FIRST DRAFT

The implementation of the CLS Constructor in ARTEMIS

Carlos Periñán-Pascual

Francisco Arcas-Túnez

Abstract

Most natural language processing researchers highlight the benefits of interlingua-based systems in multilingual settings. In this scenario, Role and Reference Grammar can contribute to build a cross-language semantic representation of the input text in terms of its logical structure. Our goal is to describe the various stages in the development of one of the first computational systems which employs a lexico-conceptual knowledge base to generate the logical structure of sentences. Our approach actually involved some changes in the standard functional model in order to convert the logical structure into an ontology-grounded representation of sentential meaning. In fact, we demonstrate that constructional schemata should become the cornerstone of the syntax-semantics interface in this computerized model of Role and Reference Grammar.

1. Introduction

In computational linguistics, lexical-semantic research is influenced by two mainstreams: syntax-driven and ontology-driven semantics (Nirenburg & Levin 1992). As has been the case in many lexicalist theories, syntax-driven semantics focuses on those meaning components which can predict the syntactic behavior of words. On the other hand, ontology-driven semantics tries to infer text meaning from some language-independent model of a world which is mapped to the lexicon of a given language. Although a single approach is not sufficiently effective, most of the knowledge-based natural language processing (NLP) systems are usually confined to one of these two models.

The goal of this chapter is to describe the design and development of an NLP system whose lexical-semantic model is not only oriented to the syntax of a given language but is also linked to a language-independent ontology-grounded representation of text meaning. As a result, we have implemented ARTEMIS (Automatically Representing TExt Meaning via an Interlingua-based System), a proof-of-concept prototype which is based on Role and Reference Grammar (RRG) as its linguistic model (Van Valin & LaPolla 1997; Van Valin 2005) and which exploits FunGramKB as its knowledge base (Periñán-Pascual & Arcas-Túnez 2004, 2005, 2007, 2008, 2010a, 2010b; Periñán-Pascual & Mairal-Usón 2009, 2010, 2011; Mairal-Usón & Periñán-Pascual 2009). We also intend to demonstrate that

projectionist and constructivist approaches to sentential meaning can be conflated in a computer-tractable model of RRG.

This chapter is organised as follows: sections 2 and 3 sketch out the FunGramKB and RRG models respectively, where in the latter we also introduce our major changes to the standard functional model; and section 4 analyses the different processing stages involved in the CLS Constructor, which outputs a cross-language representation of the input text.

2. FunGramKB

ARTEMIS is a knowledge-based system, since it operates with a repository of knowledge (in our case, lexical, constructional and conceptual knowledge) which is clearly separated from the rest of the system and with an inference engine whose role is to apply relevant knowledge in problem resolution.[1] More particularly, ARTEMIS is provided with FunGramKB, a multipurpose lexico-conceptual knowledge base to be implemented in natural language understanding applications.[2] On the one hand, FunGramKB is multipurpose in the sense that it is both multifunctional and multilingual. Thus, FunGramKB has been designed to be potentially reused

[1] In the case of computational linguistics, one of these problems is word sense disambiguation.

[2] We use the name "FunGramKB Suite" to refer to our knowledge-engineering tool (www.fungramkb.com) and "FunGramKB" to the resulting knowledge base. FunGramKB Suite was developed in C# using the ASP.NET platform and a MySQL database.

in many NLP tasks (e.g. information retrieval and extraction, machine translation, dialogue-based systems, etc) and with many natural languages.[3] On the other hand, our knowledge base comprises three major knowledge levels, consisting of several independent but interrelated modules:

a. Lexical level:

a.1. The Lexicon stores morphosyntactic and collocational information about lexical units. The FunGramKB lexical model is not a literal implementation of the RRG lexicon, although some of the major linguistic assumptions of RRG are still preserved, e.g. the logical structure.

a.2. The Morphicon helps our system to handle cases of inflectional morphology.

b. Grammatical level:

b.1. The Grammaticon stores the constructional schemata which help RRG to construct the syntax-semantics linking algorithm. More particularly, the Grammaticon is composed of several Constructicon modules that are inspired in the four levels of the Lexical Constructional Model (LCM) (Ruiz de Mendoza & Mairal-Usón 2008; Mairal-Usón & Ruiz de Mendoza 2009), i.e. argumental, implicational, illocutionary and discursive.

c. Conceptual level:

c.1. The Ontology is presented as a hierarchical catalogue of the concepts that a person has in mind, so here is where semantic knowledge is stored in

[3] English and Spanish are fully supported in the current version of FunGramKB Suite, although we have just begun to work with other languages, such as German, French, Italian, Bulgarian and Catalan.

the form of meaning postulates. The Ontology consists of a general-purpose module (i.e. Core Ontology) and several domain-specific terminological modules (i.e. Satellite Ontologies).

c.2. The Cognicon stores procedural knowledge by means of scripts, i.e. schemata in which a sequence of stereotypical actions is organised on the basis of temporal continuity, and more particularly on Allen's temporal model (Allen 1983; Allen & Ferguson 1994).

c.3. The Onomasticon stores information about instances of entities and events, such as Bill Gates or 9/11. This module stores two different types of schemata (i.e. snapshots and stories), since instances can be portrayed synchronically or diachronically.

In the FunGramKB architecture, every lexical or grammatical module is language-dependent, whereas every conceptual module is shared by all languages. In other words, linguists must develop one Lexicon, one Morphicon and one Grammaticon for English, one Lexicon, one Morphicon and one Grammaticon for Spanish and so on, but knowledge engineers build just one Ontology, one Cognicon and one Onomasticon to process any language input conceptually. In this scenario, FunGramKB adopts a conceptualist approach, since the Ontology becomes the pivotal module for the whole architecture.

FunGramKB is the product resulting from a knowledge-engineering project, where our major concern has always been its application in linguistically-aware and psychologically-plausible NLP systems. It is for

this reason that the linguistic level in our knowledge base is grounded on the RRG theory, briefly described in the following section.

3. Role and Reference Grammar

3.1 The standard model

RRG is one of the most relevant functional models of language in current linguistics. RRG was not actually designed for NLP, but this linguistic theory presents three characteristics which make it a suitable model for NLP:

a. RRG is a model where morphosyntactic structures and grammatical rules are explained in relation to their semantic and communicative functions.

b. RRG is a monostratal theory, where the syntactic and semantic components are directly connected through a "linking algorithm".

c. RRG is a model which owns typological adequacy.

The features (a-c) are essential for a computational model which aims to provide the capability of natural language understanding. Firstly, a functional view of language allows us to capture syntactic-semantic generalizations which are fundamental to explain the semantic motivation of grammatical phenomena. Secondly, the system is more effectively designed if an algorithm is able to account for both the comprehension and the

production of linguistic expressions. Thirdly, typological adequacy becomes an added value when working in a multilingual environment.

RRG is concerned with two fundamental aspects of language description: the relational structure, which deals with relations between a predicate and its argument(s), and the non-relational structure, which accounts for the hierarchical organization of phrases, clauses and sentences. Consequently, the notions of "logical structure" and "layered structure of the clause" (LSC) are fundamental in the analysis of language.

On the one hand, RRG rejects the standard formats for representing clause structure (e.g. grammatical relations), as can be seen in Figure 1 (Van Valin & LaPolla 1997: 38).[4]

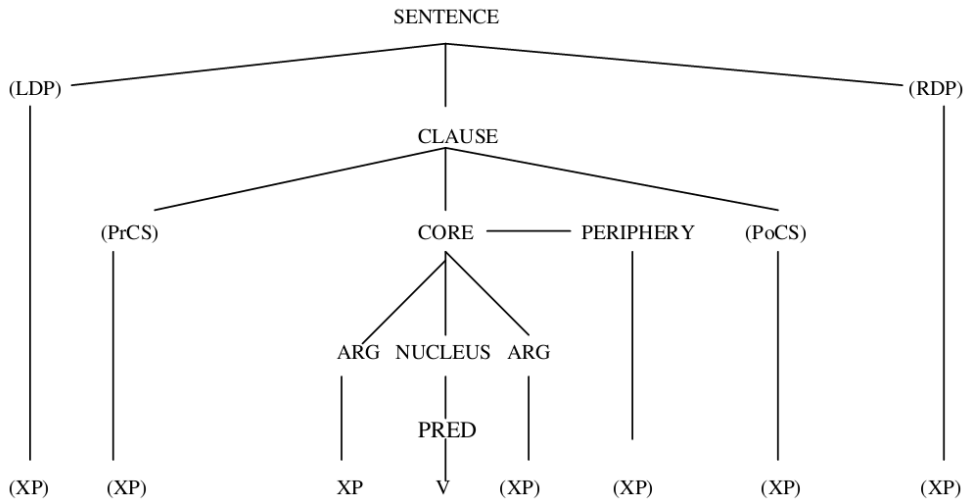

Figure 1. RRG layered structure of the clause.

This hierarchical structure is both semantically and pragmatically motivated,

[4] Abbreviations: LDP 'left-detached position', RDP 'right-detached position', PrCS 'precore slot', PoCS 'postcore slot', ARG 'argument', and PRED 'predicate'.

and not only syntactically based: whereas constituents such as the nucleus, core, periphery and clause are semantically motivated, the detached phrases and the extra-core slots seem to be pragmatically motivated (Van Valin 2005: 8). The LSC is universal; however, cross-linguistic variations are captured by means of the syntactic templates of each language, so syntactic representations are not built on phrase-structure rules.

On the other hand, the semantic representation of a sentence originates from the logical structure assigned to verbs and other predicates in the lexicon on the basis of its distribution in a typology of classes (i.e. state, activity, achievement, semelfactive, accomplishment, active accomplishment, and their corresponding causative forms). To illustrate, (1-3) show that each verb class is represented formally by means of a different logical structure, being composed of elements of a universal semantic metalanguage which consists of constants, variables and semantic operators.[5]

(1) see: **see'** (x,y) [state]

(2) run: **do'** (x, [**run'** (x)]) [activity]

(3) receive: BECOME **have'** (x,y) [accomplishment]

Operators such as aspect, modality, tense or illocutionary force, among many others, are also represented in the logical structure of sentences. Thus, the sentence "Peter broke the glass", for example, is assigned the following

[5] The verb class adscription system is based on Vendler's (1967) Aktionsart distinctions, and the decompositional system is a variant of the one proposed by Dowty (1979).

semantic representation:

(4) $<_{IF}{}^{DEC}$ $<_{TNS}{}^{PAST}$ $<_{ASP}{}^{PERF}$ <[**do'** (Peter, Ø)] CAUSE [BECOME **broken'** (glass)]>>>>

In this model, the syntax-semantics linkage is divided into five steps, as shown in Figure 2:[6]

a. Determine the LSC.

b. Identify the macroroles (i.e. Actor and Undergoer)[7] of the core arguments in terms of the privileged syntactic argument selection hierarchy (Van Valin 2005: 100).

c. Retrieve from the lexicon the logical structure of the predicate in the nucleus of the clause.

d. Assign macroroles to the arguments in the logical structure according to the Actor-Undergoer hierarchy (Van Valin 2005: 126).

e. On the basis of their macroroles, link the arguments determined in step (a) with the arguments found in step (c) until all core arguments are linked.

[6] The linkage algorithm has been hugely simplified and adapted from Van Valin (2005: 149-150).

[7] The macroroles Actor and Undergoer can be regarded as the "logical subject" and the "logical object" respectively.

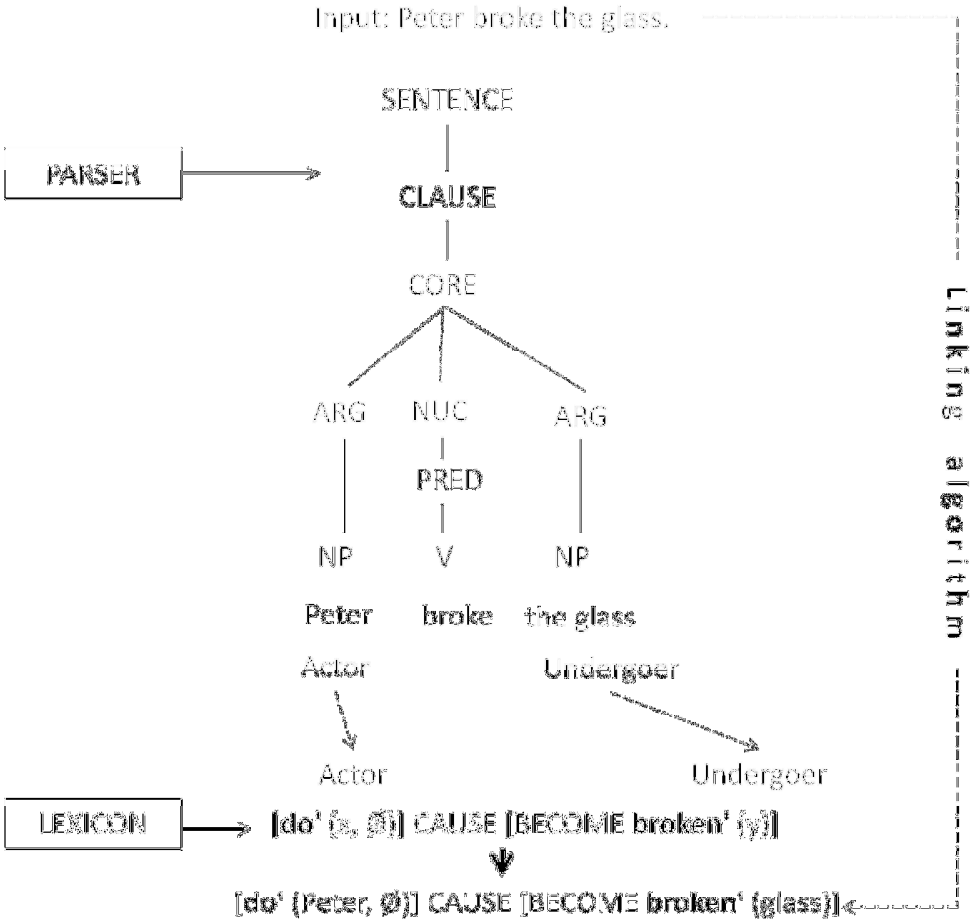

Figure 2. RRG syntax-semantics linkage.[8]

Therefore, the macroroles Actor and Undergoer become a critical component in the syntax-semantics linkage. On the whole, the lexicon is the key module in the RRG framework, since the semantic representation of a sentence is built on the logical structure of the predicate. This is the reason why RRG is viewed as a projectionist model of language.

[8] In the latest version of the RRG model, the label NP is now replaced by RP (Reference Phrase), which, unlike NP, is a non-endocentric construct: "The nucleus of an RP is neither restricted to nominals, nor is it restricted to lexical heads" (Van Valin 2009: 708).

3.2 The computational model

Up to now, there have only been two serious attempts to implement some of the aspects of the RRG theory computationally. For instance, Guest (2009) developed a parser which is able to output the LSC of an English sentence. However, a more challenging research programme can be found in UniArab—Universal Arabic Machine Translator (Salem, Hensman & Nolan 2008), an interlingua-based machine translation prototype which is able to provide a working translation of Modern Standard Arabic to English. One of the primary strengths of UniArab is to build the logical structure of an Arabic sentence, but the project does not manage to provide a robust approach to the semantics of lexical units.

ARTEMIS comes into the scene as one of the first systems which employs a robust knowledge base to generate a full-fledged logical structure to be used in NLP applications requiring language comprehension capabilities. This new approach led us to make some changes to the RRG standard model, since an enhanced representation of the logical structure was required, as explained in the following section.

3.2.1 The Construction category

Undoubtedly, constructional meaning improves the descriptive power of a semantic theory. Van Valin (2005: 3) recognised that “a theory of clause

structure should capture all of the universal features of clauses", so we integrated the construction as a universal category into the LSC. Therefore, the clause is configured now as one or more argumental constructions (L1-CONSTRUCTION)[9] which are recursively arranged, as shown in Figure 3.

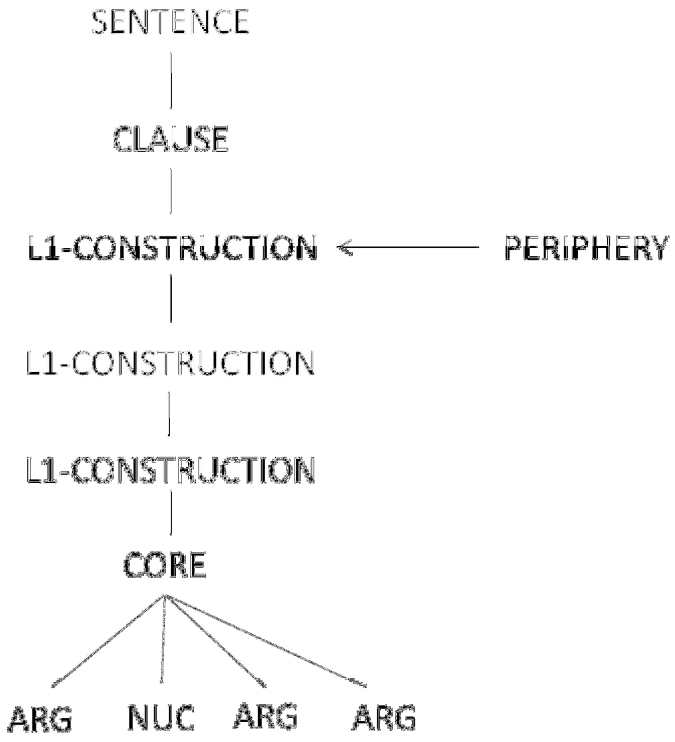

Figure 3. Enhanced model of the LSC.

It is clear that compositionality is one of the most distinctive features of sentential processing, but this is such an overused term that we want to focus on some nuances that will help us to provide a clear definition of "construction". Thus, following Pelletier's categorization (2012),

[9] In fact, this type of construction belongs to the Level 1 of the LCM, i.e. the argumental layer which accounts for the core grammatical properties of lexical items.

FunGramKB adopts a compositional wholist model of computational semantics, which integrates the "functional compositionality"—as defined in (5)—of sentential semantics with the "ontological holism"—as defined in (6)—of constructional semantics.[10]

(5) The μ of a whole is a function of the μ's of its parts and the ways those parts are combined (Pelletier 2012: 153).[11]

(6) Some properties can only be attributed to entities that are not individuals (Pelletier 2012: 156).

In other words, functional compositionality allows a complex whole (e.g. the sentence) to have things (e.g. sentential meaning) which are not present in the parts (e.g. the words), providing that the function introduces this same material every time it is faced with the same parts and manner of combination (e.g. the construction). On the other hand, ontological holism allows a complex whole (e.g. the construction) to have properties (e.g. constructional meaning) which are not properties of any part (e.g. the words). In accordance with these complementary distinctions resulting from the view of compositionality and holism, our definition of "construction" is presented as follows:

(7) A construction is a pairing of form and meaning, serving as a building block in the compositionality of sentential

[10] Unlike the computational meaning with which the word "ontology" is used throughout this chapter, the term "ontological holism" should be understood in its philosophical sense.
[11] μ symbolizes the "meaning function", i.e. X = μ(A), where A is some syntactic item and X is the meaning of A.

semantics, whose meaning cannot be fully derived from the sum of the lexical meanings of the individual constructs taking part in the utterance.[12]

Thus, from the FunGramKB approach, the sentence "John pounded the nail flat into the wall" consists of three argumental constructions: Kernel-2, Transitive Resultative and Caused-Motion.[13]

(8) [[[John pounded the nail]$_{\text{Kernel-2}}$ flat]$_{\text{Transitive-Resultative}}$ into the wall]$_{\text{Caused-Motion}}$

The remaining components can only be perceived as constructs, whose meanings are directly derived from their meaning postulates. In terms of the FunGramKB model, lexical constructs get their meaning from the meaning postulates stored in the Ontology, whereas constructional meaning is shaped by the Core Grammar in the Lexicon and the constructional schemata in the Grammaticon.

The FunGramKB constructional schema, which serves as a machine-tractable representation of the construction, is defined in terms of constraints which license functional compositionality with other constructs or constructions. To illustrate, Figure 4 presents the attribute-value matrix (AVM) of the Caused-Motion Construction.

---

[12] Derivative morphemes are not considered to be linguistic objects in the current version of FunGramKB, so the minimal constructs in the processing of linguistic realizations take the form of lexical units.

[13] Kernel Constructions correspond to basic intransitive (type 1), monotransitive (type 2) and ditransitive (type 3) constructions, where zero-argument verbs raise a Kernel-0 Construction.

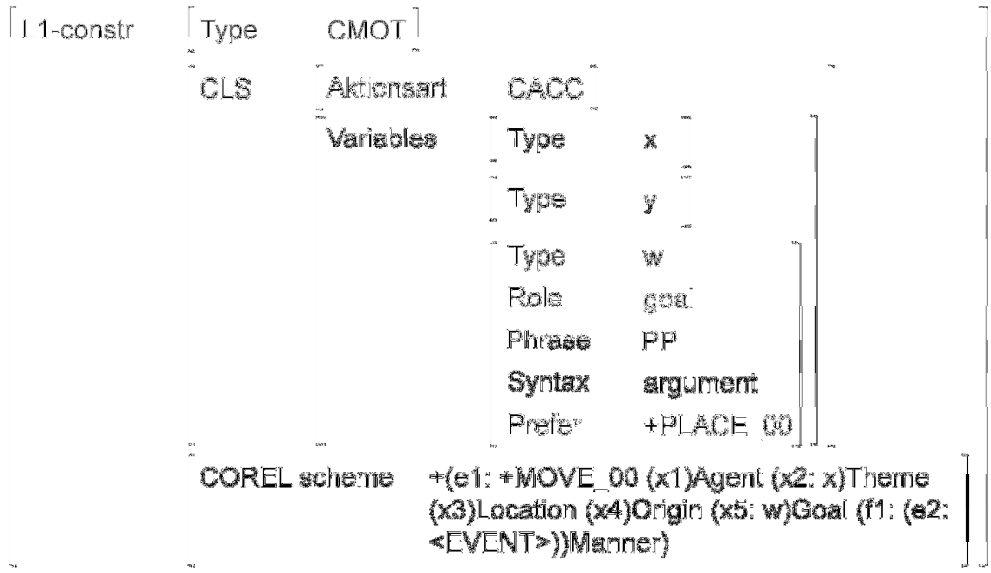

Figure 4. The constructional schema of the Caused-Motion Construction.[14]

The constructional schema contains the properties common to all the instances of a given construction. Therefore, the Grammaticon stores types of constructions to which words in the Lexicon are linked. Up to now, these types of constructions are arranged in a flat organization, instead of relating them in terms of an inheritance hierarchy.[15] The properties which are defined in constructional schemata are rather independent from language so as to determine cross-linguistic generalizations. Indeed, phrase realizations of variables (e.g. NP, PP etc) and the typical prepositions heading prepositional phrases are the only two attributes in the AVM which are language-dependent (Figure 5).

[14] Abbreviations: L1-constr 'L1-constructional schema' and Prefer 'Selectional preference'.
[15] For example, Goldberg (1995) suggested how to capture generalizations across constructions by means of an inheritance hierarchy of constructions, where the lower levels are specializations in form and function of the highest level. Whether this inheritance network is monotonic or non-monotonic is still a debatable issue.

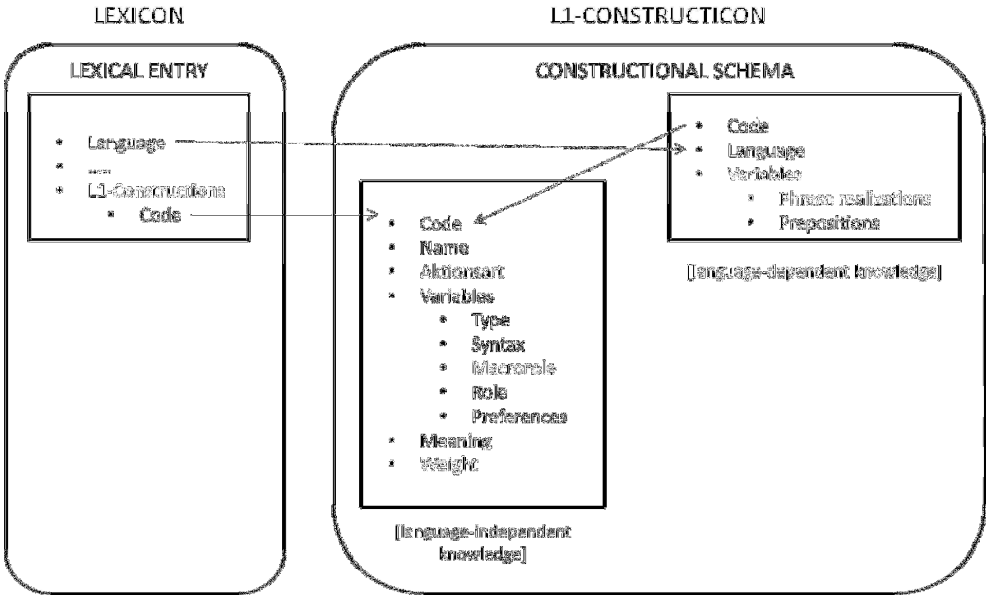

Figure 5. Attributes of the constructional schema.

It is important to bear in mind that the universality of the category construction does not involve the claim that the whole inventory of constructional schemata should be shared by any language. In fact, a given construction can be licensed in a particular language if and only if there is at least one entry in the Lexicon which contains a pointer to that construction.[16] As a result, there was a need to create an L1-Constructicon module, whose interface is shown in Figure 6, for every language in FunGramKB Suite.

[16] As shown in Figure 5, construction-type codes in the lexical entry serve as pointers to constructional schemata.

Figure 6. The L1-Constructicon interface.

Although "(…) there has been a disagreement in the CxG literature about whether or not 'constructions must have meaning'" (Sag 2012: 87), FunGramKB constructions are essentially meaning-bearing devices, where their semantic burden lies in the Aktionsart (i.e. aspectual meaning) and/or the COREL scheme (i.e. conceptual meaning).[17] Indeed, we can infer from

[17] COREL (COnceptual Representation Language) is an interface language to formalize conceptual knowledge in FunGramKB. Periñán-Pascual and Mairal-Usón (2010) described the grammar of this notational language.

the definition (7) that the *raison d'être* of a construction is its semantic contribution to that part of the meaning of the sentence which cannot be derived from the lexical units.

In short, FunGramKB adopts a hybrid approach to constructional meaning, i.e. halfway between projectionism (e.g. Jackendoff 1990; Pustejovsly 1991; Rappaport Hovav & Levin 1998) and constructivism (e.g. Goldberg 1995; Croft 2001). On the one hand, our language model is much closer to projectionism in terms of how linguistic realizations of constructions are related to their semantic descriptions; in fact, FunGramKB shows a clear-cut separation between the linguistic modules, i.e. the Lexicon and the Grammaticon, where the projection from syntax to semantics goes through the pointers in the lexical entries. On the other hand, our language model is much closer to constructivism in terms of how lexical units and constructions jointly affect sentential meaning.

3.2.2 The conceptual logical structure

Another key difference from the standard RRG model is the format of the logical structure, which now becomes a real cross-language representation to be used in multilingual NLP systems with FunGramKB as their knowledge base. As a result, there was a shift of the logical structure into the conceptual logical structure (CLS), which involved a number of changes as illustrated in (9).

(9) Peter broke the glass.

Logical structure:

$<_{IF}^{DEC}$ $<_{TNS}^{PAST}$ $<_{ASP}^{PERF}$ <[**do'** (Peter, Ø)] CAUSE [BECOME **broken'** (glass)]>>>>

CLS:

$<_{IF}^{DEC}$ $<_{TNS}^{PAST}$ $<_{ASP}^{PERF}$ $<_{CONSTR\text{-}L1}^{KER2}$ $<[_{AKT}^{CACC}$ [**+BREAK_00** (**%PETER_00**-Theme, **$GLASS_00**-Referent)]]>>>>

Firstly, the instantiation of variables takes the form not of predicates but of ontological concepts (e.g. the terminal concept $GLASS_00 instead of the predicate *glass*). Secondly, every instantiated concept is assigned a thematic role from the thematic frame of the event to which the verb is linked (e.g. %PETER_00 is the Theme and $GLASS_00 is the Referent in the cognitive situation described by the event +BREAK_00). In contrast to RRG, thematic roles do play a paramount role in the CLS. Indeed, ARTEMIS is only able to perform the lexico-conceptual linkage once the constituents in the parse tree are tagged with the FunGramKB thematic roles. The NLP system can subsequently reach a deeper level of comprehension by deriving the extended COREL scheme (10) from the CLS in (9).

(10) +(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'

We intend to enrich the extended COREL scheme not only with the

knowledge from the meaning postulates in the Ontology but also with that from the scripts in the Cognicon and from the snapshots and stories in the Onomasticon.[18] Thirdly, every argumental construction is embodied in a constructional operator (i.e. CONSTR-L1) whose scope is the core of the clause. Finally, the Aktionsart operator (i.e. AKT) together with an "argument pattern"[19] headed by the event—as shown in (11)—replaces the semantic skeleton originated by the RRG decompositional system.

(11) [event (argument-role, argument-role…)]

These two new operators in the logical structure (i.e. CONSTR-L1 and AKT) play a joint role in shaping sentential meaning, since a given argumental construction not only contributes to the enrichment of the COREL scheme but also helps to determine the Aktionsart. In particular, and due to factors such as the linearity of processing, the concatenation of grammatical constituents and the functional compositionality described in the previous section, the right-most argumental construction which the processor finds in the input is the one which directly impacts on the aspectual value of the verb, as shown in (12-14).[20]

(12) $<_{IF}^{DECL}$ $<_{Tense}^{PAST}$ $<_{CONSTR\text{-}L1}^{KER2}$ $<_{AKT}^{ACT}$ [**$POUND_02** (**%JOHN_00**-Agent, +**NAIL_01**-Goal)]>>>>

[18] For example, a snapshot containing personal knowledge about Peter could reveal that he suffers from haemophilia, adding a potential risk to the state of affairs portrayed by the sentence.

[19] Despite its name, an argument pattern can also introduce a further nucleus in the case of nuclear cosubordination, just as occurs with the Resultative Construction.

[20] It should be noted that the right-most construction in the input is represented by the left-most CONSTR-L1 operator in the bracketed representation of the CLS.

'John pounded the nail'

(13) $<_{IF}^{DECL}$ $<_{Tense}^{PAST}$ $<_{CONSTR\text{-}L1}^{RESU}$ $<_{CONSTR\text{-}L1}^{KER2}$ $<_{AKT}^{CACC}$ [**$POUND_02** (**%JOHN_00**-Agent, **+NAIL_01**-Goal, **+FLAT_00**-Result)]>>>>>

'John pounded the nail flat'

(14) $<_{IF}^{DECL}$ $<_{Tense}^{PAST}$ $<_{CONSTR\text{-}L1}^{CMOT}$ $<_{CONSTR\text{-}L1}^{KER2}$ $<_{AKT}^{CACC}$ [**$POUND_02** (**%JOHN_00**-Agent, **+NAIL_01**-Goal, **+WALL_00**-Goal)]>>>>>

'John pounded the nail into the wall'

As occurs in (12), Kernel Constructions are the only type of constructions which are not formalised in the Constructicon, but are modeled within the lexical entry of the verb. On the other hand, in (13) the L1-Constructicon raises the Transitive-Resultative Construction (RESU) which makes the state of affairs become a causative accomplishment (CACC), whereas in (14) the Caused-Motion Construction (CMOT) similarly brings forward a causative accomplishment. As shown in (13-14), constructional meaning ultimately determines aspectual meaning.[21] The following section describes how the whole CLS is automatically built.

4. The CLS Constructor in ARTEMIS

[21] Periñán-Pascual (2013) describes the interface between the Lexicon and the L1-Constructicon in sentential processing.

To avoid being distracted from implementation details, Figure 7 shows the UML diagram[22] which serves to describe the logical activity which models the behaviour of the CLS Constructor.

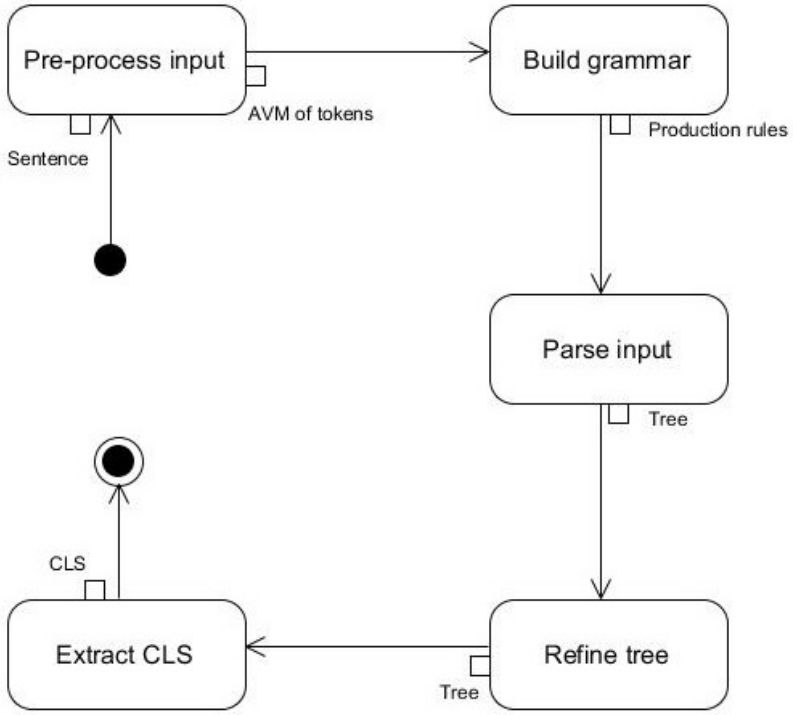

Figure 7. The ARTEMIS process (abridged version).

In the following sections, we describe the main stages involved in the syntax-semantics linking algorithm with FunGramKB. To illustrate, we will use as an example the sentence (15), whose CLS is (16).

(15) John pounded the nail flat into the wall.

(16) $<_{IF}^{DECL}$ $<_{Tense}^{PAST}$ $<_{CONSTR\text{-}L1}^{CMOT}$ $<_{CONSTR\text{-}L1}^{RESU}$ $<_{CONSTR\text{-}}$

[22] UML (Unified Modeling Language) is "a general-purpose visual modeling language that is used to specify, visualize, construct, and document the artifacts of a software system" (Rumbaugh, Jacobson & Booch 1999: 3).

$_{L1}^{KER2}$ <$_{AKT}^{CACC}$ **[$POUND_02** (**%JOHN_00**-Agent, **+NAIL_01**-Goal, **+FLAT_00**-Result, **+WALL_00**-Goal)]>>>>>>

4.1. Pre-processing

First of all, the input is split into sentences, and then into word tokens. In other words, this task consists in segmenting the input into basic units of analysis. In the following task, the word tokens are lemmatised. Finally, every token is labeled with a unique part-of-speech tag. We employed the OpenNLP library (Baldridge, Morton & Bierner 2001) for tokenization and part-of-speech tagging, and the LemmaSharp library (Jursic, Mozetic, Erjavec & Lavrac 2010) for lemmatization. Following the object-oriented paradigm, we represent feature-based structures as AVMs which are computationally implemented in the form of user-defined objects in the programming language C#.[23] For example, this multi-task stage outputs the AVMs shown in Figure 8.

[23] Together with C++ and Java, C# (C-Sharp) is one of the most popular general-purpose object-oriented programming languages in modern computing.

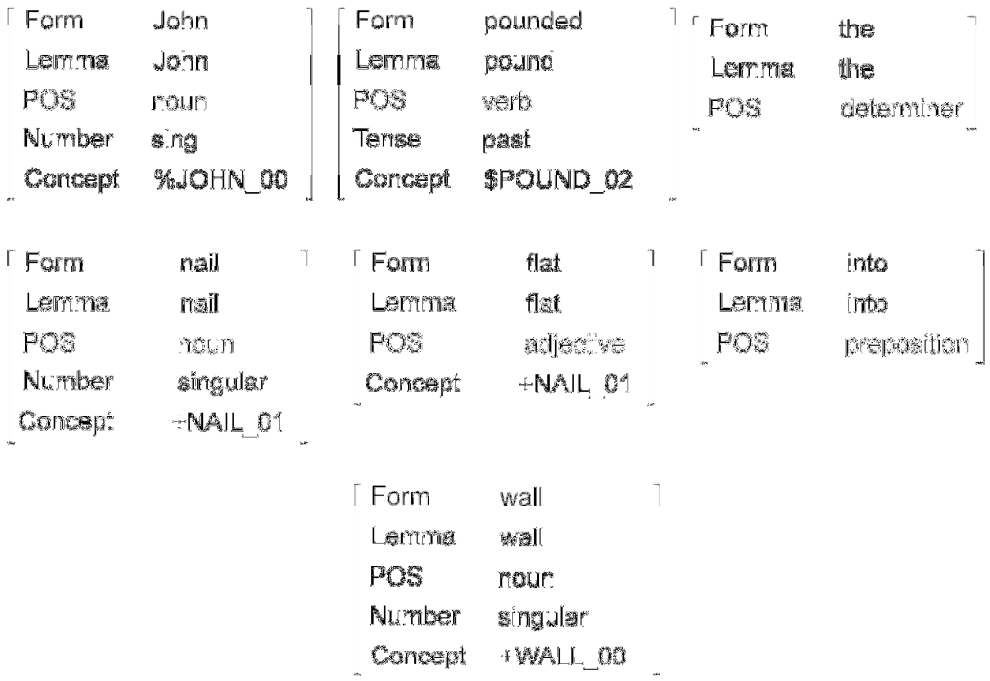

Figure 8. AVMs of word tokens.

It is worth noting that part-of-speech tagging involves predicate conceptualization, bringing in the problem of word-sense disambiguation. Since lexical information in FunGramKB is linked to the senses of words (i.e. sense-oriented approach), a word-sense disambiguator should firstly tag the lemmas with a single conceptual label from the Ontology, or, in the case of proper nouns, from the Onomasticon. This disambiguator is still work in progress, so now users must disambiguate polysemous words from the ARTEMIS interface before the parsing occurs.

4.2. Grammar building

ARTEMIS follows the well-known paradigm of constraint-based grammars, also known as unification grammars, which can encode grammatical knowledge irrespectively of the type of NLP algorithm.[24] The key component of constraint-based grammars can be found in the complex formal descriptions of grammatical units as AVMs, describing features which can be merged through the unification operation. Thus, parsing is not guided just by the sequence of phrase-structure rules but also by the satisfaction of a set of constraints,[25] which are intended to determine structural preference and semantic plausibility, where no single type of constraint is able to resolve any type of local syntactic ambiguity.

Unlike an RRG syntactic analysis such as the one shown in Figure 2, which is based on an inventory of templates, i.e. syntactic trees which do not explicitly state the order of constituents but just their hierarchical organization, ARTEMIS relies on three types of feature-based production rules, i.e. syntactic, constructional and lexical rules. Firstly, syntactic rules are aimed to build the enhanced framework of the LSC (Figure 3). Secondly, constructional rules serve to embed the constructional schemata stored in the L1-Constructicon into the enhanced LSC. When constructional rules are built, some default values in constructional schemata are replaced

[24] Moreover, from the approach of computational linguistics, the time of grammar development with the formalisms of unification grammars is significantly shorter than with phrase-structure grammars (Uszkoreit & Zaenen 1995).

[25] As Cooper (2002: 311) noted, parsing is reduced to a constraint satisfaction problem, i.e. "to ensure syntactic well-formedness of a word sequence it is necessary to ensure that all constraints are simultaneously satisfied by the sequence". For this reason, unification grammars are said to adopt a "constraint-satisfaction model".

by those values in the Core Grammar of the lexical entry of the verb; on the contrary, specific values in constructional schemata override those stated in the lexical entry, so non-monotonic inheritance takes place in the projection operation between the Lexicon and the L1-Constructicon. For instance, Figure 9 shows the Core Grammar of the verb *pound*.[26]

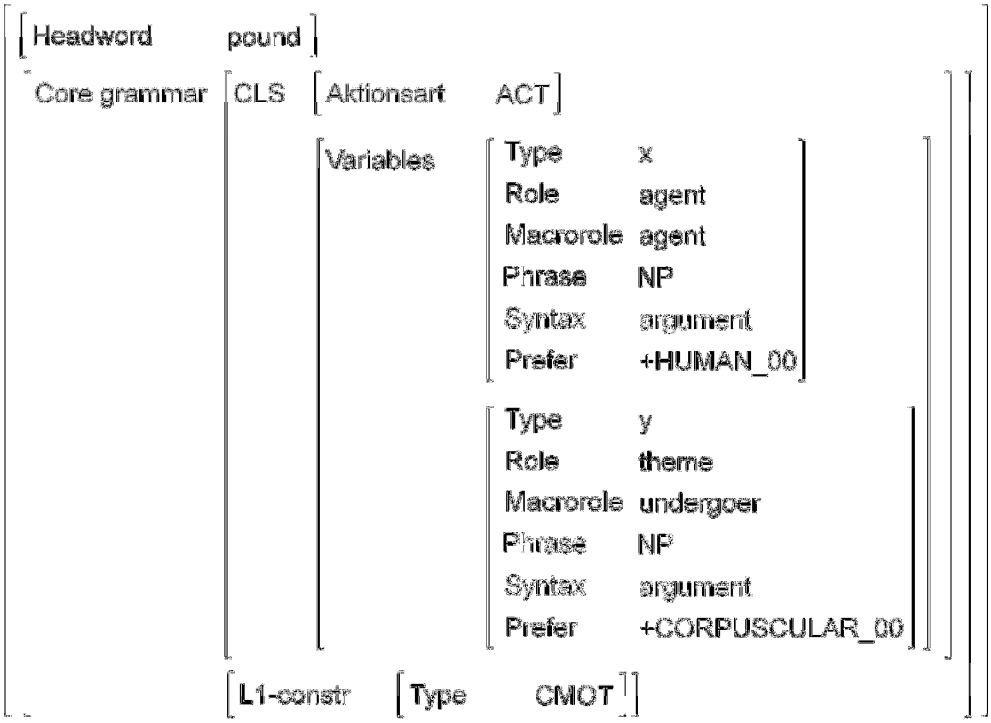

Figure 9. The Core Grammar of *pound*.

In this example, the verb is linked to the constructional schema of the Caused-Motion Construction, which was shown in Figure 4. Thus, the constructional CLS can inherit monotonically the AVMs of the X and Y

[26] It should be noted that, instead of letting lexical entries store the default macrorole, phrase realization and syntax of variable instantiations, these types of knowledge are actually retrieved from the RRG theoretical framework. Mairal-Usón and Periñán-Pascual (2009) presented the anatomy of the FunGramKB Lexicon by describing the different types of features which form part of a predicate's lexical entry.

variables from the lexical entry, whereas the AVM of the W variable is introduced and the Aktionsart value in the lexical entry is overriden. Indeed, this is grounded on one of the basic tenets of constructivist grammars to resolve the conflict between lexical semantics and constructional semantics, i.e. the override principle (Michaelis 2003: 101):

> If lexical and structural meanings conflict, the semantic specifications of the lexical element conform to those of the grammatical structure with which that lexical item is combined.

Thirdly, lexical rules provide the tokens with morphosyntactic and semantic information from the Lexicon and the Ontology respectively. Unlike syntactic rules, which users can pre-define through the Grammar Development Environment, constructional and lexical rules are created dynamically at runtime. In other words, in order to make sentential processing faster and more effective, ARTEMIS will build only those constructional and lexical rules which can be directly derived from the constructional schemata and lexical entries being linked to the predicates in the input stream. Therefore, this stage finally outputs a single text file containing those constraint-based production rules which are required to parse a given input text. In this fashion, the system can gain efficiency by handling only those rules which are potentially relevant, thus minimizing the complexity of processing.

Before we begin to describe the ARTEMIS parser, we should note that Pickering and Van Gompel (2006) remind us that one of the main concerns in the language processing driven by a feature-based unification grammar is that the whole set of constraints which can be involved in the processing should be identified beforehand, as well as the precise way these constraints can affect that processing. As a result, this type of computational model requires a large-scale repository of fine-grained morphosyntactic, semantic and pragmatic knowledge on which NLP algorithms are based. Otherwise, the system would have to deal with a "lexico-constructional knowledge bottleneck". Indeed, this is the risk that comes when you fail to develop properly one of the key components in this type of applications, i.e. the knowledge base.

4.3. Syntactic parsing

Since ARTEMIS is currently a proof-of-concept NLP system, we chose to perform our syntactic analysis with the constraint-based chart parser in the NLTK library (Bird, Klein & Loper 2009: 327-356).[27] More particularly, the parser is based on Earley's algorithm (Earley 1970), which can be described as a bottom-up chart parser with top-down prediction, thus improving the

[27] Since the FunGramKB Suite is developed in ASP.NET and C#, it was necessary to use IronPython to integrate the NLTK Python packages into the .NET framework. However, it is worth noting that Python is a script language, so the parser coding is in plain text files which are compiled at runtime and then interpreted. Therefore, as speed becomes a determining factor in any NLP system, we intend to implement a C# version of this chart parser when applied in a realistic scenario.

efficiency of parsing. Chart parsing uses dynamic programming to parse the text by iteratively adding edges to a chart. Each edge represents a hypothesis about the tree structure for a subsequence of the text. The chart parser incrementally adds new edges to the chart, where the chart rules specify the conditions under which new edges should be added to the chart. Parsing is complete when the chart reaches a stage where none of the chart rules adds any new edges. The psychologically-plausible behaviour of this 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-analyse the input if one parse structure proves incorrect (i.e. no backtracking).

From a psycholinguistic approach, in contrast with the two-stage model of sentential processing,[28] the architecture of the constraint-based parser takes the form of an interactive processor which employs both syntactic and semantic information from the beginning of the analysis. In our case, ARTEMIS is based on an interactive model of language analysis which is built upon a database whose (non-)linguistic knowledge is stored in

[28] One of the best-known instances of the two-stage model is the Garden-Path (Frazier & Fodor 1978; Frazier 1979), in which a processor makes some initial decisions on the basis of strategies defined exclusively in terms of syntactic information and in a second phase semantic information is used to check whether the initial analysis is adequate.

several rather independent modules.[29]

The second task in this stage consists in resolving global syntactic ambiguity, that is, the system should select a winner from among the multiple parse trees which can be generated from a given input sentence. A particularly common source of syntactic ambiguity is found in constructional co-occurrence, or the combination of several constructions within the same clause. Figure 10 illustrates the linguistic phenomenon of constructional co-occurrence, which is driven by the operation of constructional merger, and the impact on the syntax-semantics interface.

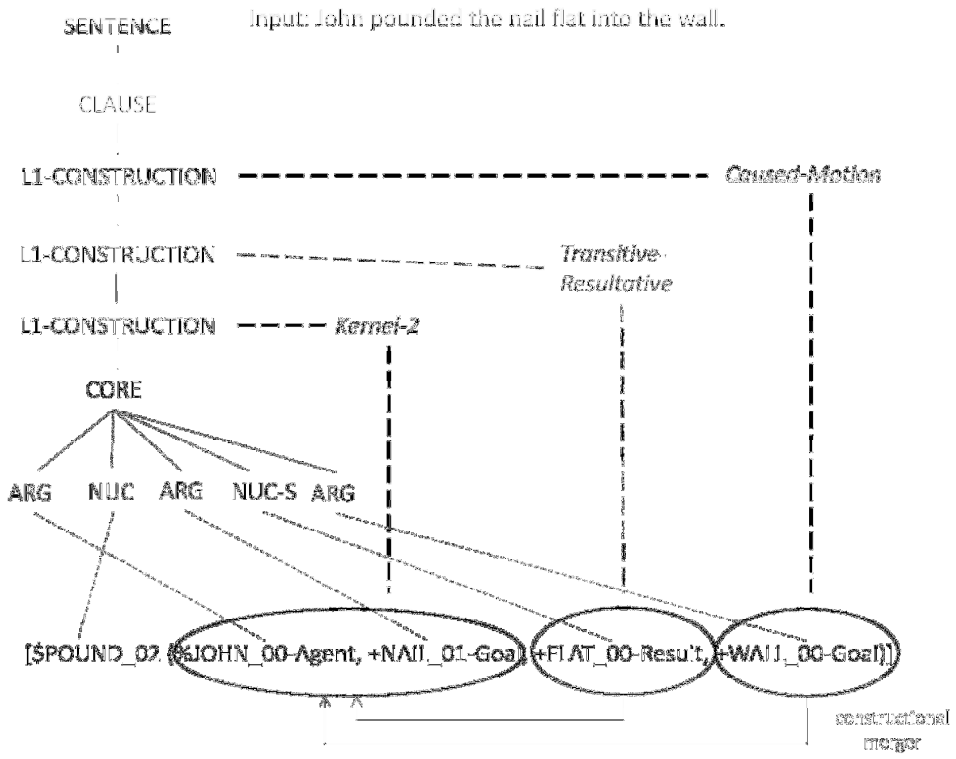

Figure 10. ARTEMIS syntax-semantics linkage.

[29] See section 2.

Each L1-CONSTRUCTION in the LSC (e.g. Kernel-2, Transitive-Resultative and Caused-Motion) typically involves the introduction of at least one argumental slot into the core of the clause (e.g. two, one and one for the previous three constructions respectively), so we could identify in the CLS a distinctive "argumental subpattern" for every instantiation of a given type of construction, where the first argumental subpattern (i.e. the bottom-most L1-CONSTRUCTION in the LSC) serves as the base of the argumental expansion. Consequently, any further L1-construction results from the merger of the new argumental slot(s) with the constructional base. For example, in Figure 10 the argumental pattern consists of three subpatterns, as shown in (17).

(17) Kernel-2, or constructional base: [**$POUND_02** (**%JOHN_00**-Agent, +**NAIL_01**-Goal)]
Resultative: [**$POUND_02** (**%JOHN_00**-Agent, +**NAIL_01**-Goal, +**FLAT_00**-Result)]
Caused-Motion: [**$POUND_02** (**%JOHN_00**-Agent, +**NAIL_01**-Goal, +**WALL_00**-Goal)]

Therefore, from the approach of the syntax-semantics linkage, we can conclude that:

a. the argumental pattern in the CLS is made up of one or more argumental slots, which are instantiated in constructs with the syntactic function of argument (ARG) or subordinated nucleus (NUC-S) in the core of the LSC,

and

b. the semantic constraints determined by the constructional schema assigned to each type of construction in the LSC should also be validated for the corresponding argumental subpattern in the CLS.

Thus, since constructional co-occurrence is a specific instance of functional compositionality, we derive as a corollary the principle of argumental compositionality:

(18) The number of slots of the argumental pattern in the CLS must be equal to the sum of the number of slots in each constructional schema in the LSC.

When dealing with the resolution of the global syntactic ambiguity raised by constructional co-occurrence, we noted that argumental constructions cannot co-occur freely. For example, the Inchoative cannot co-occur with the Middle Construction in the same clause, or the Kernel-2 with the Intransitive Resultative, among many other impossible combinations. Therefore, we devised a weight-based distribution method to restrict constructional co-occurrence in the clause. Firstly, argumental constructions were distributed into groups on the basis of the number of arguments which are involved (i.e. one, two or three) and of the FunGramKB module from where the system retrieves most of the knowledge required to build the constructional rule (i.e. the Lexicon for Kernel Constructions and the L1-Constructicon for the remainder). Secondly, a weight was assigned to each constructional group resulting from the previous step. As shown in Table 1,

the weight of the construction is fixed in proportion to the number of its argumental slots, and the weight is also greater if the constructional schema is derived from the L1-Constructicon rather than from the Lexicon.[30]

Table 1. Weight-based distribution of argumental constructions.

Group	Description	Weight	Example
A	An argument subpattern derived from a one-argument constructional schema in the L1-Constructicon	1	Inchoative Unexpressed object
B	An argument subpattern derived from the one-argument constructional schema in the Lexicon Core Grammar	2	Kernel-1
C	An argument subpattern derived from a two-argument constructional schema in the L1-Constructicon	3	Middle Instrument subject Intransitive resultative Location subject
D	An argument subpattern derived from the two-argument constructional schema in the Lexicon Core Grammar	4	Kernel-2
E	An argument subpattern derived from a three-argument constructional schema in the L1-Constructicon	5	Benefactive Caused motion Dative Transitive resultative
F	An argument subpattern	6	Kernel-3

[30] Although constructional merger needs further in-depth corpus-based research, our initial experiments have led us to define these two criteria as determining factors in the arrangement of constructional groups.

	derived from the three-argument constructional schema in the Lexicon Core Grammar		

Thirdly, we determined the four principles which license constructional merger:

a. Principle of Constructional Anchorage: any L1-CONSTR node above the bottom-most one in the LSC should be argumentally anchored on the latter. In other words, the argumental slots in the constructional base should also be taken into account for the validation of the schemata linked to subsequent constructional types in the clause.

b. Principle of Argumental Expansion: the introduction of an L1-CONSTR node in the LSC typically involves the introduction of at least one argumental slot in the core. More particularly, in the case of those constructions serving as the constructional base, the number of arguments ranges from zero to three; on the other hand, any other further constructional type usually introduces one new argument.

c. Principle of Constructional Base Restriction: the constructional base can only be performed by Kernel Constructions or by one-argument constructional schemata (i.e. A, B, D and F group constructions). Indeed, these are the only types of constructions allowed to take this position in the LSC; otherwise, the compliance with the Principle of Constructional Anchorage would inevitably violate the Principle of Argumental Expansion.

d. Principle of Constructional Unicity: a given type of construction can occur only once in the same clause.

According to the above four principles, the operation of constructional merger can now be defined as the binary relation R^{Merger}, which is logically formalised in (19).

(19) $R^{Merger} \subsetneq \Phi \times \Omega \wedge ((\exists\varphi \in \Phi)(\exists\omega \in \Omega)(\langle\varphi, \omega\rangle \in R^{Merger})) \wedge ((\varphi' R^{Merger} \omega' \wedge \varphi'' R^{Merger} \omega'') \rightarrow (\varphi' \neq \varphi'') \wedge (\omega' = \omega''))$, where $\Phi = \{c, e\}$ and $\Omega = \{a, b, d, f\}$

where the variables [a-f] represent the argumental constructions belonging to the constructional groups [A-F] respectively. In particular, the above principles are applied to guide the mappings between constructional sets, since $\neg((\forall\varphi \in \Phi)(\forall\omega \in \Omega)(\langle\varphi, \omega\rangle \in R^{Merger}))$. For example, $\{\langle c, d\rangle, \langle e, f\rangle\} \notin R^{Merger}$, because two constructions which have the same number of argumental slots cannot co-occur; or for instance, $\langle c, f\rangle \notin R^{Merger}$, because the schema of any further construction added to the constructional base should have a higher number of argumental slots than those in the constructional base. In both examples, the Principles of Constructional Anchorage and Argumental Expansion could not be jointly complied with. In the end, the different possibilities of combining constructions are finally guided by the constructional merger relation, i.e. $R^{Merger} = \{\langle c, a\rangle, \langle c, b\rangle, \langle e,$

a›, ‹e, b›, ‹e, d›},[31] provided that constraints in constructional schemata are satisfied. To illustrate, the constructional merger taking place in Figure 10 is defined as $R^{\text{Merger: John pounded the nail flat into the wall}}$ = {‹Transitive-Resultative, Kernel-2›, ‹Caused-Motion, Kernel-2›}.

However, although many incompatibilities in constructional co-occurrence are detected with the previous weight-based distribution method, global syntactic ambiguity can still persist, so there would be finally more than one parse tree. At this time, each tree would be provided with a weighted value as a result of the addition of all the weights corresponding to the constructions involved in the constructional merger operation.[32] Consequently, the winning parse tree is that which has the highest weighted value.

The idea of creating a weight-based scheme for argumental constructions was motivated primarily by the existence of psycholinguistic evidence showing that the human sentence parser can take into account the frequency of some constructions in order to resolve local ambiguity (cf. Pickering & Van Gompel 2006). In this regard, our current prototype can be improved in two key features. Firstly, it would be more effective to apply the "weight-based priority" from the beginning of the syntactic parsing with the purpose of minimizing global syntactic ambiguity. In fact, in line with

[31] Strictly speaking, constructional merger is an asymmetric relation, since it is always the new argumental slot which is merged with the constructional base, and not the other way around.
[32] See Table 1.

the development of a psychologically-plausible interactive model of the sentence processor, we should not allow constraints to be satisfied in two stages, but we should consistently integrate the various constraint sources from the very beginning. Secondly, the relative frequency of constructions belonging to the same group can also be taken into consideration to resolve syntactic ambiguity, requiring us to perform a corpus-based research to obtain this probabilistic knowledge. In a nutshell, any improvement in the syntactic parser should be aimed at minimizing global syntactic ambiguity by resolving all instances of local structural ambiguity as they arise during the processing.

At the end of this stage, the winning parse tree is stored in an XML file. To illustrate, Appendix 1 shows the XML-formatted parse tree for the sentence "John pounded the nail into the wall", and Appendix 2 presents the XSD schema against which XML-formatted parse trees are validated to check well-formedness.

4.4. Parse tree refinement

This stage is aimed to structure the parse tree *à la RRG*, in such a way that we facilitate the building of the graphical representation of the tree[33] and the CLS. This stage consists of two basic tasks, i.e. relocating tree nodes and

[33] More particularly, the XML file was mapped to the DOT language in GraphViz (www.graphviz.org) to draw the directed acyclic graph.

filtering out node attributes, whose XML-based procedures are actually very similar: both of them have as input the winning XML-formatted parse tree and support the refinement of that tree in accordance with the knowledge stored in other XML files. For example, when relocating tree nodes in the parse tree, (20) provides the system with the following instructions: any argument (ARG) or subordinated nucleus (NUC-S) introduced by L1-constructions (CONSTR-L1) must be moved to the core in the clause, whereas all adjuncts must be placed under the same category of periphery (PER).

(20)
```
<Nodes>
   <Parent Node=”CONSTR-L1”>
      <Child Target=”CORE”>ARG</Child>
      <Child Target=”CORE”>NUC-S</Child>
      <Child Target=”PER”>ADJUNCT</Child>
   </Parent>
</Nodes>
```

For example, when filtering out attributes, (21) is used to identify the relevant attributes of the argument node (ARG). If the node is accompanied by non-relevant attributes, then they will be removed from the parse tree. On the contrary, if a required attribute does not go with the node, then the search will be performed among its subordinated nodes and, in the case of a failed attempt, among its superordinates. If the given attribute is finally not found, then it will be created with a null value.[34]

[34] Since ARTEMIS is still a work-in-progress system, a grammatical feature provided with a null-value attribute should be understood as a grammatical gap in the computational implementation of the RRG theory. When ARTEMIS becomes a full-fledged system to process real-life texts, then null values will be non-existent.

(21) <Node Type="ARG">
<Att>Type</Att>
<Att>Concept</Att>
<Att>Role</Att>
<Att>Macrorole</Att>
</Node>

XPath is the technology used for the search of elements such as nodes and attributes.

4.5. CLS extraction

At the final stage, the CLS results from the extraction of the most relevant semantic units together with their attributes from the XML-formatted refined tree in the previous stage. To accomplish this stage, we designed an XSLT[35] stylesheet which can define the pattern-matching rules for transforming the XML document into a bracketed representation of the CLS, resulting in tasks such as defining the style properties of elements,[36] changing the order in which elements appear in the CLS, and filtering some elements on the basis of a certain property. In other words, the syntax-driven semantics is so embedded in the parse tree itself, certainly much more than in the RRG model, that the system will do nothing but remove the morphosyntactic units of the LSC and relocate the operators according to their scope. For the formal description of the CLS notation, Appendix 3

[35] XSLT stands for "Extensible Stylesheet Language Transformations".
[36] For instance, the FunGramKB concepts are rendered in boldface, and the type and value of operators in sub- and superscript characters respectively.

presents the context-free grammar written in EBNF[37] as well as its graphical representation.

To summarize, Appendix 4 shows a fine-grained activity diagram for the whole process of CLS construction.

4.6. Final remark

At this point, we return to the issue of the linguistic models influencing our system and focus on the so-called "linking problem". In this respect, ARTEMIS adopts a monostratal model of language, but we do not fully agree with RRG that the relationship between semantic and syntactic relations should be conditioned by a hierarchy of semantic roles. It is true that macroroles in the enhanced LSC are identified according to the RRG privileged syntactic argument selection hierarchy, but these macroroles do not play a critical role in the syntax-semantics linkage. In fact, they become irrelevant, but they are still retrieved just in the case that they could play some role in the linguistic generation process.[38] As illustrated in Figure 10, constructional schemata actually become the cornerstone of the syntax-semantics interface, so a constructivist approach is adopted in this regard; indeed, some arguments are directly contributed by the L1-Constructicon.

[37] EBNF (Extended Backus-Naur Form) is a standard notational language used to define context-free grammars formally.

[38] Should this not be the case, macroroles would certainly end up disappearing in our computational model.

On the other hand, lexical entries should be provided with pointers to those types of constructions in which a given verb can occur, so a functional projectionist approach is adopted in this regard.

This position is much more in line with the LCM, which intends to build a bridge between constructivist and projectionist views of language comprehension. In this model of meaning construction, the notions of lexical template (i.e. low-level representation of the semantic and syntactic properties of a predicate) and constructional template (i.e. high-level representation of the semantic properties of a construction) are essential to explain the syntax-semantics interface. More particularly, the semantic interpretation results from a process of lexico-constructional subsumption, i.e. the unification between a lexical template and a constructional template. In ARTEMIS, the elaboration of constructional rules also involves a lexico-constructional subsumption. However, in contrast with the LCM, the lexico-constructional subsumption in ARTEMIS is not a projection process which is regulated by internal constraints—e.g. variable suppression or lexical blocking, among many others (cf. Mairal-Usón & Ruiz de Mendoza 2008), but it should be simply understood as a process of constructional activation within the framework of non-monotonic inheritance: that is, when constructional schemata are activated, in the case of attributes shared by the AVM of the constructional schema and that of the Core Grammar of the verb, specific values in the former override default values in the latter. Therefore, we prefer not to use the term "projection rules" but

"constructional activation rules", where only some values are projected from the Lexicon to the Constructicon.

5. Conclusions

In this chapter, we have described ARTEMIS, a proof-of-concept NLP system which exploits FunGramKB as its knowledge base within the RRG framework to model the semantic representation of the input text in terms of a CLS. Although the RRG logical structure has been formally simplified for the sake of computational efficiency, we have finally achieved a cross-language semantically-enhanced representation by replacing predicates by ontological units and introducing the constructional operator, among some other changes. In fact, we regard constructions as meaning-bearing devices which play a paramount role in the LSC. Although we have focused this chapter on argumental constructions, we intend to progressively incorporate other types of constructional meanings into the CLS, such as implicational, illocutionary and discursive. Moreover, since ARTEMIS retrieves lexico-conceptual knowledge from FunGramKB, it will also be possible to exploit its reasoning engine to reach a deeper level of comprehension through the COREL scheme derived from the CLS. To conclude, we agree with Guest (2009) that RRG is a promising theory for extracting sentential meaning from a computational viewpoint, making this functional model a better

alternative to Head-Phrase Structure Grammar and Dependency Grammar.

6. Acknowledgments

Financial support for this research has been provided by the DGI, Spanish Ministry of Education and Science, grants FFI2011-29798-C02-01, FFI2010-17610 and FFI2010-15983. We would like to thank Christopher Butler for detailed comments on the first draft. Any error is ours.

References

Allen, James F. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM* 26(11): 832-843.

Allen, James F. & Ferguson, George. 1994. Actions and events in interval temporal logic. *Journal of Logic and Computation* 4(5): 531-579.

Baldridge, Jason, Morton, Thomas & Bierner, Gann. 2001. The OpenNLP Toolkit. <http://sharpnlp.codeplex.com> (21 September 2012).

Bird, Steven, Klein, Ewan & Loper, Edward. 2009. *Natural Language Processing with Python*. Sebastopol (California): O'Reilly.

Cooper, Richard P. 2002. *Modelling High-Level Cognitive Processes*. Mahwah (New Jersey): Lawrence Erlbaum Associates.

Croft, William. 2001. *Radical Construction Grammar*. Oxford: Oxford University Press.

Dowty, David. 1979. *Word Meaning and Montague Grammar*. Dordrecht: Reidel.

Earley, Jay. 1970. An efficient context-free parsing algorithm. *Communications of the ACM* 13(2): 94-102.

Frazier, Lyn. 1979. On Comprehending Sentences: Syntactic Parsing Strategies. PhD dissertation, University of Connecticut.

Frazier, Lyn & Fodor, Janet Dean. 1978. The sausage machine: a new two-stage parsing model. *Cognition* 6: 1-34.

Goldberg, Adele E. 1995. *Constructions: A Construction Grammar Approach to Argument Structure*. Chicago: University of Chicago Press.

Guest, Elizabeth. 2009. Parsing using the Role and Reference Grammar paradigm. <http://repository-intralibrary.leedsmet.ac.uk/open_virtual_file_path/i42n47491t/Parsing Using the Role and Reference Grammar Paradigm.pdf> (21 September 2012).

Jackendoff, Ray S. 1990. *Semantic Structures*. Cambridge (Mass.): MIT Press.

Jursic, Matjaz, Mozetic, Igor, Erjavec, Tomaz & Lavrac, Nada. 2010. LemmaGen: multilingual lemmatisation with induced ripple-down rules. *Journal of Universal Computer Science* 16(9): 1190-1214.

Mairal-Usón, Ricardo & Periñán-Pascual, Carlos. 2009. The anatomy of the lexicon component within the framework of a conceptual knowledge base. *Revista Española de Lingüística Aplicada* 22: 217-244.

Mairal-Usón, Ricardo & Ruiz de Mendoza, Francisco José. 2008. Internal and external constraints in meaning construction: the lexicon grammar continuum. In *Estudios de Filología Inglesa: Homenaje a la Dra. Asunción Alba Pelayo*, Laura Alba Juez & María Teresa Gibert Maceda (eds), 219-237. Madrid: Universidad Nacional de Educación a Distancia.

Mairal-Usón, Ricardo & Ruiz de Mendoza, Francisco José. 2009. Levels of description and explanation in meaning construction. In *Deconstructing Constructions*, Christopher Butler & Javier Martín Arista (eds), 153-198. Amsterdam-Philadelphia: John Benjamins.

Michaelis, Laura A. 2003. Word meaning, sentence meaning, and syntactic meaning. In *Cognitive Approaches to Lexical Semantics*, Hubert Cuykens, René Dirven & John R. Taylor (eds.), 93-122. Berlin-New York: Mouton de Gruyter.

Nirenburg, Sergei & Levin, Lori. 1992. Syntax-driven and ontology-driven lexical semantics. In *Lexical Semantics and Knowledge Representation: First SIGLEX Workshop,* James Pustejovsky & Sabine Bergler (eds.), 5-20. Berlin-Heidelberg: Springer.

Pelletier, Francis Jeffry. 2012. Holism and compositionality. In *The Oxford Handbook of Compositionality*, Markus Werning, Wolfram Hinzen

& Edouard Machery (eds.), 149-174. Oxford: Oxford University Press.

Periñán-Pascual, Carlos. In press. "Towards a model of constructional meaning for natural language understanding". In *Linking Constructions into Functional Linguistics: The Role of Constructions in RRG Grammars*, Brian Nolan & Elke Diedrichsen (eds.). Amsterdam: John Benjamins.

Periñán-Pascual, Carlos & Arcas-Túnez, Francisco. 2004. Meaning postulates in a lexico-conceptual knowledge base. In *Proceedings of the 15th International Workshop on Databases and Expert Systems Applications*, 38-42. Los Alamitos: IEEE Computer Society.

Periñán-Pascual, Carlos & Arcas-Túnez, Francisco. 2005. Microconceptual-Knowledge Spreading in FunGramKB. In *Proceedings of the 9th IASTED International Conference on Artificial Intelligence and Soft Computing*, 239-244. Anaheim-Calgary-Zurich: ACTA Press.

Periñán-Pascual, Carlos & Arcas-Túnez, Francisco. 2007. Cognitive modules of an NLP knowledge base for language understanding. *Procesamiento del Lenguaje Natural* 39: 197-204.

Periñán-Pascual, Carlos & Arcas-Túnez, Francisco. 2008. A cognitive approach to qualities for NLP. *Procesamiento del Lenguaje Natural* 41: 137-144.

Periñán-Pascual, Carlos & Arcas-Túnez, Francisco. 2010a. Ontological commitments in FunGramKB. *Procesamiento del Lenguaje Natural*

44: 27-34.

Periñán-Pascual, Carlos & Arcas-Túnez, Francisco. 2010b. The architecture of FunGramKB. In *Proceedings of the 7th International Conference on Language Resources and Evaluation*, 2667-2674. Malta: European Language Resources Association.

Periñán-Pascual, Carlos & Mairal-Usón, Ricardo. 2009. Bringing Role and Reference Grammar to natural language understanding. *Procesamiento del Lenguaje Natural* 43: 265-273.

Periñán-Pascual, Carlos & Mairal-Usón, Ricardo. 2010. La gramática de COREL: un lenguaje de representación conceptual. *Onomázein* 21: 11-45.

Periñán-Pascual, Carlos & Mairal-Usón, Ricardo. 2011. The COHERENT methodology in FunGramKB. *Onomázein* 24: 13-33.

Pickering, Martin J. & Van Gompel, Roger P.G. 2006. Syntactic parsing. In *Handbook of Psycholinguistics*, Matthew J. Traxler & Morton A. Gernsbacher (eds.), 455-503. Amsterdam: Elsevier.

Pustejovsky, James. 1991. The Generative Lexicon. *Computational Linguistics* 17(4): 409-441.

Rappaport Hovav, Malka & Levin, Beth. 1998. Building verb meanings. In *The Projection of Arguments: Lexical and Compositional Factors*, Miriam Butt & Wilhelm Geuder (eds.), 97-134. Stanford: CSLI Publications.

Ruiz de Mendoza, Francisco José & Mairal-Usón, Ricardo. 2008. Levels of

description and constraining factors in meaning construction: an introduction to the Lexical Constructional Model. *Folia Linguistica* 42(2): 355-400.

Rumbaugh, James, Jacobson, Ivar & Booch, Grady. 1999. *The Unified Modeling Language Reference Manual*. Reading (Mass.): Addison-Wesley.

Sag, Ivan A. 2012. Sign-based Construction Grammar: an informal synopsis. In *Sign-based Construction Grammar*, Hans C. Boas & Ivan Sag (eds.), 69-202. Stanford: CSLI Publications.

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

Uszkoreit, Hans & Zaenen, Annie. 1995. Grammar formalisms. In *Survey of the State of the Art in Human Language Technology*, Giovanni Varile & Antonio Zampolli (eds.), 100-101. Cambridge: Cambridge University Press.

Van Valin, Robert D. Jr. 2005. *Exploring the Syntax-Semantics Interface*. Cambridge: Cambridge University Press.

Van Valin, Robert D. Jr. 2009. Role and Reference Grammar as a framework for linguistic analysis. In *The Oxford Handbook of Linguistic Analysis*, Bernd Heine & Heiko Narrog (eds.), 703-738. Oxford: Oxford University Press.

Van Valin, Robert D. Jr. & LaPolla, Randy J. 1997. *Syntax, Structure, Meaning and Function*. Cambridge: Cambridge University Press.

Vendler, Zeno. 1967. *Linguistics in Philosophy*. Ithaca: Cornell University Press.