perm | psbl | vol, num: pl | sg | null, per: 1 | 2 | 3 | null, syn= toverb | null, t= past | pres | null] AUX [asp: pf, syn= apar] APAR[asp: pr syn= apar + ving] PRED[concept: ?, syn= ving, tpl=?] || MODST [illoc= dec, num= pl | sg, | null per= 1 | 2 | 3, | null, sta= inf | nec | poss | subj, syn= toverb | null, t= past | pres | null] PRED[concept= ?, syn= toverb | null, tpl=?] ||MODST[illoc= dec, num= pl | sg, | null, per= 1 | 2 | 3, | null, sta: inf | nec | poss | subj, syn= toverb | null, t= past | pres | null] AUX [asp: pf, syn= toverb | null + apar] APAR[asp: pr syn= toverb | null + apar] PRED[concept: ?, syn= ving, tpl:?] ||MODST [illoc= dec, num= pl | sg, | null, per= 1 | 2 | 3, | null, sta= inf | nec | poss | subj, syn= toverb | null, t= past | pres | null] MODD[mod= abl | obl | perm | psbl | vol, syn= toverb | null + toverb] PRED[concept= ?, syn= toverb | null, tpl:?] || MODST [illoc= dec, num= pl | sg, | null, per= 1 | 2 | 3, | null, sta= inf | nec | poss subj, syn= toverb | null, t= past | pres | null] MODD[mod= abl | obl | perm | psbl | vol, syn= toverb | null + toverb] AUX [asp= pf | pr, syn= toverb + vpar | toverb+ ving] PRED[concept= ?, syn= vpar, tpl=?] || MODST [illoc= dec, num= pl | sg, | null, per= 1 | 2 | 3, | null, sta: inf | nec | poss | subj, syn= toverb | null, t: past | pres | null] MODD[mod= abl | obl

| perm | psbl | vol, syn= toverb | null + toverb] AUX [asp= pf, syn= toverb+ apar] AUX [asp= pr, syn= apar + ving] PRED[concept= ?, syn= ving, tpl=?]

This rule, which is apparently very complex, results just from the combination of two types of lexical units: the PRED (predicate node) which can appear alone (first option in the rule) or in combination with one or several types of auxiliary verbs (as encoded in the other options).

Despite this, the internal configuration of the NUC node is rather complex if compared with the way it is analyzed in the constituent projection within RRG. This is due to the fact that function items (like auxiliary verbs) are not usually taken as part of the constituent projection but as elements that participate in the operator projection in RRG analyses. However, since there is only one single projection in ARTEMIS, they must now be integrated in the set of syntactic rules that form part of the GDE. Let us recall that the GDE consists of feature-based production rules subject to the linearity of constituents, since parsing —in accordance with Earley's algorithm— proceeds in a bottom up fashion complemented with top-down predictions. Furthermore, as stated by Periñán-Pascual & Arcas (2014: 182):

> the psychologically-plausible behavior of the parser lies in the fact that it is: a. an incremental left-corner parser, where each successive word being encountered is incorporated into a larger structure by combining bottom-up processing with top-down predictions, and b. a parallel parser, since multiple parse structures can be generated locally, so there is no need to re-analyze the input if one parse structure proves incorrect (i.e. no backtracking). (UNDERLINING IS OURS)

Thus, every lexical unit must find a room in our syntactic rules, and it seems plausible to incorporate all auxiliary verbs together with the predicate (PRED) within the NUC node; this seems close to the analysis within Systemic Linguistics

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar
102

of the so-called Verbal Group. The different types of Auxiliary verbs are codified by means of the following Nodes: AUX ('Auxiliary verb'), APAR ('Auxiliary verb - past participle'), MODD ('Modal Auxiliary verb - Deontic'), MODS ('Modal Auxiliary verb - Epistemic'). It has been necessary to establish a distinction between the two types of modal verbs as they are the formal realization of different operators in the LSC, namely the CORE operator of Modality and the CLAUSE operator of Status. The AVMS corresponding to these nodes are then as follows (the AVM for MODD is repeated here to ease the explanation; see appendix 3 for the AVMs of the other constituents):

(13)
```
‹Category Type="MODD ">
    ‹Attribute ID="Illoc" />
    ‹Attribute ID="Mod" />
    ‹Attribute ID="Num" />
    ‹Attribute ID="Per" />
    ‹Attribute ID="Pol" />
    ‹Attribute ID="Syn"/›
    ‹Attribute ID="Tense" />
</Category›
```

(14)
```
‹Category Type="MODST ">
    ‹Attribute ID="Illoc" />
    ‹Attribute ID="Num" />
    ‹Attribute ID="Per" />
    ‹Attribute ID="Pol" />
    ‹Attribute ID="Sta" />
    ‹Attribute ID="Syn"/›
    ‹Attribute ID="Tense" /›
</Category›
```

Another interesting feature that arises for the first time in rule (12) is the complex structure of the nodes in ARTEMIS, once they are analyzed as non-atomic AVMs comprising a number of grammatical features corresponding to the original Operators in RRG. These AVMS which are storehouses of grammatical operators have a very distinctive characteristic: Whereas in RRG the Ope-

rator projection is concerned only with the layer over which operators have scope, unification processes require for grammatical features to be encoded not only in the Layer which they modify, but in all the nodes dominated by such a layer down to the lexical token which is the formal expression of the operator concerned. Feature-Unification involves the percolation from such a lexical unit to the node over which the Operator has scope. For instance, in the case of the AVM for NUC, the percolation process (or 'Feature-Unification Path') of the attribute "Person" starts always in the first (leftmost) token within NUC (i.e. the first verb in the Verbal Group) and percolates up to PRED and from there to NUC, where Unification takes place, as shown in figure 12.

The dotted arrows illustrate the inheritance of features from the AUX (blue arrows) and APAR (red arrows) tokens to the NUC node. The symbol ? which appears as a Value for the other Attributes in the NUC node indicates that the Values for such Attributes are also inherited from the other dominated constituents (PRED and VING) in Unification processes.

Feature Unification can in fact run up the whole structure of the clause. A very interesting case in this regard concerns the 'Illoc(utionary force)' Attribute, which appears encoded in the AVMs of the first constituent of NUC (the first auxiliary verb or the lexical predicate if there are no auxiliaries), even though unification will finally take place at Clause level, which is the layer over which this operator has scope (cf. figure 13).

Once the AVMs for MODD and MODST were registered as new POSs in ARTEMIS, two further AVMs must be created for the Attributes that are also new, namely "Modality" and "Status" (this last label includes epistemic modals and subjunctive mood, in accordance with the values of the equivalent operator in RRG):

(15)
```
‹Attribute ID="Modality " obl="*" num="1">
    ‹Value›?mod ‹/Value›
    ‹Value›ability›abl‹/Value›
```

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

103

FIGURE 12

Feature percolation from Auxiliaries to the NUCleus

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

104
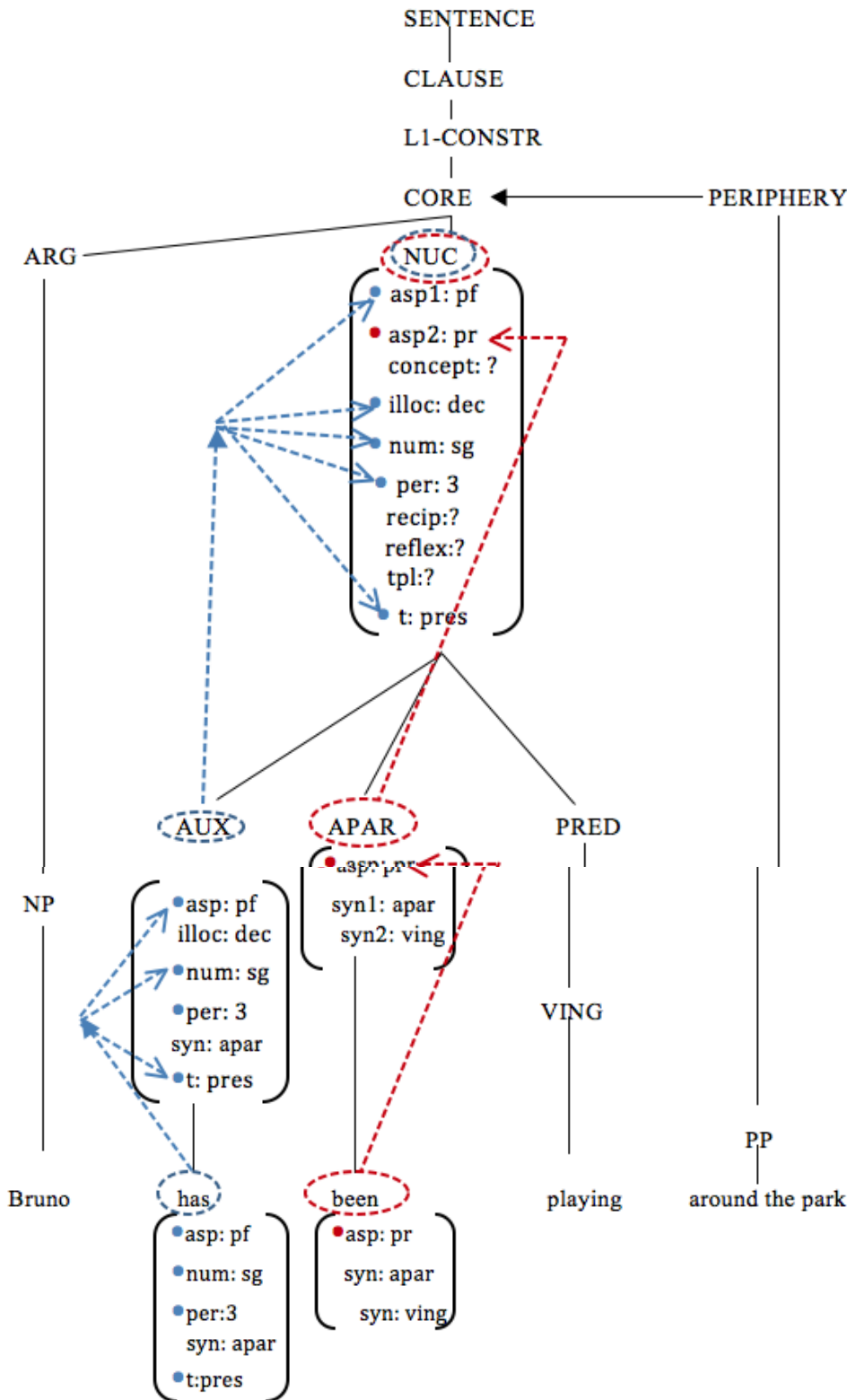
FIGURE 13

Feature Unification Path of Illocutionary Force Attribute
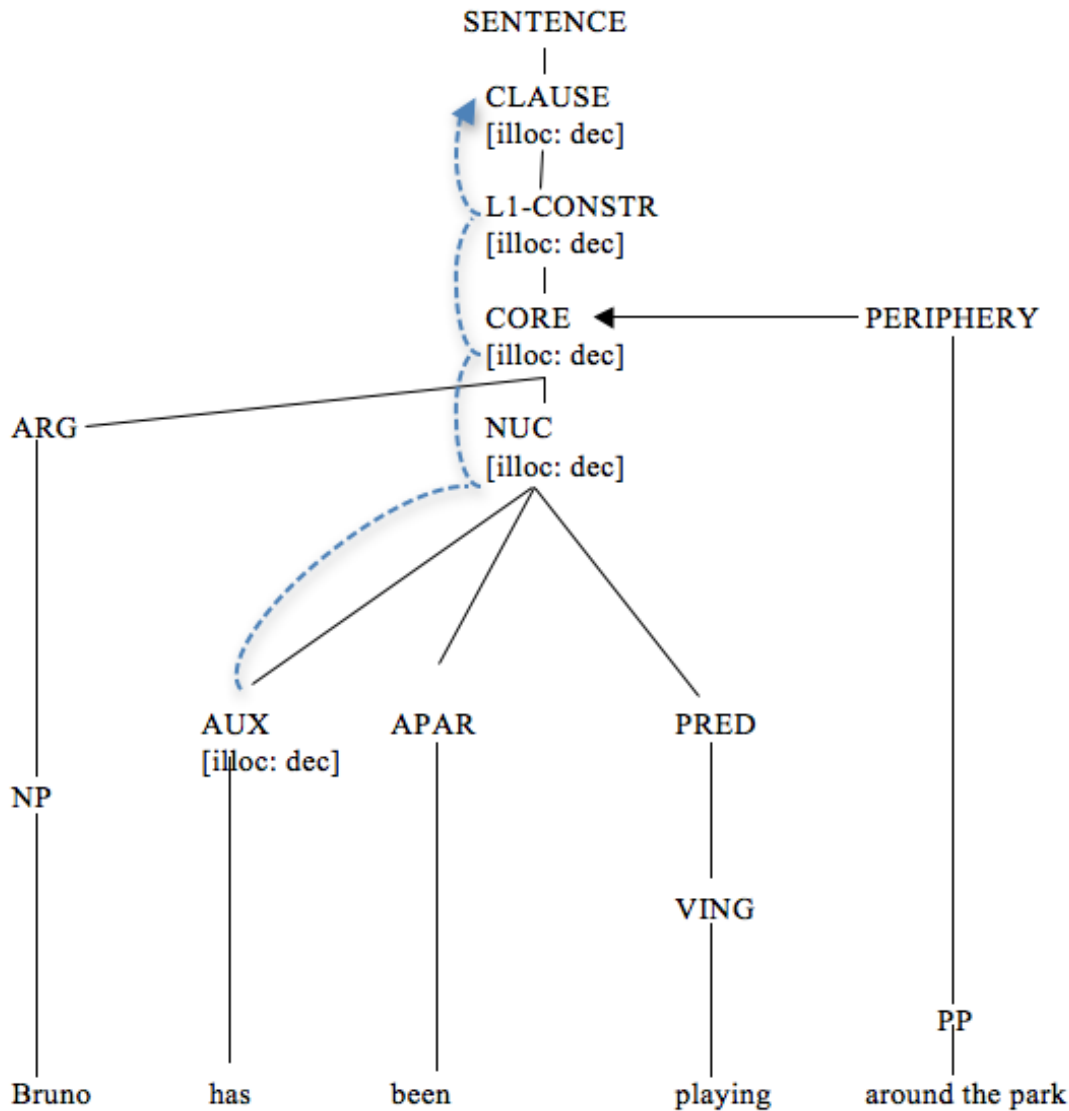
<Value›obligation›obl</Value›
<Value› permission›perm</Value›
<Value› possibility›psbl</Value›
<Value› volition›vol</Value›
</Attribute›

(16)
    <Attribute ID="**Status** " obl="*" num="1">
    <Value›?sta</Value›
    <Value›inference›inf</Value›
    <Value›necessity›nec</Value›

<Value›possibility›poss</Value›
<Value›subjunctive›subj</Value›
</Attribute›

The specific tokens for grammatical units of this type are encoded as instances of the lexical rules in the GDE; each of these tokens will encode only the attributes they codify, with their specific values. Thus, the following will be instances of how modal verbs are encoded in the lexical rules:

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

105

(17)
    MODD

can: [mod:abl, pol:pos, tns:pres]   *can't/cannot*: [mod:abl, pol:neg, tns:pres]

can: [mod:psbl, pol:pos, tns:pres]   *can't/cannot*: [mod:psbl, pol:neg, tns:pres]

could: [mod:abl, pol:pos, tns:past]   *couldn't*: [mod:abl, pol:neg, tns:past]

could: [mod:psbl, pol:pos, tns:past]   *couldn't*: [mod:psbl, pol:neg, tns:past]

may: [mod:psbl, pol:pos, tns:pres]   *might*: [mod:psbl, pol:pos, tns:past]

may: [mod:perm, pol:pos]   *might*: [illoc: int, mod:psbl, pol:pos, tns:past]

must: [mod:obl, pol:pos]   *mustn´t*: [mod:obl, pol:neg]

should:[mod:obl, pol:pos]   *shouldn't*: [mod:obl, pol:neg]

ought: [mod:obl, pol:pos, syn: toverb]   *oughtn't*: [mod:obl, pol:neg, syn: toverb]

have:[num:sing, per:1 | 2, mod:obl, pol:pos, , syn: toverb]

have:[num:pl, mod:obl, pol:pos, syn: toverb]   *has*:[num:sing, per:3, mod:obl, pol:pos, syn: toverb]

will: [mod:vol, pol:pos, tns:pres]   *won't*: [mod:vol, pol:neg, tns:pres]

shall: [mod:vol, pol:pos, tns:pres]   *shan't*: [mod:vol, pol:neg, tns:pres]

would:[mod:vol, pol:pos, tns:past]   *wouldn't*: [mod:vol, pol:neg, tns:past]

should: [mod:vol, pol:pos, tns:past]   *shouldn´t*: [mod:vol, pol:neg, tns:past]

(18)
    MODST

may: [sta:poss, pol:pos, tns:pres]   *might*: [sta:poss, pol:pos, tns:past]

must: [sta:nec, pol:pos]   *mustn´t*: [sta:nec, pol:neg]

should:[sta:inf, pol:pos]   *shouldn't*: [sta:inf, pol:neg]

ought: [sta:inf, pol:pos, syn: toverb]   *oughtn't*: [sta:inf, pol:neg, syn: toverb]

have:[num:sing, per:1 | 2, sta:nec, pol:pos, syn: toverb]

have:[num:pl, sta:nec, pol:pos, syn: toverb]   *has*: [num:sing, per:3, sta:nec, pol:pos, syn: toverb]

needn't [sta:nec, pol:neg]   *may* [sta:subj, pol:neg]

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

106

The other types of auxiliary verbs that may form part of the NUC node are grouped under the labels AUX ("Auxiliary verb") and APAR ("Auxiliary verb-participle"); their corresponding AVMs are:

(19)
    ‹Category Type="AUX"›
        ‹Attribute ID="Aspect" /›
        ‹Attribute ID="Illoc" /›
    ‹Attribute ID="Num" /›
    ‹Attribute ID="Per " /›
    ‹Attribute ID="Tense" /›
    ‹/Category›

(20)
    ‹Category Type="APAR"›
        ‹Attribute ID="Aspect" /›
        ‹Attribute ID="Syn " /›
    ‹/Category›

Within the category AUX several functional items must be included, as is the case of all tokens of the verbs *be*, *have* and *do* not encoded in other more restrictive subcategories of auxiliary verbs, like APAR. The lexical tokens of our category APAR would only be the following:

(21)
    APAR
    been [asp: pr; syn: apar + aing | ving] *been* [syn: apar + vpar]

They correspond to the participle form of the progressive auxiliary and the auxiliary *be* for passives, respectively. Other possible APAR forms like *done* and *had* are not relevant since they do not really participate in any grammatical operation. Therefore there is no need to encode them in our lexical rules.

The following figure illustrates one of the most complex combinations of Auxiliary verb forms and Predicate available within the Nucleus. It also shows how the Values of the Attribute "Syn" are unified within this layer (cf. figure 14).

FIGURE 14
Unification of "Syn" Features in a complex NUC

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

107
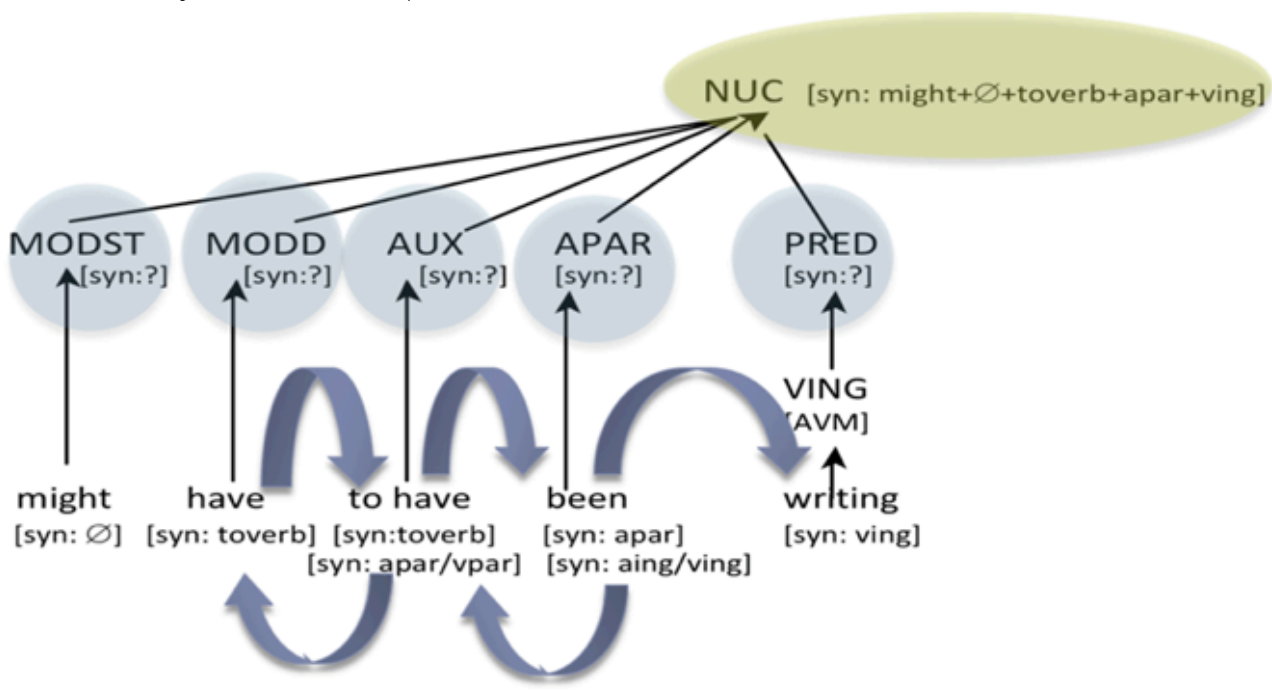
The combination of the auxiliaries plus the PRED has yielded a long rule with several disjunctive possibilities on the internal constituency of the NUCLEUS. It has also led us to set up different alternative rules to account for the realization possibilities of the PRED constituent:

(22)

PRED[concept: ?, illoc:?, num:?, per:?, recip:?, reflex:?, tpl:?, t: ?] → VERB [concept:?, illoc: dec | imp | int, num: pl | sg, per: 1 | 2 | 3, recip: g | n | o, reflex: g | n | o, tpl: ?, t: past | pres]

PRED[concept: ?, recip:?, reflex:?, tpl:?] → VPAR[concept: ?, recip: g | n | o, reflex: g | n | o, tpl: ?]] ‖ VING[concept: ?], recip: g | n | o, reflex: g | n | o, tpl:?]]

PRED[concept: ?, recip:?, reflex:?, tpl:?]→VERB [concept: ?, recip: g | n | o, reflex: g | n | o, tpl: ?]

The choice among one of the different types of tokens for the PRED constituent captured in these rules depends crucially on the absence/presence of auxiliary verbs and on the type of auxiliary that immediately precedes the lexical predicate (i.e. the PRED constituent). The following section shows several case studies, which will help to motivate the creation of the rules in (22).

4. Some case studies

Let us consider, firstly, the case in which the NUC consists of only a verbal predicate, as in the following clauses:

(23)   I write sth / he writes sth

This is an interesting case despite its apparent simplicity, as it involves a very different situation from those in which there is an AUX constituent,

as will be seen in the next case. Here the category VERB has a very rich AVM as many of the attributes encoding verbal morphology (bolfaced in the next AVM) must have a definite value:

(24)
    ‹Category Type="VERB"›
      ‹Attribute ID="Aspect" /›
      ‹Attribute ID="Concept" /›
      ‹Attribute ID="Illoc" /›
      ‹Attribute ID="Num" /›
      ‹Attribute ID="Per" /›
      ‹Attribute ID="Recip" /›
      ‹Attribute ID="Reflex" /›
      ‹Attribute ID="Template" /›
      ‹Attribute ID="Tense"
    ‹/Category›

The attributes aspect and illocutionary force are not bolfaced as aspectual distinctions in English are marked ONLY by means of AUX constituents; the same does not hold in Spanish, where there is an aspectual opposition between imperfective and perfective aspect encoded in the VERB category. However, in the AVM corresponding to the Attribute Aspect there will be at least 4 possible values —progressive, perfect, indefinite and imperfective—, thus allowing for the distinctions that hold in Spanish. The AVM for the Attribute "Aspect" would then be as follows:

(25)
    ‹Attribute ID="Aspect" obl="*" num="s"›
    ‹Value›?asp‹/Value›
        ‹Value Tag="indefinite"›ind‹/Value›
        ‹Value Tag="imperfective"›imp‹/Value›
        ‹Value Tag="perfect"›pf‹/Value›
        ‹Value Tag="progressive"›pr‹/Value›
    ‹/Attribute›

Somewhat similar is the behavior of illocutionary force in Spanish, which quite often is only

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

108

marked at the phonological level[14]. The (partial) rules to account for the internal structure of the NUCLEUS in (23) are:

(26)
NUC [asp: ?, concept: ?, illoc:?, num:?, per:?, recip:?, reflex:?, tpl:?, t: ?] → PRED[concept: ?, illoc:?, num:?, per:?, recip:?, reflex:?, tpl:?, t: ?]

PRED[concept: ?, illoc:?, num:?, per:?, recip:?, reflex:?, tpl:?, t: ?] → VERB[concept:?, illoc: dec | imp | int, num: pl | sg, per: 1 | 2 | 3, recip: g | n | o, reflex: g | n | o, tpl: ?, t: past | pres]

There are other possibilities within the NUC in English sentences once Auxiliary verbs are introduced in clauses, as in:

(27)
I have written something/ I must write something / He may come

This second case illustrates the situation in which there is an Auxiliary verb (AUX, if primary auxiliary, MODD or MODST if modal auxiliary) in the NUC, bearing all morphological marking and leaving, consequently, the VERB AVM housed in the PRED of this clause empty of such marks. This can only be encoded in a very different subset of syntactic rules:

(28)
NUC [asp: ?, concept: ?, illoc:?, num:?, per:?, recip:?, reflex:?, tpl:?, t: ?] → AUX[asp: pf | pr, illoc: dec | imp, num: pl | sg, per: 1 | 2 | 3, t: past | pres] PRED[concept: ?, recip:?, reflex:?, tpl:?] ‖ MODD[illoc:dec, mod: abl | obl | perm | psbl | vol, num: pl | sg | null, per: 1 | 2 | 3, null t: past | pres | null] PRED[concept: ?, recip:?,

reflex:?, tpl:?] ‖ MODST[illoc:dec, num: p | sg, | null per: 1 | 2 | 3, | null sta: inf | nec | poss | subj, t: past | pres]   PRED[concept: ?, recip:?, reflex:?, tpl:?]

PRED[concept: ?, recip:?, reflex:?, tpl:?] → VPAR[concept: ?, recip: g | n | o, reflex: g | n | o, tpl: ?]] ‖ VING[concept: ?], recip: g | n | o, reflex: g | n | o, tpl:?]]

Note that in this case, there is a redistribution of attributes: The AVMS for VPAR and VING retain only the attributes for the grammatical features that are encoded in the lexical entry of the corresponding verbal unit:

(29)
```
<Category Type="VPAR">
  <Attribute ID="Concept" />
  <Attribute ID="Recip" />
  <Attribute ID="Reflex" />
  <Attribute ID="Template" />
</Category>

<Category Type="VING">
  <Attribute ID="Concept" />
  <Attribute ID="Recip" />
  <Attribute ID="Reflex" />
  <Attribute ID="Template" />
</Category>
```

The rest of features are saturated in the AVMs of the auxiliaries.

Example (30) is another case which shows the complexity of value distribution when there is a complex verbal group with several AUX constituents:

(30)   I have been writing something

---

14 Arguably, imperative illocution in English could be analyzed as just involving a certain phonological contour of the clause, as in *You write something!* However, facts are a bit more complicated since illocutionary force interacts with tense, and in the case of imperative clauses, they are tenseless.

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

109

The first AUX retains the same values as in (28), and the second AUX has a specific "pr(ogressive)" value for the "Aspect" Attribute:

(31)
NUC [asp:?, concept: ?, illoc:?, num:?, per:?, recip:?, reflex:?, tpl:?, t: ?] → AUX[asp: pf , illoc: dec | imp, num: pl | sg, per: 1 | 2 | 3, t: past | pres] APAR [asp: pr] PRED[concept: ?, recip:?, reflex:?, tpl:?]

PRED[concept: ?, recip:?, reflex:?, tpl:?] → VING [concept: ?], recip: g | n | | , reflex: g | n | o, tpl:?]]

A different distribution of attributes and values takes place when there is a combination of one lexical verbal head plus a number of auxiliary verbs (AUX and MODD o MODST categories), as in the clauses in (32):

(32)
He must be writing a new chapter / He must have written something / He must have been writing a new chapter.

The first of the auxiliaries is responsible of bearing the marks for number, person and tense, plus an additional attribute for either modality or status.

(33)
NUC [asp: ?, concept: ?, illoc:?, mod: ?, num:?, per:?, recip:?, reflex:?, tpl:?, t: ?]→MODD[illoc: dec, mod: abl | obl | perm | psbl | vol, num: pl | sg | null, per: 1 | 2 | 3, null t: past | pres | null] AUX [asp: pf | pr] PRED[concept: ?, recip:?, reflex:?, tpl:?] ‖ MODD[illoc:dec, mod: abl | obl | perm | psbl | vol, num: pl | sg | null, per: 1 | 2 | 3, null t: past | pres | null] AUX [asp: pf] APAR[asp: pr] PRED[concept: ?, recip:?, reflex:?, tpl:?]

PRED[concept: ?, recip:?, reflex:?, tpl:?]→ VING[concept: ?], recip: g | n | o, reflex: g | n

| o, tpl: ?]] ‖ VPAR[concept: ?], recip: g | n | o, reflex: g | n | o, tpl:?]

If the modal verb encodes epistemic modality, as in (34):

(34)
He may be writing / He may have written / He may have been writing now

the rules are:

(35)
NUC [asp: ?, concept: ?, illoc:?, num:?, per:?, recip:?, reflex:?, sta: ?, tpl:?, t: ?]→MODST [illoc:dec, num: pl | sg, | null per: 1 | 2 | 3 | null sta: inf | nec | poss | subj, t: past | pres | null] PRED[concept: ?, recip:?, reflex:?, tpl:?] ‖ MODST[illoc:dec, num: pl | sg, | null per: 1 | 2 | 3, | null sta: inf | nec | poss | subj, t: past | pres | null] AUX [asp: pf] APART[asp: pr] PRED[concept: ?, recip:?, reflex:?, tpl:?]

PRED[concept: ?, recip:?, reflex:?, tpl:?]→ VING[concept: ?, recip: g | n | o, reflex: g | n | o, tpl:?] ‖ VPAR[concept:?, recip: g | n | o, reflex: g | n | o, tpl:?]

There are cases in which the combination concerns two modal auxiliaries (one MODD and one MODST) and, possibly, some aspectual AUX element, as in the following example:

(36)
He may have to write something

The subset of rules for these cases is as follows:

(37)
NUC [concept: ?, illoc:?, mod: ?, num:?, per:?, recip:?, reflex:?, sta: ?, tpl:?, t: ?]→MODST [illoc:dec, num: pl | sg, | null per:1 | 2 | 3, | null sta: inf | nec | poss | subj, t: past | pres | null] MODD[mod: abl | obl | perm | psbl | vol] PRED[concept: ?, recip:?, reflex:?, tpl:?]

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

110

PRED[concept: ?, recip:?, reflex:?, tpl:?]→
VERB [concept: ?, recip: g | n | o, reflex: g | n
| o, tpl: ?]

The following examples offer a more complex combination of two modal verbs and an aspectual AUX form; the introduction of this AUX aspectual auxiliary triggers an allomorphic variation of the head element in PRED, as it must be a gerund (VING) or a past participle (VPAR):

(38)
He may have to be writing something/ He may have to have written something

The parsing rules for these structures are:

(39)
NUC [asp: ?, concept: ?, illoc:?, mod: ?, num:?, per:?, recip:?, reflex:?, sta: ?, tpl:?, t: ?]→ MODST [illoc:dec, num: p | sg, | null per: 1 | 2 | 3, | null sta: inf | nec | poss | subj, t: pas | pres | null] MODD[mod: abl | obl | perm | psbl | vol] AUX [asp: pf | pr] PRED[concept: ?, recip:?, reflex:?, tpl:?]

PRED[concept: ?, recip:?, reflex:?, tpl:?] →
VING[concept: ?], recip: g | n | o, reflex: g | n
| o, tpl: ?] ‖VPAR[concept: ?, recip: g | n | o, reflex: g | n | o, tpl:?]

Very infrequent, though not impossible, are the cases in which there is the full gamut of auxiliaries in the NUC, as in (40):

(40)
He might have to have been writing for the whole afternoon

The rules show a slight variation with regard to those in (39), as the rule for NUC includes two modal auxiliares and two AUX aspectual forms, and the PRED is either a VING or a VPAR lexical unit:

(41)
NUC [asp: ?, concept: ?, illoc:?, mod: ?, num:?, per:?, recip:?, reflex:?, sta: ?, tpl:?, t: ?]→ MODST [illoc:dec, num: pl | sg, | null per: 1 | 2 | 3, | null sta: inf | nec | poss | subj, t: past | pres | null] MODD[mod: abl | obl | perm | psbl | vol] AUX [asp: pf] AUX [asp: pr] PRED[concept: ?, recip:?, reflex:?, tpl:?]

PRED[concept: ?, recip:?, reflex:?, tpl:?] →
VING[concept: ?, recip: g | n | o, reflex: g | n | o, tpl: ?] ‖VPAR[concept: ?, recip: g | n | o, reflex: g | n | o, tpl:?]]

5. Conclusions

This paper provides a first approximation towards the computational implementation of the LSC in RRG. Within the framework of ARTEMIS, this paper concentrates on the GDE, which stores three types of rules: lexical, constructional and syntactic. While lexical and constructional rules are created automatically, syntactic rules are manually constructed and are in charge of providing a parsed syntactic tree following the LSC distinctions. ARTEMIS is inspired in unification approaches and morphosyntactic parsing is carried out jointly by a set of production rules and a number of feature unification operations intended to satisfy the structural and semantic constraints encoded in the AVMs. Several case studies, each representing complex grammatical structures within the NUCLEUS layer in the enhanced LSC, have been discussed in order to illustrate how the interaction of the components of the GDE manage to provide an effective description of such structures.

6. References

Bird, Stephen, Ewan Loper and Edward Klein, 2009: *Natural Language Processing with Python*, Bei-

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

111

jing, Cambridge, Farnham, Köln, Sebastopol, Tokyo: O'Reilly Media Inc.

Boas, Hans C., 2005: "From theory to practice: Frame Semantics and the design of FrameNet" in Stefan Langer and Daniel Schnorbusch (eds.): *Semantik im Lexikom*, Tübingen: Narr, 129-160.

Boas, Hans C., and Ivan Sag (eds.), 2012: *Sign-Based Construction Grammar*, Stanford: CSLI.

Cortés-Rodríguez, Francisco J., 2016: "Towards the computational implementation of Role and Reference Grammar: Rules for the syntactic parsing of RRG Phrasal constituents", *Círculo de lingüística aplicada a la comunicación* 65, 76-108 [http://www.ucm.es/info/circulo/no65/cortes.pdf].

Debrauwer, Laurent and Fien Van der Heyde, 2010: *UML 2.Modelización de objetos*, Barcelona: ENI.

Díaz-Galán, Ana C. and M. Carmen Fumero Pérez, 2015: "Developing parsing rules within ARTEMIS: the case of DO auxiliary insertion" in Carlos Periñán-Pascual and Eva Mestre I Mestre (eds.): *Understanding Meaning and Knowledge Representation: From Theoretical and Cognitive Linguistics to Natural Language Processing*, Newcastle: Cambridge Scholars Press, 283-302.

Diedrichsen, Elke, 2011: "The theoretical importance of constructional schemas in RRG" in Wataru Nakamura (ed.): *New Perspectives in Role and Reference Grammar*, Newcastle upon Tyne: Cambridge Scholars Publishing, 168-197.
Diedrichsen, Elke, 2014: "A Role and Reference Grammar Parser for German" in Brian Nolan and Carlos Periñán-Pascual (eds): *Language Processing and Grammars*, Amsterdam/Philadelphia: John Benjamins, 105-142.

Foley, William A. and Robert D.Jr. Van Valin, 1984: *Functional Syntax and Universal Grammar*, Cambridge: Cambridge University Press.

Fillmore, Charles J., Christopher R. Johnson and Miriam R. L. Petruck, 2003a: "Background to FrameNet", *International Journal of Lexicography* 16(3), 235-250.

Fillmore, Charles J., Miriam R. L. Petruck, Josef Ruppenhofer and Abby Wright, 2003b: "FrameNet in Action. The case of Attaching", *International Journal of Lexicography* 16(3), 297-332.

Goldberg, Adele E., 1995: *Constructions: A Construction Grammar approach to argument structure*, Chicago: University of Chicago Press.

Goldberg, Adele E., 2006: *Constructions at Work. The Nature of Generalization in Language*, Oxford: Oxford University Press.

Guest, Elisabeth, 2009: "Parsing using the Role and Reference Grammar paradigm" [http://eprints.leedsbeckett.ac.uk/778/6/Parsing%20Using%20the%20Role%20and%20Reference%20Grammar%20Paradigm.pdf, accessed 19 February 2016].

Mairal Usón, Ricardo, 2012: "La arquitectura de una base de conocimiento léxico conceptual: implicaciones lingüísticas" in Mabel Giammatteo, Laura Ferrari and Hilda Albano (eds.): *Léxico y Sintaxis*, Mendoza: FFyL, UNCuyo y SAL, 183-210 [http://ffyl.uncu.edu.ar/spip.php?article3638].

Mairal Usón, Ricardo, Lilian Guerrero and Carlos González (eds.), 2012: *El funcionalismo en la teoría lingüística. La Gramática del Papel y la Referencia. Introducción, avances y aplicaciones*, Madrid: Akal.

Mairal Usón, Ricardo, Carlos Periñán-Pascual and M. Beatriz Pérez Cabello de Alba, 2012: "La representación léxica. Hacia un enfoque ontológico" in Ricardo Mairal Usón, Lilian Guerrero and Carlos González (eds.): *El funcionalismo en la teoría lingüística. La Gramática del Papel y la Referencia.*

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

112

Introducción, avances y aplicaciones, Madrid: Akal, 85-102.

MAIRAL USÓN, Ricardo and Carlos PERIÑÁN-PASCUAL, 2014: "Representing Constructional Schemata in FunGramKB Grammaticon" in Jens FLEISCHHAUER, Anja LATROUITE and Rainer OSSWALD (eds.): *Exploring the Syntax-Semantics Interface*, Düsseldorf: Düsseldorf University Press.

MAIRAL USÓN, Ricardo, and Francisco J. RUIZ DE MENDOZA, 2008: "New challenges for lexical representation within the Lexical-Constructional Model", *Revista Canaria de Estudios Ingleses* 57, 137-158.

MAIRAL USÓN, Ricardo, and Francisco J. RUIZ DE MENDOZA, 2009: "Levels of description and explanation in meaning construction" in Christopher BUTLER and J. MARTÍN ARISTA (eds.): *Deconstructing Constructions*, Amsterdam/Philadelphia: John Benjamins, 53-198.

NOLAN, Brian, and Carlos PERIÑÁN-PASCUAL (eds.), 2014: *Language Processing and Grammars*, Amsterdam: John Benjamins.

NOLAN, Brian, and Yasser SALEM, 2011: "UniArab;: RRG Arabic-to-English machine translation" in Wataru NAKAMURA (ed.): *New Perspectives in Role and Reference Grammar*, Newcastle upon Tyne: Cambridge Scholars, 312-346.

OVCHINNIKOVA, Ekaterina, Laure VIEU, Alessandro OLTRAMARI, Stephano BORGO and Theodore ALEXANDROV, 2010: "Data-driven ontological analysis of Frame Net for Natural Language Reasoning", *Proceedings of the 7th Conference on International Language Resources and Evaluation (LREC'10)*, Valetta (Malta) [http://ovchinnikova.me/papers/LREC_OOVBA_final.pdf].

PAVEY, Emma, 2010: *The Structure of Language. An Introduction to Grammatical Analysis*, Cambridge: Cambridge University Press.

PERIÑÁN-PASCUAL, Carlos, 2013: "Towards a model of constructional meaning for natural language understanding" in Brian NOLAN and Elke DIEDRICHSEN (eds.): *Linking Constructions into Functional Linguistics: The Role of Constructions in a Functional Grammar*, Amsterdam and Philadelphia: John Benjamins, 205-230

PERIÑÁN-PASCUAL, Carlos, and Francisco ARCAS, 2007: "Cognitive modules of an NLP knowledge base for natural language understanding", *Procesamiento del Lenguaje Natural* 39, 197-204.

PERIÑÁN-PASCUAL, Carlos, and Francisco ARCAS, 2010: "The Architecture of FungramKB", *Proceedings of the 7th International Conference on Language Resources and Evaluation*, Malta: European Language Resources Association, 2667-2674.

PERIÑÁN-PASCUAL, Carlos, and Francisco ARCAS, 2014: "The implementation of the FunGramKB CLS Constructor in ARTEMIS" in Carlos PERIÑÁN-PASCUAL and Brian NOLAN (eds.): *Language Processing and Grammars*, Amsterdam/Philadelphia: John Benjamins,165-196.

PERIÑÁN-PASCUAL, Carlos, and Ricardo MAIRAL USÓN, 2009: "Bringing Role and Reference Grammar to natural language understanding", *Procesamiento del Lenguaje Natural* 43, 265-273.

PERIÑÁN-PASCUAL, Carlos, and Ricardo MAIRAL USÓN, 2012: "La dimensión computacional de la Gramática del Papel y la Referencia: la estructura lógica conceptual y su aplicación en el procesamiento del lenguaje natural" in Ricardo MAIRAL USÓN, Lilian GUERRERO and Carlos GONZÁLEZ (eds.): *El funcionalismo en la teoría lingüística. La Gramática del Papel y la Referencia. Introducción, avances y aplicaciones*, Madrid: Akal, 333-348.

RUIZ DE MENDOZA, Francisco J., 2013: "Meaning construction, meaning interpretation and formal expression in the Lexical Constructional Model" in Brian NOLAN and Elke DIEDRICHSEN (eds.): *Linking Constructions into Functional Linguistics*, Amsterdam/Philadelphia: John Benjamins, 231-270.

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

113

Ruiz de Mendoza, Francisco J., and Alicia Galera, 2014: *Cognitive modelling: A Linguistic Perspective*, Amsterdam/Philadelphia: John Benjamins.

Ruiz de Mendoza, Francisco J., and Ricardo Mairal Usón, 2007: "Levels of semantic representation: Where lexicon and grammar meet", *Interlingüística* 17, 26-47.

Rumbaugh, James, Ivar Jacobson and Grady Booch, 1999: *The Unified Modeling Language Reference Manual*, Reading MA: Addison-Wesley.

Sag, Ivan. A., Thomas Wasow and Emily M. Bender, 2003: *Syntactic Theory: Formal Introduction*, Stanford: CSLI Publications.

Salem, Yasser, Arnold Hensman and Brian Nolan, 2008: "Towards Arabic to English machine translation", *ITB Journal* 17, 20-31.

Shen, Dan, and Mirella Lapata, 2007: "Using Semantic Roles to Improve Question Answering", *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL),* 12-21 [https://www.aclweb.org/anthology/D/D07/D07-1002.pdf].

Van Valin, Robert D. Jr., 2005: *Exploring the Syntax-Semantics Interface*, Cambridge: Cambridge University Press.
Van Valin, Robert D. Jr., 2008: "RPs and the nature of lexical and syntactic categories in Role an Reference Grammar" in Robert D. Jr. Van Valin (ed): *Investigations of the Syntax-Semantics-Pragmatics Interface*, Amsterdam/Philadelphia: John Benjamins, 161-178.
Van Valin, Robert D. Jr., 2013: "Lexical representation, co-composition, and linking syntax and semantics" in James Pustejovsky, Pierrette Bouillon, Itoshi Isahara, Kyoko Kanzaki and Chungmin Lee (eds.), *Advances in the Generative Lexicon*, Dordrecht: Springer, 67-108.

Van Valin, Robert D. Jr., and Ricardo Mairal Usón, 2014: "Interfacing the Lexicon and an Ontology in a Linking Algorithm" in M. Ángeles Gómez, Francisco J. Ruiz De Mendoza and Francisco Gonzálvez-García (eds.): *Theory and Practice in Functional-Cognitive Space*, Amsterdam: John Benjamins, 205-228.

7. Appendixes
7.1. Appendix 1: List of abbreviations

AAJ	Argument-adjunct
ADV	Adverb
ADJ	Adjunct
APAR	Auxiliary (participle)
ARG	Argument
AUX	Auxiliary verb
AVM	Attribute-Value Matrix
CL	Clause
CONSTR-L1	Level 1 Construction
GDE	Grammar Development Environment
LDP	Left detached Position
LSC	Layered Structure of the Clause
MODD	Modal verb (deontic)
MODST	Modal verb (epistemic)
N	Noun
NUC	Nucleus
NUC-S	Secondary Nucleus
XP	Phrase
PER	Periphery
PoCS	Post-Core Slot
POS	Part of speech
PP	Prepositional Phrase
PrCS	PreCore Slot
PreC-L1	Pre L1 Construction Slot
PRED	Predicate
RDP	Right Detached Position
RRG	Role and Reference Grammar
S	Sentence
VING	Verb (gerund form)
VPAR	Verb (participle)

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

114

7.2. Appendix 2: SYNTACTIC RULES

The Nucleus

NUC [asp: ?, concept: ?, illoc:?, num:?, per:?, recip:?, reflex:?, tpl:?, t: ?] →
PRED[concept: ?, illoc:?, num:?, per:?, recip:?, reflex:?, tpl:?, t: ?] ‖ AUX[asp: pf | pr, illoc: dec | imp, num: pl | sg, per: 1 | 2 | 3, t: past | pres] PRED[concept: ?, recip:?, reflex:?, tpl:?] ‖ MODD[illoc:dec, mod: abl | obl | perm | psbl | vol, num: pl | sg | null, per: 1 | 2 | 3, null t: past | pres | null] PRED[concept: ?, recip:?, reflex:?, tpl:?] ‖ MODST[illoc:dec, num: pl | sg, | null per: 1 | 2 | 3, | null sta: inf | nec | poss | subj, t: past | pres]   PRED[concept: ?, recip:?, reflex:?, tpl:?] | AUX[asp: pf , illoc: dec | imp, num: pl | sg, per: 1 | 2 | 3, t: past | pres] APAR [asp: pr] PRED[concept: ?, recip:?, reflex:?, tpl:?] ‖ MODD[illoc: dec, mod: abl | obl | perm | psbl | vol, num: pl | sg | null, per: 1 | 2 | 3, null t: past | pres | null] AUX [asp: pf | pr] PRED[concept: ?, recip:?, reflex:?, tpl:?] ‖ MODD[illoc:dec, mod: abl | obl | perm | psbl | vol, num: pl | sg | null, per: 1 | 2 | 3, null t: past | pres | null] AUX [asp: pf] APAR[asp: pr] PRED[concept: ?, recip:?, reflex:?, tpl:?] ‖ MODST [illoc:dec, num: pl | sg, | null per: 1 | 2 | 3, | null sta: inf | nec | poss | subj, t: past | pres] PRED[concept: ?, recip:?, reflex:?, tpl:?] ‖ MODST[illoc:dec, num: pl | sg, | null per: 1 | 2 | 3, | null sta: inf | nec | poss | subj, t: past | pres] AUX [asp: pf] APART[asp: pr] PRED[concept: ?, recip:?, reflex:?, tpl:?] ‖ MODST [illoc:dec, num: pl | sg, | null per: 1 | 2 | 3, | null sta: inf | nec | poss | subj, t: past | pres | null] MODD[mod: abl | obl | perm | psbl | vol] PRED[concept: ?, recip:?, reflex:?, tpl:?] ‖NUC [asp: ?, concept: ?, illoc:?, mod: ?, num:?, per:?, recip:?, reflex:?, sta: ?, tpl:?, t: ?]→ MODST [illoc:dec, num: pl | sg, | null per: 1 | 2 | 3, | null sta: inf | nec | poss | subj, t: past | pres | null] MODD[mod: abl | obl | perm | psbl | vol] AUX [asp: pf | pr] PRED[concept: ?, recip:?, reflex:?, tpl:?] | MODST [illoc:dec, num: pl | sg, | null per: 1 | 2 | 3, | null sta: inf | nec | poss | subj, t: past | pres | null] MODD[mod: abl | obl | perm | psbl | vol] AUX [asp: pf] AUX [asp: pr] PRED[concept: ?, recip:?, reflex:?, tpl:?]

The Predicate

PRED[concept: ?, illoc:?, num:?, per:?, recip:?, reflex:?, tpl:?, t: ?] → VERB [concept:?, illoc: dec | imp | int, num: pl | sg, per: 1 | 2 | 3, recip: g | n | o, reflex: g | n | o, tpl: ?, t: past | pres]

PRED[concept: ?, recip:?, reflex:?, tpl:?] → VPAR[concept: ?, recip: g | n | o, reflex: g | n | o, tpl: ?]] ‖ VING[concept: ?], recip: g | n | o, reflex: g | n | o, tpl: ?]]

PRED[concept: ?, recip:?, reflex:?, tpl:?]→ VERB [concept: ?, recip: g | n | o, reflex: g | n | o, tpl: ?]

7.3. Appendix 3: ATTRIBUTE-VALUE MATRIXES (AVMs)

Parts of Speech (POSs)
Layer: PREDICATE

```
<Category Type="PRED">
  <Attribute ID="Concept" />
  <Attribute ID="Illoc" />
  <Attribute ID="Num" />
  <Attribute ID="Per" />
  <Attribute ID="Recip" />
  <Attribute ID="Reflex " />
  <Attribute ID="Template" />
  <Attribute ID="Tense" />
</Category>

<Category Type="VERB">
  <Attribute ID="Concept" />
  <Attribute ID="Illoc" />
  <Attribute ID="Num" />
  <Attribute ID="Per" />
  <Attribute ID="Recip" />
  <Attribute ID="Reflex" />
  <Attribute ID="Template" />
  <Attribute ID="Tense" />
</Category>

<Category Type="VPAR">
  <Attribute ID="Concept" />
```

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

115

```
    <Attribute ID="Recip" />
    <Attribute ID="Reflex" />
    <Attribute ID="Template" />
</Category>

<Category Type="VING">
    <Attribute ID="Concept" />
    <Attribute ID="Recip" />
    <Attribute ID="Reflex" />
    <Attribute ID="Template" />
</Category>
```

Layer: NUCLEUS

```
    <Category Type="NUC">
    <Attribute ID="Aspect" />
    <Attribute ID="Concept" />
    <Attribute ID="Illoc" />
    <Attribute ID="Mod" />
    <Attribute ID="Num" />
    <Attribute ID="Per" />
    <Attribute ID="Recip" />
    <Attribute ID="Reflex" />
    <Attribute ID="Sta" />
    <Attribute ID="Template" />
    <Attribute ID="Tense" />
</Category>

<Category Type="AUX">
    <Attribute ID="Aspect" />
    <Attribute ID="Illoc" />
    <Attribute ID="Num" />
    <Attribute ID="Per" />
    <Attribute ID="Syn" />
    <Attribute ID="Tense" />
</Category>

<Category Type="APAR">
    <Attribute ID="Aspect"
    <Attribute ID="Syn " />
</Category>

<Category Type="MODD ">
    <Attribute ID="Illoc" />
```

```
    <Attribute ID="Mod" />
    <Attribute ID="Num" />
    <Attribute ID="Per" />
    <Attribute ID="Pol" />
    <Attribute ID="Syn "
    <Attribute ID="Tense" />
</Category>

<Category Type="MODST ">
    <Attribute ID="Illoc" />
    <Attribute ID="Num" />
    <Attribute ID="Per" />
    <Attribute ID="Pol" />
    <Attribute ID="Sta" />
    <Attribute ID="Syn "
    <Attribute ID="Tense" />
</Category>
```

Attributes

```
<Attribute ID="Aspect" obl="*" num="s">
    <Value>?asp</Value>
    <Value Tag="indefinite">ind</Value>
    <Value Tag="imperfective">imp</Value>
    <Value Tag="perfect">pf</Value>
    <Value Tag="progressive">pr</Value>
</Attribute>

<Attribute ID="Concept" obl="*" num="s">
    <Value>[FIND: core › concept › concept |
    CHECK: %\w*]</Value>
</Attribute>

<Attribute ID="Illoc " obl="*" num="1">
    <Value>?illoc</Value>
    <Value Tag="declarative" >dec< /Value>
    <Value Tag="interrogative" >int< /Value>
    <Value Tag="imperative" >imp< /Value>
</Attribute>

<Attribute ID="Modality " obl="*" num="1">
    <Value>?mod </Value>
    <Value>ability›abl</Value>
    <Value>obligation›obl</Value>
```

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

116

```
    <Value> permission>perm</Value>
    <Value> possibility>psbl</Value>
    <Value> volition>vol</Value>
</Attribute>

<Attribute ID="Num" obl="*" num="s">
    <Value>?n</Value>
    <Value>pl</Value>
    <Value>sg</Value>
    <Value>null</Value>
</Attribute>

<Attribute ID="Per" obl="*" num="s">
    <Value>1</Value>
    <Value>2</Value>
    <Value>3</Value>
</Attribute>

<Attribute ID="Recip" obl="*" num="1">
    <Value Tag="grammatical">g</Value>
    <Value Tag="never">n</Value>
    <Value Tag="optional">o</Value>
</Attribute>

<Attribute ID="Reflex" obl="*" num="1">
    <Value Tag="grammatical">g</Value>
    <Value Tag="never">n</Value>
    <Value Tag="optional">o</Value>
</Attribute>

<Attribute ID="Template" obl="+" num="s">
    <Value>?tpl</Value>
    <Value>[FIND: constructicon1 > construc >
    code]</Value>
</Attribute>

<Attribute ID="Tense" obl="*" num="1">
    <Value>?t</Value>
    <Value>past</Value>
    <Value>pres</Value>
</Attribute>

<Attribute ID="Status " obl="*" num="1">
    <Value>?sta</Value>
    <Value>inference>inf</Value>
```

```
    <Value>necessity>nec</Value>
    <Value>possibility>poss</Value>
    <Value>subjunctive>subj</Value>
</Attribute>
```

ONOMÁZEIN 34 (diciembre de 2016): 86 - 117
Francisco Cortés Rodríguez y Ricardo Mairal-Usón
Building an RRG computational grammar

117

7.4. Appendix 4: FunGramKB